UNIVERSIDAD NACIONAL AUTÓNOMA

DE MÉXICO

FACULTAD DE INGENIERÍA

CONSTRUCCIÓN ELECTROMECÁNICA DE UN CUADRÚPEDO UNIDIRECCIONAL.

T E S I S P R O F E S I O N A L

QUE PARA OBTENER EL TÍTULO DE:

I N G E N I E R O MECÁNICO ELECTRICISTA

ÁREA: ELÉCTRICA-ELECTRÓNICA

P R E S E N T A

RICHAUD BARRERA VICTOR MANUEL

DIRECTOR: DR. JAIME BALTAZAR MORALES SANDOVAL

MEXICO, D. F., CIUDAD UNIVERSITARIA, 2006

Gracias.

Papá: por tu paciencia y esperanza que siempre haz tenido en mi.

Mamá: por estar siempre conmigo siguiéndome mis locuras y ya ves “ACABAMOS”.

Abuelito Pepe: por aquel barco con motor de jabón y el hombre araña, que me llevaron poco a poco, a querer construir sueños y que ¡SI SIRVE SOÑAR!

Chack: por creer en mí, aunque casi siempre nos agarrábamos del chongo, pero lo reconozco, siempre fuiste la mayor para mí.

Yessi: por parar el tiempo y siempre hacerme sentir un veinteañero.

Abuelita Mary: porque, se que siempre estoy en tus rezos y estuviste conmigo en momentos difíciles. “Que la fuerza te acompañe”.

Muñequita (Nayeli): por estar en todo momento, en cuerpo y alma, dejando el pasado atrás, viendo solamente esa pasión y amor que nos tenemos.

Tío Miguel: por cubrir toda mi vida con cuidados y atenciones; por enseñarme que, “LOS FAVORES SE HACEN CUANDO SE PUEDEN, PERO SIEMPRE CON LA MEJOR INTENCIÓN”.

LABCOM :
Al Ingeniero José Luis Antón Macín y al Licenciado Francisco de Hoyos: por su confianza.

Al Ingeniero José Luis Hernández Ramírez: por prestarme la Biblioteca de Alejandría y ser compartido conmigo, lo cual siempre te estaré agradecido.

AUDIOVISUALES F.I.:
A Erick, Agustín y Ángel: por ser mis amigos y estar pendientes de mí. ¡ME AYUDARON MUCHÍSIMO VIEJOS¡

Maya (Alberto Fuentes): por tenerme paciencia y enseñarme que debo disfrutar la vida, sin importar como este la situación. ¡HAY QUE VER TODO BLUE!

A mi director el Dr. Jaime Baltazar: por mostrarme que un sueño se puede volver tesis.

Al Ingeniero Alejandro Figueroa Páez: que cuando sentí que no avanzaba en las asignaturas de la carrera, externó que “DE POQUITO EN POQUITO SE VA LLENANDO EL CARRIT0”.

Al Ingeniero Salvador Zamora: por su interés en mis proyectos y aportar soluciones a estos.

A mis 3 alegres compadres (Andrei, Paco, Alex) y a mis 3 alegres comadres (Tamara, Solange, Irma): por ese lazo tan grande de hermandad y ejemplo, de que ni la distancia ni el tiempo, nos han podido separar. “BIEN HECHO CHICOS”.

A los Walker Boys (Rodrigo, Marcos, José Luis, Erick y Agust) y a mi Walker Girl (Nayeli): que me ayudaron en el momento de la construcción del robot; así como, al Walker Chief (Alberto Fuentes Maya): por dejarme armar al robot, aprendiendo de las fallas que me sacaron canas verdes y que sabías que iban a suceder ¿VERDAD?

A Juanjo: que me recomendó estudiar esta carrera y apoyó hasta donde le fue posible, así como, por manifestar su preocupación de mi avance en la carrera.

Edgar Alberto: por ser ejemplo de que el rebelde sin causa no es eterno y estoy orgulloso de lo que haz hecho.

A Juan Gracida: por sacarme un rato del universo “STAR WARS” y enseñarme el mundo de la electrónica y las alarmas, de manera empírica, así como, por los consejos para estar contento conmigo. ¡ALA PRIMO¡ ¿TE CHILLÓ, VERDAD?

A Charly: que aunque siempre te pensé como mi némesis, aprendí muchas cosas buenas de ti y te estoy agradecido de que siempre me ves como tu amigo, así como, me integraste en tu circulo de amigos (que es muy selecto), y si no recordemos a Edgar o al guaya-guaya. OYE CHARLY. ¿Y SABES?

A Rafael Gracida: por sus ocurrencias y siempre llevarle la contraria al Charly, mostrando que: “ANTE TODO, VE SU REALIDAD”.

A Héctor Castillo y a Chucho (Maniwis): por ser mis amigos. SIEMPRE LOS VOY A RECORDAR.

A Ricardo (Aretes), Cristian (Arny), Juan Carlos (Sapito) y a Mary Carmen: quienes me enseñaron a tener gente a mi cargo, así como, su invaluable amistad y dedicación.

A los chicos del PATO (Arturo, Uriel, Raúl, Gustavo, Adrián y Gerardo): que me integraron a su grupo de inmediato, considerándome su amigo, y si Batman “NO ESTAS INCLUIDO”

A José Pablo González Chávez (Batman) y Octavio Flores Orozco (Amigo Octa), vecinitos de laboratorio: por ser mis amigos y aguantar todas mis bromas.

A Julio Verne y a George Lucas: que con sus visiones me enseñaron a plasmar una de mis tantas ideas, que en algún momento pensé como “UN JUEGO DE NIÑOS”.

A MIS SINODALES:

Ing. Arturo Zapata y Rosales, M.C. Reynaldo Alanis Cantu y M.A. Nelly Gayosso: por apoyarme en esta tesis con sus opiniones y recomendaciones.

Quiero agradecerte que estuviste conmigo durante ciertas etapas de mi vida y que me diste ánimos para salir adelante sin importar cual fuese la situación y siempre con buena vibra.

SOLO UNOS CUANTOS AMIGOS, PERO DE CORAZÓN, GRACIAS.

Flavio López Vallejo.
Iván Santamaría.
Juanita Covarrubias.
Rosalía Guerrero.
Ricardo Peña Perez.
Juan Manuel Solana Jiménez.
Jorge García Solís.
Arturo Menchaca Lobato.
Sara Ivonne Franco.
Zorayda Jaime.
Teresa Araiza.
Susana Sánchez.
Yazmin Gómez León.
Ricardo Chagoya Morales.
Nadia Mónica López Garcés.
Alma Rebecca (Cuauhtemoc).
Juan Carlos Sabido Alcántara.
Oscar “Güero” Franco.
Vanesa Saldaña Cabrera.
Karina Malvaez Buenrostro.
Gustavo Santos Gutierrez.
Gabriela Ávila Zarazua.
Sandra (Amiga de Gaby Zarazua).
Cinthia Adriana Redonda Godoy.
Juan Santoyo A. (Prau-prau).
Leonardo Fabio Garay Medina.
Héctor Medrano Arellano.
Zayra (Esposa de Carlos).
Salvador Alvarado Monroy.
Sandy Madrid Lee.
Adrián Pacheco.
Erica Domínguez .
Myriam Domínguez.
América Zarrabal.
Maestro José Luis (IRMEXCO).
Gerardo Velásquez Carmona.
Uriel Rendón.
Arturo Mayorga.
Gustavo Cruz White.
José Pablo (Batman).
El Amigo Octa.
Patricia Iniestra.
Lucero Yuridia Jiménez Diaz.
Susana y Gris (Amigas de SKAR Syst.).
Mary Carmen Chávez Abrego.
Ariadna Álvarez Lara.
Domingo Eduardo Rodríguez Almaraz.
Myrna Morones Mendoza y su Novio.
Paty Hong Cirion.
Paco y Marta (Amigos de Paty Hong).
Adriana Barrera Pareyón.
Maria Dolores Rebollo Tello.
Heron Jaramillo Tello.
Enrique Estrada Castillo.
Fernando Gómez de Paz.
Heron Velasco y Arce.
Esperanza Rodríguez Carpinteyro.
Candy Sandy te Ama.
Moisés y Peter Child.
Edgar Alberto Kapamas Luna.
Leticia Gaytan Hernández Magro.
Carmelita Gaytan Hernández Magro.
Carmelita Hernández Magro.
Ernesto Zuno.
Patricia Becerril.
Evelia (Ingenieria).
Juan Almeida Estrada.
Sofía y su Papa. (Vecinos).
José Luis Sandoval Barrera.
Teresita Sandoval Barrera.
Lucemi Herrera.
Maria Richaud Herrera.
Cristina Richaud Herrera.
Humberto Richaud Colorado.
Bety Autran Richaud.
Jennifer Richaud Torres.
Erick Pedro Villarejo González.
Agustín González Morales.
Ángel Daniel Mateos Alonso.
Leticia Reyes Licón.
Rubén Rodríguez Tapia.
Vanessa Schmoller.
Paul Fettes.
José David Guzmán Arévalo.
Gabriela Jiménez Meneses.
Luis Vidal Díaz González.
Yo (Yolanda).
David Lugo.
Silvia Rosales Sánchez.
Raúl Armando Rodríguez Sánchez.
Román Christian Pérez Gil.
Agustin Raya.
José Luis Medellín.
Edgar (James Bond).
Ing. Javier Calderón.
Carlos “El niño”.
Faridhy (Mamá e hija).
Martha Valdivieso.
Todos los Estudiantes de Petrolera que me visitaron por casi 3 años.

POR MÍ

Y

POR TODOS

MIS

COMPAÑEROS.

Tabla de Contenidos

Tabla de Contenidos

INTRODUCCIÓN

"Tal vez caminar no sea tan rápido,
pero así se ha llegado a lugares
donde las ruedas no lo han hecho".

Dr. Herbert Hörtlehner

Antes de ingresar a la carrera de Ingeniería Electrónica, la inquietud de fabricar robots me llevó a investigar mecanismos, los cuales pude observar en juguetes, y así, conocer más sobre engranes, que son de gran utilidad en las extremidades de cualquier robot. También había un dispositivo, que en ese tiempo llamé el "*cerebro*", y que en ese momento no podía entender. Conforme avance los semestres de la carrera, estuve en asignaturas que comenzaron a generar ideas sobre el control electrónico, conociendo cada vez más al "*cerebro*", por medio de circuitos electrónicos que utilizaban interruptores, diodos emisores de luz y que en algunos casos controlaban motores.

A lo largo del desarrollo de esta tesis, se recurrió a las asignaturas de la carrera que sirvieron de plataforma para la propuesta final: ***El aportar un diseño y construcción de un robot cuadrúpedo, controlando su movimiento en una forma unidireccional.***

En el capítulo I se muestra el significado de la palabra robot, así como, la definición actual de estos, su trascendencia en la historia, que se remonta desde antes de Cristo, hasta los robots que se han fabricado en los últimos años, a los que nombramos "robots del mundo real". Se abordan a los robots cuadrúpedos, mostrando sus diferentes aplicaciones y algunos ejemplos de fabricación de estos en el mundo, concluyendo con los principales retos que tienen los fabricantes de robots cuadrúpedos, que nos da una panorámica del futuro de estos.

En el capítulo II se plantean de manera general las etapas que constituyen al diseño preliminar del robot cuadrúpedo unidireccional, comenzando con la electrónica y el concepto de cada dispositivo electrónico a usarse como la alimentación regulada que energiza a los circuitos electrónicos, el mando remoto que controla al robot a distancia, el microcontrolador que funciona como cerebro en la secuencia de movimiento y el Puente H que se encarga de vincular el pulso que emite el microcontrolador y la potencia hacia cada motor para que estos puedan funcionar. Después de la electrónica se muestran las partes que comprenden la mecánica, el concepto de engranes y algunas de sus características, así como, el concepto de extremidades. Este capítulo concluye con la propuesta de solución, donde se plantea el curso que va a tomar la construcción del robot cuadrúpedo unidireccional.

En el capítulo III se comienza la implementación del sistema prototipo, por medio del modelo Torrebot que clasifica los dispositivos electrónicos y mecánicos, aprovechando esta misma para mostrar los dispositivos que se probaron antes de llegar a la propuesta de solución, mostrando sus características más importantes.

En esta tesis se utilizan 3 niveles del modelo Torrebot que son: nivel físico (estructura física, unidades motoras y las etapas de potencia), nivel de reacción (formado por el conjunto de sensores y los sistemas básicos para su manejo), y nivel de control (incluye los circuitos más básicos que relacionan las salidas de los sensores con las restantes unidades). Cabe mencionar que como el microcontrolador 16F84A se clasifica en el nivel de control, se abordan de manera general, las herramientas necesarias para su programación (MPLAB) y grabación (IC-Prog).

En el capítulo IV se presentan los componentes del robot cuadrúpedo unidireccional en un diagrama de bloques, que muestra los módulos componentes que son: la fuente de alimentación, el módulo Transmisor/Receptor Remoto, el módulo de control (microcontrolador y periféricos básicos), el módulo electrónico (Puente H y motores), y el módulo mecánico (extremidades y engranajes).

En cada módulo se muestran las características de cada uno de los dispositivos que lo conforman, así como sus diagramas esquemáticos.

Adicionalmente en el módulo de control se plantea un diagrama de flujo, que determina el ciclo Inicio/Paro ensamblado y grabado en el microcontrolador con una velocidad determinada mediante la técnica PWM y en los módulos mecánicos se muestran los motores y la integración de los engranes a las extremidades, la cual es descrita pieza por pieza permitiendo el movimiento de las mismas.

El capítulo V expone la construcción de cada uno de los módulos (electrónicos y mecánicos) y las pruebas por separado, realizadas previamente a su integración y prueba integral del diseño final.

CAPÍTULO I

1 ROBOTS

La palabra "*robot"* proviene del vocablo Checo "robota" que significa servidumbre o trabajo forzado, haciendo especial referencia a los llamados "trabajadores alquilados" que vivieron en el Imperio Austro-Húngaro hasta 1848. Aunque la palabra *robot* no fue utilizada por primera vez sino hasta 1923 por Karel Čapek en su obra de teatro "R.U.R.".[1]

1.1 ¿QUÉ ES UN ROBOT?

Actualmente, un **robot** se puede definir como una entidad hecha por el hombre con un cuerpo y una conexión de retroalimentación inteligente entre el sentido y la acción –no bajo la acción directa del control humano-, donde usualmente, la inteligencia es una computadora o un microcontrolador ejecutando un programa. Sin embargo, cabe destacar se ha avanzado mucho en el campo de los robots con inteligencia artificial cuyas acciones son generalmente llevadas a cabo por motores o motores con engranes, que mueven extremidades o impulsan al robot.[2]

Este concepto para el presente proyecto es el más acertado ya que al plantear la idea del controlador para el cuadrúpedo, toma en cuenta que el movimiento al ser activado por el mando remoto tiene que cumplir un ciclo de movimiento en el cual sólo se interviene en el momento de su programación y este activa a las etapas de potencia que hacen funcionar a los motores de las extremidades notando así su inteligencia artificial.

---

[1] ALEXANDER HRISTOV. *Robot.* 2004. <http://www.ciencia.net/VerArticulo/Robot?idArticulo=dsfjusjfr7sogujnry9x21m> (Enero/2005).

[2] Ibid.

1.2 HISTORIA DE LOS ROBOTS DEL MUNDO REAL

Uno de los primeros robots fue CLEPSYDRA o “reloj de agua”, el cual fue hecho en el año 250 a.C. por Ctesibius, físico e inventor griego que viviera en Alejandría. Este dispositivo medía el tiempo por medio de la caída de agua a través de una pequeña ranura. [3]

Pese a los avances existentes desde entonces, los más notables se consideran a finales del siglo XIX con el primer vehículo manipulado por radio control construido por Nikola Tesla (inventor Croata). Tesla es mejor conocido como el inventor de la corriente alterna, la radio (aún antes que Guglielmo Marconi), los motores de inducción, la rueda de Tesla y otros dispositivos eléctricos.

Ya en el siglo XX, en la década de los cuarentas, John Hopkins creó a SHAKEY, una caja con ruedas que utilizaba una memoria lógica que razonaba para resolver problemas en el Instituto de Investigaciones de Stanford (SRI) en Palo Alto, California. También en los cuarentas, se desarrollo el primer robot cuadrúpedo con cerebro electrónico, creado por Ralph Moser, que estuvo en el transporte cuadrúpedo (a modo de montacargas) de General Electric que caminaba a 4 millas por hora.

Durante esa misma década y los años cincuentas, Grey Walters creó a ELSIE, una tortuga; mientras que Joe Engleberger y George Devol entre los años cincuentas y sesentas crearon su compañía de robots llamada UNIMATIONS, gracias a la cual Engleberger fue llamado “*el padre de la robótica*“.

También en los años setenta se crearon el robot FREDDY de la Universidad de Edimburgo y el primer robot comercial-industrial -controlado por una mini computadora de la corporación MILACRON-, llamado T3, “la herramienta del

[3] <http://www.faculty.ucr.edu/~currie/roboadam.htm> (Enero/2005).

mañana". Freddy es un vehículo con inteligencia artificial orientado con un ojo bajo el programa de ensamblador "Versatile" con términos de secuencias de posición en el plano cartesiano el cual se sigue utilizando hasta nuestros días.

Para 1977, ASEA, una compañía de robots Europea, ofrece 2 robots con fuente eléctrica-industrial los cuales utilizaban microcontroladores para su programación y operación. Siendo ese mismo año cuando la compañía Unimation compró Vicarm, Inc., la empresa líder en el mercado de los robots y con experiencia en el diseño, dando como resultado un año más tarde, el PUMA (Maquina Universal Programable por Ensamblador, por sus siglas en español) desarrollado por Unimation, con técnicas de Vicarm y bajo el soporte de General Motors.

A lo largo de la década de los ochenta, destacó ROBOT III, un insecto de seis patas desarrollado por Roger Quinn y Roy Ritzmann.

En la década de los noventas, el Instituto de Robótica de CMU creó a DANTE II, un robot caminante de seis extremidades el cual exploraría volcanes en Alaska para la recolección de muestras de gases y a mediados de esta misma década se crea la "cirugía intuitiva" por Fred Moll, Rob Younge y John Freud diseñando y haciendo mercado de los sistemas robóticos quirúrgicos basados en trabajos de IBM y del Massachusetts Institute of Technology (en adelante, MIT).

En 1997 la NASA llevó a cabo una misión a Marte con la ayuda del PATHFINDER -el cual aterrizó en la superficie marciana- y el SOJOURNER -un vehículo de seis ruedas, mismo que tomó muestras y mando imágenes del planeta rojo-. Mientras, Honda mostraba el P3, el octavo prototipo del diseño de un *humanoide*[4] que empezó en 1986.

[4] Se le llama así al robot que imita en articulaciones y acciones a un ser humano.

Pero no fue sino hasta el año 2000 que Honda sacó a la luz a ASIMO, la siguiente generación de robots humanoides, en tanto que Sony fundaba su división Robots de Ensueño de Sony (SDR, por sus siglas en Inglés); hoy, creadores del perro robot AIBO.

Para el año 2004, el Dr. Mark W. Tilden (experto de US Robotics y BEAM), quien desarrollara varios robots para la NASA y otras agencias de investigación del gobierno Norteamericano, desarrolló el más increíble robot articulado, interactivo y animado, evitando la programación del humanoide creando al robot SAPIEN.

1.3 LOS ROBOTS CUADRÚPEDOS

Cuando los ingenieros intentaron por primera vez imitar los pasos humanos y animales descubrieron que era muy difícil, requería más poder computacional que aquél disponible en ese tiempo. Así que el énfasis se centró en otras áreas de investigación -tales como la conducta, la navegación y el planeo de ruta- para las que se utilizaron robots equipados con una sola rueda. Cuando estuvieron listos para trabajar nuevamente con los robots caminantes, comenzaron con pequeños hexápodos y otro tipo de robots de extremidades múltiples. Estos robots imitaban insectos y artrópodos en funciones y forma.

Con los robots de seis y ocho extremidades se ha observado que son estáticamente más estables debido a que tienen suficiente apoyo en el piso y que al caminar, dichas extremidades les proporcionan la seguridad y el balance que necesitan haciendo que el trabajar con ellos sea más sencillo. Aunque recientemente sólo se han hecho progresos hacia los robots con locomoción bipedal y cuadrúpedos que pierden su estabilidad por pocos segundos durante el proceso de un paso al próximo. [5]

[5] < http://www.atrox.at/robots/> (Enero/2005).

1.3.1 Aplicaciones de los robots cuadrúpedos

El sector industrial es la que más se ha enfocado a manifestar ciertas necesidades de la existencia de robots cuadrúpedos, por ejemplo:

- En la industria nuclear se necesitan robots cuadrúpedos que sean escaladores ya que podrían desarrollar tareas primarias remotas tales como inspecciones y fungir como vehículos de mantenimiento.

- En la industria de la construcción se plantea la utilización de cuadrúpedos escaladores para limpiar ventanas y evitar que el hombre se arriesgue en las alturas para tan ardua labor.

- Para industrias involucradas en actividades subacuáticas los cuadrúpedos podrían inspeccionar puentes, limpiar conductos en plantas hidroeléctricas, así como en drenajes profundos.

- En la industria minera se requiere de un cuadrúpedo para inspeccionar daños después de una explosión de gas, que ha ocurrido en donde las técnicas de monitoreo no son totalmente efectivas.

- En la industria química se utilizarían para la recolección de muestras en tierra contaminada, así como, la inspección y reparación de equipo el cual es demasiado caro o difícil de descontaminar y asegurarlo para los trabajadores humanos.

- En el caso de servicios de emergencia donde se deba entrar a áreas devastadas por el fuego o por terremotos, donde se tomen muestras y la máxima prioridad sea la búsqueda de sobrevivientes humanos.

- En el caso de misiones humanitarias se piensa en un cuadrúpedo que pueda buscar y desactivar minas antipersonales en sustitución de las personas y los perros entrenados para realizar dichas tareas.

- En la industria médica y en el caso concreto de la ortopedia, se ha pensado en usar cuadrúpedos para cargar a humanos y sustituir a la silla de ruedas; ya que con este proyecto podrían realizar actividades como el subir escaleras.

1.3.2 Algunos casos de robots cuadrúpedos

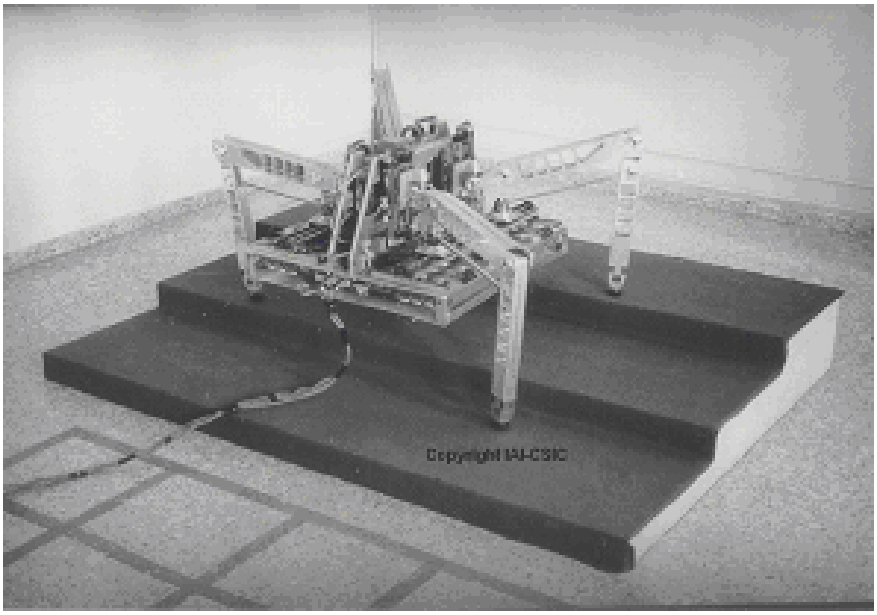


Figura 1.1 RIMHO.

RIMHO[6] es un prototipo desarrollado para plantas de energía nuclear fabricado por el Departamento de Control Automático del Instituto de Automática Industrial (en adelante CSIC por sus siglas en Inglés). Este robot también ha sido usado en pruebas concernientes al caminado de un cuadrúpedo.

RIMHO realiza un avance como el que hace un insecto, sus extremidades están basadas en un plano cartesiano tridimensional. Su estructura esta hecha de

[6] <http://www.iai.csic.es/dca/clawar_txt.htm > (Enero/2005).

aluminio y pesa aproximadamente 65 kilogramos. Su programación estructural le permite caminar en plano y hasta subir escaleras.

Figura 1.2 AIBO.

AIBO[7] es un robot mascota desarrollado y presentado en 1999 por la empresa Japonesa Sony. Su nombre quiere decir “robot” (BO) con “Inteligencia Artificial” (AI).

La unidad de procesamiento central de AIBO es un procesador llamado RISC de 64 bits de 576 MHz con 256 MB en memoria RAM, con un sistema operativo OPEN-R v.1.1.2 desarrollado por Sony.

Algunas otras características con las que cuenta son:

- Red inalámbrica IEEE 802.11b
- Salida MIDI de 64 canales
- Sensores táctiles
- Micrófonos en estero
- Panel de LEDS en la cara que indican su estado de ánimo.
- Sensor de aceleración y angular

[7] < http://www.aibo.com/> (Enero/2005)

- Sensor de vibración
- Sensor detector de bordes
- Sensor de distancia
- Sensor de imágenes de 350K píxeles

Y mecánicamente AIBO tiene 18 coyunturas siendo capaz de producir hasta 250 diferentes tipos de movimientos.

La necesidad de crear inteligencia artificial animal antes que humana es una de las premisas en que Sony basa sus trabajos, centrándose en robots que tengan procesos cognitivos y emocionales. Es así como tres mil comportamientos distintos han sido ya identificados en este pequeño ser robótico; por ejemplo, la indicación de levantarse.

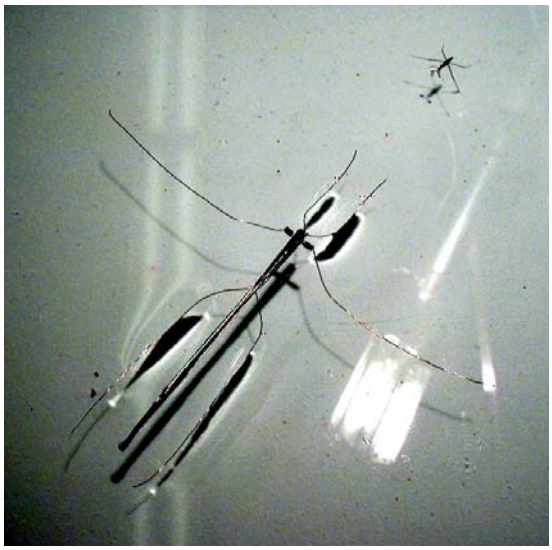


Figura 1.3 ROBOSTRIDER.

ROBOSTRIDER[8] es un pequeño robot, de un gramo de peso, capaz de caminar sobre el agua. Ha sido creado por ingenieros de la Universidad Carnegie Mellon en Pennsylvania en colaboración con el MIT.

[8] < http://www.laflecha.net/canales/ciencia/200409212/> (Enero/2005)

Por el momento, el robot no tiene cerebro ni sensores, sino una especie de músculos formados por tres interruptores eléctricos controlados por otros circuitos conectados a una fuente de energía.

El funcionamiento de este robot se basa en un reciente descubrimiento pues hasta 1993 se creía que los insectos utilizaban sus patas para crear ondas en el agua que serían las que les permitirían avanzar. Este robot, creado con el mismo sistema, se desplaza a menor velocidad que sus homólogos naturales ya que su cuerpo esta compuesto por fibra de carbono y patas de acero inoxidable accionadas por interruptores piezoeléctricos, que son los que consiguen impulsarlo sobre el agua.

Figura 1.4 Robot Granjero.

Ingenieros agrónomos de la Universidad de Illinois han desarrollado una gama de pequeños ROBOTS GRANJEROS[9] baratos (de entre 150 y 500 dólares cada uno) especialmente concebidos para realizar tareas agrícolas y sustituir a las

[9] <http://ageweb.age.uiuc.edu/faculty/teg/Research/BiosystemsAutomation/AgRobots/robotics_illinois_alumni.jpg> (Enero/2005).

pesadas y costosas maquinarias que se emplean actualmente para sembrar, fumigar, recolectar y arar la tierra.

En la actualidad, estos pequeños robots de 30 centímetros de largo sólo realizan tareas de búsqueda y transmisión de información sobre el terreno de una forma totalmente nueva en el sector agrícola ya que tienen la habilidad de funcionar como un ecosistema, es decir, se comunican entre sí y al igual que hacen las abejas, se ayudan mutuamente en caso de necesitarlo. Distribuidos por hectáreas de terreno, son capaces de orientar sus pesquisas, intercambiando información con otras unidades y de detectar epidemias e insectos peligrosos, advirtiendo de ello a los demás robots desplegados sobre el terreno.

La principal ventaja de esta generación de robots es que son pequeños, ligeros y autónomos. El peso es muy importante porque sus desplazamientos no alteran las condiciones del terreno, en contra de lo que ocurre con las actuales máquinas agrícolas grandes y pesadas, que afectan al entorno.

1.3.3 Retos actuales para los desarrolladores de robots cuadrúpedos

Compañías como Sony, Honda y algunos organismos que se dedican al diseño y construcción de robots como el CSIC buscan que los diseños experimentales se desarrollen previamente por medio de la construcción de simuladores que sean menos problemáticos en su manejo y que les permita ver los aciertos y errores.

En el aspecto electrónico con la ayuda de los microprocesadores y ahora con la nanoelectrónica, se pretende que el espacio asignado a los circuitos en el robot sea mínima y se empiece a eliminar problemas de peso.

Respecto a los motores se esta experimentando con un músculo eléctrico compuesto por filamentos de una aleación de Níquel y Titanio por el cual recibió el

nombre de Nitinol (del Inglés Nickel Titanium Naval Ordenance Laboratory), que llegará a sustituir al motor y quizá hasta al engranaje mismo, ya que al darle una excitación eléctrica el músculo se comprime y al cesar dicha excitación regresa a su tamaño original.

Un reto más de los desarrolladores ahora es librar la limitante de que el cuadrúpedo aun no sea todo terreno, ya que los existentes no están del todo armados para poder soportar la intemperie y eso llevara más tiempo de desarrollo en términos de materiales y protección.

BIBLIOGRAFÍA ESPECÍFICA

- Capítulo I -

<http://ageweb.age.uiuc.edu/faculty/teg/Research/BiosystemsAutomation/AgRobots/robotics_illinois_alumni.jpg>

<http://trueforce.com/Articles/Robot_History.htm>

<http://www.aibo.com/>

<http://www.atrox.at/robots/>

<http://www.bath.ac.uk/~en1tji/history.htm>

<http://www.ciencia.net/VerArticulo/Robot?idArticulo=dsfjusjfr7sogujnry9x21m>

<http://www.csic.es/prensa/Noticias%202004/robots.html>

<http://www.elytradesign.com/ari/html/clepsydra.html>

<http://www.faculty.ucr.edu/~currie/roboadam.htm>

<http://www.iai.csic.es/dca/clawar_txt.htm>

<http://www.iirobotics.com/webpages/robothistory.php>

<http://www.ilustrados.com/publicaciones/EpyuZlVyZlsWULVYYw.php>

<http://www.laflecha.net/canales/ciencia/200409212/>

<http://www.laflecha.net/canales/ciencia/noticias/200410041/>

<http://www.lt-automation.com/nitinol.html>

<http://www.newscientist.com/news/news.jsp?id=ns99994409>

<http://www.revista-nanociencia.ece.buap.mx/articulo1.pdf>

<http://www.roboticajoven.mendoza.edu.ar/rob_dis4.htm>

<http://www.uwe.ac.uk/clawar/home.htm>

CAPÍTULO II

2 DISEÑO PRELIMINAR

En cualquier proyecto de ingeniería es necesario tener claro el concepto del proyecto a desarrollar. Este capítulo presenta las características básicas de los robots cuadrúpedos, y da a conocer las características y limitaciones del robot que se planea diseñar y construir.

En forma general podremos definir las etapas para diseño, construcción, de partes y subsistemas así como la integración en módulos mismos que en el diseño final lograrán el movimiento unidireccional requerido.

2.1 DIAGRAMA DEL CUADRÚPEDO UNIDIRECCIONAL

El siguiente diagrama ilustra de manera general el esquema del cuadrúpedo cuya función principal es caminar hacia delante y en forma unidireccional lo mas parecido posible a un animal cuadrúpedo, como elefantes, rinocerontes o los perros. La estructura básica comprende los siguientes módulos:

- Alimentacion regulada
- Mando remoto
- Microcontrolador
- Unidad de potencia (Puente H)
- Motores de corriente continua (CC)
- Engranes
- Articulaciones

Cada símbolo (figura geométrica) que se muestra a partir del Puente H representa una extremidad del cuadrúpedo.

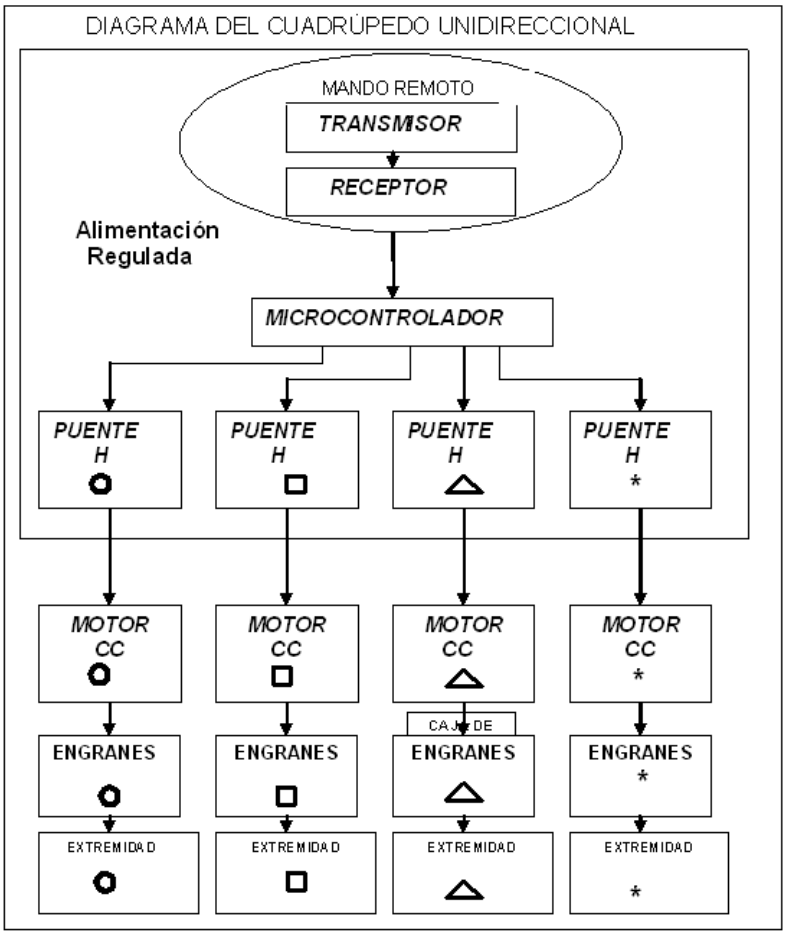


Figura 2.1 Diagrama del diseño del Cuadrúpedo.

2.1.1 Alimentación regulada

La alimentación se refiere al suministro y consumo de energía a utilizarse en el robot. Esta puede llevarse a cabo por medio de baterías comerciales ubicadas en dos partes:

- en el interior del cuadrúpedo
- en el transmisor

Otra forma de alimentación es por medio de un eliminador de baterías, pero esta alternativa incluye cableado eléctrico que no es muy funcional y atractivo para esta tesis.

En el caso particular del interior del cuadrúpedo se necesita un dispositivo que distribuya la tensión en el circuito electrónico de acuerdo al requerimiento de los módulos tanto mecánico como electrónico.

Un *regulador* es un dispositivo electrónico creado para obtener una tensión de salida deseada en función al nivel de entrada. Este consiste en fijar el valor de la tensión de salida, siendo esta típicamente de 9, 12, 15 ó 18 Volts, en función de la entrada y las condiciones de la pista. Un ejemplo mecánico es una llave de agua donde se regula el flujo que pasa por ella. Un ejemplo eléctrico puede ser el cargador de un aparato donde la entrada es la línea eléctrica y la salida, un voltaje requerido por el aparato.

Los reguladores son de dos tipos: fijos y ajustables. De esta forma se puede tener cualquier gama de tensiones con un bajo costo.

En sistemas de control se requieren valores fijos precisos, en los cuales, los reguladores desempeñan un papel muy importante para mantener el voltaje específico requerido por los dispositivos empleados.

2.1.2 Mando remoto

Para poder controlar al cuadrúpedo, se plantea el uso de un mando remoto, el cual permite manipular a distancia por medio de dispositivos electrónicos que realizan la comunicación deseada. Potencialmente esta puede ser: por medio de cables (comunicación alámbrica), o por medio de señales inalámbricas ya sean infrarrojas (IR) o de radio frecuencia.

Para los mandos remotos -alimentados usualmente con baterías-, son posibles tres técnicas fundamentales[1]:

- *La electrónica* que genera señales de mando.

[1] < http://es.wikipedia.org/wiki/Control_remoto> (Julio/2006).

- *La eléctrica* que proporciona la energía a los dispositivos, o

- *La mecánica* que transforma las señales eléctricas en movimientos mecánicos que dan lugar al desplazamiento del robot.

El mando remoto generalmente consta de 2 unidades:

- Unidad Transmisora[2]
- Unidad Receptora[3]

La primera unidad –transmisora-, es un dispositivo electrónico que porta el usuario. Su función es -usando un canal definido- codificar en un formato digital a partir de fenómenos ya sean ópticas, mecánicas o digitales para generar señales eléctricas. Estas señales se amplifican y son emitidas en forma de ondas a través de una antena.

La unidad receptora la porta el cuadrúpedo; la función del receptor es:

- Amplificar
- Filtrar
- Convertir a digital

para enviarla a nuestro dispositivo de control que procesa las secuencias de movimiento programadas.

[2] < http://es.wikipedia.org/wiki/Transmisor> (Julio/2006).
[3] < http://es.wikipedia.org/wiki/Receptor_de_radio> (Julio/2006).

2.1.3 Microcontrolador

Cuando se usa la electrónica, ya sea en computadoras, electrodomésticos y hasta en automóviles, es bueno señalar que ésta es usada para controlar alguna acción, en el caso particular del robot cuadrúpedo, requerimos de un cerebro para poder controlar sus movimientos, esto conlleva en forma general a la sincronización de movimiento de cada extremidad, y entre las 4 simultáneamente.

Una forma de hacer esto es por medio de un microcontrolador, que controle la sincronización de movimiento de las articulaciones para su desplazamiento.

Un microcontrolador es un circuito integrado[4] (también llamado chip), que tiene la mayor parte de los elementos de un controlador[5] y que cumple las funciones de cerebro de cualquier aplicación (como el encender un diodo de luz o hasta el telecontrol), siendo responsable de la buena funcionalidad del circuito que gobierna.

Como todo cerebro, este chip tiene que procesar alguna información contenida en su memoria y de esta manera decidir que hacer. A esta información se le llama programa de aplicación. Es responsabilidad del diseñador y del programador, el enviar la adecuada información a este chip para que trabaje bien[6].

El microcontrolador o chip puede ser visto externamente como un circuito integrado (llamado TTL o CMOS normal), pero internamente dispone de todos los dispositivos típicos de estos sistemas[7].

[4] <http://cholopic.pe.nu/> (Abril/2005).
[5] <http://platea.cnice.mecd.es> (Abril/2005).
[6] <http://cholopic...>, Op.Cit. (Abril/2005).
[7] Ibid.

Figura 2.2 Circuito Integrado TTL.

2.1.4 Puente H

El Microcontrolador por medio de la programación, controla el tiempo de encendido y apagado del motor, pero no puede controlar directamente al motor, para esto se requiere de otro dispositivo que ayude al motor a tener fuerza para poder mover la caja de engranes y las extremidades. Este dispositivo puede ser el Puente H.

El efecto del Puente H en un motor de corriente continua (en adelante CC), es pasar energía de forma alternada hacia el motor; es decir, energiza al motor, así como, deja de hacerlo dependiendo del mando que reciba de uno de los puertos de salida del microcontrolador. Por ejemplo, al generar un “uno lógico” si se coloca un diodo emisor de luz indicará que esta encendido y el Puente H interpretará esto como que el motor debe estar activo; pero también puede apagar a este y eso lo define con un “cero lógico” (retomando el ejemplo del LED, este se apagará)[8].

El Puente H -encargado de entender el mando lógico del microcontrolador concentra la etapa de potencia del cuadrúpedo, además de alimentar al motor con la tensión que el diseñador crea conveniente, haciendo independiente a esta ultima respecto de la tensión recibida del puerto de salida del microcontrolador.

El Puente H puede tener dos presentaciones: por transistores; siendo muy aparatosa y tiende a presentar más problemas en su operación e implementación, pues aunque puede dar soporte tanto en tensión como en potencia, abarca mucho

[8] < http://cedicyt.usach.cl/microcomputadores/proyectos/sensor/circuito.htm> (Junio/2006).

espacio ya que tiene los dispositivos de manera externa (transistores, resistencias, etc.) y es algo que se tiene que evitar en el diseño electrónico del cuadrúpedo, ya que se debe optimizar el espacio y minimizar el peso.

O bien, en circuito integrado, el cual dispone de todos los dispositivos típicos de este sistema, que lo hace más práctico de armar y obviamente ocupa muy poco espacio, y la convierte en la presentación ideal para este proyecto.

2.1.5 Motor de corriente continua

El Motor CC, es una máquina que convierte la energía eléctrica en mecánica, principalmente mediante el movimiento rotativo.

Esta máquina de corriente continua es una de las más versátiles en la industria. Su fácil control de posición, par y velocidad la han convertido en una de las mejores opciones en aplicaciones de control y automatización de procesos.

Accionar un motor de CC es muy simple, solo se necesita aplicar tensión de alimentación entre sus bornes. Ahora que el sentido de giro de un motor CC depende del sentido relativo de las corrientes circulantes por el devanado inductor e inducido. Así, la inversión del sentido de giro del motor CC se consigue invirtiendo el sentido del campo magnético o de la corriente del inducido[9].

En la construcción de robots se usan otra clase de motores como:

- Motores de Paso[10]: Este tipo de motor en el momento de ser energizado gira su eje con un ángulo definido y puede ser gobernado por mando lógico.
- Servomotores[11]: Dispositivo electromecánico utilizado en la robótica, el cual tiene la capacidad de lograr y mantener la posición que se le indica por

[9] <http://es.wikipedia.org/wiki/Motor_de_corriente_continua> (Julio/2006).
[10] <http://es.wikipedia.org/wiki/Motor_paso_a_paso> (Julio/2006).
[11] <http://es.wikipedia.org/wiki/Servomotor> (julio/2006).

medio de una señal de control. Posee únicamente tres líneas de entrada que son: tierra, Vcc, y control. La línea de tierra, está conectada al negativo de la batería; la de Vcc, al positivo; y la línea de control espera recibir un pulso positivo cada 20 milisegundos.

A diferencia de los motores de paso y los servomecanismos, los motores de CC no pueden ser posicionados en específico. Estos simplemente giran a la máxima velocidad y en el sentido que la alimentación aplicada se los permite[12].

Existen en el mercado motores de CC que tienen integrada una caja de engranes, cuya característica es, además de disminuir la velocidad, el darle más par; es decir, que puede mover al robot cuadrúpedo con su estructura y batería que proporcionalmente pesan mucho[13].

2.1.6 Engranes

El gran problema en la mecánica es la transmisión de movimiento entre un conjunto de motor y máquinas conducidas. Desde épocas muy remotas se han utilizado cuerdas y elementos fabricados de madera para solucionar los problemas de transporte, impulso, elevación y movimiento, pero nada tan eficiente como los engranajes.

El inventor de los engranajes en todas sus formas básicas fue Leonardo Da Vinci, quien a su muerte en la Francia de 1519, dejó sus valiosos dibujos y esquemas de muchos de los mecanismos que hoy en día son utilizados.

---

[12] <www.todorobot.com.ar> (Julio/2006).
[13] PALACIOS, ENRIQUE; RAMIRO, FERNANDO; LOPEZ, J. LUCAS. *Microcontrolador PIC16F84. Desarrollo de proyectos.* Colombia: Alfaomega RA-MA, 2005, p. 545.

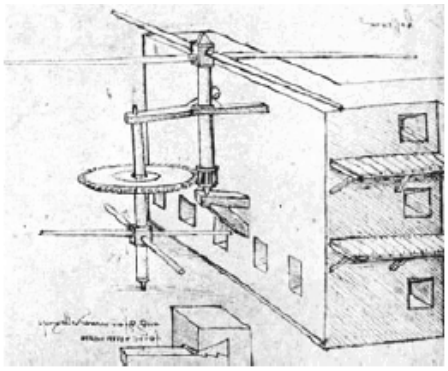


Figura 2.3 Esquema de mecanismo de DaVinci.

La forma más básica de un engrane es una pareja de ruedas, una de ellas provistas de barras cilíndricas y la otra formada por dos ruedas unidas por barras cilíndricas.

En la figura 2.3 se observa un mecanismo para repeler ataques enemigos, que consiste de aspas al nivel del techo movidas por un eje vertical, unido a una básica caja de engranes; el movimiento lo producen los soldados que giran una rueda a nivel del piso y provocan que los enemigos que alcancen el techo sean expulsados[14].

Este mecanismo muestra la transmisión entre dos ejes paralelos, uno de ellos es el eje del motor y el otro el eje conducido.

Da Vinci se dedicó mucho a la creación de máquinas de guerra para la defensa y el ataque, elaboradas todas de manera RUDIMENTARIA pues sus materiales eran madera, hierro y cuerdas; pero sus esquemas e invenciones trascienden el tiempo y muestran múltiples alternativas que brindan mecanismos básicos de palancas, engranes y poleas unidas entre sí en una máquina cuyo diseño geométrico es notable.

[14] <http://html.rincondelvago.com/engranajes_3.html> (Julio/2006).

De lo anterior, cabe destacar que la posición entre los ejes para diseñar una transmisión plantea tres situaciones: ejes paralelos, ejes que se cortan y ejes que se cruzan.

En la siguiente figura se muestra una manivela que mueve un elemento al que llamaremos engrane de gusano o tornillo sin fin, que a su vez mueve la rueda unida a él. En este caso, el mecanismo se utiliza como polea para subir por ejemplo un balde. Los ejes se encuentran en una posición ortogonal, es decir que se cruzan a 90 grados.

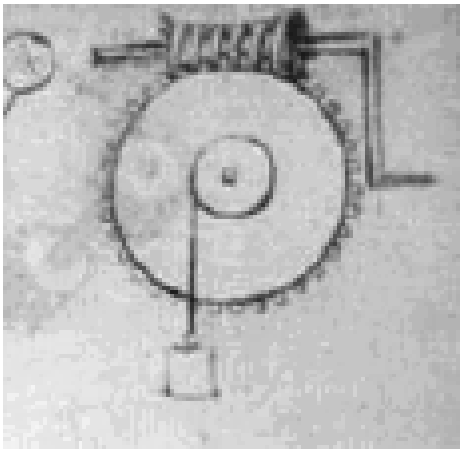


Figura 2.4 Esquema de mecanismo de DaVinci.

Así, el esquema indica los tres diámetros que definen el tamaño del diente y nos permite clasificar a los engranajes en tres grupos:

- Engranes Solidarios: los cuales tienen ejes paralelos y no se cruzan.
- Engranes Cónicos: ejes que se cortan y se cruzan.
- Engranes de gusano y rueda helicoidal: ejes Ortogonales.

Al igual que el engrane que se conecta a la polea para subir el balde del ejemplo anterior, se puede poner en este engrane un disco paralelo el cual puede mover una extremidad misma que se transforme en parte de la articulación del robot cuadrúpedo y lo haga caminar.

2.1.7 Extremidades

Se llaman extremidades a los órganos externos, normalmente articulados, que muchos animales usan como medio de locomoción. En lenguaje vulgar, muchas veces llamamos *patas* a las extremidades de los animales no humanos. Y en el caso de un cuadrúpedo, se nombran como extremidades delanteras y extremidades traseras.

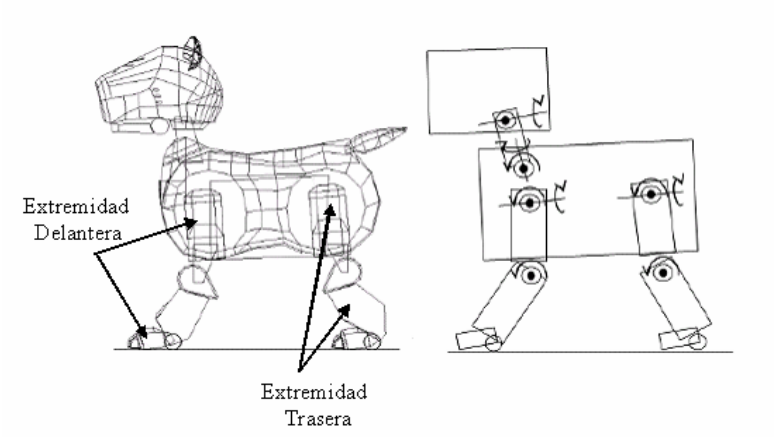


Figura 2.5 Extremidades de un cuadrúpedo[15].

Para la construcción del cuadrúpedo es necesario observar animales que tengan este tipo de locomoción. En particular, animales muy pesados como los elefantes, los cuales he observado durante mucho tiempo y han inspirado esta tesis.

[15] ZAGAL, J.C.; RUIZ DEL SOLAR, J. 2004. "UCHILSIM: A Dynamically and Visually _Realistic Simulator for the Robocup Four Legged League". *Robocup 2004 International Symposium.*

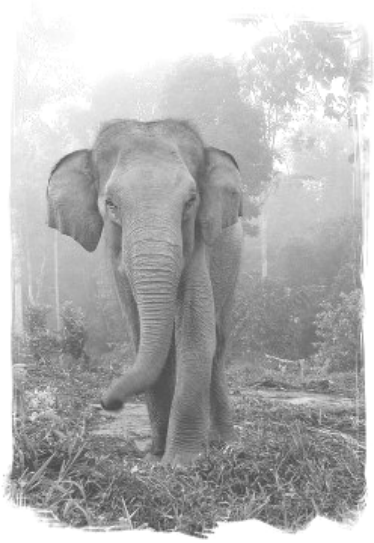


Figura 2.6 Fotografía del sitio <http://www. elephantcare.org/>

2.2 PROPUESTA DE SOLUCIÓN

Una vez definido el diagrama del cuadrúpedo unidireccional (figura 2.1), se propone la generación de cada módulo para su posterior diseño.

- **Mando remoto**

Para esta parte existen varias soluciones y se plantean como posibles las siguientes:

- Mando remoto infrarrojo limitado, el cual consta de un transmisor que funciona con transistores y para el receptor un amplificador operacional y un decodificador de tonos.

- Mando remoto de mediano alcance:

El transmisor utiliza un módulo TWS-BS-3 a 434 Mhz, que tiene un circuito integrado Encoder HT12E que transmite hasta 200 metros de distancia y es energizado por una pila de 9 Volts comercial.

El receptor usa el módulo RWS-371 a 434 Mhz que tiene un circuito integrado Decoder HT12D, energizado por una alimentación regulada a 5 Volts.

- **Microcontrolador**

Este genera la secuencia de pulsos requeridos por medio de un microcontrolador importante en el aspecto de control, ya que hace la secuencia que se necesita para que el cuadrúpedo pueda caminar.

Otra de sus características es que en el momento en que deja de recibir el pulso del mando remoto y una extremidad que esté a mitad del paso, le permite al motor por medio de pulsadores, seguir girando hasta tener la extremidad en el piso nuevamente y con ello llevar al robot a una posición estable.

Al emitir la secuencia de pulsos una vez más, estos se dirigen al Puente H que interpreta el pulso y lo convierte en movimiento otra vez, y así sucesivamente.

- **Puente H**

Este proyecto trabaja con un circuito integrado, el cual reciba cada señal del microcontrolador y mueva cada una de las extremidades, ya que al pasar por el Puente H mande a su salida la tensión necesaria para activar cada motor que mueve al engranaje. Este se debe energizar con 5 Volts y entregar una tensión de 9 Volts como la que mueve a los motores usuales.

- **Motor CC y Engranes**

Se propone un motor CC como los utilizados en auto estéreos que frecuentemente son como el BG050Y de la marca Fbelec el cual se energiza con una tensión de 9 Volts.

Figura 2.7 Motor CC propuesto.

El motor de CC del autoestéreo viene con el engranaje que expulsa al CD. Este engranaje trae un tornillo sin fin y tres engranes cilíndricos, que por el tamaño del motor y de la estructura que sostiene al engranaje, permite que se le pueda fijar sin ocupar mucho espacio en lo que será la cadera del cuadrúpedo.

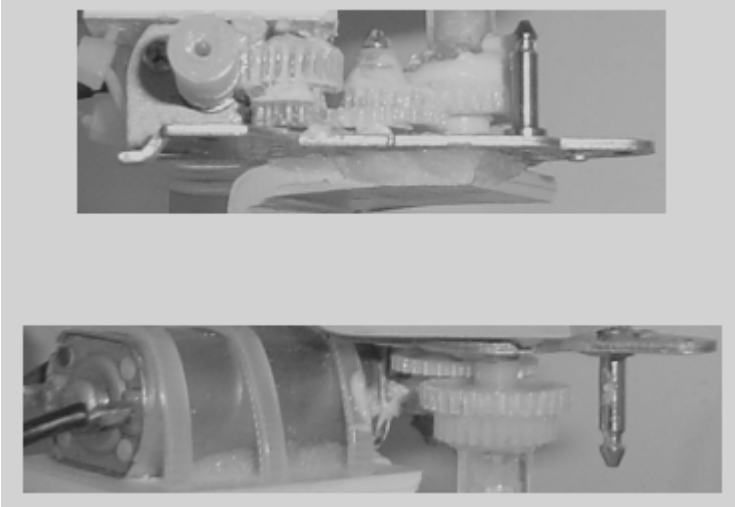


Figura 2.8 Motor CC con engranaje.

- **Extremidad**

Para la construcción de la extremidad, observé algunos modelos de juguete en forma de cuadrúpedo. Todos estos modelos tienen algo en común, cuentan con un eje descentrado y guías en forma de canal, el cual permite el desplazamiento de un eje fijo.

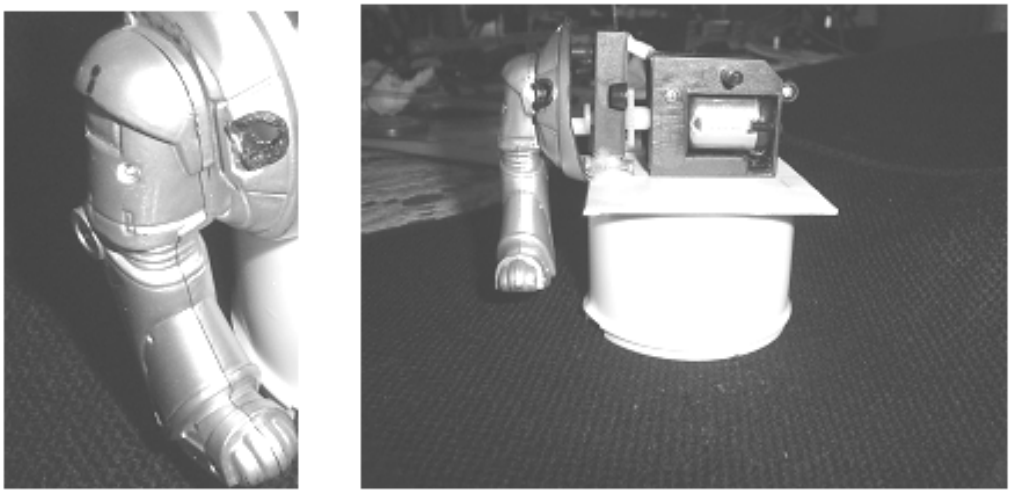


Figura 2.9 Ejemplo de extremidad de un modelo de juguete.

Este principio se observa funcionar bien en pruebas preliminares al hacer cada una de las extremidades con hoja plástica de estireno de 3mm de espesor incluyendo cadera, muslo y rodilla pegada a la entrepierna.

Ninguno de los juguetes estudiados cuenta con un motor por extremidad. La solución propuesta en ellos son engranajes que se dirigen a dos ejes, donde cada eje controla un par de extremidades.

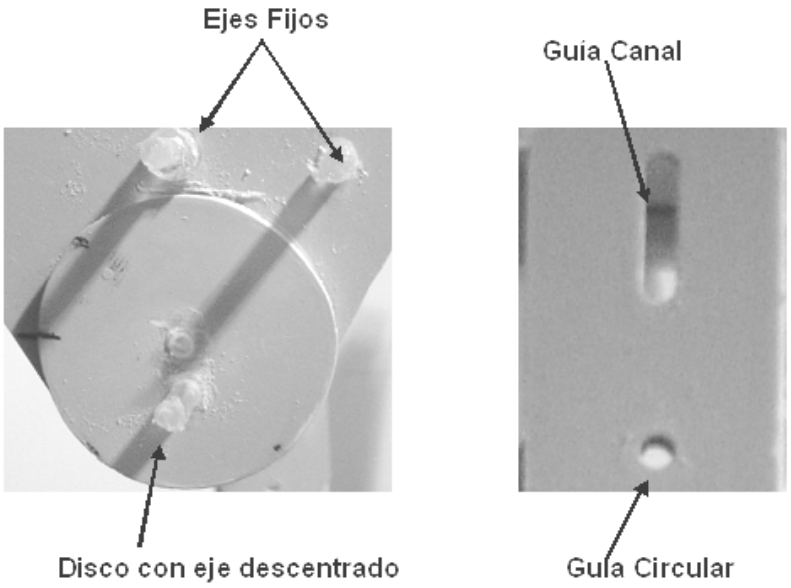


Figura 2.10 Ejemplo de extremidad con guías y ejes.

BIBLIOGRAFÍA ESPECÍFICA

- Capítulo II -

<http://cedicyt.usach.cl/microcomputadores/proyectos/sensor/circuito.htm>

<http://cholopic.pe.nu/>

<http://es.wikipedia.org/wiki/Control_remoto>

<http://es.wikipedia.org/wiki/Extremidad>

<http://es.wikipedia.org/wiki/Motor_de_corriente_continua>

<http://es.wikipedia.org/wiki/Motor_paso_a_paso>

<http://es.wikipedia.org/wiki/Receptor_de_radio>

<http://es.wikipedia.org/wiki/Regulador>

<http://es.wikipedia.org/wiki/Servomotor>

<http://es.wikipedia.org/wiki/Transmisor>

<http://html.rincondelvago.com/engranajes_3.html>

<http://platea.cnice.mecd.es>

<http://www.elephantcare.org/>

<http://www.fbelec.com/fbelec/ProductDetail.asp?ProductNO=1202&catid=51&subid=106>

PALACIOS, ENRIQUE; RAMIRO, FERNANDO; LOPEZ, J. LUCAS. *Microcontrolador PIC16F84. Desarrollo de proyectos.* Colombia: Alfaomega RA-MA, 2005.

<www.todorobot.com.ar>

ZAGAL, J.C.; RUIZ DEL SOLAR, J. 2004. “UCHILSIM: A Dynamically and Visually _Realistic Simulator for the Robocup Four Legged League”. *Robocup 2004 International Symposium.*

CAPÍTULO III

3 IMPLEMENTACIÓN DEL SISTEMA PROTOTIPO

La robótica es una de las aplicaciones más apasionantes de la electrónica ya que ésta involucra áreas de la ingeniería como son: la Mecánica y la Eléctrica.

En los últimos años, el desarrollo de la electrónica y su ímpetu por querer reducir los componentes, hace factible que se pueda construir un prototipo de robot cuadrúpedo de una forma integral, el cual puede realizar tareas concretas de movimiento.

Al plantear la implementación de un robot en general, es importante conocer un poco de la clasificación de estos, misma que considera Microrobótica (empresa española, pionera en este campo) al desarrollar sus proyectos.

Existen diferentes tipos de clasificaciones en la literatura del control moderno. La tesis se basa en el modelo de la Torre de Bot o *TorreBot*, mostrado a continuación (figura 4.1):

Figura3.1 Representación de la Torrebot.

3.1 MÓDULOS COMPONENTES

El modelo de Torrebot, describe y clasifica cada uno de los niveles tanto en su composición electrónica como mecánica y se presenta brevemente a continuación:

Nivel físico: comprende la estructura física, las unidades motoras, y las etapas de potencia. Es posible encontrar desde sistemas sumamente sencillos basados en un motor único hasta estructuras sumamente complejas que buscan emular capacidades mecánicas de algunos insectos o animales.

Nivel de reacción: está formado por el conjunto de sensores y los sistemas básicos para su manejo. Estos sensores cubren un amplio espectro de posibilidades, así podemos encontrar desde simples interruptores hasta cámaras digitales con sistemas de reconocimiento. Un robot que haya superado en cuanto a su construcción tanto el nivel físico como el de reacción, se denomina *robot reactivo*. Estas unidades trabajan cumpliendo la premisa, “acción-reacción”. En este caso los sensores son los propios controladores de las unidades motoras, sin ningún tipo de control intermedio.

Nivel de control: incluye los circuitos más básicos que relacionan las salidas de los sensores con las restantes unidades. Partiendo de una simple lógica digital y llegando hasta potentes microcontroladores que buscan dotar al robot de la capacidad para procesar la información obtenida por los sensores, así como, las funciones de actuar de manera controlada sobre las unidades motoras.

Nivel de Inteligencia: determina la característica más sobresaliente de estos robots y abarca la planificación a largo plazo. En este nivel se detallan los objetivos del robot que tienen relativa independencia de los sensores. Este es el nivel más alto de inteligencia que puede alcanzar un robot con una unidad individual.

Nivel de comunidad: se trata de la puesta en funcionamiento de más de un robot dentro de un mismo entorno, todos ellos funcionando de forma simultánea y sin que ninguno tenga conocimientos explícitos de la existencia de los otros. A estos recintos se les denomina granjas. Los centros de investigación utilizan las granjas como entornos de observación de los robots. Dichos establecimientos pueden contar con sistemas sofisticados que permiten a un operario monitorear y evaluar el comportamiento de la comunidad, así como alterar las condiciones externas del sistema (agregar obstáculos, cambiar la temperatura, etc.).

Nivel de cooperación: comprende los sistemas donde a partir de un nivel de comunidad se planifican o programan los robots para que tengan conocimiento de la existencia de otros, de manera que posean la capacidad de cooperar para el buen desarrollo de una tarea. Dentro de este grupo estarían los populares equipos de fútbol constituidos por robots.

Este trabajo llega hasta el *nivel de control* dentro de la clasificación de la Torrebot, a partir de la cual se desglosa cada uno de los elementos que componen los diferentes niveles.

3.2 IMPLEMENTACIÓN A NIVEL FÍSICO

Para su elaboración a nivel físico es necesario considerar en primer lugar el factor del peso, ya que es una de las principales limitantes a vencer, así como es la estructura de implementación para los movimientos. Tomando en cuenta esto, el diseño del prototipo primero debe determinar a los motores para la realización del movimiento deseado.

3.2.1 Motores

Para este proyecto se probaron varios motores, como el MABUCHI modelo FA-280RA, el cual se energiza con una tensión de 1.5 a 3.0Volts como el mostrado en la figura 3.2.

Figura 3.2 Motor Mabuchi.

Conforme se desarrollaron las pruebas, se observó que el motor se calienta, requiriendo de mayor tensión para los diferentes movimientos, por lo que fue importante escoger motores que no presentasen estas características. De igual manera, se probaron algunos otros, pero para nuestro diseño final se determinó que el más óptimo es el motor BG050Y de la marca Fbelec, como ya se mencionó anteriormente en el capítulo II (figura 2.8).

3.2.2 Etapas de potencia

Para el control de los diferentes motores, es necesario un dispositivo que suministre la suficiente corriente constante para el movimiento. Este dispositivo pertenece a la etapa de potencia.

Para las pruebas iniciales se uso el L293[1] también llamado Puente H, pero este además de suministrar la suficiente corriente, presentaba calentamientos provocando fallas en los movimientos. Esto llevo a un diseño de Puente H con transistores, que puede ser mas eficiente pero generalmente muy grande y por ello resulta necesario reducir en lo posible los dispositivos, ya que no debemos olvidar que en el prototipo, una de las limitaciones es el peso así como el suministro de energía del robot, siendo que para tal fin se utilizaron baterías alcalinas en las pruebas realizadas.

En la figura 3.3 se muestra el diagrama esquemático del Puente H armado con transistores; mientras que la figura 3.4 permite observar la configuración del L293.

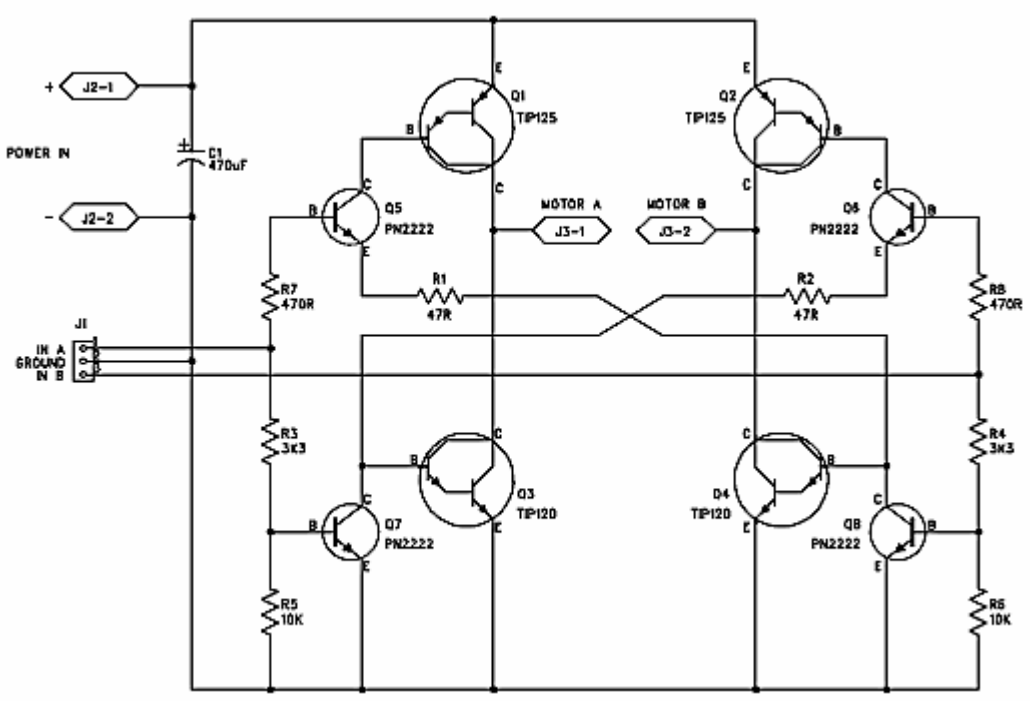


Figura 3.3 Diagrama con transistores.

[1] El L293 es un manejador clásico de motores que se puede utilizar como control bidireccional para dos motores (Puente H), pero esta limitado por una corriente de tan solo 600 mA, y de hecho solo se puede usar con corrientes más pequeñas estando limitado a pequeños motores.

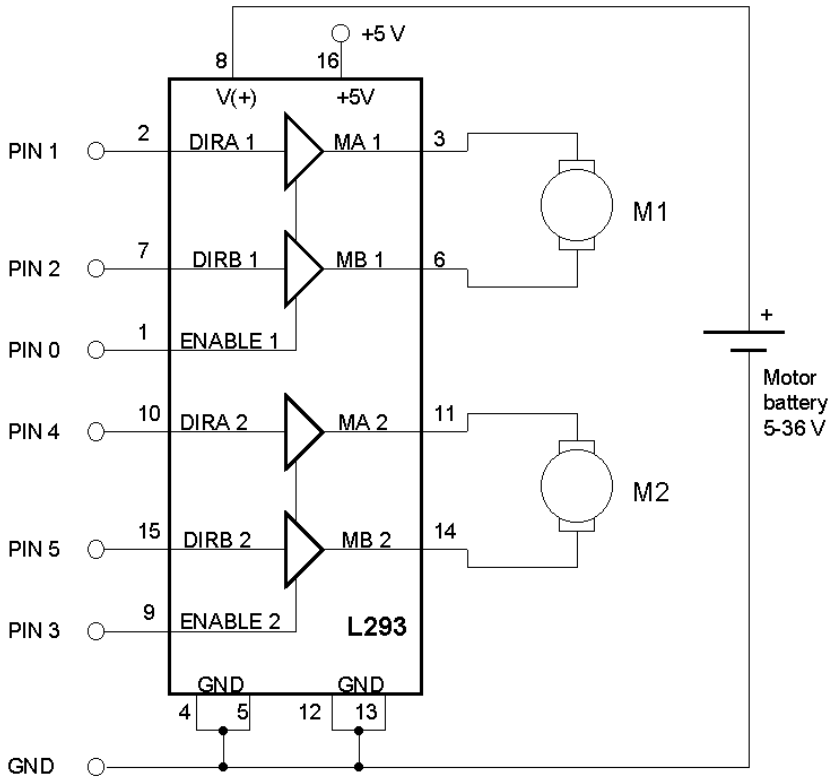


Figura 3.4 Configuración del L293.

Los Puentes H son al igual que el L293, circuitos que permiten controlar motores eléctricos de corriente directa en dos direcciones desde un circuito digital (TTL, CMOS, el puerto de una computadora, desde un microcontrolador, etc.). Se les llama Puentes H porque precisamente su forma en el diagrama recuerda de manera vaga la letra *H*[2].

Es importante que la etapa de potencia no genere fallos, para que el control de los movimientos pueda ser estable, por lo que en el prototipo final se probó un L298.

[2] <http://www.creaturoides.com/anterior/puentesh.htm> (Abril/2005).

3.2.3 Engranes

Al tener el motor de corriente continua seleccionado, la idea de adaptarle engranes comenzó a ser viable y confiable para los movimientos, por esta razón es importante hacer hincapié en cómo funciona el engrane, adaptado a un motor para sus diferentes movimientos.

Un ejemplo sencillo de engranes para movimiento es la figura 3.5, que representa un conjunto engranes que se planeó moverían las extremidades con unos ejes de acero manteniéndolos sincronizados. Esto dejó de ser funcional en el momento en que se decidió que las extremidades serían independientes.

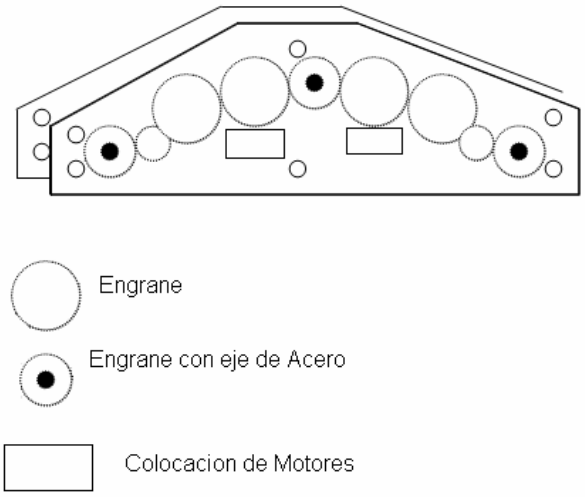


Figura 3.5 Prototipo de caja de engranes.

La caja de engranes de la figura anterior (3.5) tiene 3 ejes. En cada uno se montaban dos discos con ejes descentrados de acero, de tal forma que las posiciones de los ejes descentrados sean contrarias como muestra se muestra a continuación (figura 3.6).

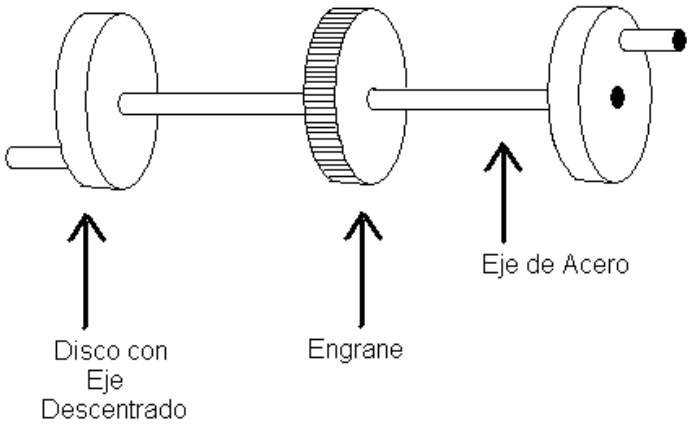


Figura 3.6 Engranes con eje de acero y discos con ejes descentrados.

El modelo anterior de engranes da la base del uso de un eje descentrado tal y como lo usan los modelos comerciales (por ejemplo, el robot cuadrúpedo marca Star's Dog), el cual se muestra enseguida (figura 3.7):

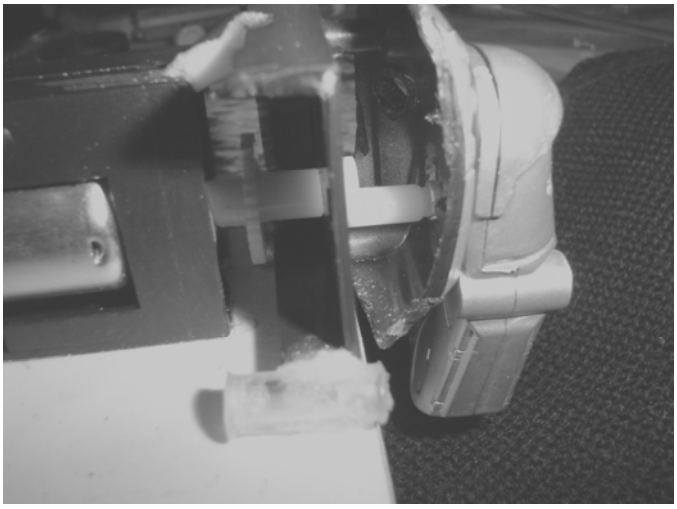


Figura 3.7 Modelo comercial de engranes.

La figura también permite observar un motor MABUCHI modelo FA-280RA con un disco con eje descentrado, un enlace plástico que une al motor con el disco, y una base para fijar la extremidad mostrada.

Al adaptar la extremidad al motor, generalmente no resulta sencillo mantenerlas estables, provocando que se despeguen los enlaces, lo que hizo pensar en buscar una solución más integral como lo son los motores CC con engranajes.

Esta clase de motores CC los podemos encontrar frecuentemente en juguetes (como el Mecano, Lego, Tomy, entre otros), que además de disminuir la velocidad -como ya se comentó antes-, le dan más par, lo que le permite mover robots con estructuras y batería que proporcionalmente pesan mucho.

En este sentido, los engranes se usan tanto para acelerar o frenar al robot, como para hacerlo más fuerte o más débil[3].

En general la relación de una combinación de engranes y motor (que en su versión más simple es de velocidad constante), da movimientos mas finos y de fácil control. Si se aumenta la velocidad del motor, el torque es menor al principio de los engranes, esto funciona de manera similar para el caso del motor que disminuye su velocidad.

3.2.3.1 Tipos de engranes

Existen muchos tipos de engranes, pero aquí sólo revisaremos las características de cinco de ellos:

Engranes Solidarios: son ruedas con dientes que comparten un plano pero con ejes distintos y permiten aumentar o disminuir el par, que se aplica a uno u otro engrane, en otras palabras son bidireccionales como los mostrados en la figura 3.8.

[3] <http://www.me.umn.edu/education/courses/me2011/robot/technotes/L293/L293.html> (Abril/2005).

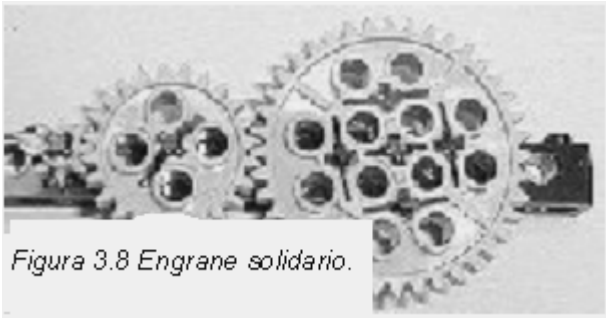


Figura 3.8 Engrane solidario.

Engranes Cónicos: coinciden en ángulos (de preferencia ángulos rectos), de manera que cambian la dirección de la rotación y permiten cambiar la dirección del desplazamiento del motor como se aprecia en la figura 3.9.

Figura 3.9 Engrane cónico.

Engranes de Gusano: estos engranes son llamados tornillos sin fin y pueden adaptarse a los engranes solidarios (figura 3.10). Este tipo de engranes tienen las siguientes propiedades:

1) Cambia la dirección de rotación donde el eje de salida es perpendicular al eje de entrada (similar a los engranajes cónicos).

2) Producen un gran incremento de la fuerza con una consecuente disminución de la velocidad. Cuando el tornillo da una vuelta, avanza un diente del engrane solidario conectado a él. Esto significa que si un engranaje de 24 dientes esta conectado al tornillo, la relación será de 24 a 1.

3) Los engranes de gusano generalmente sólo se mueven en una sola dirección. El engranaje de gusano puede mover al solidario pero el solidario no puede mover al engranaje de gusano.

En el caso particular del cuadrúpedo que nos ocupa, este integra unos de estos engranes para poder detener el movimiento de la caja de engranes cuando no este energizada.

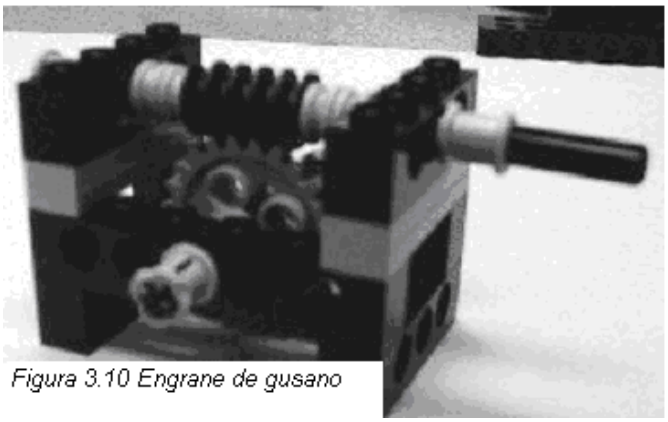


Figura 3.10 Engrane de gusano

Sistema de piñón y cremallera: son un par de engranes especiales que constan de una cremallera que es como un engrane solidario pero estirado y puesto en línea. El piñón es el pequeño engrane solidario conectado al rack.

Cuando el piñón rota, la cremallera avanza o retrocede o si la cremallera avanza o retrocede, rota el piñón. Así, el sistema de cremallera y piñón transforman movimiento rotacional en movimiento lineal y viceversa como se muestra en la figura 3.11.

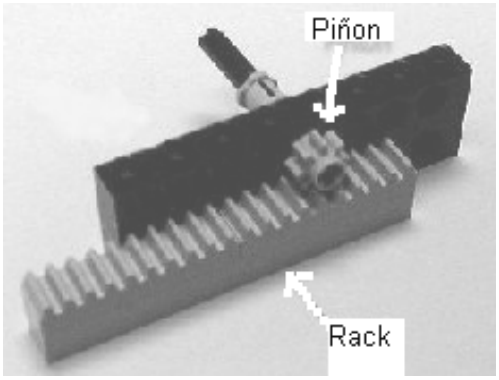


Figura 3.11 Sistema de piñón y cremallera.

Razón compuesta de Engranes: cuando se usan más de un par de engranes, el montaje se denomina tren compuesto. La relación de los engranes para cada par individual se multiplica por la de los otros para obtener la razón compuesta de engranes para el total de los engranes en el tren.

Para el modelo final, el motor BG050Y -de la marca Fbelec- funcionará con una razón compuesta de un engrane de gusano y 3 ruedas dentadas en posición de engranes solidarios, que permite a la extremidad mantener la posición que le de esta en caso de que deje de ser energizado el motor.

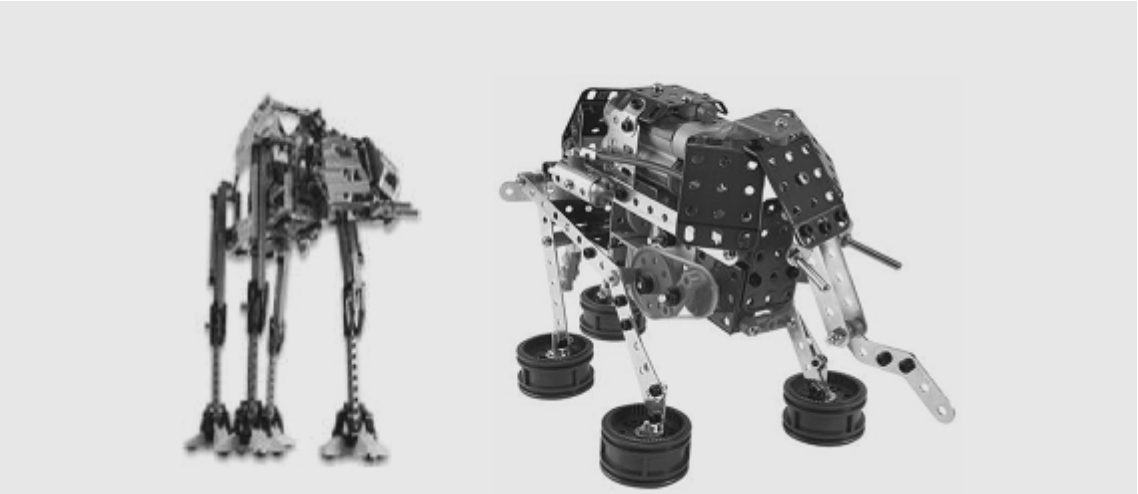
3.2.4 Estructuras

Para construir un robot se puede usar varias estructuras para poder tener el cuerpo del robot donde se montan los motores con engranes y hasta las extremidades, dependiendo de lo que se quiera realizar. Hay estructuras comerciales que son las que más se usan con fines educativos.

3.2.4.1 Estructuras comerciales

Las estructuras más utilizadas son las de los juegos educativos de construcción tipo Lego, Mecano o Eitech, interesantes por su flexibilidad. Para un diseño un poco más profesional, se pueden utilizar las estructuras de Fischer Technik que fueron diseñadas originalmente para aplicaciones técnicas, tanto estáticas como de estructuras mecánicas con movimiento. La figura 3.12 muestra modelos de robots cuadrúpedos de las marcas Lego y Mecano[4].

[4]<http://www.otherlandtoys.co.uk/product529/product_info.html?name=Meccano%20Best%20of%2040%20Set&&prod=529>(Julio/2006).



a) Lego b) Mecano

Figura 3.12 Modelos comerciales de estructuras de robots cuadrúpedos.

Un ejemplo de estructura comercial lo introdujó la compañía Fischer Technik[5], en la cual se puede colocar el circuito electrónico, la batería, además de la estructura mecánica, como se muestra en la figura 3.13.

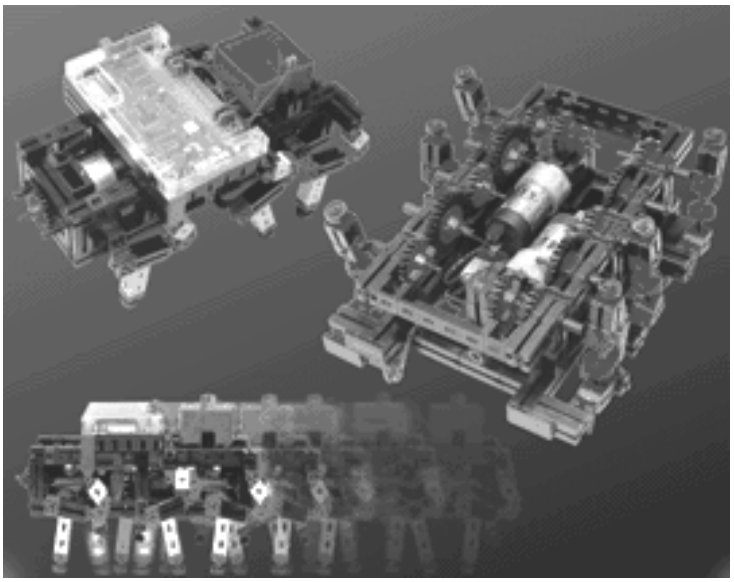


Figura 3.13 Estructura comercial de Fischer Technik.

[5] < http://www.personal-robotics.de/personal-robotics/index_hauptshop.html?http://www.personal-robotics.de/personal-robotics/fischertechnik_computing.htm>(Julio/2006).

En el modelo prototipo de la caja de engranes (figura 3.5) y en el modelo de juguete (figura 3.7), se uso una hoja de acrílico (de 15cm x 10cm y 3mm de espesor) de estructura. En el primer caso, se fijaron los engranes a esta hoja con buenos resultados, pero en el segundo caso, el acrílico se quebró con facilidad debido al peso, lo que provocó que este tipo de material quedara fuera del modelo final.

3.2.5 Extremidades

Una vez elegida la estructura donde se va a colocar la electrónica, el mecanismo y las baterías, se fijan las extremidades del cuadrúpedo, ya que se tienen que hacer los arreglos necesarios para la mejor posición y si es necesario la modificación de la estructura, para que no tengan ninguna obstrucción al ser energizado el motor y la extremidad cuando se mueva.

Las pruebas experimentadas a las extremidades del cuadrúpedo se basaron principalmente en las que realizan los juguetes, ya que su movimiento es lo más cercano al movimiento que se busca. Para conocer los fundamentos del movimiento de un modelo comercial cuadrúpedo se recurrió a la extremidad del perro robot marca Star’s Dog por ser accesible.

3.2.5.1 Extremidad del perro robot marca Star’s Dog

Este juguete fue un intento de hacer al cuadrúpedo con 4 motores independientes; hecho con piezas comerciales y fijado a una estructura de acrílico como la ya mencionada. La figura 3.14 muestra el prototipo de robot cuadrúpedo con piezas comerciales del perro robot de Star´s Dog.

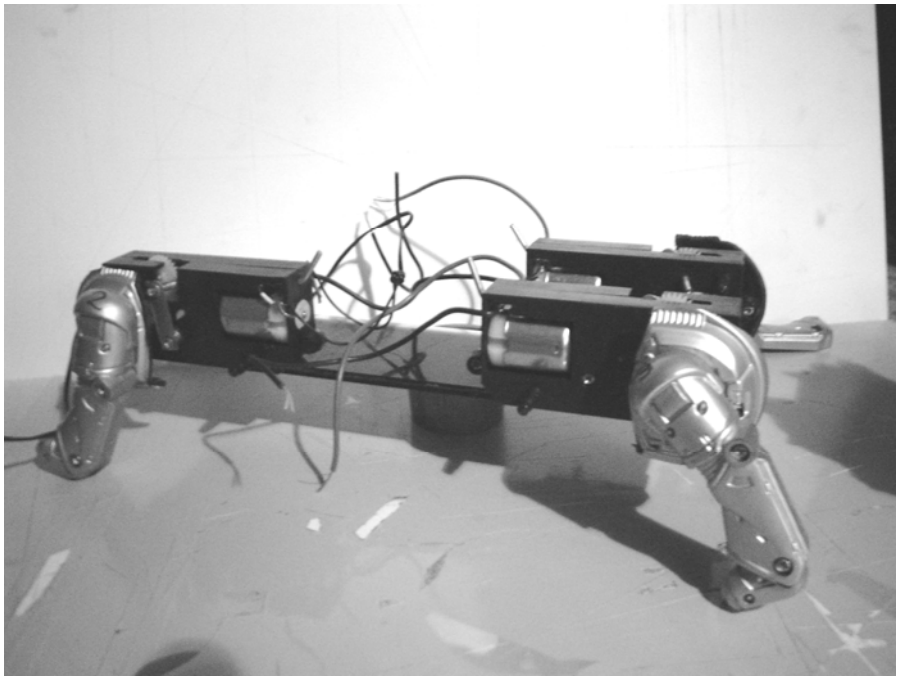


Figura 3.14 Prototipo de robot cuadrúpedo.

La extremidad es rígida (por lo que no esta provista de rodillas ni de patas móviles). Se conecta a la cadera y se fija esta a la estructura de acrílico como se muestra en la figura 3.15.

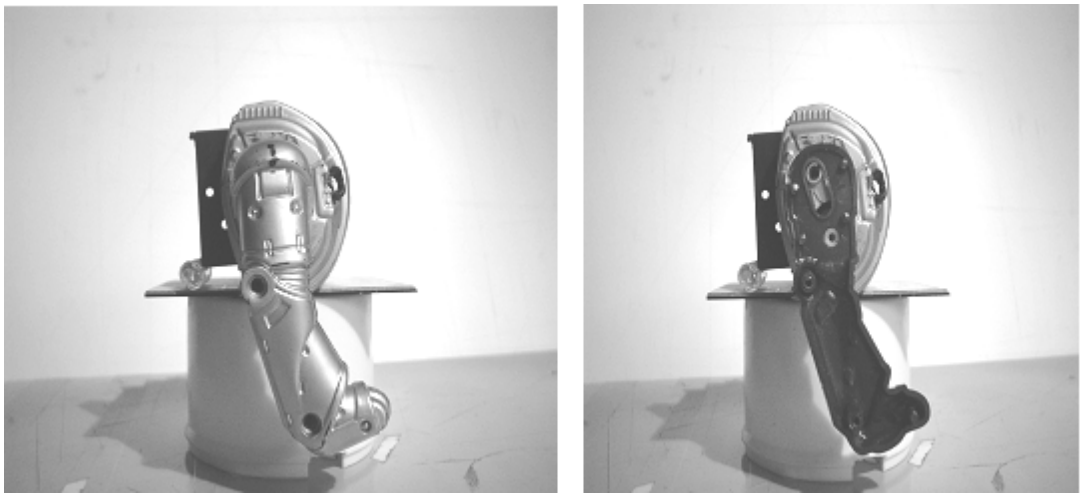


Figura 3.15 Fijado de la extremidad a la estructura.

La cadera cuenta con un eje fijo y un disco con eje descentrado en el cual se conecta al engranaje y este al motor que se encarga de generar el movimiento de la extremidad como se muestra en la figura 3.16.

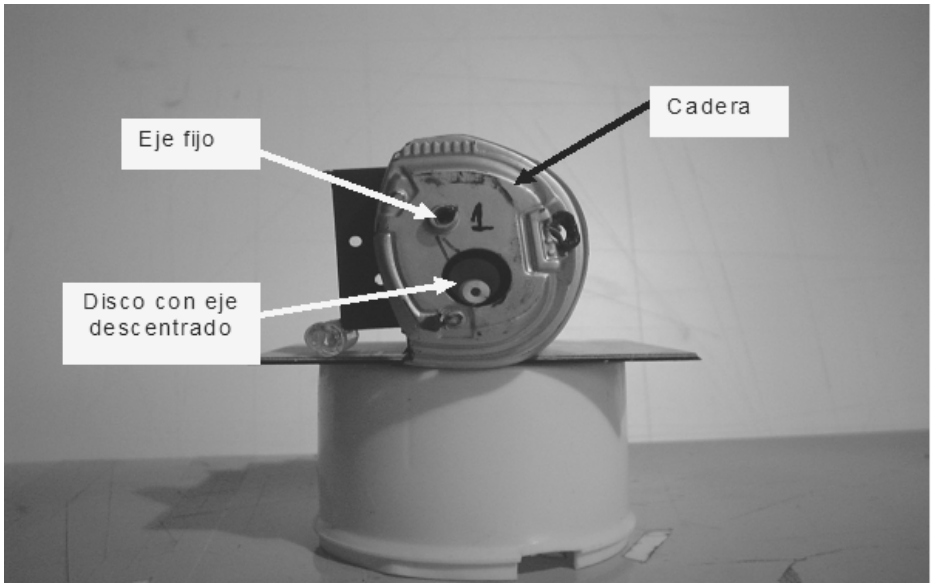


Figura 3.16 Partes de la cadera.

La extremidad en la parte superior (muslo) tiene una guía canal, la cual se une al eje fijo que se encuentra en la parte superior de la cadera y debajo de ella una guía circular que se conecta al eje fijo que tiene el disco con el eje descentrado. La figura 3.17 muestra como se fija la extremidad a la cadera.

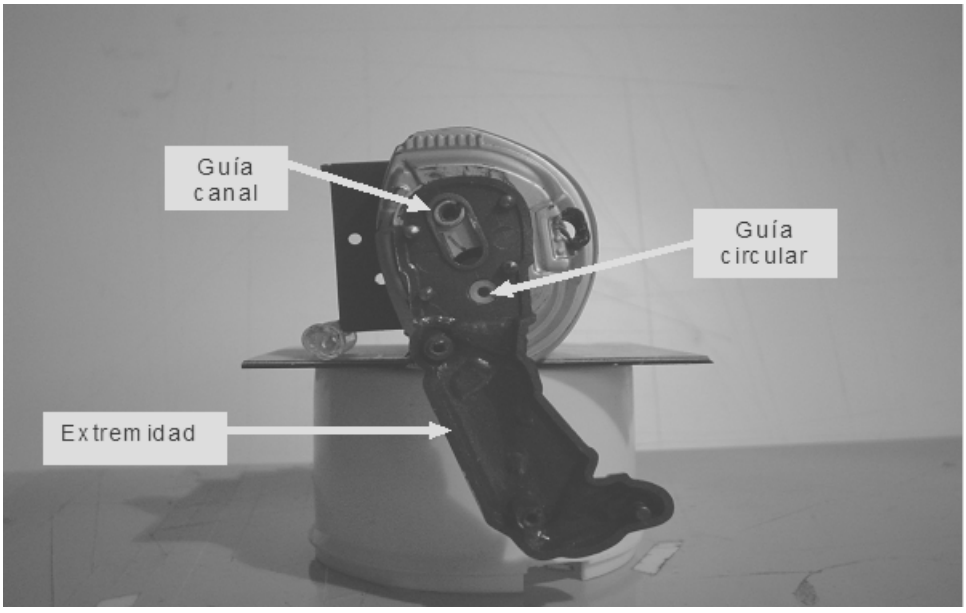


Figura 3.17 Fijado de la extremidad a la cadera.

Cuando se hace girar al eje descentrado, las guías y los ejes generan en la extremidad el movimiento que permite el desplazamiento.

Se decidió no usar este modelo porque no se pudo fijar bien las extremidades a la estructura, por lo tanto el robot no cumplió con el movimiento unidireccional.

3.3 IMPLEMENTACIÓN A NIVEL DE REACCIÓN

Para la implementación a nivel de reacción en la cual se requiere de sensores para manipular al robot, se propone mantener el cuadrúpedo en una posición inicial (una vez que haya terminado la secuencia de PARO). Los motores pueden mantener una posición mediante la sincronización, pero siempre es mejor la ayuda de sensores para lograr que estos se detengan justo en donde se requiere.

3.3.1 Tipos de sensores

Tipos de sensores que se utilizan para controlar a un robot:

A) LDR, detector pasivo de luz: Las resistencias dependientes de la luz. LDR (del Inglés Light Dependent Resistor) o Fotorresistencias, son dispositivos que varían su resistencia en función de la luz que incide sobre su superficie.

B) CNY70, sensor óptico reflexivo por infrarrojo con salida a transistor: Fabricado por Vishay Telefunken Semiconductor. Tiene una construcción compacta donde el emisor de luz y el receptor se colocan en la misma dirección para detectar la presencia de un objeto por medio del empleo de la reflexión del haz de luz infrarroja (conocido por sus siglas IR del ingles Infrared) sobre el objeto.

C) OPB703/4/5, sensor óptico reflexivo por infrarrojo: Fabricado por Optek Technology, son también del tipo reflexivo y utilizan como emisor un diodo LED emisor de infrarrojos y como receptor un fototransistor. En este caso están montados en una carcasa negra y la forma del sensor permite que el haz refleje en una superficie más concreta que el CNY70.

D) H21A1, sensores óptico de barrera: Fabricado por Isocom Components y Fairchild Semiconductors. Estos sensores también tienen como emisor un diodo de infrarrojos y como receptor un fototransistor. En este caso el emisor y el receptor están enfrentados a una distancia de 3mm y entre ellos existe un espacio para que un objeto pueda introducirse y romper la barrera infrarroja.

E) GP2Dxx, sensores infrarrojos para medición a distancia: Los IR Sharp GP2Dxx son una familia de sensores de infrarrojos compactos de alta sensibilidad para la detección y medida de distancia. Estos dispositivos emplean el método de triangulación, utilizando un pequeño Sensor Detector de Posición lineal (conocido por sus siglas PSD del Inglés Position Sensitive Detector), que determina la distancia o la presencia de los objetos dentro de su campo de visión. Básicamente su modo de funcionamiento consiste en la emisión de un pulso de luz infrarroja. que se transmite a través de su campo de visión, mismo que se refleja contra un objeto o que, por el contrario no lo hace. Si no encuentra ningún obstáculo el haz de luz no refleja y en la lectura que se hace indica que no hay ningún obstáculo. En el caso de encontrar un obstáculo el haz de luz infrarroja se refleja creando un triangulo formado por el emisor, el punto de reflexión (obstáculo) y el detector.

F) IS471F, sensor detector de proximidad de obstáculos por infrarrojos: El IS471F fabricado por Sharp Corporation es un detector de obstáculos infrarrojo que es capaz de detectar cuando se refleja sobre un objeto el haz que emite un fotodiodo emisor de infrarrojos.

G) BUMPER, final de carrera mecánico para detección de obstáculos: Es un conmutador que cambia de posición al realizar la presión necesaria sobre la lámina de metal que hace de brazo de una palanca de primer orden que a su vez activa un pulsador que dispone de un muelle de recuperación. Se comercializan BUMPERS con diferentes longitudes de lámina según la aplicación a utilizar.

H) SRF04, sensor de obstáculos y medidor de distancias por ultrasonido. Los detectores de obstáculos por ultrasonidos emiten pulsos de ultrasonido mediante un dispositivo transmisor, cuando las ondas ultrasónicas se reflejan sobre algún objeto, a través de una cápsula sensible se captan los pulsos reflejados. El tiempo que tardan en volver los pulsos reflejados es proporcional a la distancia del objeto que se reflejan.

Para el diseño final, los sensores que se utilizaron son los bumpers, debido a su sencillez, además de la lamina que activa al bumper es de gran utilidad para determinar a la extremidad en la posición inicial.

3.4 IMPLEMENTACIÓN A NIVEL DE CONTROL

En la implementación a nivel de control principalmente se incluye al mando remoto (que de manera inalámbrica se comunica transmisor y receptor), y a la secuencia del movimiento del robot cuadrúpedo.

3.4.1 Mando remoto

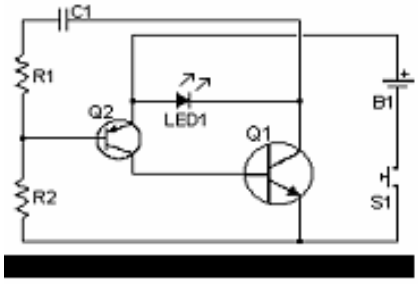
Dentro del desarrollo del control existen varios circuitos para este fin tales como: el mando remoto infrarrojo, el cual en su transmisor funciona con transistores y se energiza con una tensión de 3 Volts y para el receptor un amplificador operacional y un decodificador. Para polarizar al amplificador se usan

las tensiones de ± 9 Volts y el decodificador se energiza con una tensión de 9 Volts. Para la salida del decodificador se usa un relevador con una tensión de 5 Volts.

Uno de los principales problemas es la distancia de comunicaciones entre el transmisor y el receptor, teniendo otra limitación del circuito: su tamaño, que en general se busca la opción mas compacta para cumplir con la condición del espacio y el peso.

A continuación se muestra el diagrama esquemático del mando remoto infrarrojo genérico.

-Transmisor



-Receptor

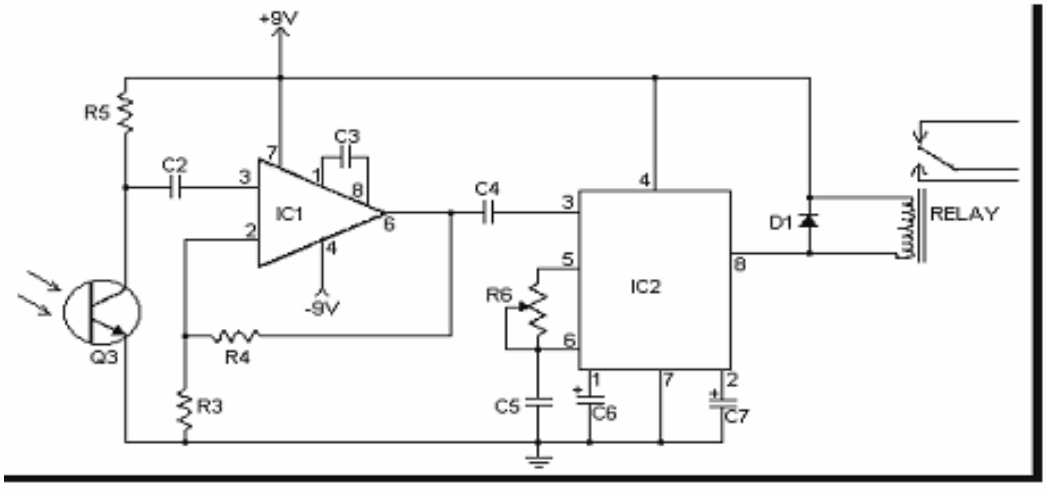


Figura 3.18 Mando remoto infrarrojo genérico.

En donde,

R1_____22K 1/4W Resistencia
R2_____1 Meg 1/4W Resistencia
R3_____1K 1/4W Resistencia
R4,R5_____100K 1/4W Resistencia
R6_____50K Potenciómetro
C1,C2_____0.01uF 16V Capacitor Cerámico
C3_____100pF 16V Capacitor Cerámico
C4_____0.047uF 16V Capacitor Cerámico
C5_____0.1uF 16V Capacitor Cerámico
C6_____3.3uF 16V Capacitor Electrolítico
C7_____1.5uF 16V Capacitor Electrolítico
Q1_____ Transistor 2N2222 NPN
Q2_____Transistor 2N2907 PNP
Q3_____NPN Fototransistor NPN
D1_____ Diodo 1N914
IC1_____ Amplificador Operacional LM308
IC2_____ Decodificador de tonos 567
LED1_____ LED Infrarrojo
RELAY_____ Relevador 5Volts
S1_____SPST Interruptor Push Button
B1_____ Batería de 3Volts

Para el diseño final se propone usar un control remoto de radio frecuencia (conocido por sus siglas RF) que consta de un módulo receptor RWS-371 a 434 Mhz, tiene un circuito integrado Decoder HT12D y se energiza por la alimentación regulada a 5 Volts y un módulo transmisor TWS-BS-3 a 434 Mhz, que tiene un circuito integrado Encoder HT12E que transmite hasta 200 metros de distancia, alimentado por una pila de 9 Volts comercial.

3.4.2 Circuito para la secuencia de movimiento

La secuencia de movimiento por medio del circuito combinacional adaptada a osciladores, es engorroso e impracticó, ya que si existe un cambio en relación al tiempo de la secuencia, se tiene que rediseñar la electrónica. Para evitar este tipo de problemas, se usan los microcontroladores con el cual es fácil cambiar la secuencia por medio de programación y evitar los cambios de electrónica. Este dispositivo tiene la ventaja, de no ser muy grandes, y no requieren de muchos componentes físicos, ya que casi todo lo tiene en su interior. Así el microcontrolador se convierte en el cerebro del robot cuadrúpedo.

Inicialmente se probo el Microcontrolador 68HC-11 con la tarjeta entrenadora FACIL 1B el cual se energiza con 5 Volts por un regulador 7805 (el cual venía incluido en la tarjeta entrenadora), la cual por medio de la salida en el Puerto B (que tenía un LED conectado en cada salida del Puerto), pudo hacer una secuencia de pulsos que se programa y compila desde una PC y notando que cada LED puede ser reemplazado por cualquier otro dispositivo como por ejemplo: *un motor*. Lo que además permitió que el microcontrolador pudiera definirse como controlador de entrada/salida con capacidad de decisión.

3.4.2.1 Microcontrolador 68HC-11

El Microcontrolador 68HC-11[6] destaca por sus recursos, simplicidad y facilidad de manejo. Este es un circuito integrado que contiene un CPU (Unidad Central de Proceso), el cual puede ejecutar instrucciones almacenadas en su memoria.

El 68HC-11 también tiene varios recursos internos como la memoria RAM, la memoria ROM, la memoria EEPROM, puertos serie, puertos de entrada/salida, temporizadores, comparadores, etc.

[6] <http://www.iearobotics.com/proyectos/libro6811/libro-6811.pdf>(Julio/2006).

El Microcontrolador 68HC-11 de Motorola es de 8 bits fabricado con tecnología HCMOS con una frecuencia de bus de 2 Mhz. Los recursos disponibles según el modelo A1 son:

- 256 bytes de memoria RAM
- 5 puertos de 8 bits, con pines de entrada, salida y de entrada /salida
- Conversor Analógico – Digital de 8 canales y 8 bits de resolución.
- Una UART para comunicaciones series asíncronas (SCI)
- Un módulo de comunicaciones serie síncronas (SPI)
- 5 Comparadores con salida hardware
- 3 capturadores de entrada
- Un acumulador de pulsos externos de 8 bits
- Interrupciones en tiempo real
- 2 entradas de interrupciones externas
- Software en ROM para cargar un programa externo en la RAM interna

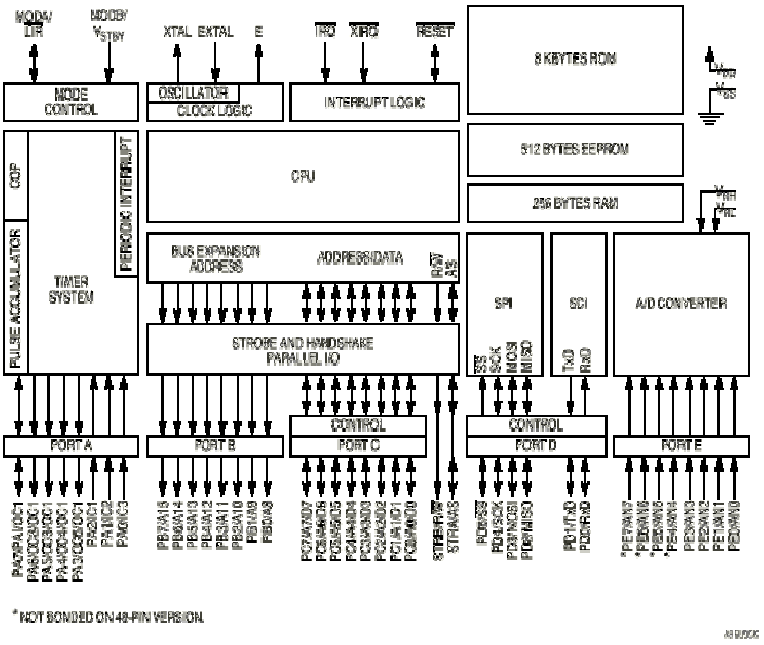


Figura 3.19 Diagrama de Bloques del 68HC-11

Pero existen microcontroladores tales como el PIC 16F84A que no requieren una tarjeta entrenadora y que son operables con un mínimo de componentes, así como de espacio para poder colocarla en una tarjeta junto a todos los demás dispositivos necesarios para su control.

3.4.2.2 Microcontrolador 16F84A[7]

El controlador ha permanecido invariable a través del tiempo, pero su implementación física ha variado frecuentemente.

Los recursos del Microcontrolador 16F84A son:

- Procesador o UCP (Unidad Central de Proceso)
- Memoria RAM para Contener los datos
- Memoria para el programa tipo ROM/PROM/EPROM
- Líneas de E / S para comunicarse con el exterior
- Diversos módulos para el control de periféricos (temporizadores, Puertas Serie y Paralelo, CAD: Conversores Analógico / Digital, CDA: Conversores Digital / Analógico, etc.)
- Generador de impulsos de reloj que sincronizan el funcionamiento de todo el sistema

[7]<http://platea.cnice.mecd.es>(Marzo/2005).

La figura 3.20 muestra el diagrama de bloques del 16F84A el cual se muestra con mayores detalles en el *anexo 1*.

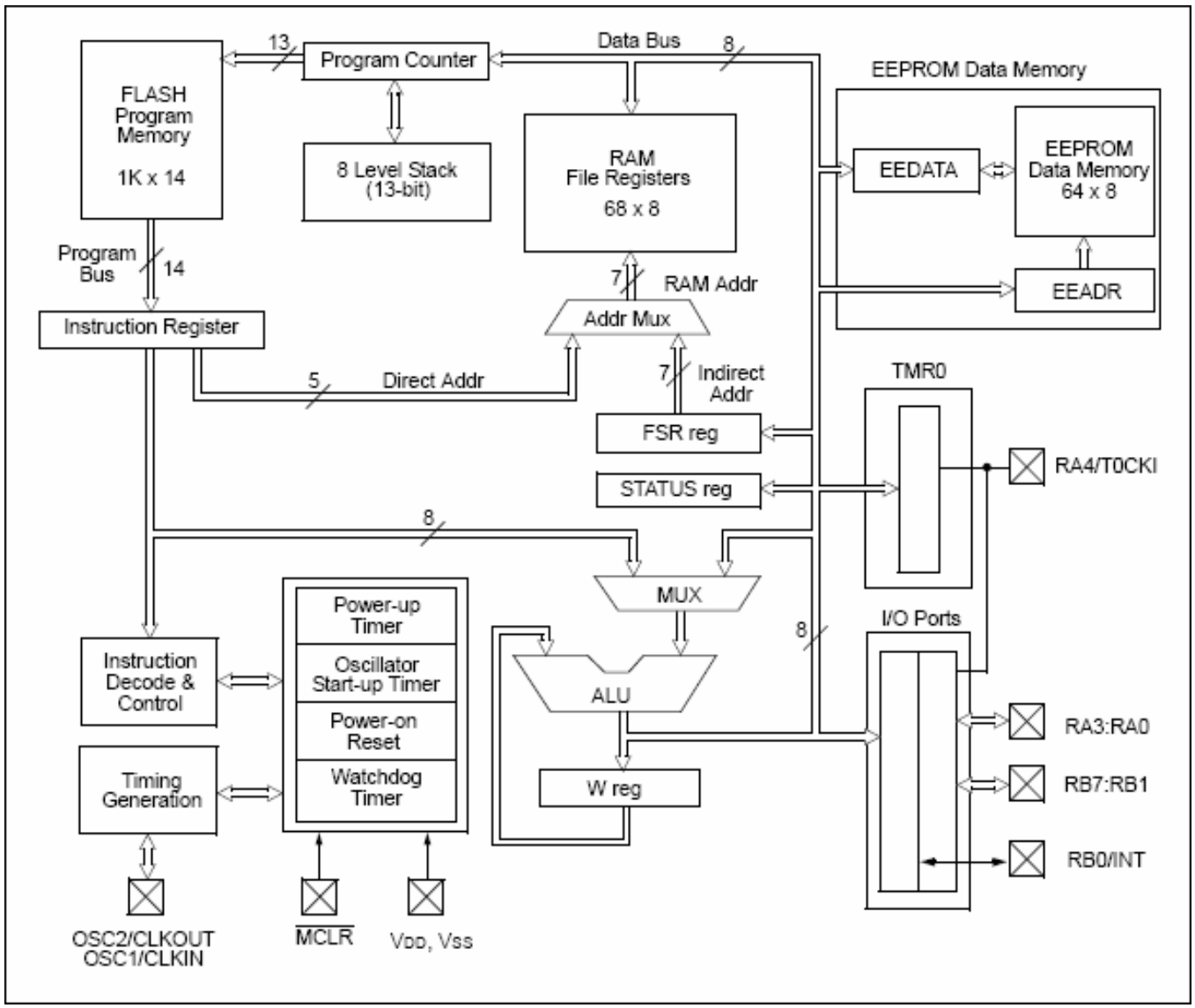


Figura 3.20 Diagrama de Bloque del 16F84A.

3.4.3 Programando los microprocesadores “PIC”

En un microcontrolador es importante el soporte tanto en software como en hardware y las herramientas de desarrollo son fundamentales en la decisión de cual utilizar, ya que ello impacta el desarrollo de un proyecto.

Para comprender las fases de implementación de la programación del microcontrolador, estas se muestran en el siguiente diagrama de flujo:

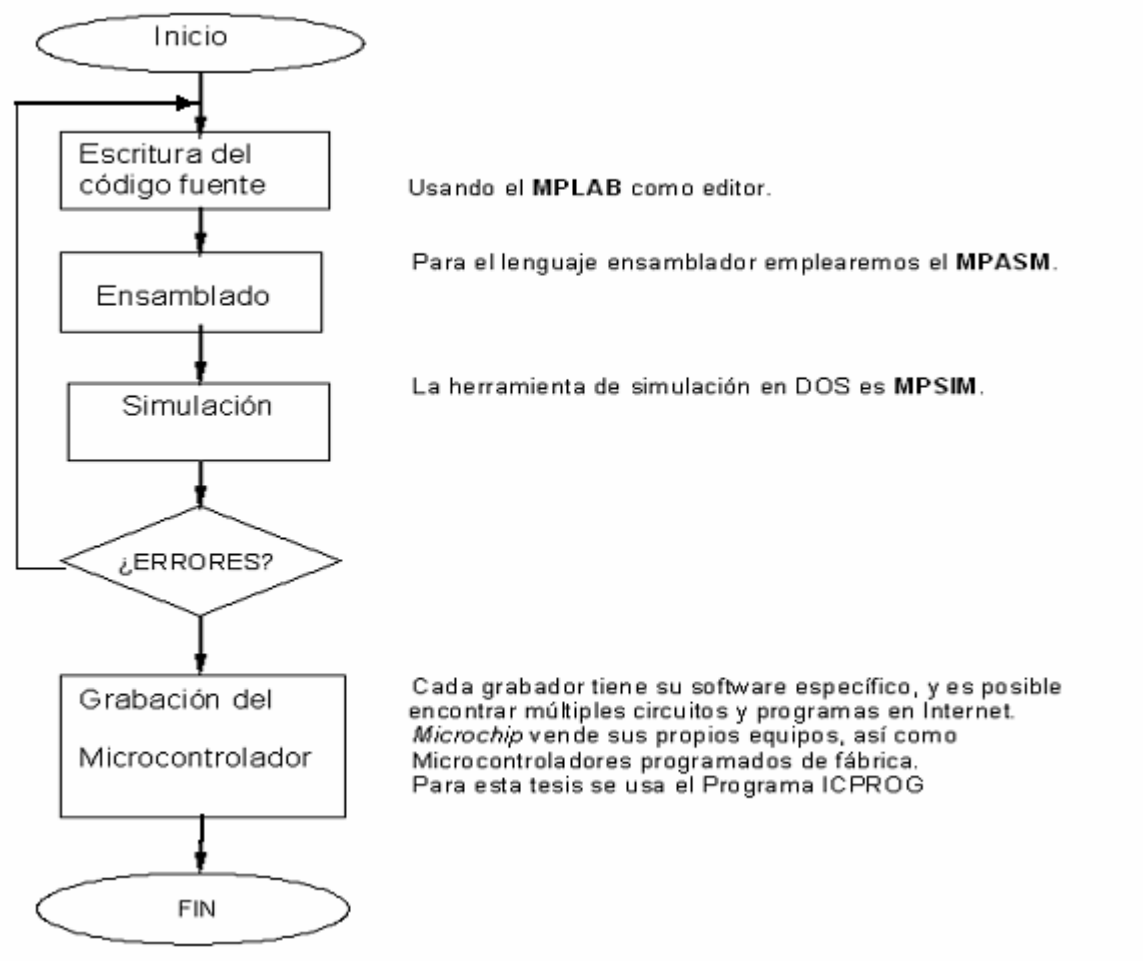


Figura 3.21: Organigrama de las Fases de Implementación.

3.5 MPLAB[8]

MPLAB es la herramienta más accesible con la que se cuenta para escribir programas, además se puede descargar gratuitamente de la página de Internet de la Compañía Microchip y funciona perfectamente en ambiente Windows.

El entorno de MPLAB es de tipo contenedor, es decir que sus distintas opciones se asocian a programas, que serán ejecutadas cuando se les indique.

[8] < http://platea.cnice.mecd.es>(Marzo/2005).

En la figura 3.22 se muestra la vista inicial del programa MPLAB, confirmando su entorno contenedor, y del cual se da más información en el *anexo 2*.

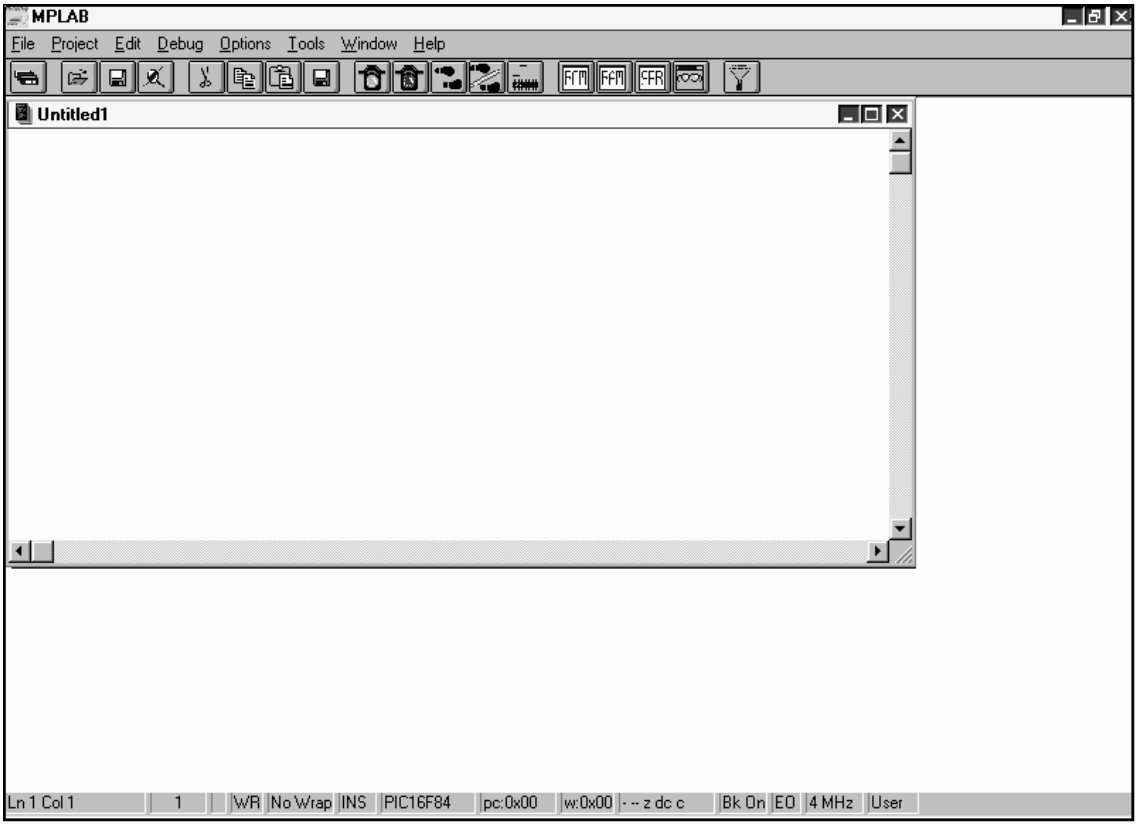


Figura 3.22 Vista inicial de MPLAB.

3.5.1 El ensamblador[9].

El lenguaje de máquina es difícil de utilizar por el hombre ya que se aleja de la forma natural de expresarse, por eso se prefiere el lenguaje ensamblador, que es la forma de expresar las instrucciones de una forma más sencilla, sin embargo, es muy cercana al microcontrolador porque cada una de las instrucciones corresponde con otra en código de máquina que el microcontrolador es capaz de interpretar.

[9] <http://www.physics.brandeis.edu/phys32b_2005/pic16f84A.pdf>(Abril/2005).

A continuación se presentan las partes que generalmente hay en un programa:

- **Modelo de procesador y Sistema de Numeración:**

Los programas comienzan con la directiva ***list*** que referencia al modelo de microcontrolador. También se suele especificar el tipo de numeración que se empleará con la directiva ***radix,*** Usando el sistema hexadecimal donde los valores se expresan precedidos de “0x”.

- ***Variables.***

Las posiciones de la memoria de datos se utilizan para guardar operandos y resultados, además de almacenar registros especiales.

Cuando se realiza el programa, en lugar de hacer referencia a las posiciones de la memoria donde se encuentran los datos que va a emplear, a cada una de estas posiciones se le asocia un nombre. La directiva *equ* relaciona un nombre con la dirección que se asigna.

- ***Origen del programa.***

Antes de escribir el programa se debe direccionar la memoria donde se va a comenzar a cargar el programa. Para esto se utiliza la directiva *org.* En los PIC el origen del programa se pone en la dirección 0x00 porque es donde comienza a ejecutarse el programa después de hacer un reset.

Para el caso de interrupciones, el programa no comienza desde 0x00, porque si se genera una interrupción el programa que la atiende es la dirección 0x04 (vector de interrupción).

- ***Cuerpo del programa.***

Indicando la dirección donde comienza a cargar el programa, sigue el cuerpo del mismo compuesto por las instrucciones de máquina y los operandos de éstas.

El código se estructura en columnas. La primera columna se utiliza para etiquetar referencias a partes del programa y poder dar saltos a otras partes que nosotros indiquemos.

Las siguientes columnas contienen el campo de instrucciones, el campo de datos y el campo de comentarios. Los comentarios comienzan con ;).

Al finalizar el programa se coloca la directiva *end*.

El ensamblador exige cierta tabulación mínima de sus distintos elementos, por lo que la definición de variables se escribe en la 1ª columna de cualquier línea, mientras que las directivas e instrucciones en la 2ª columna, como mínimo.

Las cifras se expresan de acuerdo a la siguiente tabla:

BASE	REPRESENTACIÓN
DECIMAL	d'12'
HEXAGESIMAL	0x0c / h'0c' / 0c / 0ch
BINARIO	b'1010'

Figura3.23 Tabla de cifras.

El uso de mayúsculas y minúsculas en este código, obedece a la siguiente serie de reglas:

- Directivas del compilador en mayúsculas.
- Nombres de variables en minúsculas.
- Nemónicos (instrucciones) en mayúsculas.
- Programa bien tabulado.

3.6 GRABACIÓN DE UN MICROCONTROLADOR

Un microcontrolador es un circuito integrado programable que contiene todos los componentes necesarios para ejecutar en forma ordenada una tarea determinada como el control de un teclado de ordenador, una impresora, un sistema de alarma, una lavadora, etc. El microcontrolador dispone de una memoria de programa donde almacena el programa en números hexadecimales.

El programa de control se graba en la memoria del programa mediante un equipo físico denominado grabador o programador. El grabador se conecta a un ordenador normalmente a través de un puerto serie COM 1 ó COM 2. En el ordenador se ejecuta el software que controla la grabación de la memoria de programa del microcontrolador. Este proceso se denomina grabar o programar el microcontrolador, el cual se detalla en el *anexo 3*.

3.6.1 Grabador

El grabador o programador –como ya se comentó- es el equipo físico donde se procede a grabar la memoria del microcontrolador con las instrucciones del programa de control. Tiene un zócalo libre sobre el que se inserta el circuito

integrado a grabar, el cual debe orientarse adecuadamente siguiendo la señal de la cápsula del chip. Hay multitud de grabadores comerciales que se pueden adquirir en cualquier tienda de electrónica.

La marca Microchip ofrece el grabador PICSTART PLUS, de muy fácil utilización y respaldado por el fabricante.

Para este cuadrúpedo se usa uno de los más populares, el JDM del diseñador Jens Dyekjaer.

3.6.2 Software de programación IC-Prog

IC-Prog es uno de los softwares más populares para la grabación de microcontroladores PIC. Permite la programación de muchos dispositivos y esta probado con numerosos programadores entre ellos el JDM.

Este programa se adquiere de manera gratuita por internet; una vez que se descarga a la computadora, el IC-Prog tiene el código necesario para su funcionamiento, con versiones para cualquier sistema operativo Windows.

3.6.3 Proceso de grabación

Lo primero que se tiene que hacer es conectar el programador con el PIC en el zócalo a uno de los puertos serie COM de la computadora. Una vez que el programa esta instalado, los pasos a seguir para trabajar el IC-Prog según el libro de “Microcontroladores 16F84” son los siguientes:

Paso 1: Iniciar el programa ejecutando icprog.exe o pulsando el icono correspondiente.

Paso 2: Seguidamente aparece una pantalla en ingles donde se presenta toda la información necesaria para programar el dispositivo. Esta pantalla posee al menos:

- Un área de código (Program Code), donde se almacena la información a grabar. La columna de la izquierda contiene la dirección física de memoria del dispositivo, (Address). En el centro del campo se presenta el valor hexadecimal y la columna de la derecha contiene la misma información en código ASCII.

- Un área de configuración (Configuration), donde se indica el valor de algunos parámetros necesarios para la correcta grabación.

Paso 3: Para cambiar el idioma se debe seleccionar en el menú *Setting* > *Options* > *Language* y elegir el idioma.

Paso 4: Configurar el hardware para programar al PIC, adaptando al IC-Prog al programador utilizado, para el JDM en particular debemos acceder al menú *Ajustes* > *Tipo hardware* y elegir el tipo de programador como JDM y seleccionar el puerto serie adecuado (COM 1 ó COM 2), según lo tenga conectado al ordenador.

Paso 5: Seleccionar el dispositivo a grabar, en este caso el microcontrolador PIC 16F84A, en el menú Ajustes > Dispositivo > Microchip PIC > Mas > PIC16F84A.

Paso 6: Elegir el oscilador que va a utilizar el microcontrolador en cuestión (LP, RC, XT, HS). Para ello en la ventana Oscilador se elige el tipo XT (oscilador a cristal de cuarzo), que es el empleado en esta tesis.

Paso 7: Activar los bits de configuración, que permiten seleccionar varias configuraciones de los dispositivos. En IC-prog se muestran tres:

- WDT (del Inglés Watchdog Timer). Se deshabilita.

- PWRT (del Inglés Power-up Timer). Temporizador al encendido. En aplicaciones sencillas se activa.

- CP (del Inglés Code Protect). Protección de código de programa. Cuando se programa la protección de código del programa no se puede copiar, ni alterar, aunque si se puede volver a borrar completamente todo el microcontrolador. En aplicaciones sencillas se puede deshabilitar.

Paso 8: El IC-Prog ya esta en condiciones de proceder a la grabación de datos en el dispositivo insertado en el programador. Para ello, en la pantalla de edición se escriben los datos del programa de control a grabar.

Paso 9: Para proceder a la grabación del chip basta con activar el menú *Comando* > *Programa todo* o bien pulsar la tecla función F5. También puede pulsar sobre el icono correspondiente de la barra de herramientas (rayo sobre el chip). Entonces el chip comienza a ser programado con los datos cargados en el buffer activo.

Paso 10: El proceso de grabación se ira mostrando. El tiempo empleado en la grabación del PIC16F84A dependerá de la rapidez de la computadora con la que se este trabajando.

Paso 11: Una vez terminada la programación se procede automáticamente a la verificación de los datos escritos en el chip, informando de este proceso con una pantalla. De este modo se asegura que la programación del dispositivo haya sido efectuada correctamente.

Paso 12: Una vez grabado el PIC 16F84A se debe extraer del programador y comprobar su correcto funcionamiento dentro del circuito correspondiente.

Paso 13: Los datos grabados en el microcontrolador y la configuración se pueden salvar a un fichero utilizando el procedimiento usual de Windows mediante la selección del menú: Archivo > Guardar como y poniendo al archivo un nombre con extensión *.bin.

BIBLIOGRAFÍA ESPECÍFICA

- Capítulo III -

<http://www.iearobotics.com/proyectos/libro6811/libro-6811.pdf>

<http://www.otherlandtoys.co.uk/product529/product_info.html?name=Meccano%20Best%20of%2040%20Set&&prod=529>

< http://www.personal-robotics.de/personal-robotics/fischertechnik_computing.htm>

<http://platea.cnice.mecd.es>

<http://www.iearobotics.com/proyectos/libro6811/libro-6811.pdf>

<http://www.me.umn.edu/education/courses/me2011/robot/technotes/L293/L293.html>

<http://www.creaturoides.com/anterior/puentesh.htm>

<http://www.mabuchi-motor.co.jp/cgi-bin/catalog/e_catalog.cgi?CAT_ID=fa_280rasa >

<http://www.physics.brandeis.edu/phys32b_2005/pic16f84A.pdf>

CAPÍTULO IV

4 DISEÑOS ELECTRÓNICO Y MECÁNICO

En este capítulo se detalla la construcción del cuadrúpedo experimental, a partir del cual se explica básicamente su diseño, mismo que esta formado por los diferentes módulos componentes tanto electrónicos como mecánicos y de control.

4.1 MÓDULOS COMPONENTES

Enfocándose en los componentes del cuadrúpedo unidireccional y basados en la información de los capítulos II y III, la figura 4.1 muestra el diagrama a bloques que lo constituyen y que principalmente forman los diferentes módulos: el electrónico, el mecánico y de control; los cuales son descritos en este capítulo, así como la solución en el problema de diseño planteado.

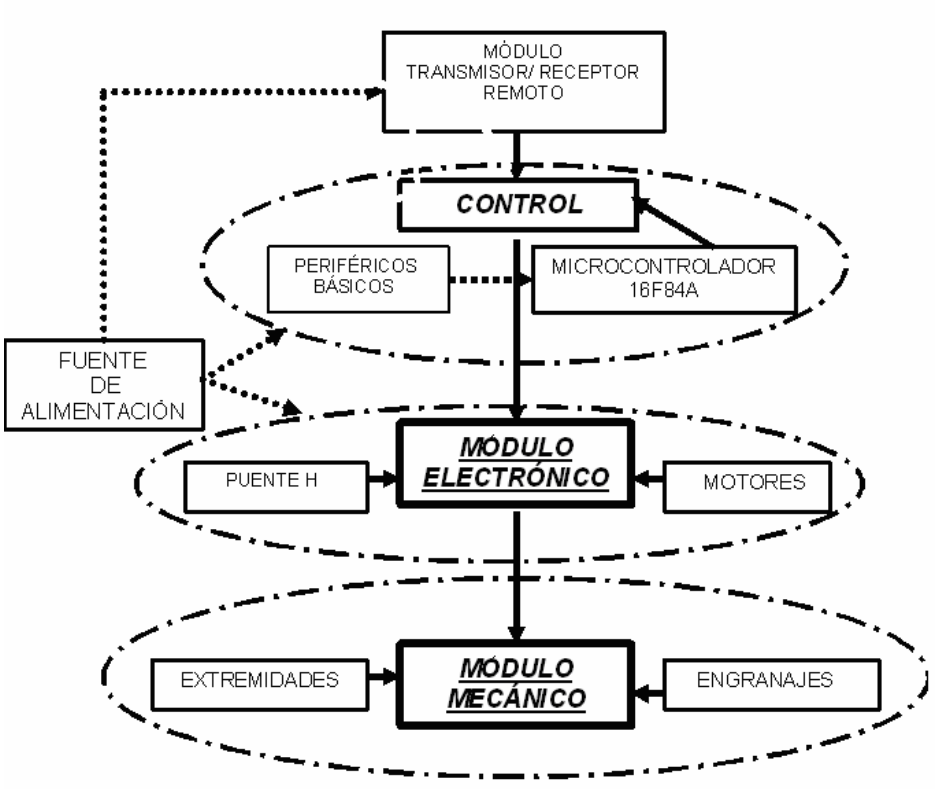


Figura 4.1 Módulos Componentes del Cuadrúpedo Unidireccional.

4.2 FUENTES DE ALIMENTACIÓN

La fuente de alimentación a los sistemas de control, mecánicos y electrónicos es sustentada mediante 4 baterías alcalinas de 9 Volts colocadas en la estructura del robot, y una batería adicional de 9 Volts en la unidad de mando. Para proveer la energía a los circuitos electrónicos situados en la estructura del robot es necesario que su tensión sea regulada, independientemente de que sea suministrada por las baterías, con la finalidad de asegurar una solución confiable y robusta en este sentido. Para la sección de entrada, 4 baterías de 9 Volts en 2 pares, cada par conectado en serie y estos se juntan en conexión paralela, considerándolo una alternativa al requerimiento de los motores. Para la sección de regulación con la misión de mantener la salida en los valores prefijados, se plantean en esta tesis el uso de 2 tensiones (9 Volts para los motores y 5 Volts para los circuitos integrados), por lo que 2 reguladores de voltaje, el 7809 y 7805, son una muy buena solución.

La siguiente figura muestra los dos reguladores: el 7809 que da una tensión de salida de 9Volts e inmediatamente el 7805 que da una tensión de salida de 5Volts.

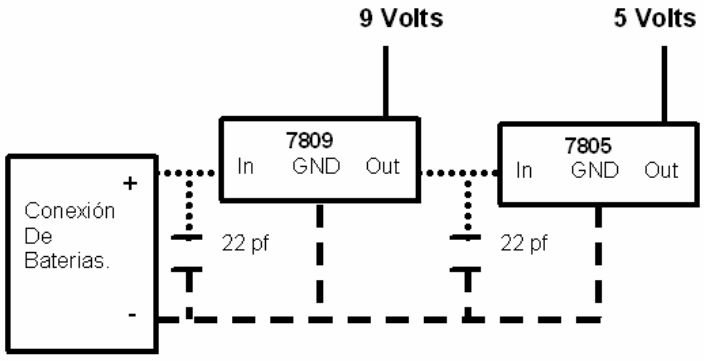


Figura 4.2 Reguladores de Voltaje.

4.3 MÓDULO TRANSMISOR / RECEPTOR REMOTO

Se establece el uso de un módulo transmisor/receptor (de radio frecuencia) para generar el movimiento del cuadrúpedo, evitando así el uso de cableado que comunique al robot con la unidad de mando. Los módulos transmisor/ receptor no deben ser voluminosos, es necesario considerar el espacio que hay en la estructura del robot cuadrúpedo como una de las limitantes del diseño.

La unidad remota tiene la finalidad de dar la secuencia Inicio/Paro de movimiento, la cual se diseña de la siguiente forma: Al momento de presionar el botón del transmisor de radio frecuencia, se genera un pulso (comunicación) que es captado por el receptor (este es colocado en el interior del robot cuadrúpedo), emitiendo el pulso que inicia el programa de movimiento para el microcontrolador.

En respuesta a la necesidad de una unidad de mando con las características mencionadas y vistas en el capítulo anterior, se utiliza un trasmisor y un receptor más complejos para evitar que las comunicaciones sean sólo para distancia cortas. De esta forma se usa el transmisor **TWS-BS3** y el receptor **RWS-371-6** que cumplen con especificaciones de largo alcance, así como la sencillez de su uso y diminuto tamaño.

Estos componentes utilizan radio frecuencia (RF) a 433.92 Mhz, con modulación ASK, que pueden ser usados en alarmas para vehículos, sistemas de seguridad, teléfonos inalámbricos, control de robots y otros sistemas de control remoto.

4.3.1 Transmisor

El TWS-BS3 tiene una potencia de salida de hasta 8 mW con una ultra alta frecuencia (conocidas por sus siglas UHF), de 433.92MHz, alcanzando distancias de aproximadamente 140 metros en espacios abiertos y de 60 metros en espacios cerrados donde se tengan obstáculos.

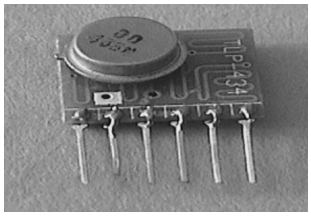


Figura 4.3 Módulo TWS-BS3.

El transmisor TWS-BS3 acepta tanto señales lineales como digitales de entrada y puede operar con una tensión que va desde 1.5Volts hasta 12Volts-DC. La disposición y función de cada pin de este módulo se muestra en el gráfico inferior.

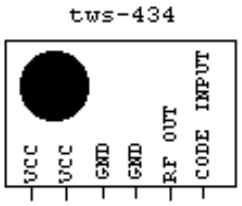


Figura 4.4 Pin Out del TWS-BS3.

El codificador HT-12E de Holtek, es un integrado que se utiliza en controles remotos de 4 bits y tiene 8 bits de direcciones.

En la figura 4.5 se muestra el circuito transmisor que se usa en este diseño. Se usa el bit AD8, de 4 que tiene el transmisor y por medio de un pulsador, este activa el bit D8 del receptor. Los pines A0-A7, son canales que representan receptores con 4 bits, es decir se pueden controlar 8 receptores con un solo transmisor. De estos 8 bits se trabaja con el bit A0, pues es el canal del receptor a activar. Esta característica puede ser usada para controlar por ejemplo: un equipo de 4 robots con un solo transmisor.

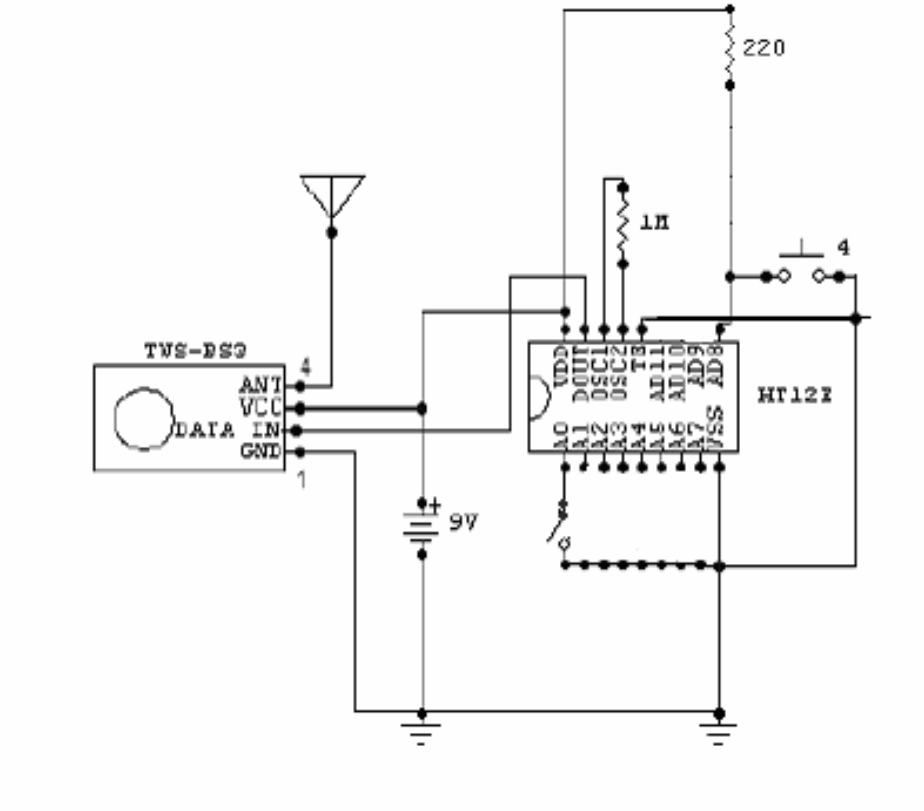


Figura 4.5 Ejemplo de Transmisor de 1 Bit.

4.3.2 Receptor

El RWS-371-6 es un módulo receptor que opera con una frecuencia ultra alta (UHF), de 433.92MHz, y tiene una sensibilidad de 3 µVolts. El receptor RWS-371-6 opera con una alimentación entre 4.5Volts hasta 5.5Volts-DC y tiene tanto salida lineal analógica como digital, además contiene un capacitor variable para el ajuste de la frecuencia de recepción utilizando un destornillador plástico como se aprecia en la fotografía.

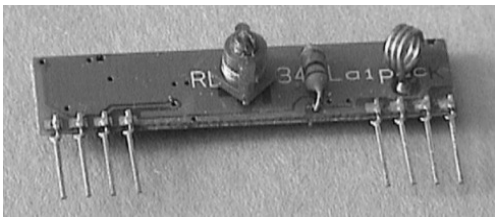


Figura 4.6 Módulo RWS-371-6.

La disposición y función de cada pin de este módulo es la siguiente:

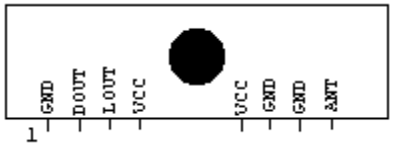


Figura 4.7 Pin Out RWS-371-6

El decodificador HT-12D de Holtek, es un integrado que se utiliza en controles remotos de 4 bits y tiene 8 bits de direcciones.

En la figura 4.8 se muestra el circuito receptor que se utiliza en este diseño que aprovecha un bit D8, de 4 que tiene el receptor y por medio de un LED se confirma la transmisión. El bit D8 se conecta al PIN RA4 del microcontrolador que activa la secuencia de avance. Los pines A0-A7, son canales que representan receptores que tienen 4 bits. De estos 8 bits sólo se usa el bit A0, el cual es el canal que activa al transmisor.

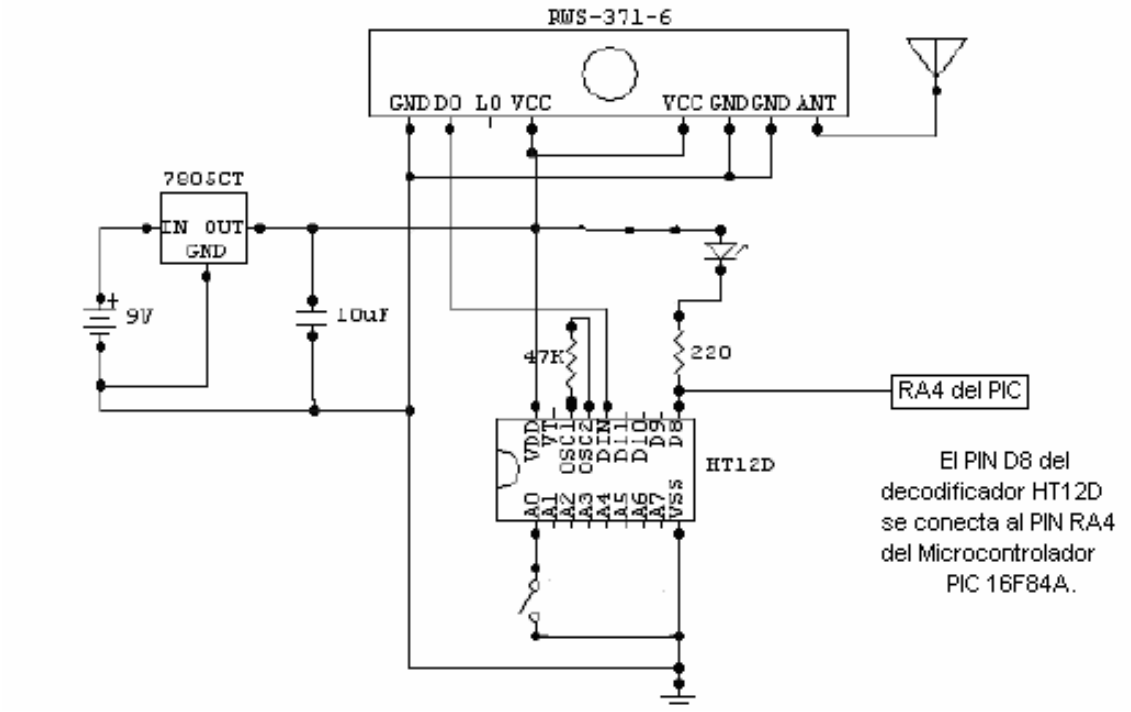


Figura 4.8 Diagrama del Receptor de 1Bit.

De esta manera queda implementada la unidad de transmisión/recepción para que el cuadrúpedo pueda iniciar el movimiento unidireccional.

4.4 MÓDULO DE CONTROL

Una vez que el receptor emite el pulso que inicia el programa del movimiento, el microcontrolador activa la subrutina de secuencia de movimiento y cuando el receptor deje de transmitir el pulso antes mencionado, el microcontrolador es capaz de arrancar la secuencia de paro de movimiento, todo esto como parte del mismo programa. Para la realización del control se utiliza el PIC 16F84A, el cual se describe a continuación.

4.4.1 Microcontrolador 16F84A

El microcontrolador PIC 16F84A es muy popular, esta encapsulado en un económico DIL (Zócalo) de 18 pines. Debido a sus múltiples aplicaciones y facilidad de uso es uno de los microcontroladores mas considerado como herramienta electrónica de control, en la actualidad.

El 16F84A trabaja con una frecuencia máxima de 10 Mhz y se energiza con una tensión de 5Volts aplicados entre los pines V_{dd} y V_{ss} que son, respectivamente, la alimentación y la tierra del chip, tal y como se observa en la siguiente figura:

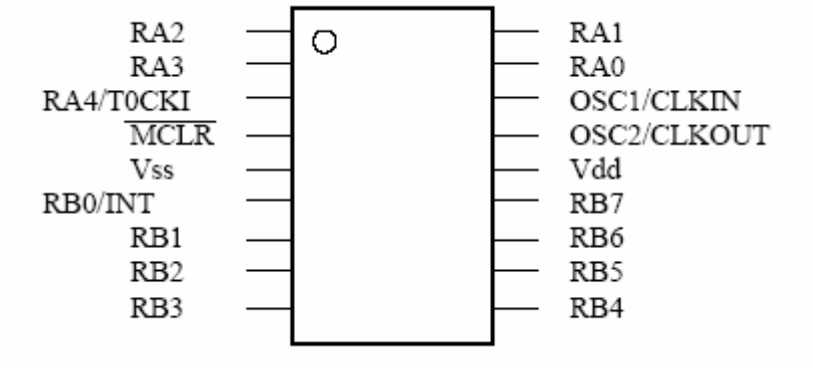


Figura 4.9 16F84A con encapsulado Zócalo de 18 pines.

4.4.2 Puertos de entrada y salida

El microcontrolador se comunica con el mundo exterior a través de los puertos. Los puertos se pueden configurar como entradas para recibir datos o como salidas para gobernar dispositivos externos.

El PIC tiene 2 puertos:

- El Puerto A con 5 líneas, pines RA0 a RA4
- El puerto B con 8 líneas, pines RB0 a RB7

Cada una de estas líneas puede ser configurada como entrada o como salida, independientemente unas de otras, según se programe.

4.4.3 Oscilador

Todo microcontrolador requiere de un circuito que le indique la velocidad de trabajo, este es el llamado oscilador ó reloj. Este genera una onda cuadrada de alta frecuencia de 10 Mhz que se utiliza como señal para sincronizar todas las operaciones del sistema. Este circuito es simple pero de vital importancia para el buen funcionamiento del sistema. Generalmente todos los componentes del reloj se encuentran integrados en el propio microcontrolador y tan solo se requieren unos pocos componentes externos, como un cristal de cuarzo o una red RC (resistor-capacitor), para definir la frecuencia de trabajo.

En el PIC 16F84A, los pines OSC1/CLKIN y OSC2/CLKOUT son las líneas utilizadas para este fin. Permite cinco tipos de osciladores para definir la frecuencia de funcionamiento:

- XT: Cristal de Cuarzo
- RC: Oscilador con resistencia y condensador

- HS: Cristal de alta velocidad
- LP: Cristal para baja frecuencia y bajo consumo de potencia
- Externa: Cuando se aplica una señal de reloj externa

4.4.4 Periféricos básicos

Por lo regular, en las salidas de los puertos de los microcontroladores se colocan LEDS para verificar que el puerto esta funcionando y en las entradas de los puertos se usan interruptores que están dentro de la clasificación de sensores, vistos en el capítulo anterior.

4.4.5 DIODO EMISOR DE LUZ

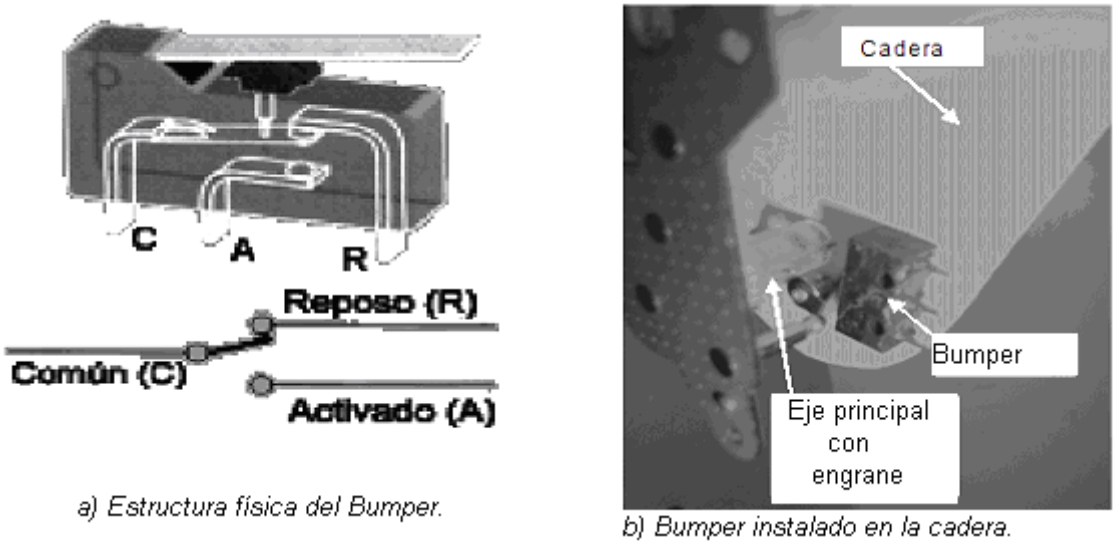
El diodo emisor de luz o LED (del Inglés Light Emited Diode) es un dispositivo que permite comprobar el funcionamiento de los circuitos de forma cómoda mediante la emisión de luz. Es barato y fácil de conectar a la salida de un microcontrolador. Se polariza directo con una tensión en extremos entre 1.2Volts y 2.2Volts según el modelo, y sólo requiere de una corriente de 5mA a 30mA para su encendido.

4.4.6 Sensores

En el capítulo anterior se mencionaron diferentes tipos de sensores que se utilizan en el diseño de robots; para este caso se usan “BUMPERS” que nos permiten introducir un nivel lógico 0 ó 1 según la posición en que se encuentren, cerrado o abierto, respectivamente.

El BUMPER es un conmutador de 2 posiciones con muelle de retorno a la posición de reposo y con una palanca de accionamiento más o menos larga según el modelo elegido, siendo la figura 4.10 donde se muestra físicamente, y se destacan las partes que lo componen.

En estado de reposo la terminal común (C) y la de reposo (R) están en contacto permanente hasta que la presión aplicada a la palanca del BUMPER hace saltar la pequeña pletina acerada interior y entonces el contacto pasa de la posición de reposo a la de activo (A), se puede escuchar cuando el BUMPER cambia de estado, porque se oye un pequeño clic, esto sucede casi al final del recorrido de la palanca.



a) Estructura física del Bumper.

b) Bumper instalado en la cadera.

Figura 4.10 Bumper.

En el PIC debe haber un vínculo con los sensores para que se interprete la posición Inicio/Paro. En el caso de posición de Inicio, la señal generadora del receptor remoto se concentra en una entrada del puerto A (señal de entrada conocida como RA4), habilitando en el puerto B, 4 salidas del microcontrolador (señal de salidas conocidas como RB0, RB1, RB2, RB3) donde cada una emite un pulso (verificable con un LED), con una secuencia de tiempo definido que al pasar por la etapa de potencia podrá activar el motor.

En el caso de la posición de paro, -donde no hay señal Inicial del receptor remoto-, se deja de pulsar el botón del transmisor remoto y habilita las 4 salidas del puerto B (RB0, RB1, RB2, RB3), que pondrán en marcha por la etapa de potencia a los motores. El freno de cada motor es producido por 4 BUMPERS,

concentrados en 4 entradas del puerto A (conocidas como RA0, RA1, RA2, RA3). Cada BUMPER se coloca en el eje principal del mecanismo, controlando la extremidad, y en el momento que se activa el BUMPER, el microcontrolador deja de mandar su señal por la salida del puerto B parando al motor cuando alcanza la misma posición inicial

La colocación de los sensores así como la conexión de este con el PIC se observa detalladamente en el siguiente diagrama:

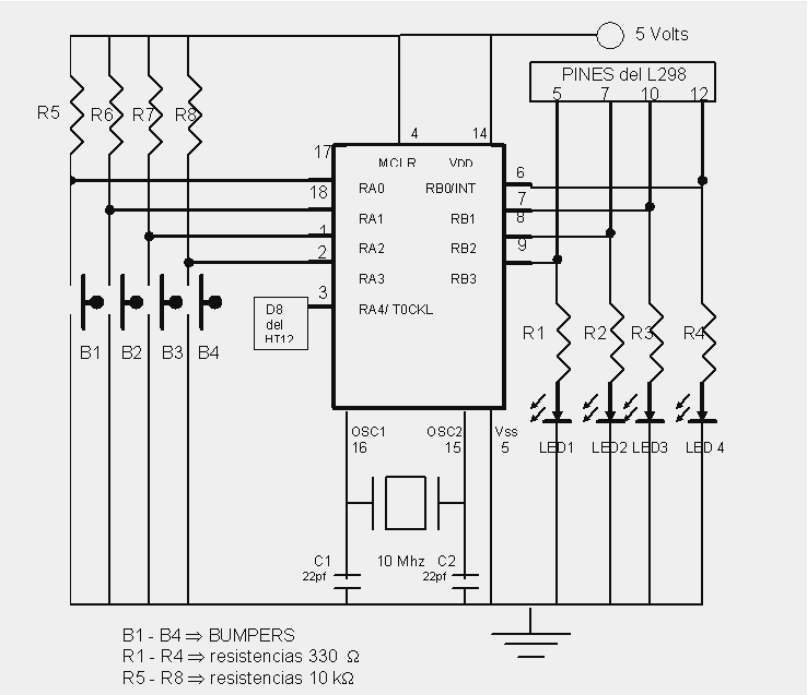


Figura 4.11 Diagrama esquemático del PIC 16F84A.

Para la implementación del programa de control de Inicio/Paro del movimiento se desarrolló el diagrama de flujo que muestra la secuencia y manera de obtener el código de programación para el microcontrolador.

4.4.7 Diagrama de flujo de Inicio/ Paro

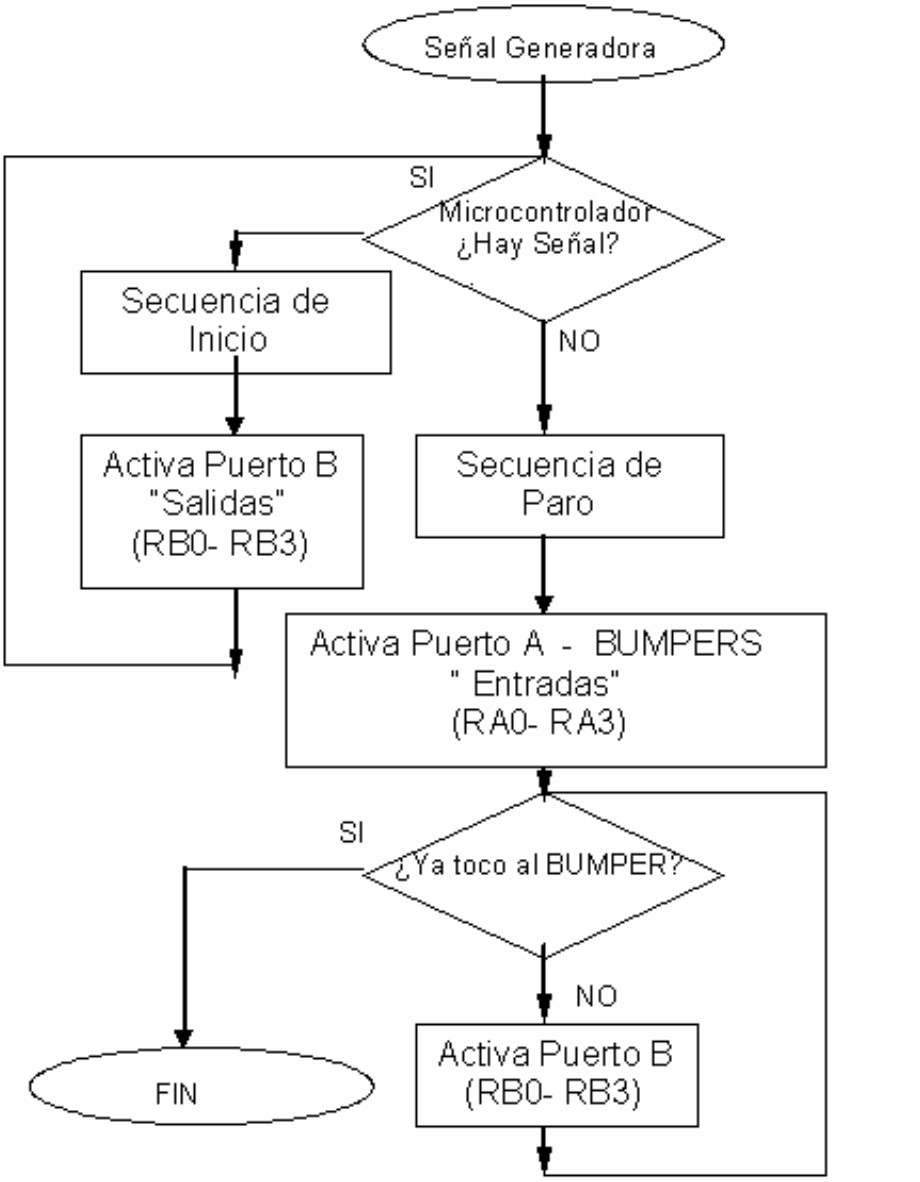


Figura 4.12 Diagrama de flujo: Posición Inicio/ Paro.

Al obtener el código que realiza la subrutina de Inicio/Paro, el microcontrolador elije si se activa la secuencia de Inicio o lo secuencia de Paro

dependiendo de la señal recibida. El código fuente activa la secuencia de movimiento que se muestra a continuación.

```
;*********************************Pulsador_Armagedon12.asm*************************************************************
*******
;
;Cada vez que presione el pulsador conectado en el pin RA4 se activara la secuencia de inicio y cuando no se presione
el pulsador
; se activara la secuencia de paro.
;
;ZONA DE CÓDIGOS*************************************************************************************************************

                ORG  0
Inicio

                bsf  STATUS,RP0 ;Acceso al Banco 1.
                bcf  LED1
                bcf  LED2
                bcf  LED3
                bcf  LED4
            bsf  Pulsador

                bcf  STATUS,RP0;Acceso al Banco 0.

Principal


                btfsc  Pulsador                 ; ¿Pulsador presionado?, ¿(Pulsador)=0?
                goto   INICIO             ; No. Vete a la subrutina Independencia ( Prender RB1)
                btfsc  Pulsador                 ; Comprueba si es un rebote.
                goto   Fin
Encendido
             btfss  Pulsador                                    ;¿Dejo de pulsar?, ¿(Pulsador)=1?
             goto   PARO            ; No. Vete a Independencia

Fin     goto  Principal
```

El programa Inicio/Paro conlleva a los movimientos del motor. Al alimentar un motor de corriente continua, se puede notar que la velocidad de este es constante dependiendo del voltaje de alimentación. En este diseño el motor CC esta conectado a una caja de engranes, para mantener diferentes velocidades generando torques que amortigüen las extremidades una vez que tenga un obstáculo; por esta razón, se recurre a la técnica PWM aplicable en casos de velocidades variables.

4.4.8 PWM

El PWM significa modulación de ancho de pulso (del Inglés Pulse Width Modulation). Una forma muy sencilla de entenderlo es por medio de un LED para ver su funcionamiento que simula la regulación de la velocidad de un motor.

Si se prende y se apaga la señal de alimentación de la carga con pulsos iguales (LED o motor) el microcontrolador tendrá que dar la señal que se ilustra enseguida:

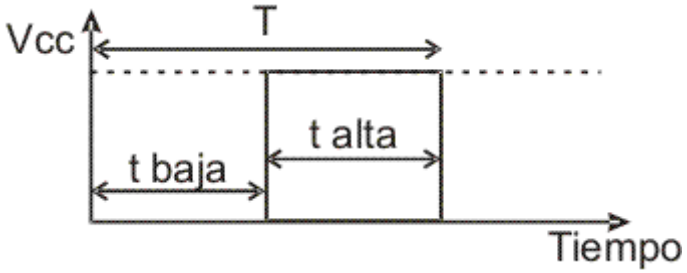


Figura 4.13 LED con 50% de eficiencia.

El esquema anterior representa un pulso con un ciclo de trabajo (*del Inglés duty cycle*), donde el motor tendrá velocidad constante y una eficiencia del 50%, en caso que se requiera mayor eficiencia los pulsos tendrán que ser diferentes.

La técnica PWM soluciona este problema de tiempo. La forma de lograrlo es dejar el pulso fijo en amplitud y variar sus tiempos de activación del pulso y desactivación de este.

Lo que se hace con PWM es variar dinámicamente el ciclo de trabajo, de manera que el tiempo de alta disminuya o aumente y en proporción inversa, el de baja aumente o disminuya dependiendo de si queremos un LED más atenuado o más brillante, o un motor más lento o más rápido, respectivamente.

En las figuras siguientes se muestran las variaciones del ciclo de trabajo con un motor más lento y otro más rápido.

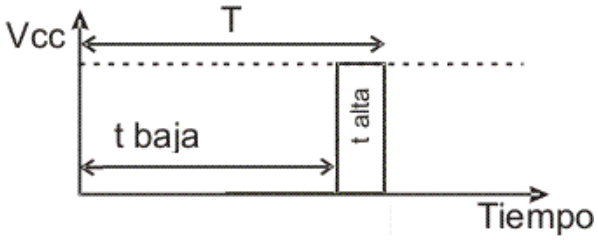


Figura 4.14 Motor mas lento.

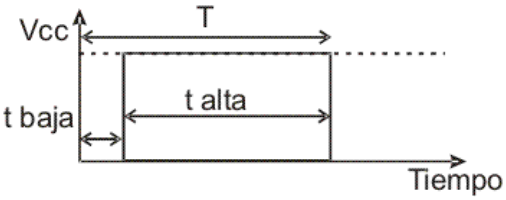


Figura 4.15 Motor mas rápido.

En el caso del motor habrá que determinar el ciclo de trabajo empíricamente ya que depende de sus características eléctricas y mecánicas, pero su periodo se repite con un tiempo definido, como se muestra abajo:

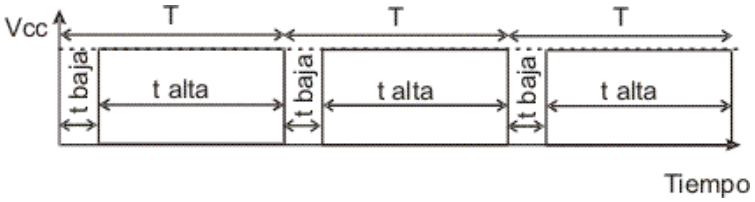


Figura 4.16: Periodo con tiempo definido.

De esta forma tenemos 2 fórmulas muy sencillas para el cálculo de los tiempos:

1) Si variamos, fijamos o controlamos t alta entonces: t baja= T- t alta
2) Si variamos, fijamos o controlamos t baja entonces: t alta= T- t baja

Y el ciclo de trabajo lo calculamos así:

DT= t alta / (t alta + t baja)

La señal de salida del PWM es generada por el microcontrolador 16F84A. El siguiente diagrama de flujo muestra la implementación del PWM.

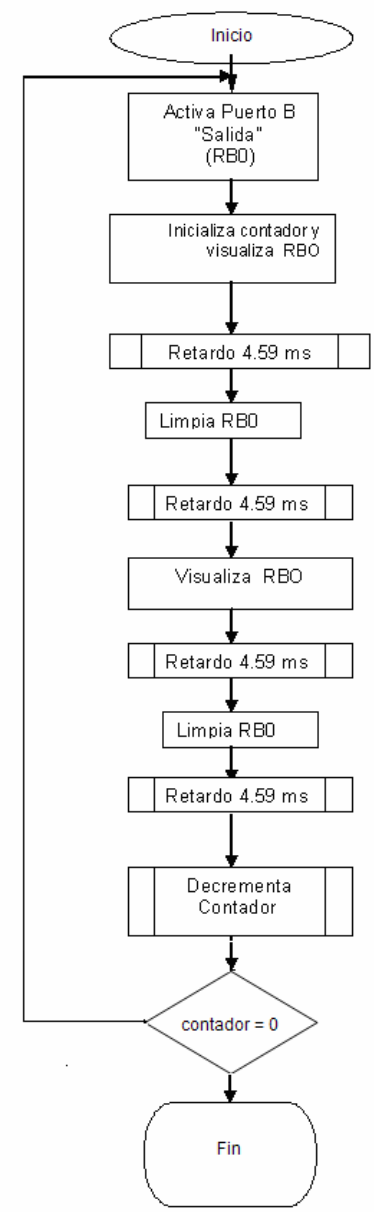


Figura 4.17 Diagrama de flujo que genera el PWM en un motor.

Utilizando el diagrama de la figura 4.18 para obtener el código que realizará la subrutina del PWM se requiere de un programa de la misma forma como se implementó para los sensores. El código fuente utilizado aquí y que genera la secuencia de PWM en un motor, se nombró cuadrito y se muestra a continuación:

```
;Subrutina  " movimiento de motor mediante técnica PWM"----------
;
Independencia

Cuadrito
            movlw  9f
                   movwf  reg1
uno    bsf    LED1

                   call   Retardo_2ms
                call   Retardo_2ms
                   call   Retardo_500micros
                   call   Retardo_20micros
                   call   Retardo_20micros

                   bcf    LED1

                call   Retardo_200micros
                call   Retardo_200micros
                   bsf    LED1

                call   Retardo_2ms
                   call   Retardo_2ms
                   call   Retardo_500micros
                   call   Retardo_20micros
                   call   Retardo_20micros


                   bcf    LED1

                call   Retardo_200micros
                call   Retardo_200micros
    decfsz  reg1,1
                   goto   uno
                   call  hola
                   call    Circulito
```

Figura 4.18 Código fuente del programa para su implementación en el microcontrolador.

4.5 MÓDULO ELECTRÓNICO

Mediante la generación del PWM con el microcontrolador tendremos los pulsos para el movimiento de los motores. En general cuando se coloca directamente la salida del 16F84A al motor no se cuenta con la corriente necesaria para activar los motores, de tal forma que se recurre a una etapa de potencia, que para este caso usa un puente H que también funciona como una etapa de aislamiento entre el 16F84A y los motores. Se utilizara un circuito integrado de 15 PINES llamado L298.

4.5.1 Puente H

La interfaz de potencia para motores de corriente continua, Puente H, es un sistema que permite controlar motores con una tensión máxima de 50 Volts y con consumos de corriente de 2.5 Amperes por medio de señales de baja potencia provenientes de un circuito digital (16F84A).

El circuito integrado de 15 pines llamado L298 tiene 2 Puentes H y se pueden conectar 2 motores cada uno con movimiento en los dos sentidos (configuración direccional). En nuestro caso colocamos los 4 motores al puente (configuración unidireccional), permitiéndoles girar en una sola dirección, (configuración necesaria para este trabajo); quedando entonces como lo muestra el diagrama de bloques del L298, y la conexión de este con el PIC.

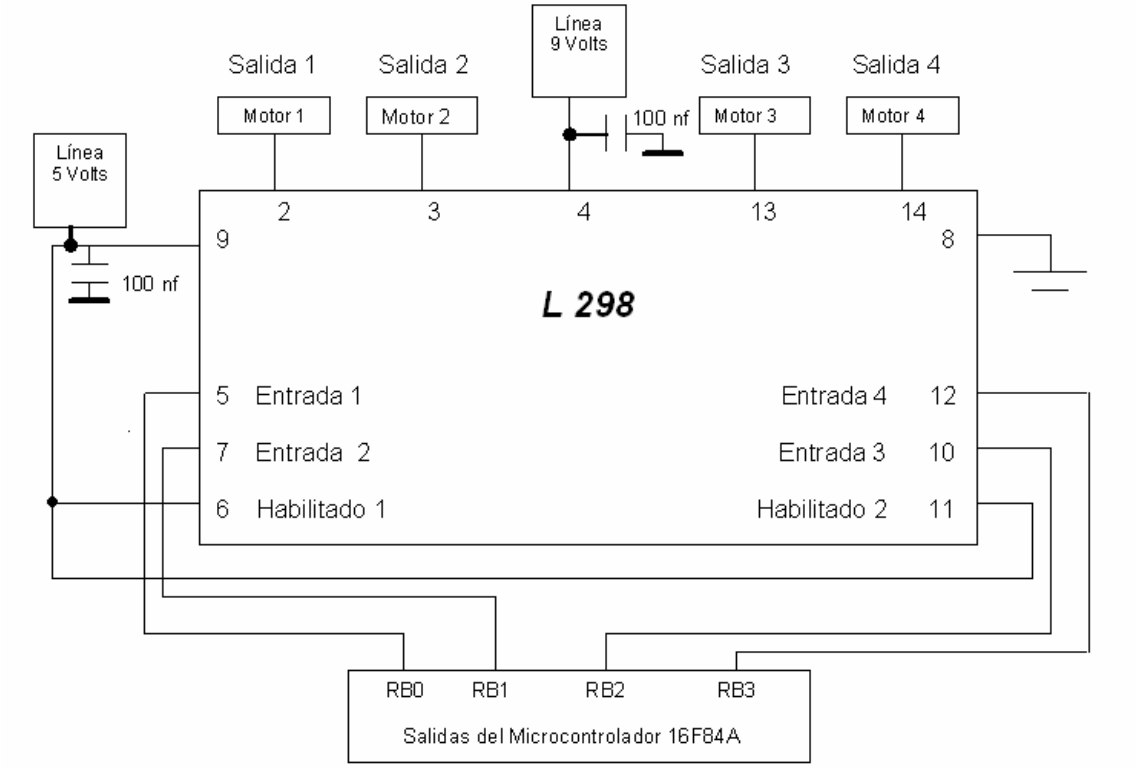


Figura 4.18 Diagrama de bloques del L298.

Características generales de la interfaz Puente H:

- Activación de motores en un rango entre 9 y 50 Volts - DC
- Capacidad para entregar hasta 2.5 Amperes a la carga
- Capacidad para activar el giro del motor en cualquiera de los dos sentidos
- Reducido tamaño

Un puente H se pude ver como un arreglo de 4 interruptores ensamblados en el circuito integrado por ejemplo:

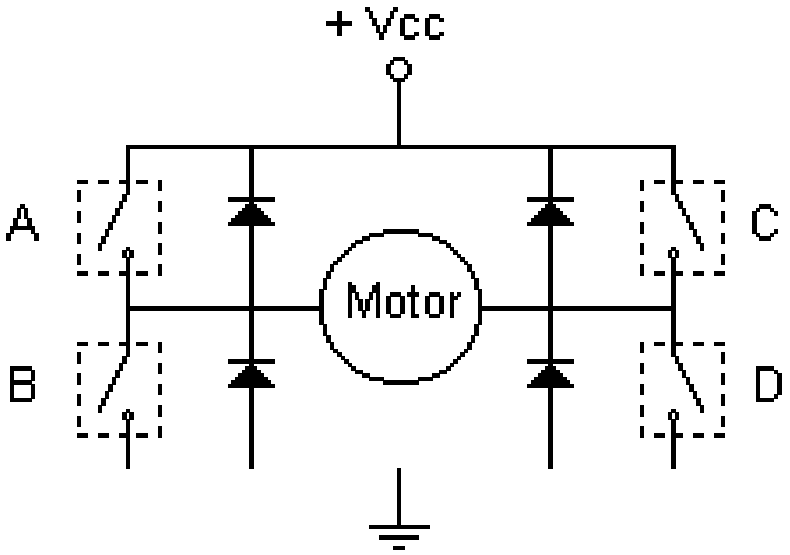


Figura 4.19 Representación del Puente H.

En la figura 4.19 los interruptores (A, B, C y D), están compuestos de transistores, relevadores o cualquier otra combinación de elementos. Si se cierran los interruptores A y D la corriente circulara en un sentido y si se cierran los circuitos B y C la corriente circulará en sentido contrario.

4.5.2 Motor

Como se mencionó anteriormente, el motor modelo FF-050SK-09250 utilizado es de las siguientes características:

Modelo: FF-050SK-09250

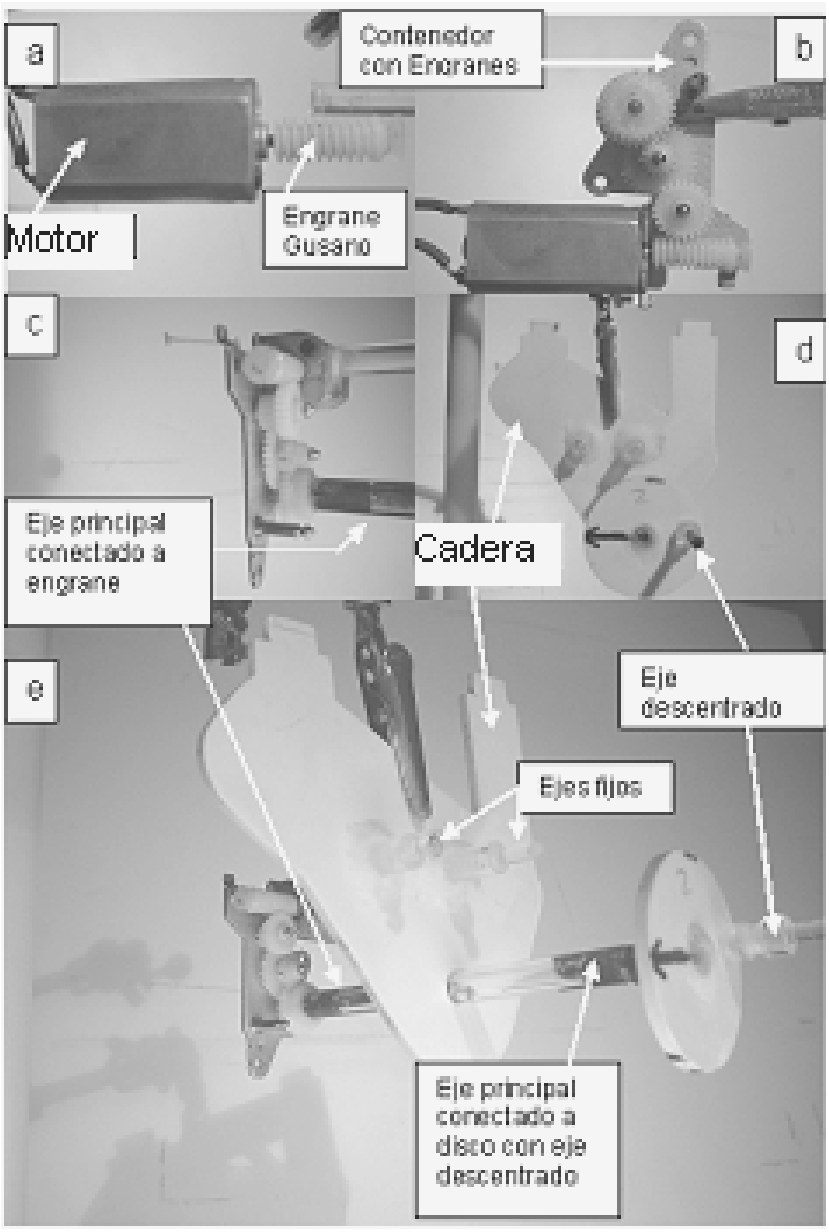
Tensión		Sin Carga		Máxima Eficiencia				Paro	
		Velocidad	Corriente	Velocidad	Corriente	TORQUE	Salida	TORQUE	Corriente
Rango de Operación	Nominal	rpm	A	rpm	A	g.cm / mN.m	W	g.cm / mN.m	A
5.0-11.5 V	8V CONSTANTE	7900	0.027	6300	0.11	7.0 / 0.69	0.45	35 / 3.43	0.43

4.6 MÓDULOS MECÁNICOS

El microcontrolador genera pulsos modulados (PWM) al Puente H para activar a los diferentes motores acoplados al engranaje utilizados para el movimiento de las extremidades como se muestra en la siguiente sección la cual permite conocer las partes que compone al módulo mecánico.

4.6.1 Motor y engranes

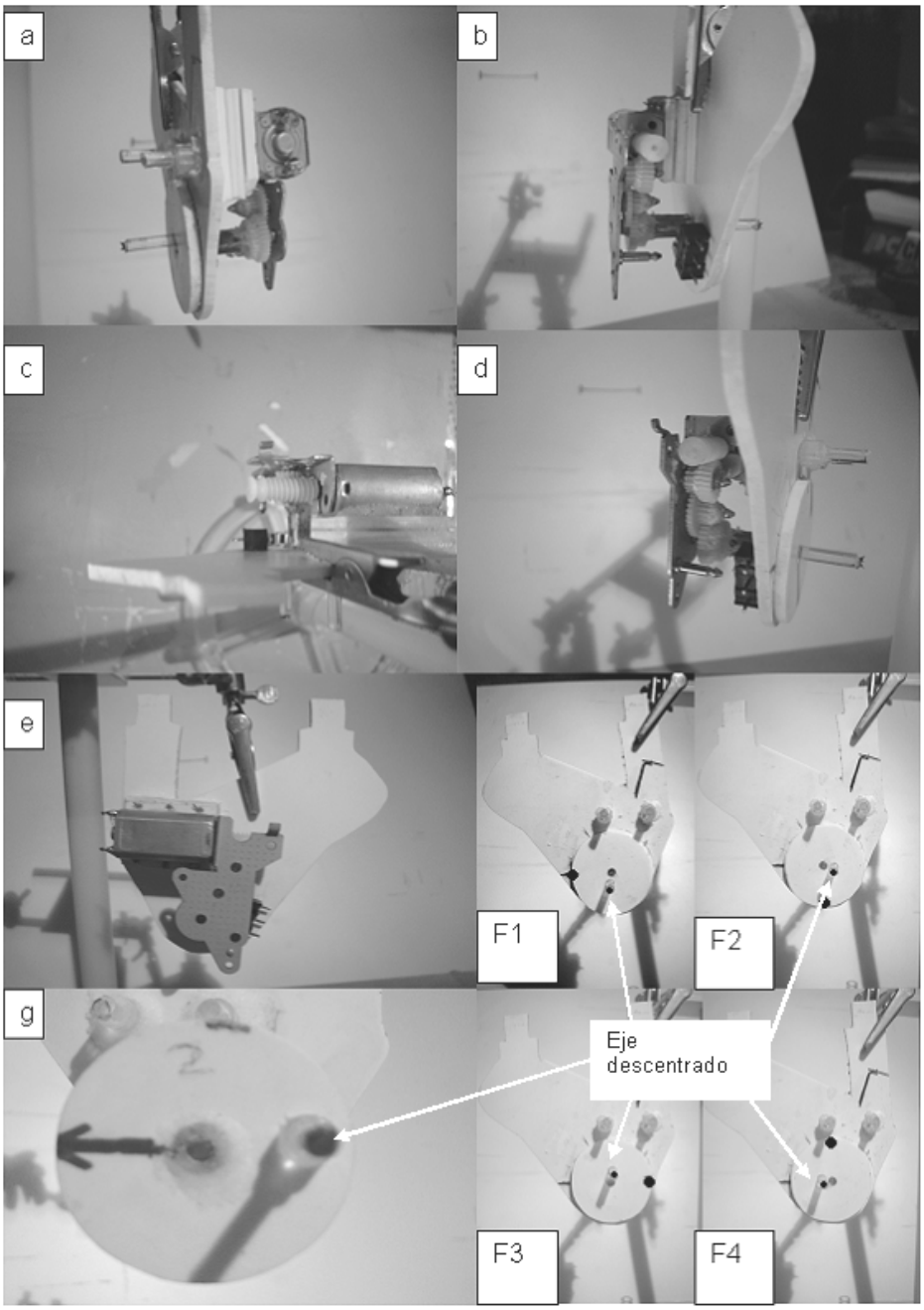
El motor Fbelec, se une al acoplamiento mecánico -carcasa con 3 ejes fijos metálicos y paralelos, cada uno contienen ruedas dentadas formando engranes solidarios, donde el inferior de estos se une al eje del motor por medio de un tornillo sin fin-, y a la extremidad. El ensamble del motor con engranes, así como, el eje principal de transmisión para la extremidad se muestran en la siguiente figura:



a) Motor con engrane de gusano. b) Acoplamiento del Motor con el contenedor con solidarios. c) Engrane con eje principal d) Cadera con el eje principal y conectado el disco con eje descentrado e) Presentación del Acoplamiento con el motor, cadera y disco con eje descentrado.

Figura 4.20 Acoplamiento del Motor con la Caja de Engranes y Cadera.

Al activar el motor, los engranes giran el eje principal que atraviesa la cadera del robot, así como, el disco con eje descentrado que se puede observar a continuación:



a) Vista trasera de la cadera con el motor y engranes b) Vista delantera de la cadera con el motor y engranes c) Vista superior de la cadera con el motor y engranes d) Vista delantera de la cadera con el motor y engranes, donde podemos ver al disco con eje descentrado e) Vista oculta de la cadera con el motor y engranes. F1) Disco con eje descentrado a 270 grados. F2) Disco con eje descentrado a 180 grados. F3) Disco con eje descentrado a 90 grados. F4) Disco con eje descentrado a 0 grados.

Figura 4.21 movimiento del eje descentrado en la cadera del robot cuadrúpedo.

El disco con eje descentrado, es la generadora de movimiento en la extremidad. Al girar el disco, el eje descentrado cambia de posición debido a que esta fijo al disco, como se observo en la figura anterior.

Al girar el disco con eje descentrado, la extremidad comienza a moverse, y al estar dividida en varias piezas, estas ayudan al desplazamiento del robot cuadrúpedo que se describen a continuación.

4.6.2 Extremidad

La extremidad del robot cuadrúpedo están formadas por varias piezas, las cuales se muestran en la figura 4.22.

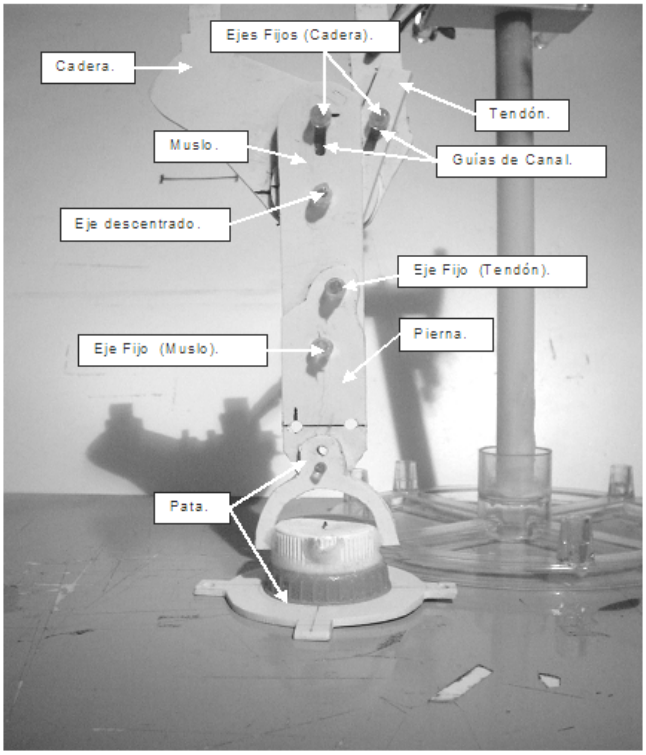


Figura 4.22 Partes de la extremidad del robot cuadrúpedo.

De las piezas que forman las extremidades del robot cuadrúpedo, la cadera contiene al motor con engrane, así como, el engrane con el eje principal que

atraviesa a la cadera uniéndose al disco con el eje descentrado, que se une con guías circulares al tendón y al muslo.

Para mantener las posiciones del muslo y el tendón, en sus partes superiores tienen guías de canal que son atravesadas por ejes fijos situados en la parte superior de la cadera.

En la parte inferior del muslo se encuentra un arreglo para flexionar la rodilla que consta de un eje fijo y una guía de canal. El eje fijo del muslo atraviesa una guía circular de la pierna volviéndose la rodilla. La guía de canal del muslo (parte inferior) y la guía circular de la pierna (parte superior), son atravesadas por un eje fijo del tendón (parte inferior), quedando empalmados el tendón, el muslo y la pierna.

La flexión de la rodilla levanta ligeramente a la pata. La pata esta empalmada a la parte inferior de la pierna con ejes circulares situados en los arcos que tienen la pata con otro eje circular en la parte inferior de la pierna y que son empalmadas por una varilla de acrílico, permitiendo a la pata moverse con cierta libertad ya que es limitada por topes que están entre la pierna y la pata.

Una vez analizado la estructura y funcionamiento de la extremidad, su movimiento puede sintetizarse en 4 pasos:

Paso 1.- La extremidad está estacionada. La pata está apoyada en el piso y tanto el muslo como el tendón tienen sus guías canal con los ejes fijos de la cadera en el punto más alto (punto de Inicio). Esto se debe a que están conectados mediante el disco con el eje descentrado a 270 grados y hace que el tendón ponga a la pierna en posición recta al igual que al muslo.

Paso 2.- La extremidad comienza a moverse, inclinándose 30 grados a la derecha, por lo que la pata comienza a despegarse del piso. En el muslo y en el

tendón sus guías canal con los ejes fijos de la cadera se encuentran en el punto medio. El disco con eje descentrado esta a 180 grados y hace que el tendón ponga a la pierna en posición inclinada al igual que al muslo.

Paso 3.- La extremidad mantiene los 30 grados a la derecha de inclinación, pero la pierna comienza a inclinarse, debido a que el tendón activa al arreglo de rodilla, que pone a la pierna en una posición más inclinada que el muslo levantando a la pata. En el muslo y en el tendón sus guías canal están en el punto mas bajo. El disco con eje descentrado está a 90 grados.

Paso 4.- La extremidad está por regresar a la posición de inicio, inclinándose 30 grados a la izquierda, por lo que la pata comienza a descender al piso. En el muslo y en el tendón sus guías canal regresan al punto medio. El disco con eje descentrado esta a cero grados y hace que el tendón ponga a la pierna en posición recta al igual que al muslo volviendo al Paso 1.

El movimiento de la extremidad se muestra en la siguiente secuencia:

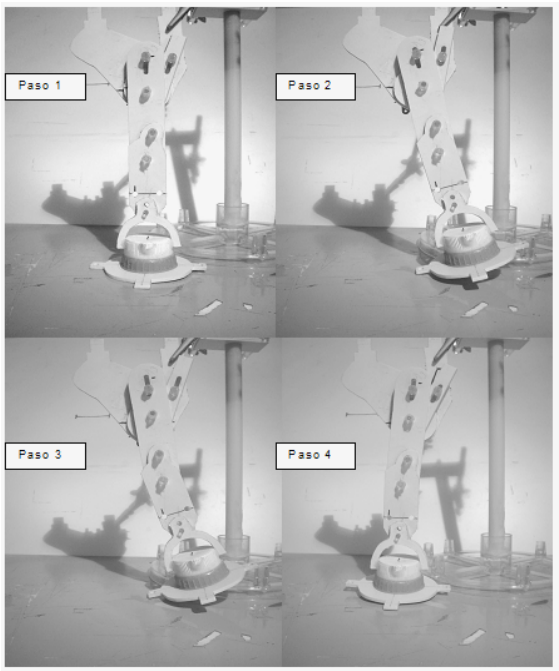


Figura 4.23 Movimiento de la extremidad.

CAPÍTULO V

5 CONSTRUCCIÓN Y PRUEBAS

Tras lo visto en los capítulos anteriores, se puede construir un robot cuadrúpedo unidireccional. Pero para ello, la implementación final de este trabajo debe integrarse en módulos, justo como se detallaran a lo largo de los capítulos III y IV, para llegar así al diagrama de bloques (figura 5.1) que sintetiza la implementación final a partir de los dos principales módulos componentes: el electrónico y el mecánico.

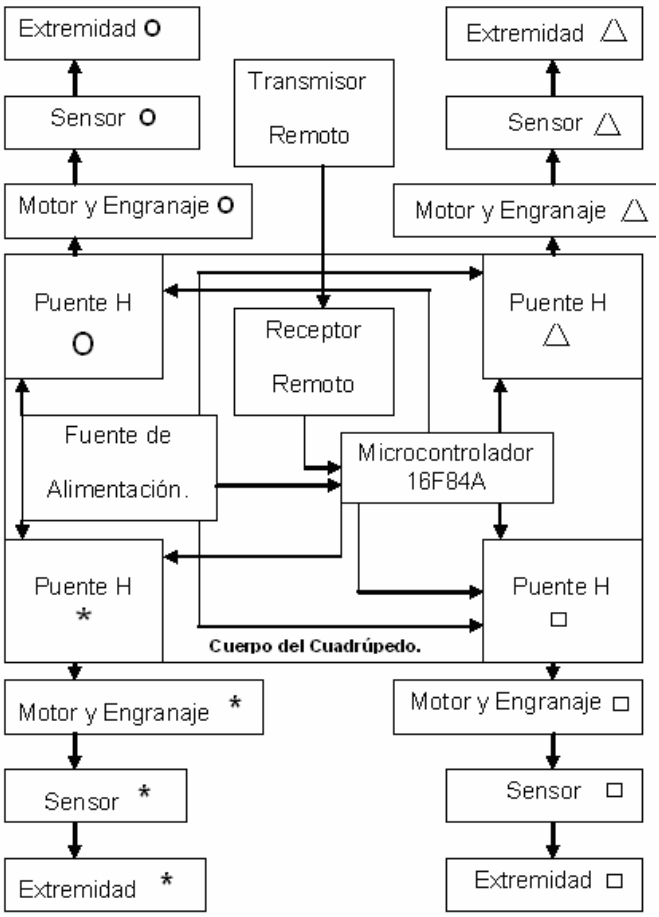


Figura 5.1 Diagrama de bloques de la implementación final.

5.1 CONSTRUCCIÓN Y PRUEBAS DE MÓDULOS ELECTRÓNICOS

De los diagramas esquemáticos de cada módulo electrónico (diseño) se implementan las diferentes tablillas mediante una tarjeta perforada para cada uno de los componentes como reguladores, circuitos integrados, LEDS, resistores y capacitores, mostrando también las fotografías de los diseños finales, así como sus diagramas de diseño.

5.1.1 Fuente de alimentación

La implementación de la fuente de alimentación, comprende los reguladores de voltaje 7805 y 7809 (capítulo IV, figura 4.2), donde cada regulador –protegido del calentamiento con un disipador comercial-, es colocado en la tarjeta perforada que lleva el robot cuadrúpedo misma que llamaremos en adelante, tarjeta principal.

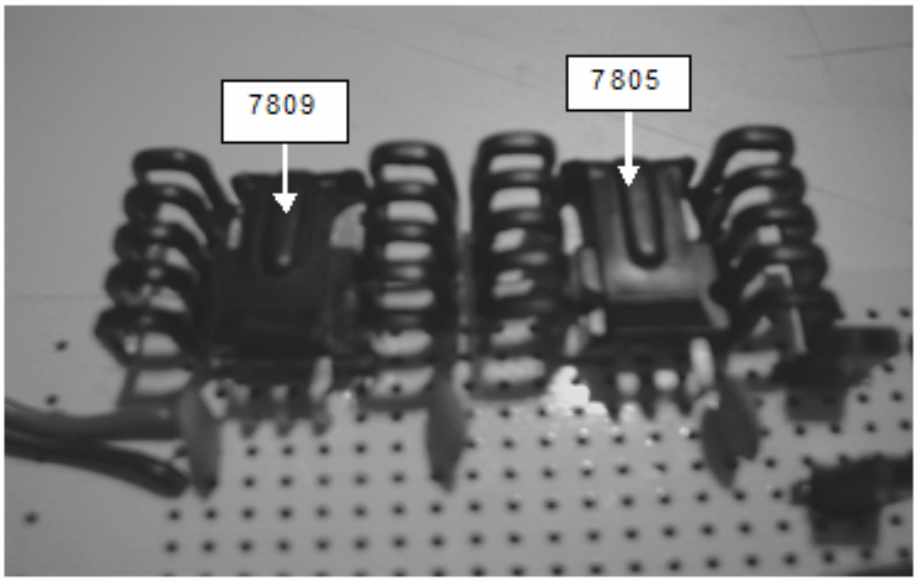


Figura 5.2 Implementación de la fuente de alimentación.

Adicionalmente, en esta tarjeta se encuentran dos líneas que se usan para las tensiones de 5 y 9 Volts, respectivamente; las cuales facilitan la conexión de los dispositivos como se muestra en la figura 5.3.

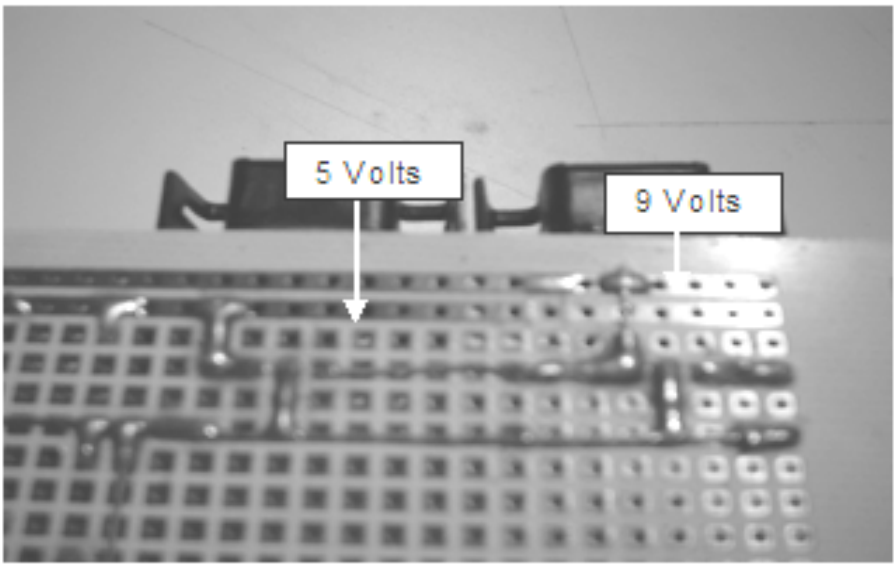


Figura 5.3 Líneas de tensión.

5.1.1.1 Pruebas en la fuente de alimentación

Una vez conectadas las baterías en la entrada del regulador 7809, se debe verificar con el voltímetro la correcta entrega de las tensiones de 9 y 5 Volts en cada línea asignada.

5.1.2 Control Transmisor/Receptor remoto

El diagrama esquemático del mando remoto (transmisor/ receptor), se muestra a continuación.

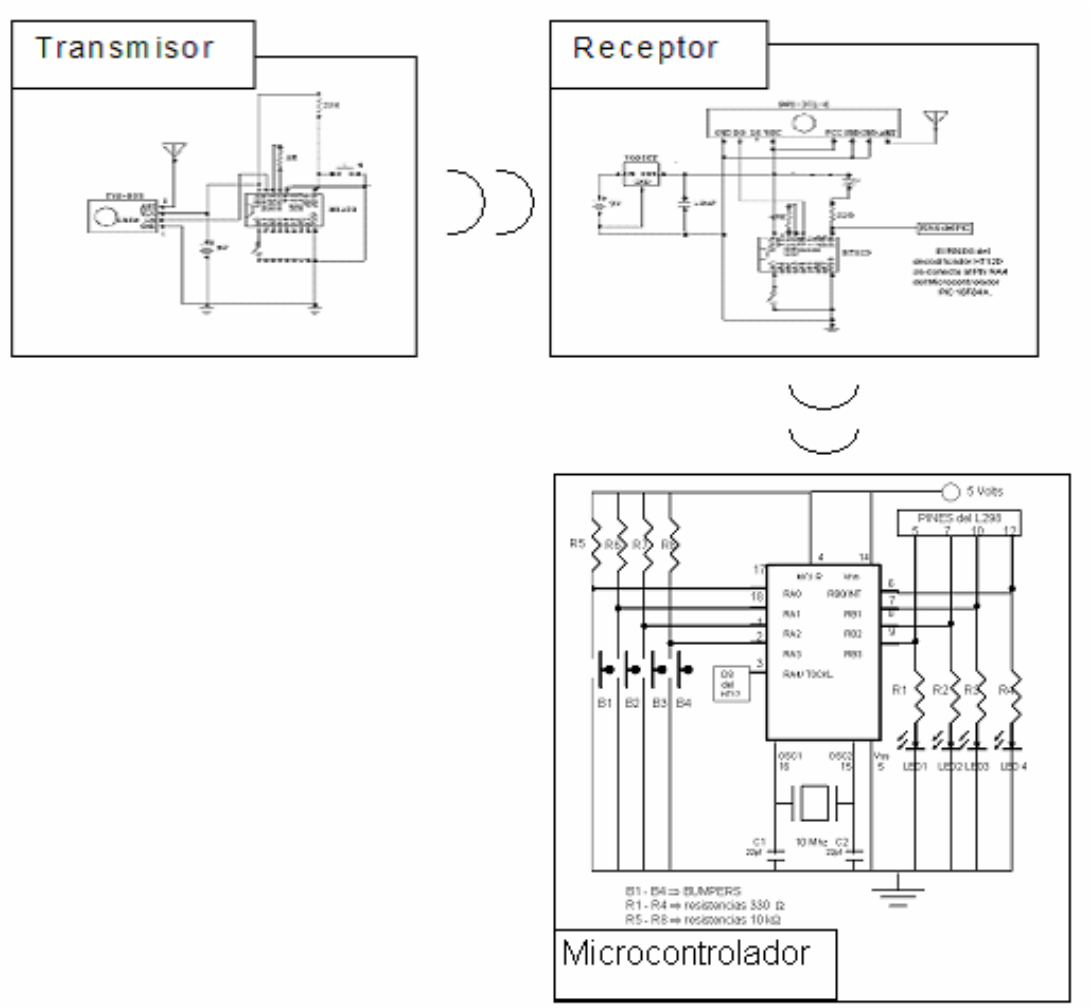


Figura 5.4 Diagrama esquemático mando Transmisor/Receptor remoto.

La implementación del mando remoto, que comprende al transmisor **TWS-BS3**, con un codificador HT-12E y al receptor **RWS-371-6**, con un decodificador HT-12D se muestran en la figura 5.5.

El receptor y el decodificador son colocados en la tarjeta principal y se energizan por una tensión de 5 Volts. El **RWS-371-6** tiene cableado externo a la tarjeta principal (al estar cubierta ésta última por una carcasa decorativa).

Por su parte, la tarjeta perforada del transmisor **TWS-BS3** con el codificador HT-12E**,** es independiente de la tarjeta principal y es energizada por una pila alcalina de 9 Volts.

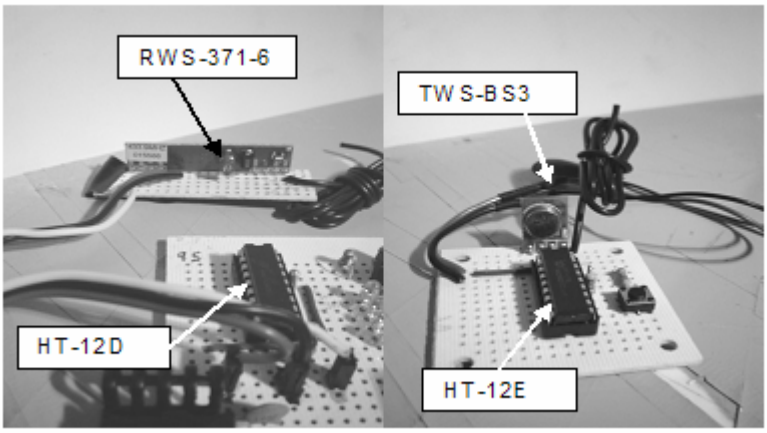


Figura 5.5 Mando remoto Receptor/Transmisor en la tarjeta perforada.

5.1.2.1 Pruebas en el mando Transmisor/Receptor remoto

Una vez conectada la batería alcalina de 9 Volts en la tarjeta perforada del transmisor y energizado el regulador 7805 de la tarjeta principal, se verifica el buen funcionamiento del Transmisor/Receptor por medio de un LED (figura 5.6) conectado al receptor, el cual al apretar el pulsador del transmisor, enciende al LED y al soltarlo (el pulsador) lo apaga.

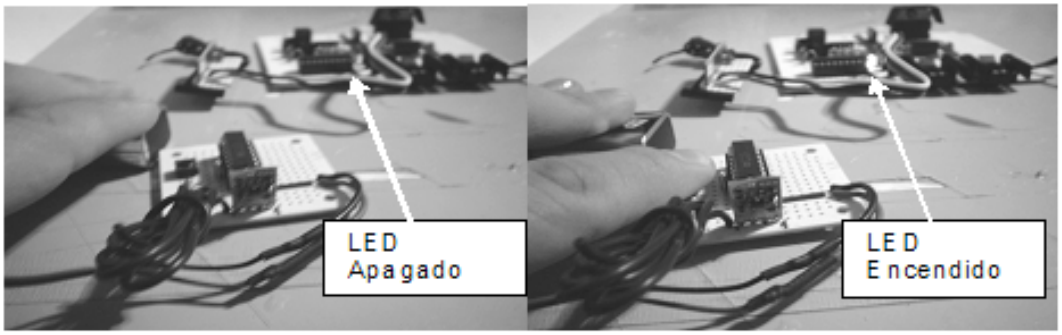


Figura 5.6 Prueba del mando Transmisor/Receptor.

5.1.3 Microcontrolador 16F84A

La implementación del microcontrolador agrupa a los LEDS, a los "bumpers" y al oscilador (figura 5.7); por lo que el microcontrolador es colocado en

la tarjeta principal y conectado al D8 (PIN 10) del decodificador de la etapa de recepción con la entrada del puerto a RA4 (PIN 3) del microcontrolador 16F84A.

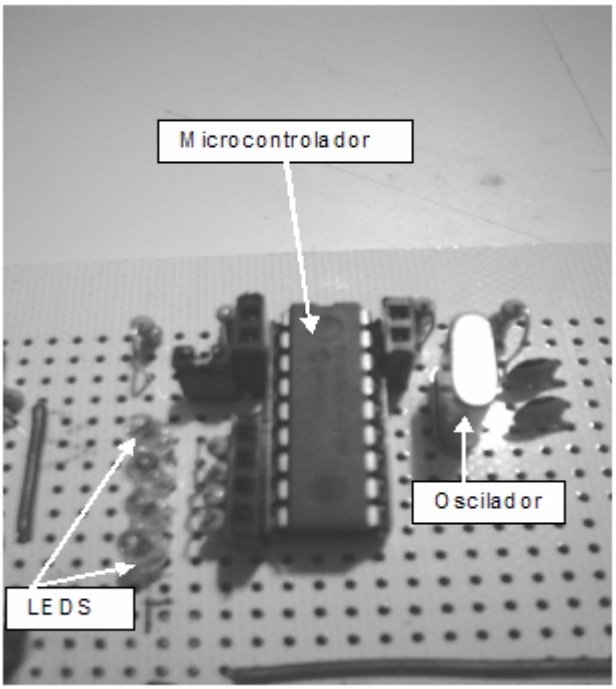


Figura 5.7 Implementación del microcontrolador.

Una vez implementado el zócalo y los componentes necesarios para que funcione el microcontrolador, se procede a realizar la prueba con el programa en código fuente de movimiento del motor mediante la técnica PWM, el cual permite ver la intermitencia en la salida del puerto B del microcontrolador.

5.1.3.1 Código fuente de movimiento del motor mediante la técnica PWM

Al tomar en cuenta el diagrama de flujo de la subrutina de movimiento del motor mediante la técnica PWM, se genera el siguiente código fuente el cual incluye PWM en cada pin de salida del puerto B asignado.

```
;*************************************Retardo_03.asm*************************************************************************
;
;El Led conectado a la linea 0 del puerto de salida se enciende durante 400 ms y se apaga durante 300 ms.
;
;ZONA DE DATOS*****************************************************************************************************************

            __CONFIG _CP_OFF & _WDT_OFF & _PWRTE_ON  & _XT_OSC
            LIST P=16F84A
            INCLUDE <P16F84A.INC>

            CBLOCK  0x0C
          ; En esta posición empieza la RAM de usuario.
            ENDC
reg1        equ         12
reg2        equ         13
reg3        equ         14
reg4    equ 15

#DEFINE LED1  PORTB,0
#DEFINE LED2  PORTB,1
#DEFINE LED3  PORTB,2
#DEFINE LED4  PORTB,3
#DEFINE LED5  PORTB,4
;ZONA DE CÓDIGOS***************************************************************************************************************

                    ORG   0
Inicio

     bsf    STATUS,RP0
                    bcf    LED1
                    bcf    LED2
                    bcf    LED3
                    bcf    LED4
                    bcf    STATUS,RP0


Principal

Cuadrito
              movlw  9f
                        movwf   reg1
uno     bsf    LED1

                        call    Retardo_2ms
                    call    Retardo_2ms
                        call    Retardo_500micros
```

```
                    call   Retardo_20micros
                    call   Retardo_20micros

                    bcf    LED1

                call   Retardo_200micros
                call   Retardo_200micros
                    bsf    LED1

                call   Retardo_2ms
                    call   Retardo_2ms
                    call   Retardo_500micros
                    call   Retardo_20micros
                    call   Retardo_20micros



                    bcf    LED1

                call   Retardo_200micros
                call   Retardo_200micros
    decfsz  reg1,1
                    goto   uno
                    call   hola
                    call     Circulito
Circulito

          movlw  9f
                    movwf  reg2
dos   bsf    LED2
    call   Retardo_2ms
           call   Retardo_2ms
                    call   Retardo_500micros
          call   Retardo_20micros
          call   Retardo_5micros
                    bcf    LED2

                call   Retardo_500micros
                    bsf    LED2

                call   Retardo_2ms
                  call   Retardo_2ms
                  call   Retardo_500micros
                  call   Retardo_20micros
                  call   Retardo_5micros
                    bcf    LED2
```

```
          call    Retardo_500micros
     decfsz  reg2,1
                     goto    dos
                     call   hola
                     call     Asterisco

Asterisco

                     movlw  9f
                     movwf   reg3
tres    bsf    LED3

                     call    Retardo_2ms
                   call    Retardo_1ms
                     call    Retardo_50micros
                     call    Retardo_5micros
                   bcf     LED3

                   call    Retardo_200micros
                   call    Retardo_200micros
                     bsf     LED3

                   call    Retardo_2ms
                     call    Retardo_1ms
                     call    Retardo_50micros
                     call    Retardo_5micros

                     bcf     LED3

                   call    Retardo_200micros
                   call    Retardo_200micros
     decfsz  reg3,1
                     goto    tres
                     call   hola
                     call     Triangulito

Triangulito
     movlw   9f
                     movwf   reg4
cuatro  bsf    LED4

                     call    Retardo_2ms
                   call    Retardo_2ms
                     call    Retardo_20micros
                     call    Retardo_20micros
                      call    Retardo_5micros
```

```
                bcf    LED4

                call   Retardo_200micros
                call   Retardo_200micros
                   bsf    LED4

                call   Retardo_2ms
                   call   Retardo_2ms
                   call   Retardo_20micros
                   call   Retardo_20micros
                  call   Retardo_5micros


                   bcf    LED4

                call   Retardo_200micros
                call   Retardo_200micros
    decfsz  reg4,1
                   goto   cuatro
                   call   hola
                   goto   Inicio

;Subrutina hola*****************************************************************************

hola
                  bsf LED5
                  call Retardo_5s
                  bcf LED5

                 return
```

5.1.3.2 Pruebas con el Microcontrolador 16F84A

Al cargar el programa de movimiento del motor mediante la técnica PWM al microcontrolador, y se energiza al microcontrolador con 5 Volts. Se implementa un arreglo de LEDS conectados a la salida del microcontrolador (puerto B), para simular las secuencias de apagado y encendido de cada motor. En la figura 5.8 se muestra la secuencia que realiza los LEDS en la tarjeta principal.

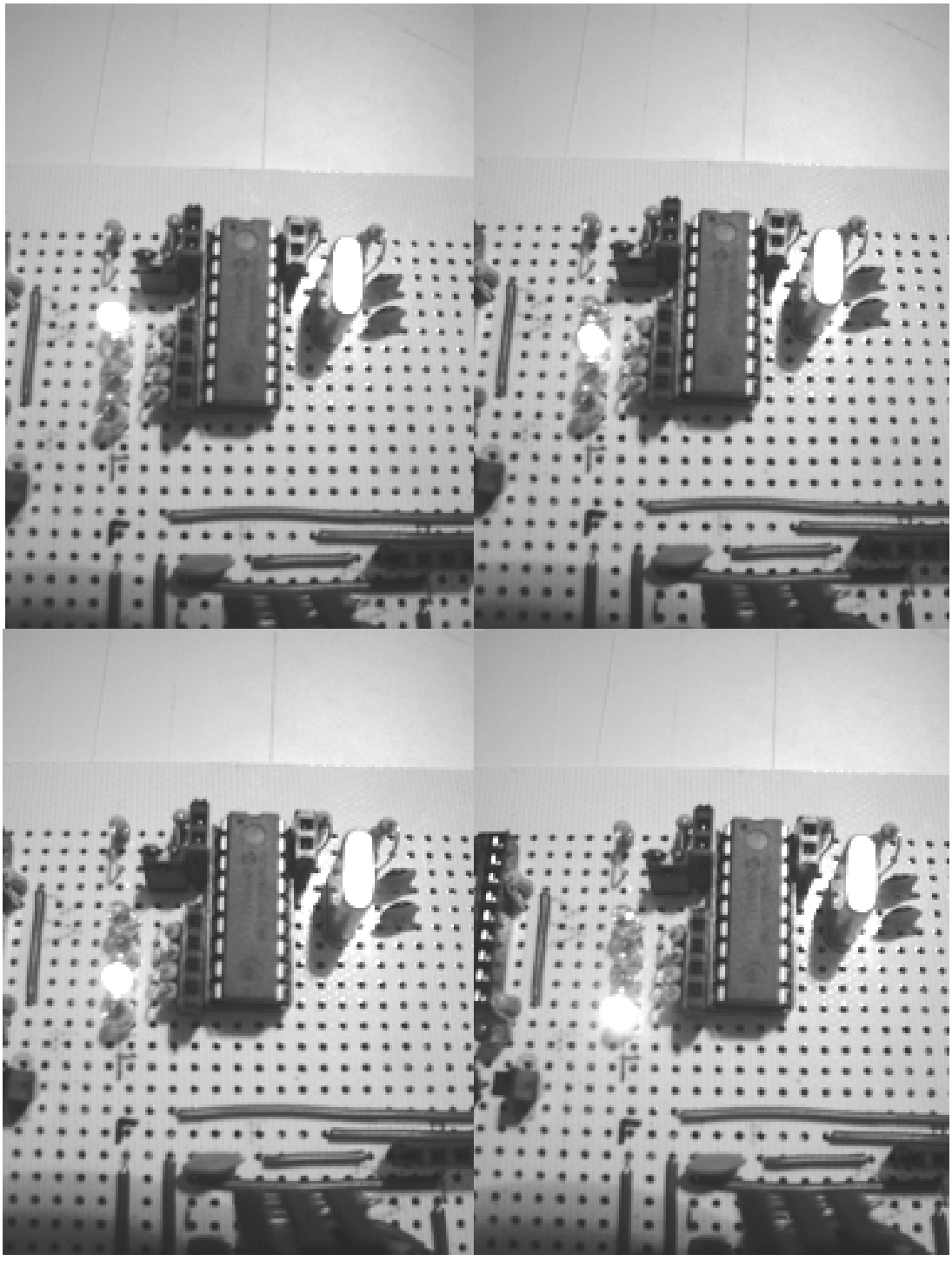


Figura 5.8 Implementación de la secuencia de movimiento del motor mediante la técnica PWM.

5.1.4 L298 y motor

En la implementación del circuito L298 –como se observa en la figura 5.9-, éste se debe energizar con dos tensiones; una de 5 Volts para el funcionamiento del Puente H, y otra con 9 Volts para energizar al motor. Además, se le coloca un disipador comercial para protegerlo del calentamiento.

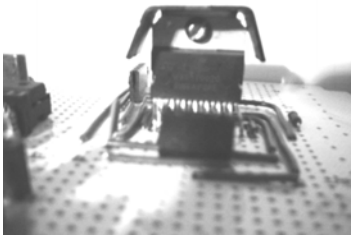


Figura 5.9 Implementación del L298.

Un componente importante en el L298 es el dispositivo que se alimenta con éste (carga). En este caso, la carga corresponde a cada motor –energizado en su entrada, con una fuente de poder- que mueve a los engranes y permite dar movimiento a la extremidad (figura 5.10).

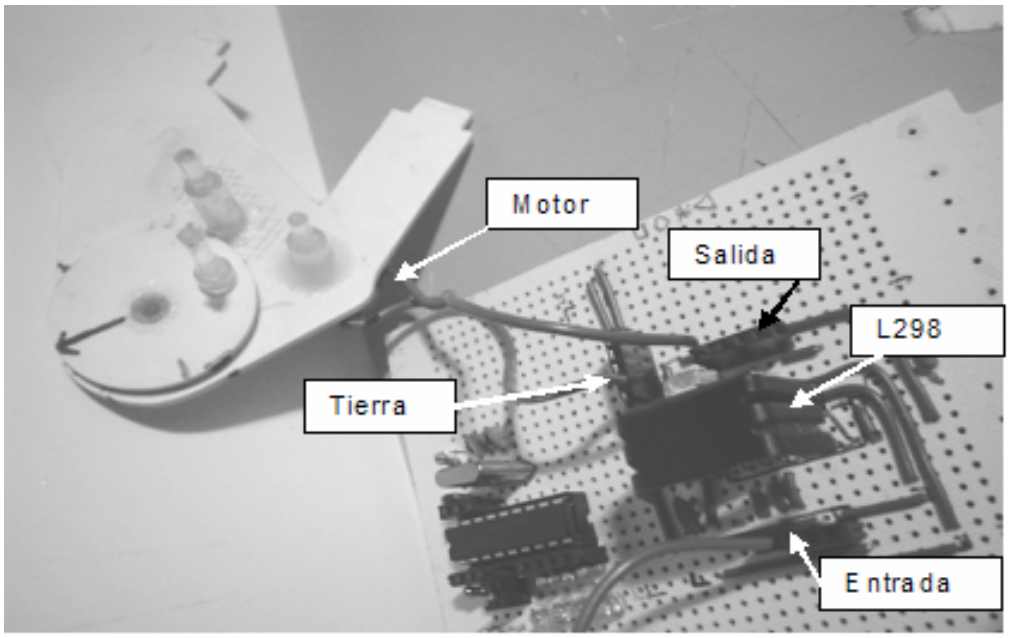


Figura 5.10 Conexión del L298 y el motor.

5.1.4.1 Prueba con el L298 y el motor

Para verificar el desempeño del L298 con certeza, se realiza una prueba sin integrarlo al microcontrolador. Esto quiere decir, que la entrada del Puente H debe tener una tensión de 5 Volts y a su salida debe conectarse el motor, donde una terminal se conecta al pin de salida y la otra terminal a tierra. Para comprobar el funcionamiento del L298, éste debe encender al motor sin dificultad alguna.

5.2 CONSTRUCCIÓN Y PRUEBAS DE MÓDULOS MECÁNICOS

La fabricación del prototipo del robot cuadrúpedo unidireccional conlleva un trabajo completamente artesanal debido al corte, lijado, ajuste y pegado de piezas, mismas que presentan diminutas variantes en relación a su peso y dimensión como lo son por ejemplo: las extremidades, el acoplamiento de sus motores, así como la colocación de las extremidades en la base. No obstante, son tareas tratadas con la mayor precisión posible, ya que se usa una pieza-molde y sobre esta la reproducción de las demás.

Algunas de las herramientas usadas para poder construir las extremidades son adaptaciones de otras herramientas o creaciones específicamente diseñadas para dichas tareas de precisión.

5.2.1 Herramientas

Para que las extremidades no pesen tanto, se trabaja con estireno en láminas de diferentes espesores. Para el caso del cuadrúpedo, la placa de estireno a usarse es de 3mm de espesor, lo que en el rango de las láminas de estireno significa que no se deforma con facilidad y proporciona un soporte con la suficiente resistencia mecánica para funcionar como el esqueleto de las extremidades.

El corte de todas las piezas se traduce en dos herramientas principalmente: una segueta de mano y la navaja para cortar estireno. No obstante, también se requieren limas para quitar imperfecciones (figura 5.11) y realizar ajustes milimétricos para los acoplamientos y guías.

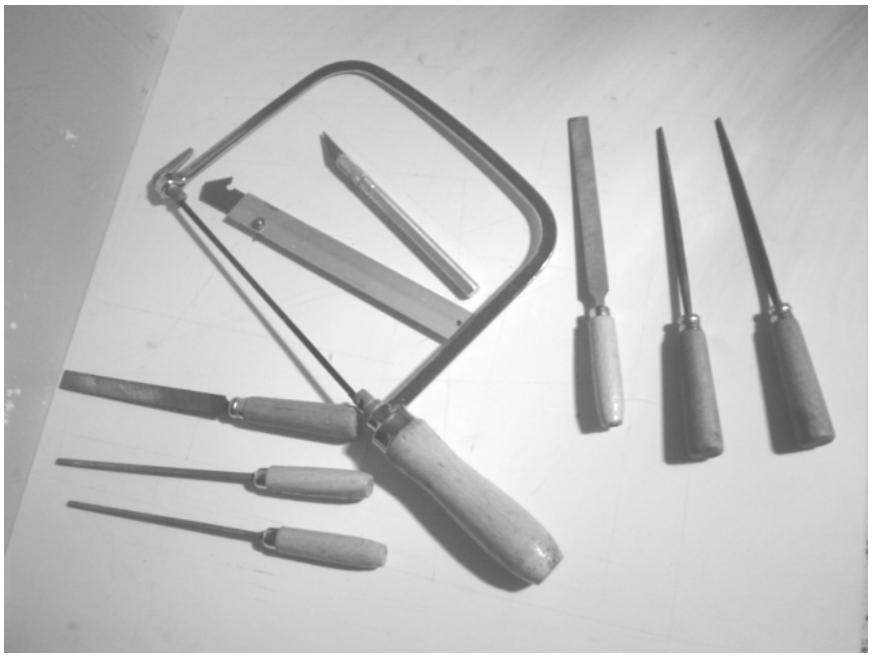


Figura 5.11 Segueta de mano, limas y cortador de estireno.

Para marcar y hacer las guías y canales se necesita otra herramienta: el taladro con una burbuja de nivel adaptada al mango, lo que permite la elaboración mas precisa y balanceada de las extremidades al ser perforadas, en el caso de los ejes permite eliminar fricciones debidas a la imperfección del agujero con el eje. Sin embargo, también hay que considerar el calibre de la broca que debe ser de 1/8 “como se muestra abajo en la figura 5.12, que muestra por si misma de la finura requerida para realizar este trabajo.

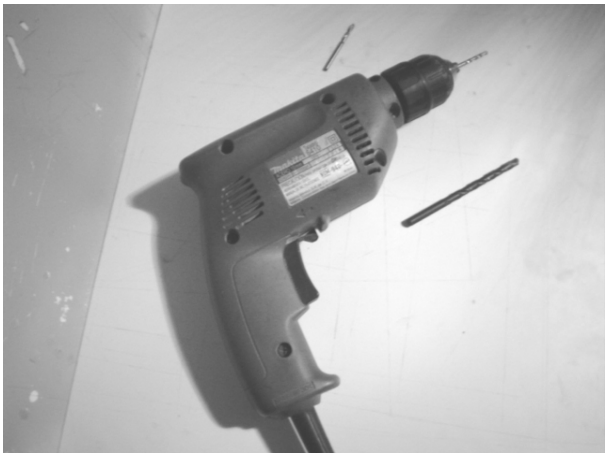


Figura 5.12 Taladro de velocidad variable y brocas.

Para aquellas piezas que requieren de ejes fijos se usa varilla dura de acrílico de 1/8 y en el caso del eje principal, varilla hueca de 1/4” como las que se ilustran a continuación. Estos materiales tienen un buen desempeño para soportar las cargas mecánicas y permiten una adaptación más sencilla que otros materiales plásticos.

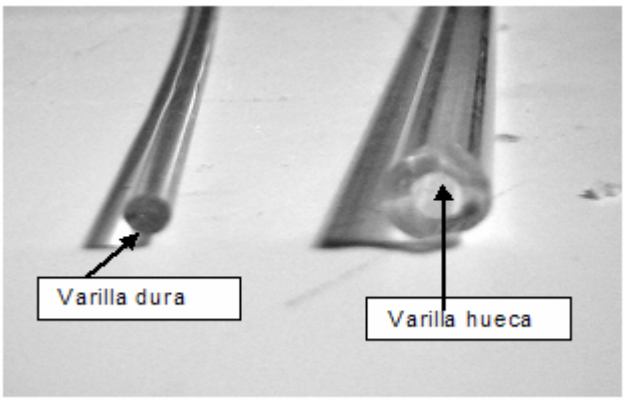


Figura 5.13 Varillas de acrílico.

Ahora bien, cuando se llega a la etapa de ensamblado (pegado de las piezas plásticas), es importante no utilizar cualquier tipo de pegamento ya que no todos funcionan, reaccionan y adhieren igual en todos los plásticos.

Lo ideal en este proceso son el pegamento instantáneo (a base de cianoacrilato, por lo que las medidas de seguridad en su manejo son altas por el contenido de cianuro en los vapores que genera su aplicación sobre las superficies plásticas) y el pegamento de silicón. Respecto al pegamento instantáneo, cabe mencionar que cualquier marca comercial es adecuada, ya que, presentan una adherencia muy similar.

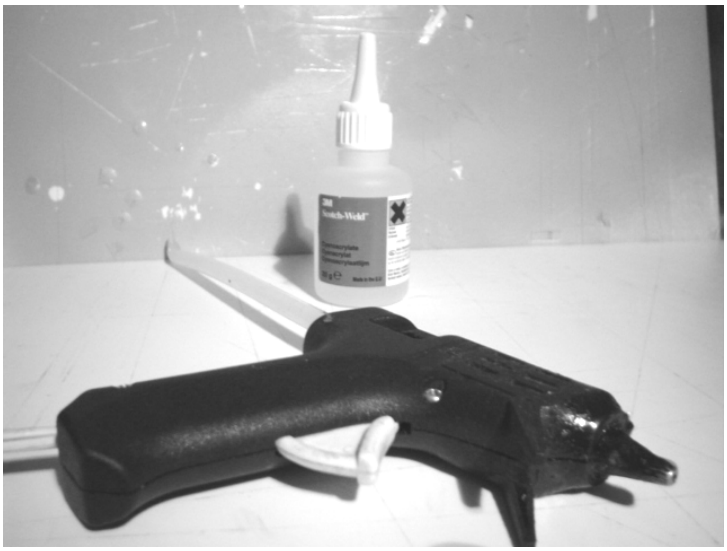


Figura 5.14 Pegamentos: (al fondo) instantáneo y (al frente) de silicón.

5.2.2 Elaboración de guías

Como se mencionó en el capítulo anterior, las guías son de vital importancia para el adecuado desplazamiento de la extremidad o movimientos giratorios de esta. De ahí que se requiera la elaboración de dos tipos diferentes de guías (figura 5.15):

Guía Circular: formada básicamente por un agujero, hecho con taladro con nivel y una broca de 1/8”.

Guía de Canal: trazada sobre una línea recta en la pieza. En cada extremo de la línea se hace un agujero con la broca de 1/8, mismos que se unen, trazando

otras 2 líneas tangentes, una en la parte superior y otra en la inferior de cada agujero para entonces ser cortadas por la navaja para cortar estireno, quitando el sobrante con una lima.

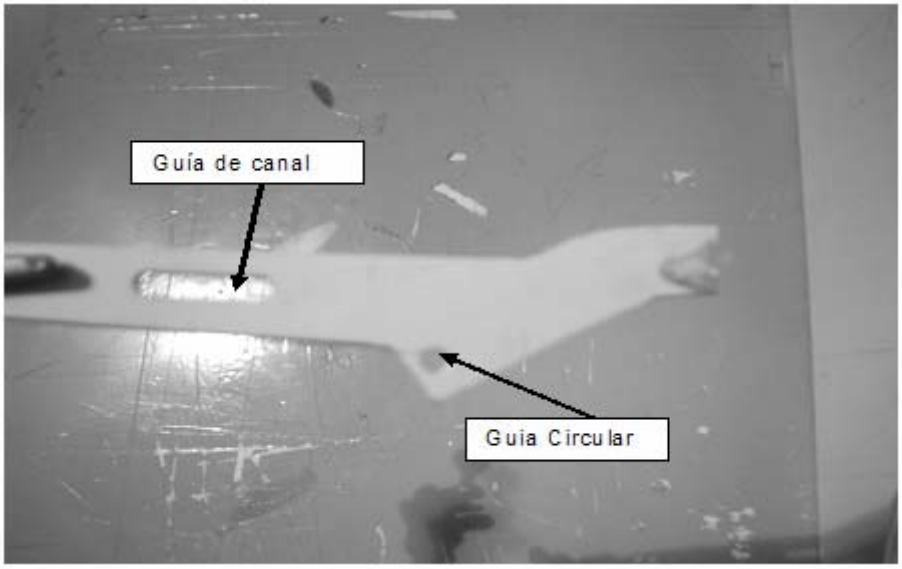


Figura 5.15 Guía circular y guía de canal para una pieza.

5.2.3 Elaboración de ejes

Los ejes son aditamentos que deben de tener cierta rigidez ya que acoplan a una o más piezas que por lo general pueden desarrollar (en ellos) cierta resistencia o tensión mecánica.

Existen 2 tipos de ejes en el robot cuadrúpedo: principal (5.16) y fijos (5.17), esto es debido a que cada tipo tiene funciones muy específicas que se expone a continuación.

Eje Principal: su elaboración requiere varilla hueca de estireno de 10” y fijarse al engrane con pegamento de contacto y bicarbonato de sodio, el cual le brinda una mayor dureza. Para el agujero en la cadera que atraviesa el eje principal se utiliza una broca de 1/4”.

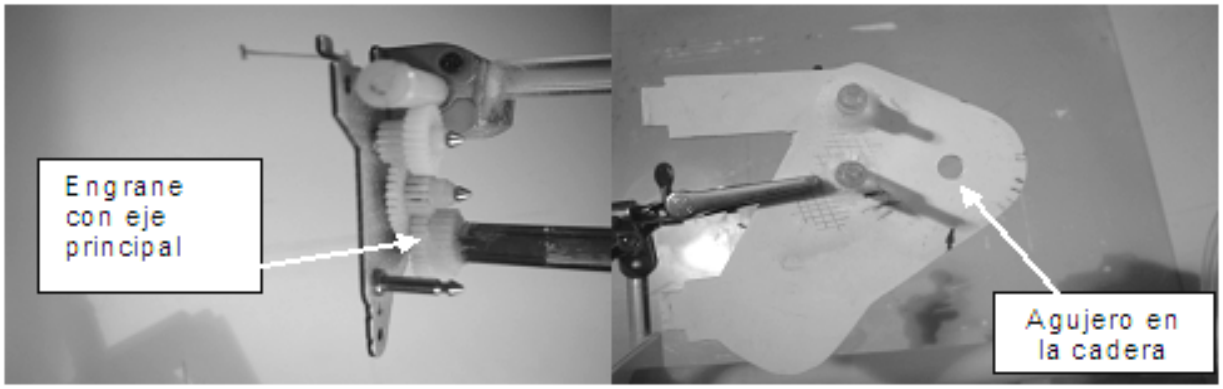


Figura 5.16 Eje Principal.

Eje Fijo: su elaboración precisa de un agujero en la pieza con el taladro y broca de 1/8”, ya que el eje en sí, es de varilla dura de 1/8” y generalmente se fija con pegamento instantáneo.

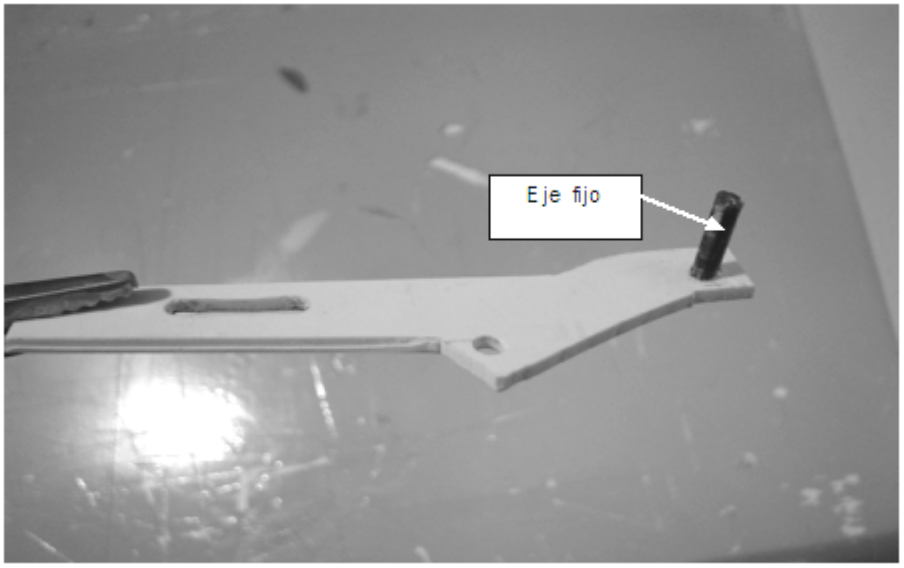


Figura 5.17 Eje Fijo.

5.2.4 Elaboración de la extremidad

Una vez entendido el cómo se elaboran las piezas y sus accesorios, se puede proceder al diseño y hechura de las extremidades formadas por ocho piezas cada una de ellas: 1) base rectangular, 2) cadera, 3) disco con eje descentrado, 4) muslo, 5) tendón, 6) pierna y 7) pata. Esto quiere decir, que de la mayoría de las piezas se deben hacer por cuatro tantos.

1) Base rectangular

Para sujetar a las cuatro extremidades se necesita una base rectangular de 10cm x 30cm (figura 5.18) con unas guías de canal en donde se acoplan las caderas. Esta base, además, permite contar con un contenedor para mantener a la tarjeta principal en el interior de la base de forma vertical**.**

Se requiere 4 tantos de esta pieza para el robot, sobre estas piezas se colocan y adaptan los motores junto con sus cajas de engranes; posteriormente todo este arreglo de cadera-motor se coloca debajo de la base del robot.

Figura 5.18 Base rectangular.

2) Cadera

Cada cadera necesita una guía circular de 1/4” para el eje principal (antes mencionado) y dos ejes fijos como se aprecia en las ilustraciones de la figura 5.19.

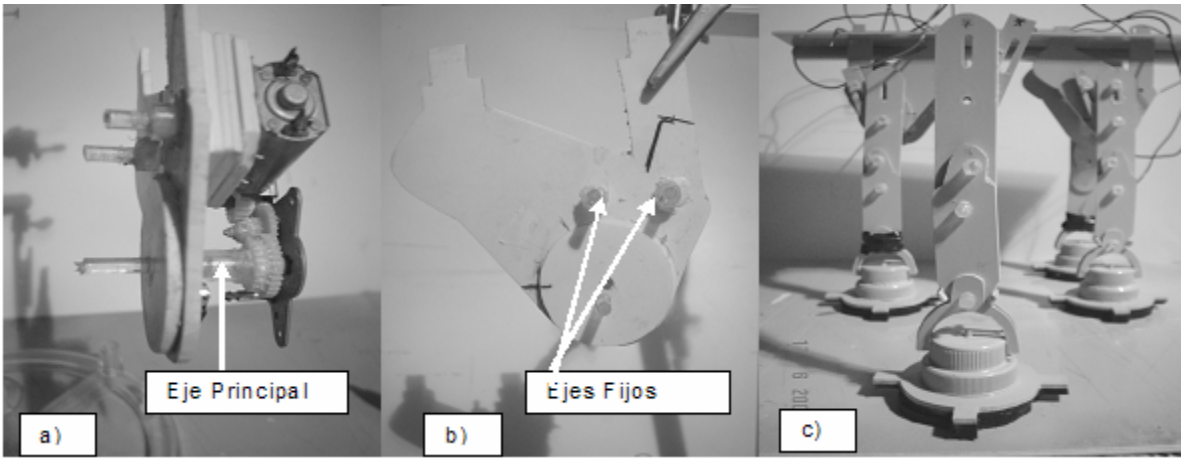


Figura 5.19 Cadera: a) eje principal instalado en la cadera; b) cadera con disco y ejes fijos; c) extremidades fijas a la base rectangular por medio de la cadera.

3) Disco con eje descentrado

Uno de los engranes necesita un eje principal al que se le coloca un disco con un eje descentrado (figura 5.20) que permita el movimiento del muslo, por lo que se requiere la construcción de cuatro de estos discos con ejes descentrados.

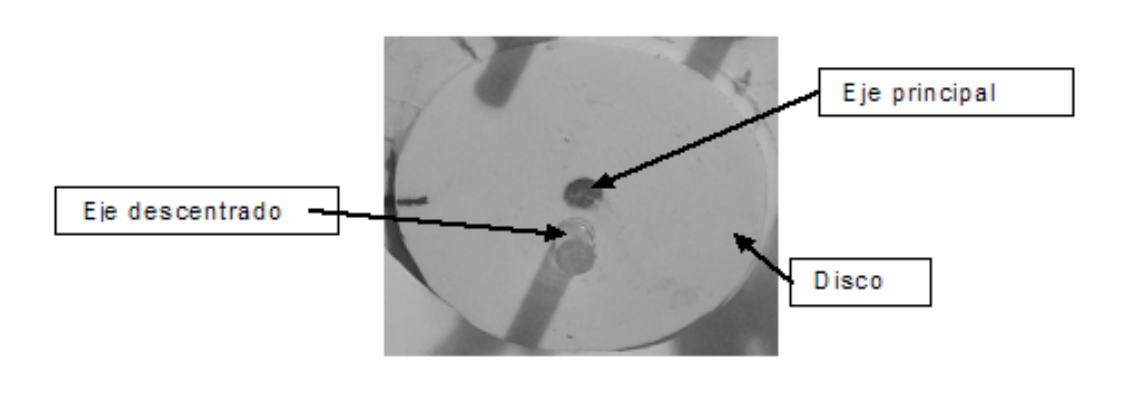


Figura 5.20 Disco con eje descentrado.

4) Muslo

Del muslo (figura 5.21) se requieren cuatro tantos. Cada uno lleva dos guías de canal, así como una guía circular de 1/4“ y un eje fijo.

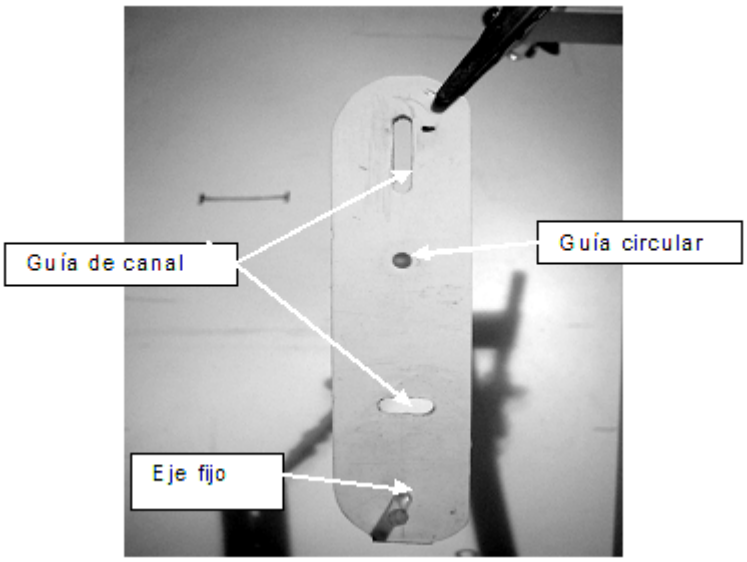


Figura 5.21 Muslo.

5) Tendón

Del tendón (figura 5.22) también se diseñan cuatro tantos. Cada uno con una guía de canal, una guía circular y un eje fijo.

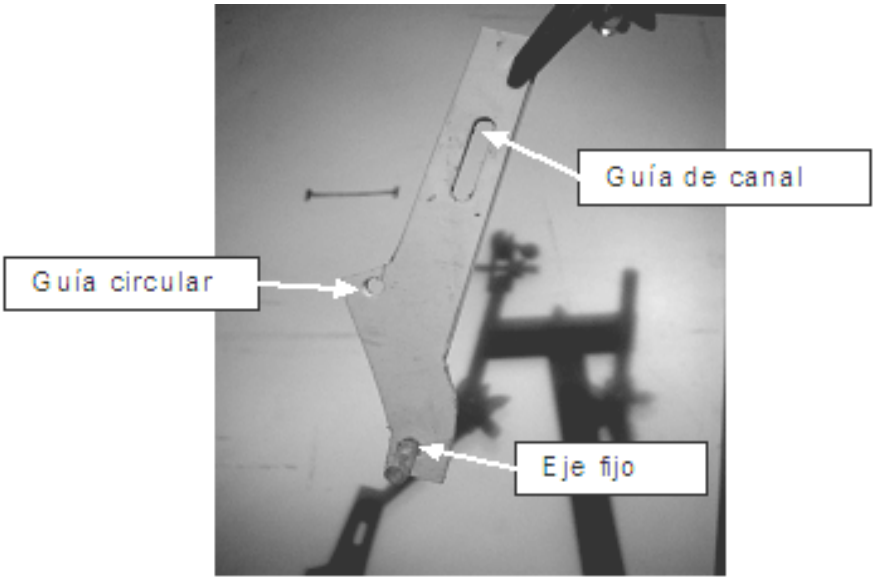


Figura 5.22 Tendón.

6) Pierna

De la pierna (figura 5.23) se requieren también cuatro piezas, cada una con tres guías circulares, cada guía circular tiene un diámetro de ¼ “ y utiliza un eje de acrílico.

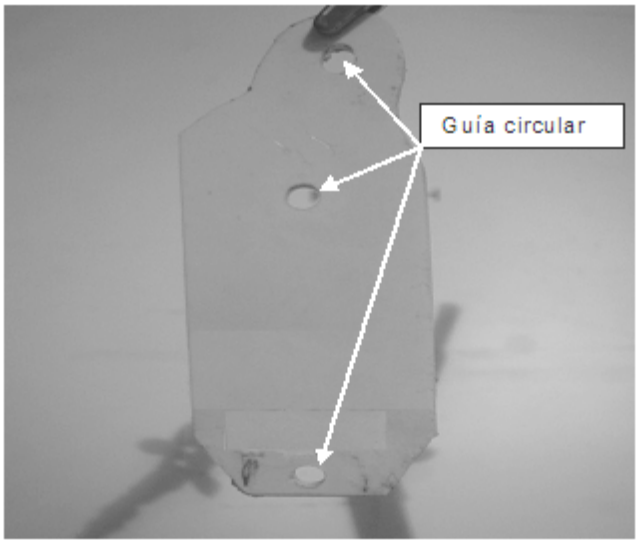


Figura 5.23 Pierna.

7) Pata

Para las patas (figura 5.24), se necesitan construir ocho arcos -dos por cada pierna- con una guía circular por arco. También, se requiere adaptar dos tipos diferentes de tapas de plástico y fijarlas a una base de estireno.

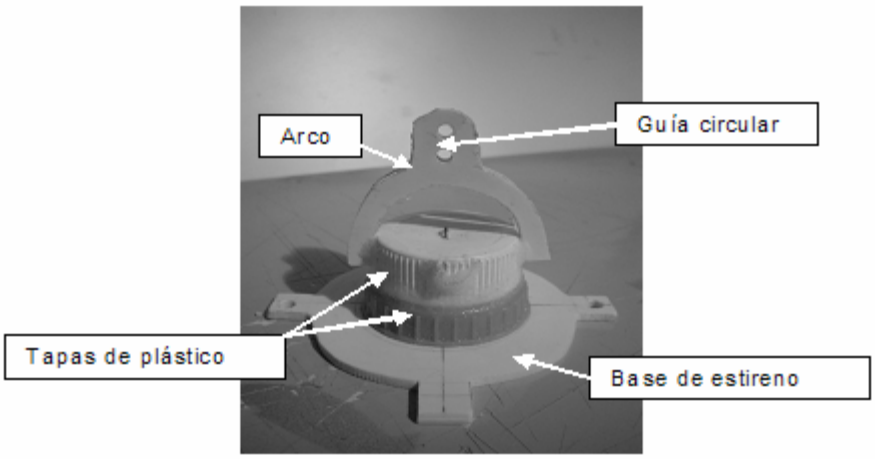


Figura 5.24 Pata.

5.2.5 Pruebas de las piezas de la extremidad

Prácticamente todas las piezas de las extremidades son sometidas a pruebas de diseño, montaje, funcionalidad y resistencia.

- Guías de canal y circulares: con la varilla dura de acrílico se prueba que la pieza no se atore con el eje, ya que cualquier fricción afecta el movimiento, ya sea en el torque, la velocidad, el avance, etc.

- Ejes: se verifica que estén bien pegados y derechos, para lo que se utiliza un tubo hueco de acrílico colocado al eje en el momento del fijado.

- Eje principal: se verifica que no se troce la varilla hueca y que pueda pasar con libertad en al agujero que se hizo en la cadera.

- Piezas: se verifica que no tengan excesos o rebabas en las orillas ni fisuras causadas por el corte o perforación.

5.3 INTEGRACIÓN Y PRUEBAS

Una vez hechas las pruebas en el módulo electrónico, de control y mecánico se unen cada uno de estos para tener la implementación final. Pero primero se debe integrar el módulo de mando remoto (receptor) con el microcontrolador.

5.3.1 Integración del receptor con el microcontrolador

Para llevar a cabo esta integración es necesario conectar el D8 (PIN 10) del decodificador de la etapa de recepción con la entrada del puerto a RA4 (PIN 3) del microcontrolador 16F84A. Para no poner cableado entre estas, se integra en la tarjeta perforada.

Una vez implementado el zócalo y los componentes necesarios para que funcione el microcontrolador, es necesario el código fuente para que se genere la secuencia de Inicio/Paro.

5.3.1.1 Código fuente de la secuencia Inicio/Paro

Tomando en cuenta el diagrama de flujo de la secuencia Inicio/Paro mostrado en el capítulo 4, se genera el siguiente programa cuyo código fuente se muestra en el *anexo 4* , el cual incluye el PWM para cada motor:

5.3.1.2 Pruebas con el microcontrolador 16F84A

Una vez conectado el pin de salida del decodificador que esta en la etapa de recepción con la entrada del RA4 del microcontrolador, se aprieta el pulsador del transmisor y entonces, el receptor manda un pulso que habilita en el microcontrolador la secuencia de Inicio (haciéndose evidente en los LEDS de salida en el microcontrolador). Al dejar de apretar el pulsador, termina la secuencia de Inicio procediendo a revisar que los pulsadores de los “bumpers” estén apretados, para finalizar la secuencia de Paro.

Con el osciloscopio se verifican las diferentes señales que se generaron para el PWM de cada motor. El tiempo entre cada motor varía por muy pocos mseg ó microseg y eso se debe a que cada motor es diferente aunque se trate del mismo modelo, ya que puede tener pequeñas variaciones propias de los materiales con que se construye, por ejemplo la resistencia de de su bobina o la intensidad de sus imanes.

5.3.1.3 Integración del microcontrolador con el L298 y el motor

Una vez integrado el receptor y el microcontrolador, se debe conectar cada salida de éste con cada entrada del L298 -ya implementado e integrado en la

tarjeta principal- procurando que la conexión entre estos permanezca de forma alámbrica.

Una vez conectado el microcontrolador al L298, se conectan a éste los motores (figura 5.25) que están integrados en cada cadera de las extremidades, los cuales permiten mover al robot cuadrúpedo. La conexión de los motores es alámbrica y se energizan cada uno con una tensión de 9 Volts y cada uno consume 0.45 Watts.

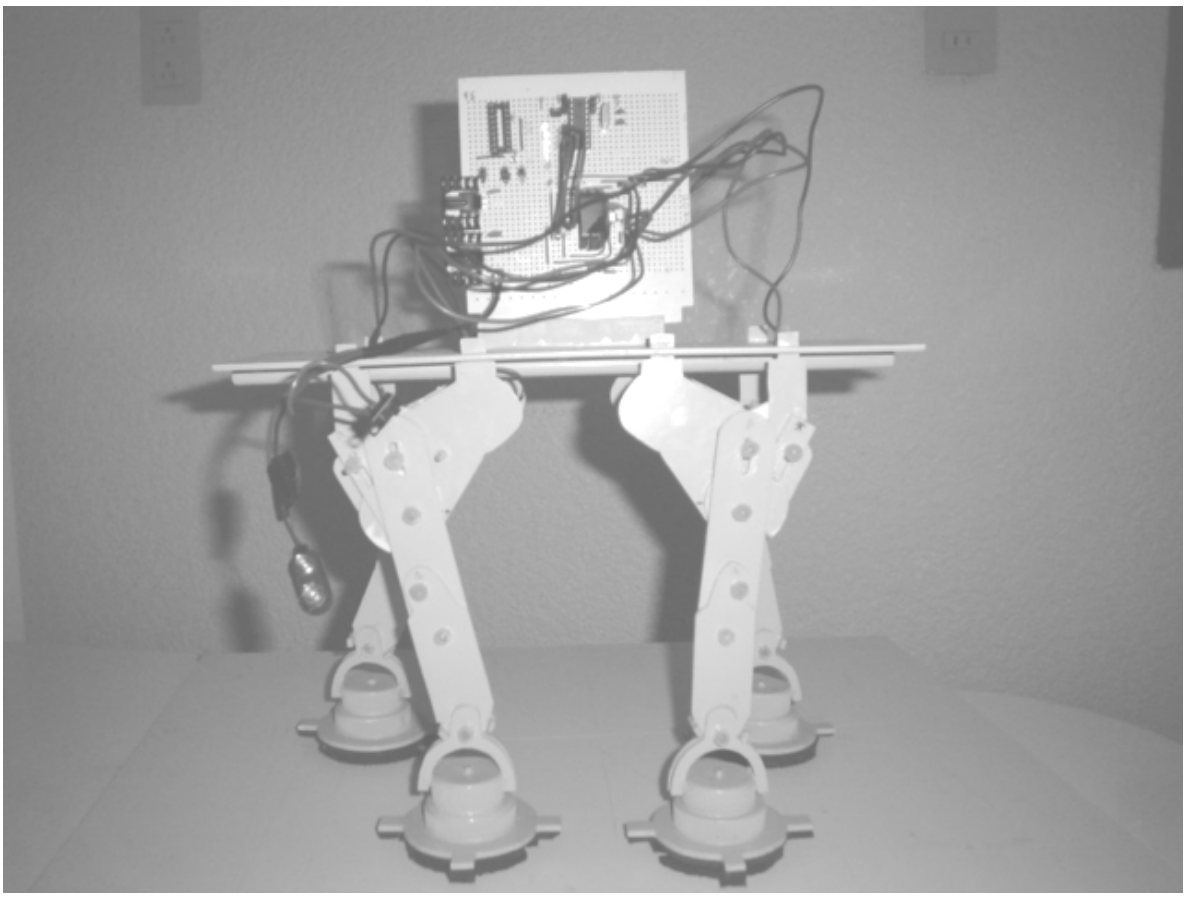


Figura 5.25 Conexión del L298 a los motores y la cadera.

5.3.1.3.1 Prueba con el L298 y el motor a la cadera en secuencia de Inicio

Para probar el L298, se integra al microcontrolador y se conectan los cuatro motores al L298; cada motor se conecta a cada pin de salida del L298. Con el

L298 se puede mostrar también la forma de funcionar del motor con la secuencia de Inicio y la secuencia de Paro.

Al activar el pulsador del transmisor, el receptor manda el pulso que activa la secuencia de Inicio en el microcontrolador, que a su vez ejecuta el programa de PWM para cada motor. El L298 recibe las señales y enciende cada motor de forma sincronizada por el microcontrolador, moviendo los engranes acoplados a cada extremidad, las cuales marcamos previamente en algún punto para saber si están ejecutando el ciclo completo de movimiento.

5.3.1.3.2 Prueba con el L298 y el motor a la cadera en secuencia de Paro

Al dejar de presionar el pulsador del transmisor, el receptor manda el pulso que activa la secuencia de Paro, lo que genera nuevamente un PWM para cada motor. El L298, opera los motores, moviendo los engranes acoplados a cada extremidad donde el motor con la ayuda del “bumper” que se integra en el eje principal le indica el momento en que la extremidad debe parar.

5.3.2 Integración de las piezas de cada extremidad

Cada extremidad está formada por seis piezas –sin contar la base rectangular que las sujeta a todas. Su integración (figura 5.26) paso a paso se origina a partir de la cadera con el eje descentrado; el eje descentrado con el tendón y el muslo; y finalmente, la pierna con la pata. Este procedimiento se realiza exactamente igual en todas las extremidades.

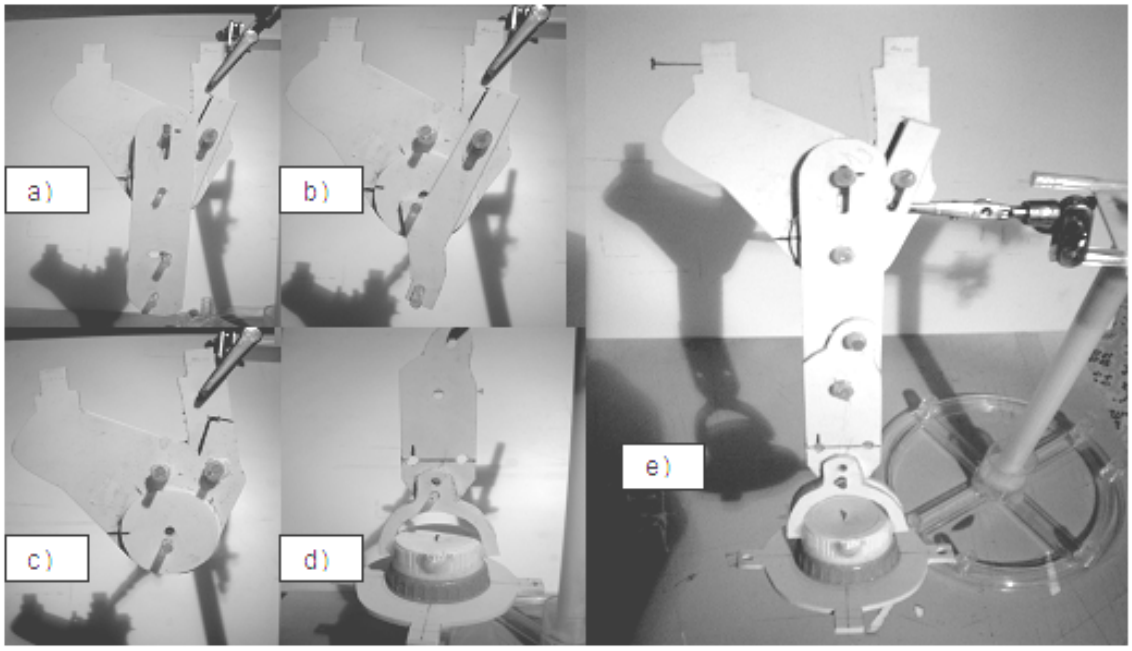


Figura 5.26 Integración de las piezas de una extremidad: a) cadera con eje descentrado. b) eje descentrado con tendón. c) eje descentrado con tendón y muslo. d) pierna con pata. e) extremidad terminada.

5.3.2.1 Prueba de la extremidad

Para esta prueba solamente se necesita verificar que las piezas no estén muy pegadas entre sí, para que la resistencia sea mínima cuando el motor se ponga en marcha. De cualquier manera y con el fin de evitar cualquier fricción, se unta grasa para engranes entre las piezas, facilitando, así, el movimiento de la extremidad.

5.3 DISEÑO FINAL

El mecanismo propuesto para controlar las extremidades del robot cuadrúpedo se basa en cuatro motores de DC (uno por cada extremidad) y electrónica, con el fin de:

- Desarrollar una trayectoria unidireccional
- Utilizar el microcontrolador PIC 16F84 para la conmutación
- Programar el 16F84 para realizar su movimiento
- Inicializar los movimientos por medio de un mando remoto
- Utilizar la configuración Puente H para la potencia en los motores

De esta forma se construyó el robot que finalmente se desarrolla de la siguiente manera:

La figura 5.27 muestra al robot cuadrúpedo unidireccional terminado y habilitado para realizar los movimientos que se establecieron como requisitos del robot, objeto del trabajo de tesis de licenciatura, quedando sus módulos electrónico y de control, así como demás elementos ocultos al interior de una carcasa decorativa.

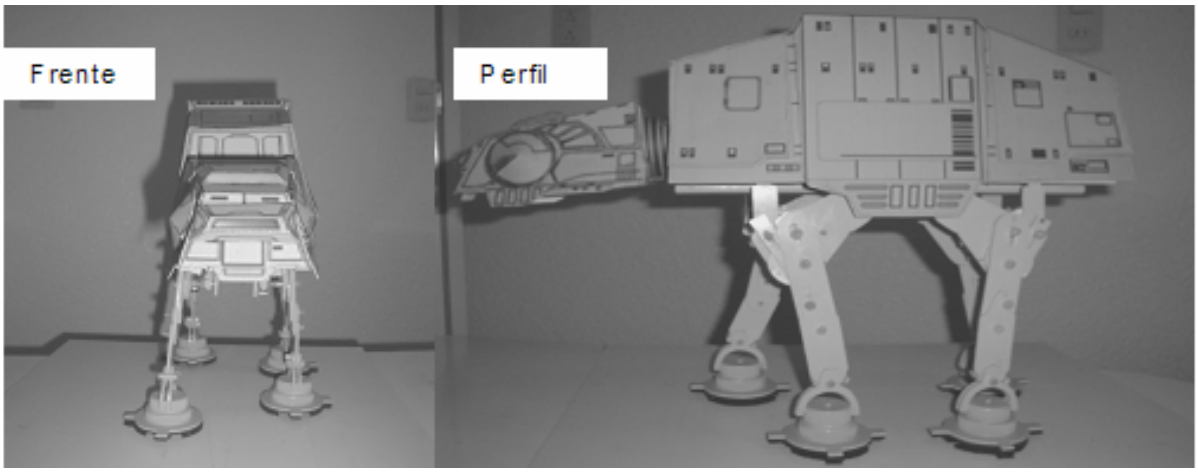


Figura 5.27 Robot Cuadrúpedo Unidireccional.

Conclusiones

La investigación del presente trabajo y el estudio de algunos modelos de robots con extremidades permitieron la implementación de un robot cuadrúpedo unidireccional, el cual se desarrolló y construyó mediante módulos. Este documento incluyendo los detalles de construcción, programas y componentes se espera ayuden a otros estudiantes en proyectos de robótica.

El diseño de control fue en lazo abierto en el cual la acción del control es independiente de la salida.

El diagrama del robot cuadrúpedo unidireccional de este trabajo quedó integrado por cada una de las etapas de diseño, -considerando cada elemento electrónico, mecánico y de mando remoto-, permitiendo llegar al planteamiento de una propuesta de solución, y así, ubicar al robot cuadrúpedo unidireccional dentro de un estándar para construir robots, usada por diseñadores de robots para ver su nivel de clasificación la cual es llamada Torrebot, donde se clasificaron los módulos ya especificados en niveles, con la característica principal de que son combinables, tanto sus dispositivos como elementos, lo que permitió ir mostrando las primeras integraciones entre estos. Con cada elemento de los módulos se mostraron las implementaciones que nos llevaron a un diagrama de bloques, acercándonos al modelo final.

Cuando todos los módulos del robot cuadrúpedo estuvieron listos en el diagrama de bloques, se detallaron las características de cada uno de los dispositivos y elementos, conociéndolos físicamente, así como sus diagramas esquemáticos. Por ejemplo, en el microcontrolador, donde se utiliza el PIC 16F84 (aunque en clase usamos el 68HC11), mostró bondades para sistemas mínimos, al ser una plataforma amigable adaptable al diseño propuesto.

El microcontrolador también genera la secuencia de Inicio/Paro (planteada en el diagrama de flujo), al programarlo en lenguaje ensamblador; lo que nos

permite la técnica PWM. Con esta técnica se calibró el tiempo que requería cada motor para la secuencia de movimiento de cada extremidad.

Para el mando remoto, se estudio en principio al mando infrarrojo, pero se recurrió a un modelo más profesional conformado por el transmisor **TWS-BS3**, con un codificador HT-12E y al receptor **RWS-371-6**, con un decodificador HT-12D, el cual se adapto correctamente para usar distancias remotas de hasta 200 metros, que en el futuro podría usarse en aplicaciones mas complejas dando base al diseño y construcción de otros proyectos.

En el aspecto de potencia se obtuvo un gran avance, ya que el dispositivo L298 nos permitió controlar al motor y acoplarlo con el microcontrolador, disminuyendo el espacio requerido por el arreglo de transistores que conforman a un Puente H.

Durante el desarrollo de esta tesis se diseñó y construyó un robot cuadrúpedo unidireccional por medio de módulos (de mando remoto, electrónico, mecánico y de control) realizándose pruebas a cada uno de estos en forma individual y finalmente en forma integrada. Se muestra el modelo final y el armado de cada uno de los módulos, donde, en el caso de la electrónica, se presenta los circuitos ensamblados en una tarjeta principal que se encuentra en el interior del cuadrúpedo; en el caso de la mecánica, se integra los motores a las cajas de engranes acoplándolas a su vez a las extremidades. Fue así que se lograron los primeros pasos del robot cuadrúpedo unidireccional y posteriormente se mejoraron con ajustes en la sincronización.

Esta tesis y su trabajo de investigación inmediatamente trabajos futuros, por ejemplo: cambiando el microcontrolador 16F84A por el 16F877, que dentro de sus puertos tiene dos canales que están asignados al PWM. En este caso, el programa se llamaría a estos canales asignándoles el tiempo por medio de otro programa realizado en lenguaje de nivel medio o alto, con menores tiempos de

programación y mayor facilidad de implementación de cambios o adición de funciones para el robot.

El software Mikropascal es un ejemplo de los programas que utilizan lenguaje de alto nivel, pero este requiere de licencias ya que el uso gratuito es limitado y los programas para el tipo de control que se necesitan requieren de la licencia del programa. Con todo esto el robot cuadrúpedo unidireccional en futuro podrá ser un control en lazo cerrado en el cual se determine la posición de sus extremidades por medio de sensores.

Otras posibilidades van desde la construcción de cuadrúpedos de menor tamaño para la realización de tareas de difícil acceso o bien de mayor tamaño para el manejo o desplazamiento de cuerpos de mayor volumen y/o peso.

En ambientes muy hostiles por condiciones del terreno, radioactividad, temperatura, humedad o quizás de acceso restringido, estos robots pueden ofrecer funciones y ventajas que los motorizados pueden tener limitadas. El subir y bajar escaleras son retos que deberán considerarse en futuros proyectos.

Anexo 1

MICROCONTROLADOR 16F84

En un principio cuando no había microcontroladores, se diseñaban circuitos que implicaban muchos componentes electrónicos y demasiados cálculos matemáticos. A partir de 1971 con el primer microprocesador se fue facilitando el diseño electrónico (sería más pequeño y simplificado).

Antes de hablar de la programación del PIC 16F84 hablaremos un poco de las características del microcontrolador.

1.1 RECURSOS COMUNES DEL MICROCONTROLADOR

En un principio los microcontroladores tenían la arquitectura básica von Neumann (con una sola memoria principal donde se almacenan datos e instrucciones de forma indistinta), que se accede por medio de un sistema de buses (direcciones, datos y control).

El microcontrolador PIC funciona con arquitectura Harvard la cual tiene 2 memorias independientes (La primera contiene solo instrucciones y la segunda datos), que además disponen de sus respectivos buses de acceso y es posible realizar operaciones de acceso (lectura o escritura) simultáneamente en ambas memorias.

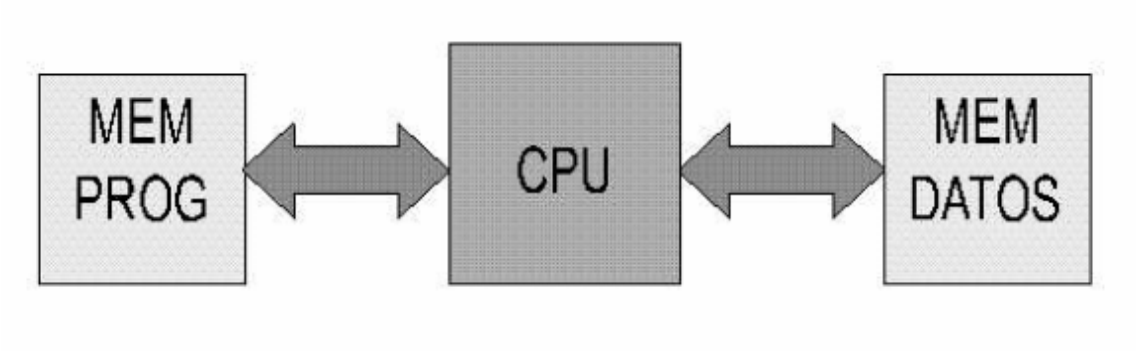


Figura 1.1: Arquitectura Harvard.

Las características de los microcontroladores integrados en un chip son muy parecidas. Deben disponer esencialmente de:

- Procesador
- Memoria de Datos y de instrucciones
- Líneas de Entrada y de Salida
- Oscilador de Reloj
- Módulos de Controladores de Periféricos.

1.1.1 Procesador o UCP

Elemento importante del microcontrolador con características importantes tanto en hardware como en software.

El procesador se encarga de:

- Direccionar la memoria de instrucciones.
- Recibe el código OP de la instrucción en curso.
- Decodifica y ejecuta la operación que implica la instrucción.
- Busca los operandos y el almacenamiento del resultado de la instrucción antes dada.

Anexo 1: Microcontrolador 16F84

Los procesadores actuales tienen 3 orientaciones según su arquitectura y funcionalidad como son:

- CISC (Computadores de Juego de Instrucciones Complejo).

Disponen de más de 80 instrucciones en su repertorio, sofisticadas y potentes y requieren muchos ciclos para ejecutarse.

Le ofrecen al programador la ventaja de instrucciones complejas que actúan como macros.

- RISC (Computadores de Juego de Instrucciones Reducido).

Favorita de la industria de las computadoras comerciales y de los microcontroladores. Las instrucciones maquina son muy reducidas, simples y se ejecutan en un ciclo, optimizando al hardware como al software del procesador.

- SISC (Computadores de Juego de Instrucciones Específico).

Microcontrolador que utiliza instrucciones muy reducidas, adaptándose a la aplicación prevista.

1.1.2 Memoria

Para el microcontrolador, la memoria de instrucciones y datos está integrada en el chip. Una parte debe ser volátil (tipo ROM), que contiene el programa de instrucciones que gobierna la aplicación. La otra parte de memoria será volátil (RAM) que guardara variables y datos de información durante el programa.

¿Cual seria la diferencia de los microcontroladores y los computadores personales?

- No existen sistemas de almacenamiento masivo (disco duro, discos flexibles)
- El microcontrolador se destina a una tarea en ROM (único programa de trabajo).

Al existir un programa activo, no se necesita guardar una copia en RAM ya que se ejecutara en ROM.

En las PC´s estamos acostumbrados a usar Megabytes de memoria, pero en los microcontroladores se usan capacidades de ROM entre 512 bytes y 8 K bytes. Para el caso de la memoria RAM entre 20 y 512 bytes.

Dependiendo de la memoria ROM en los microcontroladores, sus aplicaciones y la utilización de estos difiere, porque se tienen que describir las cinco versiones de memoria no volátil de los microcontroladores en el mercado que se nombran a continuación:

-ROM con máscara.

Memoria no volátil de solo lectura, la cual se graba en el momento en el que se esta fabricando el chip y solo se fabrican en caso de cantidades de varios miles de unidades.

- One Time Programmable (OTP).

Memoria no volátil de solo lectura, que graba el usuario una sola vez desde una computadora (donde se hará el programa que se escribirá en un grabador).

Para proteger el código contenido se usa la encriptación mediante fusibles.

“La memoria OTP es recomendada en prototipos”.

- Erasable Programmable Read Only Memory (EPROM).

Esta memoria puede borrarse y grabarse muchas veces, esta funciona gracias a un grabador gobernado desde una PC. En el caso de quererlo borrar por una ventana de cristal en su superficie se somete a la EPROM a rayos ultravioleta por varios minutos. El encapsulado de la EPROM es cerámico por lo que son más caros que los OTP que son hechos de plástico.

- Electrical Erasable Programmable Read Only Memory (EEPROM).

Esta memoria de solo lectura, se programa y se borra eléctricamente desde un grabador y bajo el control programado de una PC. Esta memoria es muy práctica, pero no disponen de ventana de cristal en la superficie.

Los microcontroladores con memoria EEPROM al instalarse en el circuito, pueden grabarse y borrarse un sin numero de veces sin ser retirados del circuito ya que cuenta con grabadores de circuito, que ayuda en las modificaciones en el programa de trabajo.

Las memorias EEPROM pueden grabarse y borrarse de manera finita por lo que no se recomienda la reprogramación continua.

Este tipo de memoria es relativamente lenta.

- FLASH.

Memoria no volátil de bajo consumo, que puede escribir y borrar.
Funciona como ROM y RAM consumiendo menos energía y es más pequeña. Puede programarse en el circuito, siendo más rápida y de mayor densidad que la EEPROM.

"**La Memoria Flash tolera varios ciclos de escritura/ borrado".**

1.1.3 Puertas de entrada y salida

Una de las características del microcontrolador es soportar líneas de entrada/salida que comunican al computador interno con los periféricos exteriores, proporcionando el soporte a las señales de entrada, salida y control.

1.1.4 Reloj principal

Todos los microcontroladores disponen de un circuito oscilador, que genera una onda cuadrada de alta frecuencia y como reloj sincroniza todas las operaciones del sistema.

EL circuito reloj esta incorporado al microcontrolador, necesitando de muy pocos componentes exteriores (cristal de cuarzo, resonador magnético, red R-C) para seleccionar y estabilizar la frecuencia de trabajo.

Cuando se aumenta la frecuencia de reloj disminuye el tiempo de ejecución de las instrucciones, incrementando la energía.

1.2 RECURSOS ESPECIALES

Al diseñar se debe encontrar el microprocesador que tenga el modelo mínimo que satisfaga los requerimientos de la aplicación, para reducir costo, así como el hardware y el software.

Los principales recursos específicos que incorporan los microcontroladores son:

- Temporizadores ó Timers.
- Perro guardián ó Watchdog.
- Protección ante fallo de alimentación o Brownout.
- Estado de reposo o de bajo consumo.
- Conversor A/D.
- Conversor D/A.
- Comparador analógico.
- Modulador de ancho de pulsos ó PWM.
- Puertas E/S digitales.
- Puertas de comunicación.

1.2.1 Temporizadores o timers

Controlan periodos de tiempos (temporizadores), para llevar la cuenta de acontecimientos del exterior (contadores).

Para medir el tiempo se carga un registro con el valor adecuado y este se va incrementando o disminuyendo al impulso del reloj o algún múltiplo hasta que se desborde y llegue a cero, momento en el que se produce un aviso.

Para contar acontecimientos que serán cambios de nivel o flancos en alguna pata del microcontrolador, el registro se va incrementando ó disminuyendo al ritmo de los impulsos.

1.2.2 Perro guardián o watchdog

Si la PC se bloquea, ya sea por el software u otra razón, se pulsa el botón reset que reinicia el sistema. El perro guardián consiste en un temporizador que, cuando se desborda y pasa por cero, se resetea todo el sistema.

La modalidad de perro guardián se debe programar antes del reset, ya que si se bloquea provocara que se use el reset.

1.2.3 Protección ante fallo de alimentación o brownout

Circuito que resetea al microcontrolador en caso de que el voltaje de alimentación (VDD) es inferior a un voltaje mínimo (del Inglés “brownout”) y comenzara a funcionar cuando sobrepase dicho valor.

1.2.4 Estado de reposo o de bajo consumo

En el caso de que el microcontrolador este en espera de un acontecimiento externo para funcionar, se dispone de una instrucción especial en los PIC (como un SLEEP), que los pasa a un estado de bajo consumo, deteniendo el reloj principal de este y congelando sus circuitos asociados.

Al activarse una interrupción ocasionada por algún acontecimiento esperado, el microcontrolador se despierta y reanuda su trabajo.

1.2.5 Convertidor A/D (CAD)

Al tener un convertidor Analógico/ Digital (CAD), pueden procesar señales analógicas. También dispone de un multiplexor que permite aplicar a la entrada del CAD diversas señales analógicas desde las patas del circuito integrado.

1.2.6 Comparador analógico

Algunos microcontroladores internamente poseen un Amplificador Operacional que actúa como comparador entre una señal fija y otra variable. La salida del comparador genera un nivel lógico 1 ó 0 según sea mayor o menor que la otra.

En algunos microcontroladores tienen un módulo de tensión de referencia que proporciona diversas tensiones de referencia que se puede aplicar a los comparadores.

1.2.7 Moduladores de anchura de pulsos O PWM

Son circuitos que proporcionan en su salida impulsos de anchura variable, que se ofrecen al exterior en patas del encapsulado.

1.2.8 Puertas de e/s digitales

Los microcontroladores destinan algunas patas a líneas E/S digitales, alineadas de ocho en ocho formando puertas.

Las líneas digitales se pueden configurar como Entrada o Salida por medio de un 1 ó 0 en el bit que corresponda el registro destinado a su configuración.

1.2.9 Puertas de comunicación

Para que el microcontrolador se comunique con dispositivos externos, buses tanto de microprocesadores como de sistemas ó de redes y que se adapten con otros elementos a otras normas y protocolos. Se cuenta con modelos que permiten esta tarea como son:

- Adaptador de Comunicación Serie Asíncrona (UART)
- Adaptador de Comunicación seria sincronía y asíncrona (USART)
- Puerta paralela esclava para conectar los buses de otros microprocesadores.
- Bus de Serie Universal (USB), muy usado en PC.
- Interfaz serie de dos hilo (Bus I^2C) creado por Philips
- Red de Área de Control (CAN), para adaptar redes de conexionado multiplexado usado en automóviles y desarrollado por Bosch e Intel.

1.3 HERRAMIENTAS PARA EL DESARROLLO DE APLICACIONES

En un microcontrolador es importante el soporte tanto en software como en hardware que disponga y las herramientas de desarrollo es decisivo ya que supone una ayuda inestimable en el desarrollo de un proyecto.

1.3.1 Desarrollo del software

Las principales herramientas de ayuda al desarrollo del sistema, basados en microcontroladores son:

Ensamblador. La programación en lenguaje ensamblador es ardua para el principiante, pero desarrolla programas eficientes ya que se puede dominar el sistema. Los fabricantes proporcionan gratis el programa ensamblador.

Compilador. La programación en un lenguaje de alto nivel (como el C), permite disminuir el tiempo de desarrollo del producto, pero si no se programa con cuidado, el código resultante puede ser más ineficiente que el programado en ensamblador. Las versiones más potentes son muy caras y las versiones demo están limitadas, aunque también se encuentran compiladores gratuitos.

1.3.2 Depuración

Como los microcontroladores van a controlar dispositivos físicos, los desarrolladores necesitan herramientas para comprobar el funcionamiento del microcontrolador al ser conectado con otros circuitos.

Simulador. Ejecutan en una PC programas realizados para el microcontrolador. Los simuladores permiten controlar el programa siendo ideales para la depuración del mismo. El único inconveniente es la simulación de entrada y salida de datos del microcontrolador (sin contar los ruidos en la entrada), pero aun con eso permiten el paso físico de la implementación de modo seguro y menos costoso, ya que ahorraremos en grabaciones de chips para la prueba in-situ.

Placas de evaluación. Se trata de pequeños sistemas con un microcontrolador ya montado, que se conecta a una PC cargando los programas que ejecuta el microcontrolador. Las placas pueden incluir visualizadores LCD, teclados, LEDs, acceso fácil a los pines de E/S, etc.

El sistema operativo (PROGRAMA MONITOR), carga programas y datos en la memoria del microcontrolador, permitiendo la ejecución paso a paso verificando el estado del microcontrolador o modifica valores almacenados en la memoria.

Emuladores en circuito. Instrumento que se coloca entre la PC y el zócalo de la tarjeta del circuito impreso, donde se va a alojar el microcontrolador

definitivo. El programa es ejecutado desde la PC, pero para la tarjeta de aplicación es como si lo hiciese el microcontrolador que ira en el zócalo. Presenta en pantalla toda la información tal y como se va a almacenar en el encapsulado.

1.4 EL MICROCONTROLADOR "PIC"

Los microcontroladores PIC son baratos, rápidos, fáciles de usar, resumiendo que son sencillos y útiles. Pero hay que constatar que otros microcontroladores son más eficientes en aplicaciones específicas, en el caso de que este fabricado para una característica concreta y pueda desarrollarlo mejor otra familia.

Los detalles importantes que hacen al PIC una buena opción para los diseñadores son:

Sencillez de manejo: Con un juego reducido de instrucciones sumando 35 en la gama media.

Buena información, fácil de conseguir y económica.

Precio: Su precio es inferior a los competidores.

Elevada Velocidad de Funcionamiento. Con un buen promedio de parámetros como son: velocidad, consumo, tamaño, alimentación, código compacto, etc.

Herramientas de desarrollo fáciles y baratas. Muchas herramientas software se pueden recoger libremente a través de Internet desde Microchip.

Variedad de herramientas hardware. Estas nos permiten grabar, depurar, borrar y comprobar el comportamiento de los PIC.

Diseño rápido.

Variedad de PIC´s. Nos permite elegir el que mejor responde a los requerimientos de la aplicación.

1.4.1 Características relevantes

La variedad de modelos de microcontroladores PIC permite al usuario seleccionar el más conveniente para su proyecto.

A continuación se describen las características más representativas de los PIC.

1.4.1.1 Arquitectura

Basada en el modelo Harvard (donde el CPU se conecta en forma independiente, con buses distintos con la memoria de instrucciones y con la de datos), que permite al CPU acceder simultáneamente a las 2 memorias. Mencionado al principio del anexo.

1.4.1.2 Segmentación

En este caso se utiliza la técnica de segmentación “ Pipe-Line” para ejecutar las instrucciones, las cuales le permitirá al procesador ejecutar una instrucción y buscar el código de la siguiente, ejecutando cada instrucción en un ciclo (un ciclo de instrucción equivale a cuatro ciclos de reloj).

1.4.1.3 Formato de las Instrucciones

"El formato de las instrucciones es de la misma longitud"

Para las instrucciones de los microcontroladores de la gama baja tienen una longitud de 12 bits. Las de gama media tienen 14 bits y mas las de la gama alta (Importante en la optimización de la memoria de instrucciones y facilita la construcción de ensambladores y compiladores.

1.4.1.4 Juego de instrucciones

Procesador RISC (Computador de juego de instrucciones Reducido).

1Para el caso de los modelos de la gama baja disponen un repertorio de 33 instrucciones, 35 los de la gama media y casi 60 los de alta.

1.4.1.5 Instrucciones ortogonales

Toda instrucción ortogonal maneja cualquier elemento de la arquitectura como fuente o como destino.

1.4.1.6 Arquitectura basada en un "banco de registros"

Donde todos los objetos del sistema (puertas de E/S, temporizadores, posiciones de memoria, etc.) están implementados físicamente como registros.

1.4.1.7 Herramientas de soporte

Las empresas que usan los PIC ofrecen a los usuarios numerosas herramientas para desarrollar hardware y software. Afortunadamente hay abundantes programadores, simuladores de software, emuladores en tiempo real, Ensambladores, Compiladores C, Intérpretes y Compiladores Basic, etc.

La arquitectura Harvard y la técnica de segmentación son los principales recursos que genera un elevado rendimiento en estos dispositivos programables, mejorando la velocidad de ejecución y la eficiencia en la compactación del código.

1.5 LAS GAMAS DE PIC

La compañía Microchip pone a disposición del diseñador desde el microcontrolador sencillo y barato (para atender aplicaciones simples), así como otros complejos y costosos (para aplicaciones de mucha envergadura).

Microchip dispone de cuatro familias de microcontroladores de 8 bits para adaptarse a las necesidades de la mayoría de los clientes potenciales.

1.5.1 La gama enana: PIC12C(F)XXX de 8 patitas

PIC de reciente aparición que ha llamado la atención por su reducido tamaño y tener todos sus componentes en 8 patitas. Su Voltaje de alimentación esta comprendido entre los 2.5 Vcc y 5.5 Vcc consumiendo menos de 2mA cuando trabajan a 5V y 4 Mhz. Su formato de instrucciones puede ser de 12 ó 14 bits y contando con un repertorio de 33 ó 35 instrucciones, respectivamente.

Aunque los PIC enanos solo tienen 8 patitas, 6 de ellas pueden ser destinadas como líneas de E/S para los periféricos, ya que disponen de un oscilador interno de R-C.

1.5.2 Gama baja ó básica: PIC16C5X con instrucciones de 12 bits

PIC de recursos limitados, con una buena relación coste/prestaciones. Sus versiones están encapsuladas con 18 y 28 patitas, alimentándose con una tensión de 2.5 Vcc, lo que las hace ideales en aplicaciones que funcionan con pilas teniendo en cuenta su bajo consumo (menos de 2 mA a 5 Volts y 4 Mhz). Cuenta con un repertorio de 33 instrucciones cuyo formato consta de 12 bits.

"NO ADMITEN NINGÚN TIPO DE INTERRUPCIÓN Y LA PILA SOLO DISPONE DE 2 NIVELES"

Los componentes de la gama baja se caracterizan por poseer los siguientes recursos:

Sistema de Reinicialización (del Inglés Power On Reset ó POR): En donde el PIC genera una auto reinicialización al conectar la alimentación.

Perro Guardián (del Inglés Watchdog ó WDT): Existe un temporizador que produce un reset automático si no se recarga antes de un tiempo prefijado.

Código de Protección: Al grabar el programa se puede proteger contra lectura, así como número de serie, códigos de identificación, prueba, etc.

Líneas de E/S de alta corriente: Donde las líneas de E/S proporcionan o absorben una salida de 20 y 25 mA, suficiente para excitar a los periféricos.

Modo de reposo (Bajo consumo ó del Inglés SLEEP): Al ejecutarse SLEEP, la CPU y el oscilador principal reduciendo notablemente su consumo de energía.

1.5.2.1 Restricciones de los componentes de la gama baja

La Pila (del Inglés STACK): Dispone de 2 niveles en las cuales no puede encadenar más de 2 rutinas.

Los microcontroladores de la gama baja **NO ADMITEN INTERRUPCIONES**.

1.5.3 Gama media. PIC16CXXX con instrucciones de 14 bits

Gama variada y completa de los PIC. Cuenta con modelos encapsulados desde 18 patitas hasta 68 cubriendo a varios periféricos.

EN ESTA GAMA SE ENCUENTRA EL PIC16F84, QUE ES EL QUE SE UTILIZA EN ESTA TESIS.

Los componentes de esta gama tienen nuevas prestaciones a los de la gama baja, haciéndolos adecuados para aplicaciones complejas.

Estos microcontroladores aceptan interrupciones, poseen comparadores de magnitudes analógicas, convertidores A/D, puertos serie y diversos temporizadores.

El repertorio de instrucciones es de 35 (14 bits cada una) y compatible con la gama baja.

Encuadrado en la gama media se encuentra el PIC14C00, con el cual se diseñan controladores para cargadores de baterías y cualquier sistema adquisición y procesamiento de señales que requiera de alimentación, aceptando cualquier tecnología de las baterías como Li-Ion, NiMH, Ni-Cd, Ph y Zinc.

En esta gama se tiene un temporizador TMR1, el cual por medio de un oscilador trabaja asíncronamente del cual la aplicación más popular es la implementación de un reloj en tiempo real.

Las líneas de E/S presentan una carga “ pull-up” activada por software.

1.5.4 Gama alta: PIC17CXXX con instrucciones de 16 bits

Alcanzan las 58 instrucciones de 16 bits en el repertorio, disponiendo de interrupciones vectorizadas muy potentes, a la vez contiene varios controladores de periféricos, puertas de comunicación serie y paralelo con elementos externos, un multiplicador hardware de gran velocidad y mayores capacidades de memoria, que alcanza los 8K palabras en la memoria de instrucciones y 454 bytes en la memoria de datos. Esta gama cuenta con una arquitectura abierta (es decir, que puede ampliar al microcontrolador con elementos externos), por lo que tiene un numero elevado de patitas (entre 40 y 44), que se empleaba mas en microprocesadores que en microcontroladores.

1.6 LOS REGISTROS DE LA GAMA MEDIA

1.6.1 Organización de la memoria de datos

Para utilizar los recursos del microcontrolador (Puertos, Timers, ALU, etc), el microcontrolador posee una serie de posiciones de memoria en RAM, llamadas registros de control, que permiten controlar ciertas partes del chip.

Los registros para el 16F84 son los siguientes:

Desde la posición 0Ch hasta la 4Fh tenemos posiciones de memoria libres que se pueden utilizar como variables (donde podemos acceder desde las direcciones 8Ch hasta CFh).

REGISTROS:

Dir.	Registros Pagina 0	Registros Pagina 1	Dir
00h	INDF	INDF	80h
01h	TMR0	OPTION	81h
02h	PCL	PCL	82h
03h	STATUS	STATUS	83h
04h	FSR	FSR	84h
05h	PORTA	TRISA	85h
06h	PORTB	TRISB	86h
07h	----	----	87h
08h	EEDATA	EECON1	88h
09h	EEADR	EECON2	89h
0Ah	PCLATCH	PCLATCH	8Ah
0Bh	INTCON	INTCON	8Bh

Figura 1.2: Registros del PIC 16F84.

- ***STATUS (03h ,83h):*** Registro de estado donde se encuentra la información básica del integrado, como el estado de los banderines Z, C y DC y la página de RAM que estamos utilizando. La descripción de sus bits es la siguiente:

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
IRP	RP1	RP0	-TO	-PD	Z	DC	C

Figura1.3: Registro STATUS.

IRP: Debe ser siempre cero ya que no se usa en el PIC 16F84.

RP1:RP0: Estos bits se utilizan para saber en que página estamos (00 = Página 0, 01=Página 1, 10 = Página 2, 11=Página 3). Para el caso del PIC 16F84 solo tiene 2 páginas por lo que RP1 siempre valdrá 0 y RP0 indicara la página en la que estamos.

-TO: Indica si el Watch-dog esta activo.

-PD: Si vale 0 esta en modo de standbye y si vale 1 esta recién encendido o tras CLRWDT.

Z: Si esta casilla vale 1, el resultado de la última operación aritmética es 0. En caso que la casilla vale 0, el resultado es distinto de 0.

DC: Si el valor de esta casilla es 1, existe un acarreo del bit más significativo en la última operación.

C: Si el valor de esta casilla es 1, existe un acarreo del bit más significativo en la última operación.

- ***OPTION (81h).***

La descripción de sus bits es la siguiente:

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
-RBPU	INTEDG	T0CS	T0SE	PSA	PS2	PS1	PS0

2.- Figura 3.4: Registro OPTION.

-RBPU: Si el valor de la casilla es 0, el puerto B tendrá resistencias de pull-up activadas cuando estén configuradas como entradas. Si el valor de la casilla es 1 estarán desactivadas las resistencias de pull-up.

INTEDG: Selecciona el flanco por donde se activara la interrupción del pin RBO/INT. SI INTEDG=1 se hará por flanco de subida, si es 0 el origen será por flanco de bajada.

T0CS: Indica el origen de los pulsos de clk para el timer; si es 1 el origen es la señal introducida por el pin RA4/T0CLK, si es 0 el origen será el ciclo de instrucción (CLKOUT = Frecuencia Xtal/4).

T0SE: Indica el flanco por el que se incrementará la cuenta del contador TMR0. Si vale 0 se hará por el flanco de bajada, si es 1 se hará por el flanco de subida.

PSA: El chip tiene internamente un divisor, este se puede asignar mediante este bit al WDT con un 1 o al TMR con un 0.

PS2:PS0: Este es el valor del Divisor de frecuencia, según la siguiente tabla de valores.

Valor PS2:PS0	TMR0	WDT
000	1:2	1:1
001	1:4	1:2
010	1:8	1:4
011	1:16	1:8
100	1:32	1:16
101	1:64	1:32
110	1:128	1:64
111	1:256	1:128

Figura 1.5: Tabla de Valores del Divisor de Frecuencias.

- ***INTCONT (0Bh, 8Bh):***

La descripción de sus bits es la siguiente:

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
GIE	EEIE	T0IE	INTE	RBIE	T0IF	INTF	RBIF

Figura 1.6: Registro INTCONT.

GIE: Si el valor en la casilla es 1, activa el permiso para interrupciones pero si el valor en la casilla es 0, no se permitirán las interrupciones.

EEIE: Si el valor en la casilla es 1, permite que se genere una interrupción cuando termine de escribir en la EEPROM.

T0IE: Si el valor de la casilla es 1, permite una interrupción cada vez que el TMR pase de 255 a 0.

INTE: Si el valor de la casilla es 1, permite la interrupción en el pin RB0/INT.

RBIE: Si el valor de la casilla es 1, permite la interrupción por cambio en el estado de los pines RB7:RB4.

T0IF: Si el valor de la casilla es 1, hay una interrupción en el TMR.

INTF: SI la casilla esta en 1, se produjo una interrupción porque al menos unote los bits RB7:RB4 ha cambiado de estado.

- ***PCL (02h, 82h):***

Este registro contiene el valor de los 8 bits menos significativos del contador del programa.

- ***PCLATCH (0Ah, 8Ah):***

En este match almacenamos la parte alta del contador del programa.

- ***INDF y FSR:***

INDF se utiliza para un direccionamiento indirecto, es decir que el valor que leamos o escribamos en el será el valor escrito o leído en el registro cuya dirección esta en FSR.

Si FSR = 0Ch y leemos INDF, lo que obtengamos será el valor almacenado 0ch, así como si escribimos 5 en INDF tendremos un 5 en el registro 0Ch.

- ***TMR0:***

Contiene el contador de 8 bits del timer 0.

- ***TRISA, TRISB :***

Sirve para indicar cuales pines son de salida poniendo un 0 en el correspondiente bit, y en el caso que sean de entrada poniendo un 1 en el correspondiente bit.

- ***PORTA,PORTB:***

Sirve para indicar si un determinado pin del puerto, que esta configurado como salida, esta a 1 ó a 0.

- ***EECON1, EECON2, EEDATA Y EEADR:***

Sirve para gestionar la transferencia de datos con la EEPROM

- ***PCON:***

Es el registro identificador del Reset.

						R/W /POR	R/W /BO

Figura 1.7: Registro PCON.

/POR: Funciona como el Reset por activación del micro (del Inglés Power on Reset mencionado anteriormente), el cual se activara si el valor de la casilla es 0.

/BO: Identificado para la caída de tensión (del Ingles Brown-Out), el cual se activara si el valor de la casilla es 0

1.7 REPERTORIO DE INSTRUCCIONES

1.7.1 Características generales

Por la filosofía RISC en el PIC 16F84, esta consta de 33 instrucciones simples en la gama baja (las cuales son sencillas y rápidas), que se ejecutan en un ciclo de máquina (equivalente a 4 del reloj principal).

Las instrucciones cuentan con 3 tipos de direccionamiento como son:

1ro Inmediato: EL valor del dato esta incluido en el propio código OP, junto a la instrucción.

2do Directo: La dirección del dato está incluido en el propio código OP, junto a la instrucción.

3ero Indirecto: La dirección de la memoria de datos que guarda el operando esta contenida en un registro.

Para el caso de la gama media consta con 35 instrucciones, idénticas a los de la gama baja, en los que podemos encontrar 5 tipos de etiquetas:

f: Registro de memoria con valor entre 00h y 7fh

W: Registro interno de trabajo del PIC.

b: Con esta etiqueta se puede saber el bit de referencia en las instrucciones de manipulación dentro de un registro de 8 bits.

k: Etiqueta que equivale a constante.

d: El resultado de esta etiqueta define el almacenamiento, ya sea en W (d=0), ó en f (d=1)

PC: Contador de programa que se encarga de direccionar la memoria de instrucciones.

TOS: Cima de la pila, con 2 niveles en la gama baja y 8 niveles en la media.

WDT: Perro Guardián (Watchdog).

Dest: Referido al destino ya sea W ó f.

TO: Bit " Time Out " del registro de estado.

PD: Bit " Power Down " del registro de estado.

x: Valor indeterminado (puede ser un 0 o un 1), pero para mantener la compatibilidad con la marca Microchip, conviene que x = 0.

Label: Nombre de la etiqueta.

[]: Opciones.

(): Contenido.

$\rightarrow$***:*** Se asigna a.

<>: Campo de bits de un registro.

$\in$***:*** Pertenece al conjunto.

Z: Señalador de Cero en W que pertenece al registro de estado.

C: Señalizador del acarreo en el octavo bit del W que pertenece al registro de estado.

DC: Señaliza el 4 bit del acarreo en W, que pertenece al registro de estado.

Itálicas: Términos definidos por el usuario.

Repertorio de instrucciones de la gama media

1.7.3 Instrucciones de la gama baja

ADDLW Suma un literal

Sintaxis: [label] ADDLW k
Operandos: 0 ≤ k ≤ 255
Operación: : (W) + (k)⇒ (W)
Flags afectados: C, DC, Z
Código OP: 11 111x kkkk kkkk

Descripción: Suma el contenido del registro W y k, guardando el resultado en W.

Ejemplo: ADDLW 0xC2

Antes: W = 0x17
Después: W = 0xD9

ADDWF W + F

Sintaxis: [label] ADDWF f,d
Operandos: d e [0,1], 0 ≤ f ≤ 127
Operación: (W) + (f) ⇒ (dest)
Flags afectados: C, DC, Z
Código OP: 00 0111 dfff ffff

Descripción: Suma el contenido del registro W y el registro f. Si d es 0, el resultado se almacena en W, si d es 1 se almacena en f.

Ejemplo: ADDWF REG,0

Antes: W = 0x17., REG = 0xC2
Después: W = 0xD9, REG = 0xC2

ANDLW W AND literal

Sintaxis: [label] ANDLW k
Operandos: 0 ≤ k ≤ 255
Operación: : (W) AND (k)⇒ (W)
Flags afectados: Z
Código OP: 11 1001 kkkk kkkk

Descripción: Realiza la operación lógica AND entre el contenido del registro W y k, guardando el resultado en W.

Ejemplo: ADDLW 0xC2

Antes: W = 0x17
Después: W = 0xD9

ANDWF W AND F

Sintaxis: [label] ANDWF f,d
Operandos: d e [0,1], 0 ≤ f ≤ 127
Operación: (W) AND (f) ⇒ (dest)
Flags afectados: Z
Código OP: 00 0101 dfff ffff

Descripción: Realiza la operación lógica AND entre los registros W y f. Si d es 0, el resultado se almacena en W, si d es 1 se almacena en f.

Ejemplo: : ANDWF REG,0

Antes: W = 0x17., REG = 0xC2
Después: W = 0x17, REG = 0x02

BCF Borra un bit

Sintaxis: [label] BCF f,b
Operandos: 0 ≤ f ≤ 127, 0 ≤ b ≤ 7
Operación: : 0 ⇒ (f<b>)
Flags afectados: Ninguno
Código OP: 01 00bb bfff ffff

Descripción: Borra el bit b del registro f

Ejemplo: : BCF REG,7

Antes: REG = 0xC7
Después: REG = 0x47

BSF Activa un bit

Sintaxis: [label] BSF f,b
Operandos: 0 ≤ f ≤ 127,, 0 ≤ b ≤ 7
Operación: 1 ⇒ (f<b>)
Flags afectados: Ninguno
Código OP: 01 01bb bfff ffff

Descripción: Activa el bit b del registro f

Ejemplo: : BSF REG,7

Antes: REG = 0x0A
Después: REG = 0x8A

BTFSC Test de bit y salto

Sintaxis: [label] BTFSC f,d
Operandos: d e [0,1], 0 ≤ f ≤ 127
Operación: Salto si (f<b>) = 0
Flags afectados: Ninguno
Código OP: 01 10bb bfff ffff

Descripción: Si el bit b del registro f es 0, se salta una instrucción y se continúa con la ejecución. En caso de salto, ocupará dos ciclos de reloj.

Ejemplo:
```
BTFSC REG,6
GOTO NO_ES_0
SI_ES_0 Instrucción
NO_ES_0 Instrucción
```

BTFSS Test de bit y salto

Sintaxis: [label] BTFSS f,d
Operandos: d e [0,1], 0 ≤ f ≤ 127
Operación: Salto si (f<b>) = 1
Flags afectados: Ninguno
Código OP: 01 11bb bfff ffff

Descripción: Si el bit b del registro f es 1, se salta una instrucción y se continúa con la ejecución. En caso de salto, ocupará dos ciclos de reloj.

Ejemplo:
```
BTFSS REG,6
GOTO NO_ES_0
SI_ES_0 Instrucción
NO_ES_0 Instrucción
```

CALL Salto a subrutina

Sintaxis: [label] CALL k
Operandos: 0 ≤ k ≤ 2047
Operación: PC ⇒ Pila; k ⇒ PC
Flags afectados: Ninguno
Código OP: 10 0kkk kkkk kkkk

Descripción: Salto a una subrutina. La parte baja de k se carga en PCL, y la alta en PCLATCH. Ocupa 2 ciclos de reloj.

Ejemplo: ORIGEN CALL DESTINO

Antes: PC = ORIGEN
Después: PC = DESTINO

CLRF Borra un registro

Sintaxis: [label] CLRF f
Operandos: 0 ≤ f ≤ 127
Operación: : 0x00 ⇒ (f), 1 ⇒ Z
Flags afectados: Z
Código OP: 00 0001 lfff ffff

Descripción: El registro f se carga con 0x00. El flag Z se activa.

Ejemplo: : CLRF REG

Antes: REG = 0x5A
Después: REG = 0x00, Z = 1

CLRW Borra el registro W

Sintaxis: [label] CLRW
Operandos: Ninguno
Operación: : 0x00 ⇒ W, 1 ⇒ Z
Flags afectados: Z
Código OP: 00 0001 0xxx xxxx

Descripción: El registro de trabajo W se carga con 0x00. El flag Z se activa.

Ejemplo: : CLRW

Antes: W = 0x5A
Después: W = 0x00, Z = 1

CLRWDT Borra el WDT

Sintaxis: [label] CLRWDT
Operandos: Ninguno
Operación: 0x00 ⇒ WDT, 1 ⇒ /TO
1 ⇒ /PD
Flags afectados: /TO, /PD
Código OP: 00 0000 0110 0100
Descripción: Esta instrucción borra tanto el WDT como su preescaler. Los bits /TO y /PD del registro de estado se ponen a 1.

Ejemplo: : CLRWDT
Después: Contador WDT = 0,
Preescales WDT = 0,
/TO = 1, /PD = 1

COMF Complemento de f

Sintaxis: [label] COMF f,d
Operandos: d ∈ [0,1], 0 ≤ f ≤ 127
Operación: : (/ f), 1 ⇒ (dest)
Flags afectados: Z
Código OP: 00 1001 dfff ffff

Descripción: El registro f es complementado. El flag Z se activa si el resultado es 0. Si d es 0, el resultado se almacena en W, si d es 1 se almacena en f..

Ejemplo: : COMF REG,0

Antes: REG = 0x13
Después: REG = 0x13, W = 0XEC

DECF Decremento de f

Sintaxis: [label] DECF f,d
Operandos: d ∈ [0,1], 0 ≤ f ≤ 127
Operación: : (f) – 1 ⇒ (dest)
Flags afectados: Z
Código OP: 00 0011 dfff ffff

Descripción: Decrementa en 1 el contenido de f. Si d es 0, el resultado se almacena en W, si d es 1 se almacena en f.

Ejemplo: : DECF CONT,1

Antes: CONT = 0x01, Z = 0
Después: CONT = 0x00, Z = 1

DECFSZ Decremento y salto

Sintaxis: [label] DECFSZ f,d
Operandos: d ∈ [0,1], 0 ≤ f ≤ 127
Operación: (f) -1 ⇒ d; Salto si R=0
Flags afectados: Ninguno
Código OP: 00 1011 dfff ffff
Descripción: Decrementa el contenido del registro f. Si d es 0, el resultado se almacena en W, si d es 1 se almacena en f. Si la resta es 0 salta la siguiente instrucción, en cuyo caso costaría 2 ciclos.
Ejemplo: DECFSC REG,0
GOTO NO_ES_0
SI_ES_0 Instrucción
NO_ES_0 Salta instrucción anterior

GOTO Salto incondicional

Sintaxis: [label] GOTO k
Operandos: 0 ≤ k ≤ 2047
Operación: k ⇒ PC <8:0>
Flags afectados: Ninguno
Código OP: 10 1kkk kkkk kkkk

Descripción: Se trata de un salto incondicional. La parte baja de k se carga en PCL, y la alta en PCLATCH. Ocupa 2 ciclos de reloj.

Ejemplo: ORIGEN GOTO DESTINO

Antes: PC = ORIGEN
Después: PC = DESTINO

INCF Decremento de f

Sintaxis: [label] INCF f,d
Operandos: d ∈ [0,1], 0 ≤ f ≤ 127
Operación: : (f) + 1 ⇒ (dest)
Flags afectados: Z
Código OP: 00 1010 dfff ffff

Descripción: Incrementa en 1 el contenido de f. Si d es 0, el resultado se almacena en W, si d es 1 se almacena en f.
Ejemplo: : INCF CONT,1

Antes: CONT = 0xFF, Z = 0
Después: CONT = 0x00, Z = 1

INCFSZ Incremento y salto

Sintaxis: [label] INCFSZ f,d
Operandos: d ∈ [0,1], 0 ≤ f ≤ 127
Operación: (f) -1 ⇒ d; Salto si R=0
Flags afectados: Ninguno
Código OP: 00 1111 dfff ffff

Descripción: Incrementa el contenido del registro f. Si d es 0, el resultado se almacena en W, si d es 1 se almacena en f. Si la resta es 0 salta la siguiente instrucción, en cuyo caso costaría 2 ciclos.
Ejemplo: INCFSC REG,0
GOTO NO_ES_0
SI_ES_0 Instrucción
NO_ES_0 Salta instrucción anterior

IORLW W OR literal

Sintaxis: [label] IORLW k
Operandos: 0 ≤ k ≤ 255
Operación: : (W) OR (k) ⇒ (W)
Flags afectados: Z
Código OP: 11 1000 kkkk kkkk

Descripción: Se realiza la operación lógica OR entre el contenido del registro W y k, guardando el resultado en W.

Ejemplo: : IORLW 0x35

Antes: W = 0x9A
Después: W = 0xBF

IORWF W AND F

Sintaxis: [label] IORWF f,d
Operandos: d ∈ [0,1], 0 ≤ f ≤ 127
Operación: (W) OR (f) ⇒ (dest)
Flags afectados: Z
Código OP: 00 0100 dfff ffff

Descripción: Realiza la operación lógica OR entre los registros W y f. Si d es 0, el resultado se almacena en W, si d es 1 se almacena en f.

Ejemplo: : IORWF REG,0

Antes: W = 0x91, REG = 0x13
Después: W = 0x93, REG = 0x13

MOVLW Cargar literal en W

Sintaxis: [label] MOVLW f
Operandos: 0 ≤ f ≤ 255
Operación: (k) ⇒ (W)
Flags afectados: Ninguno
Código OP: 11 00xx kkkk kkkk

Descripción: El literal k pasa al registro W.

Ejemplo: MOVLW 0x5A

Después: REG = 0x4F, W = 0x5A

MOVF Mover a f

Sintaxis: [label] MOVF f,d
Operandos: d ∈ [0,1], 0 ≤ f ≤ 127
Operación: (f) ⇒ (dest)
Flags afectados: Z
Código OP: 00 1000 dfff ffff
Descripción: El contenido del registro f se mueve al destino d. Si d es 0, el resultado se almacena en W, si d es 1 se almacena en f. Permite verificar el registro, puesto que afecta a Z.

Ejemplo: MOVF REG,0

Después: W = REG

MOVWF Mover a f

Sintaxis: [label] MOVWF f
Operandos: 0 ≤ f ≤ 127
Operación: W ⇒ (f)
Flags afectados: Ninguno
Código OP: 00 0000 1fff ffff

Descripción: El contenido del registro W pasa el registro f.

Ejemplo: MOVWF REG,0

Antes: REG = 0xFF, W = 0x4F
Después: REG = 0x4F, W = 0x4F

NOP No operar

Sintaxis: [label] NOP
Operandos: Ninguno
Operación: No operar
Flags afectados: Ninguno
Código OP: 00 0000 0xx0 0000

Descripción: No realiza operación alguna. En realidad consume un ciclo de instrucción sin hacer nada.

Ejemplo: : CLRWDT

Después: Contador WDT = 0,
Preescales WDT = 0,
/TO = 1, /PD = 1

RETFIE Retorno de interrup.

Sintaxis: [label] RETFIE
Operandos: Ninguno
Operación: : 1 ⇒ GIE; TOS⇒PC
Flags afectados: Ninguno
Código OP: 00 0000 0000 1001

Descripción: El PC se carga con el contenido de la cima de la pila (TOS): dirección de retorno. Consume 2 ciclos. Las interrupciones vuelven a ser habilitadas.

Ejemplo: : RETFIE

Después: PC = dirección de retorno
GIE = 1

RETLW Retorno, carga W

Sintaxis: [label] RETLW k
Operandos: 0 ≤ k ≤ 255
Operación: : (k)⇒ (W); TOS⇒PC
Flags afectados: Ninguno
Código OP: 11 01xx kkkk kkkk

Descripción: El registro W se carga con la constante k. El PC se carga con el contenido de la cima de la pila (TOS): dirección de retorno. Consume 2 ciclos.

Ejemplo: : RETLW 0x37

Después: PC = dirección de retorno
W = 0x37

RETURN Retorno de rutina

Sintaxis: [label] RETURN
Operandos: Ninguno
Operación: : TOS ⇒ PC
Flags afectados: Ninguno
Código OP: 00 0000 0000 1000

Descripción: El PC se carga con el contenido de la cima de la pila (TOS): dirección de retorno. Consume 2 ciclos.

Ejemplo: : RETURN

Después: PC = dirección de retorno

RLF Rota f a la izquierda

Sintaxis: [label] RLF f,d
Operandos: d ∈ [0,1], 0 ≤ f ≤ 127
Operación: Rotación a la izquierda
Flags afectados: C
Código OP: 00 1101 dfff ffff

Descripción: El contenido de f se rota a la izquierda. El bit de menos peso de f pasa al carry (C), y el carry se coloca en el de mayor peso. Si d es 0, el resultado se almacena en W, si d es 1 se almacena en f.

Ejemplo: RRF REG,0

Antes: REG = 1110 0110, C = 0
Después: REG = 1110 0110,
W = 1100 1100, C = 1

RRF Rota f a la derecha

Sintaxis: [label] RRF f,d
Operandos: d ∈ [0,1], 0 ≤ f ≤ 127
Operación: Rotación a la derecha
Flags afectados: C
Código OP: 00 1100 dfff ffff

Descripción: El contenido de f se rota a la derecha. El bit de menos peso de f pasa al carry (C), y el carry se coloca en el de mayor peso. Si d es 0, el resultado se almacena en W, si d es 1 se almacena en f.

Ejemplo: RRF REG,0

Antes: REG = 1110 0110, C = 1
Después: REG = 1110 0110,
W = 01110 0011, C = 0

SLEEP Modo bajo consumo

Sintaxis: [label] SLEEP
Operandos: Ninguno
Operación: 0x00⇒ WDT, 1 ⇒ /TO
0 ⇒ WDT Preescaler, 0 ⇒ /PD
Flags afectados: /PD, /TO
Código OP: 00 0000 0110 0011

Descripción: El bit de energía se pone a 0, y a 1 el de descanso. El WDT y su preescaler se borran. El micro para el oscilador, yendo al modo "durmiente".

Ejemplo: : SLEEP

Preescales WDT = 0,
/TO = 1, /PD = 1

SUBLW Resta Literal - W

Sintaxis: [label] SUBLW k
Operandos: 0 ≤ k ≤ 255
Operación: (k) - (W) ⇒ (W)
Flags afectados: Z, C, DC
Código OP: 11 110x kkkk kkkk
Descripción: Mediante el método del complemento a dos el contenido de W es restado al literal. El resultado se almacena en W.

Ejemplos: SUBLW 0x02

Antes:W=1,C=?. Después: W=1, C=1
Antes:W=2,C=?. Después: W=0, C=1
Antes:W=3,C=?.Después:W=FF,C=0
(El resultado es negativo)

SUBWF Resta f – W

Sintaxis: [label] SUBWF f,d
Operandos: d ∈ [0,1], 0 ≤ f ≤ 127
Operación: (f) – (W) ⇒ (dest)
Flags afectados: C, DC, Z
Código OP: 00 0010 dfff ffff
Descripción: Mediante el método del complemento a dos el contenido de W es restado al de f. . Si d es 0, el resultado se almacena en W, si d es 1 se almacena en f.

Ejemplos: SUBWF REG,1
Antes: REG = 0x03, W = 0x02, C = ?
Después:REG=0x01, W = 0x4F, C=1
Antes: REG = 0x02, W = 0x02, C = ?
Después:REG=0x00, W =0x02, C= 1
Antes: REG= 0x01, W= 0x02, C= ?
Después:REG=0xFF, W=0x02, C= 0
(Resultado negativo)

SWAPF Intercambio de f

Sintaxis: [label] SWAPF f,d
Operandos: d ∈ [0,1], 0 ≤ f ≤ 127
Operación: : (f <3: 0>)⇔ (f <7:4>)
Flags afectados: Ninguno
Código OP: 00 1110 dfff ffff

Descripción: Los 4 bits de más peso y los 4 de menos son intercambiados. Si d es 0, el resultado se almacena en W, si d es 1 se almacena en f.

Ejemplo: : SWAPF REG,0

Antes: REG = 0xA5
Después: REG = 0xA5, W = 0x5A

XORLW W OR literal

Sintaxis: [label] XORLW k
Operandos: 0 ≤ k ≤ 255
Operación: : (W) XOR (k)⇒ (W)
Flags afectados: Z
Código OP: 11 1010 kkkk kkkk

Descripción: Se realiza la operación lógica XOR entre el contenido del registro W y k, guardando el resultado en W.

Ejemplo: : XORLW 0xAF

Antes: W = 0xB5
Después: W = 0x1A

XORWF W AND F

Sintaxis: [label] XORWF f,d
Operandos: d ∈ [0,1], 0 ≤ f ≤ 127
Operación: (W) XOR (f) ⇒ (dest)
Flags afectados: Z
Código OP: 00 0110 dfff ffff

Descripción: Realiza la operación lógica XOR entre los registros W y f. Si d es 0, el resultado se almacena en W, si d es 1 se almacena en f.

Ejemplo: : XORWF REG,0

Antes: W = 0xB5, REG = 0xAF
Después: W = 0xB5, REG = 0x1A

La gama media tiene un total de 35 instrucciones, cada una de las cuales ocupan 14 bits.

1.73 INSTRUCCIONES DE LA GAMA BAJA

La gama baja no tiene 4 de las instrucciones de la gama media. Estas son ADDLW, RETFIE, RETURN y SUBLW. En cambio, y puesto que los registros OPTION y TRIS no son accesibles, se añaden las dos siguientes:

OPTION y TRIS no son accesibles, se añaden las dos siguientes:

OPTION Carga del registro option

Sintaxis: [label] OPTION
Operandos: Ninguno
Operación: (W)⇒ OPTION
Flags afectados: Ninguno
Código OP: 0000 0000 0010

Descripción: : El contenido del registro W se carga en el registro OPTION, registro de sólo lectura en el que se configura el funcionamiento del preescaler y el TMR0.

Ejemplo: : OPTION
Antes: W= 0x06, OPTION = 0x37
Después: W= 0x06, OPTION = 0x06

TRIS Carga del registro TRIS

Sintaxis: [label] TRIS f
Operandos: 5 ≤ f ≤ 7
Operación: (W)⇒ Registro TRIF<f>
Flags afectados: Ninguno
Código OP: 0000 0000 0fff

Descripción: El contenido del registro W se carga en el registro TRISA, TRISB o TRISC, según el valor de f. Estos registros de sólo lectura configuran las patillas de un puerto como de entrada o salida.

Ejemplo: : TRIS PORTA
Antes: W=0xA5, TRISA=0x56

Observe que el código OP de las instrucciones de la gama baja sólo ocupa 12 bits, que no corresponden por lo tanto con la gama media.

Otras diferencias fundamentales entre ambas gamas son:

- Que en la gama baja en el vector de reset, la primera línea que se ejecuta tras un reset es la última de la memoria correspondiente, mientras que en la gama media es la 0x00.

- El vector de interrupción (situado en la 0x04) no existe en la gama baja.

- En la gama baja se necesita emplear los bits PA2-PA0 como parte alta de la dirección de programa en las instrucciones CALL y GOTO.

- Las páginas de la gama baja son más pequeñas, y se deben revisar todas las escrituras de los registros OPTION, ESTADO y FSR en la conversión del código.

Por lo tanto, fácilmente comprobamos que el código entre ambas gamas no es 100% compatible.

Anexo 2

MPLAB

2.1 CREANDO UN NUEVO PROYECTO

Un proyecto es un conjunto de archivos fuente e instrucciones que permiten construir el objeto y código ejecutable para una aplicación.

Para esto usaremos el ambiente de desarrollo integrado MPLAB:

Figura 2.1 Presentación del Ensamblador MPLAB.

Para abrir un nuevo documento daremos clic sobre FILE - >NEW, como se muestra en la siguiente figura:

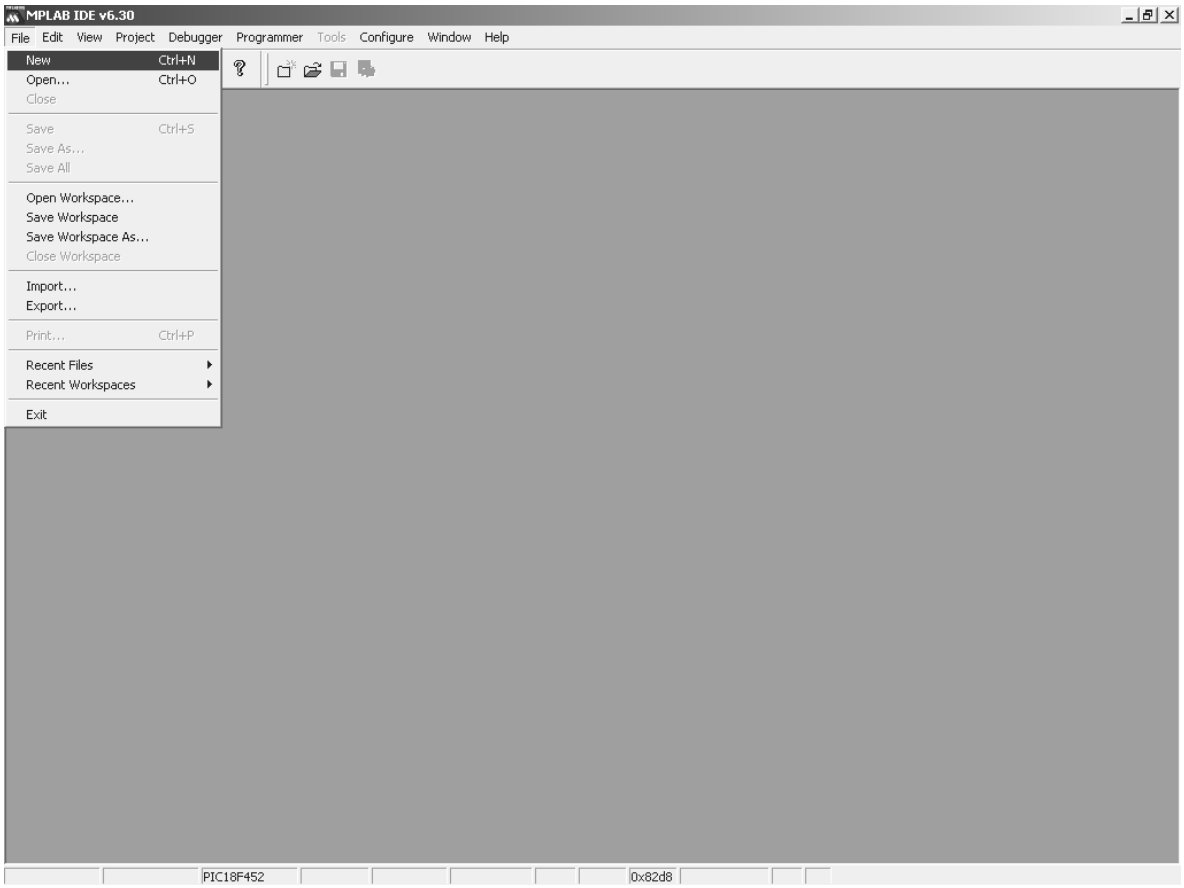


Figura 2.2 Abrir nuevo documento con FILE - >NEW.

Otra forma de abrir un documento nuevo es por medio de las teclas CTRL.+ N y aparece la pantalla del nuevo documento de forma directa como se muestra a continuación:

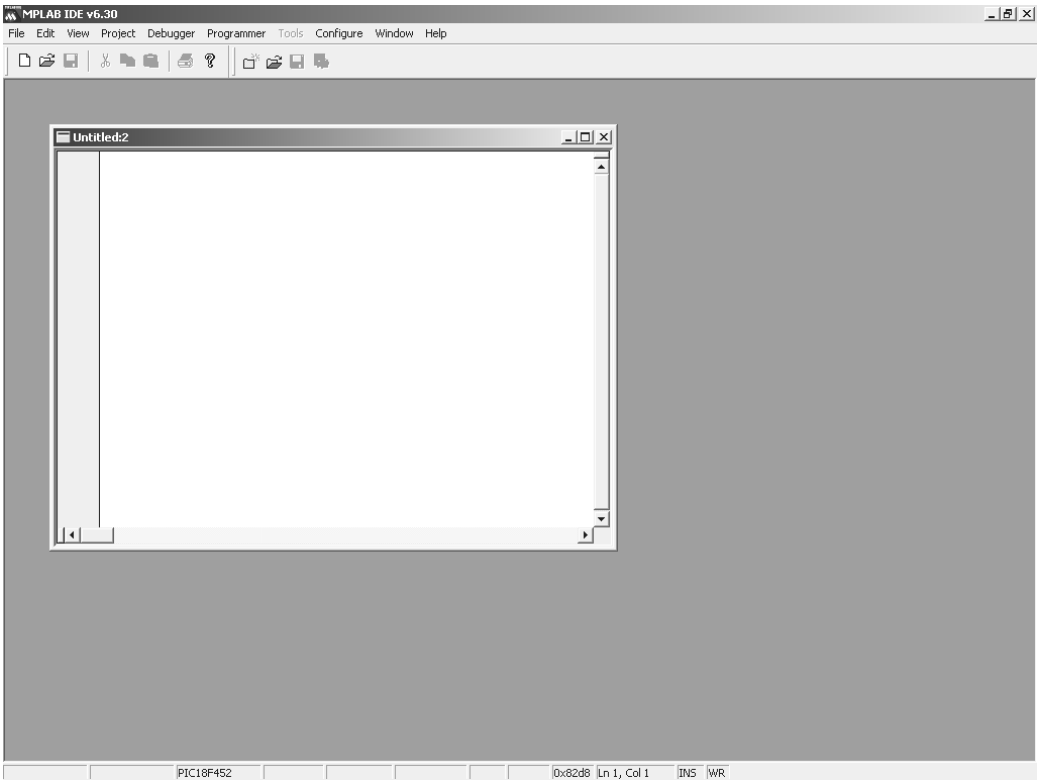


Figura 2.3 Abrir nuevo documento con CTRL.+ N.

Con esto se tiene un área donde trabajar y en donde se edita el archivo fuente, no ahondaremos en el tema de la programación pero a continuación se lista un ejemplo de un archivo fuente:

```
LIST P=PIC16F84

 INCLUDE <P16f84.INC>
 RADIX HEX

AUX EQU 0X0C

 org 0
 goto INICIO
 org 5

INICIO      bsf     STATUS,5
            movlw       0X1F
```

```
                movwf       PORTA
                movlw       0X00
                movwf       PORTB
                bcf         STATUS,5

                clrf    PORTB

EXP             movf    PORTA,w
                movwfAUX
                comf    AUX,w
                movwf   PORTB
                goto    EXP
 END
```

Figura 2.4 Archivo fuente desde un editor de texto.

Que una vez capturado en el editor queda de la siguiente forma:

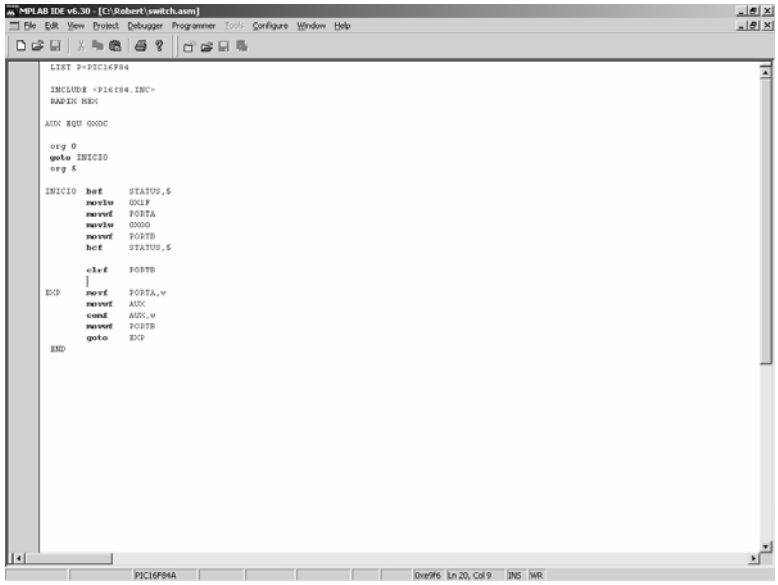


Figura 2.5 Archivo fuente desde el editor en MPLAB.

En este punto se salva el programa con FILE - > SAVE y se le pone un nombre, con la extensión *.ASM.

2.2 ENSAMBLANDO

Una vez capturado se compila con PROJECT - > QUICK BUILD o ALT + F10

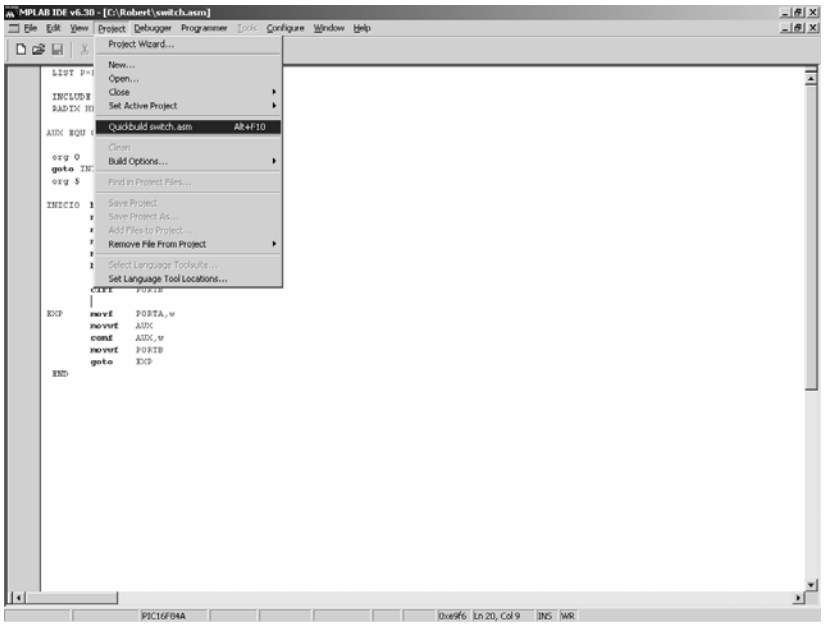


Figura 2.6 Compilando en MPLAB.

Si el programa tuviera algún error, aparece una barra en color rojo sobre la última ventana abierta como se muestra a continuación:

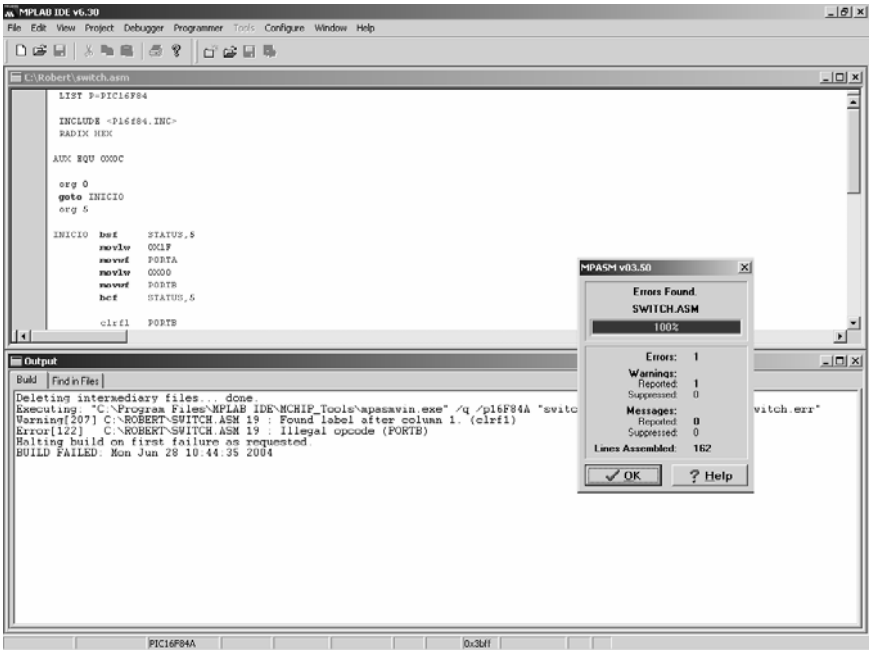


Figura 2.7 Error al compilar.

Una vez corregidos los errores aparece esta misma ventana pero ahora con una barra verde indicando que no hay errores.

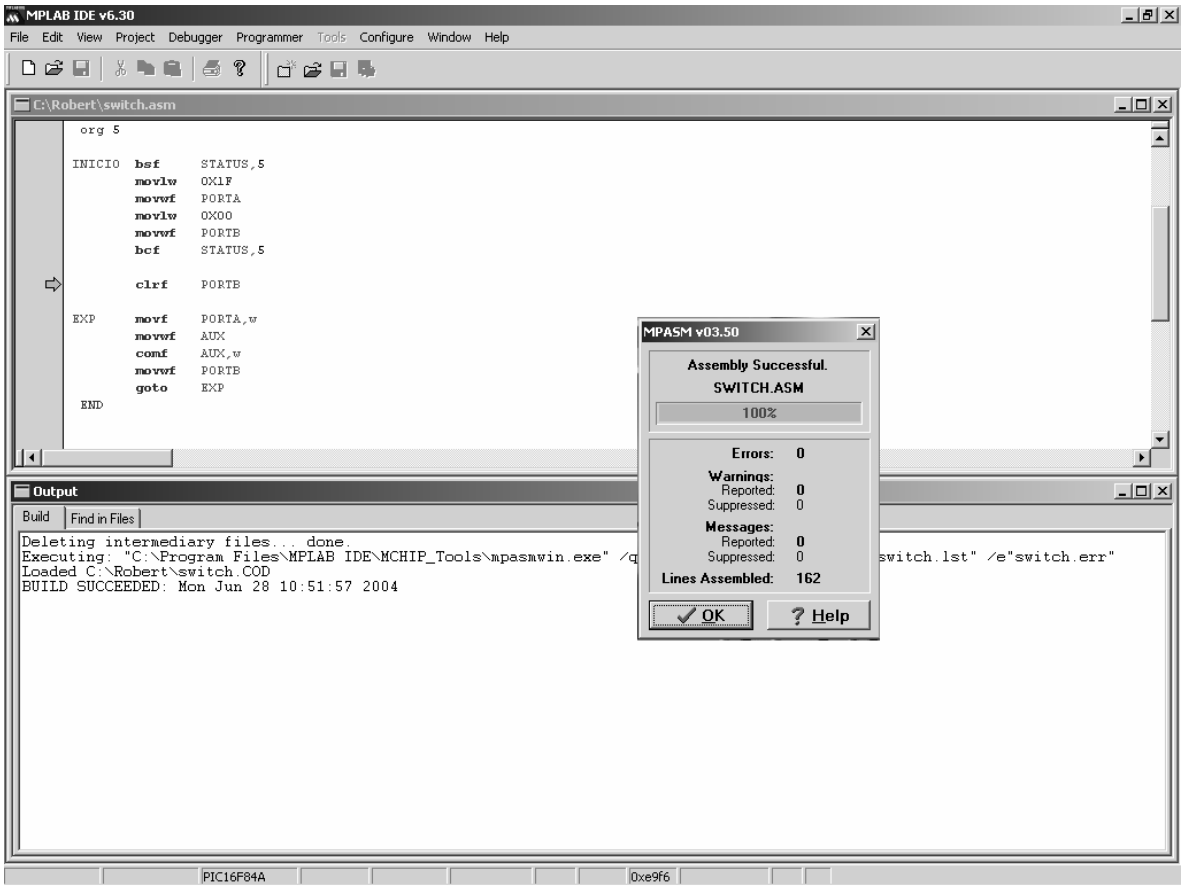


Figura 2.8 Ensamblado satisfactorio.

En la misma carpeta donde fue salvado el archivo se generarán otros archivos con el mismo nombre pero con diferentes extensiones, el más importante para nosotros es el *.HEX, que utilizaremos en el programa IC-Prog.

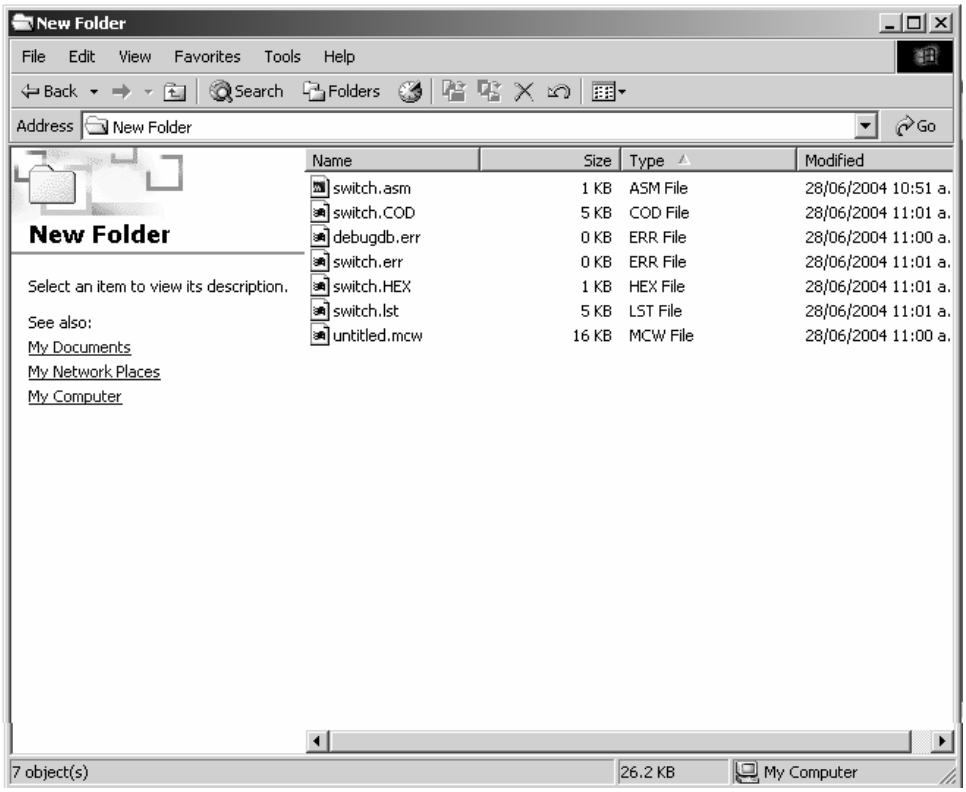


Figura 2.9 Archivo con diferentes extensiones.

Anexo 3

GRABADOR

3.1| GRABADOR uJDM

El grabador uJDM es uno de los programadores de PICs más simples.

El diseño viene del programador de bajo costo JDM hecho por Jens Dyekjær Madsen. Este se conecta directamente al puerto serial, utilizando el software IC-PROG. Este grabador no requiere de energizase de forma externa. El diseño mostrado en la siguiente figura y que es limitado para el 16F84A.

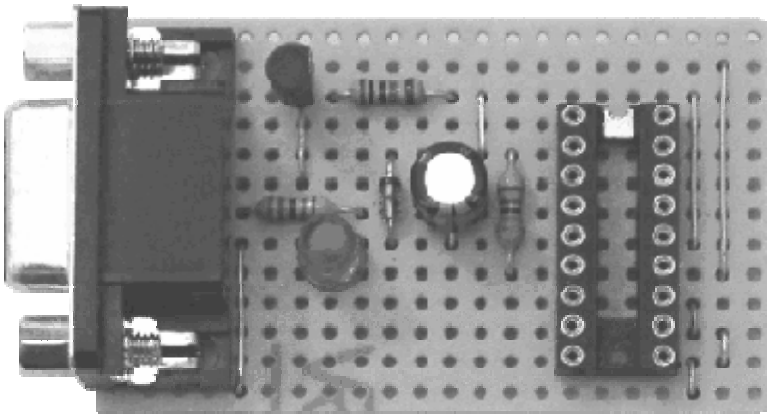


Figura 3.1 Grabador uJDM.

3.1.1 Partes del Circuito uJDM

El circuito uJDM se conecta a la computadora por el puerto serial a través del conector DB9F hembra. R3 y el LED son opcionales pero son útiles para saber si el circuito esta bien conectado al puerto serial.

En la siguiente figura se muestra el diagrama esquemático del grabador uJDM con los valores de los componentes que se necesitan para su construcción.

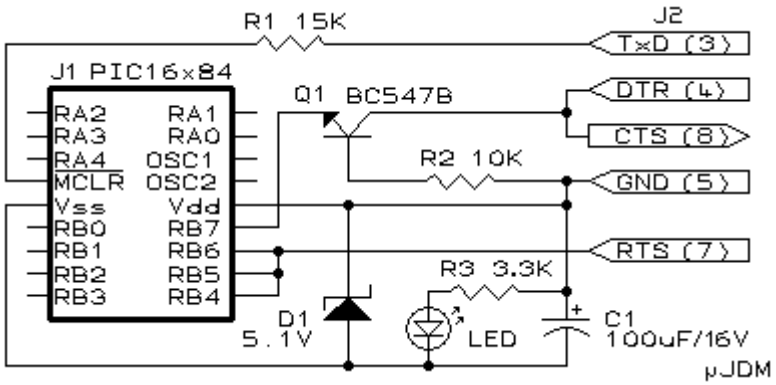


Figura 3.2 Circuito grabador uJDM.

3.1.2 Construcción del grabador

El programador uJDM es construido en una tarjeta perforada y unidas en rejillas de 22 agujeros X 13 agujeros**.** La figura 3.3 muestra la colocación de los componentes en la tarjeta. Hay espacios en blanco que significan que dividen a la línea para no hacer contacto.

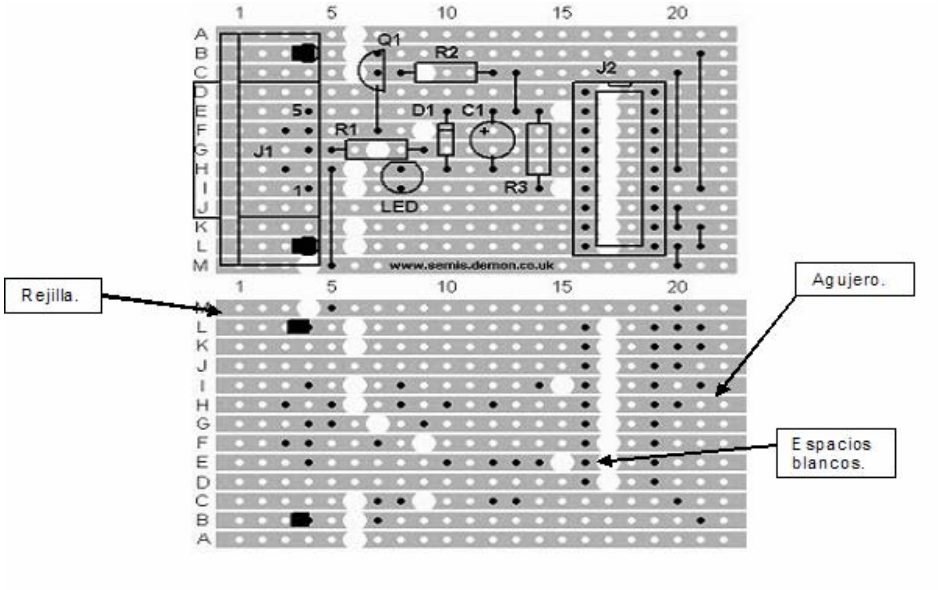
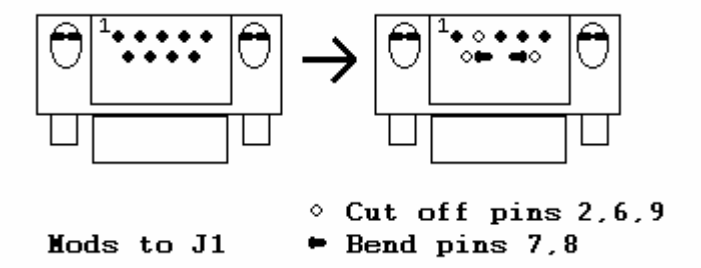


Figura 3.3 Colocación de componentes al circuito grabador uJDM.

El conector serial DBF9F se debe adaptar para colocarlo en la tarjeta perforada, en el cual se tienen que doblar los pines 7, 8 y cortando los pines 2,6,9, como se muestra en la siguiente figura.



*Fig*ura 3.4 Conector serial.

3.2 Probando el circuito grabador

Una vez construido el circuito grabador se usa el programa IC-Prog, el cual no es necesario instalar, sino que se utiliza directamente.

Al ser cargado por primera vez, pregunta la configuración del circuito grabador, el puerto y la interfase que se va a usar, por lo que se debe configurar como se muestra en la siguiente figura:

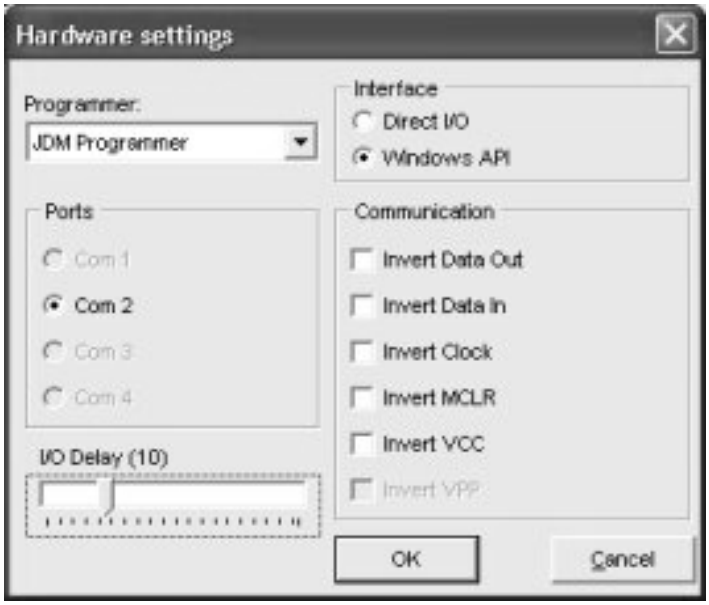


Figura 3.5 Configuración del circuito grabador.

Para saber el correcto funcionamiento del circuito grabador se conecta el PIC en el zócalo del grabador, conectando esta al puerto serial, habilitando el reloj como se muestra en la siguiente figura:

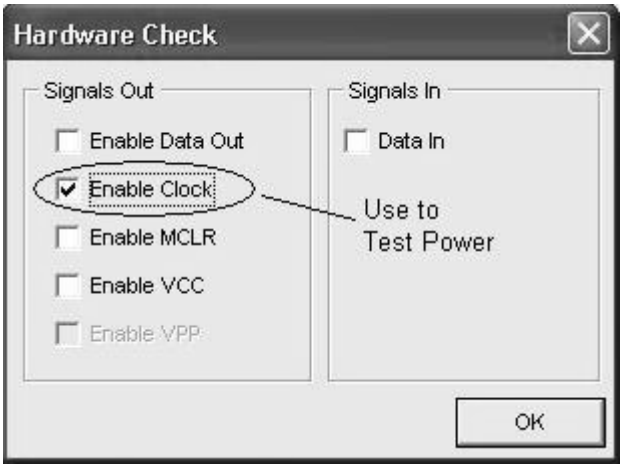


Figura 3.6 Habilitando al reloj.

Si el circuito grabador esta armado correctamente se encenderá el LED que tiene este. En caso de que esto no suceda, revisar la conexión del zócalo con el PIC.

Abrir la opción *File* y elegir *Open File* y busca el archivo con extensión .hex que creo el MPLAB al crear el proyecto, en el caso de nuestro ejemplo debemos elegir ejemplo.hex como se muestra en la siguiente figura:

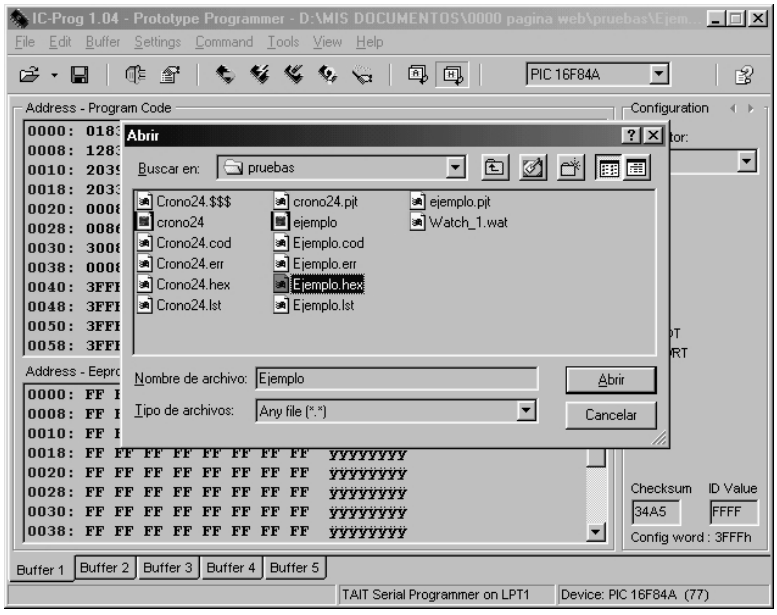


Figura 3.6 Abrir archivo .hex.

3.3 GRABANDO AL PIC

Cargado el programa con extensión .hex se procede a estos 6 pasos que se muestran en la figura 3.7:

1. Lee la información contenida en el PIC.
2. Carga el programa al PIC.
3. Limpia el PIC, es decir, borra cualquier programa que este contenga.
4. Verifica el buen funcionamiento del PIC.
5. Elegir el PIC con el cual estamos trabajando.

6.- Seleccionar el tipo de oscilador.

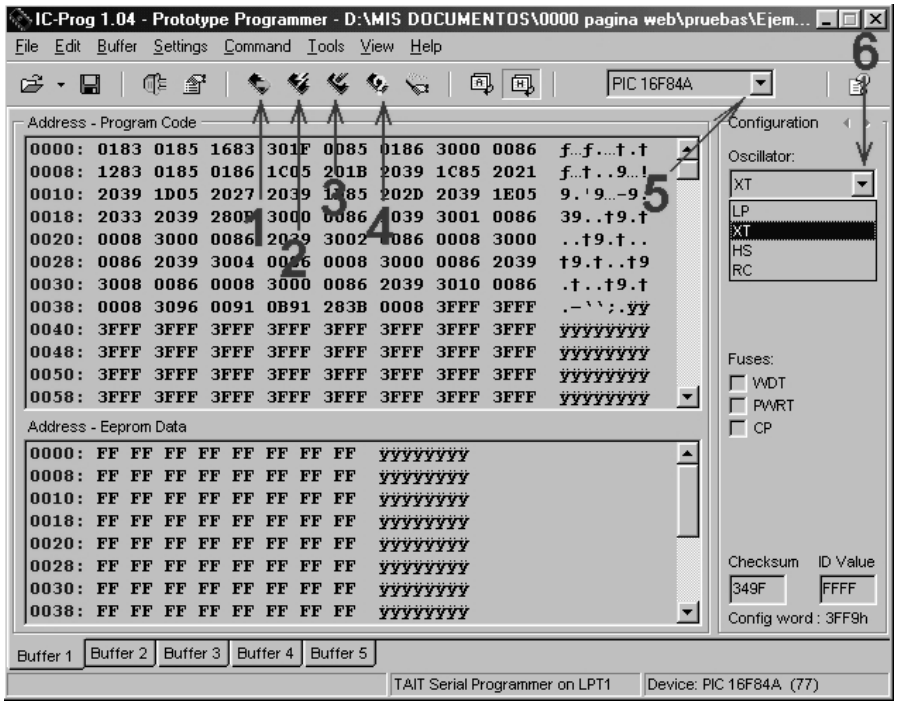


Figura 3.7 Pasos para grabar con IC-Prog.

Una vez que presionamos en la ventana Nº2, la información se carga en el PIC y el programa entrega un mensaje diciendo si la transmisión fue exitosa o no.

Se debe tener cuidado de configurar correctamente el programa, especificando el PIC con el que se va a trabajar, así como el tipo de oscilador.

Anexo 4

CÓDIGO FUENTE DE LA SECUENCIA INICIO/PARO

```
*********************************Pulsador_Armagedon12.asm*************************************************************
******************
;
;Cada vez que presione el pulsador conectado en el pin RA4 se prendera RB0 y cuando no se
presione el pulsador
; se prendera RB1.
;
;ZONA DE
DATOS*********************************************************************************************************************
***

      __CONFIG _CP_OFF & _WDT_OFF & _PWRTE_ON & _XT_OSC
          LIST P=16F84A
          INCLUDE <P16F84A.INC>

          CBLOCK 0x0C

          ENDC


reg1     equ      12
reg2     equ      13
reg3     equ      14
reg4   equ    15

#DEFINE Pulsador  PORTA,4
#DEFINE sensor1  PORTA,3

#DEFINE LED1  PORTB,0
#DEFINE LED2  PORTB,1
#DEFINE LED3  PORTB,2
#DEFINE LED4  PORTB,3
#DEFINE LED5  PORTB,4

;ZONA DE
CÓDIGOS*******************************************************************************************************************

               ORG  0
Inicio

               bsf   STATUS,RP0  ;Acceso al Banco 1.
               bcf   LED1
               bcf   LED2
               bcf   LED3
               bcf   LED4
          bsf   Pulsador

               bcf   STATUS,RP0;Acceso al Banco 0.
```

```
Principal


                btfsc   Pulsador                ; ¿Pulsador presionado?, ¿(Pulsador)=0?
                goto    Independencia            ; No. Vete a la subrutina Independencia (
Prender RB1)
                btfsc   Pulsador                ; Comprueba si es un rebote.
                goto    Fin
Encendido
            btfss   Pulsador                                        ;¿Dejo de pulsar?,
¿(Pulsador)=1?
            goto    Libertad            ; No. Vete a Independencia

Fin     goto    Principal



;Subrutina  " Independencia"----------------------------------------------------------------------------------------
;
Independencia



Cuadrito
            movlw  9f
                    movwf   reg1
uno    bsf    LED1

                    call    Retardo_2ms
                call    Retardo_2ms
                    call    Retardo_500micros
                    call    Retardo_20micros
                   call    Retardo_20micros

                   bcf    LED1

                call    Retardo_200micros
                call    Retardo_200micros
                    bsf    LED1

                call    Retardo_2ms
                    call    Retardo_2ms
                    call    Retardo_500micros
                    call    Retardo_20micros
                    call    Retardo_20micros
```

Anexo 4: CÓDIGO FUENTE DE LA SECUENCIA INICIO/PARO

```
bcf    LED1

                call   Retardo_200micros
                call   Retardo_200micros
    decfsz reg1,1
                   goto   uno
                   call  hola
                   call    Circulito
Circulito

          movlw  9f
                   movwf  reg2
dos    bsf   LED2
    call  Retardo_2ms
           call   Retardo_2ms
                  call   Retardo_500micros
          call  Retardo_20micros
          call  Retardo_5micros
                  bcf    LED2

                  call   Retardo_500micros
                     bsf    LED2

                  call   Retardo_2ms
                    call   Retardo_2ms
                    call   Retardo_500micros
                    call   Retardo_20micros
                    call   Retardo_5micros
                     bcf    LED2

                  call   Retardo_500micros
    decfsz reg2,1
                   goto   dos
                   call  hola
                   call    Asterisco

Asterisco

                   movlw  9f
                   movwf  reg3
tres   bsf   LED3

                   call   Retardo_2ms
                  call   Retardo_1ms
                     call   Retardo_50micros
                     call   Retardo_5micros
                  bcf    LED3

                  call   Retardo_200micros
                  call   Retardo_200micros
```

```
        bsf   LED3

                    call   Retardo_2ms
                       call   Retardo_1ms
                       call   Retardo_50micros
                       call   Retardo_5micros

                       bcf   LED3

                    call   Retardo_200micros
                    call   Retardo_200micros
        decfsz reg3,1
                       goto   tres
                       call   hola
                       call    Triangulito

    Triangulito
        movlw  9f
                       movwf  reg4
    cuatro bsf   LED4

                       call   Retardo_2ms
                    call   Retardo_2ms
                      call   Retardo_20micros
                      call   Retardo_20micros
                       call   Retardo_5micros

                    bcf   LED4

                    call   Retardo_200micros
                    call   Retardo_200micros
                       bsf   LED4

                    call   Retardo_2ms
                       call   Retardo_2ms
                       call   Retardo_20micros
                       call   Retardo_20micros
                      call   Retardo_5micros


                       bcf   LED4

                    call   Retardo_200micros
                    call   Retardo_200micros
        decfsz reg4,1
                       goto   cuatro
                       call   hola
                       goto   Inicio


;Subrutina hola*****************************************************************
```

Anexo 4: CÓDIGO FUENTE DE LA SECUENCIA INICIO/PARO

```
hola
                    bsf  LED5
                    call Retardo_5s
                    bcf  LED5

                    return



; Subrutina " Libertad"----------------------------------------------------------------------------
Libertad


                    btfsc   sensor1                 ; ¿Pulsador presionado?, ¿(Pulsador)=0?
                    bsf     LED1
                    btfsc   sensor1                 ;
                    goto    loco
jala
             btfss   Pulsador                                           ;¿Dejo de pulsar?,
¿(Pulsador)=1?
             bcf   LED1
             goto loco

loco      goto    Inicio



; Subrutina "Retardos" ----------------------------------------------

  CBLOCK
        R_ContA                         ; Contadores para los retardos.
        R_ContB
        R_ContC
        ENDC
;
; RETARDOS de 4 hasta 10 microsegundos ----------------------------------------
;
; A continuación retardos pequeños teniendo en cuenta que para una frecuencia de 10 MHZ,
; la llamada a subrutina "call" tarda 2 ciclos máquina, el retorno de subrutina
; "return" toma otros 2 ciclos máquina y cada instrucción "nop" tarda 1 ciclo máquina.
;
Retardo_10micros                        ; La llamada "call" aporta 2 ciclos máquina.
        nop                             ; Aporta 1 ciclo máquina.
        nop                             ; Aporta 1 ciclo máquina.
        nop                             ; Aporta 1 ciclo máquina.
        nop                             ; Aporta 1 ciclo máquina.
        nop                             ; Aporta 1 ciclo máquina.
Retardo_5micros                         ; La llamada "call" aporta 2 ciclos máquina.
        nop                             ; Aporta 1 ciclo máquina.
Retardo_4micros                         ; La llamada "call" aporta 2 ciclos máquina.
```

```
        return                          ; El salto del retorno aporta 2 ciclos máquina.
;
; RETARDOS de 20 hasta 500 microsegundos ----------------------------------------
;
Retardo_500micros                               ; La llamada "call" aporta 2 ciclos máquina.
        nop                             ; Aporta 1 ciclo máquina.
        movlw   d'414'                  ; Aporta 1 ciclo máquina. Este es el valor de "K".
        goto    RetardoMicros           ; Aporta 2 ciclos máquina.
Retardo_200micros                               ; La llamada "call" aporta 2 ciclos máquina.
        nop                             ; Aporta 1 ciclo máquina.
        movlw   d'164'                  ; Aporta 1 ciclo máquina. Este es el valor de "K".
        goto    RetardoMicros           ; Aporta 2 ciclos máquina.
Retardo_100micros                               ; La llamada "call" aporta 2 ciclos máquina.
        movlw   d'81'                   ; Aporta 1 ciclo máquina. Este es el valor de "K".
        goto    RetardoMicros           ; Aporta 2 ciclos máquina.
Retardo_50micros                        ; La llamada "call" aporta 2 ciclos máquina.
        nop                             ; Aporta 1 ciclo máquina.
        movlw   d'39'                   ; Aporta 1 ciclo máquina. Este es el valor de "K".
        goto    RetardoMicros           ; Aporta 2 ciclos máquina.
Retardo_20micros                        ; La llamada "call" aporta 2 ciclos máquina.
        movlw   d'15'                   ; Aporta 1 ciclo máquina. Este es el valor de "K".
;
; El próximo bloque "RetardoMicros" tarda:
; 1 + (K-1) + 2 + (K-1)x2 + 2 = (2 + 3K) ciclos máquina.
;
RetardoMicros
        movwf   R_ContA                 ; Aporta 1 ciclo máquina.
Rmicros_Bucle
        decfsz  R_ContA,F               ; (K-1)x1 cm (cuando no salta) + 2 cm (al saltar).
        goto    Rmicros_Bucle           ; Aporta (K-1)x2 ciclos máquina.
        return                          ; El salto del retorno aporta 2 ciclos máquina.
;
;En total estas subrutinas tardan:
; - Retardo_500micros:      2 + 1 + 1 + 2 + (2 + 3K) = 1250 cm = 500 µs. (para K=414 y 10 MHz).
; - Retardo_200micros:      2 + 1 + 1 + 2 + (2 + 3K) = 500 cm = 200 µs. (para K=164 y 10 MHz).
; - Retardo_100micros:      2   + 1 + 2 + (2 + 3K) = 250 cm = 100 µs. (para K= 81 y 10 MHz).
; - Retardo_50micros :      2 + 1 + 1 + 2 + (2 + 3K) = 125 cm =  50 µs. (para K= 39 y 10 MHz).
; - Retardo_20micros :      2   + 1   + (2 + 3K) =  50 cm =  20 µs. (para K= 15 y 10 MHz).
;
; RETARDOS de 1 ms hasta 200 ms. ------------------------------------------------
;
Retardo_200ms                           ; La llamada "call" aporta 2 ciclos máquina.
        movlw   d'200'                  ; Aporta 1 ciclo máquina. Este es el valor de "M".
        goto    Retardos_ms             ; Aporta 2 ciclos máquina.
Retardo_100ms                           ; La llamada "call" aporta 2 ciclos máquina.
        movlw   d'100'                  ; Aporta 1 ciclo máquina. Este es el valor de "M".
        goto    Retardos_ms             ; Aporta 2 ciclos máquina.
Retardo_50ms                            ; La llamada "call" aporta 2 ciclos máquina.
        movlw   d'50'                   ; Aporta 1 ciclo máquina. Este es el valor de "M".
        goto    Retardos_ms             ; Aporta 2 ciclos máquina.
Retardo_20ms                            ; La llamada "call" aporta 2 ciclos máquina.
```

```
movlw   d'20'                   ; Aporta 1 ciclo máquina. Este es el valor de "M".
        goto    Retardos_ms             ; Aporta 2 ciclos máquina.
Retardo_10ms                            ; La llamada "call" aporta 2 ciclos máquina.
        movlw   d'10'                   ; Aporta 1 ciclo máquina. Este es el valor de "M".
        goto    Retardos_ms             ; Aporta 2 ciclos máquina.
Retardo_5ms                             ; La llamada "call" aporta 2 ciclos máquina.
        movlw   d'5'                    ; Aporta 1 ciclo máquina. Este es el valor de "M".
        goto    Retardos_ms             ; Aporta 2 ciclos máquina.
Retardo_2ms                             ; La llamada "call" aporta 2 ciclos máquina.
        movlw   d'2'                    ; Aporta 1 ciclo máquina. Este es el valor de "M".
        goto    Retardos_ms             ; Aporta 2 ciclos máquina.
Retardo_1ms                             ; La llamada "call" aporta 2 ciclos máquina.
        movlw   d'1'                    ; Aporta 1 ciclo máquina. Este es el valor de "M".
;
; El próximo bloque "Retardos_ms" tarda:
; 1 + M + M + KxM + (K-1)xM + Mx2 + (K-1)Mx2 + (M-1) + 2 + (M-1)x2 + 2 =
; = (2 + 4M + 4KM) ciclos máquina. Para K=624 y M=1 supone 1002 ciclos máquina
; que a 10 MHz son 1002 µs = 1 ms.
;
Retardos_ms
        movwf   R_ContB                 ; Aporta 1 ciclo máquina.
R1ms_BucleExterno
        movlw   d'249'                  ; Aporta Mx1 ciclos máquina. Este es el valor de "K".
        movwf   R_ContA                 ; Aporta Mx1 ciclos máquina.
R1ms_BucleInterno
        nop                             ; Aporta KxMx1 ciclos máquina.
        decfsz  R_ContA,F               ; (K-1)xMx1 cm (cuando no salta) + Mx2 cm (al saltar).
        goto    R1ms_BucleInterno               ; Aporta (K-1)xMx2 ciclos máquina.
        decfsz  R_ContB,F               ; (M-1)x1 cm (cuando no salta) + 2 cm (al saltar).
        goto    R1ms_BucleExterno       ; Aporta (M-1)x2 ciclos máquina.
        return                          ; El salto del retorno aporta 2 ciclos máquina.
;
;En total estas subrutinas tardan:
; - Retardo_200ms:          2 + 1 + 2 + (2 + 4M + 4KM) = 500007 cm = 200 ms. (M=200 y K=624).
; - Retardo_100ms:          2 + 1 + 2 + (2 + 4M + 4KM) = 250007 cm = 100 ms. (M=100 y K=624).
; - Retardo_50ms :2 + 1 + 2 + (2 + 4M + 4KM) = 125007 cm =  50 ms. (M= 50 y K=624).
; - Retardo_20ms :2 + 1 + 2 + (2 + 4M + 4KM) =  50007 cm =  20 ms. (M= 20 y K=624).
; - Retardo_10ms :2 + 1 + 2 + (2 + 4M + 4KM) =  25007 cm =  10 ms. (M= 10 y K=624).
; - Retardo_5ms  : 2 + 1 + 2 + (2 + 4M + 4KM) =  12507 cm =   5 ms. (M=  5 y K=624).
; - Retardo_2ms  : 2 + 1 + 2 + (2 + 4M + 4KM) =   5007 cm =   2 ms. (M=  2 y K=624).
; - Retardo_1ms  : 2 + 1    + (2 + 4M + 4KM) =   2505 cm =   1 ms. (M=  1 y K=624).
;
; RETARDOS de 0.5 hasta 20 segundos ----------------------------------------------
;
Retardo_20s                             ; La llamada "call" aporta 2 ciclos máquina.
        movlw   d'200'                  ; Aporta 1 ciclo máquina. Este es el valor de "N".
        goto    Retardo_1Decima         ; Aporta 2 ciclos máquina.
Retardo_10s                             ; La llamada "call" aporta 2 ciclos máquina.
        movlw   d'100'                  ; Aporta 1 ciclo máquina. Este es el valor de "N".
        goto    Retardo_1Decima         ; Aporta 2 ciclos máquina.
Retardo_5s                              ; La llamada "call" aporta 2 ciclos máquina.
```

```
movlw   d'50'                   ; Aporta 1 ciclo máquina. Este es el valor de "N".
        goto    Retardo_1Decima ; Aporta 2 ciclos máquina.
Retardo_2s                      ; La llamada "call" aporta 2 ciclos máquina.
        movlw   d'20'           ; Aporta 1 ciclo máquina. Este es el valor de "N".
        goto    Retardo_1Decima ; Aporta 2 ciclos máquina.
Retardo_1s                      ; La llamada "call" aporta 2 ciclos máquina.
        movlw   d'10'           ; Aporta 1 ciclo máquina. Este es el valor de "N".
        goto    Retardo_1Decima ; Aporta 2 ciclos máquina.
Retardo_500ms                   ; La llamada "call" aporta 2 ciclos máquina.
        movlw   d'5'            ; Aporta 1 ciclo máquina. Este es el valor de "N".
;
; El próximo bloque "Retardo_1Decima" tarda:
; 1 + N + N + MxN + MxN + KxMxN + (K-1)xMxN + MxNx2 + (K-1)xMxNx2 +
;  + (M-1)xN + Nx2 + (M-1)xNx2 + (N-1) + 2 + (N-1)x2 + 2 =
; = (2 + 4M + 4MN + 4KM) ciclos máquina. Para K=623, M=100 y N=1 supone 250000
; ciclos máquina que a 10 MHz son 100000 µs = 100 ms = 0,1 s = 1 décima de segundo.
;
Retardo_1Decima
        movwf   R_ContC         ; Aporta 1 ciclo máquina.
R1Decima_BucleExterno2
        movlw   d'100'          ; Aporta Nx1 ciclos máquina. Este es el valor de "M".
        movwf   R_ContB         ; Aporta Nx1 ciclos máquina.
R1Decima_BucleExterno
        movlw   d'623'          ; Aporta MxNx1 ciclos máquina. Este es el valor de "K".
        movwf   R_ContA         ; Aporta MxNx1 ciclos máquina.
R1Decima_BucleInterno
        nop                     ; Aporta KxMxNx1 ciclos máquina.
        decfsz  R_ContA,F       ; (K-1)xMxNx1 cm (si no salta) + MxNx2 cm (al saltar).
        goto    R1Decima_BucleInterno   ; Aporta (K-1)xMxNx2 ciclos máquina.
        decfsz  R_ContB,F       ; (M-1)xNx1 cm (cuando no salta) + Nx2 cm (al saltar).
        goto    R1Decima_BucleExterno   ; Aporta (M-1)xNx2 ciclos máquina.
        decfsz  R_ContC,F       ; (N-1)x1 cm (cuando no salta) + 2 cm (al saltar).
        goto    R1Decima_BucleExterno2  ; Aporta (N-1)x2 ciclos máquina.
        return                  ; El salto del retorno aporta 2 ciclos máquina.
;
;En total estas subrutinas tardan:
; - Retardo_20s:    2 + 1 + 2 + (2 + 4N + 4MN + 4KMN) = 49920807 cm = 20 s.
;                           (N=200, M=100 y K=623).
; - Retardo_10s:    2 + 1 + 2 + (2 + 4N + 4MN + 4KMN) = 24960407 cm = 10 s.
;                           (N=100, M=100 y K=623).
; - Retardo_5s:             2 + 1 + 2 + (2 + 4N + 4MN + 4KMN) =  12480207 cm =  5 s.
;                           (N= 50, M=100 y K=623).
; - Retardo_2s:             2 + 1 + 2 + (2 + 4N + 4MN + 4KMN) =  4992087 cm =  2 s.
;                           (N= 20, M=100 y K=623).
; - Retardo_1s:             2 + 1 + 2 + (2 + 4N + 4MN + 4KMN) =  2496047 cm =  1 s.
;                           (N= 10, M=100 y K=623).
; - Retardo_500ms:          2 + 1    + (2 + 4N + 4MN + 4KMN) =  1248025 cm = 0,5 s.
;                           (N=  5, M=100 y K=623).
        END
```

BIBLIOGRAFÍA GENERAL

< http://www.personal-robotics.de/personal-robotics/fischertechnik_computing.htm>

<http://ageweb.age.uiuc.edu/faculty/teg/Research/BiosystemsAutomation/AgRobots/robotics_illinois_alumni.jpg>

http://cedicyt.usach.cl/microcomputadores/proyectos/sensor/circuito.htm

http://cholopic.pe.nu/

http://es.wikipedia.org/wiki/Control_remoto

http://es.wikipedia.org/wiki/Extremidad

http://es.wikipedia.org/wiki/Motor_de_corriente_continua

http://es.wikipedia.org/wiki/Motor_paso_a_paso

http://es.wikipedia.org/wiki/Receptor_de_radio

http://es.wikipedia.org/wiki/Regulador

http://es.wikipedia.org/wiki/Servomotor

http://es.wikipedia.org/wiki/Transmisor

<http://html.rincondelvago.com/engranajes_3.html>

http://platea.cnice.mecd.es

<http://trueforce.com/Articles/Robot_History.htm>

<http://www.aibo.com/>

<http://www.atrox.at/robots/>

<http://www.bath.ac.uk/~en1tji/history.htm>

<http://www.ciencia.net/VerArticulo/Robot?idArticulo=dsfjusjfr7sogujnry9x21m>

<http://www.creaturoides.com/anterior/puentesh.htm>

<http://www.csic.es/prensa/Noticias%202004/robots.html>

<http://www.elephantcare.org/>

<http://www.elytradesign.com/ari/html/clepsydra.html>

<http://www.faculty.ucr.edu/~currie/roboadam.htm>

<http://www.fbelec.com/fbelec/ProductDetail.asp?ProductNO=1202&catid=51&subid=106>

<http://www.iai.csic.es/dca/clawar_txt.htm>

<http://www.iearobotics.com/proyectos/libro6811/libro-6811.pdf>

<http://www.iirobotics.com/webpages/robothistory.php>

<http://www.ilustrados.com/publicaciones/EpyuZlVyZlsWULVYYw.php>

<http://www.laflecha.net/canales/ciencia/200409212/>

<http://www.laflecha.net/canales/ciencia/noticias/200410041/>

<http://www.lt-automation.com/nitinol.html>

<http://www.mabuchi-motor.co.jp/cgi-bin/catalog/e_catalog.cgi?CAT_ID=fa_280rasa >

<http://www.me.umn.edu/education/courses/me2011/robot/technotes/L293/L293.html>

<http://www.newscientist.com/news/news.jsp?id=ns99994409>

<http://www.otherlandtoys.co.uk/product529/product_info.html?name=Meccano%20Best%20of%2040%20Set&&prod=529>

<http://www.physics.brandeis.edu/phys32b_2005/pic16f84A.pdf>

<http://www.revista-nanociencia.ece.buap.mx/articulo1.pdf>

<http://www.roboticajoven.mendoza.edu.ar/rob_dis4.htm>

<www.todorobot.com.ar>

http://www.uwe.ac.uk/clawar/home.htm

PALACIOS, ENRIQUE; RAMIRO, FERNANDO; LOPEZ, J. LUCAS. *Microcontrolador PIC16F84. Desarrollo de proyectos.* Colombia: Alfaomega RA-MA, 2005.

RIGO ARNAVAT, A; GENESCA DUEÑAS, G. *Como presentar una tesis y -trabajos de investigación. Barcelona: Eumo Octaedro, 2002.*

ZAGAL, J.C.; RUIZ DEL SOLAR, J. 2004. “UCHILSIM: A Dynamically and Visually _Realistic Simulator for the Robocup Four Legged League”. *Robocup 2004 International Symposium.*