INSTITUTO POLITÉCNICO NACIONAL

CENTRO DE INVESTIGACION EN COMPUTACIÓN

“DISEÑO DE UN SISTEMA DE DESARROLLO PARA LA ENSEÑANZA DE LA ROBÓTICA BÁSICA”

TESIS

PARA OBTENER EL GRADO DE: MAESTRÍA EN CIENCIAS DE LA COMPUTACIÓN.

PRESENTA:

ING. ENRIQUE BUCHÁN CASTILLO

DIRECTOR DE TESIS:

DR. ALFONSO GUTIÉRREZ ALDANA

MÉXICO D.F.

JULIO DE 2011

Agradecimientos.

Es aquí donde agradezco a Dios primeramente por darme la oportunidad del conocimiento, a mis padres por apoyarme y alentarme para alcanzar mis objetivos, agradezco la educación que me enseñaron en el inicio de la vida y que ha venido a darme buenos resultados en esta etapa de mi carrera.

Agradezco al Instituto Politécnico Nacional, al CIC, a sus profesores y a mi director de tesis, por los conocimientos transmitidos y la oportunidad de cursar la maestría en este lugar, así como también al CONACYT sin cuyo apoyo económico no hubiera alcanzado la meta de terminar este posgrado.

Gracias.

Resumen.

El estudio e investigación en el campo educativo y el desarrollo de la robótica ha generado una nueva disciplina conocida como robótica pedagógica, ésta tiene el propósito de que el estudiante interactúe con un robot educativo, para que el proceso de aprendizaje se vea reforzado en áreas del conocimiento como matemáticas, tecnología, ciencias de la información y la comunicación.

El presente proyecto de tesis tiene como objetivo, diseñar y construir un sistema de desarrollo o robot educativo, con el cual los estudiantes o personas que tengan interés en el tema de la robótica básica puedan iniciarse. Este sistema de desarrollo tiene la característica de que está construido con materiales que se pueden adquirir de forma fácil en las tiendas de electrónica locales de México. Con esta característica se obtienen muchas ventajas. Una de ellas es que el estudiante no requiere construir el kit completo sino sólo una parte de él y con esto poder iniciar su uso, tarea que no se puede realizar al adquirir un robot educativo comercial ya que la mayoría son de importación.

El diseño y construcción que se muestra en la presente tesis, tiene como base el trabajo e investigación del producto Mindstorm NXT de la empresa LEGO.

El resultado de este proyecto es un robot educativo muy completo tanto en el hardware como en el software de programación. Además de ser un proyecto escalable a sistemas más complejos utilizando la comunicación inalámbrica con dispositivos Bluetooth o Wi-Fi.

Abstract.

The study and investigation in the educational field and the development of the robotics has generated a new discipline known as pedagogic robotics, this one has the intention of which the student interacts with an educational robot, in order that the learning process meets reinforced in areas of the knowledge as mathematics, technology, sciences of the information and the communication.

The present project of thesis has as aim, design and constructs a system of development or educational robot, with which the students or persons who have interest in the topic of the basic robotics could begin. This system of development has the characteristic of which it is constructed by materials that can be acquired of easy form in the local shops of electronics of Mexico. With this characteristic many advantages are obtained. One of them is that the student does not need to construct the complete kit but only a part of it and with this to be able to initiate his use, task that it is not possible to realize on having acquired an educational commercial robot since the majority they are of import.

The design and construction that appears in the present thesis, takes as a base the work and investigation of the product Mindstorm NXT of the company LEGO.

The result of this project is an educational very complete robot both in the hardware and in the software of programming. In addition being a scalable project to more complex systems using the wireless communication with devices Bluetooth or Wi-Fi.

Tabla de contenido

CAPÍTULO I

INTRODUCCIÓN

1.1 Robots educativos.

Un robot educativo es una herramienta tecnológica que se utiliza para la enseñanza de la robótica básica y de otras disciplinas. En la figura 1.1 se muestra un ejemplo.

Figura 1.1 Robot educativo.

El robot educativo está formado principalmente por tres módulos.

Uno es el módulo de entrada que lo conforman los sensores, en el ejemplo A) de la figura 1.1 se puede observar el sensor de luz, el sensor de sonido, el sensor de toque y el de distancia.

El segundo módulo es el de proceso, ejemplo B) de la figura 1.1, lo compone el bloque de control en el cual se van a programar las tareas del robot.

Y el tercer módulo es el de salida que está formado por los actuadores como son los motores y el display, ejemplo C) de la figura 1.1.

Esta herramienta educativa debe tener la característica de poder armar diferentes estructuras con las partes que la conforman. En la figura 1.2 tenemos algunos ejemplos de diferentes estructuras y del software para programar al robot.

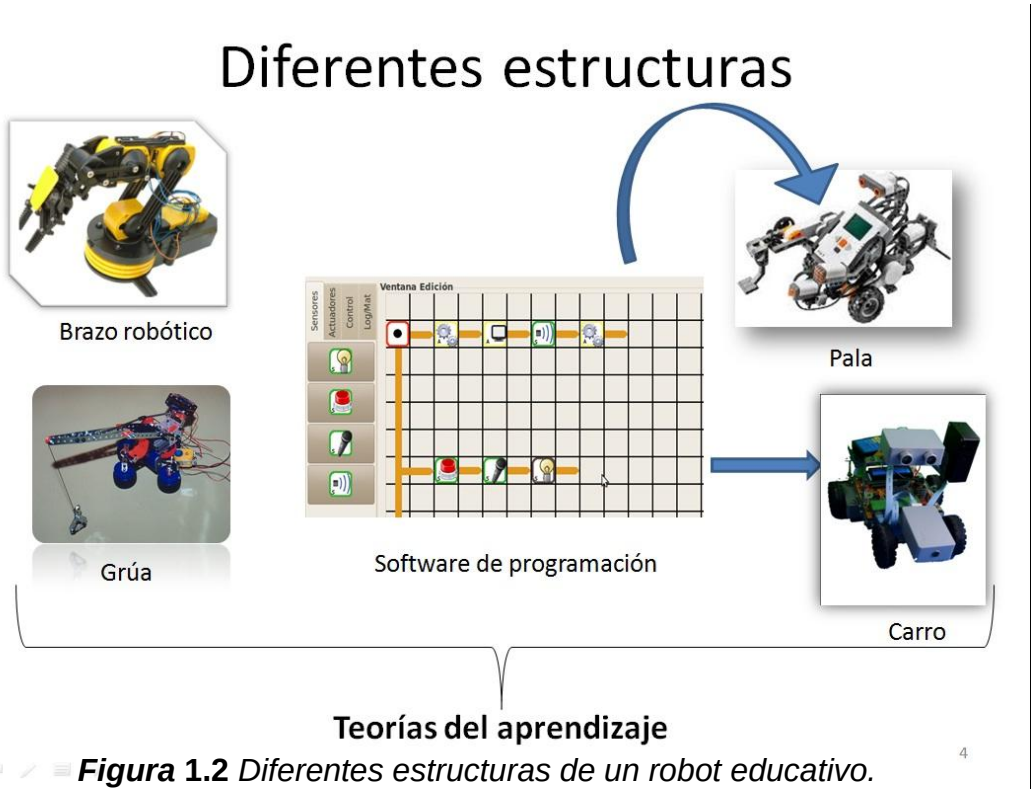

***Figura* 1.2** *Diferentes estructuras de un robot educativo.*

Otra característica de este tipo de robot es que se le puede programar acciones que el estudiante o el usuario requiere que la estructura realice.

Esta programación se efectúa por medio de un software que se instala en una PC, y ese programa es descargado al bloque de control mediante alguna comunicación que debe existir entre ellos.

También es parte del robot educativo un conjunto de partes mecánicas para armar las estructuras que el estudiante o el usuario necesite.

Detrás del robot educativo están las teorías del aprendizaje que respaldan su uso como herramienta educativa.

1.2 Planteamiento del problema.

Los robots educativos pueden ser utilizados, ya sea como objeto mismo del estudio de la robótica, o como medio para la enseñanza de distintas áreas del conocimiento (Enrique Ruiz Velasco, 2007, p.104), es por esto que su utilización ha ido en incremento.

Aunque su desarrollo en México ha sido promovido por algunos grupos, éste ha sido muy lento, y un factor importante de este desarrollo lento es que los robots educativos que se utilizan son importados.

Como consecuencia, al tratar de obtener algunas partes extras se tiene el problema de que no se encuentran en el país, además de que tenemos el factor económico que siempre es importante en los países en desarrollo como México y en áreas como la educación pública.

El desarrollo de un diseño de un robot educativo, en el que todo el material electrónico y mecánico con el que se construya sea adquirido con los proveedores locales ayudaría a incrementar el uso y desarrollo de las herramientas lúdico-tecnológicas, por las escuelas tanto privadas como del sector público.

1.3 Justificación.

Actualmente existen varias empresas que comercializan a los robots educativos como son:

- Múltiplo
- Lego RCX

- Lego Mindstorm NXT
- Cebek
- Bioloid
- Mecano
- Kondo
- Ma-Vin
- FischerTechnik

De las existentes, la que ha tenido una mayor aceptación en los usuarios es LEGO con su producto RCX primeramente y actualmente con el MINDSTORM NXT 2.0.

Esta aceptación en los usuarios se debe a que LEGO trabajó en conjunto con el laboratorio de medios del MIT, a mediados de los años ochentas. Este laboratorio se encarga de realizar investigaciones sobre las teorías del aprendizaje, y como resultado de esas investigaciones y la utilización de los productos LEGO, se obtuvo un robot educativo como el de Mindstorm.

El presente proyecto de tesis se diseñó con base en el LEGO Mindstorm. Aunque se tiene acceso a este robot, si diseñamos un robot educativo aquí en México, obtenemos muchas ventajas:

- Se pueden construir las partes de los módulos que el usuario necesite, a diferencia de la de marca LEGO, que aquí en México se tendría que comprar otro kit completo o importar una sola parte si es requerida.
- Las partes con las que se construyen los diferentes componentes del robot educativo, se encuentran con facilidad en las tiendas de electrónica locales.
- No se requiere comprar todo el material para iniciar la construcción de este robot; el estudiante o usuario puede únicamente empezar con el módulo de control, algún elemento de entrada y uno de salida para poder realizar una estructura y programarla de forma interactiva.

- Con la construcción de cada elemento de robot el estudiante adquiere conocimientos de diferentes disciplinas.
- Una vez que el estudiante construyó algunas partes del robot puede diseñar estructuras a controlar.
- El software y código fuente del microcontrolador serán libres.
- Es más económico.

1.4 Objetivos.

1.4.1 Objetivo general.

La presente tesis propone diseñar e implementar un sistema integrado por partes electromecánicas controladas por un microcontrolador y que tenga elementos básicos de la robótica como la unión de piezas, la programación de acciones de forma interactiva y la obtención de información del medio ambiente por medio de sensores (robot educativo), para ser utilizado como herramienta en la enseñanza de la robótica básica.

Nota: Este proyecto se enfocará únicamente en la parte física del robot educativo, la parte del software de control se desarrolló en el proyecto de tesis que lleva por nombre “Diseño y desarrollo de un Compilador Visual para la enseñanza de la robótica básica” (Tonche Ronny, 2010).

1.4.2 Objetivos específicos.

1. Diseñar e implementar el módulo de entrada o de percepción.
    a. Sensor de sonido.
    b. Sensor de luz.
    c. Sensor de distancia.

   d. Sensor de tacto.
2. Diseñar e implementar el módulo de proceso.
   a. Comunicación con la PC.
   b. Estructura del despachador de tareas.
   c. Manejo de los periféricos.
3. Diseñar e implementar el módulo de salida.
   a. Motores con encoders.
   b. Manejo del display.
4. Implementar los módulos en forma conjunto para su funcionamiento.

Resumen.

En este capítulo se dio la introducción a los robots educativos, su definición, las partes que lo integran y sus principales características. También se presentó el problema que se tiene con los robots educativos que se venden comercialmente, así como las ventajas que se obtendrían al construir uno de ellos en nuestro país, de esta forma, se plantearon los objetivos del presente proyecto de tesis.

En el siguiente capítulo se hará un análisis a algunos de los robots educativos comerciales y a otros propuestos por diferentes universidades.

CAPÍTULO 2

ESTADO DEL ARTE DE LOS ROBOTS EDUCATIVOS

2.1 Introducción

Se puede definir a un robot pedagógico, como un dispositivo tecnológico, construido con la finalidad de que cumpla con los principios de la robótica pedagógica, y que tiene propiedades educativas que ayudan a complementar los entornos de enseñanza aprendizaje, con lo cual se mejoran los procesos cognoscitivos de los estudiantes de distintas edades y niveles educativos, cuando inician el estudio de las ciencias y la tecnología. Como se puede concluir de su definición, no es trivial la construcción de un robot educativo. (Ruiz Velasco, 2007, p.275).

La robótica pedagógica tiene como uno de sus objetivos que los estudiantes construyan su conocimiento a partir de un ambiente de aprendizaje, un elemento importante de ese ambiente lo tiene el robot educativo, ya que desde el punto de vista cognitivo permite tener mejores condiciones de apropiación del conocimiento. Desde que la empresa de juguetes LEGO se unió al grupo de trabajo de epistemología y aprendizaje del laboratorio de medios del MIT, se han desarrollados estos robots educativos como resultado de las investigaciones realizadas por ellos. Y siendo una disciplina la robótica pedagógica prácticamente nueva, otras empresas y universidades han tomado también interés en el desarrollo de productos semejantes para ofrecerlos al campo de la educación.

2.2 Los robots educativos en la actualidad.

En el área comercial algunos de los robots educativos que existen son:

2.2.1 Lego Mindstorms NXT 2.0.- Este kit educativo que los investigadores prefieren utilizar cuando realizan estudios del área de aprendizaje, es de la empresa LEGO, una empresa que aunque está en Dinamarca tiene un acuerdo

con el laboratorio de medios del Instituto Tecnológico de Massachusetts, para el desarrollo de investigaciones en el campo del conocimiento y aprendizaje.

El grupo de este laboratorio ha realizado proyectos educativos que son sustentados por el aporte de las teorías del aprendizaje y fue fundado por el investigador que creó la teoría del construccionismo, Symour Papert, quien trabajó con otro gran personaje como lo es el psicólogo Jean Piaget, a quien se le considera creador de las bases de la educación actual.

El robot educativo Mindstorms NXT 2.0 de LEGO (s.f.) http://mindstorms.lego.com es un dispositivo que se compone de un sistema de percepción el cual está conformado por una serie de sensores como el sensor de luz, el sensor de sonido, el sensor de distancia, y sensor de color; por otro lado el sistema de actuadores está formado principalmente por motores y en el módulo de control programable se encuentra un display y una bocina.

Este módulo programable es el sistema de proceso, que en su última versión trabaja con dos microcontroladores de la marca Atmel, el ATmega48/v y el AT91SAM7S256.

Este módulo utiliza una interfaz USB para comunicarse con la computadora y poder guardar los programas que hayan sido creados por los estudiantes; esta comunicación también puede realizarse por medio del dispositivo bluetooth que ya tiene incorporado, y con el cual puede comunicarse hasta con 3 módulos de control más y poder interactuar para realizar alguna actividad en conjunto.

2.2.2 Múltiplo.- Es un kit de robótica diseñado por RobotGroup una empresa argentina dedicada a la robótica educativa, su producto Múltiplo está conformado por partes mecánicas y sensores; al bloque de control programable le denominan plataforma y proponen dos modelos el Controlador Brain M644 y el DuinoBot v1.2 implementados con microprocesadores Atmel AVR ATMega644 y ATMega32U4 respectivamente.

El módulo de los motores viene con un reductor de engranes y el control de los mismos se puede hacer con un encoder externo sencillo.

El robot educativo Múltiplo está diseñado para el aprendizaje de la robótica básica, y la limitación es con respecto a los sensores ya que los trabaja sin una caja que pueda cuidar y soportar un uso rudo.

Como son desarrollados en Argentina implica un costo el adquirir este producto y sus complementos para un proyecto más complejo.

2.2.3 Bioloid.- Este es un kit de una plataforma robótica modular que se utiliza para construir robots avanzados.

La publicidad de esta marca menciona que “es como una versión para adultos del LEGO Mindstorm NXT y Meccano y está hecho de muchos bloques constructivos”.

En este kit encontramos que los módulos de los motores son servo motores que utilizan un microcontrolador cada uno de ellos y su comunicación con el bloque de control programable o módulo controlador, como le llaman en este kit, es una comunicación serie mediante un bus de dos líneas una de ellas bidireccional la otra es la línea común y un tercer hilo para la energía que se proporciona a los motores.

El sistema de percepción está constituido por un solo módulo que contiene tres sensores que utiliza un sistema de infrarrojos que se pueden emplear para medir distancia, luminosidad y sonido, además de una pequeña bocina. El módulo controlador CM-5, es el núcleo de la plataforma Bioloid con base en el microcontrolador Atmel ATMega128.El cual cuenta con una batería recargable.

La placa controladora viene encajada en una cápsula con leds y botones que permiten activar manualmente operaciones y monitorizar con los leds los distintos estados del robot.

El CM-5 de Bioloid viene con el firmware preinstalado que permite la interoperación con el software para PC proporcionado en el kit, y comunicarse con

la PC mediante un cable serie de tipo interfaz RS 232 proporcionado para cargar en el CM-5 los programas que controlan el robot. Para la comunicación entre los motores y sensores con el módulo de control se utiliza una comunicación serie asíncrona de 8 bits con un bit de paro y sin paridad.

Como se puede observar de las anteriores características, es un robot que no es precisamente ideal para la enseñanza de la robótica básica ya que sus sistemas son más complejos y por lo tanto es uno de los robots educativos más caros.

Además los kits Bioloid son fabricados en Estado Unidos, lo que implica una importación de elementos que se quieran agregar a los proyectos.

Otra observación es que el tipo de comunicación sigue siendo mediante la interfaz RS232, y no se ha actualizado a la comunicación por USB o bluetooth.Sin embargo existe una posibilidad de comunicación inalámbrica de tipo ZigBee con un módulo que vende por separado el fabricante.

2.2.4 Kondo.- Es un robot educativo japonés muy similar a Bioloid pero con mejoras en el aspecto de la comunicación con la computadora ya que utiliza la interfaz USB y un software más fácil de utilizar.

Sus limitaciones son parecidas a las del robot Bioloid pero con un precio más alto en el mercado y dado que es fabricado en Japón este detalle se hace más presente.

2.2.5 Ma-Vin.- "Es un kit de robótica modular, versátil y educativo, con una interfaz de programación visual basada en iconos para los principiantes, y programación en "C" para los avanzados." Este robot educativo utiliza un microcontrolador ATMEGA64L y tiene comunicación con la computadora mediante una interfaz USB 1.1.

Al analizar este robot observamos que su armado se limita a una sola forma, con lo que no cumple con las bases de las teorías de aprendizaje que se requieren para considerarlo como un constructor del conocimiento.

2.2.6 FischerTechnik.- Fischertechnik, es una marca alemana parte de la compañía fischer-werke (www.fischer.de). Es un sistema de construcción modular, el cual utiliza piezas principalmente plásticas fabricadas con los mejores materiales y los más altos estándares de calidad. Aunque el producto es fabricado desde 1965, éste llegó de manera oficial a México en 2003 por medio de la empresa hitech INGENIUM (www.hitechingenium.com), empresa que distribuye de manera exclusiva a este versátil sistema de construcción.

Debido a que es una empresa que no solamente se dedica a los juguetes sino también a la industria, este producto que se vincula a la robótica educativa fue pensado para diseñar y desarrollar propuestas a soluciones en la robótica de la industria.

Esto lo podemos ver en sus productos los cuales están fabricados con piezas que tienen más sentido en el contexto de las fábricas, ya sea en sus líneas de producción o en sus sistemas de transporte.

Dentro de sus productos ofrece un kit para iniciarse en la robótica pero siempre con miras a ensayar los proyectos en el ámbito industrial; este kit maneja como todos los robot educativos los módulos de sensores, módulos de actuadores y su módulo de control programable mediante una interfaz USB o bluetooth para cargar el programa que el estudiante ha realizado en la computadora mediante el software de FisherTechnik.

Es un producto fabricado con materiales de alta calidad y con piezas más sofisticadas que las de otros robots educativos; todas estas características hacen que FisherTechnik sea un producto de un alto precio, por lo que en México no es una opción muy viable.

2.2.7 Robo-ed.- En México una empresa que ha desarrollado kits educativos en robótica es Robo-ed, empresa situada en el estado de Chihuahua, y que tiene como premisa el desarrollar e integrar conceptos de Robótica orientados a la educación, promoviendo éstos a las escuelas junto con una guía curricular para implementarla en ellas.

Robo-ed ofrece varios niveles de aprendizaje, en particular el que nos interesa por su característica de poder programar, mediante una computadora, es un módulo que podría ser el bloque de control programable es el de nivel intermedio, éste cuenta con sensores, servomotores, engranes, ruedas, ejes metálicos, etcétera.

Este robot educativo tiene la desventaja de que sus partes electrónicas no están encapsuladas y en circuitos electrónicos esto es más delicado debido a que los proyectos con los estudiantes deben ser considerados a manejarse con cierto uso rudo.

2.3 Desarrollos universitarios.

El desarrollo de los robots educativos también ha sido del interés de otras universidades alrededor del mundo y como ejemplo tenemos el del Departamento de Diseño Industrial de la Universidad de Tecnología de Eindhoven, en Holanda, cuyo desarrollo lleva por título:

2.3.1 "AdMoVeo: A robotic platform for Teacher Creative Programming to Designers" . (Sjriek Alers, 2009)

Este proyecto presenta el diseño de hardware y software de una plataforma robótica, para enseñar la programación a los estudiantes a través de elaborar un programa y ver el comportamiento en el robot educativo, el cual fue diseñado con un chasis de forma redonda en el cual están integradas dos ruedas controladas por motores, en este chasis también están montados los sensores que son dos lectores de línea en la parte inferior; tres sensores infrarrojos de distancia en los costados y en la parte frontal; dos sensores de luz también en la parte frontal, dos sensores de sonido a los lados y los sensores encoders de los motores acoplados a las ruedas.

Este proyecto fue diseñado primeramente con un microcontrolador PIC18F4550 y posteriormente en una nueva versión optaron por una tarjeta Arduino Diecimila que contiene un microcontrolador ATmega168 y la cual ya contenía muchas prestaciones como la comunicación inalámbrica.

Al robot educativo AdMoVeo se le pueden hacer varias observaciones importantes con respecto a las características que debiera tener una herramienta de esta clase, por ejemplo sólo tiene una forma de armarse, esto es, no se pueden construir otras formas que el diseñador tenga en mente; otra observación es que el sistema de percepción también está fijo con lo que se limita aún más el poder realizar diferenciación en los proyectos, con esto se limita mucho al estudiante, como lo menciona Mitchel Resnick (2005) del laboratorio de medios del MIT.

"*Cuando evaluamos nuestros kits de construcción consideramos resultados diversos como un indicador de éxito. Si las creaciones de los estudiantes de una clase todas son similares una con otra, sentimos que algo hicimos mal."*

Y en el caso del robot AdMoVeo al tener una forma definida no hay variedad en los resultados de construcción de los estudiantes.

Otro proyecto realizado recientemente es el de:

2.3.2 "Nuevos modelos de aprendizaje y desarrollo de la creatividad usando agentes robóticos." (Jovani Alberto Jiménez Builes, 2007)

En este proyecto se diseñó un robot educativo para los estudiantes de Colombia, la propuesta fue el dispositivo denominado "E-Smart" el cual pretende ser una plataforma integral para la apropiación de conocimientos de tecnología a través del estudio y desarrollo de sistemas robóticos.

Esta plataforma está constituida por un sistema de percepción que está formado por sensores de distancia por infrarrojo modelo GP2D12 y sensores de distancia por ultrasonido modelo SRF10; para el movimiento utiliza motores servos modificados para que puedan realizar giros de 360 grados pero a una velocidad menor por los engranes de reducción que tienen internamente.

El microcontrolador de este robot es de marca Motorola modelo MC68HC908AP, este diseño también está confinado a tener una sola forma de armarse o a no variar mucho la misma; utiliza módulos comerciales por lo que el costo de su

fabricación se incrementa, aunque su objetivo sea interactuar con otros robots para practicar sistemas colaborativos.

Un proyecto más de robot educativo proveniente de China es el de:

2.3.3 "Flexible Educational Robotic System for a Practical Course" (Houxiang Zhang, 2007)

Este robot educativo está basado también en el Mindstorm NXT de la empresa LEGO, el sistema de percepción está constituido por sensores infrarrojos y sensor ultrasónico; estos sensores son módulos separados que se pueden colocar en cualquier lugar del diseño del estudiante; de manera similar los motores son módulos independientes.

Para el bloque de control programable utilizaron una tarjeta SBC-2410x la cual está basada en un chip Samsung ARM9; con esta tarjeta es como si tuvieran una pequeña PC en el robot y por lo cual se tienen las prestaciones de comunicación como Wireless, USB, RS232, Ethernet. Este proyecto es muy similar al de Mindstorm pero con la desventaja de la tarjeta de control que trabaja como bloque de control programable, ya que al ser prácticamente una PC su costo se incrementa en demasía.

Debido al conjunto de características importantes que han desarrollado los investigadores del grupo de medios del MIT, la mayoría de los proyectos encaminados al diseño de robot educativos, como se pudo observar anteriormente, se basan en el robot educativo de LEGO.

El presente proyecto de tesis pretende desarrollar un robot educativo con similares características, pero con la particularidad de que los materiales con los que se construya el robot sean adquiridos fácilmente a través de proveedores locales tanto en partes mecánicas como electrónicas, y proponiendo una ventaja adicional la cual permita construir al robot educativo por el propio alumno así como utilizarlo ya armado para que los alumnos realicen actividades de propuesta de soluciones

a problemas planteados o simplemente como ejercicio de diseño para aumentar su conocimiento.

El software de programación para este proyecto de tesis, ya fue desarrollado en el proyecto “Diseño y desarrollo de un Compilador Visual para la enseñanza de la robótica básica” (Tonche Ronny, 2010), con el cual el estudiante o usuario podrán programar las acciones para su estructura armada y descargarla al bloque de control.

Resumen.

En este capítulo se analizaron algunos de los robots educativos comerciales que existen así como los proyectos de este tipo de robots de universidades alrededor del mundo. Se pudo observar que todos se basan en el Mindstorms NXT 2.0 de la empresa LEGO, en el siguiente capítulo dará un resumen de las teorías del aprendizaje en las cuales se sustenta éste producto, y se definirán los conceptos de los componentes utilizados en el desarrollo del presente proyecto de tesis.

CAPÍTULO 3

Teorías del aprendizaje y componentes en los que se basa el desarrollo del robot.

3.1 Teorías del aprendizaje.

"Una teoría del aprendizaje es el conjunto de ideas que tratan de explicar lo que es el conocimiento y como éste se desarrolla en la mente de las personas."**(**Arón Falbel, 1993).

El construccionismo es una teoría del aprendizaje propuesta por Symour Papert, en la cual propone que el estudiante construya su propio conocimiento a partir de sus propias experiencias.

Las experiencias las adquiere el estudiante de su ambiente, por lo tanto se tiene que establecer al estudiante en un ambiente de acuerdo a lo que se quiera que el estudiante aprenda, siempre construyendo su conocimiento.

"El crear mejores oportunidades para que los educandos puedan construir su conocimiento, ha conducido a Papert y a su equipo de investigación del Instituto Tecnológico de Massachusetts a diseñar varios conjuntos de "materiales de construcción" para niños, así como escenarios o ambientes de aprendizaje dentro de los cuales, éstos pueden ser mejor utilizados", **(**Arón Falbel, 1993).

Para Papert el uso de la tecnología dentro de los diseños de los materiales de construcción es parte fundamental en el proceso de la construcción del conocimiento.

"Así la computadora podrá tener efectos más fundamentales en el desarrollo intelectual que el que han tenido otras tecnologías; por poner al sujeto del aprendizaje en un tipo de relación cualitativamente nueva con un dominio importante del conocimiento, el aprendizaje se torna más activo y auto dirigido." (Obaya Adolfo, 2003).

Uno de los resultados de estos ambientes de aprendizaje es el robot educativo Mindstorms NXT de la empresa LEGO y en el cual está basado el presente proyecto de tesis.

3.2 Elementos en los que se basa el desarrollo.

En el diseño y construcción se utilizaron componentes electrónicos y mecánicos los cuales serán descritos en este capítulo para comprender el funcionamiento de cada uno de ellos y después como trabajan en conjunto.

3.2.1 Microcontrolador.

Un microcontrolador es un circuito integrado que trabaja como una computadora, esto es, que incluye las tres unidades funcionales, un CPU, una memoria y periféricos de entrada y salida, pero todos estos componentes están dentro del microcontrolador, es por ello que al inicio sus creadores lo llamaron microcomputador de un solo chip. Aunque todas las capacidades del microcontrolador son pequeñas en comparación con una computadora, con su microprocesador y sus componentes externos, es suficiente para el trabajo que fueron diseñados.

Para este proyecto se optó por un microcontrolador de la marca Microchip y los motivos para esta elección fueron:

- Ya se había trabajado con los PICs de esta marca.
- El costo es bastante accesible.
- La información que ofrece el fabricante es abundante y está disponible en la red.
- Las herramientas para programar y depurar son económicas o gratuitas.
- El proyecto va dirigido a estudiantes de nivel medio superior.

Y las motivaciones para elegir en especial el PIC 18F4550 fueron:

a) La memoria.- en este microcontrolador hay tres tipos de memoria:

- Memoria de programa
- Memoria de datos RAM
- Memoria de datos EEPROM

Debido a la arquitectura Harvard con la que trabaja este microcontrolador se puede tener acceso al mismo tiempo tanto a la memoria de programa como a la memoria de datos RAM, ya que tienen buses independientes.

La memoria de programa tiene una capacidad de 32 Kbytes y es direccionada mediante un contador de programa de 21 bits con lo que se puede acceder directamente a cualquier dirección. Figura 3.1.

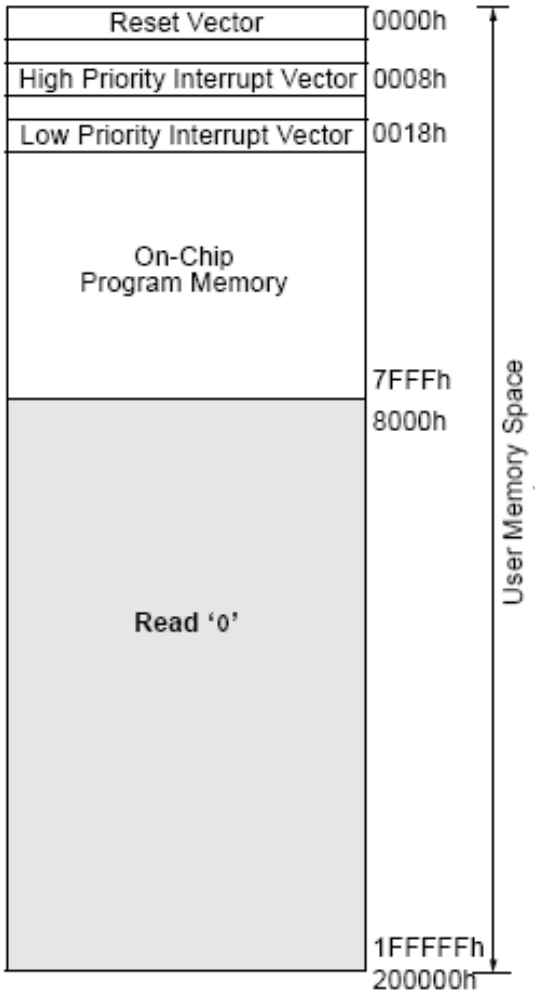

Figura 3.1 *Memoria de programa del PIC 18F4550.*

En esta memoria de programa se pueden almacenar hasta 16,384 instrucciones, ya que cada instrucción ocupa dos espacios.

Existen dos vectores de interrupción: en la dirección 0008h y la 0018h, uno es de alta prioridad y el otro de baja prioridad respectivamente.

Después de la dirección 8000h se leerán como “0” ya que esa área no está implementada.

Este microcontrolador también tiene la característica de poder escribir su propio programa en la memoria de programa con un control de firmware interno conocido como rutina de cargador o “bootloader”, con esta característica únicamente cuando se carga el firmware cargador se utilizará un programador de microcontroladores, posteriormente para descargar las instrucciones a la memoria de programa se utilizará alguno de los dos modos de comunicación con la PC, ya sea interfaz USB o interfaz serie asíncrona.

El módulo de control del proyecto podrá utilizar sólo uno de los dos modos del cargador, uno de ellos es por medio del cargador USB y el otro cargador es el de tipo serie asíncrono el cual utiliza el periférico USART (Universal Synchronous Asynchronous Receiver Transmitter), el módulo de control tendrá las dos conexiones pero sólo uno de los dos firmware del cargador se programará en el PIC, el usuario decidirá con cuál de estas dos opciones trabajará su robot y en cualquier momento puede cambiar de cargador.

La memoria de datos RAM es de tipo estática y en este microcontrolador se tiene un direccionamiento de 12 bits lo cual permite direccionar hasta 4096 bytes, pero únicamente pueden ser utilizadas 2048 bytes, esta memoria está dividida en 16 bancos de 256 bytes cada uno, figura 3.2

Con el registro BSR se selecciona el banco al cual se quiere acceder.

La memoria de datos contiene registros de función especial y registros de propósito general, los primeros son lo que utiliza el microcontrolador para manejar los diferentes periféricos que tiene, mientras que los de propósito general son los que ocupa el usuario para almacenar los datos de sus aplicaciones. Cuando se accede a una dirección no implementada se leerá como “0”.

La memoria de datos puede accederse de diferentes modos: el direccionamiento directo y el direccionamiento indirecto.

El direccionamiento directo es cuando la dirección del dato es la que está junto con la instrucción.

El direccionamiento indirecto permite al usuario acceder a una localidad de la memoria sin dar una dirección fija en la instrucción.

Para permitir que los registros más usados, SFRs y GPRs, se puedan acceder en un solo ciclo, el microcontrolador tiene implementado una instrucción Access Bank, significa un espacio de memoria de 256 bytes al inicio del banco 0, que no utiliza el registro BSR.

b) **Comunicación USB.-** el módulo de control del robot tendrá dos modos de comunicación con la PC, uno de ellos es la interfaz USB, y este microcontrolador contiene un periférico de interfaz serial USB de velocidad baja y velocidad alta, lo cual permite una rápida comunicación entre un servidor USB y el PIC. Mediante esta comunicación el PIC puede ser programado sin necesidad de un programador comercial, sino directamente desde la PC.

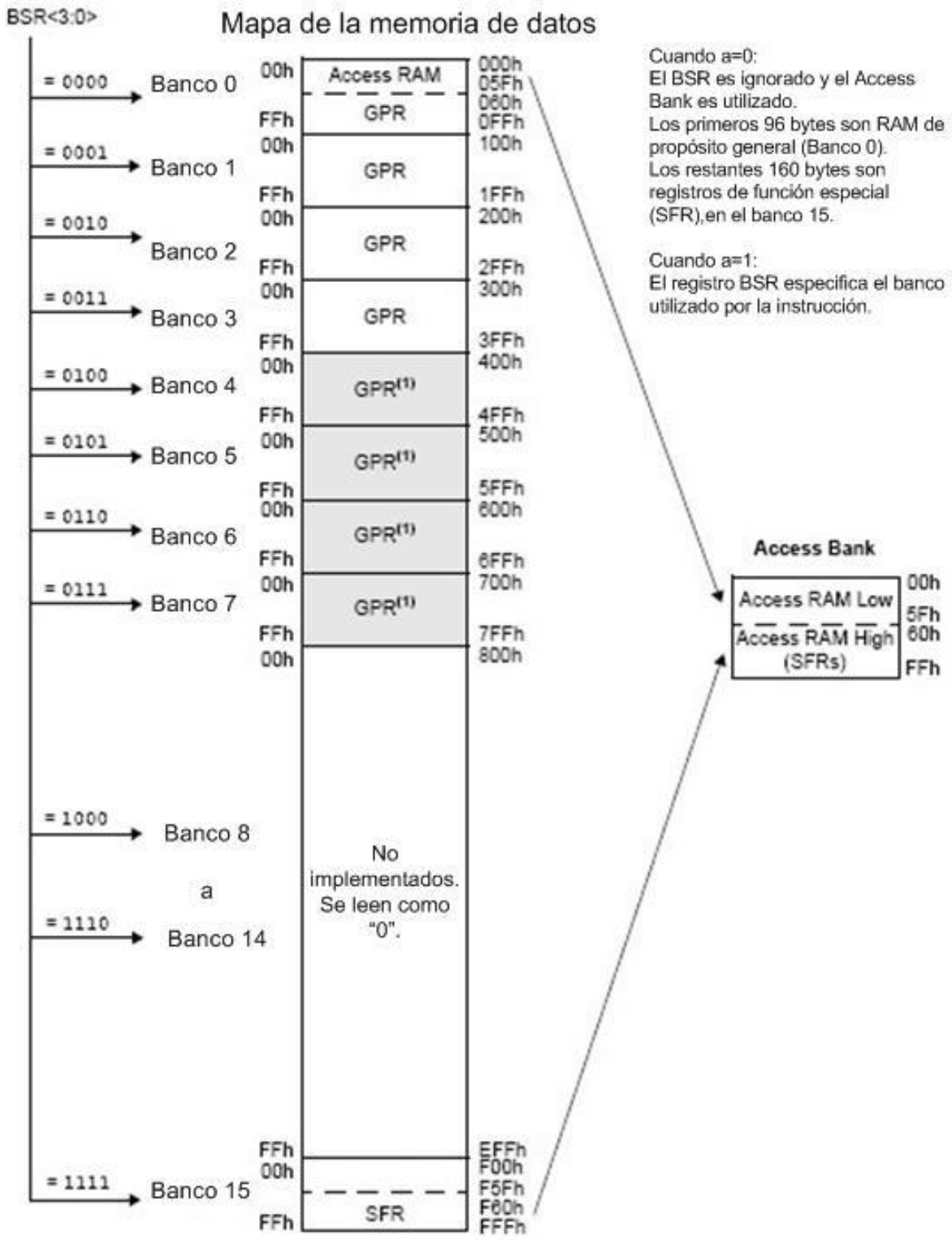

Figura 3.2 *Mapa de la memoria de datos del PIC 18F4550.*

c) **Comunicación serie asíncrona.-** este microcontrolador tiene un periférico para comunicación serie asíncrona que se puede configurar para trabajar en forma bidireccional simultánea. En el presente proyecto este periférico se utiliza para programa al microcontrolador mediante el firmware cargador que previamente fue programado en el PIC.

En el caso de los dos cargadores, la empresa Microchip proporciona el firmware cargador, de forma libre.

d) **Comunicación I2C.-** el PIC 18f4550 cuenta con un periférico para realizar una comunicación de tipo I2C, y que en este proyecto de tesis, se utilizará para establecer comunicación con el sensor de distancia y también se utilizará para poder agregar algún otro sensor o actuador al robot educativo.

e) **Convertidor analógico digital.-** el microcontrolador trabaja con la información de tipo digital por lo que para utilizar la información de los sensores una vez adaptada tiene que cambiarla a tipo digital ya que ésta viene en forma analógica. Para realizar esto el PIC dispone de un periférico llamado “convertidor analógico-digital” el cual se va a encargar de recibir el valor analógico del sensor y transformarlo a su valor digital. Ya con este valor el PIC lo comparará con el límite establecido por el usuario en el programa para poder ejecutar la tarea requerida. En el caso especial del microcontrolador 18F4550, éste tiene 13 canales que funcionan como entradas analógicas al convertidor analógico digital, y para este diseño se ocuparon como entrada para los cuatro sensores las terminales AN0, AN1, AN2 y AN3. El convertido analógico digital es un periférico del microcontrolador 18F4550 el cual permite la conversión de una entrada de señal analógica a su correspondiente valor digital, en el caso particular del PIC ese valor es de 10 bits. Para realizar esta conversión el microcontrolador utiliza el método de aproximaciones sucesivas que es el más empleado para los convertidores analógico-digitales (CAD).

3.2.2.- Micrófono Electret.

El sensor de sonido que se utilizó, es una variante del micrófono de condensador conocido como micrófono tipo **Electret**, en este tipo de micrófono el diafragma está hecho de un material plástico que al momento de su fabricación, queda cargado electrostáticamente y esta carga dura mucho tiempo; su funcionamiento es muy similar al de condensador normal pero con la ventaja de que no se requiere ninguna alimentación de voltaje para su funcionamiento. Además el micrófono tipo Electret comercial tiene internamente un pre-amplificador y es este preamplificador el que requiere el voltaje para trabajar, el cual es suministrado por medio de las terminales del micrófono (figura 3.3).

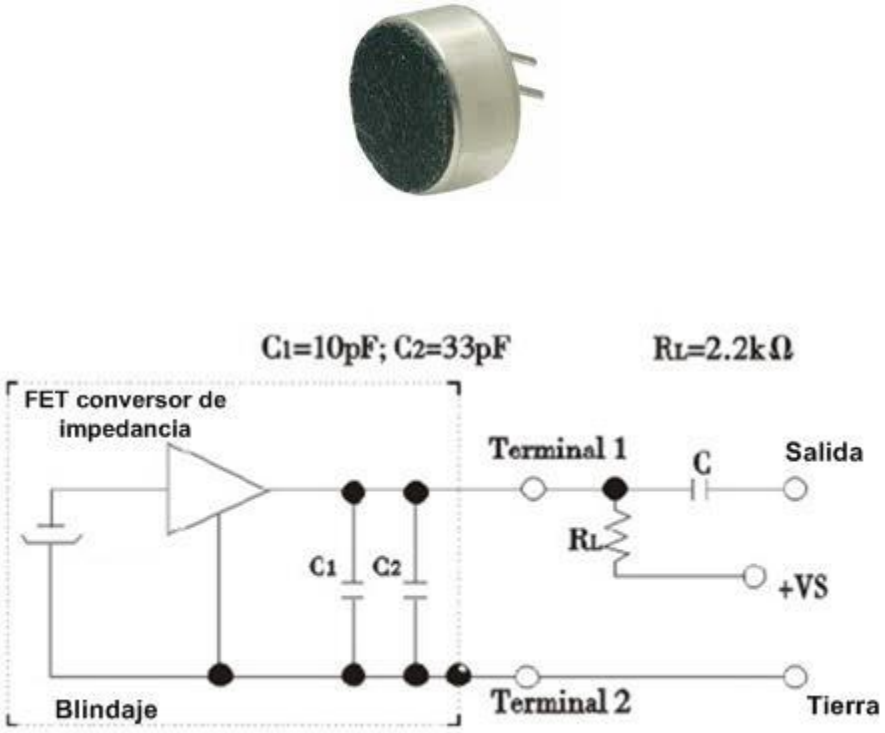

Figura 3.3 *Micrófono Electret.*

Las ventajas que llevaron a utilizar este tipo de micrófono en el sensor de sonido son las siguientes:

- Entrega una señal pre amplificada.
- Es económico.
- Es de tamaño pequeño.
- Su construcción lo hace resistente.
- Es sensible.
- Tiene buena respuesta en las frecuencias audibles.
- La tensión que requiere es de 2 a 12 v.
- Baja respuesta a tonos altos (no capta los del sensor ultrasónico).

3.3.3.- Transductor ultrasónico.

El sensor de distancia se basa en el fenómeno de reflexión, el cual consiste en el que las ondas generadas por algún evento al viajar por el espacio y chocar con algún objeto, son reflejadas y de acuerdo a la ley de reflexión el ángulo de incidencia es el mismo que el de reflexión. En el caso del sensor de distancia, que se diseñó, se genera un paquete de ocho ondas ultrasónicas, que tienen una frecuencia de aproximadamente 40 Khz; este paquete de ondas reflejadas llegarán al transductor que está configurado como receptor.

Este conjunto de ondas que llega al receptor es conocido como eco, el tiempo en que tarda en llegar el eco es el que se utiliza para realizar los cálculos y así conocer la distancia del objeto con el cual la onda fue reflejada.

En el caso del sensor de distancia se utilizó un microcontrolador PIC para poder generar la ráfaga de pulsos, detectar el eco, realizar el cálculo y comunicar la información al bloque de control, esta comunicación es de tipo serial con el protocolo I2C; el microcontrolador que se eligió fue el PIC16F690 de Microchip, este PIC tiene el periférico de comunicación como módulo SSP (Synchronus Serial Port) para realizar la comunicación serial síncrona, sólo en modo esclavo.

Para generar y recibir la señal ultrasónica se utilizó un transductor transmisor y receptor modelo UCM-R40K1, figura 3.4.

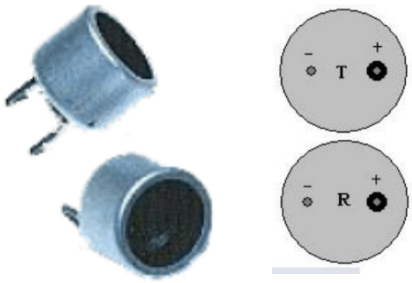

Figura 3.4 *Transductores ultrasónicos.*

Los sensores de distancia ultrasónicos realizan las mediciones en su zona útil de trabajo a la cual se le conoce como área de detección, la cual es el resultado de la diferencia entre la distancia máxima y la zona ciega o también conocida como zona muerta, que posea el sensor. La distancia máxima es la que posee el sensor a la cual puede detectar un objeto de acuerdo a sus características de fabricación y el circuito diseñado, la zona ciega es el área cercana al sensor en la cual no se puede detectar o calcular la distancia de un objeto. En la figura 3.5 se pueden observar las respectivas áreas.

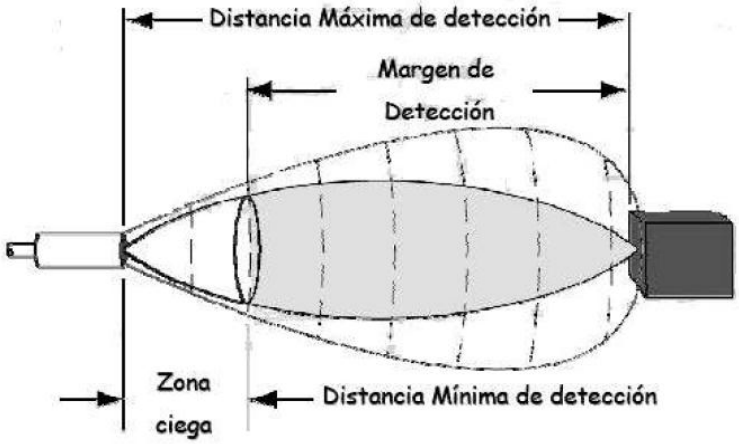

Figura 3.5 *Zonas de trabajo del transductor ultrasónico.*

3.3.4 Motores.

Los motores que se utilizaron en el desarrollo son los motores de reducción de plástico comerciales figura 3.6. Estos motores trabajan con un voltaje 6 a 12 v. CD. Tienen una relación de reducción de 1:180.

Para obtener esta reducción se utiliza un tren de engranajes, el eje motor tiene un engrane de 8 dientes, el segundo engrane es reductor y tiene una relación de 36/8 dientes, el tercer engrane también reductor de 36/9 dientes y el cuarto con una relación de 35/14 y por último el engrane conductor es de 32 dientes.

Figura 3.6 *Motor con engranes reductores.*

Para obtener la relación total de reducción utilizamos la siguiente fórmula:

$$\frac{1}{NDEM} * RE1 * RE2 * RE3 * NDEC = Número\ de\ vueltas$$

Fórmula 1.

Donde:

- NDEM = Número de dientes del eje motor.
- RE1, RE2 y RE3 = La relación de cada engrane reductor.
- NDEC = Número de dientes del engrane conductor.

- Número de vueltas = son las que necesita dar el eje motor para que el engrane conductor realice una vuelta completa.

Sustituyendo los valores en la fórmula anterior:

$$\frac{1}{8} * \frac{36}{8} * \frac{36}{9} * \frac{35}{14} * 32 = 180$$

En este caso el eje motor requiere dar 180 vueltas para que el engrane conductor de una vuelta, y se representa 1:180.

Encoder.

Un encoder es un transductor rotativo que se encarga de transformar un movimiento angular en una serie de pulsos digitales. Los pulsos generados pueden ser utilizados para controlar los giros de un motor. Las señales eléctricas del encoder pueden ser procesadas por un microcontrolador para una tarea específica. La generación de pulsos de un encoder puede ser por medio de una exploración fotoeléctrica, con el uso de un fototransistor figura 3.7, o una exploración magnética.

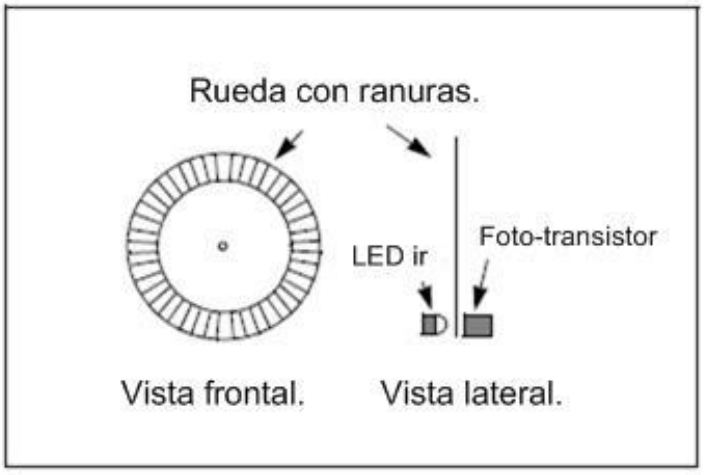

Figura 3.7 *Encoder de exploración fotoeléctrica.*

Control por PWM

En el control de velocidad de los motores se utilizó la técnica de modulación por ancho de pulso (PWM), este control se basa en aplicar un determinado voltaje durante cierto tiempo y otro tiempo apagado. Al realizar este procedimiento rápidamente, parecería que el voltaje es constante. Este apagado y encendido genera un tren de pulsos que va a tener un periodo, un estado alto conocido como "duty cycle" o ciclo de trabajo (T_{ON}), y un estado bajo o apagado (T_{OFF}). En la figura 3.8 se pueden observar las diferentes partes de una señal de PWM. Entre más largo sea el tiempo del ciclo de trabajo más se acerca a que el motor reciba el 100% de voltaje en promedio, y si disminuimos ese ciclo el motor recibirá menos voltaje en promedio y con esto se puede variar la velocidad del motor.

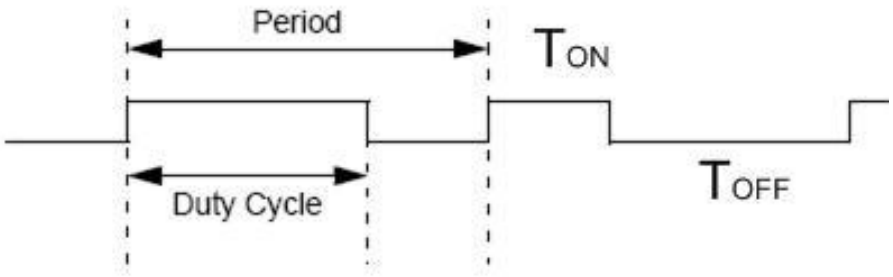

Figura 3.8 *Ciclo de trabajo del PWM.*

El microcontrolador PIC 18F4550 tiene dos módulos internos denominados CCP (Capture/Compare/PWM), cada módulo trabaja con un registro de 16 bits, y se puede configurar para trabajar en modo "Captura" de 16 bits, modo "Comparador" de 16 bits o en modo de modulación por ancho de pulso (PWM). Para el control de los motores se utiliza un módulo PWM para cada uno de los motores, es por esto que se puede controlar solo dos motores de forma directa por el microcontrolador.

En el control de los motores se debe poder seleccionar si queremos que el motor gire en un sentido u otro, para los motores de CD que se utilizaron en el presente proyecto, el cambio de giro se realiza invirtiendo la polaridad del voltaje aplicado, por lo que para tener este control se recurrió a el circuito integrado L293D, el cual es conocido como un puente “H” , este chip está construido especialmente para poder controlar el sentido de un motor DC, por medio de dos señales lógicas, también con este integrado se pueden manejar voltajes diferentes en los motores con relación a las señales lógicas. Figura 3.9

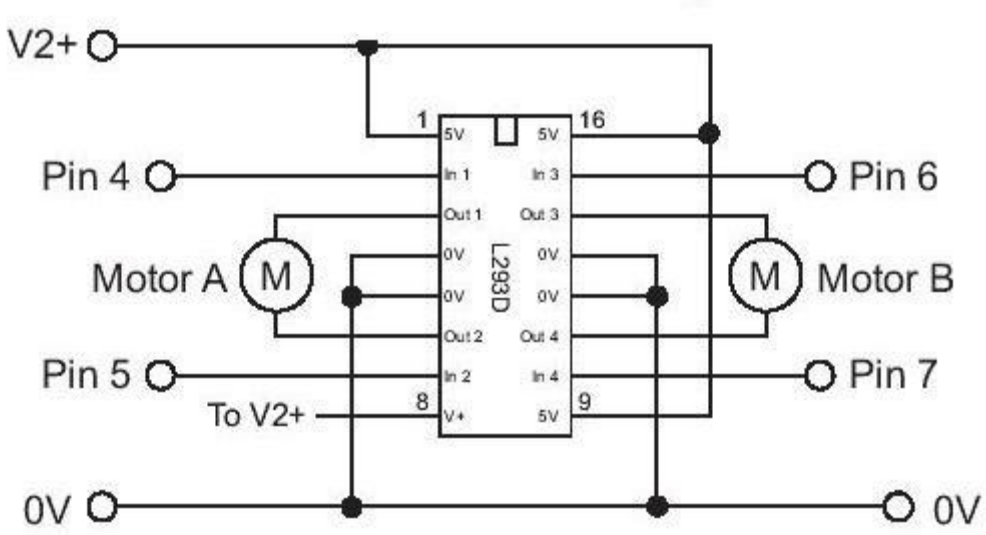

Figura 3.9 *Control de motores CD, con un circuito integrado L293D.*

3.2.5 Display LCD.

El display LCD o pantalla de cristal líquido es un dispositivo de salida para la visualización de gráficos como pueden ser caracteres, símbolos e incluso dibujos y el control de funcionamiento lo tiene un microcontrolador que viene integrado en el display. El display que se utilizó en el presente proyecto fue el que está gobernado por el microcontrolador Hitachi HD55780, figura 3.10.

El cual dispone de 2 filas con 16 caracteres cada una de ellas y los caracteres están formados por una matriz de 5 x 7 puntos (pixeles), las características que nos da el fabricante son las siguientes:

- Pantalla de caracteres ASCII.
- Caracteres kanji y griegos.
- Memoria de 40 caracteres por línea de pantalla.
- Informa de la dirección de la posición del carácter.
- Movimiento del cursor.
- Programación de 8 caracteres por el usuario.
- Conexión a un procesador con interfaz de 4 u 8 bits.

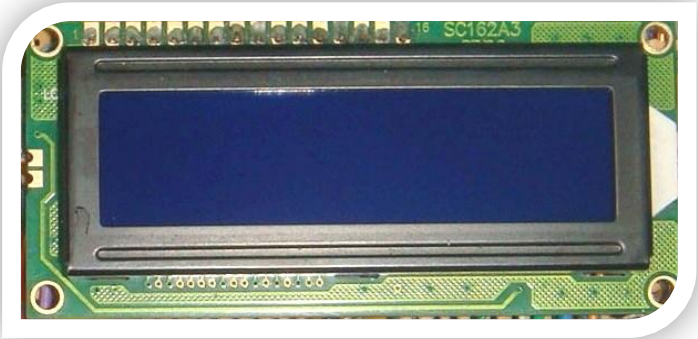

Figura 3.10 *Display LCD, de 2x16.*

Resumen.

En este capítulo se dio una introducción a los fundamentos de las teorías del aprendizaje en que se deben basar los robots educativos. Se mostraron las características de los componentes que serán utilizados en el siguiente capítulo para construir los módulos que conforman al robot del presente proyecto de tesis.

CAPÍTULO 4

Construcción de los componentes del sistema de desarrollo.

En este capítulo se tratará el diseño del robot educativo o como un dispositivo electrónico al cual llamaremos bloque de control y al que se le conectan sensores y motores para que realice una tarea programada. Esta tarea se programa en una PC mediante el software diseñado especialmente para este propósito, la descargar al robot se efectúa por medio de una interfaz serial.

4.1 Estructura del robot educativo.

Para la realización del robot educativo se diseñaron los diferentes módulos que lo conforman, estos son:

- Módulo de percepción.
  - Sensor de sonido.
  - Sensor de luz.
  - Sensor de distancia.
  - Sensor de toque.
- Módulo de proceso.
  - Comunicación con la PC.
  - Estructura del despachador de tareas.
  - Manejo de los periféricos.
- Módulo de actuadores.
  - Motores con encoder.
  - Manejo del display.

En la figura 4.1 se puede observar el diagrama a bloques del diseño del robot.

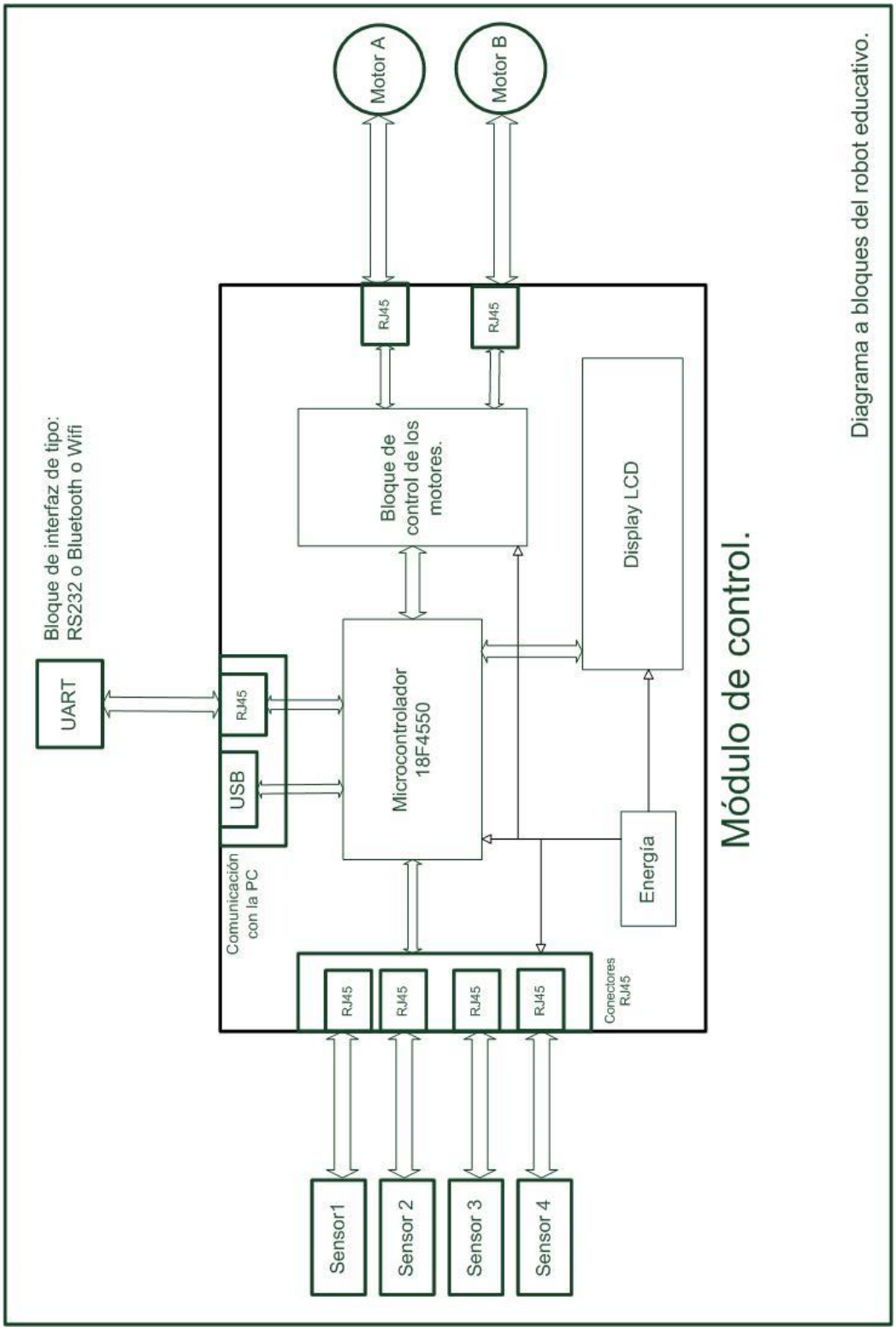

Figura 4.1 *Diagrama a bloques del robot educativo.*

4.2 Desarrollo del módulo de entrada.

Una característica del robot educativo, es que puede reaccionar ante los cambios de su entorno, para así modificar su conducta ante este cambio, de acuerdo a su programación y para obtener esa información del entorno en el que funcionará el robot educativo, se precisa de sensores que capten esa información. Estos cambios pueden ser el de la temperatura, el sonido ambiental, objetos que estén alrededor del robot, la iluminación, etcétera. Estos sensores registran el fenómeno mediante un transductor y éste entrega un nivel de voltaje que varía de acuerdo a la variación del fenómeno. Esta información es de tipo analógico y por lo regular son de pequeña amplitud y corto tiempo por lo que se tienen que adaptar al microcontrolador para usar esa información, esta adaptación consiste en amplificar la información ya sea en amplitud o en tiempo. Esa etapa de adaptación depende del fenómeno que se quiera registrar y del tipo de sensor que se utilice.

Para el desarrollo de los sensores se construyeron con base en tres partes, primero el diseño del diagrama electrónico, en segundo lugar la construcción física del sensor siguiendo el diagrama electrónico y por último se elaboró el código en lenguaje ensamblador con el cual el microcontrolador haría uso del sensor.

4.2.1 Sensor de sonido.

El sensor del sonido que se diseñó para el robot educativo consiste de un micrófono para captar las ondas audibles del espacio en que se encuentra el sensor, y éstas son amplificadas por dos etapas, cada una por un amplificador operacional LM324, figura 4.2. Como lo que se requiere es saber si el sonido ambiental rebasa cierto límite preestablecido, no precisamos de una amplificación de calidad ni que se requiera eliminar el offset, sino que se capte y amplifique el nivel en voltaje del sonido. Este será entregado a una de las entradas del convertidor analógico digital para que el PIC registre cuando ese nivel rebase el umbral que fue establecido por programa. Con esta información de acuerdo al programa el PIC realizará ciertas tareas.

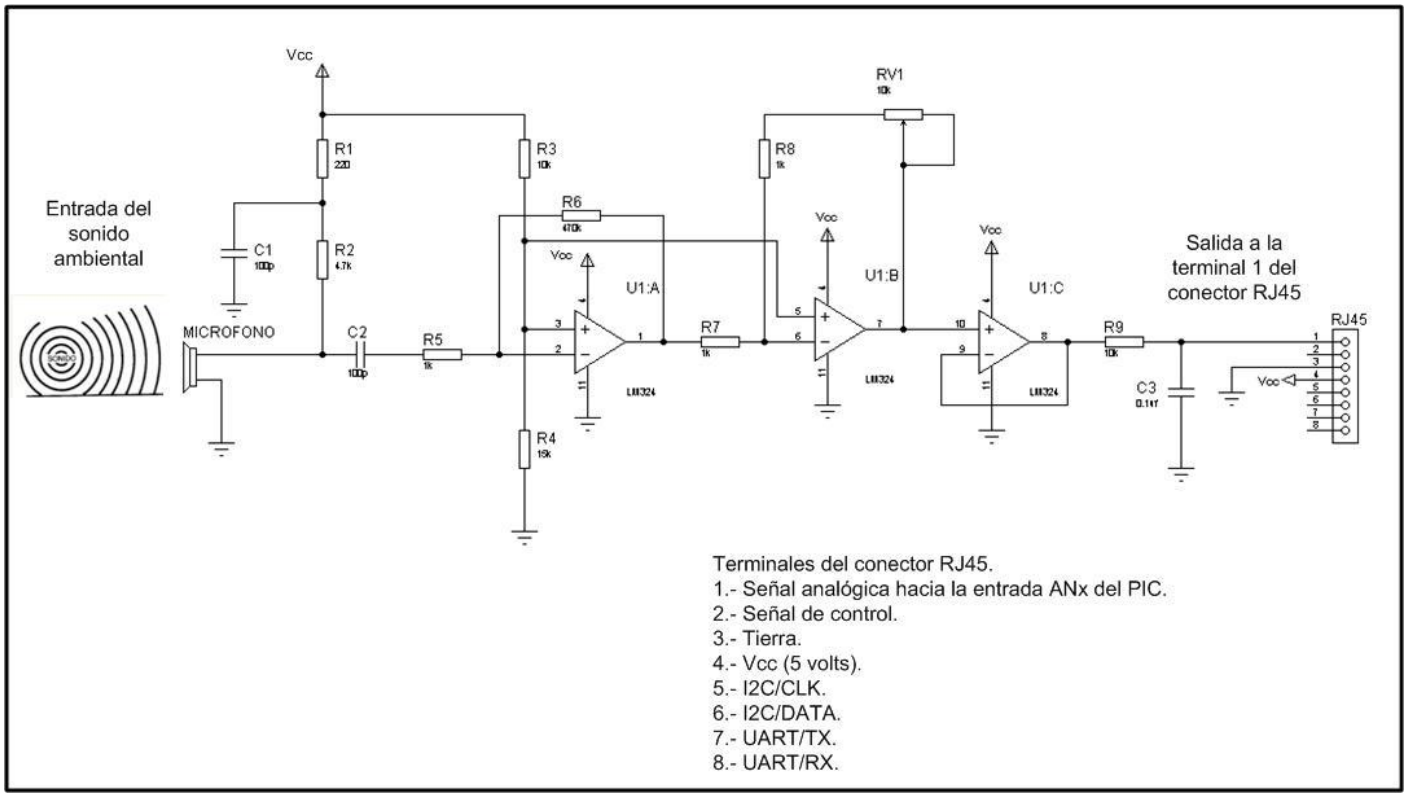

Figura 4.2 *Diagrama del sensor de sonido*.

La señal pre-amplificada que entrega el micrófono es aplicada en la primera etapa compuesta por un amplificador operacional LM324, figura 4.2, el micrófono entrega una señal entre 0.1 mv y .3 mv a través de la resistencia R5. La primera etapa entrega una señal amplificada 470 veces por lo que a la entrada de la segunda etapa tenemos una señal entre 0.047 volts y 0.141 volts y al ser amplificada 10 veces, nos entrega entre .47 y 1.41 volts la cual podemos usar para que el microcontrolador por medio del convertidor analógico digital pueda emplearla.

El código fuente en lenguaje ensamblador del sensor de sonido se puede observar en la figura 4.3, con los comentarios que tiene el código se puede entender como ejecuta las instrucciones el microcontrolador.

```
Sonido0     ;**FUNCION DEL SENSOR DE SONIDO CON EL ADC0*******
;******CONFIGURACION_ADC0
;{          INCF        FSR1L                   ; Como el bit de verificación del ADC está en el siguiente registro incrementamos el apuntador
            BTFSC       POSTDEC1,0; verificamos el bit SEMADC0 para saber si está ocupado el periférico y después se decrementa  el apuntador
                                                ; para dejarlo igual
            RETLW       0x00
            BSF         TRISA,0                 ; Bit 0 del puerto A como entrada, AN0
            MOVLW       0x0B
            MOVWF       ADCON1                  ; Configuramos AN0-AN3 como analógicas
            MOVLW       0x14
            MOVWF       ADCON2                  ; Seleccionamos los tiempos de adquisición
            INCF        FSR2L                   ; Como el bit SEMADC0 del ADC está en el siguiente registro incrementamos el apuntador
            BSF         POSTDEC2,0; Avisamos que está ocupado el periférico ADC0
;}
;*******VERIFICACION DEL Sonido0************
;{
VERIFICACIONSON0
            CLRF        ADCON0                  ; Seleccionamos el canal AN0
            BSF         ADCON0,0                ;Habilitamos el modulo ADC
            BSF         ADCON0,GO               ;inicia la adquisición GO
            NOP                                 ;  microsegundos  para realizar la conversión
            NOP
            NOP
            NOP
            NOP
            NOP
            NOP
            NOP
            NOP
            NOP
            NOP
            MOVF        VALORSON0,W             ;cargamos el valor que dio el usuario en WREG
            BTFSC       VARSON,0   ;probamos el bit "0" de la variable VARSON, si es 1 será una operación > y si es cero salta
            BRA         MAYORSON0
MENORKSON0
            CPFSLT      ADRESH                          ; saltamos aquí si fue una operación < y comparamos la medición del ADC con WREG
            BRA         SINTERMINARSON0                 ; si no es menor saltamos a SINTERMINAR0
            BRA         TERMINADOSON0                   ; si es menor se cumple la condición y saltamos a TERMINADOSON0
MAYORSON0
            CPFSGT      ADRESH                  ; saltamos aquí si fue una operación > y comparamos el valor medido por ADC con WREG
SINTERMINARSON0
            RETLW       0x00                    ; No hubo salto no ha terminado la función y salimos
;}
;***********TERMINAR EL Sonido0***************
;{
TERMINADOSON0                                   ; la condición se cumplió
            INCF        FSR2L                   ; Como el bit de SEMADC0 del ADC está en el siguiente registro incrementamos el apuntador
            BSF         POSTDEC2,0;Liberamos el periférico ADC0
            RETLW       0x01                    ; Terminó la función y regresamos con un "1" en WREG
;}
```

Figura 4.3 *Código del sensor de sonido.*

La construcción física del sensor del sonido se puede observar en la figura 4.4, en ella se ve el micrófono Electret, el circuito electrónico físicamente y el conector tipo RJ45, con el cual se conecta al módulo de control.

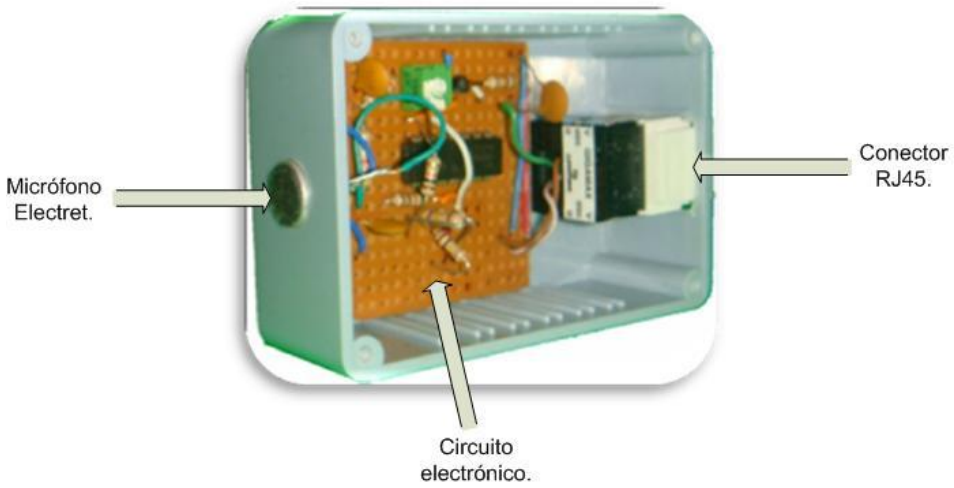

Figura 4.4 *Sensor de sonido.*

4.2.2 Sensor de luz.

El sensor de luz que se diseñó está basado en el fototransistor L14G1, el cual tiene una ventana en su encapsulado para dejar pasar la luz y cuando la recibe realiza un cambio en la corriente de base y conduce más o menos corriente de colector. Esta variación de corriente provoca una variación de voltaje que se amplifica por medio del transistor 2N222 (figura 4.5), y es el que se entregará a la entrada analógica del PIC para convertirlo a su respectivo valor digital y compararlo con el límite establecido por el usuario en el programa y así realizar las tareas necesarias.

El sensor de luz puede captar la iluminación del ambiente en que se encuentra y también se le colocó una led de luz blanca para que funcione como fuente de luz y sea registrada por el fototransistor. Esto con la finalidad de poder utilizar al sensor en superficies cercanas y poder registrar si es una superficie clara u obscura, y que regularmente se usa en las construcciones de robots seguidores de línea.

Esta fuente de luz es controlada desde el programa por lo que se puede encender o apagar de acuerdo al programa diseñado por el usuario. Para encender dicho led el PIC manda un señal alta o 1 lógico a través de la resistencia R4, poniendo en saturación al transistor Q1, el cual funciona como conmutador y de esta forma cerrar el circuito formado por el led D1 y la resistencia R1.

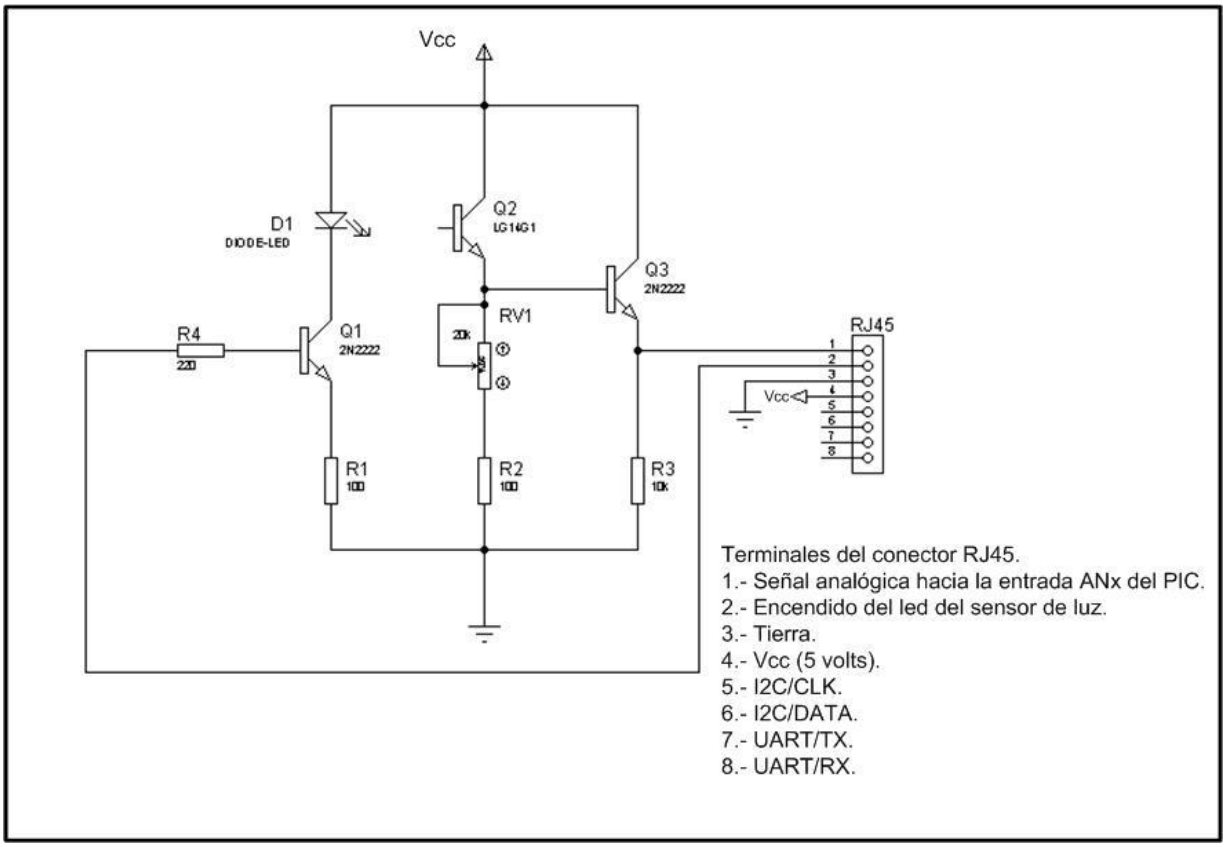

Figura 4.5 *Diagrama del sensor de luz.*

El código fuente del sensor de luz se puede observar en la figura 4.6, con sus respectivos comentarios.

```
SLuz0       ;**************FUNCION DEL SENSOR DE LUZ CON EL ADC0***************
;******CONFIGURACION_ADC0
;{
            INCF    FSR1L               ; Como el bit de verificación del ADC esta en el siguiente registro incrementamos el apuntador
            BTFSC   POSTDEC1,0; verificamos el bit SEMADC0 para saber si está ocupado el periférico y después se decrementa  el apuntador
                                        ; para dejarlo igual
            RETLW   0x00
            BSF     TRISA,0             ;Bit 0 del puerto A como entrada, AN0
            MOVLW   0x0B
            MOVWF   ADCON1              ; Configuramos AN0-AN3 como analógicas
            MOVLW   0x14
            MOVWF   ADCON2              ; Seleccionamos los tiempos de adquisición
            BCF     TRISA,5             ;Bit 5 del puerto A como salida
            BCF     PORTA,5             ;Ponemos a 0 el RA5
            BTFSC   VARLUZ,4            ;Verificamos si se prende el led del sensor si es 0 salta led OFF
            BSF     PORTA,5             ;Se enciende el led ON
            INCF    FSR2L               ; Como el bit SEMADC0 del ADC está en el siguiente registro incrementamos el apuntador
            BSF     POSTDEC2,0;Avisamos que está ocupado el periférico ADC0
;}
;*******VERIFICACION DEL SLUZ0************
;{
VERIFICACIONLUZ0
            CLRF    ADCON0              ; Seleccionamos el canal AN0
            BSF     ADCON0,0            ;Habilitamos el modulo ADC
            BSF     ADCON0,GO           ;inicia la adquisición GO
            NOP                         ; 10 microsegundos para realizar la conversión
            NOP
            NOP
            NOP
            NOP
            NOP
            NOP
            NOP
            NOP
            NOP
            NOP
            MOVF    VALORLUZ0,W         ;cargamos el valor que dio el usuario en WREG
            BTFSC   VARLUZ,0            ;probamos el bit "0" de la variable VARLUZ, si es 1 será una operación > y si es cero salta
            BRA     MAYORKLUZ0
MENORKLUZ0
            CPFSLT  ADRESH              ; saltamos aquí si fue una operación < y comparamos la medición del ADC con WREG
            BRA     SINTERMINARLUZ0     ; si no es menor saltamos a SINTERMINAR0
            BRA     TERMINADOLUZ0       ; si es menor se cumple la condición y saltamos a TERMINADOLUZ0
MAYORKLUZ0
            CPFSGT  ADRESH              ; saltamos aquí si fue una operación > y comparamos el valor medido por ADC con WREG
SINTERMINARLUZ0
            RETLW   0x00                ;No hubo salto no ha terminado la función y salimos

;}
```

Figura 4.6 *Código del sensor de luz*.

El sensor de luz terminado se puede observar en la figura 4.7.

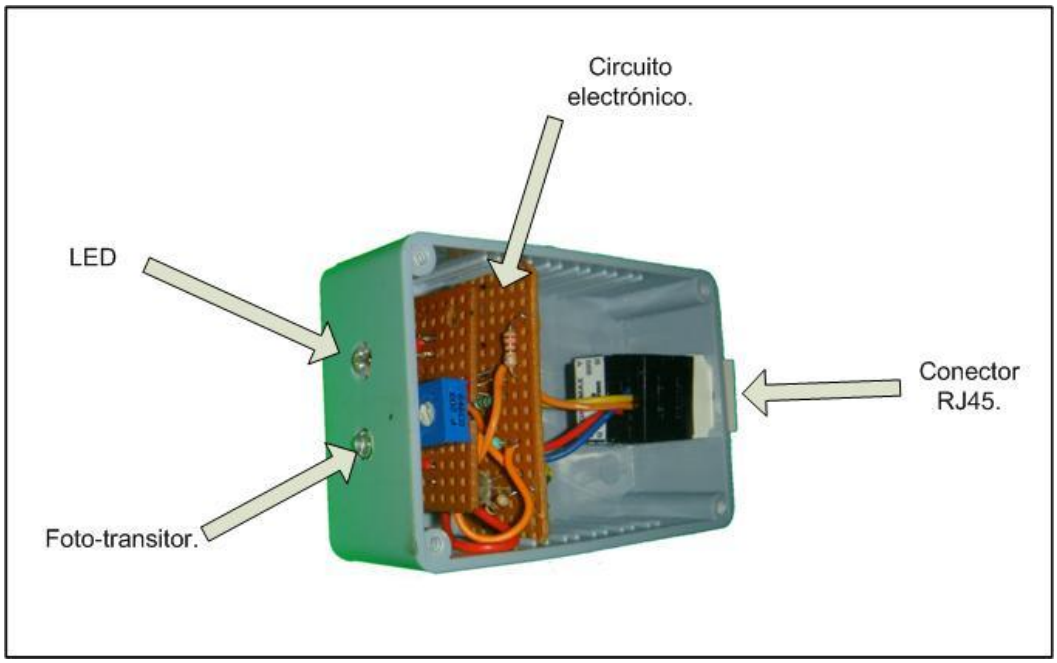

Figura 4.7 *Sensor de luz.*

4.2.3 Sensor de toque.

El sensor de toque lo forma un simple push botón que indicará al PIC si ha sido cerrado o no. El problema con estos tipos de interruptores es que pueden generar una serie de pulsos falsos debido a la naturaleza mecánica de su funcionamiento, esto es porque los materiales metálicos que deben hacer contacto para cerrar o abrir el interruptor, se unen y separan varias veces antes de estar completamente juntos, provocando varias señales y que pueden ser registradas por el PIC provocando un funcionamiento erróneo. Para eliminar estos pulsos no deseados se implementó un circuito anti rebotes, el cual está constituido por un divisor de voltaje y un capacitor como se observa en la figura 4.8. Al cerrar el interruptor con el primer contacto de las terminales, el capacitor se cargará a través de la resistencia R1 y esta carga se mantendrá por algún tiempo para que si existen separaciones y uniones de los contactos, no sean registrados a la salida del

sensor, únicamente se registrará una sola señal. En la figura 4.9, se puede observar el sensor de toque terminado.

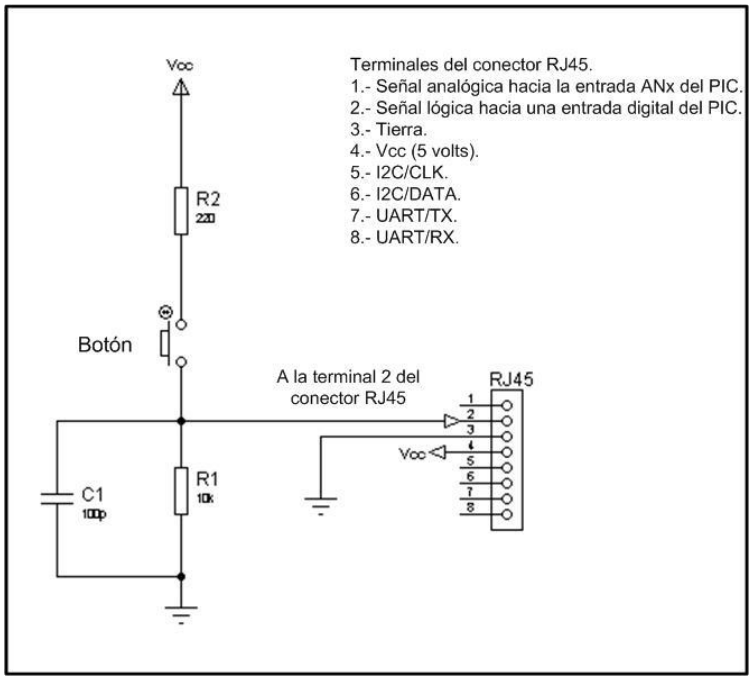

Figura 4.8 *Diagrama del sensor de toque.*

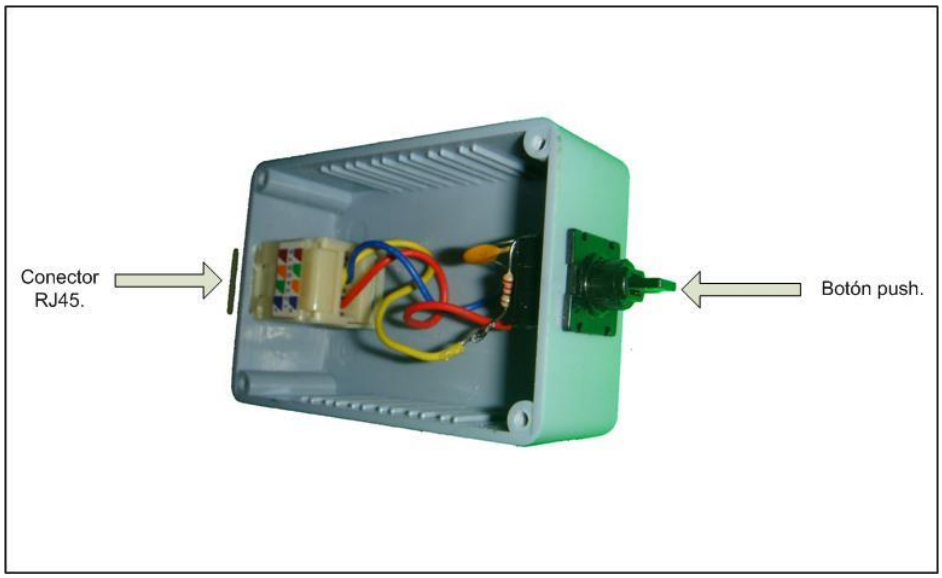

Figura 4.9 *Sensor de toque.*

4.2.2 Sensor de distancia.

El diagrama electrónico del sensor de distancia se diseñó como se observa en la figura 4.10. Con este circuito se puso a trabajar al transductor ultrasónico de la siguiente forma.

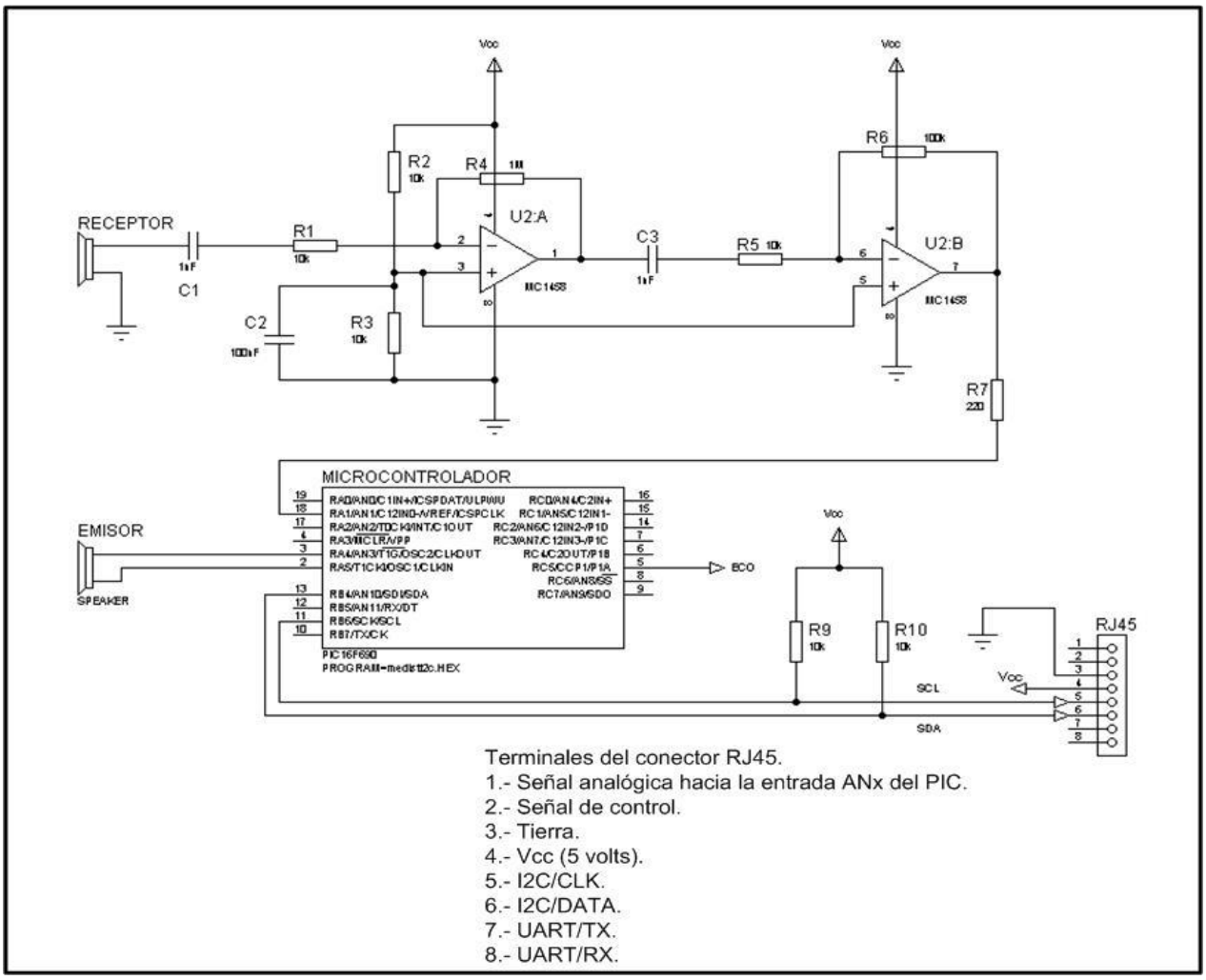

Figura 4.10 *Diagrama del sensor de distancia.*

Para generar la ráfaga de 9 pulsos se conectó el transductor emisor a dos pines de I/O de propósito general que en este caso fueron los pines 2 y 3; al iniciar el programa del PIC la salida del pin 2 se va a alto (1 lógico), mientras que la salida del pin 3 se pone en bajo (0 lógico), por un tiempo de 12 microsegundos. Después

de este tiempo se invierten los estados de los pines por otro tiempo de 12 microsegundos, completando un periodo de 24 microsegundos el cual es el periodo de una frecuencia de 41.66 Khz.

Al realizar este cambio un total de 9 veces, se genera el tren de pulsos o ráfaga, y esta ráfaga es generada cada 65 milisegundos, este tiempo de 65 milisegundos es controlado por una rutina de retardo.

Esta ráfaga es transmitida al espacio por el transductor colocado como transmisor, por otra parte el transductor colocado como receptor es conectado a dos etapas amplificadoras que son implementadas por dos amplificadores operacionales MC1458 como se puede ver en el diagrama (Figura 4.10).

La salida de la etapa de amplificación es conectada al microcontrolador el cual tiene configurado un comparador y está conectada al pin 18 que es la entrada inversa del comparador. La entrada no inversa tiene un voltaje de referencia interna a 0.5 v por lo que cuando el eco es captado por el transductor colocado como receptor y amplificado por los amplificadores operacionales y esta señal rebasa el umbral de la referencia, en la salida del comparador se obtiene un señal alta (1 lógico). Este nivel alto genera una interrupción en el microcontrolador con lo que se detiene el temporizador que se arranca al generar la ráfaga y con este temporizador se tiene el tiempo que se tardó el eco y con esta medida de tiempo se calcula la distancia del objeto con el que chocó la señal.

Parte del código del sensor de distancia se puede ver en la figura 4.11. Este código es el que utiliza el microcontrolador para pedir la distancia en microsegundos que ha obtenido el sensor.

El código del microcontrolador 16F690, con el cual trabaja el sensor de distancia se puede obtener en los anexos del presente documento.

En la figura 4.12 se puede ver el sensor de distancia ya terminado. En la figura se puede observar el microcontrolador 16F690 que se encarga de realizar todo el proceso para calcular el tiempo que tarda en regresar el eco de la señal generada.

```
SDISTANCIA
;************CONFIGURACION DE LA FUNCION SDISTANCIA**************************
            BTFSC   INDF1,0             ;verificamos el bit SEMSDIST para saber si está ocupado el periférico, INDF1 ES APUNTADOR
            RETLW   0x00                ; Como el bit está en "1" el periférico está ocupado salimos de la función
            BTFSC   FMT1,3              ;Revisamos bandera si es 0 no está configurado y salta para que se configure
            BRA     VERISD
            CALL    INI_I2C             ; se inicializa el periférico del I2C
            BSF     FMT1,3
            BSF     INDF2,0             ;avisamos que está ocupado el periférico con el bit SEMSDIST, INDF2 ES APUNTADOR
VERISD
            CALL    DATOI2C             ; llamamos a la función que le pide el dato de la distancia al sensor vía I2C
            MOVLW   0x02
            CPFSGT  DISTH
            RETLW   0x00
            MOVFF   DISTH,BIN           ;el dato que regresa el sensor es cargado en la variable DISTH y es pasado a la variable BIN
            MOVLW   0x84                ; Posición del cursor en el display es "0" en el renglón 1
            MOVWF   COMANDO  ; el valor lo cargamos en la variable COMANDO
            CALL    WRCMND              ; Se llama a la función para que ejecute la orden
            CALL    BCD                 ; llamamos a la función BCD para que muestre el valor de la variable BIN en el LCD
            MOVLW   0x87                ; posición para imprimir en el LCD la variable DISTL que entrego el sensor de distancia
            MOVWF   COMANDO  ; pasamos el valor a la variable COMANDO
            CALL    WRCMND              ; invocamos a la función para ejecutar el comando
            MOVFF   DISTL,BIN           ;pasamos el valor del registro DISTL a la variable BIN para su impresión
            CALL    BCD                 ; llamamos a la función BCD para que imprima el valor de BIN
            BTFSC   FMT1,4              ;Revisamos que operación lógica configuro el usuario si es < salta
            BRA     mayorque            ; si no salto es una operación de = ó >
                    ;*******************INSTRUCCIONES PARA LA OPERACION LOGICA MENOR QUE < ************************
menorque
            MOVF    DISTAUH,W ;vemos si son iguales el byte alto del entero de 16 bits
            CPFSEQ  DISTH               ; hacemos la comparación si es igual salta
            BRA     NOIGUAL             ; como no es igual es flujo se va a la rutina no igual
IGUAL                                   ; como fueron iguales los bytes ALTOS ahora hay que ver la operación menor que con los bytes BAJOS
            MOVF    DISTAUL,W  ;movemos el valor del usuario al WREG
            CPFSLT  DISTL               ; se realiza la operación menor que y si es verdadera salta
            RETLW   0x00                ; como no fue menor aun no se cumple la condición y la función aun no termina salimos con W=0
            BRA     TERMINADIST         ; salto se cumplió la condición y nos vamos a la rutina de terminado
NOIGUAL                                 ; rutina para cuando los dos bytes ALTOS no son iguales
            MOVF    DISTAUH,W ;movemos el valor ALTO del usuario a WREG
            CPFSLT  DISTH               ; realizamos la operación lógica < y si es cierta saltamos
            RETLW   0x00                ; fue falsa no se ha cumplido la condición salimos la función no ha terminado W=0
            BRA     TERMINADIST         ; hubo salto se cumplió la condición y nos vamos a la rutina de terminado
```

Figura 4.11 *Código del sensor de distancia que utiliza el bloque de control.*

En la figura 4.12 también se pueden encontrar los transductores que envían y reciben la ráfaga de pulsos.

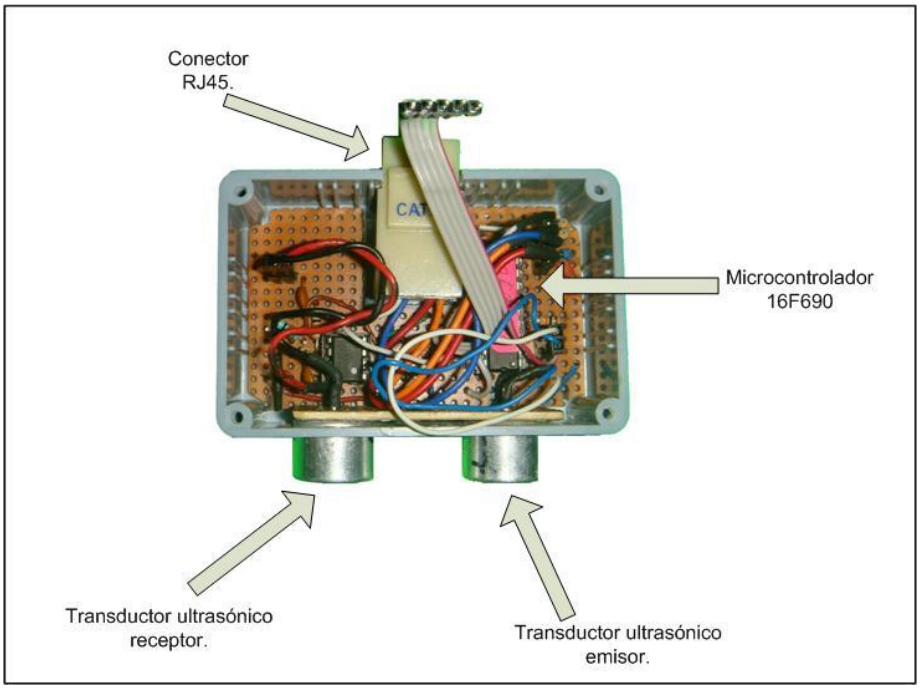

Figura 4.12 *Sensor de distancia ultrasónico.*

4.3 Módulo de control.

El módulo de procesamiento o control es el encargado de realizar las siguientes tareas:

- Comunicación con la PC.
- Cargar el programa creado por el usuario en la PC.
- Ejecutar el programa cargado.
- Recabar información del medio a través de los sensores.
- Procesar la información que envían los sensores.
- Activar los actuadores.

Todas estas tareas son realizadas por el microcontrolador PIC18F4550 que es la parte principal del módulo de proceso y es el encargado de tener el control del robot educativo, éste se encuentra en el bloque de control.

-Cargar el programa creado por el usuario.

El usuario diseña su programa mediante un software de programación visual utilizando iconos y dando valores a los parámetros de cada icono, al compilar este diseño se genera una archivo .hex que es el que se cargará en el microcontrolador mediante la comunicación USB o la interfaz RS232. Al PIC se le graba un programa bootloader para que no se necesite un programador especial, este bootloader se encarga de grabar el programa del usuario o de ejecutar el programa que ya está almacenado.

-Ejecutar el programa almacenado.

En el microcontrolador el firmware cargador al no detectar que se quiere grabar algún programa, pasa a ejecutar el que tenga en memoria, por lo que ahora toma el control el despachador que ahora es el encargado de que se ejecuten las instrucciones que contenga el programa del usuario.

El lenguaje que se utiliza para programar el robot educativo es un lenguaje visual en el cual en lugar de escribir las instrucciones que el usuario requiere, estas instrucciones se programan por medio de iconos, figura 4.13. Los iconos representan un conjunto de instrucciones y a los cuales se les ingresa ciertos parámetros requeridos por el mismo icono, figura 4.14. Estos iconos son llamados bloques.

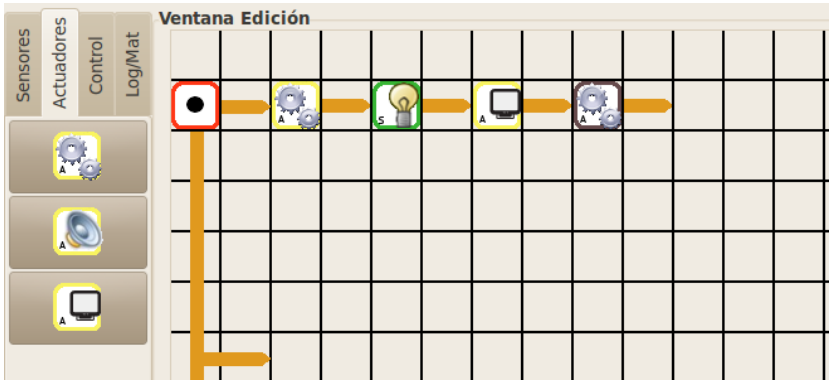

Figura 4.13 *Bloques del compilador visual.*

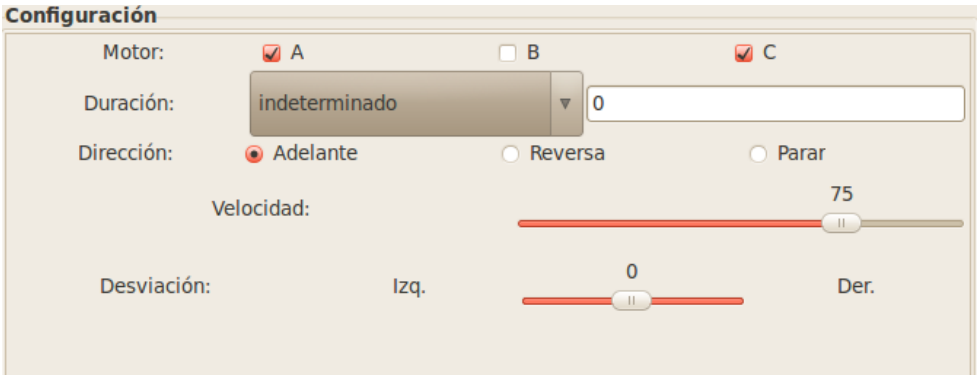

Figura 4.14 *Parámetros del bloque de motor.*

Para la programación como el ejemplo de la figura 4.13, el microcontrolador trabaja con la ejecución de los bloques de una forma lineal, esto es, que tiene que terminar la ejecución de un bloque para poder ejecutar el del bloque siguiente, y para poder realizar una programación en paralelo como en la figura 4.15, se requiere una rutina que se encargue de pasarle el control a la tarea que le corresponda.

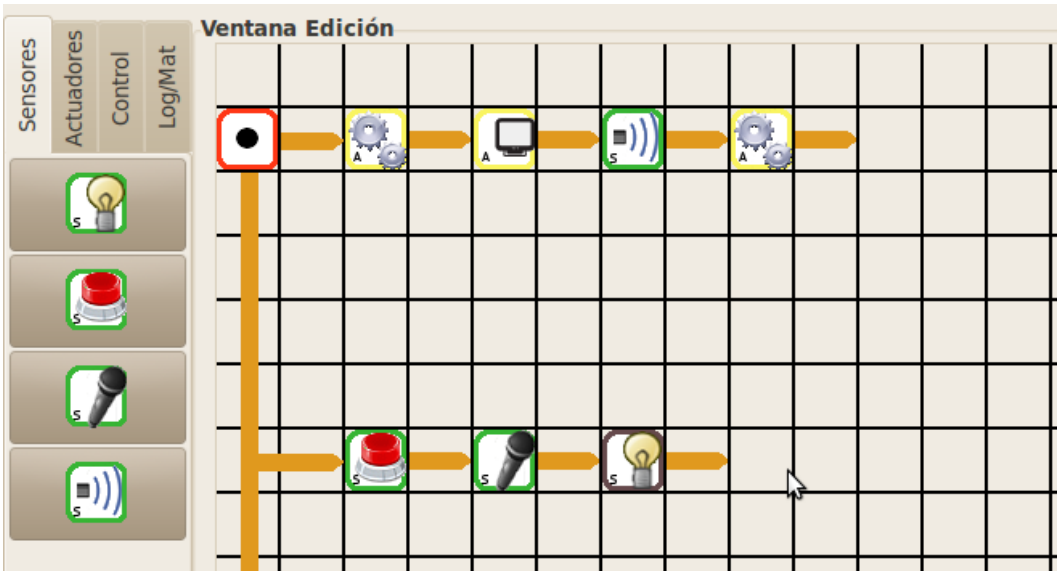

Figura 4.15 *Programación en paralelo.*

Cada una de las líneas que contienen a los bloques se le llamará “tarea”, en el ejemplo de la figura 4.15 el programa tiene dos tareas que realizar y cada tarea consta de uno o varios bloques.

La rutina que se encarga de organizar el control de los bloques se le conoce como despachador, este despachador se encarga de darle el control a la tarea que lo requiera y dentro de esa tarea al bloque que le corresponda ser ejecutado, y si el bloque dentro de su rutina tiene que esperar algún evento, éste le regresará el control al despachador para que le asigne el control a la otra tarea y dentro de ésta al bloque correspondiente(figura 4.16).

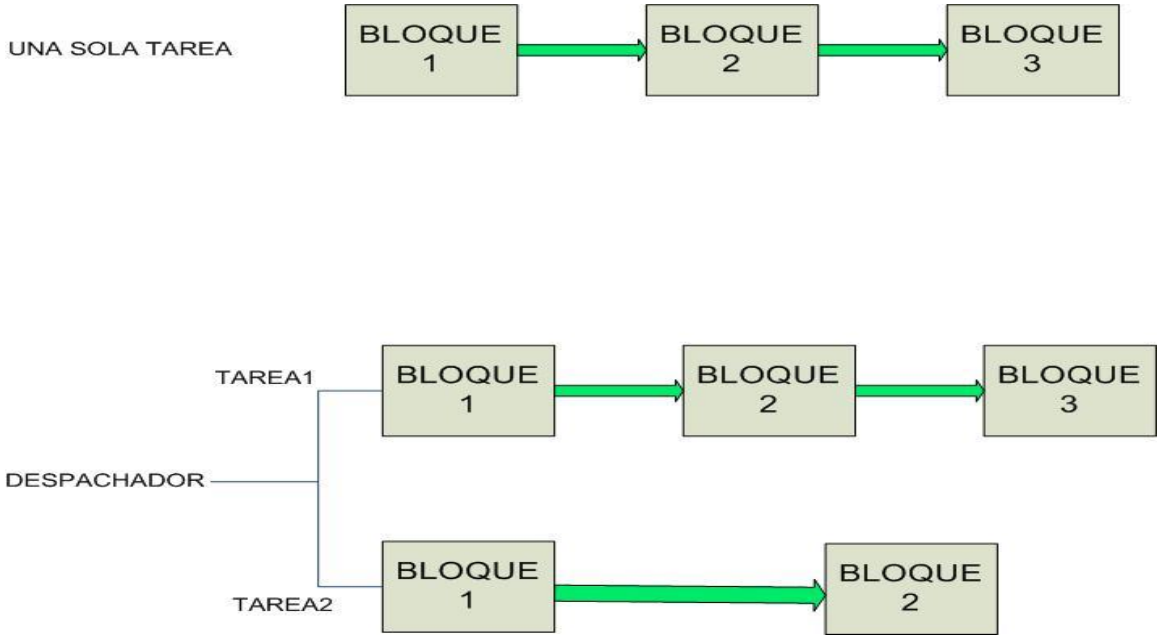

Figura 4.16 *Programación simple y programación en paralelo.*

La forma en que trabaja el despachador es con la sentencia de control SWITCH-CASE, su diagrama de flujo lo podemos observar en la figura 4.17.

El funcionamiento de la estructura es la siguiente: Se tiene una parte conocida como “main”, en la cual primeramente se llama a la tarea1, las tareas van a ejecutar cada bloque de acción mediante un control de Switch-case, por lo que el primer bloque será el caso 1, al ejecutar la función de ese bloque, primero verá que no esté ocupada esa función (Semáforo), después configurará al periférico, y si no se ha cumplido la condición que programó el usuario saldrá del switch-case, regresando el control al main. Éste llamará a la Tarea 2, y realizará el mismo procedimiento que la tarea 1, comprobará que no esté ocupado el periférico a

utilizar, y si está libre lo configura y revisa si se ha cumplido la condición programada por el usuario. Si no ha terminado hará un break y saldrá de la tarea 2 para regresar el control al main, y llamará nuevamente a la tarea1 y como el caso sigue siendo el 1, ejecutará el bloque 1 y esta vez sólo verificará si se ha cumplido la condición, si se cumplió la condición significa que el bloque 1 a terminado su trabajo, con esto se libera al periférico y se incrementa el caso. Por consiguiente ahora le toca trabajar al bloque 2 para realizar el mismo procedimiento.

Este procedimiento descrito lo realiza de la misma forma la tarea 2.

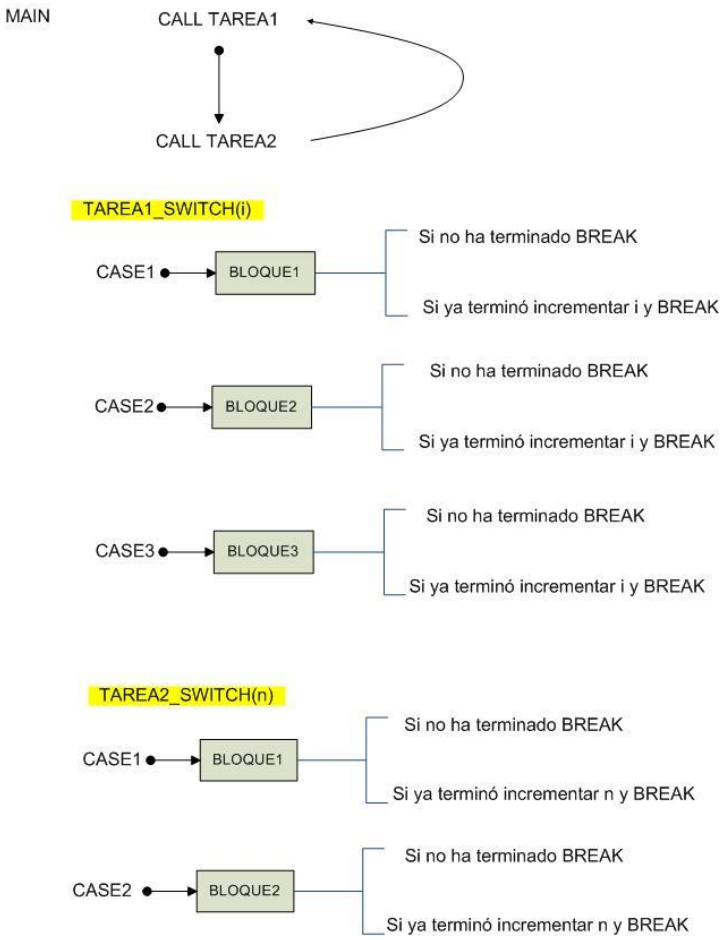

Figura 4.17 *Despachador con sentencias swicht - case*

El funcionamiento del despachador de acuerdo a los diagramas anteriores requiere que cada bloque este conformado como se muestra en la figura 4.18.

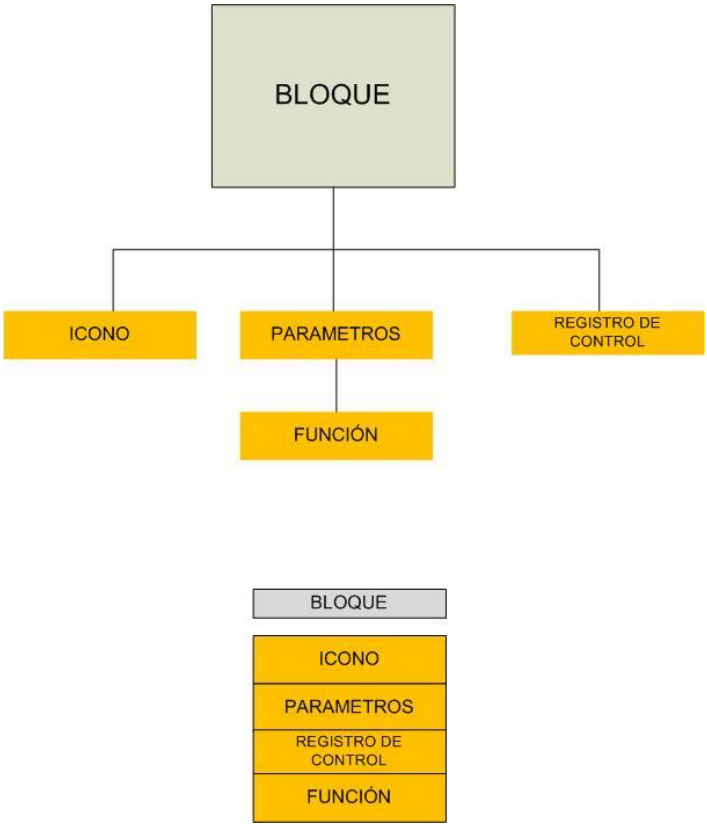

Figura 4.18 *Componentes de un bloque*

Para que el flujo del programa no se quede atorado en algún bloque, las rutinas de los mismos se diseñaron de forma que no se queden en un bucle infinito, por lo que si la rutina contiene alguna condición se verifica y sí ya se cumplió se avisa mediante una bit de bandera de su registro de control y se sale de la rutina dándola por terminada y si no se ha cumplido la condición, significa que no ha terminado y sale de esa rutina para pasar el control al despachador. Por lo que decimos que las funciones utilizan el método de “polling” en lugar de utilizar las interrupciones. La estructura que deben tener las funciones de los bloques se pueden observar en la figura 4.19.

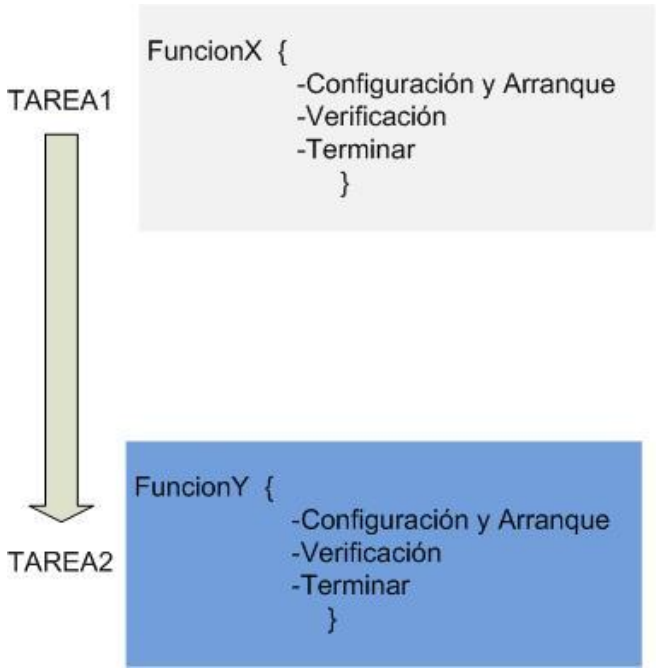

Figura 4.19 *Partes de las funciones.*

Como se puede observar las tres partes que conforman la función son:

- Configuración y arranque.
- Verificación.
- Terminar.

En la configuración se establecen las características del periférico del microcontrolador que se va a utilizar, por ejemplo el PWM, el convertidor analógico digital, etc. Dentro de esta configuración están los parámetros con los cuales va a trabajar la función y que se utilizan para cumplir la condición que tenga la función.

Una vez que se configuró el periférico del dispositivo y se le da la orden de arranque, un bit bandera del registro de control avisa que ya está configurada esta función, para que la siguiente vez que el despachador le pase el control, ya no ingrese a esta parte de la estructura y pasará a la verificación.

En esta sección del código es donde se aplica el “polling”, ya que se verifica el registro donde avise que la condición ya se ha cumplido. Si esto sucede el código se dirige a la sección de “terminar” para poner a 1 el bit de bandera del registro de control avisando que la función ha terminado, además de incrementar la variable N que utiliza la sentencia de control Swicht-Case y liberar al periférico que utilizó la función. Si la condición aún no se cumple el flujo sale de esta rutina para pasar el control al despachador. En el siguiente diagrama de flujo de la figura 4.20 se puede observar cómo trabaja la estructura de las funciones.

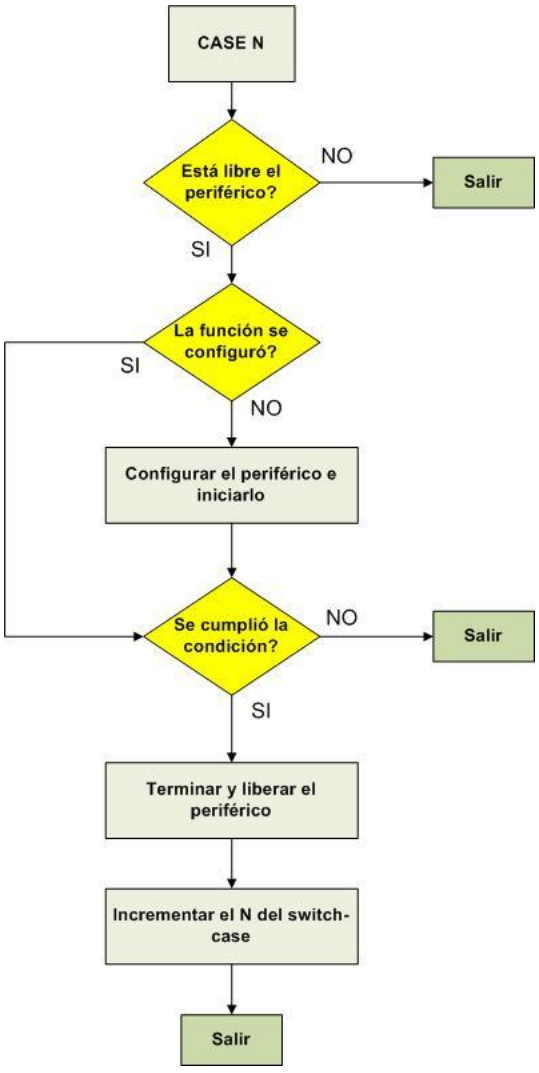

Figura 4.20 *Diagrama de flujo de la estructura de las funciones.*

El módulo de control contiene los conectores RJ45 donde los diferentes sensores son conectados, en el diagrama de la figura 4.21 se puede observar cómo están las líneas de conexión de éstos conectores con el microcontrolador.

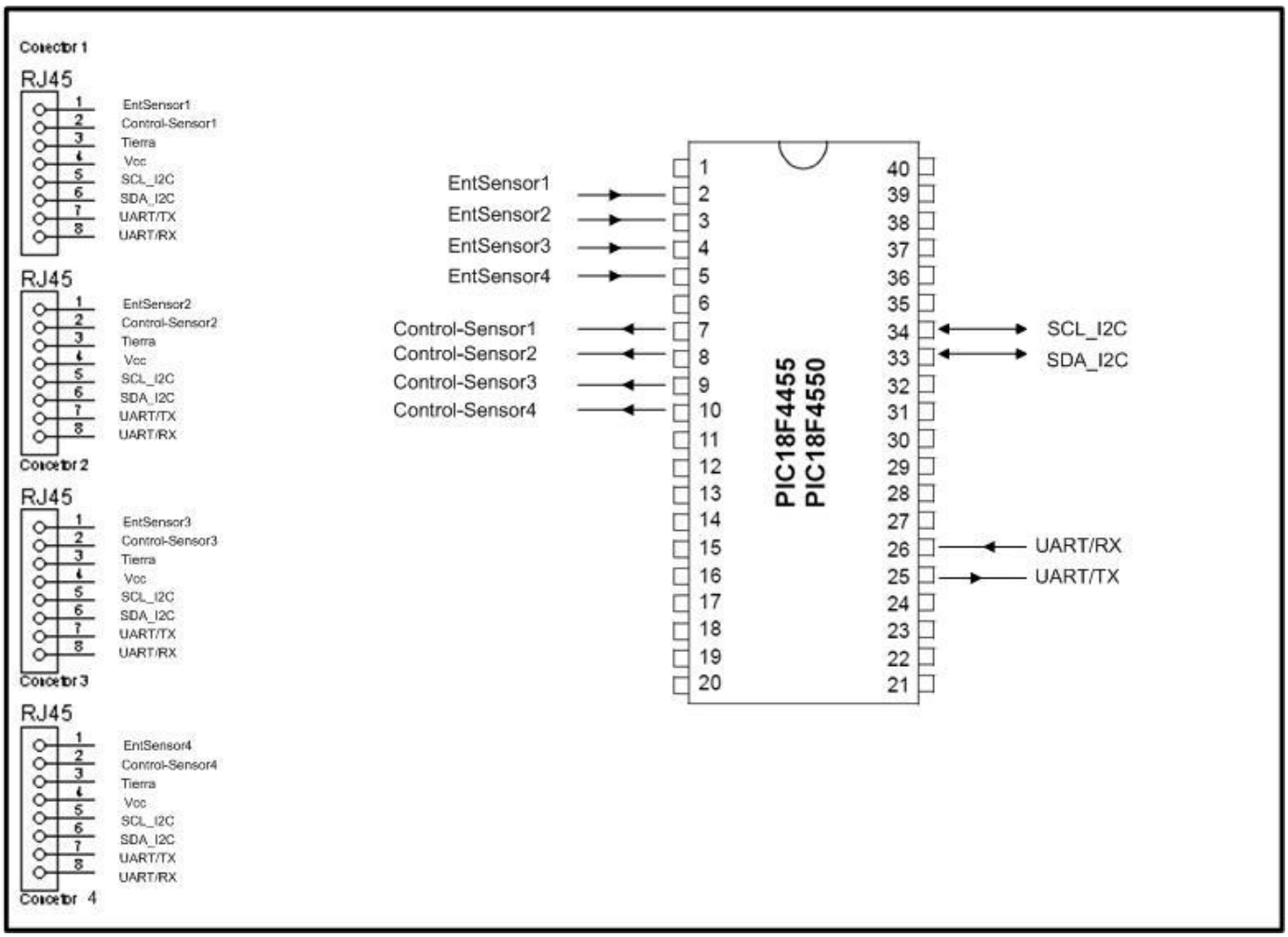

Figura 4.21 *Módulo de control con los conectores RJ45 para los sensores.*

Cada entrada de un sensor N es una entrada analógica al convertidor analógico digital del microcontrolador y las salidas de control del sensor N son pines en modo digital configuradas como salidas. Cada una de las señales de la comunicación I2C, pines 33 y 34 del microcontrolador va a las terminales respectivas de cada conector RJ45, terminal 5 y 6. Lo mismo sucede con la terminales del periférico UART, pines 25 y 26 del microcontrolador, éstas van a las terminales 7 y 8 de cada conector RJ45

El estudiante o usuario puede diseñar y construir sensores adicionales o motores que requieran su estructura, utilizando el protocolo I2C o el convertidor analógico digital. Estos sensores tendrían que cumplir con ciertas características para poder interactuar con el módulo de control.

Las características que se deben tomar en cuenta para que el usuario pueda construir más sensores son las siguientes:

- Que el circuito trabaje con 5 volts.
- La señal a entregar al convertidor analógico digital debe estar entre 1 volt y 4 volts.
- La máxima corriente que maneja cada pin de entrada o salida es de 25 ma.
- Los sensores que utilicen la comunicación I2C, deben tener sus resistencias pull-up.
- Mantener la configuración de las terminales con los conectores RJ45. Figura 4.22.

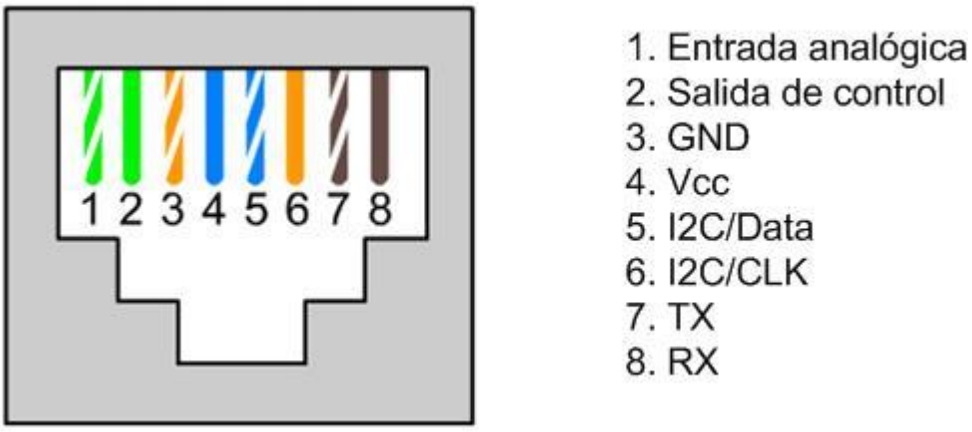

Figura 4.22 *Configuración de las terminales de los conectores RJ45 de los sensores.*

También el usuario tendría que diseñar el código de control del sensor diseñado y agregarlo a las funciones para que el módulo de control pueda hacer uso de él.

4.4 Módulo de salida.

4.4.1 Motores.

En este proyecto el módulo de salida está formado por dos motores y se utilizaron motores reductores, figura 4.23, adaptándoles un sensor de luz y modificando un engrane para obtener pulsos de acuerdo a las vueltas que dé el motor, este sistema es conocido como “encoder”. También se incluye en el módulo de salida un Display LCD, en el cual se enviará alguna información ya sea que pueda obtenerse de algún sensor o sea un texto que indique el usuario.

Figura 4.23 *Motor con reducción de plástico.*

Para poder controlar el motor se tomó en cuenta el giro del engrane conductor ya que este es el que transmite el movimiento hacia el exterior del motor, y se tiene que controlar cada grado que gire. Para poder obtener ese control se analizó el movimiento del primer engrane reductor del tren de engranaje, figura 4.24.

Éste tiene una relación de 36/8, y como el engrane eje motor es de 8 dientes, éste tiene que dar 4.5 vueltas para que el primer engrane reductor de 1 vuelta (36/8=4.5). Ahora si la relación total es 1:180, el primer engrane reductor tiene que dar 40 vueltas para que sea 1 vuelta del engrane conductor (180/4.5=40).

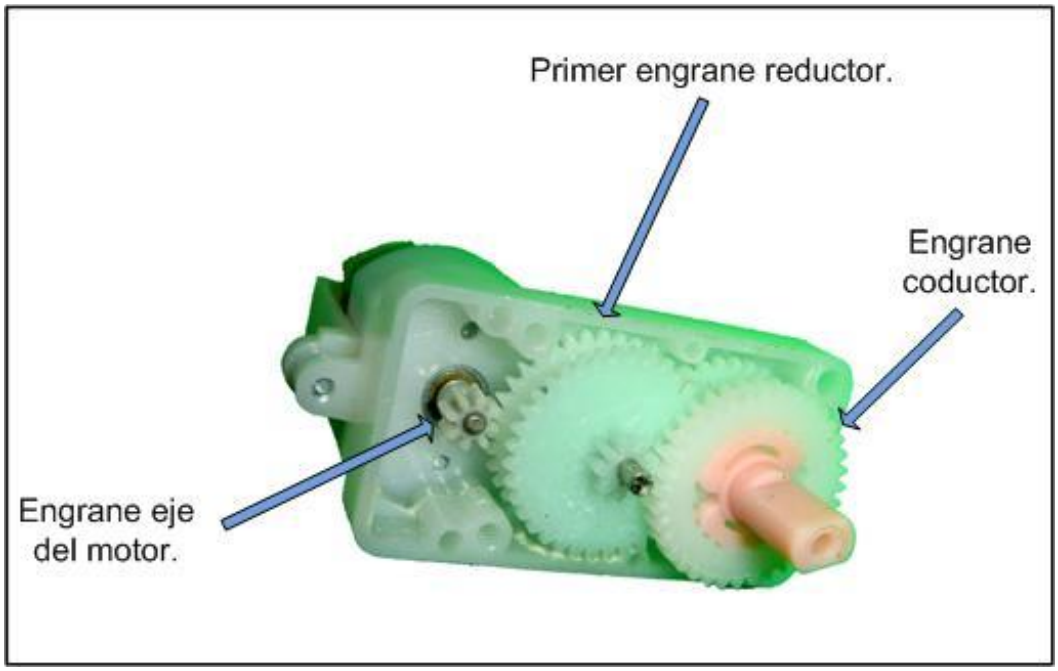

Figura 4.24 *Motor con engranes reductores sin modificar.*

Ya obtenida la relación del primer engrane reductor con el último engrane que es el conductor, ahora para controlar cada grado de giro del engrane conductor tenemos que dividirlo en 360 partes iguales o sea 1 grado, así que se dividió 360/40=9.

La modificación al primer engrane reductor fue realizar 9 orificios en la parte plana de este engrane, todos a la misma distancia del centro y la misma distancia entre sus centros. Pintar de negro el engrane para no permitir el paso del haz de luz.

Los orificios van a permitir pasar un haz de luz emitido por un diodo que es parte del sensor interruptor de luz, así cada interrupción realizada por el giro del primer engrane reductor equivale a 1 grado de giro del engrane conductor (figura 4.25).

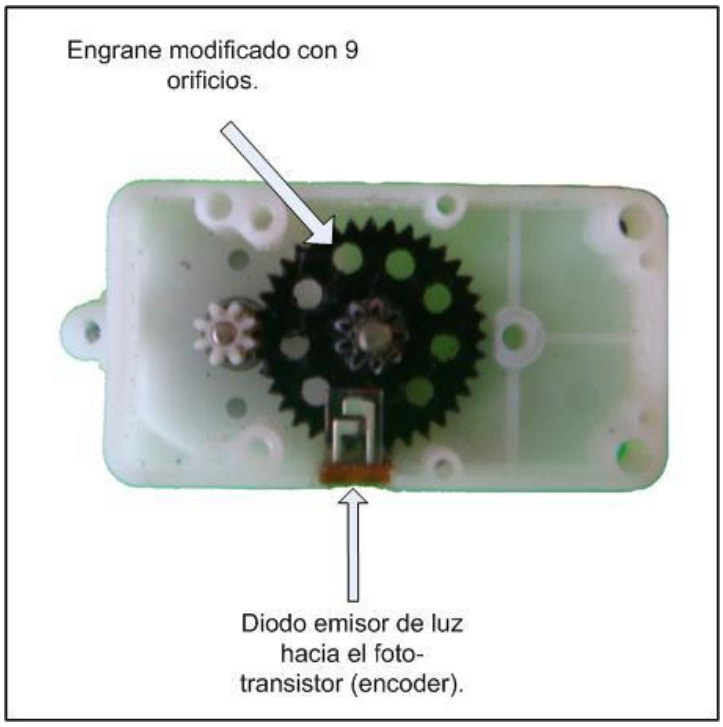

Figura 4.25 *Modificación al engrane de reducción del motor y adaptación del encoder.*

Con esto se tiene el control para que el microcontrolador pueda hacer girar el motor los grados que sean requeridos por el usuario, ya que el microcontrolador recibe los pulsos que son obtenidos por el sensor interruptor de luz.

En la figura 4.26 se puede observar el motor con las modificaciones y adaptaciones realizadas, así como con el conector RJ45 con el cual se conecta con el bloque de control.

En la figura 4.27 se pude observar el motor ya incrustado en su caja plastica.

La figura 4.28 muestra el motor terminado y lista para adaptarlo a las estructuras que se requieran armar.

En la figura 4.29 se puede observar el circuito electrónico que controla los motores DC, y que en el diagrama de bloques, figura 4.1, está identificado con el nombre de “bloque de control de los motores”.

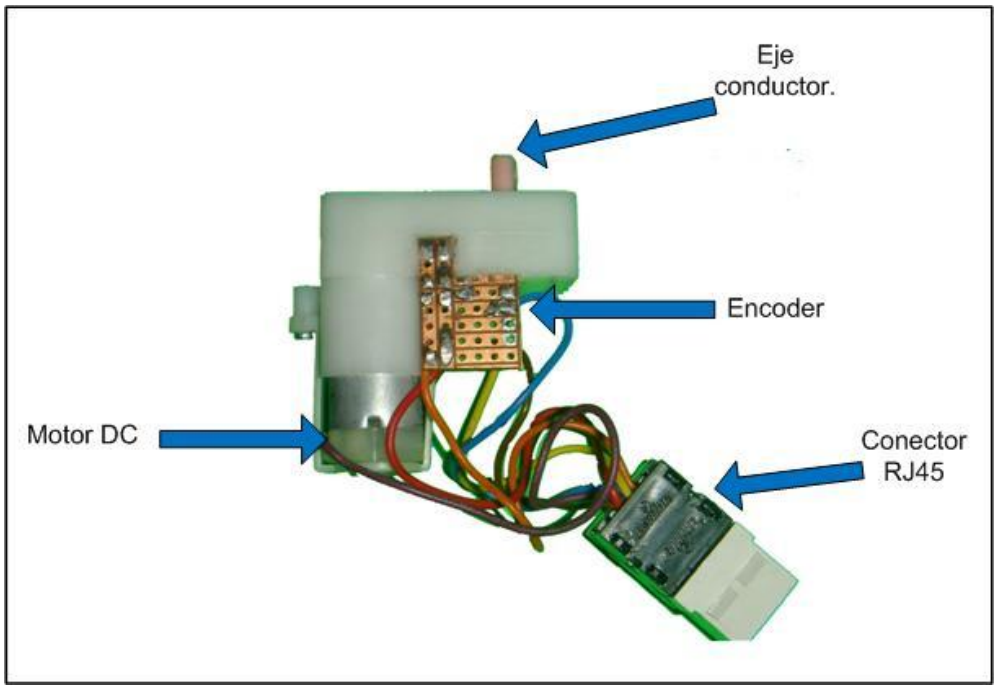

Figura 4.26 *Motor con adaptación y conectores.*

Figura 4.27 *Motor con caja plástica.*

Figura 4.28 *Motor terminado.*

En este bloque se describe el funcionamiento de un motor y el segundo motor trabaja de la misma forma. El microcontrolador genera una señal modulada por ancho de pulso que en este caso tiene su salida en el pin 17, esta salida es la del PWM1 y aplicamos el pulso a un inversor CD4049, y su salida es conectada a dos opto- acopladores. Los opto-acopladores tienen el objetivo de transmitir la señal del PWM al puente “H” L293D y de aislarlo del microcontrolador; cada uno de los opto acopladores alimenta a una de las entradas lógicas del L293D que controlan al motor 1. De acuerdo a la tabla 4.1 de control del chip L293D, dejamos una entrada lógica en “0” y la otra en estado lógico “1” para que el motor gire en un sentido izquierda y si invertimos sus estados girará en sentido contrario.

Tabla 4.1 Control del L293D.

V_{inh}	A	B	M
H	L	L	Parada rápida del motor
H	H	H	Parada rápida del motor
H	L	H	Giro a la Izquierda
H	H	L	Giro a la derecha
L	X	X	Motor desconectado, giro libre

El control de giro se realiza por medio de los opto-acopladores mediante las salidas en los pines 20 y 21 obteniendo los estados lógicos que se aplicarán al puente H. Cuando se aplica un estado lógico “1” por medio del pin 16 y 17 del microcontrolador, éste es aplicado con el ancho de pulso que trabaja el PWM del microcontrolador, de este modo se tiene control de la velocidad del motor.

El código para que el módulo de control haga uso de los motores se encuentra en el anexo de este documento.

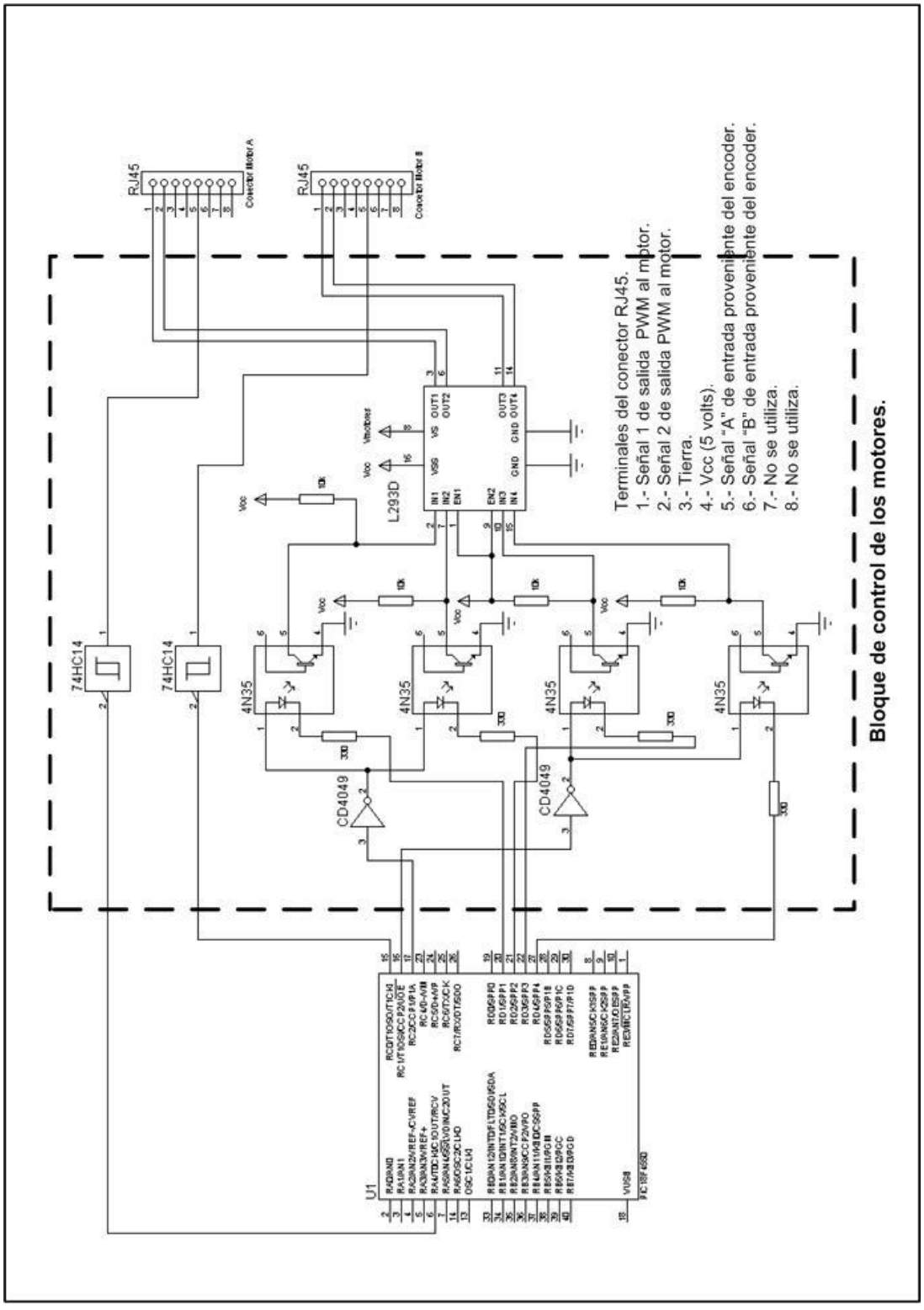

Figura 4.29 *Diagrama del bloque de control de los motores.*

4.4.2 Display LCD.

El modo de trabajo en la interfaz fue en la modalidad de 4 bits, y con las líneas de control se ocuparon en total 7 pines del microcontrolador como se ve en el siguiente diagrama de la figura 4.30.

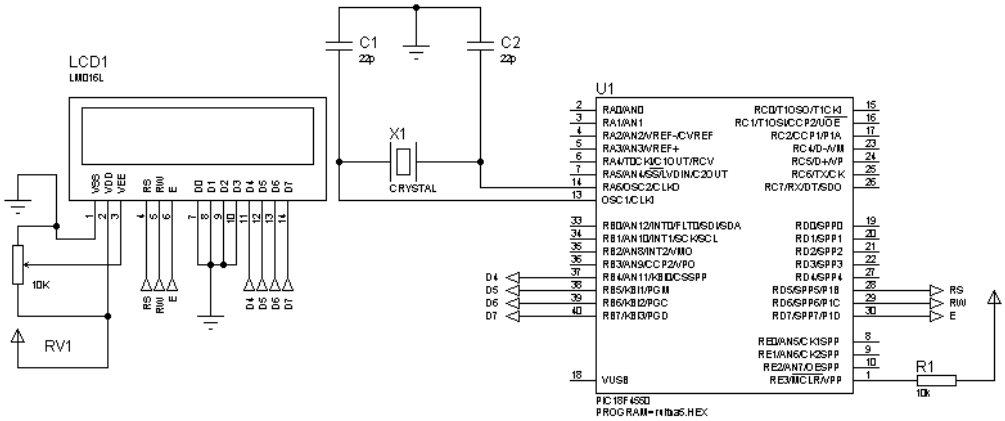

Figura 4.30 *Circuito LCD*

Resumen.

En este capítulo se describió el desarrollo de cada uno de los elementos de los módulos del sistema de desarrollo y se explicaron sus características principales mostrando sus diagramas electrónicos y las fotos de los sensores ya terminados. En el siguiente capítulo se mostrarán los resultados de las pruebas que se realizaron a cada elemento y el resultado que se obtuvo al armar algunas estructuras con los elementos presentados en este capítulo.

CAPÍTULO 5

Pruebas y resultados de los componentes del sistema de desarrollo.

En este capítulo se expondrán los resultados obtenidos de las pruebas que se realizaron.

Para el módulo de entrada primeramente se realizaron las pruebas por separado a cada uno de los sensores que se diseñaron, ajustando y modificando los circuitos hasta que se obtuvieran los parámetros deseados, posteriormente cada sensor se probó individualmente junto con el bloque de control para verificar que la función que utiliza el microcontrolador trabajara de acuerdo a lo planeado y también se realizaron modificaciones al código de la función.

Para el módulo de salida se pusieron a trabajar los motores individualmente, y con esto observar el tren de pulsos que generaban los encoder, también éstos se probaron junto con el bloque de control para establecer el buen funcionamiento del código de la función que los controlaba.

Para verificar el funcionamiento del módulo de proceso se tenía que armar una estructura con la cual se pusieran en funcionamiento los diferentes sensores y actuadores y de esta forma corregir el trabajo del despachador de tareas.

5.1 Pruebas y resultados del sensor de sonido.

El diseño del sensor de sonido mostrado en la figura 5.1 entregó las siguientes señales en su proceso de prueba.

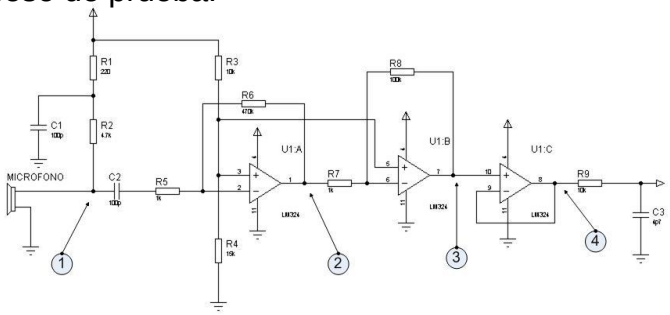

Figura 5.1 *Puntos de lectura de la señal.*

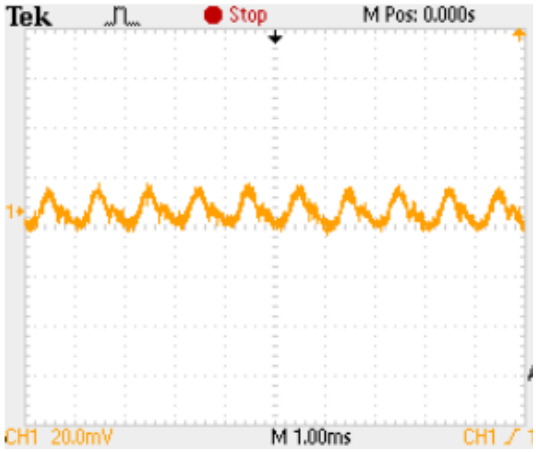

Figura 5.2 *Punto número 1, señal captada por el micrófono y aplicada al primer paso de amplificación.*

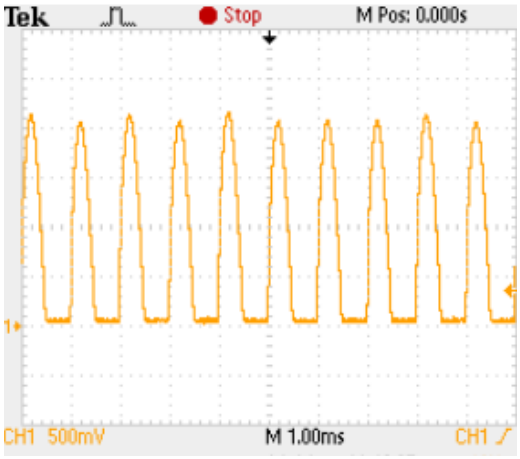

Figura 5.3 *Punto número 2, señal tomada de la salida del primer amplificador operacional.*

Figura 5.4 *Punto número 3, salida del segundo amplificador operacional.*

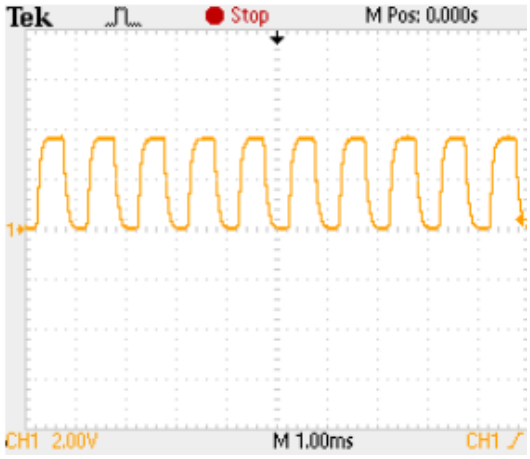

Figura 5.5 Punto número 4, salida del sensor de sonido, lectura que tomará el convertidor analógico digital del microcontrolador.

Como se puede observar en las imágenes de las figuras 5.13, 5.14, 5.15 y 5.16 tomadas del osciloscopio la señal que es captada por el micrófono es amplificada en dos etapas y entrega una señal para ser procesada por el convertidor analógico del microcontrolador.

5.2 Pruebas y resultados del sensor de distancia.

En el circuito del sensor ultrasónico de distancia que se muestra en la figura 4.7, se obtuvieron las señales que se muestran a continuación: En los pines 3 y 2 del microcontrolador que son RA4 y RA5 que están configurados como salida, se muestra la señal que se aplica al transductor transmisor para enviar la ráfaga de pulsos, figura 5.6 y figura 5.7.

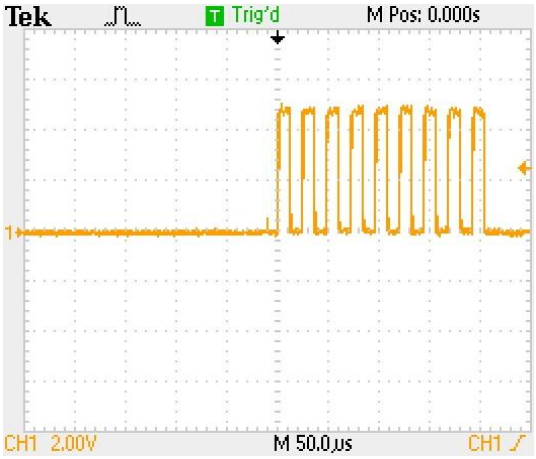

Figura 5.6 *Señal en el puerto RA4 del microcontrolador.*

Figura 5.7 *Señal en el puerto RA5 del microcontrolador.*

Esta ráfaga de pulsos que envía el transductor ultrasónico es de aproximadamente 40Khz y al ser rebotada por algún objeto es captada por el transductor receptor y aplicada al amplificador operacional por su entrada inversora, esta señal captada se puede observar en la figura 5.8.

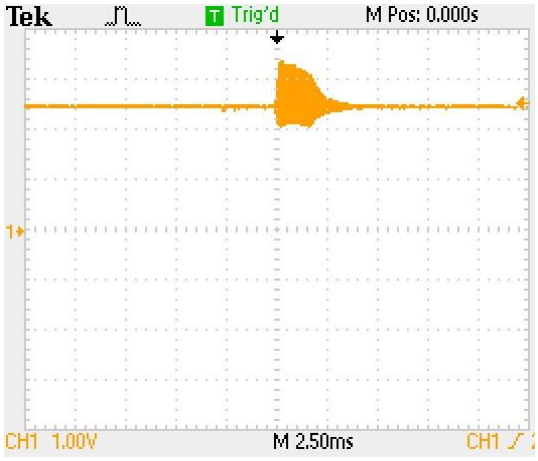

Figura 5.8 *Señal de eco recibida por el transductor del sensor ultrasónico.*

Esta señal es amplificada por dos etapas (figura 5.9) e introducida en un comparador que está dentro del microcontrolador y al rebasar cierto umbral se activará una bandera de uno de los registros del comparador.

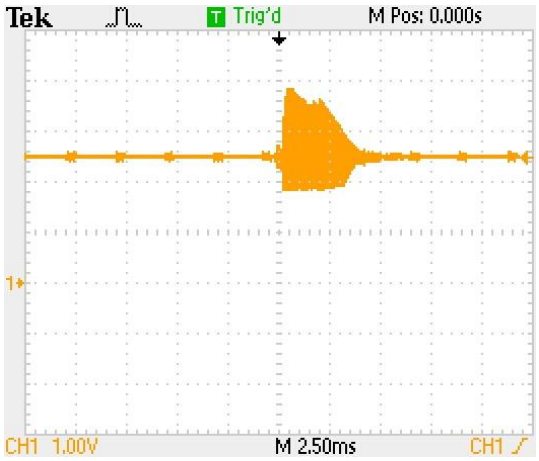

***Figura 5.9**. Eco amplificado y aplicado al microcontrolador.*

Al enviar la ráfaga de pulso, el microcontrolador pone en “1” la salida 5 del puerto C, (RC5), y cuando el eco es detectado por el microcontrolador RC5 es llevado a nivel “0” con lo cual se puede observar el tiempo que tardó en regresar la señal enviada, como lo muestra la figura 5.10 y la figura 5.11.

***Figura 5.10** Salida del puerto C, RC5, tiempo que tardo la señal enviada en regresar.*

Figura 5.11 *Un objeto detectado a menor distancia, por la señal de eco.*

El tiempo es tomado por el Timer1 del microcontrolador que es un timer de 2 bytes, y los registros respectivos son el TIMER1H y el TIMER1L, los cuales contendrán el tiempo en microsegundos que transcurrieron desde que se envió la ráfaga de pulsos hasta que fue detectado el eco por el comparador. Estos registros son los que son leídos y la información enviada cuando es solicitada por el bloque de control; esta comunicación se realiza utilizando el protocolo I2C, y en la figura 5.12 se puede observar la señal de reloj de ésta comunicación.

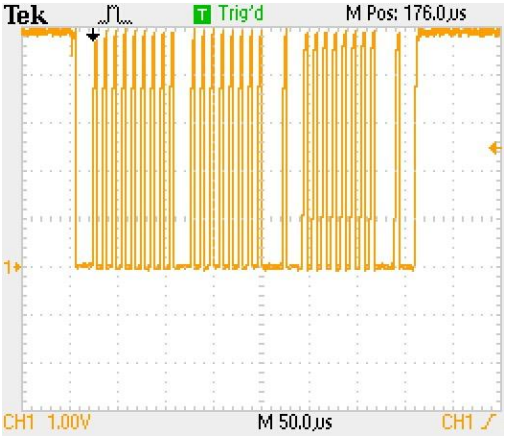

Figura 5.12 *Señal del reloj de la comunicación I2C entre el bloque de control y el sensor de distancia ultrasónico.*

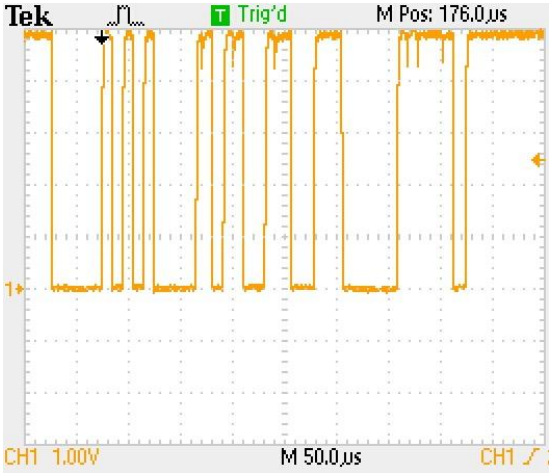

Figura 5.13 *Señal de datos transmitidos, dirección del esclavo, byte alto y byte bajo del timer1 que cuenta el tiempo del eco.*

En las figuras 5.13 se pueden observar la comunicación del bloque de control que envía en primer lugar la solicitud de comunicación con la dirección del esclavo solicitado, y posteriormente el sensor de distancia le envía los dos bytes que contienen la cantidad de microsegundos del objeto detectado.

Para realizar las pruebas del sensor de distancia el sensor fue colocado fijamente frente a un objeto móvil grande y plano de esta forma el eco de la ráfaga puede reflejarse en buena forma. También se utilizaron objetos de diferentes materiales como cartón, plástico, metal, una pared y madera. Tomando la distancia a la que se encontraba el sensor de los objetos y registrando el valor que entregaba el sensor, se obtuvieron los siguientes resultados en promedio, tabla 5.1.

5.3 Pruebas y resultados de los Motores.

Con base en el diagrama de la figura 4.23 se obtuvieron las siguientes señale que se pueden observar en las siguientes figuras.

En la figura 5.14, se puede observar la salida del PWM del microcontrolador trabajando a un 50% aproximadamente, señal que es entregada al inversor que en el diagrama de la figura 4.23 se identifica con el número CD4049.

Tabla 5.1 Lectura del tiempo obtenido por el sensor ultrasónico.

Distancia del objeto en cm.	Lectura en microsegundos	Distancia calculada cm.
20	1063	18.33
21	1173	20.22
22	1203	20.74
23	1233	21.26
24	1343	23.16
25	1420	24.48
30	1658	28.59
35	1978	34.10
40	2263	39.02
45	2573	44.36
50	2863	49.36
70	4048	69.79
90	5193	89.53
100	5766	99.41

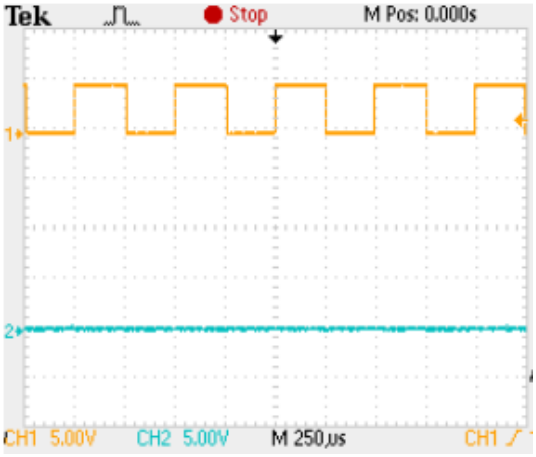

Figura 5.14 *Señal del PWM1 del microcontrolador.*

La señal obtenida del encoder es aplicada a un inversor Schmitt trigger para obtener un pulso cuadrado el cual es tomado por el microcontrolador para llevar la cuenta de los mismos, en la siguiente figura 5.15 se observa el pulso obtenido del encoder y también la salida del Schmitt tigger.

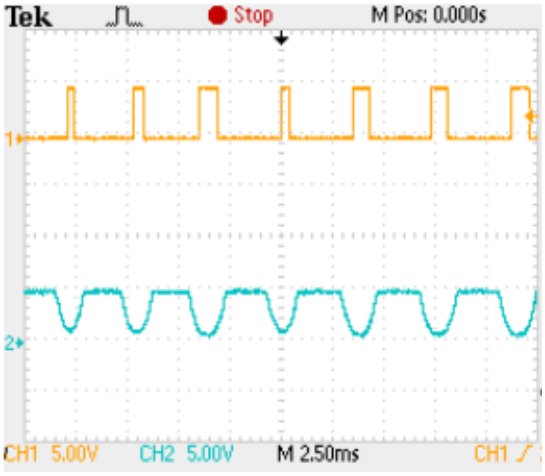

Figura 5.15 *Canal 2 salida del encoder y canal 1 salida del Schmitt tigger.*

Una vez probados los elementos de cada módulo de forma individual, se procedió a armar una estructura, en la cual involucrara a varios elementos de cada módulo, se programó una serie de acciones utilizando la programación en paralelo, posteriormente se ejecutó el programa y se obtuvieron las acciones programadas, en la siguiente figura 5.16 se puede ver la estructura armada.

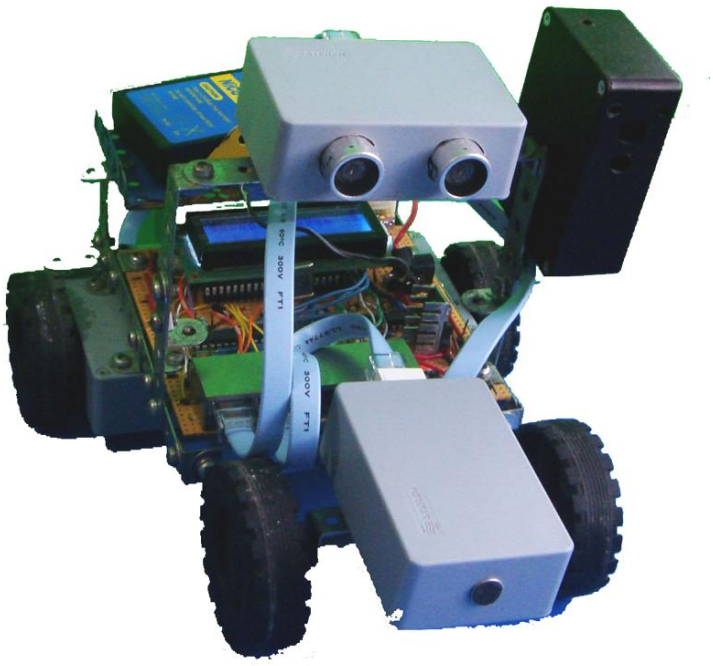

Figura 5.16 *Carrito con sensores, actuadores y bloque de control.*

CAPÍTULO 6

Conclusiones y trabajos futuros.

Conclusiones.

De acuerdo a los objetivos específicos que se plantearon en el presente proyecto se concluyen los siguientes puntos:

Para diseñar e implementar el módulo de entrada o de percepción se construyó:

a) Un sensor de sonido con el cual se capta el ruido ambiental y entrega al módulo de control una señal para su procesamiento. Se puede ajustar el nivel del umbral que el usuario necesite.
b) Un sensor con el cual se capta la intensidad de la luz y envía la información al bloque de control.
c) Un sensor de distancia que utiliza un transductor ultrasónico, y puede medir la distancia a la cual se encuentra un objeto enfrente del sensor, desde 20 cm hasta 1 metro. La comunicación con el bloque de control es mediante el protocolo I2C, y este sensor utiliza un microcontrolador 16F690 para realizar su trabajo.
d) Un sensor de toque que está diseñado con un botón de toque y éste envía un pulso eléctrico como información al bloque de control.

En el caso del módulo de proceso se construyó con base en un microcontrolador de la marca Microchip, el 18F4550 que pertenece a la familia de la gama media. Para poder cargar el programa que el usuario haya diseñado con el software correspondiente, este microcontrolador tiene los periféricos necesarios para la comunicación USB y la comunicación serie asíncrona con los cuales se puede comunicar con la PC. En el microcontrolador se graba un firmware conocido como cargador para que no sea necesario emplear un programador especial para los pic.

El bloque es capaz de ejecutar una programación en paralelo mediante el programa despachador de tareas que se diseñó.

Para conectar los diferentes sensores y los motores con módulo de control se utiliza cable tipo UTP y conectores tipo RJ45.

El módulo de salida que se diseñó está compuesto por dos motores y un display LCD. Los motores utilizados fueron moto-reductores de plástico los cuales fueron modificados para acondicionarles los encoder y de esta forma obtener el control de ellos. El display LCD que se utilizó es uno de tipo comercial con un microcontrolador tipo Hitachi HD44780.

Este proyecto complementó a la tesis de Diseño y desarrollo de un compilador visual para la enseñanza de la robótica básica, con el cual se programa el robot educativo aquí construido.

Trabajos futuros.

Para un futuro se pretende subir a la red de internet, toda la información necesaria y libre para que los estudiantes y usuarios puedan construir sus propios prototipos.

También en ese mismo sitio se establecerá un foro para resolver dudas de los participantes así como recibir propuestas de los mismos e ir incrementando el desarrollo del robot educativo.

Trabajar en los diferentes tipos de comunicación entre el robot y la PC, como son por medio de la interfaz Wifi, Bluetooth, Internet.

Desarrollar más motores los cuales se podrían comunicar con el bloque de control por medio de la protocolo I2C ya que el módulo de control ya la utiliza.

Referencias Bibliográficas.

Aguilar, M. & Melchor, V. (2004). Nociones acerca del constructivismo. Monografias.com [On-line]. Available: http://www.monografias.com/trabajos15/constructivismo/constructivismo.shtml

Alers, S. & Hu, J. (2009). AdMoVeo: A Robotic Platform for Teaching Creative Programming to Designers. In (pp. 410-421). Springer.

Ausubel, D. (2000). *Adquisición y retención del conocimiento*. (1 ed.) España: Paidos Iberica.

BBC, N. (2001). Timeline: Real robots. BBC News [On-line]. Available: http://news.bbc.co.uk/1/hi/in_depth/sci_tech/2001/artificial_intelligence/1531432.stm

Bruner, J. (1966). *Toward a theory of instruction*. (1 ed.) Massachusetts: Harvard Unersity Press.

Coll, C., Martin, E., Mauri, T., Miras, M., Onrubia, J., Solé, I. et al. (2007). *El constuctivismo en el aula*. (18 ed.) Barcelona: GRAO de IRIF. S.L.

Delgado, G. (2005). *El mundo moderno y contemporaneo I*. (5 ed.) México: Pearson.

Delval, J. (1996). La fecundidad de la epistemologia de Piaget. *Stratum, 3,* 89-125.

Dominguez, O. and García, H. (2007). Ausubel, Piaget y Vygotsky. from http://www.monografias.com/trabajos43/piaget-ausubel-vygotsky/piaget-ausubel-vygotsky2.shtml

Falbel, A. (1993). Constructionism: Tools to build (and think) with. *LEGO DACTA*.

Florez, R. (1994). *Hacia una pededagogía del conocimiento*. (1 ed.) Sta. Fe de Bogotá: McGraw-Hill.

Fu, K. S., Gonzalez, R. C., & Lee, C. S. G. (1988). *Robótica: Control, detección, visión e inteligencia*. (1 ed.) Madrid: McGraw-Hill.

Garza, R. M. & Levanthal, S. (1998). *Aprender como aprender*. (1 ed.) México D.F.: Trillas.

Iñigo, R. & Vidal, E. (2002). *Robots industriales manipuladores*. (1 ed.) Barcelona: Ediciones UPC.

Jiménez, J. A., Ovalle, D. A., & Ochoa, J. F. (2008). Propuesta de una plataforma para la difusión de la robótica móvil: E-SMART. *Revista Avances en Sistemas e Informática, 5,* 207-212.

Maldonado, M A. (2006). El aprendizaje significativo de David Paul Ausubel. from http://www.monografias.com/trabajos10/dapa/dapa.shtml

Mindell, D., Beland, C., Chan, W., Clarke, D., Park, R., & Trupiano, M. (2000). *LEGO Mindstorms: The structure of an Engineering Revolution*. (1 ed.) Massachusetts: MIT.

MInsky, M. (1988). *La sociedad de la mente*. New York: Touchstone.

Papert, S. (1980). *Mindstorms: children, computers, and powerful ideas*. (2 ed.) New York: Basic Books.

Papert, S. (1987). Microworlds: transforming education. In R.W.Lawer & M. Yazdani (Eds.), *Artificial intelligence and education; vol. 1: learning environments and tutoring systems* (pp. 79-94). Norwood: ACM.

Papert, S. (1999). INTRODUCCION: What is Logo? Who Needs It? In *Logo Philosophy and implementation* (1a ed., pp. V-XVI). Montreal: Logo Computer Systems Co.

Resnick, M. (2007). Sowing the seeds for a more a creative society. *MIT Media Lab*.

Resnick, M. & Silverman, B. (2005). Some Reflections on Designing Construction Kits for Kids. In *Interaction Design and Children 2005* (pp. 117-122). ACM Press.

Robot Terms and Definitions (2010). RIA: Robotic Industries Asocciation [On-line]. Available: http://www.robotics.org/product-catalog-detail.cfm?productid=2953

Ruiz-Velasco, E. (2007). *Educatronica*. (1 ed.) (vols. 1) Madrid: Diaz de Santos.

Sánchez, FM., Millán, F., Salvador, J., Palou, J., Rodríguez, F., Esquena, S. et al. (2007). Historia de la robótica: de Arquitas de Tarento al robot Da Vinci ⁀Parte I⁀. In *ACTAS UROLÓGICAS ESPAÑOLAS FEBRERO 2007* (pp. 69-76). Barcelona: Fundació Puigvert.

Sánchez, FM., Millán, F., Salvador, J., Palou, J., Rodríguez, F., Esquena, S. et al. (2007). Historia de la robótica: de Arquitas de Tarento al robot Da Vinci ⁀Parte II⁀. In (pp. 185-196). Barcelona: Fundació Puigvert.

Vasquez, S. (2008). Un poco de teoría y marco para el diseño. *Aprendiendo en la práctica,* 36-39.

Villar, F. (2003). *Psicología Evolutiva y Psicología de la Educación*.

Vivet, M. (1989). Robotique pedagogique.soit, mais pour appredre quoi". In Le Mans.

Zhang, H., Zheng, W., Chen, S., & Zhang, J. (2007). Flexible Educational Robotic System for a Practical Course. In *International Conference on Integration Technology* (pp. 691-696). IEEE.

Zuckerman, O. & Resnick, M. (2003). System Blocks: A Physical Interface for System Dynamics Learning . *MIT Media Lab*.

Anexo 1.

Teorías del aprendizaje

“Una teoría del aprendizaje es el conjunto de ideas que tratan de explicar lo que es el conocimiento y cómo éste se desarrolla en la mente de las personas” **(**Arón Falbel, 1993).

Algunas de las teorías en las cuales se basa el grupo de epistemología y aprendizaje del MIT, son la teoría genética que es parte del constructivismo del investigador Jean Piaget (1896-1980) y del construccionismo del científico Symour Papert (1928-), este último fue el creador del grupo, y trabajó muchos años con Jean Piaget en la Universidad de Ginebra en Suecia.

Constructivismo.

El constructivismo es una explicación científica acerca del conocimiento que es referido frecuentemente como una nueva técnica en los sistemas educativos de México y en diferentes partes del mundo. (Melchor Aguilar, 2004)

Los planteamientos más difundidos con relación al constructivismo son los de quienes toman como base las aportaciones de Piaget, Vygotsky, Bruner y Ausubel (Garza, 2000; Coll, 1993; Flórez, 1994, pp. 234-253.).

El autor Coll afirma que se pueden distinguir cuatro tipos de constructivismo; el que plantea Piaget en su teoría genética, el de las teorías del aprendizaje verbal significativo de los organizadores previos y de la asimilación que Ausubel propuso, el de la teoría de la psicología cognitiva de Bruner y el de Vygotsky con su teoría sociocultural.

Estas teorías forman la estructura del constructivismo ya que involucra la actividad cotidiana, el contexto en el cual se desarrolla esa actividad y el lado interno de la persona.

Las explicaciones de cómo se construye el conocimiento son de Piaget y de Vygotsky, mientras que Ausubel lo hace sobre la construcción del aprendizaje.

El Modelo Constructivista se centra en la persona, en sus experiencias previas de las que realiza nuevas construcciones mentales, considera que la construcción se produce:

a. Cuando el sujeto interactúa con el objeto del conocimiento (Teoría genética, Piaget).

b. Cuando esto lo realiza en interacción con otros (Teoría sociocultural, Vygotsky).

c. Cuando es significativo para el sujeto (Teoría del aprendizaje significativo, Ausubel).

d. Aprendizaje por descubrimiento (Teoría de la psicología cognitiva, Bruner).

Jean Piaget (1896-1980)

La teoría de Piaget trata del desarrollo cognitivo, que busca explicar cómo los individuos adquieren el conocimiento a través de diferentes etapas de su vida y distingue cuatro niveles que el autor Sergio Vázquez Bronfman (2008) nos describe de la siguiente forma:

"*1.- El nivel sensor-motor (desde el nacimiento a más o menos los 2 años), donde los niños exploran el mundo que les rodea a través de sus sentidos y su actividad motriz.*

2.- El nivel pre-operativo (desde los 2 a los 7 años), donde se desarrollan las habilidades de comunicación a través del lenguaje, las habilidades simbólicas (dibujar, imaginar, fingir), y las habilidades de numeración. Pero en este nivel de desarrollo el niño es aún incapaz de hacer abstracciones y necesita una concreción física de las cosas para entenderlas bien.

3.- El nivel operativo concreto (desde los 7 a los 11 años), donde se desarrollan las habilidades de abstracción a partir de experiencias concretas.

4.- Nivel de operaciones formales (a partir de los 12-15 años), donde se desarrollan habilidades como elaborar y contrastar hipótesis, organizar la información, etc."

Se puede concluir en base a estos niveles que las personas aprenden por naturaleza de lo concreto a lo abstracto y si observamos el sistema de enseñanza tradicional utiliza la técnica contraria.

En la teoría que Piaget desarrolló afirma que el ser humano construye su conocimiento. Es decir, que las personas construyen un sistema sólido de creencias a partir de su interacción con el mundo. Por ello, puede clasificarse a su teoría como constructivista **(**Arón Falbel, 1993).

El punto central es que el niño va construyendo su conocimiento y no que ese conocimiento se lo de el profesor. El niño obtiene un aprendizaje que es la resultante de una elaboración continua de nuevas estructuras mentales que tienen por objetivo entender nuevamente su entorno para dar sentido a su experiencia cotidiana. Piaget menciona que "comprender es inventar", "de donde se deduce que el papel del profesor es principalmente crear las condiciones para inventar en vez de limitarse a entregar conocimiento empaquetado y listo para utilizar." (Sergio Vázquez Bronfman, 2008) .

Para Piaget, el conocimiento esta unido a la acción, a las operaciones, es decir a las transformaciones que el sujeto realiza sobre el mundo que le rodea (Delval, 1996).

De esta forma, Piaget recurre al estudio del niño no como fin, sino como medio para dar una respuesta empírica a sus inquietudes epistemológicas. El estudio del niño será para él un instrumento. Un auxiliar imprescindible para dar cuenta del pensamiento adulto. Este es el matiz que diferencia a la psicología del niño de su

propia aproximación a la infancia, que denomina “psicología Genética”: (Feliciano Villar, 2003).

“*Si la psicología del niño estudia a este por él mismo, se tiende hoy, por el contrario. A denominar ‘psicología genética’ a la psicología general (estudio de la inteligencia, de las percepciones, etc.), pero en tanto que trata de explicar las funciones mentales por su modo de formación, o sea, por su desarrollo en el niño(…) la psicología infantil se ve promovida a ‘psicología genética’, lo cual equivale a decir que se convierte en instrumento esencial de análisis explicativo para resolver los problemas de la psicología general* (Piaget, 1969, p. 12-13 de la trad. Cast.).

La idea de Piaget es que el niño trata de asimilar el mundo que lo rodea dándole un significado para él, esto es, trata de construir su conocimiento acerca de lo que lo rodea y de el mismo, esto lo logra mediante la obtención de información de su entorno o el objeto e interactuando con él y así el niño construye poco a poco una comprensión tanto de sus propias acciones como del mundo externo. Para que se logre este conocimiento la acción del sujeto sobre su entorno es fundamental. Para conocer los objetos tiene que actuar sobre ellos y transformarlos: desplazarlos, agarrarlos, conectarlos. Combinarlos, separarlos, unirlos, etc.

Vygotsky, Lev Semiónovich (1896-1934)

Vygotsky afirma en su teoría que a la vez que reconoce la naturaleza constructivista del aprendizaje, también afirma el carácter social de ésta, dando nacimiento a la corriente pedagógica llamada *constructivismo social*. Vygotsky decía que lo que los niños aprenden se adquiere en forma directa de la cultura de su entorno. La cultura en la cual se desarrolla el niño les enseña cómo tienen que pensar, y además cómo pensarlo, y toda esta información les es transmitida por los adultos a través del lenguaje es sus diferentes formas. El desarrollo cognitivo del niño sucede cuando interactúa con otros niños y con personas adultas. El niño aprende realizando las tareas por sí mismo, y muchas veces imitando a los adultos. Sin olvidar que existe una diferencia entre lo que un niño puede hacer

por sí solo y lo que puede hacer ayudado por un adulto,… o de otros niños más avanzados que él. Esta diferencia es conceptualizada por Vygotsky como la *Zona de Desarrollo Próximo* (Sergio Vázquez Bronfman, 2008).

Este concepto nos dice que es muy importante la obtención del aprendizaje mediante la práctica lo que se conoce como “aprender-haciendo”, y el resultado que se obtenga va a ser mucho más exitoso si se recurre a la guía de una persona que domine el tema u objeto que se está aprendiendo.

Jerome Bruner

Jerome Bruner, psicólogo estadounidense, nació en Nueva York en 1915. Se graduó en la universidad de Duke en 1937. Después marchó a la universidad de Harvard, donde en 1941 consiguió su título de doctor en psicología. En 1960 fundó el Centro de Estudios Cognitivos de la Universidad de Harvard.

El psicólogo Jerome Bruner es máximo exponente de la teoría del aprendizaje por descubrimiento, en esta teoría la persona es la que tiene la mayor participación para realizar el aprendizaje ya que la información del objeto de estudio no es explicada en su totalidad por el profesor o instructor, sino que una de sus tareas es dar a conocer la meta que ha de ser alcanzada y mediante su guía y observación son las personas las que recorren el camino para alcanzar los objetivos trazados.

Por lo que se dice que el aprendizaje por descubrimiento es en el cual las herramientas que necesita la persona para realizar los descubrimientos de lo que desea aprender son dadas por el profesor.

El método por descubrimiento, permite al individuo desarrollar habilidades en la solución de problemas, ejercitar el pensamiento crítico, discriminar lo importante de lo que no lo es, preparándolo para enfrentar los problemas de la vida (Jerome Bruner, 1966).

Del mismo modo que Vygotsky, Bruner afirma que los niños aprenden de su propia cultura interactuando con ella, en particular con personas de esa cultura

que la conocen más. Pero, a partir de allí, Bruner deriva en dos nuevos conceptos: a) el aprendizaje por exploración, a través del descubrimiento, y b) la intersubjetividad, el aprendizaje en comunidades de pares.

En los modelos educativos que se han desarrollado a partir de las investigaciones de Bruner, el aprendizaje tiene su punto central en las interacciones entre profesores y estudiantes, y entre los estudiantes que interactúan. El entorno educativo en el cual se desarrollan coloca al estudiante en una confrontación de lo que piensa con las opiniones de los otros estudiantes, y eventualmente transformarlos a través de esa confrontación. La palabra no es la única forma de interacción pedagógica; al contrario, Bruner subraya la importancia de la creación colectiva de "obras" (dibujos, sitios Web, diarios murales, experimentos de ciencias naturales,…) durante la actividad educativa (Sergio Vázquez Bronfman, 2008).

David Ausubel

Ausubel aportó su teoría sobre el aprendizaje, cuando en los 70´s el aprendizaje por descubrimiento de Bruner estaba tomando auge, en esos momentos los colegios buscaban que los niños a través del descubrimiento de contenidos fueran construyendo su conocimiento. Ausubel proponía que el aprendizaje por exposición (recepción) no era tan malo ya que si se cumplían algunas características de éste podría ser igual de eficiente que el aprendizaje por descubrimiento.

Así, el aprendizaje escolar puede darse por recepción o por descubrimiento, como estrategia de enseñanza, y puede lograr un aprendizaje significativo o memorístico y repetitivo.

Y si tomamos de base al aprendizaje significativo, los conocimientos que se van aprendiendo se incorporan en la estructura cognitiva del alumno de manera importante. Esto se logra cuando el estudiante establece una relación entre los nuevos conocimientos con los adquiridos en el pasado; pero también es

importante que el alumno tenga el interés por aprender lo que se le está mostrando (María Alejandra Maldonado Valencia, 2006).

Así tenemos que la teoría de Ausubel acuña el concepto de "aprendizaje significativo" para que se pueda diferenciar del repetitivo o memorístico y señala la importancia que tienen los conocimientos previos del alumno en la captación de las nuevas informaciones. Lo significativo sólo se lleva acabo si se relacionan los nuevos conocimientos con los que ya se habían adquirido y estaban en la memoria del sujeto (Olivia Domínguez, 2007).

Ventajas del Aprendizaje Significativo:

- Produce una retención más duradera de la información.
- Facilita el adquirir nuevos conocimientos relacionados con los anteriormente adquiridos de forma significativa, ya que al estar claros en la estructura cognitiva se facilita la retención del nuevo contenido.
- La nueva información al ser relacionada con la anterior, es guardada en la memoria a largo plazo.
- Es activo, pues depende de la asimilación de las actividades de aprendizaje por parte del alumno.
- Es personal, ya que la significación de aprendizaje depende los recursos cognitivos del estudiante.

Requisitos para lograr el Aprendizaje Significativo:

1. Significado lógico del material: el material que presenta el maestro al estudiante debe estar organizado, para que se dé una construcción de conocimientos.
2. Significado psicológico del material: que el alumno conecte el nuevo conocimiento con los previos y que los comprenda. También debe poseer

una memoria de largo plazo, porque de lo contrario se le olvidará todo en poco tiempo.

3. Actitud favorable del alumno: ya que el aprendizaje no puede darse si el alumno no quiere. Este es un componente de disposiciones emocionales y de actitud, en donde el maestro sólo puede influir a través de la motivación.

(María Alejandra Maldonado Valencia, 2006).

Construccionismo.

El construccionismo es una teoría del aprendizaje propuesta por Symour Papert, él es considerado como un científico matemático, computacional y educador. El Dr. Ruiz nos menciona que “El construccionismo es una teoría del aprendizaje que posiciona al constructivismo en el centro del proceso de enseñanza/aprendizaje y es aplicable tanto a los procesos de aprendizaje de adultos como de niños. Revalora los estadios de Piaget y concluye que el pensamiento concreto de los niños no es inferior al pensamiento lógico formal de los adultos, sino es más bien un prerrequisito para activar el aprendizaje y la teoría de procesos de construcción del aprendizaje” (Enrique Ruiz Velasco Sánchez, 2007, p. 67).

Seymour Papert adaptó la palabra construccionismo, para referirse a todo lo que tiene que ver con hacer cosas y especialmente con aprender construyendo, una idea que incluye la de aprender haciendo, pero que va mas allá de ella, (Papert, 1999)

La teoría del construccionismo afirma que el aprendizaje se lleva de mejor manera cuando los niños se trabajan con gusto en la construcción de un producto significativo, tal como un castillo de arena, un poema, una máquina, un cuento, un programa o una canción. Así se realizan dos construcciones al mismo tiempo: cuando los niños construyen cosas en el mundo externo, simultáneamente construyen conocimiento al interior de sus mentes. Y pueden construir cosas mucho más complejas en el mundo externo con el conocimiento recién adquirido y a su vez va creando un conocimiento interno también más complejo y así

sucesivamente se forma un ciclo que hace que el conocimiento crezca en cada retroalimentación **(**Arón Falbel, 1993).

Piaget habla acerca de las construcciones mentales que el estudiante lleva a cabo en su cerebro. Papert dice que las construcciones físicas son un buen camino para alcanzar esas construcciones (Enrique Ruiz Velasco Sánchez, 2007, p. 67).

En los años 60´s Papert hablaba de que los niños podrían utilizar las computadoras como un medio de aprendizaje y para que su creatividad se desarrollara ampliamente, la gente de ese tiempo no lo tomaba en serio ya que el tener una computadora personal barata para cada niño en esos años era solo un sueño.

Seymour estaba entre los primeros que veían que era necesario un cambio masivo en el sistema educativo, particularmente en la educación de las matemáticas y la ciencia, reconocía el papel que la tecnología podría desempeñar en el aprendizaje. Él también era uno de los primeros en reconocer que la tecnología en los salones de clase no era una solución mágica que resolvería todos los males de la educación. El se dio cuenta mucho antes que el movimiento de la reforma educativa, que la tecnología en la educación solamente es eficaz si estaba colocada en un gran contexto que combinara a profesores bien preparados con servicios sociales integrados (Papert Seymour, 1980).

En los 70's Papert creó un lenguaje de computadora que llamó LOGO, era tan simple e intuitivo que hasta la fecha de hoy se utiliza sirviendo de base a nuevos programas o software para las herramientas lúdico tecnológicas que se han diseñado. El programa LOGO permite a los niños usar las matemáticas como material de construcción para crear diseños, animaciones, música, juegos y simulaciones en la computadora.

En 1980 Papert publicó un libro llamado "Mindstorm" en el cual detalla la invención de LOGO y las ideas filosóficas que influenciaron la construcción técnica del lenguaje, el nombre de su libro fue tomado para nombrar más adelante al producto "LEGO Mindstorm" de la empresa LEGO.

Papert fundó el grupo del Laboratorio de Medios del MIT junto con el Dr. Mitchel Resnick a mediados de los años 80's, su grupo trabajó en unión con la empresa LEGO para desarrollar juguetes que fueran una ayuda en los ambientes de aprendizaje, de esa fusión que por un lado tenia las teorías del aprendizaje, con el grupo de epistemología y aprendizaje del MIT, y por el otro una compañía que tenía ya muchos años trabajando en la fabricación de juguetes educativos como es LEGO, se obtuvieron productos que han demostrado cumplir el objetivo para el cual fueron creados.

"*El construccionismo nos recuerda que, lo mejor es fabricar algo tangible, algo fuera de nuestra mente, que también tenga significado para nosotros como personas"* (Seymour Papert).

Robótica pedagógica.

Introducción

El estudio e investigación en el campo de la educación junto con el desarrollo de la robótica ha generado una nueva disciplina conocida como robótica pedagógica o robótica educativa. Para definir esta disciplina recurrimos a la del investigador Matial Vivet (1989) del laboratorio de informática de la Universidad de Maine, quien fue de los primeros investigadores que trabajo con la aplicación de proyectos de robótica, con el objetivo de apoyar el proceso de aprendizaje en las aulas y su definición es la siguiente:

"… *una actividad de concepción, creación, puesta en práctica, con fines pedagógicos, de objetos físicos que son reducciones bastante fiables y significativas de procedimientos y herramientas robóticas realimente utilizadas en la vida cotidiana, particularmente en el medio industrial*."

En la definición anterior cuando se menciona los fines pedagógicos esta involucrando muchas de las teorías de aprendizaje que se habían expuesto anteriormente en la sección del mismo nombre.

Por lo que se establece que la robótica pedagógica o educativa tiene sus fundamentos en el constructivismo y el construccionismo, teniendo en cuenta todo el conjunto de conocimiento e investigaciones que se han llevado a cabo en estas disciplinas por grandes científicos mencionados anteriormente.

Uno de los objetivos que tiene la robótica pedagógica es explicado por el Dr. Ruiz Velasco:

"*Un objetivo tecnológico primordial de la robótica pedagógica es, mediante un uso pedagógico de la computadora, la generación de entornos tecnológicos ricos, que permitan a los estudiantes la integración de distintas áreas del conocimiento para la adquisición de habilidades generales y de nociones científicas, involucrándose en un proceso de resolución de problemas con el fin de desarrollar en ellos un pensamiento sistémico, estructurado, lógico y formal.*"

La robótica pedagógica también tiene sus bases en los principales conceptos de la epistemología y la psicología genética así como de otras teorías conceptuales por lo que pone en un lugar importante al aprendizaje inductivo y por descubrimiento guiado, con esto conforme se están diseñando y experimentando los problemas propuestos, se crean situaciones constructivistas que lograrán que el alumno construya su propio conocimiento.

En el proceso constructivista que maneja la robótica pedagógica, el concepto de error es muy importante ya que al proponer una solución a un problema, diseñarla e implementarla puede obtenerse un resultado ya sea favorable o que lleve a un error, pero en ambos casos el resultado será físicamente real, con lo que puede proponer diferentes soluciones a ese problema.

Se puede resumir muchas de las aportaciones que nos ofrece la robótica pedagógica mediante sus bondades cognoscitivas que propone el investigador Ruiz Velasco (2007):

a) Integración de distintas áreas del conocimiento.

b) Operación con objetos manipulables, favoreciendo el paso de lo concreto hacia lo abstracto.
c) Apropiación por parte de los estudiantes de distintos lenguajes (gráfico, icónico, matemático, natural, etcétera), como si se tratara de lenguaje matemático.
d) Operación y control de distintas variables de manera síncrona.
e) El desarrollo de un pensamiento sistémico y sistemático.
f) Construcción y prueba de sus propias estrategias de adquisición del conocimiento mediante una orientación pedagógica.
g) Creación de entornos de aprendizaje.
h) El aprendizaje del proceso científico y de la representación y modelización matemática.
i) Creación de un ambiente de aprendizaje lúdico y heurístico.

Los micromundos en la robótica pedagógica, un robot educativo.

Seymour Papert acuñó el término de micromundos, el fue el inventor del lenguaje de programación LOGO, que son programas de computadora por medio del cual se puede representar modelos del mundo real permitiendo que los estudiantes experimenten con esos modelos creados. Los micromundos motivan a los estudiantes a generar soluciones a ciertos problemas que le son propuestos y con la ventaja de que les proporcionan retroalimentación inmediata teniendo una variedad de conexiones dinámicas entre símbolo, gráfico y representación numérica. Con los micromundos se pueden diseñar simulaciones restringidas de fenómenos del mundo real que los estudiantes controlan y por medio de ese control pueden examinar el fenómeno. Los micromundos son quizá el más reciente ejemplo de ambiente de aprendizaje activo, en que los estudiantes pueden ejercer bastante control sobre sus creaciones (Papert, 1987).

Así lo define Sergio Vázquez Bronfman (2008): "*Un entorno de aprendizaje que tiene las reglas del mundo real, pero donde se puede experimentar sin riesgo*."

La robótica pedagógica utiliza estrategias de investigación-desarrollo para la creación de micromundos los cuales se llamaran “robot educativo” o “robot pedagógico”. Uno de los objetivos de los micromundos es que el estudiante pueda captar y modelar un fenómeno real, manipulando dispositivos con la ayuda del robot educativo. El desarrollo del robo educativo, ya sea en su construcción o su implementación es planificado y determinado por el estudiante partiendo de las instrucciones o lineamientos dados por el profesor.

El término de robot educativo es utilizado por muchos investigadores igual que robot pedagógico, y buscando su definición, según el diccionario de la Real Academia Española:

Educativo.- Que educa o sirve para educar.

Pedagógico.- Se dice de lo expuesto con claridad que sirve para educar o enseñar.

Por lo que en ocasiones se considera que cuando un robot se construye para que el estudiante genere el conocimiento a través de ese diseño se le considera “Robot Pedagógico” y cuando el robot ya está implementado en diferentes módulos para con ellos construir estructuras de ciertos problemas o diseños propuestos se considera “Robot Educativo”. Uno de los objetivos del presente proyecto de tesis es que el sistema de desarrollo que se está implementando cubra las dos conceptos, ya que el estudiante podrá construir el robot a partir de los manuales y diagramas o si el profesor lo proporciona implementado en módulos, utilizarlo para construir una solución propuesta a un problema que no sea la construcción del robot en sí.

Papert, Resnick y Steve Ocko en 1985 se juntaron para trabajar en un proyecto denominado Micromundos, término que Papert utilizó cuando creo el lenguaje LOGO, con este proyecto querían que los diseños que los jóvenes construían con la materiales que LEGO venia fabricando, como motores y bloques de luces, tuvieran un mayor impacto al proveerlos de movimientos programados utilizando la computadora que en esos tiempos empezaba a ser popular en las escuelas, y esa

programación debería ser hecha por los niños. Cuando la empresa LEGO participó en este proyecto en los años siguientes se obtuvieron resultados interesantes hasta llegar a la fabricación del módulo RCX por parte de la LEGO y de Criket por parte del laboratorio de medios del MIT.

Mitchel Resnick es actualmente director del grupo “Lifelong Kindergarden” del laboratorio de medios del MIT y es jefe del programa académico “Media Arts and Sciencies”, el ha colaborado con Papert desde los inicios del laboratorio de medios del MIT, es también Profesor Asociado Papert LEGO, también fue director del grupo de epistemología y aprendizaje del MIT.

Seymour Papert y Mitchel Resnick junto con su grupo de epistemología y aprendizaje desarrollaron el módulo programable y los submódulos que lo conforman el cual es conocido como la línea Mindstorm de LEGO, un robot educativo que en pleno 2011 se utiliza en muchas escuelas y en diferentes universidades alrededor del mundo en muchos proyectos de investigación en diferentes áreas del aprendizaje y como decía Seymour Papert en 1987:

“En la actualidad, pocos le dan importancia a esta idea, pero pronto debe formar parte de la toma de decisiones a todos los niveles en todos los países del mundo. Quien prospere y quien no lo haga, dependerá en gran medida de quien sea capaz de incursionar con éxito en la era del aprendizaje apoyado por la computación”(Seymour Papert, 1987).

Otro objetivo de la robótica pedagógica, es hacer que los estudiantes puedan construir sus propias representaciones y conceptos de ciencia y tecnología, por medio del control de entornos robotizados, al mismo tiempo que tratan de resolver problemas propuestos por ellos o por el profesor. Esto quiere decir que podrán realizar varios intentos directamente en los entornos adecuados, para que las representaciones estructuradas ayuden a construir su conocimiento.

El manejo de objetos reales hace una transferencia entre los campos de lo real, de lo concreto y el de la abstracción.

Una herramienta importante que ocupa la robótica pedagógica para cumplir sus objetivos es el robot educativo ya que es con este dispositivo que el estudiante modelará de forma física, alguna solución para un problema determinado. Esta herramienta fue diseñada primeramente por el laboratorio de medios del MIT y que lo llamó “Kit de invención robótica” (David Mindell, 2000).

En los avances de las investigaciones en la robótica pedagógica el Dr. Ruiz Velasco nos da la siguiente definición de esta herramienta:

“Los robots pedagógicos son instrumentos de laboratorio que funcionan cual periféricos (bidireccionales) de una computadora con el objetivo de provocar aprendizajes en los estudiantes adoptando como metodología la experimentación. Son reducciones fieles y significativas de modelos que tienen una estructura con sensores y accionadores y son controlados mediante un programa informático a través de la computadora.” (Ruiz Velasco, 2007).

Es por esto que se puede mencionar al robot educativo como un elemento importante o el más importante que la robótica pedagógica requiere para generar los ambientes de aprendizaje en los cuales los estudiantes puedan construir su conocimiento, teniendo únicamente como guía al profesor.

“El conocimiento no se transmite, se construye.”

Anexo II

; Archivo tipo .inc que contiene las variables que se
; ocupan en el programa

```
            CBLOCK    0x00
;*************************VARIABLES DE LA FUNCIÓN LCD*****************************************
AUX                                 ; Variable que ocupa la función Delay en el archivo Retardos
CONT                                ; Variable que ocupa la función Delay
CHRLCD                              ; Variable donde se carga el valor a mostrar en el display se utiliza en Función LCD
VLCD1                               ; Variable que se ocupa en Función LCD
VLCD2                               ; Variable que se ocupa en Función LCD
VLCD3                               ; Variable que se ocupa en Función LCD
COMANDO                             ; Variable que se ocupa en Función LCD

;*************************VARIABLES DE LA FUNCIÓN SENSOR DE DISTANCIA*****************************
DISTH                               ; Variable donde se guarda el byte alto que transmite el sensor de distancia por I2C, se
utiliza en las funciones del I2C
DISTL                               ; Variable donde se guarda el byte alto que transmite el sensor de distancia por I2C, se
utiliza en las funciones del I2C
DISTAUH                             ; Variable donde se guarda el valor que entrega el usuario parte alta
DISTAUL                             ; Variable donde se guarda el valor que entrega el usuario parte baja


;*************************VARIABLES DE LA FUNCIÓN DE MOTOR 1**************************************
DUTY1                               ; Variable para manejar el ciclo de trabajo
CONMOT1                             ; REGISTRO DE CONTROL DEL MOTOR 1
VUELTAS1L                           ; Variable del Byte bajo para limitar las vueltas que dé el motor 1
VUELTAS1H                           ; Variable del Byte alto para limitar las vueltas que dé el motor 1
FMT1                                ; REGISTRO DE BANDERAS DEL MOTOR1

;*************************VARIABLES DE LA FUNCIÓN DE MOTOR 2**************************************
DUTY2                               ; Variable para manejar el ciclo de trabajo
CONMOT2                             ; REGISTRO DE CONTROL DEL MOTOR 2
VUELTAS2L                           ; Variable del Byte bajo para limitar las vueltas que dé el motor 2
VUELTAS2H                           ; Variable del Byte alto para limitar las vueltas que dé el motor 2
FMT2                                ; REGISTRO DE BANDERAS DEL MOTOR2

;*************************VARIABLES DE LA FUNCIÓNES DE LOS CONVERTIDORES ANALOGICO-DIGITAL*******************

;VARADC0                            ; Variable 1 que ocupa el ADC0 donde se configura que operación lógica se va a realizar
;VALORADC0                          ; Variable 2 que ocupa el ADC0 donde se guarda el valor a comparar con el obtenido del
ADC0
;VARADC1                            ; Variable 1 que ocupa el ADC1 donde se configura que operación lógica se va a realizar
;VALORADC1                          ; Variable 2 que ocupa el ADC1 donde se guarda el valor a comparar con el obtenido del
ADC1
;VARADC2                            ; Variable 1 que ocupa el ADC2 donde se configura que operación lógica se va a realizar
;VALORADC2                          ; Variable 2 que ocupa el ADC2 donde se guarda el valor a comparar con el obtenido del
ADC2
;VARADC3                            ; Variable 1 que ocupa el ADC3 donde se configura que operación lógica se va a realizar
;VALORADC3                          ; Variable 2 que ocupa el ADC3 donde se guarda el valor a comparar con el obtenido del
ADC3
```

```
;*************************VARIABLES DE LA FUNCIÓNES DEL SENSOR DE LUZ*********************

VARLUZ                          ; Variable 1 que ocupa el ADC0 donde se configura que operación lógica se va a realizar
VALORLUZ0                       ; Variable 2 que ocupa el ADC0 donde se guarda el valor a comparar con el obtenido del
ADC0
VALORLUZ1                       ; Variable 2 que ocupa el ADC1 donde se guarda el valor a comparar con el obtenido del
ADC1
VALORLUZ2                       ; Variable 2 que ocupa el ADC2 donde se guarda el valor a comparar con el obtenido del
ADC2
VALORLUZ3                       ; Variable 2 que ocupa el ADC3 donde se guarda el valor a comparar con el obtenido del
ADC3

;*************************VARIABLES DE LA FUNCIÓNES DEL SENSOR DE SONIDO*********************

VARSON                          ; Variable 1 que ocupa el ADC0 donde se configura que operación lógica se va a realizar
VALORSON0                       ; Variable 2 que ocupa el ADC0 donde se guarda el valor a comparar con el obtenido del
ADC0
VALORSON1                       ; Variable 2 que ocupa el ADC1 donde se guarda el valor a comparar con el obtenido del
ADC1
VALORSON2                       ; Variable 2 que ocupa el ADC2 donde se guarda el valor a comparar con el obtenido del
ADC2
VALORSON3                       ; Variable 2 que ocupa el ADC3 donde se guarda el valor a comparar con el obtenido del
ADC3

;*************************VARIABLES DE LA FUNCIÓN BCD*********************************************
BCDH                            ; Variable para guardar el byte alto del resultado obtenido de la conversión se ocupa en la
Función BCD
BCDL                            ; Variable para guardar el byte bajo del resultado obtenido de la conversión se ocupa en la
Función BCD
CUENTA                          ; Variable que lleva la cuenta de los corrimientos deben ser 8 se ocupa en la Función BCD
BIN                             ; Variable donde se pasara el valor a convertir se ocupa en la Función BCD
BCD_TEMP                        ; Variable auxiliar      se ocupa en la Función BCD

;*************************VARIABLE DE LAS FUNCIÓNES DE SENSOR DE TOQUE 2 BITS POR FUNCIÓN*********************
CONTSTOQUE                      ; Registro de control del sensor de toque  se utiliza en las  funciones de Stoque

;*************************VARIABLES QUE SE OCUPAN EN LAS RUTINAS QUE FUNCIÓNAN COMO SWITCH
CASE************************
CHILO1                          ; Variable para realizar el switch case en la tarea 0
CHILO2                          ; Variable para realizar el switch case en la tarea 1

SEGUNDOS                        ; Variable que se utiliza en la función TIEMPO contiene los segundos de espera solicitados
por el usuario
SEG                             ; Variable interna de la función tiempo
        ENDC

SEMAFOROR0      EQU     0x64    ; Variable que funciona como semáforo donde se lee si está ocupado un periférico
SEMAFOROR1      EQU     0x65
SEMAFOROW0      EQU     0x70    ; Variable que funciona como semáforo donde escribe la tarea 0 y donde lee la otra u otras
tareas
SEMAFOROW1      EQU     0x71
```

```
; Archivo tipo inc.
; funciones para convertir el número binario de 8 bits a código BCD
; Para poderlo mostrar por el display, utiliza el algoritmo re corrimiento de bit y suma de 3


BCD     ;**********Función para mostrar un valor en un registro del PIC en el display LCD*******************
;{
                CALL    CONVER
                MOVLW   0x30                    ; Se carga el valor de 30 en W
                ADDWF   BCDH,F
                MOVFF   BCDH,CHRLCD
                CALL    LCD_CHR                 ; se imprimen las centenas


                MOVFF   BCDL,BCD_TEMP
                MOVLW   0xF0
                ANDWF   BCD_TEMP,F
                SWAPF   BCD_TEMP,F
                MOVLW   0x30
                ADDWF   BCD_TEMP,F
                MOVFF   BCD_TEMP,CHRLCD
                CALL    LCD_CHR                 ; se imprimen las decenas


                MOVFF   BCDL,BCD_TEMP
                MOVLW   0x0F
                ANDWF   BCD_TEMP,F
                MOVLW   0x30
                ADDWF   BCD_TEMP,F
                MOVFF   BCD_TEMP,CHRLCD
                CALL    LCD_CHR                 ; se imprimen las UNIDADES
                RETURN
;}



CONVER
        ;{ ;********FUNCION PARA CONVERTIR BINARIO A CODIGO BCD
                CLRF    BCDH
                CLRF    BCDL
                MOVLW   0x08
                MOVWF   CUENTA                  ; Se carga la variable CUENTA con valor de 8 para completar el re
corrimiento de un BYTE

CONVERSION
                BCF     STATUS,C                ;se pone a cero el bit de acarreo en el registro STATUS
                RLCF    BIN,F                   ;se realiza un re corrimiento a la izq. del registro BIN
                RLCF    BCDL,F                  ;se realiza un re corrimiento a la izq. del registro BCDL
                RLCF    BCDH,F                  ;se realiza un re corrimiento a la izq. del registro BCDH
                DECFSZ  CUENTA,F                ;se decremento la variable CUENTA y si llega a cero salta lo que
significa que ya se recorrieron los 8 bits
                BRA     FALTA                   ; si CUENTA no ha llegado a cero nos vamos a la rutina FALTA
```

```
                RETURN                  ; Si ya se recorrieron los 8 bits ya termino la función BCD2 y salimos de
ella
FALTA           MOVLW   0x0F            ; Cargamos el registro W con el valor 00001111 para un enmascarar el
nibble bajo
                ANDWF   BCDL,W          ;hacemos la operación AND entre W y el registro BCDL, el resultado se
deja en W
                MOVWF   BCD_TEMP        ; y lo guardamos en la variable BCD_TEMP
                MOVLW   0x05            ; cargamos el registro W con el valor 5
                SUBWF   BCD_TEMP,W      ;restamos el valor W a la fuente
                BTFSC   STATUS,C        ;revisamos si el resultado fue cero y brinca si lo hay
                CALL    SUMAX03         ; si no es cero significa que es un número mayor de 5 por lo que se
llama a la función para sumarle 3
                MOVLW   0xF0            ; Ahora se carga W con el valor 11110000 para la máscara del nibble
alto
                ANDWF   BCDL,W          ;se realiza la operación AND entre W y la variable BCDL se deja el
resulta en W
                MOVWF   BCD_TEMP        ; y ahora se pasa el valor a el registro BCD_TEMP
                MOVLW   0x50            ; se carga W con el valor de 5
                SUBWF   BCD_TEMP,W      ;se resta W de la fuente BCD_TEMP
                BTFSC   STATUS,C        ;revisamos si el resultado fue cero y brinca si lo hay
                CALL    SUMAX30         ; si no es cero significa que el valor de ese nibble es mayor de 5
                                        ; Por lo que se llama a la función para sumarle 3 a ese nibble
                GOTO    CONVERSION      ; si fue cero iteramos nuevamente la función


SUMAX03                                 ;********************Función para sumar 3 al nibble bajo cuando
es mayor de 5***************
                MOVLW   0x03            ; cargamos W con el valor de 3
                ADDWF   BCDL,F          ;se lo sumamos al registro BCDL
                BTFSS   STATUS,C        ;Revisamos C si la suma no fue cero se brinca
                RETURN                  ; si la suma resulto cero salimos de la función
                RLCF    BCDH,F          ;como la suma no fue cero se hace un re corrimiento de un bit a la izq.
                RETURN                  ; salimos de la función
SUMAX30                                 ;********************Función para sumar 3 al nibble bajo cuando
es mayor de 5***************
                MOVLW   0x30            ; cargamos W con el valor de 3
                ADDWF   BCDL,F          ;se lo sumamos al registro BCDL
                BTFSS   STATUS,C        ;Revisamos C si la suma no fue cero se brinca
                RETURN                  ; si la suma resulto cero salimos de la función
                RLCF    BCDH,F          ;como la suma no fue cero se hace un re corrimiento de un bit a la izq.
                RETURN                  ; salimos de la función

;}
```

; Archivo tipo INC para las funciones del I2C

```
INI_I2C ;{  ;**********************RUTINA DE CONFIGURACION DE PUERTOS Y DEL I2C
            MOVLW   0x0B
            MOVWF   ADCON1              ; TODOS LOS PINES COMO DIGITALES EXCEPTO AN0, AN1, AN2 Y AN3 QUE SON ANALOGOS
```

```
        BSF     TRISB,RB0       ; RB0 COMO ENTRADA DE SDA DEL I2C
        BSF     TRISB,RB1       ;RB1 COMO ENTRADA DE SCL DEL I2C
        BSF     SSPSTAT,SMP     ;SE TRABAJARA A 100KHZ
        MOVLW   0x9
        MOVWF   SSPADD          ; EFECTUA EL CÁLCULO PARA OPERAR A 100 KHZ
        MOVLW   0x28
        MOVWF   SSPCON1         ; SE CONFIGURA COMO I2C Y EN MODO MAESTRO
        RETURN
;}


                ;*****************RUTINA DE OBETENCION DEL DATO POR I2C*********************************
DATOI2C ;{
        CALL    START           ; LLAMAMOS A LA SUBRITINA DE LA CONDICION START
        MOVLW   0x15            ;MOV A W LA DIRECCION DEL ESCLAVO AL CUAL VAMOS A ENVIAR LA DIREC Y QUE VA SER
READ (DIRECCION "A" 0001010-1
        CALL    DIREC           ; LLAMAMOS A LA SUBRUTINA PARA ENVIAR LA DIREC DEL ESCLAVO
        CALL    RECIBIR         ; LLAMAMOS AL SUBRUTINA PARA PEDIR UN DATO
        MOVF    SSPBUF,W        ;CARGAMOS EL VALOR DEL REGISTRO SSPBUF QUE CONTIENE EL DATO QUE LLEGO AL REG
W
        MOVWF   DISTL
        CALL    OTRO            ; LLAMAMOS A LA SUBRUTINA PARA PREPARAR LA RECEPCION DE OTRO DATO
        CALL    RECIBIR         ; LLAMAMOS AL SUBRUTINA PARA PEDIR UN DATO
        MOVF    SSPBUF,W        ;CARGAMOS EL VALOR DEL REGISTRO SSPBUF QUE CONTIENE EL DATO QUE LLEGO AL REG
W
        MOVWF   DISTH
        CALL    NOMAS
        CALL    STOP
        RETURN

                ;***************************S U B R U T I N A S************************************************


                ;**********************RUTINA QUE EMPIEZA LA CONDICIO START
START   BCF     PIR1,SSPIF      ;PONEMOS A CERO LA BANDERA
        BSF     SSPCON2,SEN     ;ACTIVAMOS LA CONDICION DE INICIO
        BTFSS   PIR1,SSPIF      ;SSPIF NOS AVISA SI LA CONDICION DE INICIO TERMINO PONIENDOSE A "1"
        GOTO    $-1
        RETURN                  ; REGRESAMOS DEL CALL LA CONDICION DE START SE REALIZO


                ;**********************RUTINA QUE ENVIA LA DIRECCION DEL ESCLAVO

DIREC   BCF     PIR1,SSPIF      ;PONEMOS A CERO LA BANDERA
        MOVWF   SSPBUF          ; AHORA LO CARGAMOS EN EL REG SSPBUF
        BTFSS   PIR1,SSPIF      ;SSPIF NOS AVISA SI SE HA LLEGADO AL 9° PULSO DEL SCL
        GOTO    $-1
        RETURN                  ; REGRESAMOS DE ENVIAR LA DIRECCION


                ;**********************RUTINA QUE PIDE DATOS AL ESCLAVO*****************************
RECIBIR BCF     PIR1,SSPIF      ;PONEMOS A CERO LA BANDERA
        BSF     SSPCON2,RCEN    ;HABILITAMOS LA RECEPCION DEL MAESTRO
        BTFSS   PIR1,SSPIF      ;SSPIF NOS AVISA SI EL ESCLAVO ENVIO EL DATO
```

```
        GOTO    $-1
        RETURN                  ; SALIMOS DEL CALL EL DATO HA SIDO ENVIADO


                ;*****************RUTINA PARA AVISAR AL ESCLAVO QUE YA NO QUEREMOS MAS DATOS***************
NOMAS   BCF     PIR1,SSPIF      ;PONEMOS A CERO LA BANDERA
        BSF     SSPCON2,ACKDT   ;CONFIGURAMOS EL ACKDT CON 1 PARA AVISAR QUE TERMINO LA SOLICITUD DE DATOS
        BSF     SSPCON2,ACKEN   ;ENVIAMOS EL ACKDT
        BTFSS   PIR1,SSPIF      ;SSPIF NOS AVISA
        GOTO    $-1
        RETURN                  ; SALIMOS DEL CALL

                ;**********************RUTINA PARA DECIRLE AL ESCLAVO QUE ENVIE OTRO DATO**********************
OTRO    BCF     PIR1,SSPIF      ;PONEMOS A CERO LA BANDERA
        BCF     SSPCON2,ACKDT   ;CONFIGURAMOS EL ACKDT CON 0 PARA AVISAR QUE PEDIMOS OTRO DATO
        BSF     SSPCON2,ACKEN   ;ENVIAMOS EL ACKDT
        BTFSS   PIR1,SSPIF      ;SSPIF NOS AVISA
        GOTO    $-1
        RETURN                  ; SALIMOS DEL CALL


                ;**********************RUTINA QUE ENVIA UN STOP*****************************************
STOP    BCF     PIR1,SSPIF      ;PONEMOS A CERO LA BANDERA
        BSF     SSPCON2,PEN     ;ACTIVAMOS EL BIT DE PARO
        BTFSS   PIR1,SSPIF      ;SSPIF NOS AVISA SI SE HA LLEGADO AL 9° PULSO DEL SCL
        GOTO    $-1
        RETURN

;}


; FUNCIONES DEL LCD, este es un archivo .inc para ponerlo como cabecera en el archivo principal
; se utiliza B7-B4 como salida para los datos al modulo LCD
; y los pines D7-D5 para control E,RW y RS respectivamente


#define E               PORTD,7         ;Pin que se utiliza como señal E al modulo LCD
#define RW              PORTD,6
#define RS              PORTD,5
#define D4              PORTB,4
#define D5              PORTB,5
#define D6              PORTB,6
#define D7              PORTB,7
#define  LCD_HOME       0x02
#define LCD_CLR         0x01
#define LCD_CURSOROFF   0x0C
#define LCD_PARPA       0x0F
#define LCD_SPARPA      0x0E
#define ORIGEN          0x80
#define PORT            PORTB
```

```
;********************FUNCION DE INICIALIZACION DEL LCD*******************
INI_LCD
;{
        BCF     TRISB,7         ;Se configura como salida los pines 7,6,5,4 del puerto B
        BCF     TRISB,6         ;se hace individualmente para no afectar los pines restantes del puerto B ya que se ocupan
para otras cosas
        BCF     TRISB,5         ;Serán las líneas de datos D7,D6,D5 y D4 para comunicarse con el display LCD
        BCF     TRISB,4
        BCF     TRISD,7         ;E              Se configuran como salidas los pines 7, 6 y 5 del Puerto D
        BCF     TRISD,6         ;RW     que trabajaran como señales E, RW y RS respectivamente hacia el display LCD
        BCF     TRISD,5         ;RS

        BCF     E
        BCF     RS
        BCF     RW

        BCF     D7              ; la salida de datos a "0"
        BCF     D6
        BCF     D5
        BCF     D4              ; El nibble alto a "0"


        CALL    DELAY15MS       ; Tiempo de espera para que el Vcc se estabilice 15 ms
        CALL    DELAY5MS
        BSF     D4
        BSF     D5              ; Cargamos el nibble alto con 0011**** del puerto B, recomendación fabricante


        BSF     E               ; Ponemos a 1 la señal "E" se cargan el nibble
        CALL    DELAYNOP        ; Esperamos un tiempo de 3 ciclos
        BCF     E               ; Ponemos a "0" la señal "E"
        CALL    DELAY5MS        ; Esperamos 5 ms
        BSF     E               ; Ponemos a 1 la señal "E" se cargan el nibble
        CALL    DELAYNOP        ; Esperamos un tiempo de 3 ciclos
        BCF     E               ; Ponemos a "0" la señal "E"
        CALL    DELAY5MS        ; Esperamos 100 Microsegundos

        BSF     E               ; Ponemos a 1 la señal "E" se cargan el nibble
        CALL    DELAYNOP        ; Esperamos un tiempo de 3 ciclos
        BCF     E               ; Ponemos a "0" la señal "E"
        CALL    DELAY5MS        ; se realizo 3 veces esta operación x recomendación del fabricante
                ;*****************************************************************

        BCF     D4              ; nibble alto '0010',aviso de que se trabajara a bus de 4 lineas
        BSF     E
        CALL    DELAYNOP
        BCF     E
        CALL    DELAY5MS
                ;***********************************************
        BSF     E               ; se envía el nibble alto '0010' otra vez por ser ya de 4 bits
        CALL    DELAYNOP
        BCF     E

                ;***********************************************
```

```
BCF     D5
BSF     D7              ;se prepara nibble bajo '1000' display 2 lineas,5x7 dots
;CALL   BUSY_CHECK      ;Se envía el nibble
CALL    DELAY5MS
BSF     E
CALL    DELAYNOP
BCF     E
        ;*************************************************OK

BSF     D4
BCF     D7              ; se prepara nibble alto '0001' cursor

;CALL   BUSY_CHECK      ;Se envía el nibble bajo
CALL    DELAY5MS
BSF     E
CALL    DELAYNOP
BCF     E
        ;************************************************

BCF     D4              ; se prepara nibble bajo '0000' cursor move,shift izq
CALL    DELAY5MS
;CALL   BUSY_CHECK      ;Se envía el nibble alto
BSF     E
CALL    DELAYNOP
BCF     E
        ;*****************************************

;CALL   BUSY_CHECK      ;Se envía el nibble bajo
CALL    DELAY5MS
BSF     E               ; no se modifica el nibble '0000' y se pone como alto
CALL    DELAYNOP
BCF     E
        ;*****************************************
BSF     D4              ; se prepara el nibble bajo '0001' clear dispaly
CALL    DELAY5MS
;CALL   BUSY_CHECK      ;Se envía el nibble alto
BSF     E
CALL    DELAYNOP
BCF     E
        ;*********************************************************


BCF     D4              ; se prepara nibble alto '0000'
CALL    DELAY5MS
;CALL   BUSY_CHECK      ;Se envía el nibble bajo
BSF     E
CALL    DELAYNOP
BCF     E
        ;*********************************************************

BSF     D5              ; Se prepara el nibble bajo '1110' display on
BSF     D6
BSF     D7
```

```
        BSF     D4
        CALL    DELAY5MS
        ;CALL   BUSY_CHECK      ;Se envía el nibble alto
        BSF     E
        CALL    DELAYNOP
        BCF     E

        RETURN
;}

;********************SUBRUTINA PARA ENVÍAR UN CARACTER AL LCD************

LCD_CHR
;********************CONFIGURACION DE LA FUNCION*************************
;{
CONFIGURACIONLCD_CHR

        CALL    DELAY2MS
        ;call   BUSY_CHECK
        MOVF    CHRLCD,W
        ANDLW   0xF0
        MOVWF   VLCD1
        MOVF    PORTB,W         ;Pasamos el estado de PUERTOB a W
        ANDLW   0x0F            ; hacemos AND entre W y 0F para guardar el nibble bajo del puerto B
        IORWF   VLCD1,W         ;hacemos OR entre W y el valor de Char para preparar el nibble alto que se va a enviar
        MOVWF   LATB            ; lo colocamos en el puerto B
        BCF     RW
        BSF     RS              ; se escribirá un dato
        BSF     E               ; se envía el nibble alto
        NOP
        NOP
        BCF     E

        SWAPF   CHRLCD,F        ;Intercambiamos los nibbles de CHAR
        MOVF    CHRLCD,W
        ANDLW   0xF0
        MOVWF   VLCD1
        MOVF    PORTB,W         ;Pasamos el estado de PUERTOB a W
        ANDLW   0x0F            ; hacemos AND entre W y 0F para guardar el nibble bajo del puerto B
        IORWF   VLCD1,W         ;hacemos OR entre W y el valor de Char para preparar el nibble alto que se va a enviar
        MOVWF   LATB            ; lo colocamos en el puerto B
        BSF     E               ; se envía el nibble bajo
        NOP
        NOP
        BCF     E

        RETURN

;}

;********************SUBRUTINA DEL BUSY_CHECK****************************
BUSY_CHECK
;{
                        ;********************CONFIGURACION DE LA FUNCION**********************************
```

```
CONFIGURACIONBUSY_CHECK

        BSF     TRISB,7         ;configuramos las líneas de datos D7,D6,D5 y D4     como entradas
        BSF     TRISB,6
        BSF     TRISB,5
        BSF     TRISB,4
        BCF     RS              ; Sera comando
        BSF     RW              ; y de lectura
        NOP
        NOP

                ;*****************************VERIFICA LA FUNCION*************************
BUSY    BSF     E               ; disparamos E para obtener el nibble alto
        NOP
        BCF     E               ; termina el disparo
        BTFSC   PORTB,7         ;Revisamos D7 para saber si el busy flag está en "0" que significa que ya termino y saltar
        GOTO    FLAGA           ; no ha terminado saltamos a FLAGA nada más para terminar el proceso de lectura del otro
nibble
        GOTO    FLAGB

FLAGA   BSF     E               ; disparamos E para obtener el nibble bajo
        NOP
        NOP                     ; como no nos interesa leerlo hacemos nada con el
        BCF     E               ; termina el disparo
        GOTO    BUSY            ; volvemos a preguntar si ya termino el LCD

                ;*****************************TERMINA LA SUBRUTINA DE BUSY_CHECK*****************************

FLAGB   BSF                     ; este disparo solo es para complementar
        NOP                     ; que la lectura viene en dos partes
        NOP
        BCF     E
        BCF     RW              ; mandamos RW a cero
        BCF     TRISB,7         ;Se ponen como salida el D7,D6,D5 y D4 del pueto B
        BCF     TRISB,6
        BCF     TRISB,5
        BCF     TRISB,4
        RETURN                  ; salimos del busy_check
;}


;********************SUBRUTINA DE ESCRIBIR COMANDO***********************
WRCMND
                ;{*********************CONFIGURACION*******************
        CALL    DELAY2MS
        ;CALL   BUSY_CHECK
        MOVF    COMANDO,W
        ANDLW   0xF0
        MOVWF   VLCD1
        MOVF    PORTB,W         ;Pasamos el estado de PUERTOB a W
        ANDLW   0x0F            ; hacemos AND entre W y 0F para guardar el nibble bajo del puerto B
        IORWF   VLCD1,W         ;hacemos OR entre W y el valor de Char para preparar el nibble alto que se va a enviar
        MOVWF   LATB            ; lo colocamos en el puerto B
        BCF     RS              ; Configuramos RS y RW para avisar que se va a escribir un comando
```

```
        BCF     RW
        BSF     E               ; se envía el nibble alto
        NOP
        NOP
        NOP
        NOP
        BCF     E

        SWAPF   COMANDO,F       ;Intercambiamos los nibbles de CHAR
        MOVF    COMANDO,W
        ANDLW   0xF0
        MOVWF   VLCD1
        MOVF    PORTB,W         ;Pasamos el estado de PUERTOB a W
        ANDLW   0x0F            ; hacemos AND entre W y 0F para guardar el nibble bajo del puerto B
        IORWF   VLCD1,W         ;hacemos OR entre W y el valor de Char para preparar el nibble alto que se va a enviar
        MOVWF   LATB            ; lo colocamos en el puerto B
        BSF     E               ; se envía el nibble bajo
        NOP
        NOP
        NOP
        NOP
        BCF     E


        RETURN
;}
```

; Archivo tipo INC que contiene las funciones del motor1

```
        #DEFINE         ADELANTEM1      BCF     CONMOT1,5
        #DEFINE         REVERSAM1       BSF     CONMOT1,5
        #DEFINE         ADELANTEM2      BCF     CONMOT2,5
        #DEFINE         REVERSAM2       BSF     CONMOT2,5
        #DEFINE         LIBREM1         BSF     CONMOT1,7
        #DEFINE         LIBREM2         BSF     CONMOT2,7


                        ;*********************SUBRUTINA PARA QUE TRABAJE EL MOTOR
1*****************************************

MOTOR1

                        ;///////////////ETAPA DE CONFIGURACION DEL
MOTOR1////////////////////////////////////////////////////////////////////////////
CONFIGURACIONM1  ;{
                        ;**********************CONFIGURACION DEL MODULO CCP1 PARA TRABAJAR EN MODO
PWM*******************
                BTFSC   INDF1,1                 ;verificamos el bit SEMMOT1 para saber si está ocupado el periférico,
INDF1 ES APUNTADOR
```

```
                RETLW   0x00                    ; Como el bit está en "1" el periférico está ocupado salimos de la
función
                BTFSC   FMT1,1                  ;Revisamos bandera si es 0 no esta config y salta para que se configure
                GOTO    VERIFM1                 ;si es 1 ya está configurada y saltara a ser únicamente verificada
                BTFSC   CONMOT1,6               ;Revisamos si hay que detener el motor si es cero salta a la config. Y si
es 1 se dirige a terminar
                BRA     TERMINARM1
                CLRF    CCP1CON                 ; El modulo CCP está apagado
                CLRF    TMR2                    ; TMR2 apagado
                MOVLW   0x1E                    ; Cargamos el registro PR2
                MOVWF   PR2                     ; con el valor 0x1E para obtener una frecuencia de 2 KHz en el PWM
                MOVLW   0x02                    ; Asignamos el pre escalador 1:16 al timer2
                MOVWF   T2CON
                BCF     TRISC,2                 ;RC2 como salida del PWM1
                BCF     TRISD,1                 ;RD1 como salida para la dirección del motor1 va a la entrada IN1 del
puente L293D
                BCF     TRISD,2                 ;RD2 como salida para la dirección del motor1 va a la entrada IN2 del
puente L293D
                CLRF    CCPR1L
                MOVLW   0x0C                    ; Habilitamos como PWM el CCP1
                MOVWF   CCP1CON
                BCF     PORTD,1
                BCF     PORTD,2
                ;******CONFIGURACION DEL TIMER 1 COMO CONTADOR Y CONTROL DEL MOTOR1****
                MOVLW   0x83                    ; Cargamos el registro T0CON para configurar el timer 1
                MOVWF   T1CON                   ; con entrada por el RC0 del decoder proveniente del motor
                BSF     TRISC,0                 ;RC0 como entrada
                CLRF    STATUS

                        ;************Empieza a trabajar el motor**********************
                CLRF    TMR1L                   ; Ponemos a ceros el registro del timer1
                MOVFF   VUELTAS1H,TMR1H
                MOVFF   VUELTAS1L,TMR1L
                BSF     T2CON,TMR2ON            ;Inicia el incremento del TIMER2
                BSF     PORTD,1                 ;a 1 el IN1 del L293
                BSF     PORTD,2                 ;a 1 el IN2 del L293
                BSF     FMT1,1                  ;avisamos que ya se configuro el MOTOR1
                BSF     INDF2,1                 ;avisamos que esta ocupado el periférico con el bit SEMMOT1, INDF2
ES APUNTADOR
                ;}
                ;////////////////////////ETAPA DE VERIFICACION////////////////////////////////////////////////////////////
VERIFICACIONM1;{

                BTFSC   CONMOT1,5               ;REVISAMOS QUE DIRECCION SE PROGRAMO EN EL BIT 5 DEL REG
CONTROL DE MOTOR1
                BRA     REVERM1
                BCF     PORTD,2                 ;SE PONE A "0" RD2 PARA EL SENTIDO ADELANTE
                MOVF    DUTY1,W                 ;CARGAMOS EL VALOR DE DUTY1 EN REG W
                MOVWF   CCPR1L                  ; Y LO PASAMOS AL REGISTRO QUE CONTROLA EL PWM PARA QUE LO
AJUSTE
                BRA     VERIFM1

REVERM1         BCF     PORTD,1                 ;se pone a "1" RD2 para el sentido de reversa
                MOVF    DUTY1,W                 ;MOVEMOS EL VALOR DE  DUTY1 AL REG W
```

```
                MOVWF   CCPR1L                  ; Y Lo cargamos en el registro para que el PWM lo ajuste
VERIFM1         BTFSC   CONMOT1,7               ;Revisamos si se programo movimiento libre o de cuenta si es cero
salta, no es libre
                BRA     LIBRE1                  ; fue uno es libre y se dirige a LIBRE1 para terminar
                BTFSS   PIR1,TMR1IF             ;Revisamos si se desbordo el timer1
                RETLW   0x00
                ;}

                ;//////////////////////////////ETAPA DE TERMINAR//////////////////////////////////////////
TERMINARM1 ;{

                CLRF    CCP1CON                 ; Apagamos el CCP como pwm
                BCF     PORTC,1
                BCF     PORTD,1                 ;ponemos a "0" la entrada IN1 del puente L293
                BCF     PORTD,2                 ;ponemos a "0" la entrada IN2 del puente L293 para detener el motor
                BCF     STATUS,2
                CLRF    TMR0L                   ; Ponemos a ceros el registro del timer0
                ;BCF    INTCON,TMR0IF           ;ponemos a cero la bandera de interrupción del timer 0
                BCF     PIR1,TMR1IF             ;ponemos a cero la bandera de interrupción del timer 1

LIBRE1          BCF     FMT1,1                  ;ponemos a cero la bandera de configuración
                BCF     INDF2,1                 ;Liberamos el periférico, INDF2 es apuntador
                RETLW   0x01                    ; la función termino
                ;}


MOTOR2  ;{

                ;///////////////ETAPA DE
CONFIGURACION//////////////////////////////////////////////////////////////////////////////
CONFIGURACIONM2 ;{
                ;**********************CONFIGURACION DEL MODULO CCP1 PARA TRABAJAR EN MODO
PWM*********************
                BTFSC   INDF1,2                 ;verificamos el bit SEMMOT2 para saber si esta ocupado el periférico,
INDF1 ES APUNTADOR
                RETLW   0x00
                BTFSC   FMT2,1                  ;Revisamos bandera si es 0 no esta config y salta para que se configure
                GOTO    VERIFM2                 ; si es 1 ya esta configurada y saltara a ser únicamente verificada
                CLRF    CCP2CON                 ; El modulo CCP esta apagado
                CLRF    TMR2                    ; TMR2 apagado
                MOVLW   0x1E                    ; Cargamos el registro PR2
                MOVWF   PR2                     ; con el valor 0x1E para obtener una frec de 2 KHz en el PWM
                MOVLW   0x02                    ; Asignamos el pre escalador 1:16 al timer2
                MOVWF   T2CON
                BCF     TRISC,1                 ;RC1 como salida del PWM2
                BCF     TRISD,3                 ;RD4 como salida para el IN3 del puente L293 del motor2
                BCF     TRISD,4                 ;RD3 como salida para el IN4 del puente L293 del motor2

                CLRF    CCPR2L
                MOVLW   0x0C                    ; Habilitamos como PWM el CCP2
                MOVWF   CCP2CON
                BCF     PORTD,3
                BCF     PORTD,4
                ;******CONFIGURACION DEL TIMER 0 COMO CONTADOR Y CONTROL DEL MOTOR2****
```

```
                MOVLW   0xB8                    ; Cargamos el registro T0CON para configurar el timer 0
                MOVWF   T0CON                   ; con entrada por el RA4 del decoder proveniente del motor
                BSF     TRISA,4                 ;RA4 como entrada
                CLRF    STATUS


                                ;************Empieza a trabajar el motor**********************
                CLRF    TMR0L                   ; Ponemos a ceros el registro del timer0
                MOVFF   VUELTAS2H,TMR0H
                MOVFF   VUELTAS2L,TMR0L
                BSF     T2CON,TMR2ON            ;Inicia el incremento del TIMER2
                BSF     PORTD,3                 ;a 1 el enable del L293 motor en movimiento
                BSF     PORTD,4
                BSF     FMT2,1
                BSF     INDF2,2                 ;avisamos que esta ocupado el periférico con el bit SEMMOT2, INDF2
ES APUNTADOR

                ;}

                ;////////////////////////ETAPA DE VERIFICACION DEL MOTOR
2////////////////////////////////////////////////////////////////
VERIFICACIONM2;{

                BTFSC   CONMOT2,5               ;REVISAMOS QUE DIRECCION SE PROGRAMO EN EL BIT 5 DEL REG
CONTROL DE MOTOR1
                BRA     REVERM2
                BCF     PORTD,4                 ;SE PONE A "0" RD2 PARA EL SENTIDO ADELANTE
                MOVF    DUTY2,W                 ;CARGAMOS EL VALOR DE DUTY2 EN REG W
                MOVWF   CCPR2L                  ; Y LO PASAMOS AL REGISTRO QUE CONTROLA EL PWM PARA QUE LO
AJUSTE
                BRA     VERIFM2
REVERM2         BCF     PORTD,3                 ;se pone a "1" RD2 para el sentido de reversa
                MOVF    DUTY2,W                 ;MOVEMOS EL VALOR DE  DUTY2 AL REG W
                MOVWF   CCPR2L                  ; Y Lo cargamos en el registro para que el PWM lo ajuste

VERIFM2         BTFSC   CONMOT2,7               ;Revisamos si se programo movimiento libre o de cuenta si es cero
salta, no es libre
                BRA     LIBRE2
                BTFSS   INTCON,TMR0IF           ;Revisamos si se desbordo el timer0
                RETLW   0x00                    ; no ha terminado la función
                ;}

                ;////////////////////////////ETAPA DE TERMINAR MOTOR 2/////////////////////////////////////////
TERMINARM2 ;{

                BCF     PORTD,3                 ;Detenemos el motor2 poniendo a "0"  IN3 e IN4
                BCF     PORTD,4
                CLRF    CCP2CON                 ; Apagamos el CCP como pwm

                BCF     STATUS,2
                CLRF    TMR1L                   ; Ponemos a ceros el registro del timer1
                BCF     INTCON,TMR0IF           ;ponemos a cero la bandera de interrupción del timer 0
                ;BCF    PIR1,TMR1IF             ;ponemos a cero la bandera de interrupción del timer 1
LIBRE2          BCF     FMT2,1
```

```
                BCF     INDF2,2                 ;Liberamos el periférico, INDF2 ES APUNTADOR
                RETLW   0x01
                ;}

;********************RUTINA PARA MOVER LOS DOS MOTORES AL MISMO TIEMPO ******************************
MOVIMIENTO      ;{

                        ;///////////////ETAPA DE
CONFIGURACION//////////////////////////////////////////////////////////////////////////////
CONFIGURACIONMOV        ;{
                        ;***********************CONFIGURACION DEL MODULO CCP1 PARA TRABAJAR EN MODO
PWM********************
                BTFSC   INDF1,3                 ;verificamos el bit SEMMOV para saber si está ocupado el periférico,
INDF1 ES APUNTADOR
                RETLW   0x00                    ; Como el bit está en "1" el periférico esta ocupado salimos de la
función
                BTFSC   FMT1,1                  ;Revisamos bandera si es 0 no esta config y salta para que se configure
                GOTO    VERIFMOV                ;si es 1 ya esta configurada y saltara a ser únicamente verificada
                BTFSS   CONMOT1,7               ;Revisamos el bit 7 del registro CONMOT1 para saber si el mov del
motor es libre o se cuentan los giros
                CLRF    CCP1CON                 ; El modulo CCP está apagado
                CLRF    CCP2CON                 ; El modulo CCP está apagado
                CLRF    TMR2                    ; TMR2 apagado
                MOVLW   0x1E                    ; Cargamos el registro PR2
                MOVWF   PR2                     ; con el valor 0x1E para obtener una frec de 2 KHz en el PWM
                MOVLW   0x02                    ; Asignamos el pre escalador 1:16 al timer2
                MOVWF   T2CON
                BCF     TRISC,2                 ;RC2 como salida del PWM1
                BCF     TRISC,1                 ;RC1 como salida del PWM2
                BCF     TRISD,1                 ;RD4 como salida para IN1 del puente L293 del motor1
                BCF     TRISD,2                 ;RD2 como salida para IN2 del puente L293D del motor1
                BCF     TRISD,3                 ;RD3 como salida para IN3 del puente L293D del motor2
                BCF     TRISD,4                 ;RD4 como salida para IN4 del puente L293D del motor2

                CLRF    CCPR1L
                CLRF    CCPR2L
                MOVLW   0x0C                    ; Habilitamos como PWM el CCP1 y el CCP2
                MOVWF   CCP1CON
                MOVWF   CCP2CON
                BCF     PORTD,1
                BCF     PORTD,2
                BCF     PORTD,3
                BCF     PORTD,4
                        ;******CONFIGURACION DE LOS TIMER 0 Y 1 COMO CONTADOR Y CONTROL DEL MOTOR 1
y2
                MOVLW   0xB8                    ; Cargamos el registro T0CON para configurar el timer 0
                MOVWF   T0CON                   ; con entrada por el RA4 del decoder proveniente del motor
                BSF     TRISA,4                 ;RA4 como entrada
                MOVLW   0x83                    ; Cargamos el registro T0CON para configurar el timer 1
                MOVWF   T1CON                   ; con entrada por el RC0 del decoder proveniente del motor
                BSF     TRISC,0                 ;RC0 como entrada
                CLRF    STATUS

                        ;************Empieza a trabajar el motor**********************
```

```
                CLRF    TMR0L                   ; Ponemos a ceros el registro del timer0
                CLRF    TMR1L
                MOVFF   VUELTAS1H,TMR0H
                MOVFF   VUELTAS1L,TMR0L
                MOVFF   VUELTAS2H,TMR1H
                MOVFF   VUELTAS2L,TMR1L
                BSF     T2CON,TMR2ON            ;Inicia el incremento del TIMER2
                BSF     PORTD,1                 ;Ponemos a "1" las salidas que controlan el puente
                BSF     PORTD,2                 ;para prepararlas para trabajar el motor
                BSF     PORTD,3
                BSF     PORTD,4
                BSF     FMT1,1
                BSF     FMT2,1
                BSF     INDF2,3                 ;avisamos que esta ocupado el periférico con el bit SEMMOV, INDF2
ES APUNTADOR

                ;}

                ;//////////////////////////ETAPA DE VERIFICACION DE LA FUNCIÓN
MOVIMIENTO//////////////////////////////////////////////////////////////////////
VERIFICACIONMOV;{

                BTFSC   CONMOT1,5               ;REVISAMOS QUE DIRECCION SE PROGRAMO EN EL BIT 5 DEL REG
CONTROL DE MOTOR1
                BRA     REVERMOV
                BCF     PORTD,2                 ;SE PONE A "0" RD2 PARA EL SENTIDO ADELANTE del MOTOR1
                BCF     PORTD,4                 ;Se pone a "0" RD4 para el sentido adelante del MOTOR2
                MOVFF   DUTY1,CCPR1L            ;CARGAMOS EL VALOR DE DUTY1 EN EL REGISTRO QUE CONTROLA EL
PWM PARA QUE LO AJUSTE
                MOVFF   DUTY2,CCPR2L
                BRA     VERIFMOV
REVERMOV
                BCF     PORTD,1                 ;se pone a "0" RD1 para el sentido de reversa del motor1
                BCF     PORTD,3                 ;se pone a "0" RD3 para el sentido de reversa del motor2
                MOVF    DUTY1,W                 ;MOVEMOS EL VALOR DE  DUTY1 AL REG W
                MOVWF   CCPR1L                  ; Y Lo cargamos en el registro para que el PWM lo ajuste
                MOVF    DUTY2,W                 ;MOVEMOS EL VALOR DE  DUTY2 AL REG W
                MOVWF   CCPR2L                  ; Y Lo cargamos en el registro para que el PWM lo ajuste

VERIFMOV
                BTFSS   INTCON,TMR0IF           ;Revisamos si se desbordo el timer0
                RETLW   0x00
                ;}

                ;//////////////////////////////ETAPA DE TERMINAR/////////////////////////////////////////
TERMINARMOV ;{

                CLRF    CCP1CON                 ; Apagamos el CCP como pwm
                CLRF    CCP2CON                 ; Apagamos el CCP como pwm
                BCF     PORTC,1                 ;y ponemos a ceros los dos in del puente para parar el motor
                BCF     PORTC,2
                BCF     PORTD,1                 ;y ponemos a ceros IN1 e IN2 del puente para parar el motor1
                BCF     PORTD,2
                BCF     PORTD,3                 ;;y ponemos a ceros IN3 e IN4 del puente para parar el motor2
```

```
        BCF     PORTD,4
        BCF     STATUS,2
        CLRF    TMR0L                   ; Ponemos a ceros el registro del timer0
        BCF     INTCON,TMR0IF           ;ponemos a cero la bandera de interrupción del timer 0
        BCF     PIR1,TMR1IF             ;ponemos a cero la bandera de interrupción del timer 1
        BCF     FMT1,1
        BCF     FMT2,1
        BCF     INDF2,3                 ;Liberamos el periférico, INDF2 ES APUNTADOR
        RETLW   0x01                    ;la función termino
        ;}
```

; Archivo que contiene la función para realizar los tiempos de espera que especifique el usuario

; Estos tiempos se efectúan con el TIMER3 del microcontrolador

```
;*****************FUNCION DE ESPERA CON EL TIMER***************
TIEMPO
        ;**********Configuración del periférico
        BTFSC   INDF1,5                 ;verificamos el bit SEMTIMER para saber si está ocupado el periférico, INDF1 ES
APUNTADOR
        RETLW   0x00                    ; Como el bit está en "1" el periférico está ocupado salimos de la función
        BTFSC   FMT1,0                  ;Revisamos bandera si es 0 no esta config y salta para que se configure
        GOTO    VERTIEMPO               ; si es 1 ya está configurada y saltara a ser únicamente verificada
        MOVLW   b'10110000'
        MOVWF   T3CON                   ; Configuramos el TIMER3 como contador de 16 bits y preescaler 1:8
        CLRF    TMR3L
        MOVFF   SEGUNDOS,SEG            ;cargamos la variable interna SEG con el valor que dio el usuario ya multiplicado
por dos
        BSF     FMT1,0                  ;avisamos que ya se configuro la función
        BSF     INDF2,5                 ;avisamos que está ocupado el periférico con el bit SEMTIMER, INDF2 ES
APUNTADOR

TIEMP
        MOVLW   0x0B
        MOVWF   TMR3H
        MOVLW   0xDB
        MOVWF   TMR3L                   ; Cargamos el TMR3 con valor 0x0BDB, 3035 decimal para que pueda contar un
segundo como base
        BSF     T3CON,0                 ;inicia el incremento del TMR3

VERTIEMPO       ;***************VERIFICACION DE LA FUNCION ******************
        BTFSS   PIR2,TMR3IF             ;Verificamos si ya desbordo el TMR3
        RETLW   0x00                    ; No se ha desbordado el TMR3 salimos de la función
        BCF     PIR2,TMR3IF             ;Desbordo TMR3 ponemos a "0" la bandera
        DECFSZ  SEG                     ; Decrementamos la variable que contiene el tiempo solicitado y si es "0" salta
señal de que termino
        BRA     TIEMP                   ; no ha terminado el timer y se carga nuevamente el TMR3
                ;********TERMINACION DE LA FUNCION***************
        BCF     FMT1,0                  ;Ponemos a "0" el bit de configuración BVTMR
        BCF     INDF2,5                 ;Liberamos el periférico con el bit SEMTIMER, INDF2 ES APUNTADOR
        RETLW   0x01                    ; Terminó la función salimos de la misma
```

; Archivo de tipo .INC para ponerlo como cabecera en el archivo principal

```
; Funciones de retardo con reloj a 4 MHz
; Para calcular el tiempo aprox se utilizo la siguiente formula Delay=W*((3*256)+3)*Tcy
; Estos retardos se utilizan en la función de inicialización del LCD

DELAY100                                    ; Retardo de 100 microsegundos
                MOVLW   0x21
                MOVWF   CONT
LOOP1   DECFSZ  CONT
                GOTO    LOOP1
                RETURN
DELAY2MS
                MOVLW   0x03                ; Retardo de 2.313 milisegundos
                MOVWF   AUX
                CLRF    CONT
LOOP3   DECFSZ  CONT
                GOTO    LOOP3
                DECFSZ  AUX
                GOTO    LOOP3
                RETURN
DELAY5MS
                MOVLW   0x07                ; Retardo de 5.397 milisegundos
                MOVWF   AUX
                CLRF    CONT
LOOP4   DECFSZ  CONT
                GOTO    LOOP4
                DECFSZ  AUX
                GOTO    LOOP4
                RETURN
DELAY10MS
                MOVLW   0x0D                ; Retardo de 10.023 milisegundos
                MOVWF   AUX
                CLRF    CONT
LOOP5   DECFSZ  CONT
                GOTO    LOOP5
                DECFSZ  AUX
                GOTO    LOOP5
                RETURN
DELAY15MS
                MOVLW   0x14                ; Retardo de 15.420 milisegundos
                MOVWF   AUX
                CLRF    CONT
LOOP6   DECFSZ  CONT
                GOTO    LOOP6
                DECFSZ  AUX
                GOTO    LOOP6
                RETURN
DELAY20MS
                MOVLW   0x1A                ; Retardo de 20.046 milisegundos
                MOVWF   AUX
                CLRF    CONT
```

```
LOOP7   DECFSZ  CONT
                GOTO    LOOP7
                DECFSZ  AUX
                GOTO    LOOP7
                RETURN
DELAY200MS
                MOVLW   0xFF                    ; Retardo de 196.605 milisegundos
                MOVWF   AUX
                CLRF    CONT
LOOP8   DECFSZ  CONT
                GOTO    LOOP8
                DECFSZ  AUX
                GOTO    LOOP8
                RETURN

DELAYNOPNOP                                     ; Retardo de un NOP que lo hace en 6 microsegundos
                RETURN
```

; Archivo que contiene la función para solicitar la distancia

```
; Que obtiene el sensor de distancia vía i2c

#define DISTAMAYORA     BSF     FMT1,4
#define DISTAMENORA     BCF     FMT1,4


SDISTANCIA

;************CONFIGURACION DE LA FUNCION SDISTANCIA***************************
        BTFSC   INDF1,0                 ;verificamos el bit SEMSDIST para saber si está ocupado el periférico, INDF1 ES
APUNTADOR
        RETLW   0x00                    ; Como el bit está en "1" el periférico está ocupado salimos de la función
        BTFSC   FMT1,3                  ;Revisamos bandera si es 0 no esta config y salta para que se configure
        BRA     VERISD
        CALL    INI_I2C                 ; se inicializa el periférico del I2C
        BSF     FMT1,3
        BSF     INDF2,0                 ;avisamos que está ocupado el periférico con el bit SEMSDIST, INDF2 ES
APUNTADOR

VERISD
        CALL    DATOI2C                 ; llamamos a la función que le pide el dato de la distancia al sensor vía I2C
        MOVLW   0x02
        CPFSGT  DISTH
        RETLW   0x00
        MOVFF   DISTH,BIN       ;el dato que regresa el sensor es cargado en la variable DISTH y es pasado a la variable BIN
para ser mostrada en el LCD
        MOVLW   0x84                    ; Posición del cursor en el display es "0" en el renglón 1
        MOVWF   COMANDO                 ; el valor lo cargamos en la variable COMANDO
        CALL    WRCMND                  ; Se llama a la función para que ejecute la orden
        CALL    BCD             ; llamamos a la función BCD para que muestre el valor de la variable BIN en el LCD y en la
posición que se dio anteriormente
        MOVLW   0x87                    ; posición para imprimir en el LCD la variable DISTL que entrego el sensor de
distancia
```

```
        MOVWF   COMANDO                 ; pasamos el valor a la variable COMANDO
        CALL    WRCMND                  ; invocamos a la función para ejecutar el comando
        MOVFF   DISTL,BIN               ;pasamos el valor del registro DISTL a la variable BIN para su impresión
        CALL    BCD                     ; llamamos a la función BCD para que imprima el valor de BIN

        BTFSC   FMT1,4                  ;Revisamos que operación lógica configuro el usuario si es < salta
        BRA     mayorque                ; si no salto es una operación de = ó >

                ;******************INSTRUCCIONES PARA LA OPERACION LOGICA MENOR QUE < **********************
menorque
        MOVF    DISTAUH,W               ;vemos si son iguales el byte alto del entero de 16 bits
        CPFSEQ  DISTH                   ; hacemos la comparación si es igual salta
        BRA     NOIGUAL                 ; como no es igual es flujo se va a la rutina no igual
IGUAL                                   ; como fueron iguales los bytes ALTOS ahora hay que ver la operación menor que
con los bytes BAJOS
        MOVF    DISTAUL,W               ;movemos el valor del usuario al WREG
        CPFSLT  DISTL                   ; se realiza la operación menor que y si es verdadera salta
        RETLW   0x00                    ; como no fue menor aun no se cumple la condición y la función aun no termina
salimos con W=0
        BRA     TERMINADIST             ; salto se cumplió la condición y nos vamos a la rutina de terminado

NOIGUAL                 ; rutina para cuando los dos bytes ALTOS no son iguales por lo tanto vemos cual es menor que y
sabremos que se cumple la condición
        MOVF    DISTAUH,W               ;movemos el valor ALTO del usuario a WREG
        CPFSLT  DISTH                   ; realizamos la operación lógica < y si es cierta saltamos
        RETLW   0x00                    ; fue falsa no se ha cumplido la condición salimos la función no ha terminado
W=0
        BRA     TERMINADIST             ; hubo salto se cumplió la condición y nos vamos a la rutina de terminado


                ;******************INSTRUCCIONES PARA LA OPERACION LOGICA MAYOR QUE > **********************
mayorque                                ;hubo salto es una operación de >
        MOVF    DISTH,W                 ;movemos el byte ALTO de sensor de distancia a WREG
        CPFSEQ  DISTAUH                 ; lo comparamos con el byte ALTO del valor del usuario
        BRA     NOIGUAL2                ; no fueron iguales el flujo va a la rutina respectiva
IGUAL2                                  ; hubo salto son iguales los bytes ALTOS por lo que hay que compara los bytes
BAJOS
        MOVF    DISTAUL,W               ;movemos el byte BAJO del valor del usuario a WREG
        CPFSGT  DISTL                   ; se realiza la operación lógica > y si es cierta hay salto
        RETLW   0x00                    ; no se cumplió la condición y salimos de la función la cual no ha terminado W=0
        BRA     TERMINADIST             ; hubo salto se cumplió la condición y nos vamos a la rutina de terminado

NOIGUAL2        ; rutina cuando los bytes ALTOS no son iguales por lo que hay que ver cual es mayor que y con eso se puede saber
si se cumple la condición
        MOVF    DISTAUH,W               ;movemos el byte ALTO del valor del usuario al WREG
        CPFSGT  DISTH                   ; se realiza la operación lógica > y si es cierta salta
        RETLW   0x00                    ; no se cumplió la condición y salimos de la función no ha terminado W=0

TERMINADIST
        BCF     FMT1,3                  ;Ponemos a cero la bandera de configuración
        BCF     INDF2,0                 ;liberamos el periférico
        RETLW   0x01                    ; salimos de la función terminada W=1
```

; **Archivo tipo .inc que contiene las funciones del sensor de luz**

```
;

#define    SLUZ0MAYORA        BSF        VARLUZ,0
#define    SLUZ0MENORA        BCF        VARLUZ,0
#define     SLUZ1MAYORA       BSF        VARLUZ,1
#define    SLUZ1MENORA        BCF        VARLUZ,1
#define    SLUZ2MAYORA        BSF        VARLUZ,2
#define    SLUZ2MENORA        BCF        VARLUZ,2
#define    SLUZ3MAYORA        BSF        VARLUZ,3
#define     SLUZ3MENORA       BCF        VARLUZ,3
#define    LUZON0             BSF        VARLUZ,4
#define    LUZOFF0            BCF        VARLUZ,4
#define    LUZON1             BSF        VARLUZ,5
#define    LUZOFF1            BCF        VARLUZ,5
#define    LUZON2             BSF        VARLUZ,6
#define    LUZOFF2            BCF        VARLUZ,6
#define    LUZON3             BSF        VARLUZ,7
#define    LUZOFF3            BCF        VARLUZ,7
SLuz0      ;**************FUNCION DEL SENSOR DE LUZ CON EL ADC0***************
;******CONFIGURACION_ADC0
;{
           INCF       FSR1L                    ; Como el bit de verif del ADC está en el siguiente registro incrementamos el
apuntador
           BTFSC      POSTDEC1,0       ;verificamos el bit SEMADC0 para saber si está ocupado el periférico decrementamos el
apuntador para dejarlo igual
           RETLW      0x00
           BSF        TRISA,0                  ;Bit 0 del puerto A como entrada, AN0
           MOVLW      0x0B
           MOVWF      ADCON1                   ; Configuramos AN0-AN3 como analógicas
           MOVLW      0x14
           MOVWF      ADCON2                   ; Seleccionamos los tiempos de adquisición
           BCF        TRISA,5                  ;Bit 5 del puerto A como salida
           BCF        PORTA,5                  ;Ponemos a 0 el RA5
           BTFSC      VARLUZ,4                 ;Verificamos si se prende el led del sensor si es 0 salta led OFF
           BSF        PORTA,5                  ;Se enciende el led ON
           INCF       FSR2L                    ; Como el bit SEMADC0 del ADC está en el siguiente registro incrementamos el
apuntador
           BSF        POSTDEC2,0               ;Avisamos que está ocupado el periférico ADC0
;}
;*******VERIFICACION DEL SLUZ0************
;{
VERIFICACIONLUZ0
           CLRF       ADCON0                   ; Seleccionamos el canal AN0
           BSF        ADCON0,0                 ;Habilitamos el modulo ADC
           BSF        ADCON0,GO                ;inicia la adquisición GO
           NOP                                 ; 10 microseg para realizar la conversión
           NOP
           NOP
           NOP
           NOP
```

```
            NOP
            NOP
            NOP
            NOP
            NOP
            NOP
            MOVF    VALORLUZ0,W             ;cargamos el valor que dio el usuario en WREG
            BTFSC   VARLUZ,0                ;probamos el bit "0" de la variable VARLUZ, si es 1 será una operación > y si es
cero salta
            BRA     MAYORKLUZ0
MENORKLUZ0
            CPFSLT  ADRESH                  ; saltamos aquí si fue una operación < y comparamos la medición del ADC con
WREG
            BRA     SINTERMINARLUZ0         ; si no es menor saltamos a SINTERMINAR0
            BRA     TERMINADOLUZ0           ; si es menor se cumple la condición y saltamos a TERMINADOLUZ0
MAYORKLUZ0
            CPFSGT  ADRESH                  ; saltamos aquí si fue una operación > y comparamos el valor medido por ADC
con WREG
SINTERMINARLUZ0
            RETLW   0x00                    ; No hubo salto no ha terminado la función y salimos

;}
;***********TERMINAR EL LUZO***************
;{
TERMINADOLUZ0                               ; salto o llego por bra entonces la condición se cumplió

            INCF    FSR2L                   ; Como el bit de SEMADC0 del ADC está en el siguiente registro incrementamos el
apuntador
            BSF     POSTDEC2,0              ;Liberamos el periférico ADC0
            ;BCF    PORTA,5
            ;BSF    TRISA,5
            RETLW   0x01                    ; Terminó la función y regresamos con un "1" en WREG
;}


SLuz1       ;**************FUNCION DEL SENSOR DE LUZ CON EL ADC1***************
;******CONFIGURACION_ADC1
;{
            INCF    FSR1L                   ; Como el bit de verif del ADC está en el siguiente registro incrementamos el
apuntador
            BTFSC   POSTDEC1,1          ;verificamos el bit SEMADC1 para saber si está ocupado el periférico decrementamos el
apuntador para dejarlo igual
            RETLW   0x00
            BSF     TRISA,1                 ;Bit 0 del puerto A como entrada, AN0
            MOVLW   0x0B
            MOVWF   ADCON1                  ; Configuramos AN0-AN3 como analógicas
            MOVLW   0x14
            MOVWF   ADCON2                  ; Seleccionamos los tiempos de adquisición
            BCF     TRISE,0                 ;Bit 0 del puerto E como salida
            BCF     PORTE,0                 ;Ponemos a 0 el RE0
            BTFSC   VARLUZ,5                ;Verificamos si se prende el led del sensor si es 0 salta led OFF
            BSF     PORTE,0                 ;Se enciende el led ON
            INCF    FSR2L                   ; Como el bit SEMADC1 del ADC está en el siguiente registro incrementamos el
apuntador
```

```
            BSF     POSTDEC2,1              ;Avisamos que está ocupado el periférico ADC1
;}
;*******VERIFICACION DEL SLUZ 1************
;{
VERIFICACIONLUZ1
            BSF     ADCON0,2                ;Seleccionamos el canal AN0
            BSF     ADCON0,0                ;Habilitamos el modulo ADC
            BSF     ADCON0,GO               ;inicia la adquisición GO
            NOP                             ;10 microseg para realizar la conversión
            NOP
            NOP
            NOP
            NOP
            NOP
            NOP
            NOP
            NOP
            NOP
            MOVF    VALORLUZ1,W             ;cargamos el valor que dio el usuario en WREG
            BTFSC   VARLUZ,1                ;probamos el bit "1" de la variable VARLUZ, si es 1 será una operación > y si es
cero salta
            BRA     MAYORKLUZ1
MENORKLUZ1
            CPFSLT  ADRESH                  ; saltamos aquí si fue una operación < y comparamos la medición del ADC con
WREG
            BRA     SINTERMINARLUZ1         ; si no es menor saltamos a SINTERMINARLUZ1
            BRA     TERMINADOLUZ1           ; si es menor se cumple la condición y saltamos a TERMINADOLUZ1
MAYORKLUZ1
            CPFSGT  ADRESH                  ; saltamos aquí si fue una operación > y comparamos el valor medido por ADC
con WREG
SINTERMINARLUZ1
            RETLW   0x00                    ; No hubo salto no ha terminado la función y salimos

;}
;***********TERMINA ADC1***************
;{
TERMINADOLUZ1
            INCF    FSR2L                   ; Como el bit de SEMADC1 del ADC está en el siguiente registro incrementamos el
apuntador
            BSF     POSTDEC2,1              ;Liberamos el periférico ADC1 decrementamos el apuntador para dejarlo igual
            ;BCF    PORTE,0
            ;BSF    TRISE,0
            RETLW   0x01                    ; Terminó la función y regresamos con un "1" en WREG
;}


SLuz2       ;**************FUNCION DEL SENSOR DE LUZ CON EL ADC2***************
;******CONFIGURACION_ADC2
;{
            INCF    FSR1L                   ; Como el bit de verif del ADC está en el siguiente registro incrementamos el
apuntador
            BTFSC   POSTDEC1,2          ;verificamos el bit SEMADC2 para saber si está ocupado el periférico decrementamos el
apuntador para dejarlo igual
            RETLW   0x00
```

```
            BSF     TRISA,2                 ;Bit 3 del puerto A como entrada, AN2
            MOVLW   0x0B
            MOVWF   ADCON1                  ; Configuramos AN0-AN3 como analógicas
            MOVLW   0x14
            MOVWF   ADCON2                  ; Seleccionamos los tiempos de adquisición
            BCF     TRISE,1                 ;Bit 1 del puerto E como salida
            BCF     PORTE,1                 ;Ponemos a 0 el RE1
            BTFSC   VARLUZ,6                ;Verificamos si se prende el led del sensor si es 0 salta led OFF
            BSF     PORTE,1                 ;Se enciende el led ON
            INCF    FSR2L                   ; Como el bit SEMADC2 del ADC está en el siguiente registro incrementamos el
apuntador
            BSF     POSTDEC2,2              ;Avisamos que está ocupado el periférico ADC2 decrementamos el apuntador
para dejarlo igual
;}
;*******VERIFICACION************
;{
VERIFICACIONLUZ2
            BSF     ADCON0,3                ;Seleccionamos el canal AN2
            BSF     ADCON0,0                ;Habilitamos el modulo ADC
            BSF     ADCON0,GO               ;inicia la adquisición GO
            NOP                             ; 10 microseg para realizar la conversión
            NOP
            NOP
            NOP
            NOP
            NOP
            NOP
            NOP
            NOP
            NOP
            MOVF    VALORLUZ2,W             ;cargamos el valor que dio el usuario en WREG
            BTFSC   VARLUZ,2                ;probamos el bit "2" de la variable VARLUZ, si es 1 será una operación > y si es
cero salta
            BRA     MAYORKLUZ2
MENORKLUZ2
            CPFSLT  ADRESH                  ; saltamos aquí si fue una operación < y comparamos la medición del ADC con
WREG
            BRA     SINTERMINARLUZ2         ; si no es menor saltamos a SINTERMINARLUZ2
            BRA     TERMINADOLUZ2           ; si es menor se cumple la condición y saltamos a TERMINADOLUZ2
MAYORKLUZ2
            CPFSGT  ADRESH                  ; saltamos aquí si fue una operación > y comparamos el valor medido por ADC
con WREG
SINTERMINARLUZ2
            RETLW   0x00                    ; No hubo salto no ha terminado la función y salimos

;}
;***********TERMINAR***************
;{
TERMINADOLUZ2
            INCF    FSR2L                   ; Como el bit de SEMADC2 del ADC está en el siguiente registro incrementamos el
apuntador
            BSF     POSTDEC2,2              ;Liberamos el periférico ADC2
            ;BCF    PORTE,1
            ;BSF    TRISE,1
```

```
        RETLW   0x01                    ; Terminó la función y regresamos con un "1" en WREG
;}


SLuz3   ;**************FUNCION DEL SENSOR DE LUZ CON EL ADC3***************
;******CONFIGURACIONADC3
;{
        INCF    FSR1L                   ; Como el bit de verif del ADC está en el siguiente registro incrementamos el
apuntador
        BTFSC   POSTDEC1,3      ;verificamos el bit SEMADC3 para saber si está ocupado el periférico decrementamos el
apuntador para dejarlo igual
        RETLW   0x00
        BSF     TRISA,3                 ;Bit 3 del puerto A como entrada, AN3
        MOVLW   0x0B
        MOVWF   ADCON1                  ; Configuramos AN0-AN3 como analógicas
        MOVLW   0x14
        MOVWF   ADCON2                  ; Seleccionamos los tiempos de adquisición
        BCF     TRISE,2                 ;Bit 2 del puerto E como salida
        BCF     PORTE,2                 ;Ponemos a 0 el RE2
        BTFSC   VARLUZ,7                ;Verificamos si se prende el led del sensor si es 0 salta led OFF
        BSF     PORTE,2                 ;Se enciende el led ON
        INCF    FSR2L                   ; Como el bit SEMADC3 del ADC está en el siguiente registro incrementamos el
apuntador
        BSF     POSTDEC2,3              ;Avisamos que está ocupado el periférico ADC3 decrementamos el apuntador
para dejarlo igual
;}

;*******VERIFICACION ADC3************
;{
VERIFICACIONLUZ3
        BSF     ADCON0,2                ;Seleccionamos el canal AN3
        BSF     ADCON0,3
        BSF     ADCON0,0                ;Habilitamos el modulo ADC
        BSF     ADCON0,GO               ;inicia la adquisición GO
        NOP                             ;10 microseg para realizar la conversión
        NOP
        NOP
        NOP
        NOP
        NOP
        NOP
        NOP
        NOP
        NOP
        MOVF    VALORLUZ3,W             ;cargamos el valor que dio el usuario en WREG
        BTFSC   VARLUZ,3                ;probamos el bit "3" de la variable VARLUZ, si es 1 será una operación > y si es
cero salta
        BRA     MAYORKLUZ3
MENORKLUZ3
        CPFSLT  ADRESH                  ; saltamos aquí si fue una operación < y comparamos la medición del ADC con
WREG
        BRA     SINTERMINARLUZ3         ; si no es menor saltamos a SINTERMINARLUZ3
        BRA     TERMINADOLUZ3           ; si es menor se cumple la condición y saltamos a TERMINADOLUZ3
```

```
MAYORKLUZ3
        CPFSGT  ADRESH                  ; saltamos aquí si fue una operación > y comparamos el valor medido por ADC
con WREG
SINTERMINARLUZ3
        RETLW   0x00                    ; No hubo salto no ha terminado la función y salimos

;}

;***********TERMINAR***************
;{
TERMINADOLUZ3
        INCF    FSR2L                   ; Como el bit de SEMADC3 del ADC está en el siguiente registro incrementamos el
apuntador
        BSF     POSTDEC2,3              ;Liberamos el periférico ADC3
        RETLW   0x01                    ; Terminó la función y regresamos con un "1" en WREG
;}
```

; Archivo tipo .INC que contiene las funciones del sensor de SONIDO

```
#define SON0MAYORA      BSF     VARSON,0
#define SON0MENORA      BCF     VARSON,0
#define SON1MAYORA      BSF     VARSON,1
#define SON1MENORA      BCF     VARSON,1
#define SON2MAYORA      BSF     VARSON,2
#define SON2MENORA      BCF     VARSON,2
#define SON3MAYORA      BSF     VARSON,3
#define SON3MENORA      BCF     VARSON,3

Sonido0 ;**************FUNCION DEL SENSOR DE SONIDO CON EL ADC0***************
;******CONFIGURACION_ADC0
;{
        INCF    FSR1L           ; Como el bit de verif del ADC está en el siguiente registro incrementamos el apuntador
        BTFSC   POSTDEC1,0      ;verificamos el bit SEMADC0 para saber si está ocupado el periférico decrementamos el
apuntador para dejarlo igual
        RETLW   0x00
        BSF     TRISA,0                 ;Bit 0 del puerto A como entrada, AN0
        MOVLW   0x0B
        MOVWF   ADCON1                  ; Configuramos AN0-AN3 como analógicas
        MOVLW   0x14
        MOVWF   ADCON2                  ; Seleccionamos los tiempos de adquisición
        INCF    FSR2L                   ; Como el bit SEMADC0 del ADC está en el siguiente registro incrementamos el
apuntador
        BSF     POSTDEC2,0              ;Avisamos que está ocupado el periférico ADC0
;}
;*******VERIFICACION DEL Sonido0************
;{
VERIFICACIONSON0
        CLRF    ADCON0                  ; Seleccionamos el canal AN0
        BSF     ADCON0,0                ;Habilitamos el modulo ADC
```

```
            BSF     ADCON0,GO               ;inicia la adquisición GO
            NOP                             ; 10 microseg para realizar la conversión
            NOP
            NOP
            NOP
            NOP
            NOP
            NOP
            NOP
            NOP
            NOP
            MOVF    VALORSON0,W             ;cargamos el valor que dio el usuario en WREG
            BTFSC   VARSON,0                ;probamos el bit "0" de la variable VARSON, si es 1 será una operación > y si es
cero salta
            BRA     MAYORSON0
MENORKSON0
            CPFSLT  ADRESH                  ; saltamos aquí si fue una operación < y comparamos la medición del ADC con
WREG
            BRA     SINTERMINARSON0         ; si no es menor saltamos a SINTERMINAR0
            BRA     TERMINADOSON0           ; si es menor se cumple la condición y saltamos a TERMINADOSON0
MAYORSON0
            CPFSGT  ADRESH                  ; saltamos aquí si fue una operación > y comparamos el valor medido por ADC
con WREG
SINTERMINARSON0
            RETLW   0x00                    ; No hubo salto no ha terminado la función y salimos

;}
;***********TERMINAR EL LUZO***************
;{
TERMINADOSON0                               ; salto o llego por bra entonces la condición se cumplió

            INCF    FSR2L                   ; Como el bit de SEMADC0 del ADC esta en el siguiente registro incrementamos el
apuntador
            BSF     POSTDEC2,0              ;Liberamos el periférico ADC0
            RETLW   0x01                    ; Termino la función y regresamos con un "1" en WREG
;}


Sonido1     ;**************FUNCION DEL SENSOR DE LUZ CON EL ADC1***************
;******CONFIGURACION_ADC1
;{
            INCF    FSR1L                   ; Como el bit de verif del ADC está en el siguiente registro incrementamos el
apuntador
            BTFSC   POSTDEC1,1          ;verificamos el bit SEMADC1 para saber si está ocupado el periférico decrementamos el
apuntador para dejarlo igual
            RETLW   0x00
            BSF     TRISA,1                 ;Bit 0 del puerto A como entrada, AN0
            MOVLW   0x0B
            MOVWF   ADCON1                  ; Configuramos AN0-AN3 como analógicas
            MOVLW   0x14
            MOVWF   ADCON2                  ; Seleccionamos los tiempos de adquisición
            INCF    FSR2L                   ; Como el bit SEMADC1 del ADC está en el siguiente registro incrementamos el
apuntador
            BSF     POSTDEC2,1              ;Avisamos que está ocupado el periférico ADC1
```

```
;}
;*******VERIFICACION DEL Sonido 1************
;{
VERIFICACIONSON1
        BSF     ADCON0,2                ;Seleccionamos el canal AN0
        BSF     ADCON0,0                ;Habilitamos el modulo ADC
        BSF     ADCON0,GO               ;inicia la adquisición GO
        NOP                             ;10 microseg para realizar la conversión
        NOP
        NOP
        NOP
        NOP
        NOP
        NOP
        NOP
        NOP
        NOP
        MOVF    VALORSON1,W             ;cargamos el valor que dio el usuario en WREG
        BTFSC   VARSON,1                ;probamos el bit "1" de la variable VARSON, si es 1 será una operación > y si es
cero salta
        BRA     MAYORSON1
MENORKSON1
        CPFSLT  ADRESH                  ; saltamos aquí si fue una operación < y comparamos la medición del ADC con
WREG
        BRA     SINTERMINARSON1         ; si no es menor saltamos a SINTERMINARSON1
        BRA     TERMINADOSON1           ; si es menor se cumple la condición y saltamos a TERMINADOSON1
MAYORSON1
        CPFSGT  ADRESH                  ; saltamos aquí si fue una operación > y comparamos el valor medido por ADC
con WREG
SINTERMINARSON1
        RETLW   0x00                    ; No hubo salto no ha terminado la función y salimos

;}
;***********TERMINA ADC1***************
;{
TERMINADOSON1
        INCF    FSR2L                   ; Como el bit de SEMADC1 del ADC está en el siguiente registro incrementamos el
apuntador
        BSF     POSTDEC2,1              ;Liberamos el periférico ADC1 decrementamos el apuntador para dejarlo igual
        RETLW   0x01                    ; Termino la función y regresamos con un "1" en WREG
;}


Sonido2 ;**************FUNCION DEL SENSOR DE LUZ CON EL ADC2***************
;******CONFIGURACION_ADC2
;{
        INCF    FSR1L                   ; Como el bit de verif del ADC está en el siguiente registro incrementamos el
apuntador
        BTFSC   POSTDEC1,2      ;verificamos el bit SEMADC2 para saber si está ocupado el periférico decrementamos el
apuntador para dejarlo igual
        RETLW   0x00
        BSF     TRISA,2                 ;Bit 3 del puerto A como entrada, AN2
        MOVLW   0x0B
        MOVWF   ADCON1                  ; Configuramos AN0-AN3 como analógicas
```

```
            MOVLW   0x14
            MOVWF   ADCON2              ; Seleccionamos los tiempos de adquisición
            INCF    FSR2L               ; Como el bit SEMADC2 del ADC está en el siguiente registro incrementamos el
apuntador
            BSF     POSTDEC2,2          ;Avisamos que está ocupado el periférico ADC2 decrementamos el apuntador
para dejarlo igual
;}
;*******VERIFICACION************
;{
VERIFICACIONSON2
            BSF     ADCON0,3            ;Seleccionamos el canal AN2
            BSF     ADCON0,0            ;Habilitamos el modulo ADC
            BSF     ADCON0,GO           ;inicia la adquisición GO
            NOP                         ;10 microseg para realizar la conversión
            NOP
            NOP
            NOP
            NOP
            NOP
            NOP
            NOP
            NOP
            NOP
            MOVF    VALORSON2,W         ;cargamos el valor que dio el usuario en WREG
            BTFSC   VARSON,2            ;probamos el bit "2" de la variable VARSON, si es 1 será una operación > y si es
cero salta
            BRA     MAYORSON2
MENORKSON2
            CPFSLT  ADRESH              ; saltamos aquí si fue una operación < y comparamos la medición del ADC con
WREG
            BRA     SINTERMINARSON2     ; si no es menor saltamos a SINTERMINARSON2
            BRA     TERMINADOSON2       ; si es menor se cumple la condición y saltamos a TERMINADOSON2
MAYORSON2
            CPFSGT  ADRESH              ; saltamos aqui si fue una operación > y comparamos el valor medido por ADC
con WREG
SINTERMINARSON2
            RETLW   0x00                ; No hubo salto no ha terminado la función y salimos

;}
;***********TERMINAR***************
;{
TERMINADOSON2
            INCF    FSR2L               ; Como el bit de SEMADC2 del ADC está en el siguiente registro incrementamos el
apuntador
            BSF     POSTDEC2,2          ;Liberamos el periférico ADC2
            RETLW   0x01                ; Termino la función y regresamos con un "1" en WREG
;}



Sonido3     ;**************FUNCION DEL SENSOR DE LUZ CON EL ADC3***************
;******CONFIGURACIONADC3
;{
```

```
        INCF    FSR1L                   ; Como el bit de verif del ADC está en el siguiente registro incrementamos el
apuntador
        BTFSC   POSTDEC1,3      ;verificamos el bit SEMADC3 para saber si está ocupado el periférico decrementamos el
apuntador para dejarlo igual
        RETLW   0x00
        BSF     TRISA,3                 ;Bit 3 del puerto A como entrada, AN3
        MOVLW   0x0B
        MOVWF   ADCON1                  ; Configuramos AN0-AN3 como analógicas
        MOVLW   0x14
        MOVWF   ADCON2                  ; Seleccionamos los tiempos de adquisición
        INCF    FSR2L                   ; Como el bit SEMADC3 del ADC está en el siguiente registro incrementamos el
apuntador
        BSF     POSTDEC2,3              ;Avisamos que está ocupado el periférico ADC3 decrementamos el apuntador
para dejarlo igual
;}

;*******VERIFICACION ADC3************
;{
VERIFICACIONSON3
        BSF     ADCON0,2                ;Seleccionamos el canal AN3
        BSF     ADCON0,3
        BSF     ADCON0,0                ;Habilitamos el modulo ADC
        BSF     ADCON0,GO               ;inicia la adquisición GO
        NOP                             ;10 microseg para realizar la conversión
        NOP
        NOP
        NOP
        NOP
        NOP
        NOP
        NOP
        NOP
        NOP
        MOVF    VALORSON3,W             ;cargamos el valor que dio el usuario en WREG
        BTFSC   VARSON,3                ;probamos el bit "3" de la variable VARSON, si es 1 será una operación > y si es
cero salta
        BRA     MAYORSON3
MENORKSON3
        CPFSLT  ADRESH                  ; saltamos aquí si fue una operación < y comparamos la medición del ADC con
WREG
        BRA     SINTERMINARSON3         ; si no es menor saltamos a SINTERMINARSON3
        BRA     TERMINADOSON3           ; si es menor se cumple la condición y saltamos a TERMINADOSON3
MAYORSON3
        CPFSGT  ADRESH                  ; saltamos aquí si fue una operación > y comparamos el valor medido por ADC
con WREG
SINTERMINARSON3
        RETLW   0x00                    ; No hubo salto no ha terminado la función y salimos

;}

;***********TERMINAR***************
;{
TERMINADOSON3
```

```
        INCF    FSR2L                   ; Como el bit de SEMADC3 del ADC esta en el siguiente registro incrementamos el
apuntador
        BSF     POSTDEC2,3              ;Liberamos el periférico ADC3
        RETLW   0x01                    ; Termino la función y regresamos con un "1" en WREG
;}
```

; Archivo que contiene la función para el control del sensor de toque

```
#define oprimir0        BSF     CONTSTOQUE,0
#define liberar0        BCF     CONTSTOQUE,0
#define oprimir1        BSF     CONTSTOQUE,1
#define liberar1        BCF     CONTSTOQUE,1
#define oprimir2        BSF     CONTSTOQUE,2
#define liberar2        BCF     CONTSTOQUE,2
#define oprimir3        BSF     CONTSTOQUE,3
#define liberar3        BCF     CONTSTOQUE,3

Stoque0   ;**************Función del sensor de toque******************
;****************CONFIGURACION DEL STOQUE0 *********
        INCF    FSR1L                   ; Como el bit de verif  está en el siguiente registro incrementamos el apuntador
        BTFSC   POSTDEC1,4      ;verificamos el bit SEMSTOK0 para saber si está ocupado el periférico decrementamos el
apuntador para dejarlo igual
        RETLW   0x00
        BSF     TRISA,5                 ;Bit 5 del puerto A como entrada para registrar el estado del sensor de toque
        INCF    FSR2L                   ; Como el bit SEMSTOK0  está en el siguiente registro incrementamos el
apuntador
        BSF     POSTDEC2,4              ;Avisamos que está ocupado el periférico Sensor de toque0


;***************VERIFICACION Y TERMINO DEL Stoque0****************
VERIFSTOQUE0

        BTFSC   CONTSTOQUE,4            ;revisamos si se está esperando la condición de soltar si está en 1 no brinca y se
va a la etiqueta SOLTAR
        BRA     SOLTAR0
        BTFSS   PORTA,5                 ;revisamos si el sensor de toque se activo
        RETURN                          ; no se activo salimos del call
        BTFSS   CONTSTOQUE,0            ;el sensor se activo verificamos en el bit0 que función se programo
        BRA     SOLTAR0
TERMSTOQUE0                             ; Se cumplió la condición programada y se termina la función


        INCF    FSR2L                   ; Como el bit de SEMSTOK0  está en el siguiente registro incrementamos el
apuntador
        BSF     POSTDEC2,4              ;Liberamos el periférico SENSOR DE TOQUE0
        RETLW   0x01                    ; y salimos del call
SOLTAR0                                 ; código para cuando se programo la función de soltar el sensor de botón
        BSF     CONTSTOQUE,4            ;Activamos el bit 4 del reg de control para indicar que se está esperando la acción
de soltar el botón
```

```
        BTFSC   PORTA,5                 ;revisamos si el sensor de toque se soltó
        RETLW   0x00                    ; No ha terminado la condición y nos salimos del call
        BCF     CONTSTOQUE,4            ;Desactivamos el bit de soltar
        INCF    FSR2L                   ; Como el bit de SEMSTOK0 está en el siguiente registro incrementamos el
apuntador
        BSF     POSTDEC2,4              ;Liberamos el periférico SENSOR DE TOQUE0
        RETLW   0x01                    ;y salimos del call FIN DE LA FUNCIÓN


Stoque1 ;**************Función del sensor de toque 1******************
;****************CONFIGURACION DEL STOQUE1 *********
        INCF    FSR1L                   ; Como el bit de verif  está en el siguiente registro incrementamos el apuntador
        BTFSC   POSTDEC1,5      ;verificamos el bit SEMSTOK1 para saber si está ocupado el periférico decrementamos el
apuntador para dejarlo igual
        RETLW   0x00
        BSF     TRISE,0                 ;Bit 0 del puerto E como entrada para registrar el estado del sensor de toque
        INCF    FSR2L                   ; Como el bit SEMSTOK1  está en el siguiente registro incrementamos el
apuntador
        BSF     POSTDEC2,5              ;Avisamos que está ocupado el periférico Sensor de toque 1

;***************VERIFICACION Y TERMINO DEL Stoque1****************
VERIFSTOQUE1
        BTFSC   CONTSTOQUE,5            revisamos si se está esperando la condición de soltar si está en 1 no brinca y se
va a la etiqueta SOLTAR
        BRA     SOLTAR1
        BTFSS   PORTE,0                 ;revisamos si el sensor de toque se activo
        RETLW   0x00                    ; no se activo salimos del call
        BTFSS   CONTSTOQUE,1            ;el sensor se activo verificamos en el bit0 que función se programó
        BRA     SOLTAR1
TERMSTOQUE1                             ; Se cumplió la condición programada y se termina la función
        INCF    FSR2L                   ; Como el bit de SEMSTOK1  está en el siguiente registro incrementamos el
apuntador
        BSF     POSTDEC2,5              ;Liberamos el periférico SENSOR DE TOQUE1
        RETLW   0x01                    ; y salimos del call

SOLTAR1                                 ; código para cuando se programo la función de soltar el sensor de botón
        BSF     CONTSTOQUE,5            ;Activamos el bit 4 del reg de control para indicar que se está esperando la acción
de soltar el botón
        BTFSC   PORTE,0                 ;revisamos si el sensor de toque se soltó
        RETLW   0x00                    ; No ha terminado la condición y nos salimos del call
        BCF     CONTSTOQUE,5            ;Desactivamos el bit de soltar
        INCF    FSR2L                   ; Como el bit de SEMTOK1 está en el siguiente registro incrementamos el
apuntador
        BSF     POSTDEC2,5              ;Liberamos el periférico SENSOR DE TOQUE 1
        RETLW   0x01                    ; y salimos del call FIN DE LA FUNCIÓN


Stoque2 ;**************Función del sensor de toque 2******************
;****************CONFIGURACION DEL STOQUE2 *********
        INCF    FSR1L                   ; Como el bit de verif  está en el siguiente registro incrementamos el apuntador
```

```
        BTFSC   POSTDEC1,6      ;verificamos el bit SEMSTOK2 para saber si está ocupado el periférico decrementamos el
apuntador para dejarlo igual            RETLW   0x00
        BSF     TRISE,1                 ;Bit 0 del puerto E como entrada para registrar el estado del sensor de toque
        INCF    FSR2L                   ; Como el bit SEMSTOK2  está en el siguiente registro incrementamos el
apuntador
        BSF     POSTDEC2,6              ;Avisamos que está ocupado el periférico Sensor de toque0
;***************VERIFICACION Y TERMINO DEL Stoque2****************
VERIFSTOQUE2
        BTFSC   CONTSTOQUE,6            ;revisamos si se está esperando la condición de soltar si está en 1 no brinca y se
va a la etiqueta SOLTAR
        BRA     SOLTAR2
        BTFSS   PORTE,1                 ;revisamos si el sensor de toque se activo
        RETURN                          ; no se activo salimos del call
        BTFSS   CONTSTOQUE,2            ;el sensor se activo verificamos en el bit0 que función se programó
        BRA     SOLTAR2
TERMSTOQUE2                             ; Se cumplió la condición programada y se termina la función

        INCF    FSR2L                   ; Como el bit de SEMSTOK2  está en el siguiente registro incrementamos el
apuntador
        BSF     POSTDEC2,6              ;Liberamos el periférico SENSOR DE TOQUE2
        RETLW   0x01                    ; y salimos del call
SOLTAR2                                 ; código para cuando se programo la función de soltar el sensor de botón
        BSF     CONTSTOQUE,6            ;Activamos el bit 4 del reg de control para indicar que se está esperando la acción
de soltar el botón
        BTFSC   PORTE,1                 ;revisamos si el sensor de toque se soltó
        RETLW   0x00                    ; No ha terminado la condición y nos salimos del call
        BCF     CONTSTOQUE,6            ;Desactivamos el bit de soltar
        INCF    FSR2L                   ; Como el bit de SEMTOK1 está en el siguiente registro incrementamos el
apuntador
        BSF     POSTDEC2,6              ;Liberamos el periférico SENSOR DE TOQUE 2
        RETLW   0x01                    ;y salimos del call FIN DE LA FUNCIÓN



Stoque3 ;**************Función del sensor de toque 3******************
;***************CONFIGURACION DEL STOQUE3 *********
        INCF    FSR1L                   ; Como el bit de verif  está en el siguiente registro incrementamos el apuntador
        BTFSC   POSTDEC1,7      ;verificamos el bit SEMSTOK3 para saber si está ocupado el periférico decrementamos el
apuntador para dejarlo igual    RETLW   0x00
        BSF     TRISE,2                 ;Bit 2 del puerto E como entrada para registrar el estado del sensor de toque
        INCF    FSR2L                   ; Como el bit SEMSTOK3  está en el siguiente registro incrementamos el
apuntador
        BSF     POSTDEC2,7              ;Avisamos que está ocupado el periférico Sensor de toque0

;***************VERIFICACION Y TERMINO DEL Stoque3****************
VERIFSTOQUE3
        BTFSC   CONTSTOQUE,7            ;revisamos si se está esperando la condición de soltar si está en 1 no brinca y se
va a la etiqueta SOLTAR
        BRA     SOLTAR3
        BTFSS   PORTE,2                 ;revisamos si el sensor de toque se activo
        RETURN                          ; no se activo salimos del call
        BTFSS   CONTSTOQUE,3            ;el sensor se activo verificamos en el bit0 que función se programo
        BRA     SOLTAR3
```

```
TERMSTOQUE3                             ; Se cumplió la condición programada y se termina la función

        INCF    FSR2L                   ; Como el bit de SEMSTOK3  está en el siguiente registro incrementamos el
apuntador
        BSF     POSTDEC2,7              ;Liberamos el periférico SENSOR DE TOQUE3
        RETLW   0x01                    ; y salimos del call
SOLTAR3                                 ; código para cuando se programo la función de soltar el sensor de botón
        BSF     CONTSTOQUE,7            ;Activamos el bit 4 del reg de control para indicar que se está esperando la acción
de soltar el botón
        BTFSC   PORTE,2                 ;revisamos si el sensor de toque se solto
        RETLW   0x00                    ; No ha terminado la condición y nos salimos del call
        BCF     CONTSTOQUE,7            ;Desactivamos el bit de soltar
        INCF    FSR2L                   ; Como el bit de SEMTOK3 está en el siguiente registro incrementamos el
apuntador
        BSF     POSTDEC2,7              ;Liberamos el periférico SENSOR DE TOQUE 3
        RETLW   0x01                    ; y salimos del call FIN DE LA FUNCIÓN


;PROGRAMA PARA GENERAR UN PULSO DE 40 khz, envia
;una rafaga de 8 ciclos del pulso por medio del sensor ultrasonico y recibe el pulso del eco por la RA1
;que es la entrada (-) del comparador 1 y si hay un cambio provoca una interrupcion
;con lo que se para el timer1 para saber el tiempo que tardo en regresar la señal y calcular la distancia (28-07-2010)y se anexara
;la comunicacin I2C para enviar el dato por el bus i2c en cuanto se lo pida el dispositivo maestro


        LIST P=16F690
        INCLUDE "P16F690.INC"

        __CONFIG        _INTRC_OSC_NOCLKOUT & _WDT_OFF & _MCLRE_OFF;Oscilador interno y RA4 y RA5 como pines de
I/O y APAGAMO EL WATCH DOG


                ;*************************DEFINICION DE VARIABLES*****************************************
        CBLOCK  0x20
        BKW                             ; RESPALDO DE W
        BKSTATUS                        ; RESPALDO DEL REGISTRO STATUS
        TEMP                            ; VARIABLE PARA LOS PROCESOS I2C
        DISTANCIAH                      ; VARIABLE PARA GUARDAR LA DISTANCIA CALCULADA HIGH
        DISTANCIAL                      ; VARIABLE PARA GUARDAR LA DISTANCIA CALCULADA LOW
        CONTA                           ; VARIABLE QUE NOS AYUDA A OBTENER EL PERIODO DE CADA CICLO DE LA
RAFAGA
        AUX                             ; VARIABLE PARA CONTAR LOS 8 PULSOS DE LA RAFAGA

        ENDC                            ; FIN DE DEFINICIONES

        ORG   0
        GOTO   INICIO

                ;***********************************************RUTINA DE
INTRRUPCION************************************
        ORG     0x04
                                        ; SE RESPALDA EL REGISTRO W Y EL REGISTRO STATUS
        MOVWF   BKW                     ; REALIZAMOS COPIA DE W
        MOVF    STATUS,W
```

```
        BCF     STATUS,RP0              ;NOS VAMOS AL BANCO 0
        BCF     STATUS,RP1
        MOVWF   BKSTATUS                ; REALIZAMOS COPIA
                                ; REVISAMOS SI LA INTERRUPCION ES POR UN EVENTO I2C Y SI ES AFIRMATIVO LLAMAMOS
A LA SUBRUTINA RESPECTIVA
        BTFSS   PIR1,SSPIF              ;HAY UNA INTERRUPCION POR EL MODULO SSP(I2C)?
        GOTO    REBOTE                  ; NO, ATENDEMOS LA INTERRUPCION DEL REBOTE DEL ECO DEL SENSOR SONICO
        CALL    EVENI2C                 ; SI, LLAMAMOS A LA SUBRUTINA PARA TRATAR LAS INTERRUPCIONES DEL I2C

        BCF     STATUS,RP0              ;REGRESAMOS DEL CALL Y NOS VAMOS AL BANCO 0
        BCF     STATUS,RP1
        BCF     PIR1,SSPIF              ;PARA PONER A "0" LA BANDERA SSPIF CAUSANTE DE LA INTERRUPCION
        GOTO    REST                    ; NOS PASAMOS A REESTABLECER LOS REG W Y ESTATUS


                ;*********AQUI PONEMOS LA SUBRUTINA PARA LA OTRA INTERRUPCION QUE VA A SER LA DEL
COMPARADOR********
REBOTE  BCF     STATUS,RP0              ;NOS VAMOS AL BANCO 0
        BCF     STATUS,RP1
        BCF     T1CON,TMR1ON            ;DESACTIVAMOS el TIMER1


        MOVFW   TMR1H
        MOVWF   DISTANCIAH              ; LO PASAMOS A LA VARIABLE DISTANCIA PARA QUE SEA ENVIADO POR I2C
        MOVFW   TMR1L
        MOVWF   DISTANCIAL
        BCF     PORTC,5                 ;PONEMOS A CERO RC5 QUE ES LA SALDIA DEL TIEMPO
        BCF     PIR1,TMR1IF             ;PONEMOS A CERO LA FLAG DEL TMR1
        BCF     PIR2,C1IF               ;PONER A CERO LA BANDERA DE INTERR DEL COMPARADOR 1

REST    ;********************REESTABLECEMOS LOS VALORE DE W Y DE STATUS**************************
        MOVF    BKSTATUS,W
        MOVWF   STATUS                  ; SE RECUPERA EL VALOR QUE TENIA STATUS
        MOVF    BKW                     ; AHORA EL QUE TENIA W
        RETFIE                          ; SALIMOS DE LA INTERRUPCION

        ; ************************************INICIO Y CONFIGURACION DE PUERTOS,COMPARADOR Y DEL
I2C*****************************


inicio  BSF     STATUS,RP1              ;---------BANCO 2---------------------------------
        BANKSEL CM1CON0
        MOVLW   0xA4
        MOVWF   CM1CON0                 ; CONFIGURAMOS EL COMPARADOR 1 CON LA REF INTERNA
        MOVLW   0X03
        MOVWF   ANSEL                   ; CONFIGURAMOS RA0,RA1 COMO ANALOGICAS Y LAS DEMAS COMO DIGITALES
        CLRF    ANSELH
        MOVLW   0XAA
        MOVWF   VRCON                   ; AJUSTAMOS LA REF DE VOLT A 2.000 segun tabla VOLTS BAJA

        BCF     STATUS,RP1
        BSF     STATUS,RP0              ;----------CAMBIAMOS AL BANCO 1-----------
        MOVLW   0XFF
        MOVWF   TRISB                   ; PUERTO B COMO ENTRADA PARA QUE TRABAJE BIEN EL I2C
```

```
        BCF     SSPSTAT,CKE             ;DEBE ESTAR EN CERO PARA EL I2C
        BCF     SSPSTAT,SMP             ;DEBE ESTAR EN CERO PARA EL I2C
        MOVLW   0x14
        MOVWF   SSPADD                  ; LA DAMOS LA DIRECCION "A" A ESTE PIC
        MOVLW   0x67
        MOVWF   OSCCON                  ; SELECCIONAMOS EL OSCILADOR INTERNO Y QUE TRABAJE A 4 MHZ

        BSF     TRISA,1                 ;CONFIGURAMOS RA1 COMO ENTRADA -  PARA EL COMPARADOR 1
        BCF     TRISA,2                 ;CONFIGURAMOS RA2 COMO SALIDA QUE ES LA DEL COMPARADOR 1
        BCF     TRISA,4                 ;CONFIGURAMOS RA4 COMO SALIDA DEL FREC DE 40 AL EMISOR DEL SONICO
        BCF     TRISA,5                 ;CONFIGURAMOS RA5 COMO COMPLEMENTO DE RA4
                                        ; CONFIGURAMOS PUERTO RC5 COMO SALIDA AL SENSOR SONICO
        BCF     TRISC,5                 ;RC5 COMO SALIDA DE LA SEÑAL DE TIEMPO
        MOVLW   0XD7                    ; CONFIGURAMOS EL TIMER0 COMO TEMPORIZADOR CON UN
        MOVWF   OPTION_REG              ; PREDIVISOR A 256
        CLRF    INTCON                  ; INHABILITAR TODAS LAS INTERRUPCIONES
        BSF     INTCON,PEIE             ;SE HABILITA LA INTERRUPCION DE LOS PERIFERICOS
        ;BSF    INTCON,T0IE             ;HABILITAMOS LA INTERRUPCION POR DEL TIMER 0
        BSF     INTCON,GIE              ;HABILITAMOS LAS INTERRUPCIONES EN GENERAL
        BSF     PIE2,C1IE               ;HABILITAMOS LA INTERR DEL COMPARADOR 1
        BSF     PIE1,SSPIE              ;HABILITAMOS LA INTERRUPCION POR SSP
        BSF     PIE1,TMR1IE             ;HABILITAMOS LA INTERRUPCION DEL TIMER 1
        BCF     STATUS,RP0              ;-------------------------BANCO 0---------------------------------------
        BCF     PIR1,TMR1IF             ;PONER 0 EN EL INDICADOR DE DESBORDAMIENTO DEL TIMER1
        ;BCF    INTCON,T0IF             ;PONER 0 EN EL INDICADOR DE DESBORDAMIENTO DEL TIMER0
        CLRF    SSPBUF                  ; PONEMOS A CEROS EL REGISTRO SSPBUF
        MOVLW   0x36
        MOVWF   SSPCON                  ; SE CONFIGURA COMO I2C Y EN MODO ESCLAVO
        BCF     PIR1,SSPIE              ;A CERO LA BANDERA DE INTERRUPCION DE SSP
        MOVLW   0x01
        MOVWF   T1CON                   ; CONFIGURAMOS EL TIMER 1 A 1:2
        ;BSF    T1CON,0                 ;ACTIVAMOS EL TIMER1
        BCF     PIR1,TMR1IF             ;PONEMOS A CERO LA FLAG DEL TMR1

        ;*****************************PROGRAMA PRINCIPAL*****************************************
        CLRF    DISTANCIAL
        CLRF    DISTANCIAH

inicio2
        MOVLW   0X09                    ; CARGAMOS LA VARIBLE AUX CON EL NUMERO DE PULSOS DE LA RAFAGA N-1
        MOVWF   AUX

PULSO   BCF     PORTA,4                 ;RA4 Y RA5 VAN CONECTADOR AL EMISOR SONICO
        BSF     PORTA,5                 ;PONEMOS A 1 EL RA5 INICIA LA FRECUENCIA DE 40KHZ
        MOVLW   3H
        MOVWF   CONTA
PULSO1  DECFSZ  CONTA,F                 ;DECREMENTAMOS CONTA Y SALTA SI ES CERO
        GOTO    PULSO1
        BSF     PORTA,4
        BCF     PORTA,5                 ;PONEMOS A 0 EL RA0
        MOVLW   2H
        MOVWF   CONTA
PULSO2  DECFSZ  CONTA,F                 ;DECREMENTAMOS CONTA Y SALTA SI ES CERO
        GOTO    PULSO2
```

```
        DECFSZ  AUX,F                   ;DECREMENTA AUX Y SALTA SI ES CERO CON AUX GENERAMOS LOS 8 PULSOS
QUE LLEVA LA RAFAGA DE 40KH
        GOTO    PULSO                   ; CON ESTE CODIGO OBTENEMOS UNA FRECUENCIA DE 40KHZ
        BCF     PORTA,4                 ;RA4 A CERO PARA APAGAR EL EMISOR SONICO
        BSF     PORTC,5                 ;PONEMOS A UNO RC5 QUE MEDIRA EL TIEMPO AL REBOTAR LA SEÑAL SE IRA A
CERO

        CLRF    TMR1H                   ; PONER 0 EN EL REGISTRO TMR1H PARA QUE INICIE LA CUENTA DEL TIEMPO


        CLRF    TMR1L                   ; TAMBIEN A CERO EL REGISTRO TMR1L
        BSF     T1CON,0                 ;ACTIVAMOS EL TIMER1 QUE INICIE SU CUENTA PARA CALCULAR LA DISTANCIA
        CALL    RETARDO
        BCF     PORTC,5                 ;PONEMOS A CERO

        GOTO    INICIO2

                                ;*********************
SUBRUTINAS*************************************************************

EVENI2C                 ; MANEJO DE LOS EVENTOS I2C DEL MODULO SSP Y HAY QUE CHECAR EN QUE ESTADO
                        ; SE ENCUENTRA EL REGISTRO SSPSTAT ES DE LOS CINCO POSIBLE Y SI ES EL
                        ; INDICADO PARA TRANSMITIR EL DATO SEGIN DATA SHEET
                        ; STATE 1: OPERACIÓN DE ESCRITURA I2C, ULTIMO BYTE ERA DE DIRECCIÓN.
                         ; SSPSTAT BITS: S = 1, R/W = 0, D/A = 0, BF = 1
                        ; STATE 2: OPERACIÓN DE ESCRITURA I2C, ULTIMO BYTE ERA DE DATOS.
                        ; SSPSTAT BITS: S = 1, R/W = 0, D/A = 1, BF = 1
                        ; STATE 3: OPERACIÓN DE LECTURA I2C, ULTIMO BYTE ERA DE DIRECCIÓN.
                        ; SSPSTAT BITS: S = 1, R/W = 1, D/A = 0, BF = 0
                        ; STATE 4: OPERACIÓN DE LECTURA I2C, ULTIMO BYTE ERA DE DATOS.
                        ; SSPSTAT BITS: S = 1, R/W = 1, D/A = 1, BF = 0
                        ; STATE 5: RESET LÓGICO DEL SLAVE I2C POR NACK DEL MASTER.
                        ; SSPSTAT BITS: S = 1, R/W= 0, D/A = 1, BF = 0

        BSF     STATUS,RP0              ;SELECCIONAMOS EL BANCO 1
        BCF     STATUS,RP1
        MOVF    SSPSTAT,W               ;PASAMOS EL VALOR DEL REGISTRO SSPSTAT AL REG W
        ANDLW   b'00101101'             ; SOLO TOMAMOS LOS BITS QUE NOS INTERESAN S,R/W,D/A,BF
        BCF     STATUS,RP0              ;NOS PASAMOS AL BANCO 0
        MOVWF   TEMP                    ; PASAMOS LOS BITS AL REG TEMP PARA COMPARARLOS MAS ADELANTE

STATE3:                                 ;OPERACION DE LECTURA, EL ULTIMO BYTE FUE DE DIRECCION
        MOVLW   b'00001100'             ; Y EL BUFER ESTA VACIO
        BCF     STATUS,RP0
        BCF     STATUS,RP1              ;SELECCIONAMOS EL BANCO 0
        XORWF   TEMP,W
        BTFSS   STATUS,Z                ;ESTAMOS EN EL TERCER ESTADO?
        GOTO    STATE4                  ; NO, NOS VAMOS AL ESTADO 5
        CALL    WRITEI2CL               ; Y LO ESCRIBIMOS EN EL REGISTRO SSPBUF
        RETURN


STATE4:                                 ;EL ESTADO 4 ES UNA OPERACION DE LECTURA DE UN SEGUNDO O MAS DATO
        MOVLW   b'00101100'
```

```
        BCF     STATUS,RP0
        BCF     STATUS,RP1              ;SELECCIONAMOS EL BANCO 0
        XORWF   TEMP,W
        BTFSS   STATUS,Z
        GOTO    STATE5
        CALL    WRITEI2CH
        RETURN


STATE5:                                 ; EL ESTADO 5 OCURRE CUANDO EL MASTER HA ENVIADO UN NACK EN
RESPUESTA AL DATO
                                        ; QUE HA SIDO RECIBIDO DESDE EL ESCLAVO,ESTA ACCION INDICA QUE EL
MASTER NO DESEA LEER
                                        ; MÁS BYTES DE EL ESCLAVO
        MOVLW   b'00101000'
        BCF     STATUS,RP0
        BCF     STATUS,RP1              ;SELECCIONAMOS EL BANCO 0
        XORWF   TEMP,W
        BTFSS   STATUS,Z                ;ESTAMOS EN EL ESTADO 5?
        GOTO    ERRORI2C                ; NO, NOS VAMOS A LA SUBRUTINA ERROR DE I2C Y BF =0
        RETURN                          ; SI, ENTONCES SE RESETEA EL MODULO SSP POR HARDWARE

ERRORI2C                                ; NO ESTAMOS EN NINGUN ESTADO ALGO SALIO MAL
        BCF     SSPCON,SSPEN
        NOP
        BSF     SSPCON,SSPEN
        RETURN                          ; SALIMOS DE LA INTERRUPCION

                ;***************RUTINA DE ENVIO DE DATO************************
WRITEI2CL
        BCF     STATUS,RP0
        BCF     STATUS,RP1              ;SELECCIONAMOS EL BANCO 0
        MOVF    DISTANCIAL,W
        MOVWF   SSPBUF                  ; AHORA CARGAMOS W EN EL REGISTRO SSPBUF QUE A SU VEZ LO CARGA EN
SSPSR
        BSF     SSPCON,CKP              ;HABILITAMOS LA SALIDA DEL CLK (SE LIBERA)
        RETURN

                ;***************RUTINA DE ENVIO EL SEGUND00 DATO************************
WRITEI2CH
        BCF     STATUS,RP0
        BCF     STATUS,RP1              ;SELECCIONAMOS EL BANCO 0
        MOVF    DISTANCIAH,W
        MOVWF   SSPBUF                  ; AHORA CARGAMOS W EN EL REGISTRO SSPBUF QUE A SU VEZ LO CARGA EN
SSPSR
        BSF     SSPCON,CKP              ;HABILITAMOS LA SALIDA DEL CLK (SE LIBERA)
        RETURN

RETARDO         ;********RUTINA DE RETARDO*********************************
        CLRF    TMR0                    ; PONEMOS A CERO EL TIMER 0
        BTFSS   INTCON,T0IF             ;REVISAMOS LA BANDERA DEL TMR 0 SE ES UNO BRINCA PRODUCE TIEMPO DE
65 MILISEGUNDO
        GOTO    $-1                     ; SI NO ES CERO REGRESA
        BCF     INTCON,T0IF             ;PONEMOS A CERO LA BANDERA DEL TMR0
```

```
        CLRF    TMR0                    ; PONEMOS A CERO EL TIMER 0
        BTFSS   INTCON,T0IF             ;REVISAMOS LA BANDERA DEL TMR 0 SE ES UNO BRINCA PRODUCE TIEMPO DE
65 MILISEGUNDO
        GOTO    $-1                     ; SI NO ES CERO REGRESA
        BCF     INTCON,T0IF             ;PONEMOS A CERO LA BANDERA DEL TMR0
        CLRF    TMR0                    ; PONEMOS A CERO EL TIMER 0
        BTFSS   INTCON,T0IF             ;REVISAMOS LA BANDERA DEL TMR 0 SE ES UNO BRINCA PRODUCE TIEMPO DE
65 MILISEGUNDO
        GOTO    $-1                     ; SI NO ES CERO REGRESA
        BCF     INTCON,T0IF             ;PONEMOS A CERO LA BANDERA DEL TMR0


        RETURN
        end
```