ESCUELA SUPERIOR POLITÉCNICA DEL LITORAL

Facultad de Ingeniería Eléctrica y Computación

MEDIDOR DE DISTANCIA CON SENSORES ULTRASÓNICOS UTILIZANDO UN MICROCONTROLADOR AVANZADO. CON COMUNICACIÓN SERIAL A DATALOGGER E INTERFAZ GRÁFICA. FUENTE DE ENERGÍA: 4 PILAS RECARGABLES AA

TESINA DE SEMINARIO

Previo a la obtención del Título de:

INGENIERO EN ELECTRICIDAD ESPECIALIZACIÓN ELECTRÓNICA Y AUTOMATIZACIÓN INDUSTRIAL

Presentado por:

Santiago Napoleón Rubio Segovia
Kelly Shirley Villao Carrillo

GUAYAQUIL – ECUADOR

2010

A G R A D E C I M I E N T O

A Dios, Ser Supremo que guía nuestras vidas y que día a día nos llena de fortaleza y sabiduría para seguir adelante.

A los Ingenieros y maestros por los conocimientos, el apoyo y la amistad impartida en todos estos años de estudio.

A los verdaderos amigos y compañeros, que comparten las herramientas y el conocimiento.

Y un agradecimiento especial a mi compañero, amigo y novio Salvador Vallejo por su apoyo incondicional, alegría, amor y sobre todo levantarme en los momentos difíciles.

Kelly Villao Carrillo

A G R A D E C I M I E N T O

A Dios sobre todas las cosas, a mis Padres, a Juan Andrés Rubio Segovia, mi hermano, mis Tías y Tíos, mis Abuelos; quienes a lo largo de esta carrera con su apoyo tanto espiritual como material supieron guiarme al final de esta carrera politécnica

A los ingenieros y maestros por los conocimientos, el apoyo y la amistad impartida en todos estos años de estudio.

A mis amigos(as) y compañeros(as) tanto dentro como fuera de la ESPOL, que se compartió tiempos de estudio y de amistad.

Santiago Rubio Segovia

DEDICATORIA

A mis padres Shirley y Gilberth, quienes han sido mi apoyo constante. Y me han brindado a diario apoyo y amor.

A mi hermana Leslie y a mi sobrino Ángel quienes con su alegría y apoyo fueron motivación e inspiración constante para la finalización de mi carrera.

A mi abuelita América González (+) quien me ilumina con su amor y desde el cielo guía mis pasos.

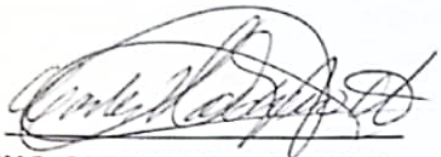
Kelly Villao Carrillo

DEDICATORIA

Dedico a Humberto Rubio Ortiz (**+**) y Juana Lozada (**+**), Ángel Segovia Lema (**+**) y María González Zumárraga, mis abuelos, por dar la vida y brindar la oportunidad de tener a dos maravillosos padres, Lucía Segovia González y Héctor Rubio Lozada, quienes con su sabiduría, paciencia y tolerancia me han guiado al final de la carrera y al comienzo de una vida profesional.

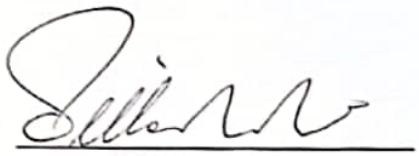
Santiago Rubio Segovia

TRIBUNAL DE SUSTENTACIÓN



ING. CARLOS VALDIVIESO A.

PROF. DEL SEMINARIO DE GRADUACIÓN



ING. HUGO VILLAVICENCIO V.

DELEGADO DEL DECANO

DECLARACIÓN EXPRESA

"La responsabilidad del contenido de esta tesina, nos corresponde exclusivamente; y el patrimonio intelectual de la misma, a la Escuela Superior Politécnica del Litoral".

(Reglamento de Graduación de la ESPOL)

Santiago Rubio Segovia

Kelly Villao Carrillo

RESUMEN

La finalidad del proyecto es desarrollar un medidor de distancia versátil y económico, que sea capaz de almacenar sus datos en un medio externo como el Datalogger y de presentar la variación de distancia con respecto al tiempo a través de la interfaz gráfica.

Se debe indicar el valor del medidor de distancia, tanto en el Datalogger como en la Interfaz Gráfica, en centímetros; el cual es la distancia que un objeto se encuentra con respecto al sensor de medición, en nuestro caso es el sensor ultrasónico PING)) diseñado por Parallax para Basic Stamp. Así mismo nuestro sensor debe cumplir con las característica dadas por el fabricante que se dará más adelante, para que este opere correctamente.

Una de las principales aplicaciones del proyecto es la medición de distancia entre objetos en movimiento y objetos estáticos. De esta manera se puede aplicar en el campo de la seguridad, al colocar un límite de distancia a la que se puede aproximar un objeto o una persona a una determinada zona de seguridad.

Para realizar nuestro proyecto vamos a utilizar las siguientes herramientas: Sensor ultrasónico Ping))) de Parallax, PIC 18F4431, dispositivo Datalogger, interfaz gráfica GLCD, BASIC Stamp HomeWork Board de Parallax, MikroBasic Pro, PROTEUS ISI y Basic Stamp.

En la programación se encuentra los códigos: del controlador del sensor PING))), el envío de la información a la interfaz gráfica y el almacenamiento de los datos por medio del dispositivo Datalogger.

Se realizarán pruebas con el sensor ultrasónico PING))) en los instrumentos y programas desarrollados por Parallax, para posteriormente ser replicado su modo de funcionamiento, pero con la utilización del PIC18F4431 y teniendo como plataforma de programación MikroBasic Pro.

Se realizarán las pruebas necesarias utilizando el programa simulador PROTEUS ISIS, de esta manera nos cercioramos que al implementar el proyecto tengamos los mejores resultados.

Además realizaremos varias pruebas con el prototipo ya implementado, a diferentes temperaturas y distancias para poder calibrar con precisión el controlador del sensor ultrasónico PING))).

INDICE GENERAL

RESUMEN

ÍNDICE GENERAL

ÍNDICE DE FIGURAS

INTRODUCCIÓN

CAPÍTULO 1 1

1 DESCRIPCIÓN GENERAL DEL PROYECTO 1

1.1 Antecedentes 1

1.2 Descripción del proyecto 2

1.3 Aplicaciones 3

CAPÍTULO 2 4

2 FUNDAMENTACION TEORICA DE LOS RECURSOS UTILIZADOS 4

2.1 HARDWARE 4

2.1.1 El sensor ultrasónico de distancia PING))) 4

2.1.2 El Microcontrolador PIC 18F4431 6

2.1.3 BASIC Stamp HomeWork Board de Parallax 8

2.1.4 El Microcontrolador Basic Stamp II (BS2) 12

2.2 SOFTWARE 13

2.2.1 MikroBasic Pro para PIC 13

2.2.2 PROTEUS 15

2.2.3 Editor BASIC Stamp 17

CAPÍTULO 3 20

3 DISEÑO E IMPLEMENTACIÓN 20

3.1 Diagrama de Bloques del proyecto 20

3.2 FUNCIONAMIENTO DEL SENSOR ULTRASÓNICO PING))) 20

3.3 CÓDIGO DE PROGRAMACIÓN BASIC STAMP 26

3.4 CÓDIGO DE PROGRAMACIÓN MIKROBASIC PRO 27

3.5 DETALLE DE COMANDOS USADOS 35
3.6 IMPLEMENTACIÓN 39
CAPÍTULO 4 40
4 SIMULACIÓN, PRUEBAS Y DATOS EXPERIMENTALES 40
4.1 Datos obtenidos en la simulación de Basic Stamp 40
4.2 Datos obtenidos con MikroBasic Pro y PROTEUS 42
4.2.1 PULSOUT 43
4.2.2 PULSIN 45
4.2.3 Comunicación serial a DATALOGGER 48
CONCLUSIONES
RECOMENDACIONES
ANEXOS
BIBLIOGRAFÍA

INDICE DE FIGURAS

FIGURA 1: Sensor de distancia por ultrasonidos PING))) de Parallax 5

FIGURA 2: Diagrama de Pin del PIC18F4X31 7

FIGURA 3: BASIC Stamp HomeWork Board 8

FIGURA 4: Esquemas del BASIC Stamp HomeWork Board 9

FIGURA 5: Diagrama esquemático y foto real del Basic Stamp 2 13

FIGURA 9: Vista de área de trabajo de PBASIC EDITOR 18

FIGURA 10: Configuración típica para su funcionamiento 18

FIGURA 11: Diagrama de tiempos del sensor PING))) 21

FIGURA 12: Funcionamiento del sensor PING))) 22

FIGURA 13: Modo de funcionamiento del sensor PING))) 23

FIGURA 13: Modo de funcionamiento del sensor PING))) 24

FIGURA 14: Materiales porosos que absorbe sonido 25

FIGURA 15: Simulación en Basic Stamp 41

FIGURA 16: Pruebas con el Basic Stamp Protoboard 41

FIGURA 17: Vista del proyecto simulado en PROTEUS 42

FIGURA 18. a: Simulación en PROTEUS de la señal PULSOUT 43

FIGURA 18. b: Simulación en PROTEUS de la señal PULSOUT 43

FIGURA 19.a: Datos reales del PULSOUT en Osciloscopio 44

FIGURA 19.b: Acercamiento de datos reales del PULSOUT 45

FIGURA 20: Simulación de la señal que envía el sensor PING))) por medio del pulso de reloj en PROTEUS 45

FIGURA 21: Simulación en PROTEUS de distancia con diferentes valores frecuencia de pulso de entrada..... 46

FIGURA 22: Implementación conectando el sensor PING)))..... 47

FIGURA 23: Datos reales del PULSIN a 91 cm de distancia..... 48

FIGURA 24: Datos reales del PULSIN a 121 cm de distancia..... 48

FIGURA 25: Simulación en PROTEUS de datos enviados al DATALOGGER 49

FIGURA 26: Pruebas al enviar datos por medio de la comunicación serial al DATALOGGER..... 50

INDICE DE TABLAS

TABLA 1: Características básicas de BASIC Stamp HomeWork Board de Parallax..... 9

TABLA 2: Descripción y ubicación de los pines del Basic Stamp 2 13

TABLA 3: Adquisición de datos con BASIC Stamp HomeWork Board..... 40

INTRODUCCIÓN

El presente Proyecto forma parte del seminario de graduación de "Microcontroladores Avanzados" y consiste en el diseño de un "Medidor de distancia con sensores ultrasónicos utilizando un microcontrolador avanzado. Con comunicación serial a Datalogger e Interfaz Gráfica. Fuente de energía: 4 pilas recargables AA.".

El principal objetivo de este proyecto es la utilización del sensor ultrasónico PING))) de Parallax controlado por medio de un microcontrolador avanzado.

De esta manera se podrán tomar datos de distancia de una manera fácil y económico, ya que no será necesario usar la plataforma de programación, ni los equipos que la empresa Parallax ha diseñado para el uso de sus sensores.

En el capítulo 1 comienza al indicar los antecedentes, puntualizando el problema que se busca resolver y plantea los lineamientos y características de la solución a ser implementada. Se introducen los conceptos relacionados al prototipo, la descripción, funcionamiento y aplicaciones del mismo.

En el capítulo 2 se describe cuales son las herramientas a utilizar y sus conceptos básico, se ha divido esta sección en herramientas de hardware y software las cuales utilizaremos para las pruebas e implementación del proyecto.

Este capítulo incluye una revisión de varias de las tecnologías actuales que atacan los problemas relacionados a la creación de prototipos.

Los paradigmas de interacción, el reconocimiento de trazos y algunas de las soluciones existentes para la creación de prototipos, de alto costos, son detallados a fin de plantear una visión general del marco teórico sobre el cual se trabajará en las secciones posteriores.

En el capítulo 3 se describen el análisis y diseño del sistema. Se expone el funcionamiento del sensor ultrasónico a utilizar, además se describe los comandos que se utilizaban como antecedente a este proyecto y se muestra la solución de programación que se utiliza para nuestro proyecto aplicando el microcontrolador avanzado.

Además, continúa describiendo el de detalle de los comandos utilizados y mostrando la implementación del proyecto en estado físico.

En el capítulo 4, se especifican las pruebas realizadas en las simulaciones y en el proyecto implementado.

Finalmente, se especifican las conclusiones, recomendaciones de este proyecto.

CAPÍTULO 1

1 DESCRIPCIÓN GENERAL DEL PROYECTO

1.1 Antecedentes

En lo que corresponde a la medición de distancia existen algunos sistemas activos para la medición de proximidad o de distancias sin necesidad de contacto físico, entre ellos podemos mencionar: Los ópticos y los ultrasónicos. Los sistemas ópticos ofrecen mejor precisión debido a que la longitud de onda involucrada es más corta, como también presentan menor sensibilidad a condiciones ambientales como por ejemplo: la presión y la temperatura. Por otra parte, las aplicaciones ultrasónicas son basadas en la medición del tiempo de vuelo, por lo que son más simples y en consecuencia económicas. Generalmente la medición de distancia utilizando sensores ultrasónicos programados por PIC son utilizados en prototipos de robótica para que puedan percibir información a cierta distancia del terreno en que se desenvuelven y así ser capaz de movilizarse. Este sensor es aconsejable para innumerables

aplicaciones que requieren de mediciones entre objetos en movimiento y en sistemas de seguridad o en ocasiones para remplazar sensores infrarrojos.

En nuestro caso vamos a utilizar el sensor ultrasónico PING))) de Parallax, el cual tiene una aplicación en Basic Stamp HomeWork Board de Parallax por medio del programa Basic Stamp.

En este software existen dos funciones que son fundamentales para el funcionamiento del sensor PING))). Primero, la función PULSOU, el cual habilita un pulso en alto a la entrada SIG del sensor PING))). Segundo, el PULSIN, el cual recibe un pulso en alto, que es el tiempo en que viaja la onda ultrasónica a partir que choca con el objeto hasta llegar al receptor del sensor PING))), para luego almacenar los valores y presentarlos en el Datalogger y la Interfaz Gráfica.

1.2 Descripción del proyecto

Nuestro proyecto consiste en la medición de distancia con un sensor ultrasónico Ping))) de PARALLAX, utilizando un microcontrolador avanzado 18F4437. Los datos de distancia serán almacenados cada cien milisegundos, además tiene la capacidad de comunicase por medio de comunicación serial hacia el DATALOGGER y la variación de distancia con respecto al tiempo serán mostrados en una Interfaz Gráfica GLCD con una fuente de energía de cuatro pilas recargables AA.

Para programar el PIC que controlará el sensor ultrasónico utilizamos el programa MikroBasic Pro y los datos serán almacenados en la memoria EEPROM del PIC. El proyecto tiene capacidad de detección de objetos que se encuentre en el rango de 3cm hasta 3 m.

Para esto tendremos que programar las funciones PULSOUT Y PULSIN en MikroBasic Pro, ya que estas no existen en este programa; como también utilizar la función UART, creada por MikroBasic Pro, para envío de datos al Datalogger y a la interfaz gráfica GLCD.

1.3 Aplicaciones

Una de las principales aplicaciones del proyecto es la medición de distancia entre objetos en movimiento y objetos estáticos. De esta manera se puede aplicar en proyectos de robótica, para que puedan percibir información a cierta distancia del terreno en que se desenvuelven y con este información controlar sus movimientos; además en el campo de la seguridad, al colocar un límite de distancia a la que se puede acercar un objeto o una persona a una determinada zona de seguridad.

Otra aplicación es para ayudar a las personas no videntes, al detectar la distancia a la que están los objetos a su alrededor, agregándole un dispositivo audible que les alerte del peligro por la cercanía de un objeto.

En el campo de la industria, para indicar la distancia a la que se encuentra un objeto y con esta información controlar el proceso a realizar.

CAPÍTULO 2

2 FUNDAMENTACION TEORICA DE LOS RECURSOS UTILIZADOS

Requerimientos para la aplicación del proyecto.

Para realizar nuestro proyecto vamos a utilizar las siguientes herramientas:

- **Hardware:** sensor ultrasónico Ping))) de Parallax, PIC 18F4431, BASIC Stamp HomeWork Board de Parallax, LCD 16x2 pixeles y GLCD 128 x 64 pixeles.
- **Software:** MikroBasic Pro, PROTEUS ISI y Basic Stamp.

2.1 HARDWARE

2.1.1 El sensor ultrasónico de distancia PING)))

El sensor nos permite efectuar la medición de distancia de objetos colocados entre 3 cm. y 3 m., es fácil de conectar y requiere únicamente para su operación un terminal de entrada /salida del microcontrolador. El funcionamiento de este sensor se basa en la utilización de ondas ultrasónicas, las cuales se caracterizan porque su frecuencia supera la capacidad de audición de los seres humanos. El

oído humano es capaz de detectar ondas sonoras de frecuencias comprendidas entre 20 y 20000 Hz, a estos se les conocen como espectro audible. Toda señal sonora que se encuentre por encima de este rango, se cataloga como ultrasónica.

FIGURA 1: Sensor de distancia por ultrasonidos PING))) de Parallax

El sensor PING))) transmite una ráfaga ultrasónica y detecta el tiempo que demora el eco en ser recibido. Este eco se produce cuando las ondas ultrasónicas golpean un objeto que se encuentra dentro del rango de medición del sensor PING))). Ver **Anexo 9**

El sensor PING))) por medio de su salida entrega un pulso digital que es proporcional al tiempo para ir desde el módulo emisor hasta golpear contra un objeto y regresar hacia el receptor. Para lograr que el microcontrolador obtenga la medición de distancia de un objeto colocado frente al sensor PING))), se mide la duración de este pulso y se multiplica por un factor constante que más adelante detallamos.

Características técnicas:

- ✓ Voltaje de Alimentación = 5 VDC.
- ✓ Consumo de Corriente = 30 – 35 mA (máx.).
- ✓ Rango de medición = 3 cm. hasta 3 m.
- ✓ Entrada de disparo = Pulso ascendente TTL con duración mínima de 5us.
- ✓ Pulso de salida = Pulso ascendente TTL comprendido entre 115 us. y 18.5 ms.
- ✓ Frecuencia del ultrasonido = 40 Hz.
- ✓ Tiempo de emisión del ultrasonido= 200 us.
- ✓ Diodo LED indicador de actividad.
- ✓ Tiempo mínimo de espera en medidas=200 us.
- ✓ Dimensiones = 22x46x16 mm.

2.1.2 El Microcontrolador PIC 18F4431

Este microcontrolador pertenece a la familia de microcontroladores avanzados PIC18FXXXX, los cuales tienen un alto rendimiento computacional a un costo asequible.

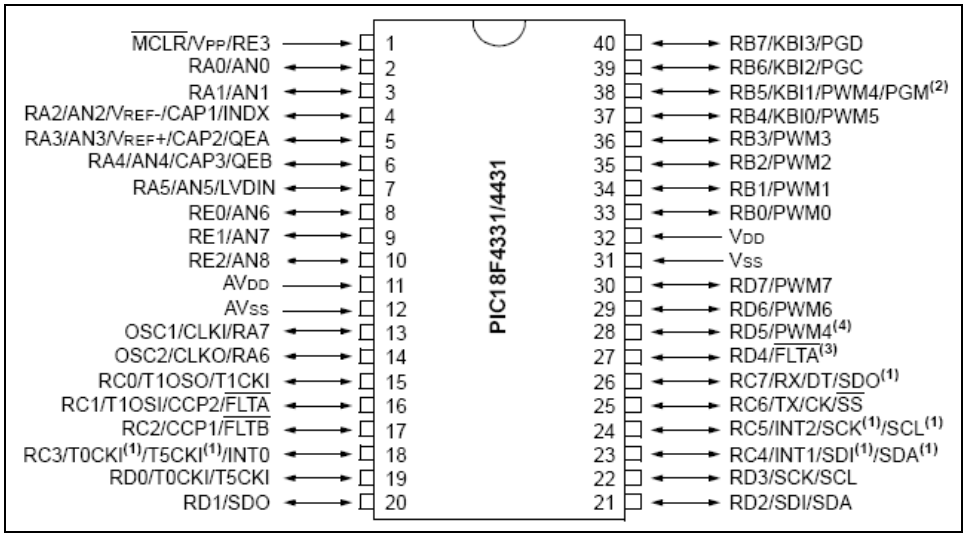


FIGURA 2: Diagrama de Pin del PIC18F4X31

Las diferencias básicas entre el PIC18F4X31 (PIC18F4431, PIC18F4331) y los otros miembros de la familia PIC18 son:

1. La memoria Flash programable es de 16 Kbyte para PIC18F4X31, mientras que la serie d PIC18F2X31 tiene solo 8 Kbyte.
2. La serie PIC18F4X31 tiene 9 canales analógicos/digitales, mientras que la serie de pic18f2x31 dispone de 5 canales A/D.
3. La serie PIC18F4X31 posee 5 puertos bidireccionales (I/O), a diferencia del los PIC18F2X31 que posee 3 puertos I/O.

Otras características específicas del PIC18F4431 son:

- ✓ Memoria de programación 16384 Bytes = 8192 Instrucciones.
- ✓ Memoria de datos 768 Bytes.
- ✓ Datos de memoria de EEPROM, 256 Bytes.
- ✓ Comunicación serial SSP y USART mejorado.

2.1.3 BASIC Stamp HomeWork Board de Parallax

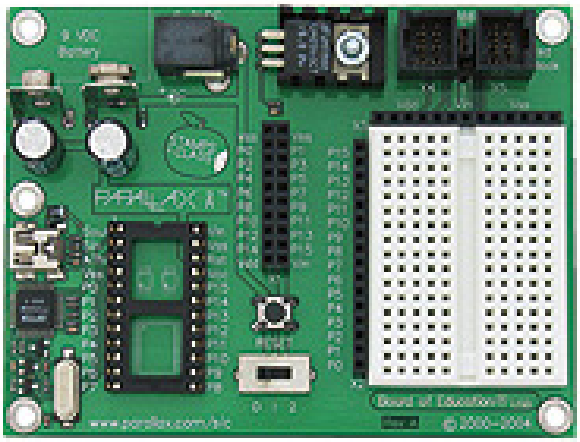


FIGURA 3: BASIC Stamp HomeWork Board

El HomeWork Board fue creado exclusivamente para desarrollo de proyectos educativos, al ser un dispositivo amigable y fácil de trabajar.

Este equipo usa el BASIC Stamp 2, montados directamente a la placa de circuito impreso. El BASIC Stamp 2 tiene una gran variedad de hardware de apoyo, ejemplos de código y aplicaciones. Por otra parte, realiza la mayor parte de las mismas funciones que las nuevas BASIC Stamp, aunque con menos velocidad y memoria. Las especificaciones técnicas de Stamp 2 de las especificaciones técnicas se muestran a continuación. Ver **Anexo 5** el Esquemático Serial BASIC Stamp HomeWork Board de Parallax.

Microcontrolador	PIC16C57 Montaje superficial
Velocidad	20 MHz/ 4000 instrucciones por segundo
EEPROM	2K bytes dato y programa
Distancia de Programa	500 líneas de PBASIC
RAM (variables)	32 bytes (6 para I/Os y 26 para variables)
Input /Outputs	16 a 17 con comunicación RS-232
Fuente de Corriente	50mA / 50 mA
Comunicación Serial	300 - 50K baudios I/O
Interfaz para la PC	Puerto Serial
Fuente de Poder	5 V a través del regulador LM2936 desde una bacteria de 9V.
Ambiente	32 a 158 F (0 a 70C), 70% humedad no condensada
Tamaño de la tarjeta de trabajo	3" x 4"
Área de Proyecto	Construido en Stamp Lab

TABLA 1: Características básicas de BASIC Stamp HomeWork Board de Parallax

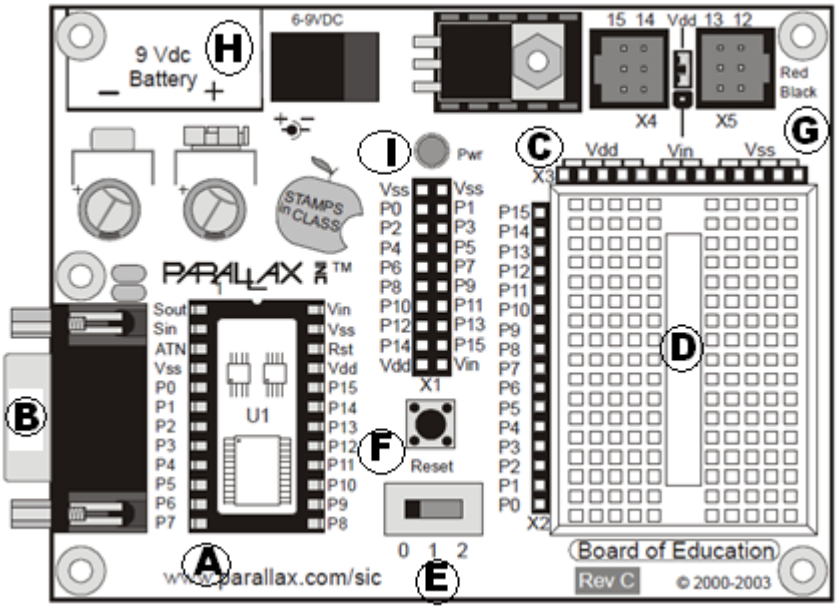


FIGURA 4: Esquemas del BASIC Stamp HomeWork Board

A. **Módulo de BASIC Stamp 2 :** El BASIC Stamp está compuesto por varios componentes: PIC16C57-20/SS- Un microchip 2K microcontrolador de 8-bit programado con BASIC Stamp "intérprete" que se ejecuta del BASIC Stamp de Parallax, código como un pequeño sistema operativo; la EEPROM 24LC16B - un microchip 2K EEPROM (eléctricamente borrable, memoria de sólo lectura programable) con una modificable por el usuario de sólo lectura, memoria (ROM) que pueden ser reprogramados con su código de BASIC Stamp, un cristal de 20 MHz para proporcionar una fuente de reloj exacto para el BASIC Stamp, 220 Ohm resistencias en todo el pines I / O para proporcionar una protección para el cableado de error de corriente, un regulador de voltaje LM2936 que proporciona 50 mA para el BASIC Stamp y sus circuitos alimentados desde la conexión de la placa universal Vdd.

B. **DB-9 hembra:** El DB-9 del puerto se conecta mediante un cable serial al puerto serial de su PC. Este puerto se utiliza para programas de descarga en el BASIC Stamp y para el BASIC Stamp para enviar datos a la PC

C. **Conexiones de potencia y tierra:** Vdd está regulada de 5V, Vin es de 9V de la batería y tierra que es Vss.

D. **Breadboard.** Dos áreas de la columna 5 x 17 filas Protoboard zona del proyecto**.** Las conexiones son horizontales separadas por el valle.

E. Botón de encendido. Este interruptor tiene tres posiciones ubicado en la parte central inferior del tablero. El extremo izquierdo posición (0) es apagado y el equipo está totalmente desconectado. Siempre coloque el interruptor en esta posición cuando se añade o cambie los componentes de la placa. La posición central (1) proporciona Vin al regulador y los conectores señalados. En esta posición, Vdd también estará disponible en el Protoboard. La posición más a la derecha (2) proporciona energía al conectores servo X4 y X5 (véase Selección de la alimentación del servo a continuación). Utilice la posición 1 para editar y probar código mientras se deja de alimentar los servos. Cuando esté listo para una prueba completa, descargue el código, mueva el interruptor de encendido en la posición 2, a continuación reiniciar.

F. Reset. Restablecer el BASIC Stamp presionando este botón.

G. Selección de Poder del Servomotor: se puede elegir la potencia suministrada servo X4 y X5 tomadas por un puente que se encuentra entre estas dos tomas de corriente; la posición por defecto es Vdd (+5 regulados). Cuando se utiliza una batería de seis voltios (como en un Boe-Bot robot, por ejemplo), se debería mover este puente en la posición Vin para proporcionar energía extra a los servos. Cuando se utiliza Vin en los puertos de servomotores, siempre verifique que la tensión de alimentación no se exceda las especificaciones de la marca en particular de los servos que está utilizando.

H. Batería: en este dispositivo se conecta la energía al Protoboard que puede ser por medio de una batería de 9 VDC o por medio de un adaptado de energía de 6 o 9 VDC.

I. AppMod Header: El encabezado AppMod proporciona una conexión y enrutamiento de la señal para el EmbeddedBlue Transceptor y la terminal LCD. Todos los pines I / O, Vdd, Vin, Vss y se enrutan a través del conector AppMod.

2.1.4 El Microcontrolador Basic Stamp II (BS2)

El BASIC Stamp II es un pequeño computador que ejecuta programas en lenguaje PBASIC. El BS2-IC tiene 16 pines de (entrada / salida) I/O que pueden ser conectados directamente a dispositivos digitales o de niveles lógicos, tales como: Botones, diodos Leeds, altavoces, potenciómetros, y registros de desplazamiento. Además, con unos pocos componentes extras, estos pines de I/O pueden ser conectados a dispositivos tales como: Solenoides, relay, servomotores, motores de paso a paso y otros dispositivos de alta corriente o tensión.

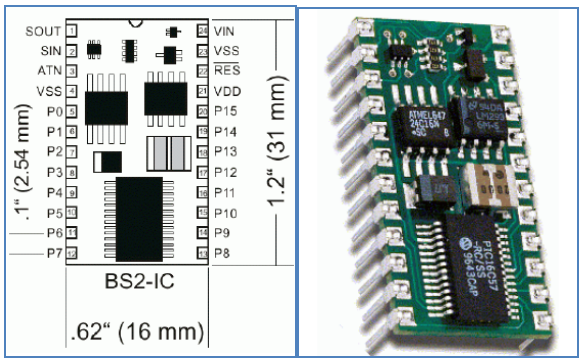


FIGURA 5: Diagrama esquemático y foto real del Basic Stamp 2

PIN	Nombre	Descripción
1	SOUT	Serial Out: Conectar al puerto serial RX (DB9 pin 2)
2	SIN	Serial In: Conectar al puerto serial TX (DB9 pin 3)
3	ATN	Atención: Conectar al puerto serial DTR (DB9 pin 4)
4	GND	Tierra entre el puerto serial y el BS2
5 -20	P0 - P15	Puerto de propósitos generales, cada uno puede entregar 25 mA, sin embargo, el total de la corriente no puede exceder los 75 mA utilizado el regulador interno y 100 mA utilizado +5V externo
21	VDD	Voltaje regulado a +5VDC
22	RES	Reset, Basta con aterrizar y el BS2 se reinicia
23	GND	Tierra del BS2
24	PWR	Voltaje no regulado entre +5,5 a +15VDC, si VDD es utilizado VIN no puede ser utilizado.

TABLA 2: Descripción y ubicación de los pines del Basic Stamp 2

2.2 SOFTWARE

2.2.1 MikroBasic Pro para PIC

MikroBasic Pro (MB-P) es un lenguaje de programación basado en el lenguaje BASIC, pero que se encuentra orientado hacia los microcontroladores.

La características más destacadas de estos compiladores, es la inclusión de un IDE (entorno de desarrollo integrado) que hace muy cómoda la programación, ya que resalta la sintaxis del lenguaje, proporciona acceso muy rápido a la excelente ayuda incluida, estadísticas sobre el uso de recursos del microcontrolador, posee periféricos externos tales como pantallas LCD y otras ventajas.

Tiene Built-in incluye un conjunto de librerías y ejemplos destinados a facilitar el desarrollo de aplicaciones. Las rutinas están documentadas en detalle y permite que de inicio rápido en programación de microcontrolador, navegar a través de los ejemplos proporcionados y se aprende a utilizar los PIC con un mínimo de código y esfuerzo.

MikroBasic Pro para PIC organiza aplicaciones en los proyectos, que consisten en un solo fichero de proyecto, los cuales son denominados módulos en lenguaje de programación MikroBasic. El compilador de MikroBasic Pro permite manejar varios proyectos a la vez, los ficheros compilan fuentes que forman parte del proyecto.

Con respecto a la programación MikroBasic Pro tiene una desventaja comparado con Basic STAMP, pues posee dos códigos: PULSIN y PULSOUT; los cuales permite controlar el sensor ultrasónico PING))).

Estos códigos como se verá posteriormente serán remplazadas por subrutinas programadas en MikroBasic Pro para poder trabajar con el microcontrolador avanzado.

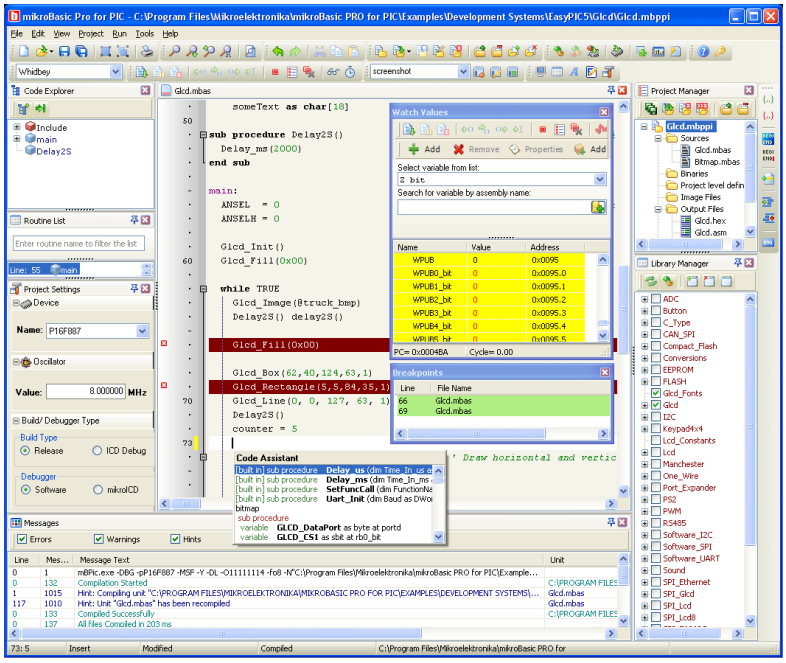


FIGURA 6: Visualización del entorno de MikroBasic Pro

2.2.2 PROTEUS

PROTEUS es una aplicación de CAD, compuesto de tres módulos: ISIS (Ver **Figura 7**)que es el módulo de captura de esquemas, VSM que es el módulo de simulación incluyendo PROSPICE y ARES (Ver **Figura 8**)el cual es módulo para la realización de circuitos impresos (PCB).

Las utilidades que posee este software son entre otras:

- Librerías de componentes.
- Conexiones automáticas entre 2 puntos del esquema.

- La lista de redes (NetList) son compatibles con la mayoría de los programas de realización de PCB.
- Capacidad de simular microcontroladores como los PIC de Microchip, Basic Stamp, entre otros.
- Se puede visualizar la RAM (Registros especiales y datos), además de la EEPROM y la memoria de programa. Se pueden establecer puntos de detención para la depuración.
- Posee una variada instrumentación virtual que nos facilita el análisis de circuitos. Estos dispositivos se pueden insertar en los circuitos, mostrando las medidas de tiempo real según se simula.

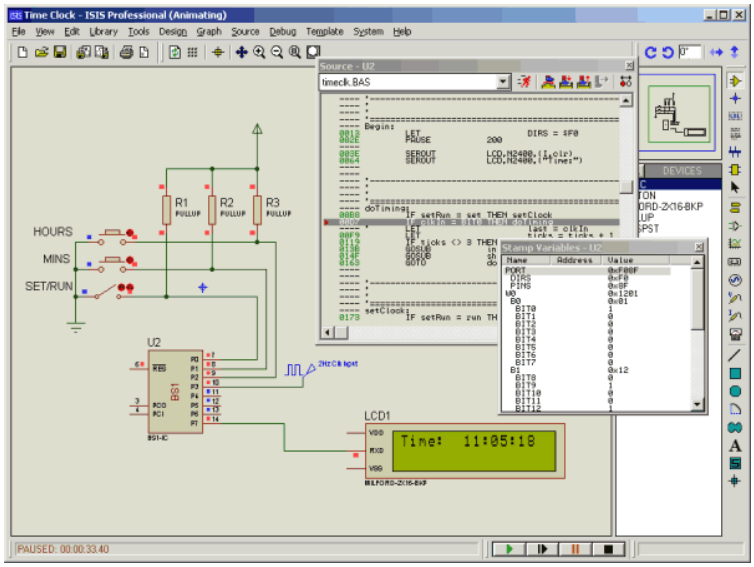


FIGURA 7: Vista de las pantallas de PROTEUS ISIS

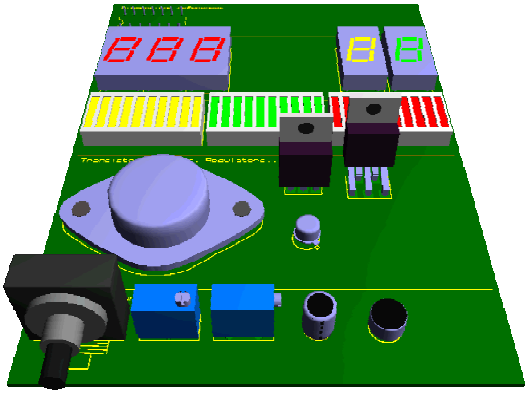


FIGURA 8: Vista de las pantallas de PROTEUS ARES

2.2.3 Editor BASIC Stamp

El PBASIC Editor es el programa donde escribimos el conjunto de instrucciones para el Basic Stamp, es similar en apariencia a cualquier editor de texto del sistema operativo WINDOWS. El editor contiene una serie de herramientas como son: Identificadores del Basic Stamp, corrector ortográfico de sintaxis, mapa de memoria y ventana del depurador.

El editor tiene la capacidad para abrir 16 ventanas simultáneamente. La capacidad de cortar, copiar y pegar se mantiene innata. Su entorno es muy sencillo y usted se familiarizara muy pronto.

Los comandos más importantes son:

F9 ó Ctrl-R Descarga el programa en el BS2.

F7 ó Ctrl-T Corrector de Sintaxis.

F8 ó Ctrl-M Muestra el mapa de memoria.

F6 ó Ctrl-I Muestra el número de versión de PBASIC.

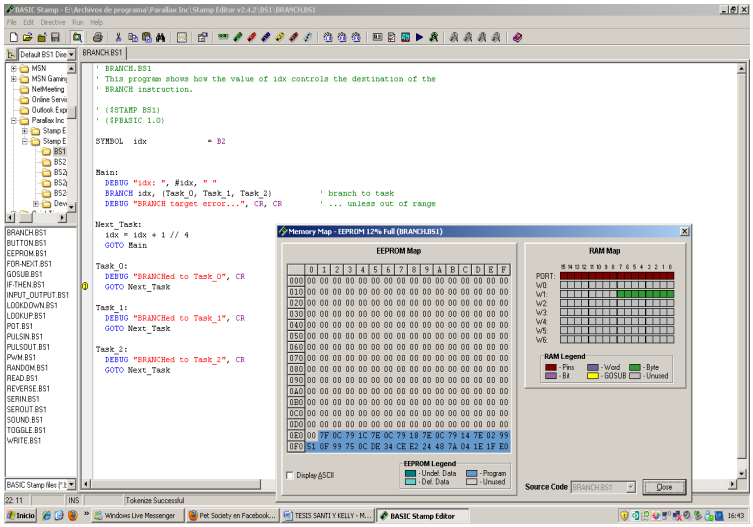


FIGURA 6: Vista de área de trabajo de PBASIC EDITOR

Procedimiento para descargar el programa al BS2:

1. Con el BS2 previamente energizado y conectado como lo indica la **Figura 10**, cargue el editor PBASIC.

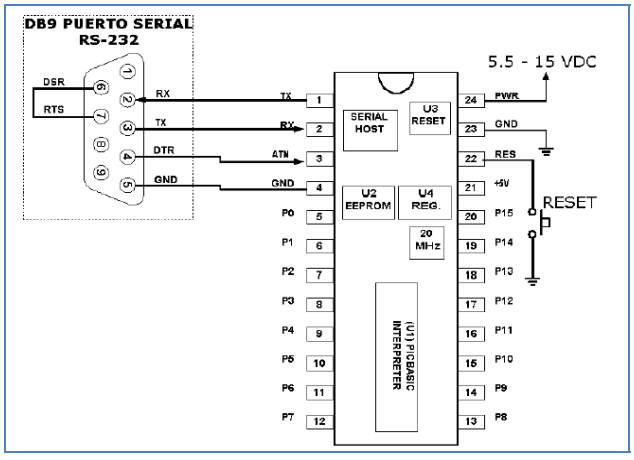


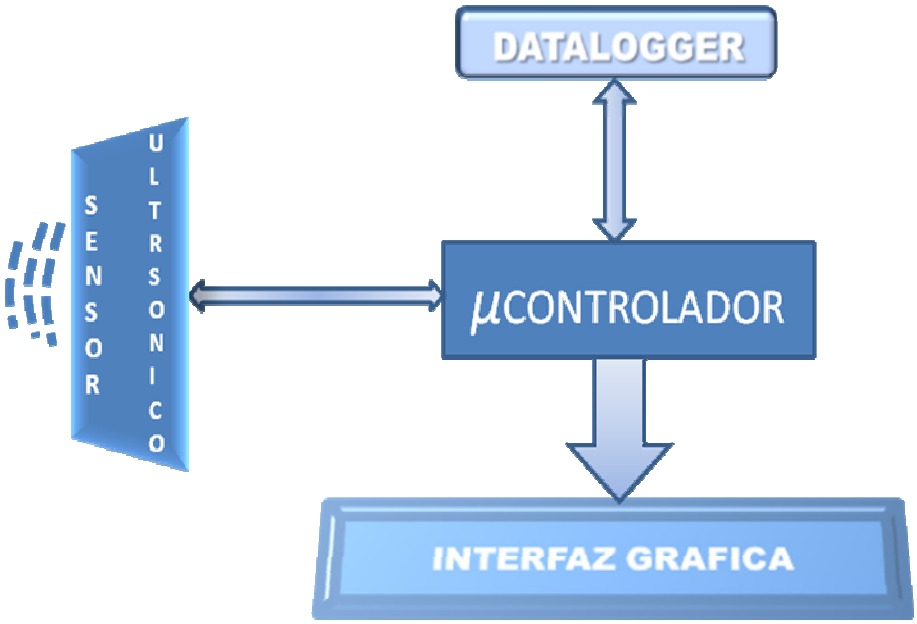
FIGURA 7: Configuración típica para su funcionamiento

2. Cuando el editor esté listo presione [Ctrl-I], si todo está bien conectado el editor le dará un mensaje de “Found BS2-IC (firmware v1.0)”. Esto indica que se encuentra en estado de confirmación.
3. Se puede digitar su programa o cargar uno previamente del disco.
4. Para asegurar que el código digitado esté bien, presione el corrector de sintaxis [Ctrl-T], si existe algún problema lo indicara con un mensaje de error. Si todo está bien le indicará un mensaje de “Tokenize Successful”
5. Ahora se puede descargar el programa en el BS2, presione [Ctrl-R], y el programa se descargara permanentemente en la EEPROM del BS2. En caso de que no se revise con el corrector de sintaxis, antes del programa descargue en el BS2, este lo realiza por su cuenta.
6. Apague el Basic Stamp 2, Retire el cable serial del BS2.
7. Encienda el Basic Stamp 2 y la aplicación permanecerán hasta que se modifique nuevamente reprogramando por el puerto serial.

CAPÍTULO 3

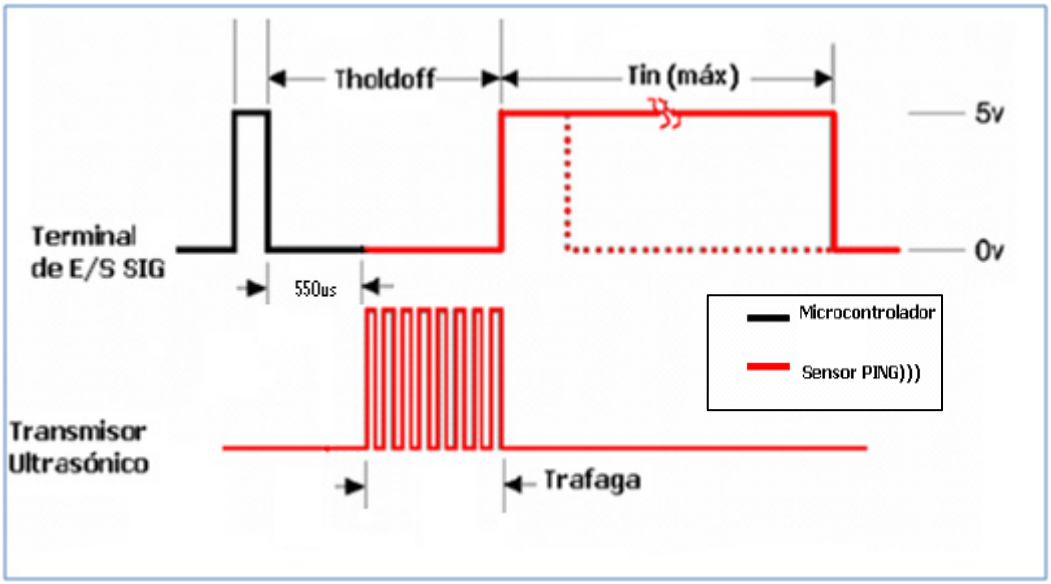
3 DISEÑO E IMPLEMENTACIÓN

3.1 Diagrama de Bloques del proyecto



3.2 FUNCIONAMIENTO DEL SENSOR ULTRASÓNICO PING)))

Para hacer la programación del PIC debemos de saber cómo es el funcionamiento del sensor PING))), el cual es explicado de una forma simple por medio del diagrama de tiempos siguiente.



Host Device	Input Trigger Pulse	t_{OUT}	2 µs (min), 5 µs typical
PING))) Sensor	Echo Holdoff	$t_{HOLDOFF}$	750 µs
	Burst Frequency	t_{BURST}	200 µs @ 40 kHz
	Echo Return Pulse Minimum	t_{IN-MIN}	115 µs
	Echo Return Pulse Maximum	t_{IN-MAX}	18.5 ms
	Delay before next measurement		200 µs

FIGURA 8: Diagrama de tiempos del sensor PING)))

El microcontrolador debe garantizar que exista un estado bajo en el pin **SIG** del sensor antes de comenzar la operación; posteriormente se genera un pulso de activación de disparo. Al concluir este pulso, el terminal del microcontrolador conectado al pin **SIG** debe convertirse en una entrada para permitir que el sensor ultrasónico PING))) tome el control del mismo. El sensor activa al transmisor de ultrasonido enviando una ráfaga que tiene una duración de 200us.

Esta ráfaga viaja en el aire a una velocidad aproximada de 1239.93 Km/hora golpea al objetivo en frente del sensor y genera una señal de rebote que es receptada por el micrófono de ultrasonidos del módulo. EL terminal **SIG** se coloca en estado alto luego de ser enviada la ráfaga de 40 Hz y permanecerá de esa manera por un tiempo mínimo de 115us y máximo 18.5 ms.

Para efectuar el cálculo de la distancia debe considerarse que el pulso recibido tiene una duración proporcional al doble de la distancia recorrida por la onda ultrasónica. Para comprender esto, analicemos la **Figura 12**.

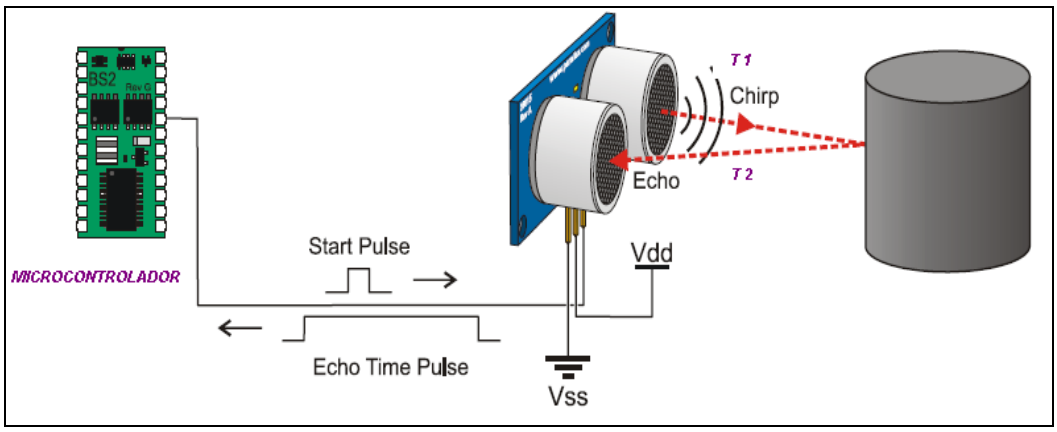


FIGURA 9: Funcionamiento del sensor PING)))

En la **Figura 12,** se observa que la señal demora un tiempo T1 en alcanzar al objeto y posteriormente toma un tiempo T2 en llegar de regreso al PING))), ya que ambas ondas se propagan por el mismo medio (aire), los tiempos T1 y T2 serán iguales.

Consideraciones prácticas para el uso.

Objeto de posicionamiento

El Sensor PING))) no puede medir con precisión la distancia a un objeto que:

a) Se encuentre a más de 3 metros de distancia.

b) Tenga su superficie reflectante en ángulos poco profundos porque de esta forma la onda ultrasónica no se refleja de vuelta hacia el sensor, ver **Figura 13.b**.

c) Es demasiado pequeño para reflejar suficiente sonido hacia el sensor, ver **Figura 13.c**.

d) Además, si el Sensor PING))) está montado en una superficie baja, el dispositivo, es posible que detecte el eco que se refleje en el piso, **Figura 13.d**.

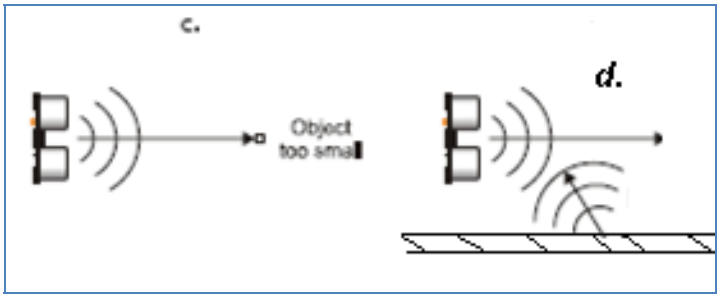


FIGURA 10: Modo de funcionamiento del sensor PING)))

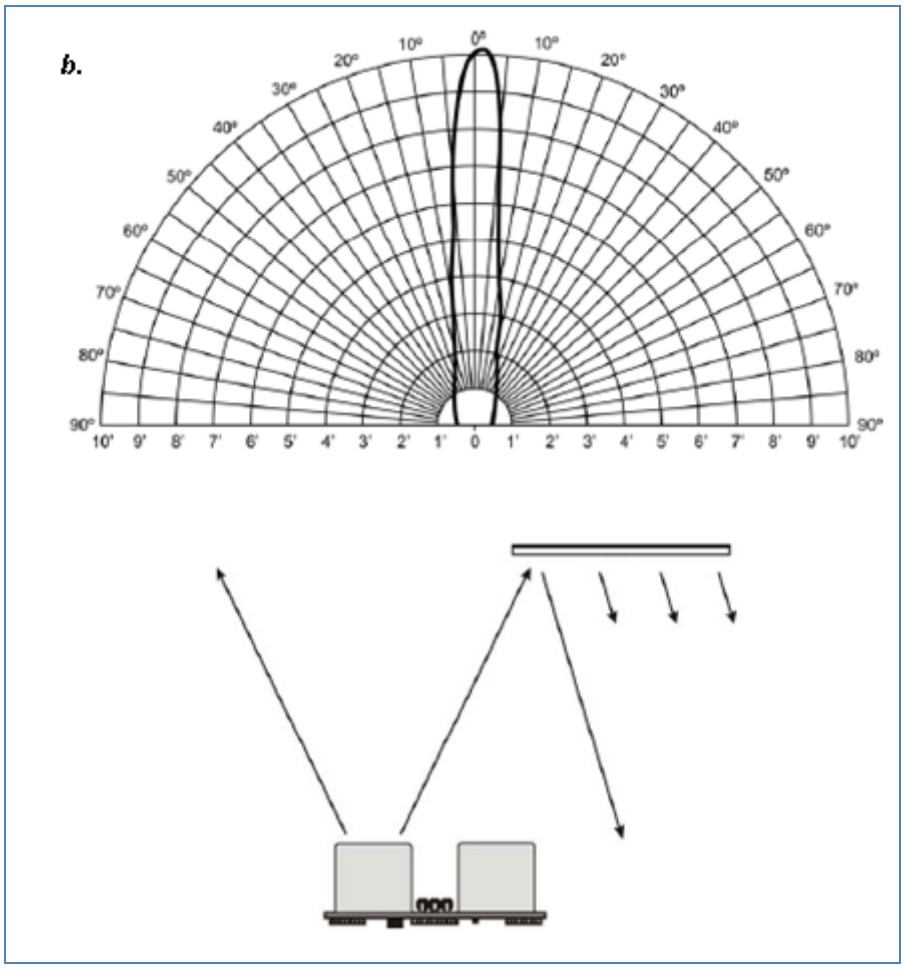


FIGURA 11: Modo de funcionamiento del sensor PING)))

Material del objeto

Existen objetos que absorben el sonido como los materiales porosos (absorben más sonido conforme aumenta la frecuencia) ver **Figura 14**, tienen una superficie blanda ó irregular, como un muñeco de peluche, en los cuales la reflexión del sonido no es lo suficientemente buena para que el sensor tenga una mayor precisión. El Sensor PING))) detectará la superficie del agua, sin embargo, no está clasificado para uso al

aire libre o el uso permanente en un ambiente húmedo. La condensación en sus transductores puede afectar el desempeño y vida útil del dispositivo.

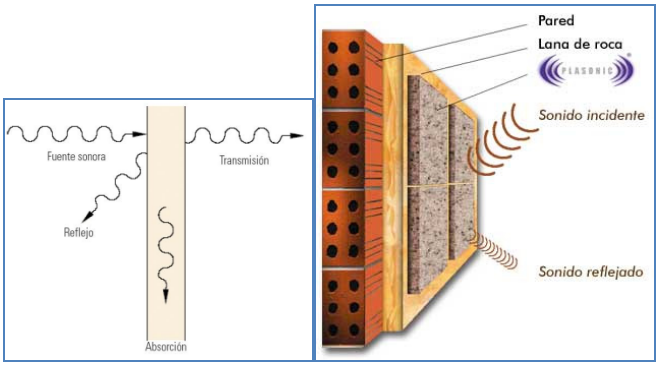


FIGURA 12: Materiales porosos que absorbe sonido

Cálculo de la distancia en centímetro.

La ecuación que describe la velocidad con la que viaja el sonido es S = t Cair, donde S es la distancia, Cair es la velocidad del sonido en el aire, y t es el tiempo. Dado que el sensor ultrasónico Ping))), detecta el tiempo que tarda el sonido para llegar al objeto y rebotar. La distancia real Sobject, es la mitad de la distancia total que el sonido se desplaza.

$$S = t \cdot C_{air}$$

$$S_{object} = \frac{S}{2} = t \cdot C_{air}$$

La velocidad del sonido en el aire esta en términos de metros por segundo (m / s). Sin embargo, los valores en centímetros son más convenientes para calcular con el MikroBasic PRO. Dado que hay 100 centímetros en un metro, vamos a

usar Sobject cm que es simplemente Sobject multiplicado por 100. El PULSIN en nuestro programa de MikroBasic Pro tiene una duración de 2us.

$$S_{object-cm} = \frac{100t \,.\, C_{air}}{2}$$

$$S_{object-cm} = \frac{100t \,.\, C_{air}\; t_{pulsin-mikropro}}{2} \times \frac{2}{1000000}$$

$$S_{object-cm} = \frac{C_{air}\;\; t_{pulsin-mikropro}}{10000}$$

La velocidad del sonido en el aire a temperatura ambiente de 22,2 ° C es 344,8 m/s. Dividiendo 10.000 en esto nos deja con Sobjecto cm = 0,03448 tpulsin_mikropro.

$$S_{object-cm} = \frac{344.8\;\; t_{pulsin-mikropro}}{10000}$$

$$S_{object-cm} = 0.0344.8\, t_{pulsin-mikropro}$$

3.3 CÓDIGO DE PROGRAMACIÓN BASIC STAMP

```
'Medidor de distancia en centímetros con el sensor ultrasónico
' {$STAMP BS2}
' {$PBASIC 2.5}
' Conversion constants for room temperature measurements.
```

```
CmConstant CON 2260
cmDistance VAR Word
time VAR Word
DO
PULSOUT 15, 5
PULSIN 15, 1, time
cmDistance = CmConstant ** time
DEBUG HOME, DEC3 cmDistance, " cm"
PAUSE 100
LOOP
```

3.4 CÓDIGO DE PROGRAMACIÓN MIKROBASIC PRO

```
program Beta18
 ' Módulo de Conexión de LCD
dim LCD_RS as sbit at RB4_bit
   LCD_EN as sbit at RB5_bit
   LCD_D4 as sbit at RB0_bit
   LCD_D5 as sbit at RB1_bit
   LCD_D6 as sbit at RB2_bit
   LCD_D7 as sbit at RB3_bit
   LCD_RS_Direction as sbit at TRISB4_bit
   LCD_EN_Direction as sbit at TRISB5_bit
   LCD_D4_Direction as sbit at TRISB0_bit
```

```
   LCD_D5_Direction as sbit at TRISB1_bit
   LCD_D6_Direction as sbit at TRISB2_bit
   LCD_D7_Direction as sbit at TRISB3_bit
' End Lcd module connections
dim Cont_Data as byte
dim distancia as word
   memoria_dist as word[30]
dim txt as string[5]
   Leer_Enter AS string[20]
 sub procedure pulseout()
   TRISA0_bit = 0      'colocamos el puerto A como salida
   porta.0 =1
   Delay_us(10)
   porta.0 =0
   Delay_us(750)
 end sub
 SUB FUNCTION  pulsein()as word
    dim Tpulse_in as word
     dim x as bit
     Tpulse_in = 0
      do
        TRISA0_BIT = 1
```

```
      x=porta.0
    loop until ((x = 1))
    do
        Tpulse_in = Tpulse_in + 1
        x =  porta.0
    loop until(x = 0)
   result= (Tpulse_in) * 0.0345        'prueba sencilla
end sub
sub procedure LecturaLcd()
        delay_us(200)
        Lcd_Cmd(_LCD_CLEAR)  ' Clear display
        Lcd_Cmd(_LCD_CURSOR_OFF) ' Cursor of
        Lcd_Out(1,1,"PING)) SENSOR DISTANCIA")
        WordToStr( distancia, txt)
        Lcd_Out(2,1,txt)                    'PRUEBA CON ECUACION
        Lcd_Out(2,8,"cm")
end sub
sub procedure LcdOcupado(dim byref TextoOcupado as string[20])
        delay_us(200)
        Lcd_Cmd(_LCD_CLEAR)  ' Clear display
        Lcd_Cmd(_LCD_CURSOR_OFF) ' Cursor of
        Lcd_Out(1,1,"PING)) SENSOR DISTANCIA")
```

```
        Lcd_Cmd(_LCD_CLEAR)  ' Clear display
        Lcd_Out(1,1,TextoOcupado)                    'PRUEBA CON
ECUACION
 end sub
 sub procedure EnvioDatalogger()
 dim j as byte
 TRISA2_bit = 0
 PORTA.2 = 1      'Enviando dato a Datalogger
 LcdOcupado("Envio a Datalogger")
  UART1_Write_Text("INI")              ' sends back text
  UART1_Write(13)
  UART1_Read_Text(txt, "CF", 10)
  UART1_Read_Text(txt, Leer_Enter, 10)
  UART1_Write_Text("CN")              ' sends back text
  UART1_Write(13)
  UART1_Read_Text(txt, "OK", 10)
  UART1_Read_Text(txt, Leer_Enter, 10)
  UART1_Write_Text("Ultrasonido")
  UART1_Write(13)
  UART1_Read_Text(txt, "OK", 10)
  UART1_Read_Text(txt, Leer_Enter, 10)
  UART1_Write_Text("WR")
```

```
   UART1_Write(13)
   UART1_Read_Text(txt, "OK", 10)
   UART1_Read_Text(txt, Leer_Enter, 10)
   delay_ms(100)
   for j = 0 to 30
      PORTD.2 = 0
      WordToStr( memoria_dist[j], txt)
      UART1_Write_Text(txt)
      UART1_Write(13)
      UART1_Read_Text(txt, "OK", 10)
      UART1_Read_Text(txt, Leer_Enter, 10)
      PORTD.2 = 1
      next j
   UART1_Write_Text("END")
   UART1_Write(13)
   UART1_Read_Text(txt, "EF", 10)
   UART1_Read_Text(txt, Leer_Enter, 10)
     PORTA.2 = 0       'Fin dato a Datalogger
end sub
 sub procedure EnvioGlcd()
  dim j as byte
  dim LecturaPrimerBit as byte
```

```
 dim ConverWordtoByte as float
dim ConverWordtoByte1 as byte
 TRISA1_bit = 0
 PORTA.1 = 1       'Enviando dato a GLCD
j = 0
 LcdOcupado("Envio a GLCD")
  while(j < 8)
   do
    if (UART1_Data_Ready() = 1) then
    LecturaPrimerBit = UART1_Read()
    end if
   loop until (LecturaPrimerBit = 0x16)
     UART1_Write(0x021)
     delay_ms(1000)
     PORTD.2 = 0
'      ConverWordtoByte = (memoria_dist[23+j] * 255) / 65536
     ConverWordtoByte1 =  memoria_dist[23+j]
     WordToStr( memoria_dist[23+j], txt)
     Lcd_Out(2,1,txt)                        'PRUEBA CON ECUACION
     Lcd_Out(2,8,"cm")
    '  WordToStr( memoria_dist[23+j], txt)
'      UART1_Write_Text(txt)
```

```
      UART1_Write(ConverWordtoByte1)
      delay_ms(5000)
      j = j+1
    wend
    PORTA.1 = 0
end sub
' PROGRAMACION PRINCIPAL---------------------------------
main:
  OSCCON = 2          'Configración de Oscilador externo
  ANSEL0 = 0          ' Configure other AN pins as digital I/O
  TRISE0_bit = 1      'Envío de Datos al Datalogger
  TRISE1_bit = 1      'Envío de Datos al GLCD
 ' Initialize UART module at 9600 bps
  UART1_Init(9600)
  Delay_ms(100)
 ' Initialize Lcd
  Lcd_Init()
  Leer_Enter[0] = 13   'Dígito ENTER
  Leer_Enter[1] = 0    'Dígito ENTER
  WHILE (true)
  Cont_Data = 0       'Inicialización de Contador de
'                  Datos adquiridos por PING))) sensor
```

```
 PORTD.2 = 0
     DO
       PORTD.2 = 1              'Señala activación del microcontrolador
       distancia = 0            'Inicialización de Distancia del objeto
       pulseout()               'Activa un alto al PING)) sensor
       distancia= pulsein()     'Distancia del objeto
       LecturaLcd()             'Valor de Distancia presenta al LCD
       memoria_dist[Cont_Data] = distancia   'Distancia reservo en una
variable
       Cont_Data = Cont_Data + 1
       if Cont_Data = 31 then
         Cont_Data = 0
       end if
       delay_ms(100)
     LOOP UNTIL((PORTE.0 = 1) or (PORTE.1 = 1) )
     IF  (PORTE.0 = 1) THEN
     EnvioDatalogger()  'Grabación de valores de distancia en Datalogger
     END IF
     IF  (PORTE.1 = 1) THEN
     EnvioGlcd()              'Grabación de valores de distancia en GLCD
     END IF
  WEND
```

end.

3.5 DETALLE DE COMANDOS USADOS

Dentro de las herramientas que nos brinda el software de MikroBasic Pro, se encuentran las herramientas necesarias como: comandos, demos y variables; que permiten al programador diseñar diversos códigos abiertos al desarrollo de funciones de alto nivel.

Una de las cosas más importantes dentro del código es fijar el modo de envío de la información ya sea por medio comunicación serial USART como módulo de conexiones del LCD. El siguiente comando permite la conexión del LCD:

```
' Lcd module connections
dim LCD_RS as sbit at RB4_bit
    LCD_EN as sbit at RB5_bit
    LCD_D4 as sbit at RB0_bit
    LCD_D5 as sbit at RB1_bit
    LCD_D6 as sbit at RB2_bit
    LCD_D7 as sbit at RB3_bit

    LCD_RS_Direction as sbit at TRISB4_bit
    LCD_EN_Direction as sbit at TRISB5_bit
    LCD_D4_Direction as sbit at TRISB0_bit
    LCD_D5_Direction as sbit at TRISB1_bit
    LCD_D6_Direction as sbit at TRISB2_bit
    LCD_D7_Direction as sbit at TRISB3_bit
' End Lcd module connections
```

Como se indicó en el anterior literal, para el funcionamiento del sensor ultrasónico PING))), se envía un pulso mínimo de 2us a 5us de ancho (ver

Figura 11), este pulso es creado por medio del sub proceso **pulseout(),** el cual hace el siguiente procedimiento:

1. Define el pin A0 como salida
2. Coloca el pin A0, en estado alto (5VD) por 5us.
3. Cambia el estado del pin A0 a estado bajo (0VDC), por 750us, según las especificaciones técnicas del sensor PING))).

```
sub procedure pulseout()
  TRISA0_bit = 0       'colocamos el puerto A como salida
  porta.0 =1
  Delay_us(10)
  porta.0 =0
  Delay_us(750)
end sub
```

Para leer la información emitida por el sensor ultrasónico PING))), primero entramos a un lazo **do loop,** luego se llama al sub proceso pulesout, se cambia el pin A0 a modo de entrada, guarda este valor recibido en la variable x y por último para salir de este lazo cuando x es igual 1, ósea la entrada al pin A0 debe ser un alto.

Cuando una señal de alto es captada se procede a dar inicio a un contador, el cual nos va a indicar el tiempo en que se encuentra el pulso de entrada en alto, por ende saldrá del lazo solo cuando la señal llegue a bajo (0V).

Este contador nos va a indicar el tiempo, por consiguiente la distancia que se encuentra el objeto del sensor PING))).

```
SUB FUNCTION  pulsein()as word
    dim   CuentaHasta1 as word
    dim CuentaHasta0 as word
     dim x as bit
     CuentaHasta1=0
     CuentaHasta0=0
      do

'      CuentaHasta1 = CuentaHasta1 + 1
      TRISA0_BIT = 1
      x=porta.0
    loop until ((x = 1))
    TRISA0_BIT = 1
    x=porta.0
    if x = 1 then
      do

      CuentaHasta0 = CuentaHasta0+1
      TRISA0_BIT = 1
      x =  porta.0
      loop until(x = 0)
     end if
  result= (CuentaHasta0) * 0.0345          'prueba sencilla

end sub
```

Estos dos sub procesos permiten encender el sensor y recaudar la información que emite, para luego ser trasformado en distancia.

Para mostrar los datos en la interfaz gráfica se tiene el siguiente proceso que consiste en mostrar los datos de distancia en centímetro y estos datos son transformados a texto para mostrar en el GLCD.

```
sub procedure EnvioGlcd()
dim j as byte
dim LecturaPrimerBit as byte
dim ConverWordtoByte as float
dim ConverWordtoByte1 as byte


TRISA1_bit = 0
PORTA.1 = 1        'Enviando dato a GLCD
j = 0
LcdOcupado("Envio a GLCD")
 while(j < 8)
  do

   if (UART1_Data_Ready() = 1) then
   LecturaPrimerBit = UART1_Read()
   end if
  loop until (LecturaPrimerBit = 0x16)

    UART1_Write(0x021)
    delay_ms(1000)

    PORTD.2 = 0
'    ConverWordtoByte = (memoria_dist[23+j] * 255) / 65536
    ConverWordtoByte1 =  memoria_dist[23+j]

    WordToStr( memoria_dist[23+j], txt)
    Lcd_Out(2,1,txt)                                    'PRUEBA CON
    Lcd_Out(2,8,"cm")
'   WordToStr( memoria_dist[23+j], txt)
'    UART1_Write_Text(txt)
    UART1_Write(ConverWordtoByte1)
    delay_ms(5000)
    j = j+1
  wend
  PORTA.1 = 0


end sub
```

Para la comunicación con el Datalogger se tiene un número de pasos a seguir según como se indica a continuación.

```
sub procedure GrabarDatos()
dim j as byte
  UART1_Write_Text("INI")                  ' sends back text
  UART1_Write(13)
  UART1_Read_Text(txt, "CF", 10)
  UART1_Read_Text(txt, Leer_Enter, 10)

  UART1_Write_Text("CN")                  ' sends back text
  UART1_Write(13)
  UART1_Read_Text(txt, "OK", 10)
  UART1_Read_Text(txt, Leer_Enter, 10)


  UART1_Write_Text("Ultrasonido")
  UART1_Write(13)
  UART1_Read_Text(txt, "OK", 10)
  UART1_Read_Text(txt, Leer_Enter, 10)

  UART1_Write_Text("WR")
  UART1_Write(13)
  UART1_Read_Text(txt, "OK", 10)
  UART1_Read_Text(txt, Leer_Enter, 10)

  delay_ms(100)

  for j = 0 to 30
      PORTD.2 = 0
      WordToStr( memoria_dist[j], txt)
      UART1_Write_Text(txt)
      UART1_Write(13)
      UART1_Read_Text(txt, "OK", 10)
      UART1_Read_Text(txt, Leer_Enter, 10)
      PORTD.2 = 1
      next j

  UART1_Write_Text("END")
  UART1_Write(13)
  UART1_Read_Text(txt, "EF", 10)
  UART1_Read_Text(txt, Leer_Enter, 10)

end sub
```

3.6 IMPLEMENTACIÓN

Para implementar nuestro proyecto realizamos el diseño del circuito impreso en el programa de ARES 7 Professional, previamente simulando el circuito en Proteus ISIS, como se puede apreciar en el **Anexo 6, 7 y 8.**

CAPÍTULO 4

4 SIMULACIÓN, PRUEBAS Y DATOS EXPERIMENTALES

4.1 Datos obtenidos en la simulación de Basic Stamp

Para verificar el funcionamiento del sensor PING))) de Parallax, realizamos pruebas en el BASIC Stamp HomeWork Board y adquirimos los siguientes datos de distancia. Ver **Anexo 1**

Frecuencia (Hz)	Distancia Real (cm)	Distancia Experimental (cm)
63	**330**	330
62	**313**	313
2.5k	**3**	3
5k	**1**	1

TABLA 3: Adquisición de datos con BASIC Stamp HomeWork Board

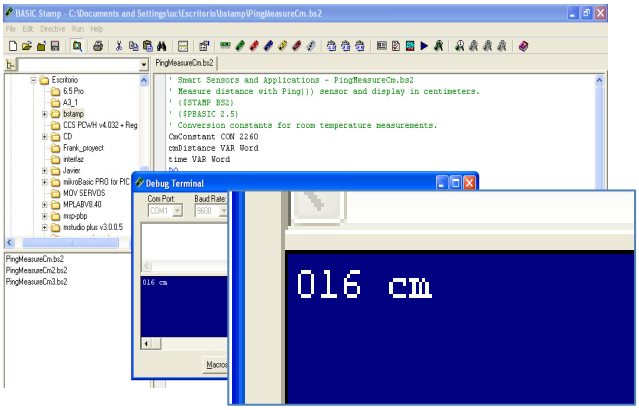


FIGURA 13: Simulación en Basic Stamp

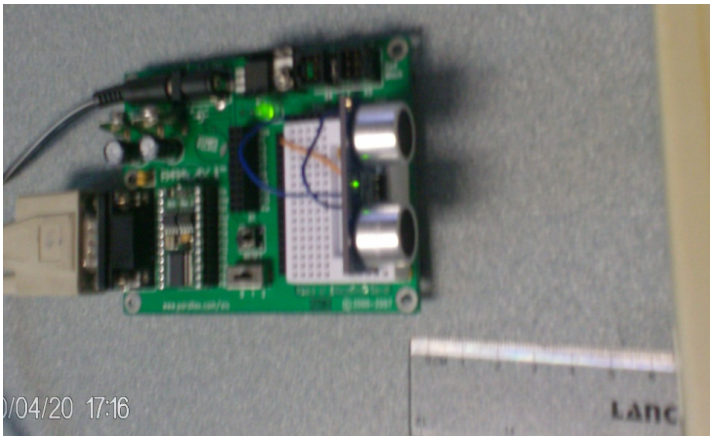


FIGURA 14: Pruebas con el Basic Stamp Protoboard

En la **Figura 16** como podemos ver se encuentra instalado el sensor PING))) en el Basic Stamp Protoboard, según la configuración que nos indica su manual (Ver Anexo 9), se realizaron varias pruebas con diferente distancias para confirmar su funcionamiento.

4.2 Datos obtenidos con MikroBasic Pro y PROTEUS

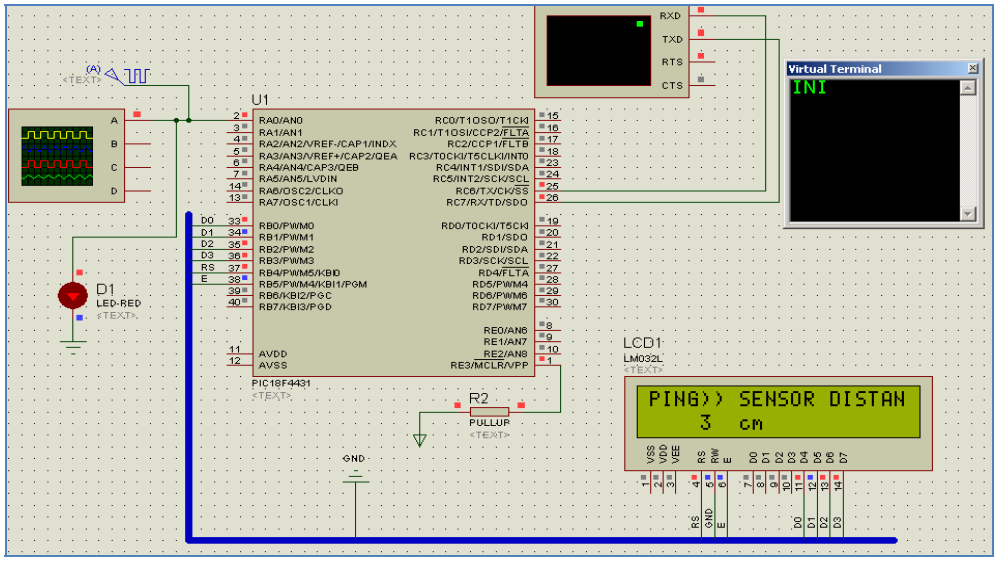


FIGURA 15: Vista del proyecto simulado en PROTEUS

Para verificar el funcionamiento de la programación del microcontrolador, se utilizó el programa Proteus ISIS (Ver **Anexo 2**), donde pudimos simular:

- Los pulsos emitidos por el PIC18F4431 para el funcionamiento del sensor PING))).
- La respuesta del sensor PING))), por medio de una señal de reloj y así comprobar las conversiones que hacemos en la programación del microcontrolador avanzado, obteniendo como resultado los valores de distancia en centímetros por medio de la interfaz gráfica y la comunicación al Datalogger.

4.2.1 PULSOUT

Simulación en PROTEUS

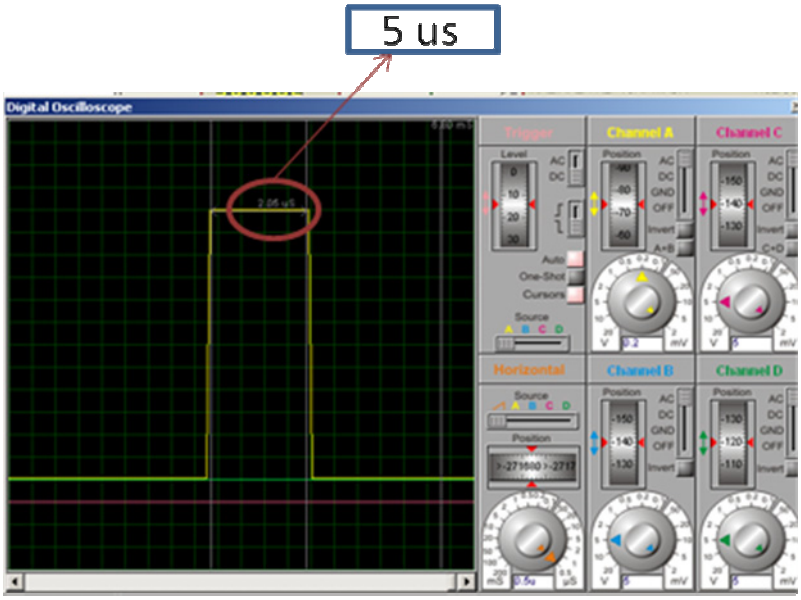


FIGURA 16. a: Simulación en PROTEUS de la señal PULSOUT

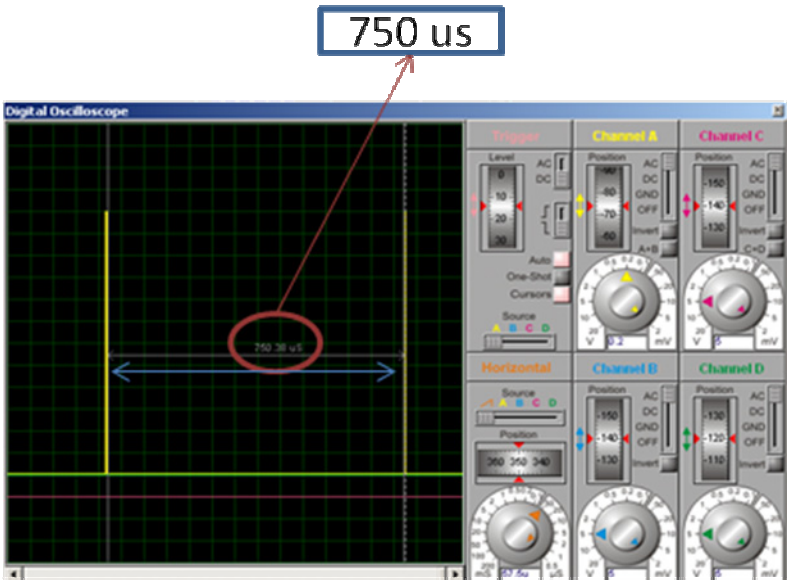


FIGURA 17. b: Simulación en PROTEUS de la señal PULSOUT

Las primeras pruebas fueron realizadas en el simulador de PROTEUS. El objetivo era ver la señal PULSOUT según las especificaciones del sensor ultrasónico PING))), por ende que tenga un ancho de pulso de 5us (Ver Figura 18.a) y que exista 750us 5us (Ver Figura 18.b)de distancia entre cada pulso.

Implementación del PULSOUT

A continuación mostramos el vista en el osciloscopio de la implementación, sin colocar el sensor PING))) para solo ver la señal de salida **PULSOUT**.

FIGURA 18.a: Datos reales del PULSOUT en Osciloscopio

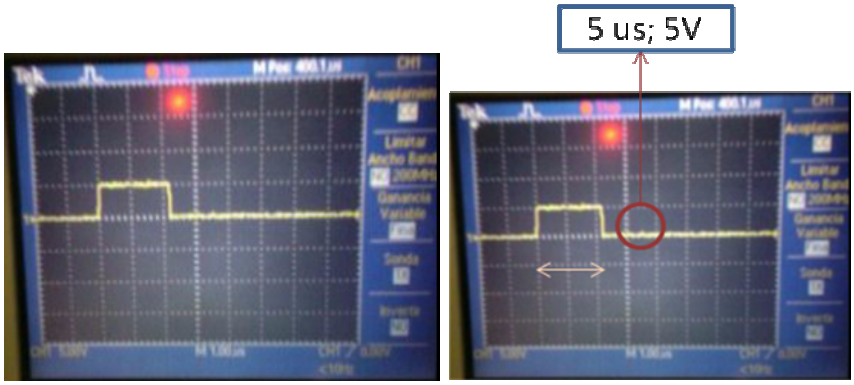


FIGURA 19.b: Acercamiento de datos reales del PULSOUT

Para la simulación de la señal de entrada en PROTEUS y probar nuestro subproceso PULSIN, colocamos un pulso de señal de reloj a la que le variamos la frecuencia, de este modo tendremos diferentes anchos de pulso de entrada que simularía las diferentes distancias.

4.2.2 PULSIN

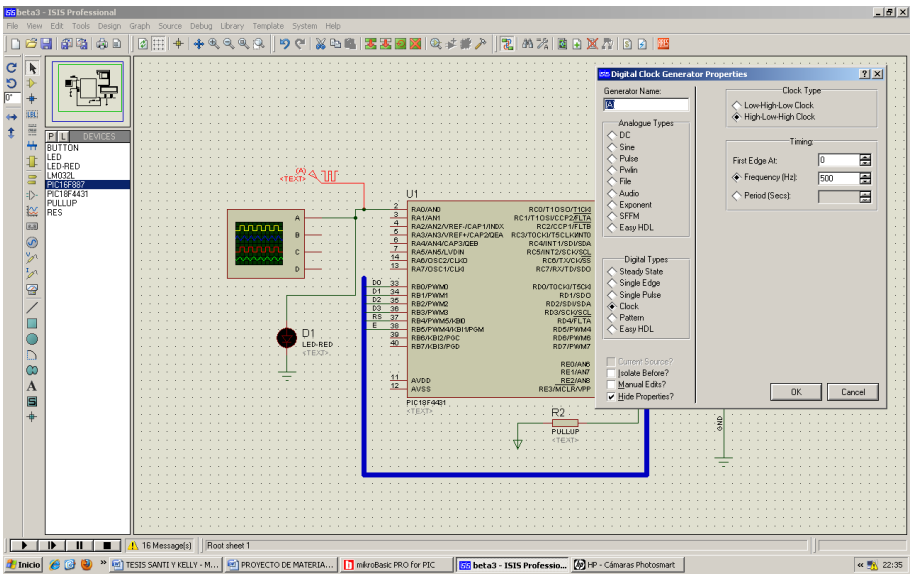


FIGURA 20: Simulación de la señal que envía el sensor PING))) por medio del pulso de reloj en PROTEUS.

En la figura anterior podemos ver que se utilizó una señal de reloj a diferente frecuencia, para simular las diferentes distancias que le debería enviar al microcontrolador, el sensor ultrasónico, ya que PROTEUS ISIS no cuenta con una herramienta virtual que reemplace al sensor PING))).

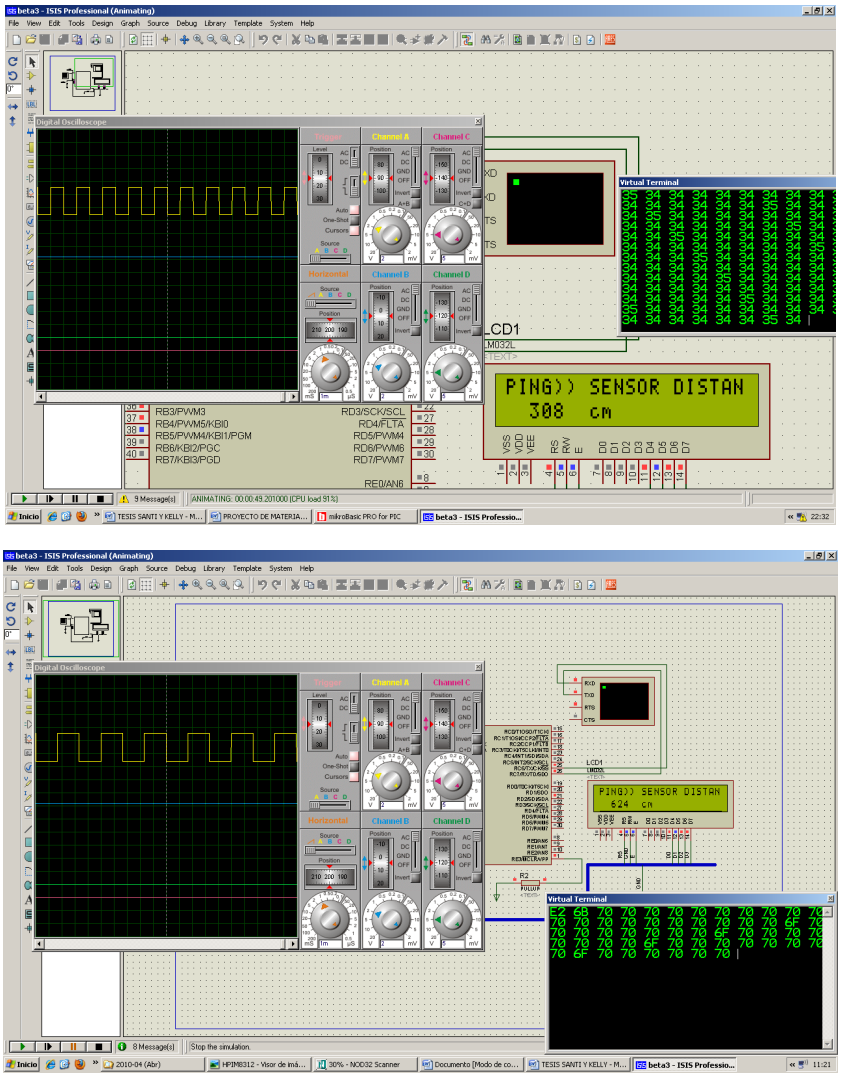


FIGURA 21: Simulación en PROTEUS de distancia con diferentes valores frecuencia de pulso de entrada

En las anteriores figuras podemos ver los diferentes valores de distancia simulada con una señal de entrada de reloj a distintas frecuencias. **Ver Anexo 3.**

Estas simulaciones de las realizaron el programa de PROTEUS ISIS, para estar seguros de los resultados al momento de la implementación.

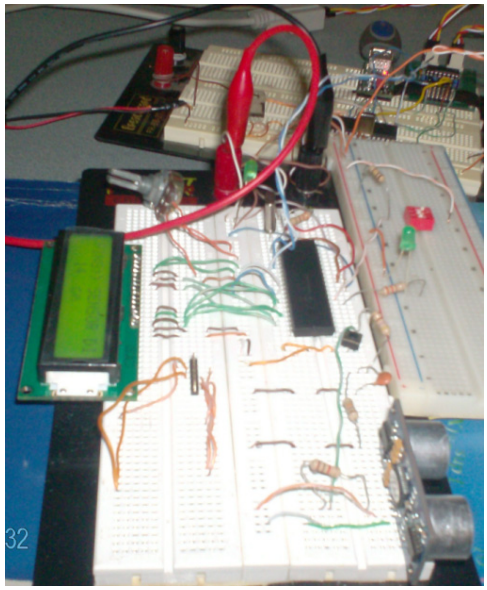


FIGURA 22: Implementación conectando el sensor PING)))

Como podemos observar en la siguiente **Figura 23,** en el pin A0 ó lo que es lo mismo el pin **SIG** del sensor, obtenemos la señal de PULSOUT (5us) y la señal que envía el sensor la cual varía su ancho de pulso según la distancia que se encuentre el objeto, de esta forma si el objeto es cerca tendremos un valor de distancia bajo en el ejemplo siguiente es 91cm.

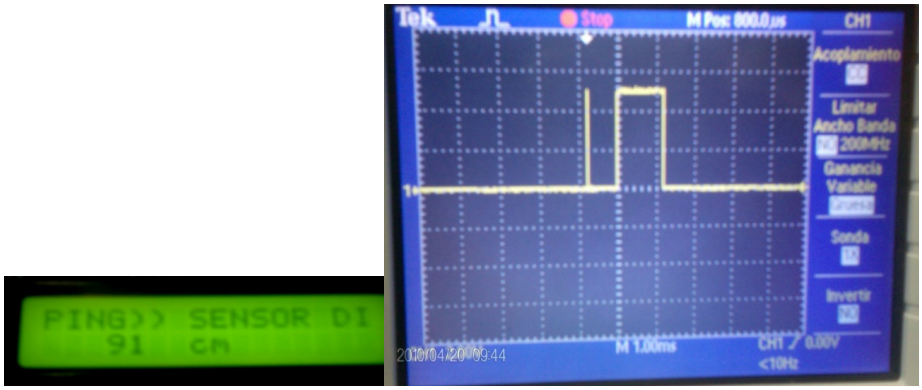


FIGURA 23: Datos reales del PULSIN a 91 cm de distancia

Entre mayor distancia exista entre el sensor y el objeto, es mayor el ancho de pulso de entrada.

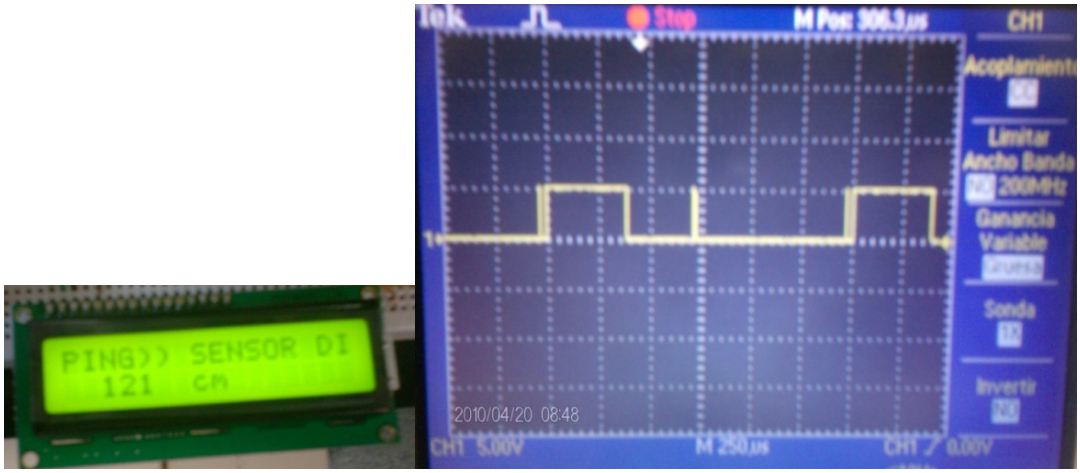


FIGURA 24: Datos reales del PULSIN a 121 cm de distancia

4.2.3 Comunicación serial a DATALOGGER

Primero se realizaron las pruebas de la comunicación con el dispositivo Datalogger, en el programa PROTEUS ISIS, simulado la comunicación con puertos virtuales. Ver **Figura 25.**

Se realizaron la pruebas de comunicación serial entre el microcontrolador avanzado PIC18f4431 y el Datalogger, obteniendo buenos resultados, al grabar nuestra información de diferentes distancias en un memoria extraíble; para posteriormente poder trabajar con estos datos con mayor facilidad, simplemente leyendo los datos de distancia en un archivo tipo texto. Ver **Figura 26** y el **Anexo 4.**

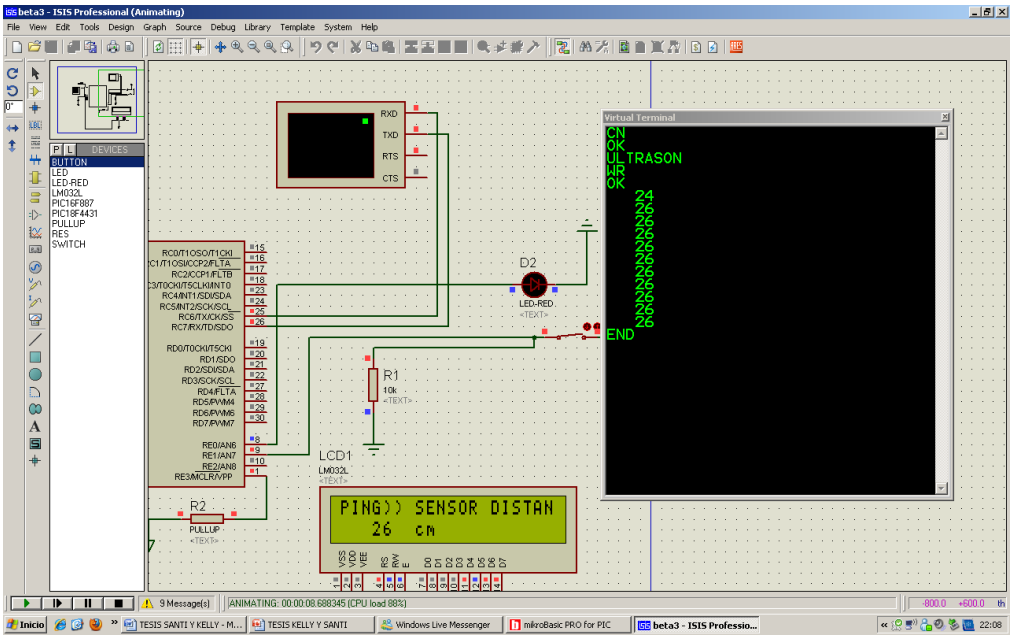


FIGURA 25: Simulación en PROTEUS de datos enviados al DATALOGGER

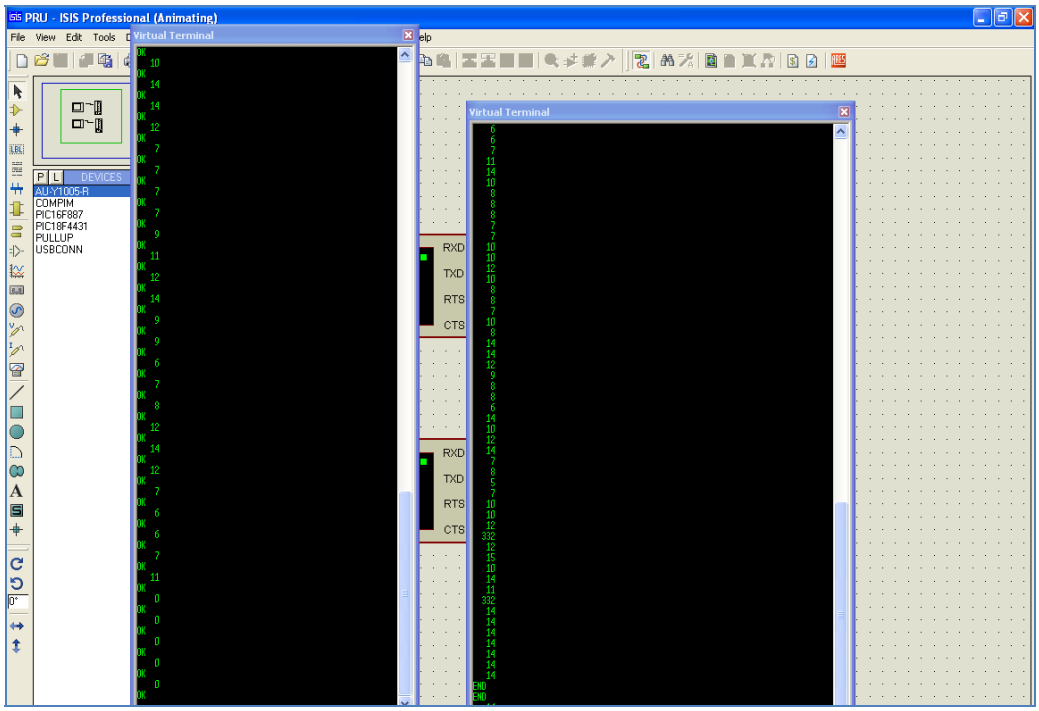


FIGURA 26: Pruebas al enviar datos por medio de la comunicación serial al DATALOGGER

CONCLUSIONES

1) Mediante la experimentación con el sensor ultrasónico PING))) se confirmó el funcionamiento según sus especificaciones técnicas:
    - Su rango de funcionamiento es de 3cm a 3m.
    - Su ángulo de operación está de 0 a 10 grados dentro del radio de 2 pulgadas.
    - El pulso de entrada al sensor PING))) tiene un ancho de pulso de 5us.
2) Logramos cumplir el objetivo de este proyecto que es la utilización del sensor ultrasónico PING))) de Parallax, controlado por medio de un microcontrolador avanzado con lenguajes de programación que son convencionales como: C++, ASM y Micro Basic Pro; de esta manera se pueden tomar datos de distancia de una manera fácil y económico, ya que no será necesario usar la plataforma de programación ni los equipos que la empresa Parallax a diseñado para el uso de este sensor.
3) Con la plataforma de programación de MikroBasic, se construyó la función PULSOUT, por lo cual concluimos que su mínimo ancho de pulso es de 2 us y luego su retardo máximo es de 750us; permitiendo así luego obtener el tiempo de ancho de pulso, calcular el valor de distancia y

presentar los valores en centímetros tanto en el Datalogger como en la interfaz Gráfica GLCD.

4) La función que permite obtener o capturar el tiempo de ancho de pulso se llama PULSIN, que hemos desarrollado también en la plataforma de MikroBasic PRO, por lo que concluimos que es otro de los parámetros muy importante para la medición de distancia, ya que es el tiempo en que viaja la onda ultrasónica a partir de que choca contra el objeto hasta ser receptado por el sensor ultrasónico PING))).
5) Para almacenar los datos en el Datalogger y utilizar la interfaz gráfica GLCD, se emplea la comunicación serial USART por parte del microcontrolador; concluimos que para ser enviado primero, necesitamos guardar los valores de distancia en el microcontrolador y luego procedemos a transmitirlos por la función UART de MikroBasic PRO, ya que el tiempo de transmisión y recepción del protocolo serial USART es mucho mayor al tiempo que el microcontrolador tarda en tomar los toma los valores de distancia del objeto.

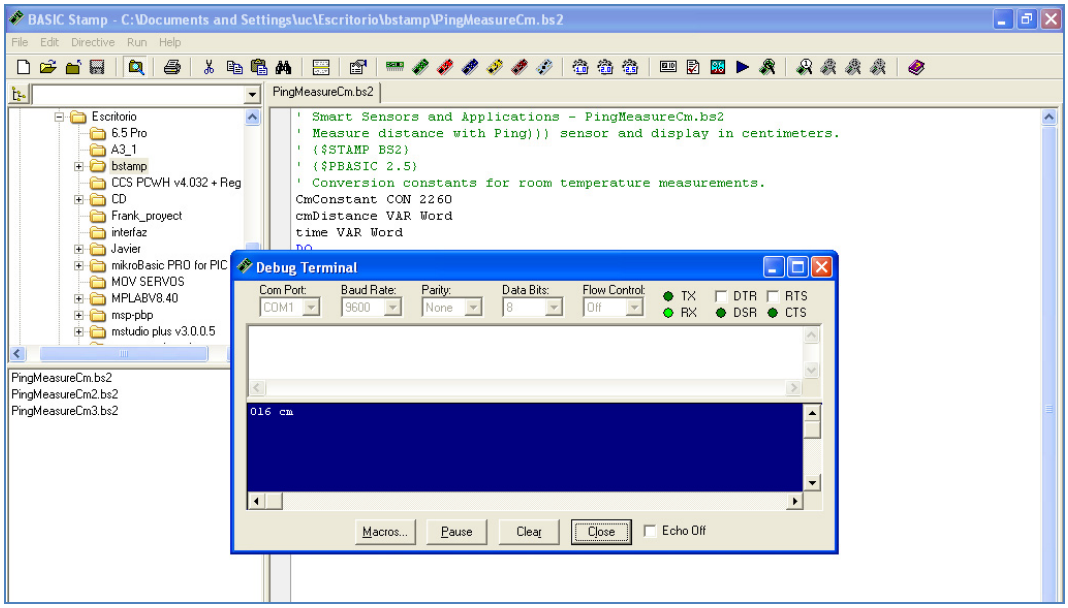
RECOMENDACIONES

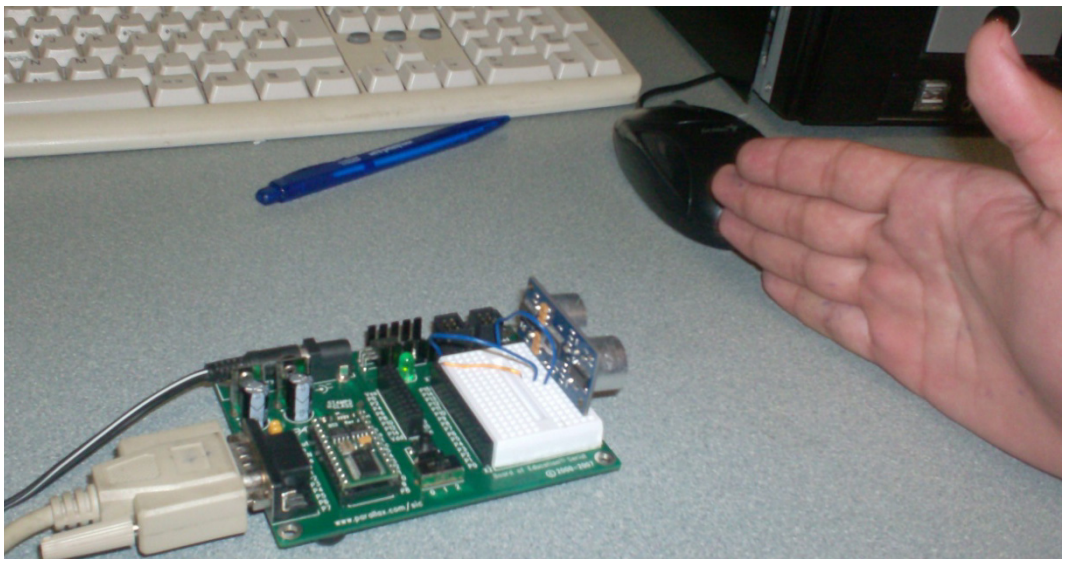
1) Verificar que el objeto esté colocado a una distancia mínima de detección de 3 cm y la máxima de 3m.
2) El sensor, debe ser colocado a 10 cm de altura sobre una mesa y apuntando en sentido horizontal. Esto se debe a que la emisión del sensor Ping))) se expande unos grados, incluso en sentido vertical (aunque menos que en sentido horizontal).
3) El sensor debe estar lo más estable y direccionado con respecto al objeto a medir.
4) Verificar que el objeto a medir tiene superficie lisa, regular y no porosa ya que los datos de distancia varían según el tipo de material a detectar.
5) Se debe cuidar y estar atentos al factor de la temperatura, ya que la velocidad del sonido varía con la misma y esto nos proporciona datos no precisos, este proyecto está configurado a una temperatura ambiente de 22 grados centígrados.

ANEXOS

ANEXO 1

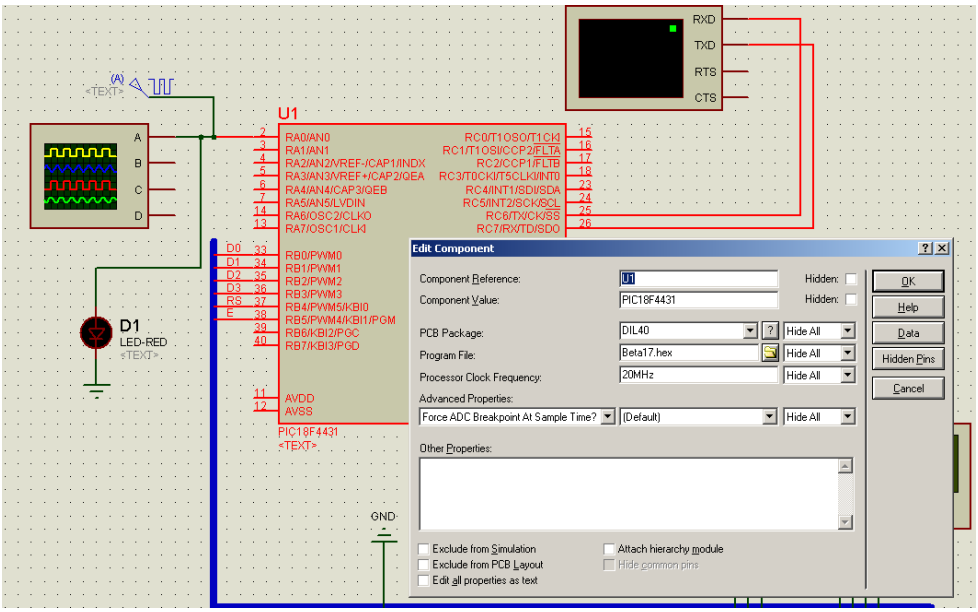
SIMULACIÓN CON EL SENSOR ULTRASÓNICO PING))) CON LOS EQUIPOS Y PLATAFORMA DE PARALLAX

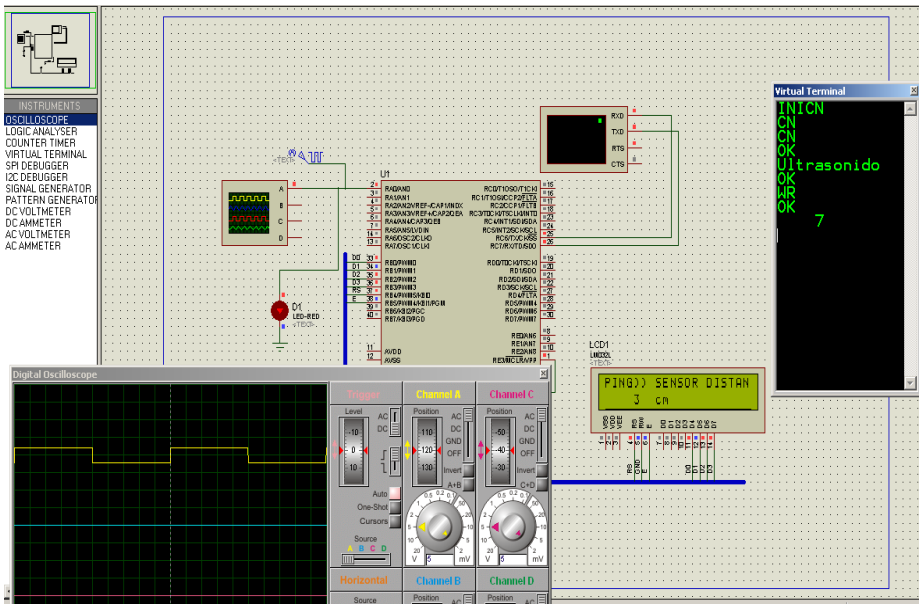




ANEXO 2

SIMULACIÓN DEL PROYECTO EN PROTEUS: CONFIGURACIÓN DEL PIC18F4431



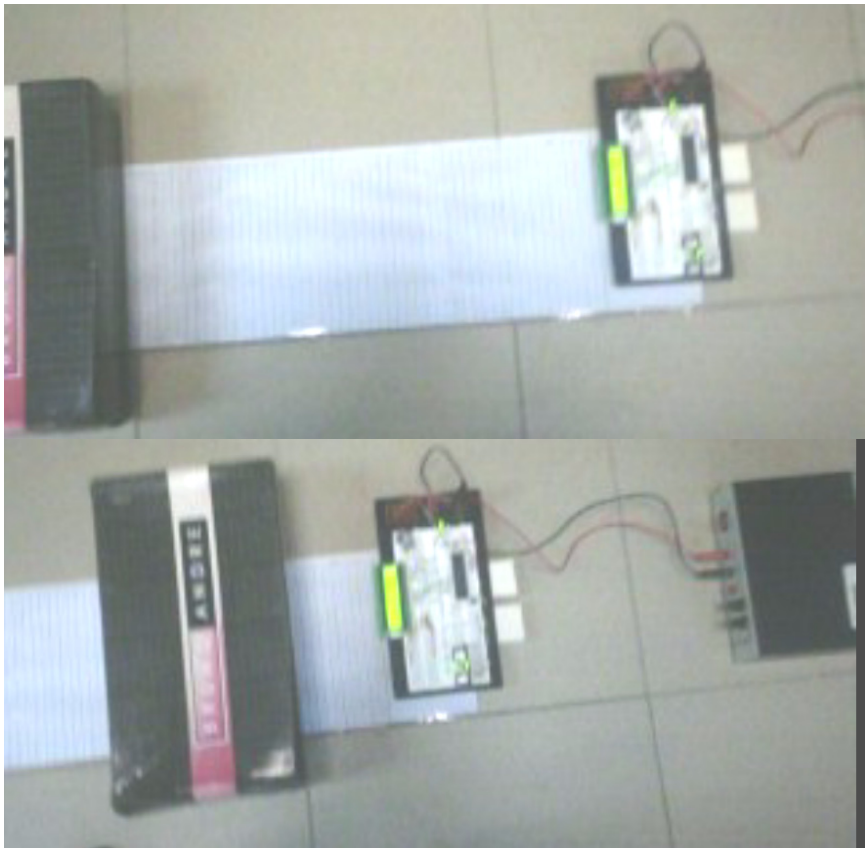


ANEXO 3

DATOS DE LAS PRUEBAS REALIZAS CON EL SENSOR A

DIFERENTE TEMPERATURA

Datos	Medición 22˚C	Medición 30˚C
3	3	5
10	10	12
20	20	23
30	30	32
40	40	44
50	50	52
60	60	63
70	70	72
80	80	83
90	90	94
100	100	103
110	110	115
120	120	124
130	130	133
140	140	143
150	150	154
160	160	163
170	170	173



ANEXO 4

DATOS GUARDADOS EN EL DATALOGGER

ULTRASON.txt

98- 98- 98- 98- 98- 98- 98- 98- 98- 98- 98-

98- 98- 98- 98- 98- 98- 98- 98- 98- 98- 98-

98- 98- 98- 98- 98- 98- 98- 98- 98- 24- 23-

24- 24- 24- 24- 24- 24- 24- 24- 24- 104- 104-

104- 104- 104- 104- 104- 104- 104- 24- 23- 23- 23-

24- 24- 24- 104- 104- 104- 104- 104- 104- 104- 104-

105- 105- 105- 105- 105- 105- 105- 105- 105- 105- 105-

106- 106- 106- 106- 106- 106- 106- 106- 106- 10- 10-

10- 10- 10- 10- 105- 105- 105- 105- 106- 105- 106-

109- 109- 109- 106- 106- 106- 106- 106- 106- 105- 106-

105- 105- 105- 105- 105- 105- 105- 105- 105- 105- 105-

105- 105- 105- 105- 105- 105- 105- 105- 105- 105- 105-

105- 105- 105- 105- 105- 105- 105- 105- 105- 105- 105-

105- 105- 105- 105- 105- 105- 106- 104- 104- 106- 109-

109- 104- 105- 105- 105- 105- 109- 109- 109- 109- 106-

105- 106- 105- 105- 14- 7- 6- 5- 4- 3- 3-

3- 3- 3- 3- 3- 3- 3- 3- 3- 2- 3- 3-

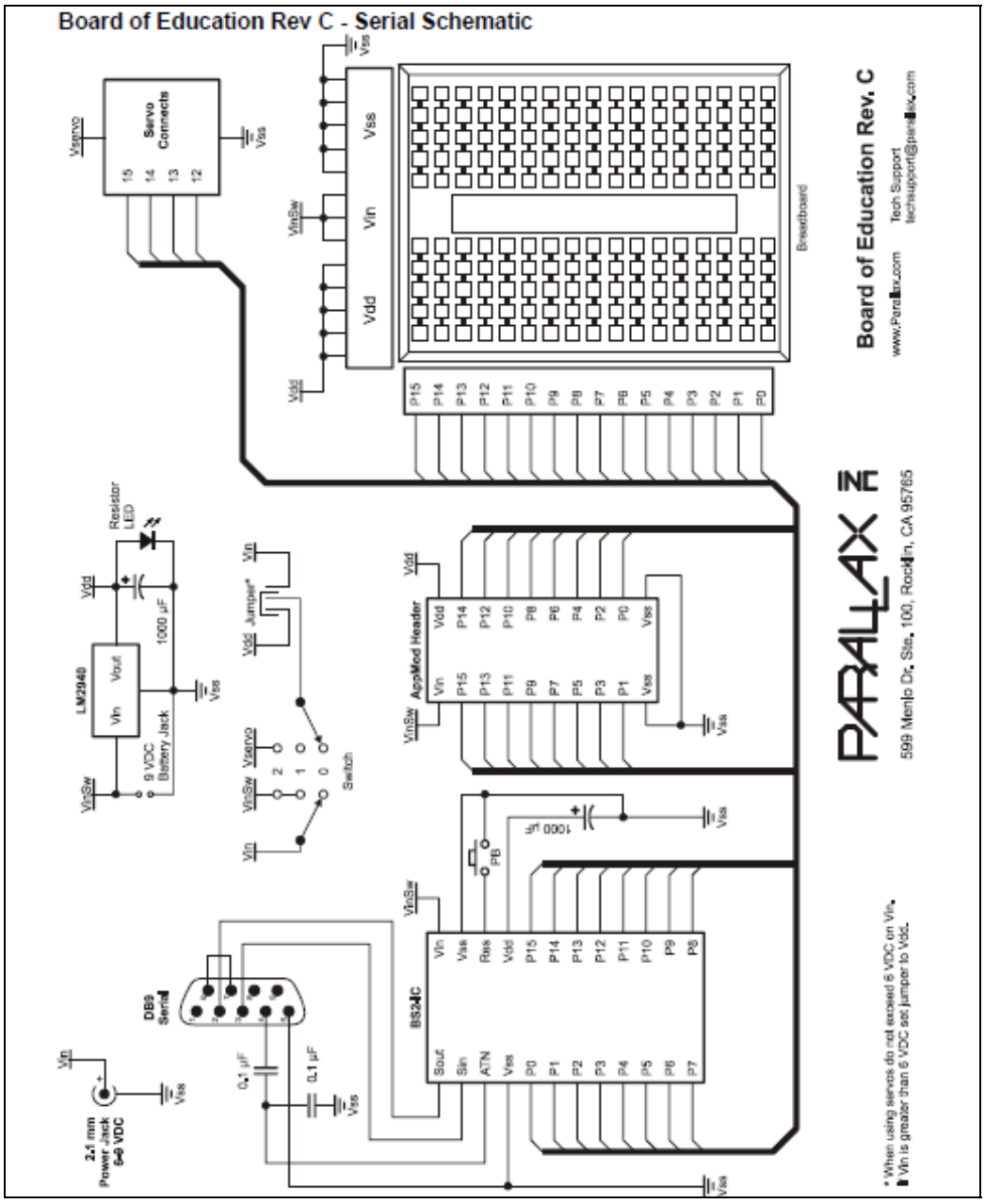
3- 3- 4- 5- 105- 15- 104- 104- 16- 16- 16- 10-

104- 7- 7-

ANEXO 5

DIAGARAMA ESQUEMATICO DEL BASIC STAMP HOMEWORK

BOARD DE PARALLAX

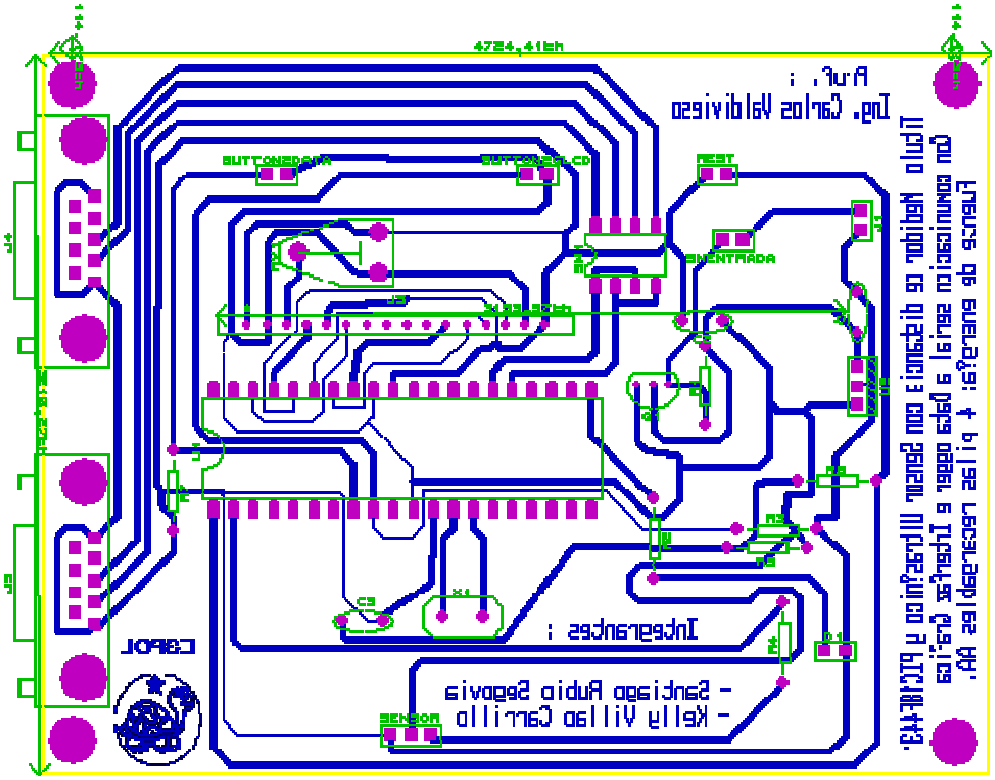


ANEXO 6

DIAGARAMA ESQUEMATICO EN PROTEUS DEL PCB

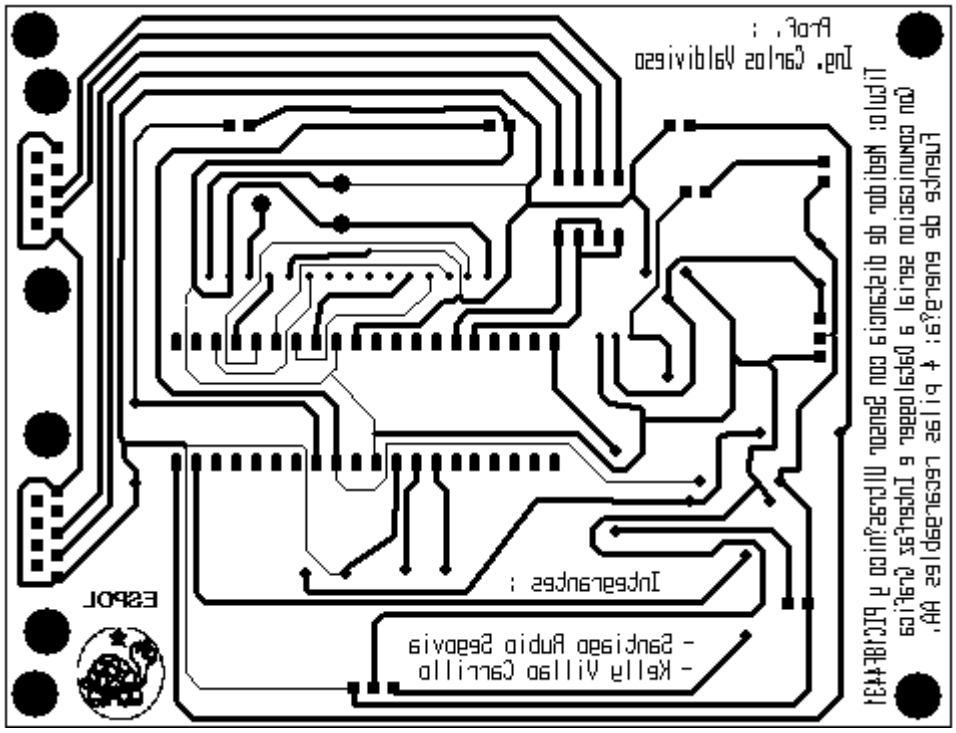
ANEXO 7

DIAGARAMA ESQUEMATICO DEL CIRCUITO IMPRESO



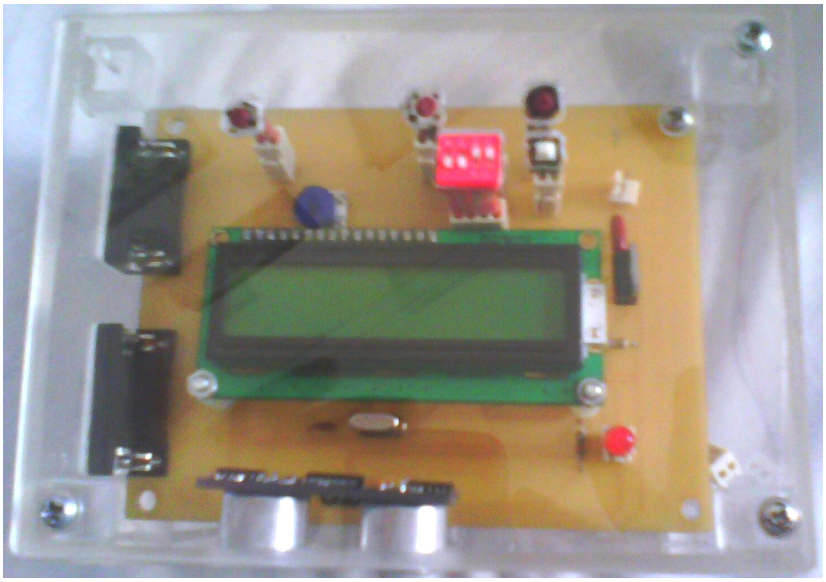
DIAGARAMA ESQUEMATICO DEL CIRCUITO

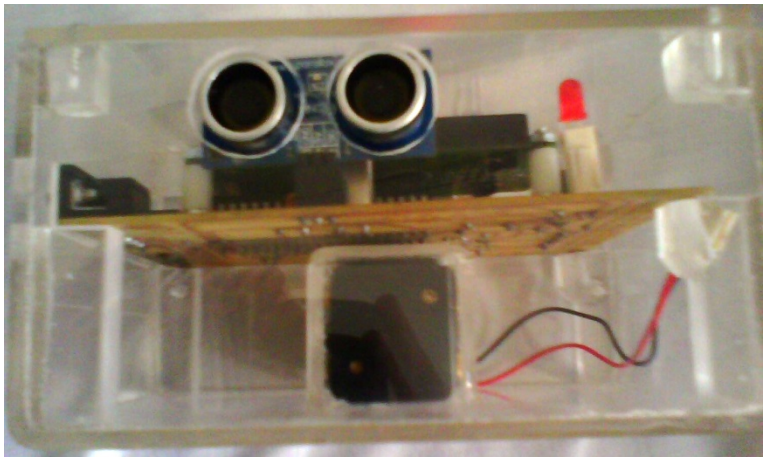
IMPRESO



Lista de componentes

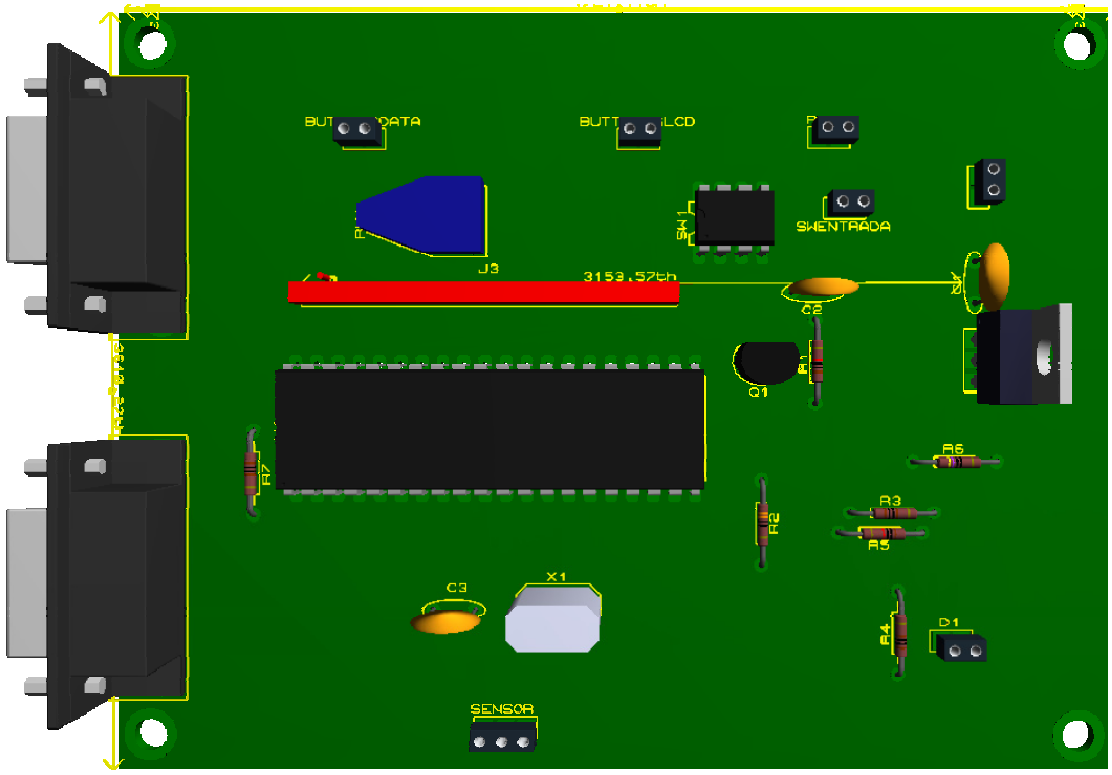
Cantidad	Componentes	Valores
1	Sensor ultrasónico PING))) Parallax	
1	PIC	18F4431
2	DB9	
1	Bloque de 4 switch	
3	Botoneras	
1	Switch on/off	
1	Led rojo	
1	Zócalo de 40 pines	
1	LCD 2x16	
1	Transistor NPN	2N3906
1	Regulador	7805
1	Capacitor	330p
1	Capacitor	100u
1	Capacitor	10n
2	Resistencia	10k
3	Resistencia	1k
1	Resistencia	470
1	Resistencia	330
1	Cristal	20MHz
1	Potenciómetro	1k

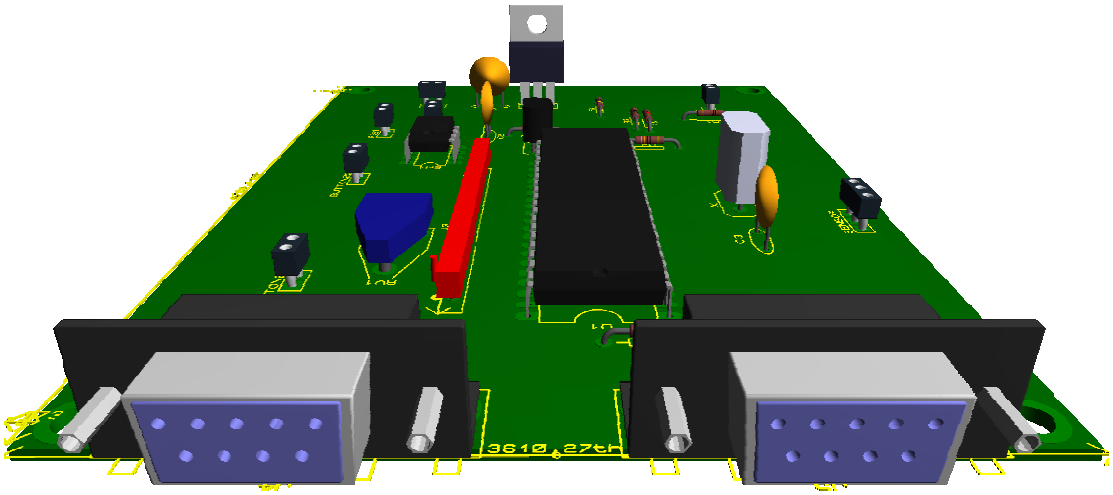




ANEXO 8

VISTAS 3D DEL CIRCUITO IMPRESO





ANEXO 9

SENSORES INTELIGENTES Y APLICACIONES DE PARALLAX

CAPITULO 2

The Ping))) Ultrasonic Distance Sensor

Chapter 2: The Ping))) Ultrasonic Distance Sensor

2

The Ping))) sensor interfaced with a BASIC Stamp can measure how far away objects are. With a range of 3 centimeters to 3.3 meters, it's a shoo-in for any number of robotics and automation projects. It's also remarkably accurate, easily detecting an object's distance down to the centimeter.

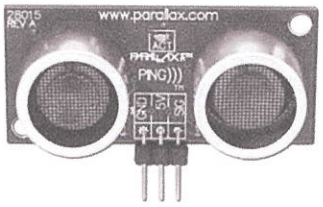
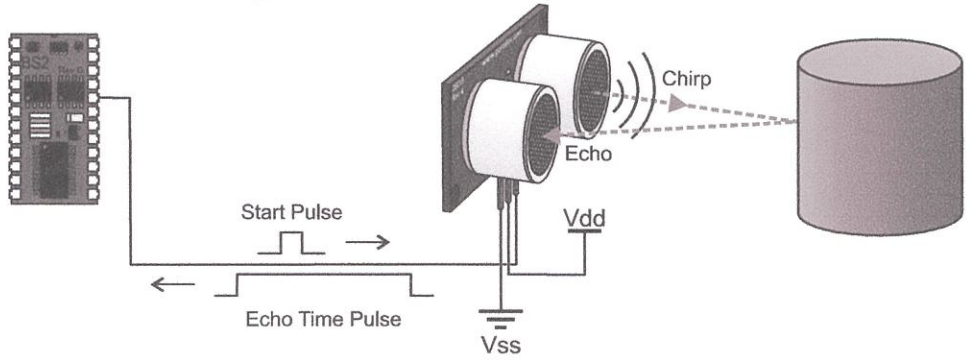


Figure 2-1
The Ping)))™ Ultrasonic Distance Sensor

HOW DOES THE PING))) SENSOR WORK?

Figure 2-2 shows how the Ping))) sensor sends a brief chirp with its ultrasonic speaker and measures the echo's return time to its ultrasonic microphone. The BASIC Stamp starts by sending the Ping))) sensor a pulse to start the measurement. Then, the Ping))) sensor waits long enough for the BASIC Stamp program to start a `PULSIN` command. Then, at the same time the Ping))) sensor chirps its 40 kHz tone, it sends a high signal to the BASIC Stamp. When the Ping))) sensor detects the echo with its ultrasonic microphone, it changes that high signal back to low.

Figure 2-2: How the Ping))) Sensor Works



The BASIC Stamp `PULSIN` command uses a variable to store how long the high signal from the Ping))) sensor lasted. This time measurement is how long it took sound to travel to the object and back. Using this measurement and the speed of sound in air, you can make your program calculate the object's distance in centimeters, inches, feet, etc.

The Ping))) sensor's chirps are not audible because 40 kHz is ultrasonic.

What we consider sound is our inner ear's ability to detect the variations in air pressure caused by vibration. The rate of these variations determines the pitch of the tone. Higher frequency tones result in higher pitch sounds and lower frequency tones result in lower pitch tones.

Most people can hear tones that range from 20 Hz, which is very low pitch, to 20 kHz, which is very high pitch. Subsonic is sound with frequencies below 20 Hz, and ultrasonic is sound with frequencies above 20 kHz. Since the Ping))) sensor's chirps are at 40 kHz, they are definitely ultrasonic, and not audible to people.

ACTIVITY #1: MEASURING ECHO TIME

In this activity, you will test the Ping))) sensor and verify that it gives you echo time measurements that correspond to an object's distance.

Parts Required

(1) Ping))) Ultrasonic Distance Sensor
(3) Jumper Wires

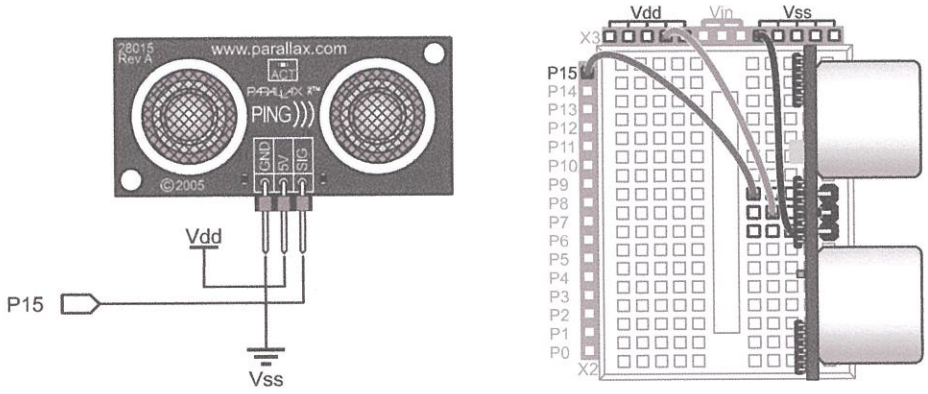
All you'll need is a Ping))) sensor and three jumper wires to make it work. The Ping))) sensor has protection against programming mistakes (and wiring mistakes) built-in, so there's no need to use a 220 Ω resistor between P15 and the Ping))) sensor's SIG terminal.

Ping))) Sensor Circuit

Figure 2-3 shows a schematic and wiring diagram for testing the Ping))) sensor.

√ Build the circuit.

Figure 2-3: Ping))) Sensor Schematic and Wiring Diagram



Testing the Ping))) Sensor

As mentioned earlier, the Ping))) sensor needs a start pulse from the BASIC Stamp to start its measurement. A pulse to P15 that lasts 10 µs (`PULSOUT 15, 5`) is easily detected by the Ping))) sensor, and it only takes a small amount of time for the BASIC Stamp to send. A `PULSIN` command that stores the duration of the Ping))) sensor's echo pulse (`PULSIN 15, 1, time`) has to come immediately after the `PULSOUT` command. In this example, the result the `PULSIN` command stores in the variable `time` is the round trip time for the Ping))) sensor's chirp to get to the object, reflect, and return.

Example Program - PingTest.bs2

You can test this next program by measuring the distances of a few close-up objects. For close-up measurements, the Ping))) sensor only needs to be 3 to 4 inches (roughly 8 to 10 cm) above your working surface. However, if you are measuring objects that are more than a half a meter away, you may need to elevate your Ping))) sensor to prevent echoes from the floor registering as detected objects.

- √ Place your Board of Education with the Ping))) sensor circuit on something to keep it at least 8 cm above the table surface.
- √ Place an object (like a water bottle, box, or paper target) 15 cm from the front of the Ping))) sensor.
- √ Enter, save, and run PingTest.bs2.
- √ The Debug Terminal should start reporting a value in the 400 to 500 range.

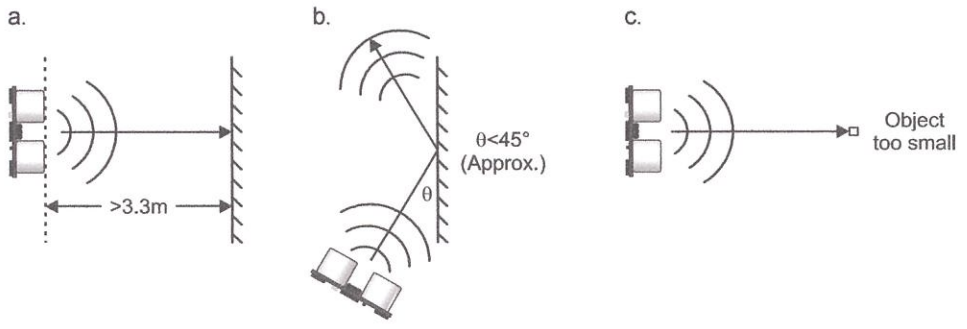
- √ Move the target to a distance of 30 cm from the Ping))) sensor and verify that the value of the time variable roughly doubled.
- √ Point your Ping))) sensor at a variety of near and far objects, and observe the time measurements.

```
' Smart Sensors and Applications - PingTest.bs2
' Tests the Ping))) ultrasonic distance sensor

' {$STAMP BS2}
' {$PBASIC 2.5}

time VAR Word

DO
  PULSOUT 15, 5
  PULSIN 15, 1, time

  DEBUG HOME, "time = ", DEC5 time

  PAUSE 100
LOOP
```

Your Turn - Testing Range, Angle and Object Size

In terms of accuracy and overall usefulness, ultrasonic distance detection is really great, especially compared to other low-cost distance detection systems. That doesn't mean that the Ping))) sensor is capable of measuring "everything". Figure 2-4 shows a few situations that the Ping))) is not designed to measure: (a) distances over 3 meters, (b) shallow angles, and (c) objects that are too small.

Figure 2-4: The Ping))) Sensor is Not Designed for these Situations:



In addition, as Ken Gracey of Parallax Inc. discovered during a classroom demonstration at his son's school, some objects with soft, irregular surfaces (such as stuffed animals) will absorb rather than reflect sound and therefore can be difficult for the Ping))) sensor to detect. Objects with smooth surfaces that readily reflect sound are easier for the sensor to detect.

- √ Try pointing the Ping))) sensor at various objects that are different distances away. What's the largest value the Ping))) sensor returns? How close do you have to get to the object before the time measurements start to decrease?
- √ Try standing one meter away from the wall, and point the Ping))) sensor at it, and record the measurement. Next, try pointing the Ping))) sensor at the wall at different angles, as shown in Figure 2-5. Do the values change? At what angle does the Ping))) sensor cease to detect the wall?

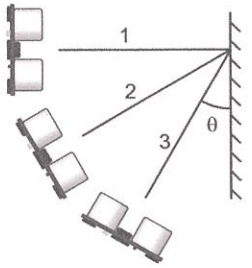


Figure 2-5
Determining the Minimum Angle of Detection

- √ Try hanging various objects from the ceiling at about 1.5 meters from the Ping))) sensor. How small can the object be? Does shape or angle matter? Does the size requirement change at 0.5 meters?
- √ Try detecting objects of similar size but made from different materials, such as a cardboard shoebox and a fuzzy slipper, to see if you have a smaller effective range with sound-absorbing objects. Can you find any objects invisible to the Ping))) sensor? How about a wad of cotton balls, or of tulle netting?

ACTIVITY #2: CENTIMETER MEASUREMENTS

This activity demonstrates how to use the speed of sound and the PBASIC Multiply High operator (**) to calculate the distance of an object based on the echo time measurement from the Ping))) sensor.

Calculating Centimeter Distance with PBASIC

The equation for the distance sound travels is $S = C_{air}\ t$, where S is distance, C_{air} is the speed of sound in air, and t is time. Since the Ping))) sensor's time measurement is the time it takes sound to get to the object and bounce back, the actual distance, S_{object}, is half of the total distance the sounds travels.

$$S = C_{air} t$$

$$S_{object} = \frac{S}{2} = \frac{C_{air} t}{2}$$

The speed of sound in air is most commonly documented in terms of meters per second (m/s). However, centimeter (cm) measurements will be more convenient to calculate with the BASIC Stamp. Since there are 100 centimeters in a meter, let's use $S_{object\text{-}cm}$ which is simply 100 times S_{object}. The **PULSIN** *Duration* measurement units for the BASIC Stamp 2 are 2/1,000,000 of a second (2 µs). So, instead of t, which has to be a measurement of seconds, we'll use $t_{PULSIN\text{-}BS2}$. When multiplied by 2/1,000,000 $t_{PULSIN\text{-}BS2}$ gives us the number of seconds. There is a pair of 2s in the numerator and denominator that cancel, and 100 in the numerator can cancel with two of the zeros in the denominator's 1,000,000. The result of these substitutions and cancellations is $S_{object\text{-}cm} = (C_{air}\ t_{PULSIN\text{-}BS2})/10{,}000$.

$$S_{object\text{-}cm} = \frac{100\, C_{air} t}{2}$$

$$S_{object\text{-}cm} = \frac{100\, C_{air} t_{PULSIN\text{-}BS2}}{2} \times \frac{2}{1{,}000{,}000}$$

$$S_{object\text{-}cm} = \frac{C_{air} t_{PULSIN\text{-}BS2}}{10{,}000}$$

The speed of sound in air at room temperature 72 °F (22.2 °C) is 344.8 m/s. Dividing 10,000 into this leaves us with $S_{object\text{-}cm} = 0.03448\ t_{PULSIN\text{-}BS2}$.

$$S_{object\text{-}cm} = \frac{344.8\ t_{PULSIN\text{-}BS2}}{10{,}000}$$

$$= 0.03448\ t_{PULSIN\text{-}BS2}$$

The BASIC Stamp can use the `**` operator to multiply a variable that stores the `PULSIN` command's *`Duration`* measurement by a fractional value that's less than 1. For example, if the `PULSIN` command stores the echo measurement in the `time` variable, this command will store the centimeter distance result in the `cmDistance` variable:

```
cmDistance = CmConstant ** time
```

With the `**` operator, `CmConstant` will have to be 2260, which is the `**` equivalent of 0.03448. Instead of a decimal denominator, like 10,000 (in the case of 0.03448), the `**` operator needs a value that would go in the numerator of a fraction with a denominator of 65536. To get that numerator, multiply your fractional value by 65536.

$$CmConstant = 0.03448 \times 65536 = 2260$$

Now, we've got the value we need to modify PingTest.bs2 so that it will measure centimeter distance. We'll also add a variable to store distance (`cmDistance`) along with the constant to store the value 2260 (`CmConstant`).

```
CmConstant   CON   2260

cmDistance   VAR   Word
```

Then, the `**` calculation can be added to the PingText.bs2's `DO...LOOP` to calculate the centimeter measurement. The `DEBUG` command in the program can then be modified to display the measurement.

```
cmDistance = CmConstant ** time

DEBUG HOME, DEC3 cmDistance, " cm"
```

Example Program: PingMeasureCm.bs2

- √ Enter, save and run PingMeasureCm.bs2.
- √ Move the target object until the measurement displays 20 cm.
- √ Align your ruler with that measurement. The 0 cm mark should align somewhere with the Ping))) sensor, typically somewhere between the printed circuit board and the front-most part of the speaker/microphone.
- √ Now, experiment with other distance measurements.

```
' Smart Sensors and Applications - PingMeasureCm.bs2
' Measure distance with Ping))) sensor and display in centimeters.

' {$STAMP BS2}
' {$PBASIC 2.5}

' Conversion constants for room temperature measurements.
CmConstant  CON   2260

cmDistance  VAR   Word
time        VAR   Word

DO

  PULSOUT 15, 5
  PULSIN 15, 1, time

  cmDistance = CmConstant ** time

  DEBUG HOME, DEC3 cmDistance, " cm"

  PAUSE 100

LOOP
```

Your Turn - Verifying the Calculations

Let's verify that that the program is correctly calculating the distance.

- √ Modify PingMeasureCm.bs2 so that it displays the values of both the time and the distance variables.
- √ Use a calculator to verify that you get the same result from the distance equation that you do from the program.

$$S_{object-cm} = 0.03448 \times t_{PULSIN-BS2}$$

BIBLIOGRAFÍA

[1]. Parallax; Tutorial Parallax, **Smart Sensors and Applications**, versión 1.0

http://www.parallax.com/Education/TutorialsTranslations/tabid/535/Default.aspx; **Fecha de consulta:** 23 de Abril – 28 de Mayo del 2010.

[2]. Robots; Sensores - **Medidores de distancia ultrasónicos** - Sensores de ultrasonido Descripción y funcionamiento; Eduardo J. Carletti;

http://axxon.com.ar/rob/Sensores_ultrasonido.htm; **Fecha de consulta:** 20 de Abril – 28 de Mayo del 2010.

[3]. Robots; **Sensores Prueba del sensor PING))) de distancia por ultrasonidos de Parallax manejado con PIC**; Eduardo J. Carletti

http://axxon.com.ar/rob/Prueba_SensorPing.htm; **Fecha de consulta:** 20 de Abril – 28 de Mayo del 2010.

[4]. Parallax; **BASIC Stamp HomeWork Board**,

http://www.parallax.com/html_pages/edu; **Fecha de consulta:** 16 de Abril – 28 de Mayo del 2010.

[4]. Parallax; **Board of Education (Serial) - Full Kit**

http://www.parallax.com/Store/Education/KitsandBoards/tabid/

182/CategoryID/67/List/0/SortField/0/Level/a/ProductID/300.aspx; **Fecha de consulta:** 16 de Abril – 05 de Mayo del 2010.

[5]. Parallax ; **Board Of Education® Rev C - Serial (#28150)** http://www.parallax.com/Store/Education/KitsandBoards/tabid/182/CategoryID/67/List/0/SortField/0/Level/a/ProductID/300.aspx; **Fecha de consulta:** 16 de Abril – 23 de Mayo del 2010.

[6]. Parallax; **PING)))™ Ultrasonic Distance Sensor (#28015);** http://www.parallax.com/dl/docs/prod/acc/28015-PING-v1.3.pdf; **Fecha de consulta:** 16 de Abril – 23 de Mayo del 2010.

[7]. Robots; Sensores **- Medidores de distancia ultrasónicos - Sensores de ultrasonido**; Eduardo J. Carletti; http://axxon.com.ar/rob/Sensores_ultrasonido.htm; **Fecha de consulta:** 16 de Abril – 02 de Mayo del 2010.

[8]. Robots; **Sensores - Conceptos generales. Descripción y funcionamiento;** Eduardo J. Carletti;

http://axxon.com.ar/rob/Sensores_general.htm; **Fecha de consulta:** 16 de Abril – 02 de Mayo del 2010.

[9]. Neoteo; **Manual de MikroBasic Pro**; Ariel Palazzesi http://www.neoteo.com/mikrobasic-primera-parte.neo ; **Fecha de consulta:** 16 de Abril – 20 de Mayo del 2010.

[10]. Grupos de Investigación de Robótica; **¿Cómo medir el ángulo de un obstáculo?**; Aristides Alvarez

http://gruposrobotica.ning.com/group/ultrasonido/forum/topics/como-medir-el-angulo-de-un-1; **Fecha de consulta:** 16 de Abril – 20 de Mayo del 2010.