ESCUELA SUPERIOR POLITÉCNICA DEL LITORAL

FACULTAD DE INGENIERÍA EN ELECTRICIDAD Y COMPUTACIÓN

"Uso del Sonar MaxSonar EZ1 en interfaz con el robot Pololu 3pi en optimización de rutinas para la detección de obstáculos"

TESINA DE SEMINARIO

Previa a la obtención de los Títulos de:

INGENIERO EN ELECTRÓNICA Y TELECOMUNICACIONES

INGENIERO EN ELECTRICIDAD

ESPECIALIZACIÓN ELECTRÓNICA Y AUTOMATIZACIÓN INDUSTRIAL

Presentado por:

Christian Andrés Carrión Carangui

Stalin Armando Torres Goyes

GUAYAQUIL – ECUADOR

AÑO 2011

AGRADECIMIENTO

A Dios.

A los amigos.

A todas las personas que contribuyeron en el desarrollo de este trabajo.

A todos quienes apuestan por el desarrollo tecnológico en Ecuador.

Al Ing. Carlos Valdivieso, por su

Incondicional ayuda y consejos para

lograr terminar con éxito nuestro proyecto.

DEDICATORIA

A Dios por ser nuestro padre y por guiarnos en la vida, a mi mami Julia Carangui, mi hermana, tíos, tías, primas, primos, mi sobrino Matías que es una nueva parte de mi vida y en especial a mi padre Manuel de Jesús Carrión Gonzales **(+)**, que con su ayuda y demostración de afecto han contribuido a formar una excelente persona y cada día llenan de felicidad mi corazón.

Christian Andrés Carrión Carangui

A Dios que siempre me ha acompañado, siendo su amor la fuente de energía para poder seguir adelante en cada paso de mi vida.

A mis padres, por su comprensión y ayuda incondicional, quienes siempre me inculcaron perseverancia con valores éticos, especialmente por sus sabios consejos y por estar a mi lado en los momentos difíciles, a mi hermana que siempre ha apoyado mis decisiones y con quien siempre puedo contar, y mi mis tías, tíos, primos que siempre han estado pendiente de mi, apoyándome en cada uno de mis retos.

Stalin Armando Torres Goyes

TRIBUNAL DE SUSTENTACION

Ing. Carlos Valdivieso A.

Profesor del Seminario de Graduación

Ing. Hugo Villavicencio V.

Delegado del Decano

DECLARACIÓN EXPRESA

"La responsabilidad del contenido de este trabajo, nos corresponde exclusivamente; y el patrimonio intelectual del mismo a la ESCUELA SUPERIOR POLITÉCNICA DEL LITORAL".

(Reglamento de exámenes y títulos profesionales de la ESPOL)

_____

Christian Andrés Carrión Carangui

_____

Stalin Armando Torres Goyes

RESUMEN

El objetivo principal es diseñar un programa de buena presentación y fácil de usar para el robot Pololu 3pi, además que nos permita mostrar los datos obtenidos por los sensores externos a conectarse con nuestro proyecto, utilizando una pantalla LCD y herramientas de software como AVR Studio y Proteus.

El proyecto que a continuación se presenta consiste en el diseño y manejo del robot Pololu 3pi con pantalla LCD, y está basado en el principio de mostrar por pantalla datos adquiridos por los diversos sensores, para luego su posterior análisis de cómo van variando las muestras con el tiempo.

Reconocer las diferentes ventajas del ATmega328p de nuestro Pololu 3pi que mediante un buena programación podremos evitar todo tipo de obstáculos mediante la medición que detecte nuestro sensor externo del cual usaremos una entrada analógica para verificar el obstáculo mas cercana y poder esquivarlo.

Se utiliza la pantalla LCD táctil para mostrar los datos, el cual es controlado por el ATmega 328p.

ÍNDICE GENERAL

AGRADECIMIENTO II

DEDICATORIA III

RESUMEN VII

INTRODUCCIÓN XII

CAPÍTULO 1 1

1. DESCRIPCIÓN GENERAL DEL PROYECTO 1

1.1. Antecedentes 1

1.2. Descripción del Proyecto 2

1.3. Aplicaciones 3

1.3.1. Sistemas de Deteccion de obtaculos 3

1.4. Descripción del problema 4

1.5. Efectos 4

1.6. Proyectos Similares 5

1.6.1. Pololu 3pi + eyes 5

1.6.2. Sensor de distancia por ultrasonido PING 5

1.6.3. Robot detector de obtaculos 6

CAPÍTULO 2 7

2. FUNDAMENTACIÓN TEÓRICA 7

2.1. Robot Pololu 3pi 8

2.1.1. Partes principales del robot pololu 3pi 9

2.1.2. Micromotores de metal 10

2.1.3. ATmega328 12
2.2. Sonar MaxSonar EZ1. 14
2.3. Herramientas de Software. 16
2.3.1. AVR Studio 4 17
2.3.2. Proteus 7.5 17
2.4. Programador AVR USB 18
CAPÍTULO 3 20
3. DESCRIPCIÓN E IMPLEMENTACIÓN DEL PROYECTO 20
3.1. Prueba Inicial 20
3.1.1. Construccion 21
3.2. Descripción del Proyecto Final 22
3.2.1. Diagrama de bloques del proyecto 23
3.3. Diagrama de Flujo 26
3.4. Programación en AVR STUDIO 28
CAPÍTULO 4 37
4. SIMULACION Y PRUEBAS 37
4.1. Simulacion en Proteus 37
4.2. Simulacion del movimiento de los motores 39
4.3. Pruebas funcionales del robot Pololu 41
CONCLUSIONES……
RECOMENDACIONES……
ANEXOS……
BIBLIOGRAFIA……

ÍNDICE DE FIGURAS

Fig. 1.1 Reconocimiento por emision recepcion..... 1

Fig. 1.2 Aplicación en la deteccion de obstaculos 3

Fig. 1.3 Pololu 3pi + eyes 5

Fig. 1.4 Sensor de distancia 5

Fig. 2.1 Robot Pololu 3pi..... 8

Fig. 2.1.1 Partes del polulo 3pi 9

Fig. 2.1.2 Partes del polulo 3pi (posterior) 10

Fig. 2.1.3 Motor DC de escobillas 10

Fig. 2.1.4 Puentes H para alternar poloridad..... 11

Fig. 2.1.5 Configuracion de pines del ATmega328p 13

Fig. 2.2 MaxSonar EZ1..... 14

Fig. 2.2.1 Dimensiones del MaxSonar EZ1 15

Fig. 2.3 Programador AVR USB 18

Fig. 3.1 Imagen inicial del pololu 21

Fig. 3.2 Conexión de pines en el Maxsonar EZ1 22

Fig. 3.3 Diagrama de bloques del proyecto..... 24

Fig. 4.1 Secuencia de inicializacion del proyeto..... 38

Fig. 4.1.1 Visualizacion del dato capturado por el sensor 39

Fig. 4.2.1 Motores ejecutando el desplazamiento hacia adelante 40

Fig. 4.2.2 Motores ejecutando el giro hacia la derecha..... 40

Fig. 4.3.1 Pololu 3pi visualizacion de voltaje 41

Fig. 4.3.2 Pololu 3pi esquivando obstaculos..... 42

Fig. 4.3.3 Pololu 3pi en un camino cerrado 43

Fig. 4.3.4 Pololu 3pi saliendo del camino cerrado..... 43

ÍNDICE DE TABLAS

Tabla 2.1 Parametros del Motor dc ..... **¡Error! Marcador no definido.**1

Tabla 2.2 Funcionamiento del Motor dc 12

INTRODUCCIÓN

El presente proyecto tiene como finalidad el diseño y construcción del 3pi de Pololu, el cual es un pequeño robot autónomo de alto rendimiento, designado para competiciones de seguimiento de línea, resolución de laberintos y detección de obtaculos. Alimentado por 4 pilas AAA (no incluidas) y un único sistema de tracción para los motores que trabaja a 9.25V, el 3pi es capaz de velocidades por encima de los 100cm/s mientras realiza vueltas precisas y cambios de sentido que no varían con el voltaje de las baterías. Los resultados son consistentes y están bien sintonizados con el código aún con baterías bajas. El robot está totalmente ensamblado con dos micromotores de metal para las ruedas, cinco sensores de reflexión, una pantalla LCD de 8x2 caracteres, un buzzer, tres pulsadores y más, todo ello conectado a un microcontrolador programable. El 3pi mide aproximadamente 9,5 cm (3,7”) de diámetro y pesa alrededor de 83 gr. (2,9 oz.) sin baterías.

El 3pi contiene un microcontrolador Atmel ATmega328 a 20 MHz con 32KB de memoria flash y 2KB de RAM y 1KB de EEPROM. El uso del ATmega328 lo hace compatible con la plataforma de desarrollo Arduino. Las herramientas gratuitas de desarrollo en C y C++ así como un extenso paquete de librerías que pueden trabajar con el hardware que lleva integrado están disponibles. Debemos tener en cuenta que es necesario un PROGRAMADOR AVR ISP externo como el USB AVR Programmer para programar el robot 3pi.

En el primer capítulo, se da un detalle general del proyecto con su debida descripción, antecedentes, planteo del problema, aplicación en el campo industrial, causales que motivaron a la elaboración de dicho proyecto y efectos al realizarlo.

En el segundo capítulo, se describen las herramientas de hardware utilizadas, que incluyen equipos y materiales adicionales en la construcción de la interfaz gráfica. Además se detalla el software utilizado AVR Studio, que es la principal herramienta de programación del microcontrolador Atmel ATmega328, por lo que se da una breve descripción de las funciones utilizadas para desarrollar este proyecto. Además hemos utilizado Proteus versión 7.5, un potente software que nos permite la simulación del proyecto y así ir modificando sentencias hasta llegar a nuestro objetivo.

En el tercer capítulo, se describe el prototipo inicial, del cual surgió la idea del proyecto final, él cual es una versión mejorada de una tarea realizada durante las clases del seminario el cual consiste de un sensor de obstáculos con muestreo en una pantalla LCD del 3pi Pololu.

En el cuarto y último capítulo encontramos todos los diagramas electrónicos para el diseño del circuito controlador del Pololu. Todos los valores de las pruebas efectuadas para la configuración correspondiente.

CAPÍTULO 1

1. DESCRIPCIÓN GENERAL DEL PROYECTO

1.1. Antecedentes

Es importante definir el concepto del funcionamiento básico de los ultrasonidos como medidores de distancia se muestra de una manera clara en el siguiente esquema (Figura), donde se tiene un receptor que emite un pulso de ultrasonido que rebota sobre un determinado objeto y la reflexión de ese pulso es detectada por un receptor:

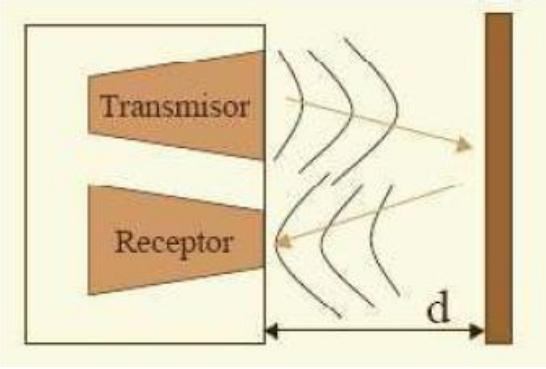


Fig. 1.1 Reconocimiento por emisión recepción

La mayoría los sensores de ultrasonido se basan en la emisión de un pulso de ultrasonido cuyo lóbulo. Campo de acción, es de forma cónica. Midiendo el tiempo que transcurre entre la emisión del sonido y la percepción del eco se puede establecer la distancia a la que se encuentra el obstáculo que ha producido la reflexión de la onda sonora.

Con el desarrollo tecnológico cada día podemos mejorar la funcionalidad de sensores de distancia que nos permitirán de una manera efectiva detectar cualquier obstáculo que se presente en el recorrido de nuestro robot pololu, para reprogramarlo de acuerdo a nuestros requerimientos en especial para de esta manera facilitar el manejo de sensores de distancia.

1.2. Descripción del proyecto

El presente capítulo explica el problema que se pretende solucionar a través del proyecto. Para este fin, es fundamental conocer de los diversos tipos de programas que el 3pi de Pololu nos presenta para el análisis de sensores externos, para nuestro proyecto nos enfocaremos en nuestro Sonar EZ1 el cual se convertirá en nuestra señal externa.

Además se trata sobre la nueva tecnología en el mercado de seguidores de línea y detectores de obstáculos que permitirá la capacidad de los usuarios por mejorar las habilidades del robot 3pi Pololu. Así mismo se detallan las justificaciones por las cuales se decidió emprender el proyecto y el posible impacto que éste vaya a tener en el medio y demás efectos.

1.3. Aplicación

La aplicación del proyecto puede ser en diversos campos como:

1.3.1. Sistemas de Detección de Obstáculos

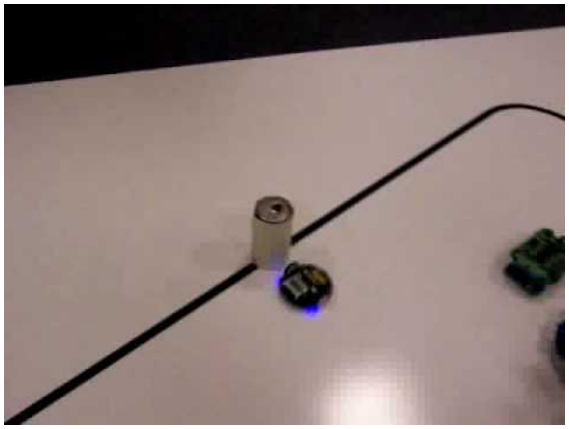


Fig. 1.2 Aplicación en la detección de obstáculos

Éste es un robot con un fuerte potencial. Fácil de utilizar ya que podemos probarlo desde los primeros minutos que lo tengamos en las manos y además dispone de muchísima documentación, por lo que es altamente flexible si queremos reprogramarlo para la aplicación que nosotros dispongamos.

Viene precargado con un programa de ejemplo con el cual podemos hacer girar los motores con la ayuda de los botones en placa, reproducir una melodía, ver gráficamente en la pantalla LCD el estado de los sensores.

Para lo cual utilizaremos nuestro robot como un detector de obstáculos capaz de detectar objetos desde 0 hasta 254 pulgadas (0 a 6.45 metros) y proporcionar una información de salida de la distancia medida en el rango de

6 a 254 pulgadas con una resolución de 1”. Los objetos u obstáculos presentes a una distancia inferior a 6” proporcionan una lectura mínima de 6”.

1.4. Descripción del problema

El problema planteado es buscar la manera de detectar obstáculos mediante el sonar ez1 un sensor externo del Pololu que mediante configuración y programación evitara los obstáculos que se encuentren en su camino.

Actualmente existen en el mercado diferentes tipos de robot detectores de obstáculos que podrían compararse con nuestro proyecto, pero la diferencia es que nosotros tenemos la capacidad de evitar los obstáculos de una manera inmediata y una fuente de energía de 4 pilas recargables AAA, teniendo como diferencia la adquisición de energía para el funcionamiento del dispositivo.

1.5. Efectos

El tiempo en la actualidad representa grandes cantidades de dinero. Los sistemas de pantalla táctil son fáciles de usar que los empleados pueden realizar su trabajo más rápido, dando como resultado una mejor calidad de servicio al cliente y con esto logramos brindar nuestro producto a mayor cantidad de personas evitando las largas colas de espera. Con la ayuda de la tecnología y de nuestro proyecto optimizamos el tiempo, haciendo más eficaz las diversas aplicaciones de atención al cliente.

1.6. Proyectos similares

1.6.1. Pololu 3pi + eyes

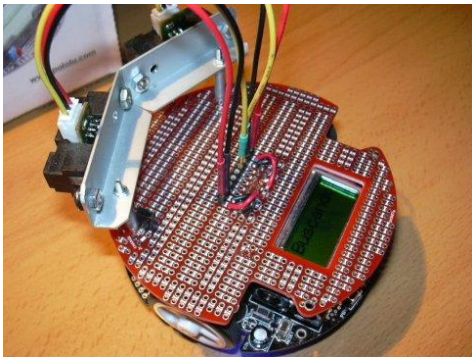


Fig. 1.3 Pololu 3pi + eyes

Este proyecto tiene como objetivo detectar obstáculos por medio de sensores de distancia Sharp 2Y0A21 en el cual el robot Pololu 3pi evita todo tipo de obstáculos de una manera precisa.

Con un alcance de detección de 4 "a 32" y un voltaje analógico que indica la distancia, este sensor es muy fácil de usar.

1.6.2. Sensor de distancia por ultrasonido

Fig. 1.4. Sensor de distancia

Éste sensor PING de Parallax es ya un clásico y no puede faltar en ningún proyecto de robótica. Funciona como un sonar mediante ultrasonidos y es

capaz de detectar objetos a una distancia de entre 2 centímetro a 3 metros. Dispone de un indicador LED y tan sólo requiere de un pin para su funcionamiento. Se puede utilizar en una placa de prototipo o directamente en tu robot.

El sensor envía ultrasonidos por un lado y mide el tiempo de rebote del sonido. En su pin de salida podremos medir el ancho de pulso PWM en función de la distancia del obstáculo. Es muy sencillo hacerlo funcionar con un Arduino, PIC o cualquier otro microcontrolador.

1.6.3. Robot detector de obstáculos

Este Robot es capaz de avanzar por su cuenta. Fue hecho con la aplicación del estudio del caos, sistemas complejos y redes neuronales. La naturaleza de este coche es de un circuito eléctrico muy simple. Sólo use PIC, tres sensores medidores de distancia y dos módulos FET.

CAPÍTULO 2

2. FUNDAMENTACIÓN TEÓRICA

En el presente capítulo se muestran los fundamentos básicos del robot Pololu 3pi y sus componentes más importantes. Además se explica sobre el sonar MaxSonar EZ1 para la detección de obstáculos, también se expone las técnicas, de interfaz con el robot Pololu 3pi, también se menciona los campos de aplicación del proyecto, por último se describen los programas desarrollados para la comunicación y hardware implementados que se utilizan para el robot pololu 3pi.

2.1. Robot Pololu 3pi

Fig. 2.1 Robot Pololu 3pi

El Robot Pololu 3pi es un robot autónomo de alto rendimiento móvil, el cual esta designado para competiciones de seguimiento de línea y resolución de laberintos. El pololu 3pi es alimentado por 4 pilas AAA para su funcionamiento y un único sistema de tracción para los motores los cuales trabajan a 9.25V, el Pololu 3pi es capaz de tener velocidades por encima de los 100cm/s mientras realiza vueltas precisas y cambios de sentido que no varían con el voltaje de las baterías. Los resultados son consistentes y están bien sintonizados con el código aún cuando las baterías están bajas.

2.1.1. Partes principales del robot pololu 3pi

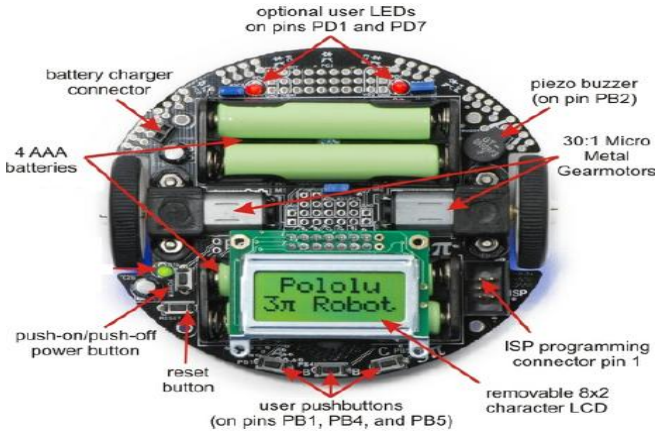


Fig. 2.1.1 Partes del pololu 3pi

El robot Pololu 3pi está totalmente ensamblado con dos micromotores de metal para las ruedas, cinco sensores de reflexión, una pantalla LCD de 8x2 caracteres, un buzzer, tres pulsadores y más, todo ello conectado a un microcontrolador programable. El 3pi mide aproximadamente 9,5cm (3,7") de diámetro y pesa alrededor de 83 gr. (2,9 oz.).

El 3pi contiene un microcontrolador Atmel ATmega328 a 20 MHz con 32KB de memoria flash y 2KB de RAM, y 1KB de EEPROM. El uso del ATmega328 lo hace compatible con la plataforma de desarrollo Arduino. Lo hace compatible con las herramientas de desarrollo en C y C++ así como un extenso paquete de librerías que pueden trabajar con el hardware que lleva integrado y que están disponibles para el seguimiento de línea y laberintos.

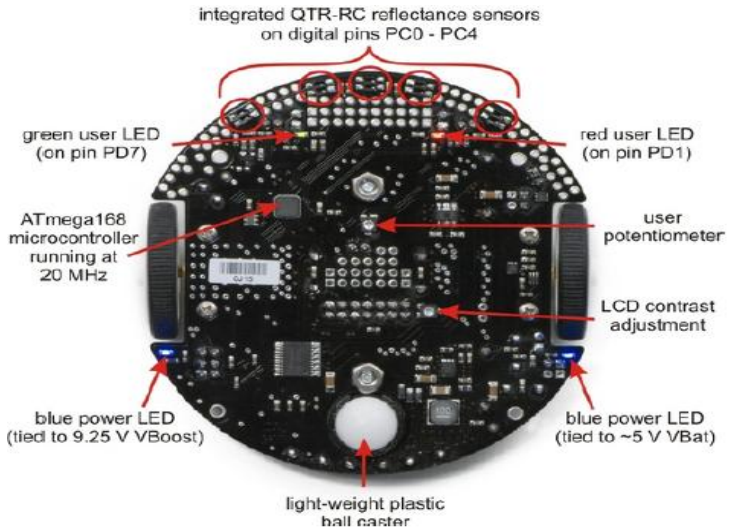


Fig. 2.1.2 Partes del pololu 3pi (Posterior)

2.1.2. Micromotores de metal

Motor DC de escobillas (reductores)

Fig. 2.1.3 Motor DC de escobillas

El motor que tiene el Pololu en cada una de sus ruedas es el motor dc de escobillas (motor reductor) el cual es usado mucho en robótica.

La velocidad libre de rodamiento de este motor DC es de varios miles de revoluciones por minuto muy alta para el desplazamiento del robot, por lo que

un dispositivo de engranajes permite reducir estas revoluciones y aumentar el par, la fuerza de rodamiento. El ratio de engranaje es de 30:1 en el pololu, es decir 30 vueltas de motor una vuelta de rueda. Estos parámetros están representados en la tabla siguiente.

Engranage:	30:1
Velocidad libre:	700 rpm
Consumo mín:	60 mA
Par máximo:	6 oz·in
Consumo máx:	540 mA

Tabla. 2.1 Parámetros del motor dc

Ya que nuestro robot Pololu 3pi se mueve en todas las direcciones los motor dc deben cambiar su dirección de rotación para ello debe alternar la polaridad del voltaje aplicado. Para eso se usan los llamados puentes H como se muestra en el diagrama:

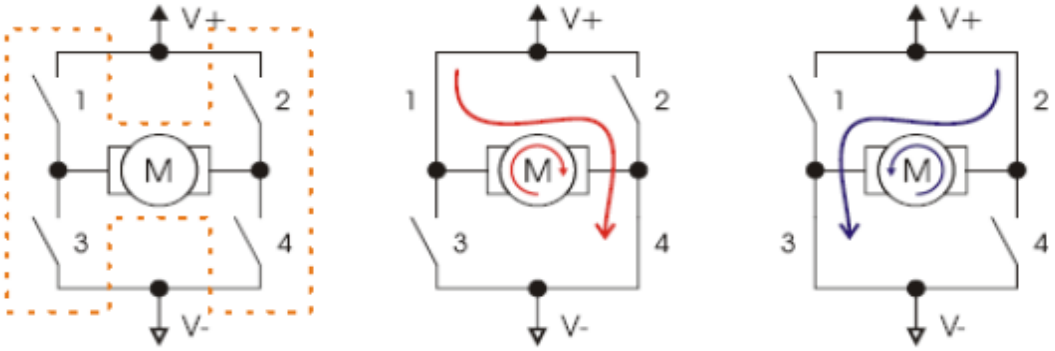


Fig. 2.1.4 Puentes H para alternar la polaridad

Los pins PD5 y PD6 para el motor M1 y para el motor M2 se utilizan los pins de control en PD3 y PB3. Podemos ver su funcionamiento en la siguiente tabla:

PD5	PD6	1	2	3	4	M1		PD3	PB3	1	2	3	4	M2
0	0	off	off	off	off	off (coast)		0	0	off	off	off	off	off (coast)
0	1	off	on	on	off	forward		0	1	off	on	on	off	forward
1	0	on	off	off	on	reverse		1	0	on	off	off	on	reverse
1	1	off	off	on	on	off (brake)		1	1	off	off	on	on	off (brake)

Tabla. 2.2 Funcionamiento del Motor DC

2.1.3 ATmega328P

Es un microcontrolador CMOS de alto rendimiento de 8 bits basado en AVR arquitectura RISC mejorada capaz de ejecutar instrucciones de gran alcance en un solo ciclo de reloj, fácil de programar. Posee internamente un circuito de Power-On Reset que eliminan la necesidad de componentes externos y expanden a 32 el número de pines que pueden ser utilizados como líneas I/O (entrada/salida; Input/ Output) de propósito general de trabajo. Proporciona una memoria de datos EEPROM de (1K), una memoria de programa FLASH (2K), una memoria de datos RAM de propósito general de 2K, tres flexibles contador de tiempo / contadores que comparar los modos, las interrupciones internas y externas, serial USART programable, un byte orientada a interfaz

de dos hilos de serie, puerto serial SPI, una de 6 canales de 10 bits convertidor A / D (8 canales en TQFP y QFN / FML paquetes), temporizador de vigilancia programable con el oscilador interno, y cinco modos de software de ahorro de energía seleccionable.

El dispositivo funciona entre 2.7 a 5.5 voltios.

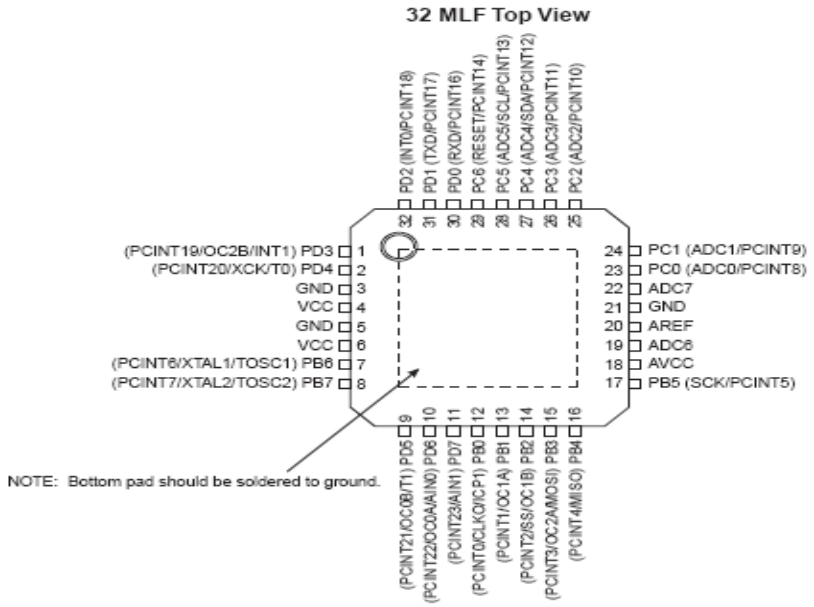


Fig. 2.1.5 Configuración de pines del ATmega328P

2.2. Sonar MaxSonar EZ1

Fig. 2.2 MaxSonar EZ1

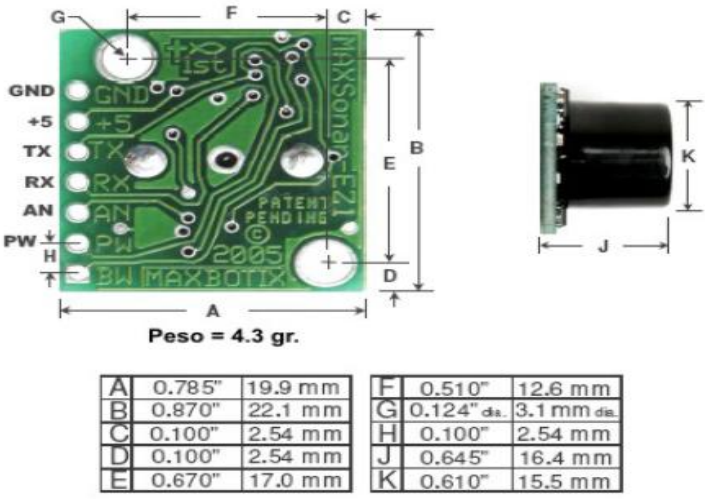
Este pequeño sensor por ultrasonido ofrece capacidades de detección de presencia y medición de distancia en rango corto y largo y un consumo muy bajo. El MaxSonar- EZ1 el cual tiene integrado el transmisor y el receptor en la misma placa. Este sensor es capaz de detectar objetos situados entre 0 y 254 pulgadas (0 y 6,45 metros) de distancia, proporcionando los datos obtenidos del cálculo de la distancia con una resolución de 1 pulgada (2,54 cm). Entre los formatos de salida se incluyen la salida de ancho de pulso, salida de tensión analógica y salida digital serie. Gracias a estos tres formatos de salida el sonar MaxSonar-EZ1 se puede conectarse con cualquier sistema basado en microcontrolador de una forma fácil y flexible.

Algunas Características del MaxSonar Ez1.

- Ganancia variable continúa para el control del haz ultrasónico y supresión de la dispersión.

- Permite detectar objetos a una distancia inferior a 6’’.
- Alimentación a 5V con consumo de 2mA.
  Se pueden realizar hasta 20 medidas por segundo.
- Las medidas y salida de información se pueden realizar de forma continua.
- Se puede emplear una señal externa para iniciar/detectar cada nuevo ciclo.
- Formato de salida con protocolo serie de 0 a 5 V con 9600 baudios, 8 bits,
  sin paridad y 1 bit de Stop.
- Formato de salida mediante tensión analógica 10mV / pulgada
- Formato de salida mediante anchura de impulso 147μs / pulgada
- Todos los formatos de salida de información están activos simultáneamente y se pueden emplear cualquiera de ellos en cualquier momento
- Diseñado para trabajo en interiores
- Los transductores ultrasónicos trabajan a 42 KHz.

Sus dimensiones comparados con otros dispositivos de su categoría es mucho menor lo cual, facilita la manipulación y trabajo con el sonar MaxSonar EZ1 haciéndolo fácil implementar en cualquier dispositivo, su circuito impreso consta con orificios para el montaje. Como se ve en la figura:



A	0.785"	19.9 mm	F	0.510"	12.6 mm
B	0.870"	22.1 mm	G	0.124" dia.	3.1 mm dia.
C	0.100"	2.54 mm	H	0.100"	2.54 mm
D	0.100"	2.54 mm	J	0.645"	16.4 mm
E	0.670"	17.0 mm	K	0.610"	15.5 mm

Fig. 2.2.1 Dimensiones del MaxSonar EZ1

La lectura de la distancia a los objetos detectados se realizan a través de los siguientes pines del sensor ultrasónico, que entregan los datos en diferentes formatos:

TX: Transmite vía serie el resultado de la distancia medida, se realiza en formato RS232 excepto que los voltajes de salida son de 0-5V. Se transmiten 5 bytes por cada medida realizada, empieza por el carácter ASCII "R", continua con tres caracteres ASCII con los dígitos de la medida propiamente dicha y comprendido entre 006 y 254 y finaliza con el código de retorno de carro (0x0D). La velocidad es de 9600 baudios, con 8 bits de datos, sin paridad y un bit de stop.

RX: Este pin está permanentemente a nivel "1" mediante una resistencia "pull-Up" interna. En estas condiciones el sonar esta realizando medidas de forma continua y transmitiendo la distancia. Sin embargo esta señal se puede

emplear para controlar externamente el inicio de una nueva medida. Efectivamente, cuando se pone a “0” el sistema está detenido. Poniéndola a nivel “1” o simplemente sin conectar, se inicia una nueva medida.

AN: Salida analógica de tensión comprendida entre 0 y 2.55 V que representa el valor de la distancia medida. El factor empleado es de 10Mv/pulgada.

PW: Este pin proporciona un pulso de salida cuya duración determina la distancia medida. La distancia se puede calcular usando el factor de 147Ms/pulgadas.

2.3. Herramientas de Software

El proyecto utilizó dos tipos de software, el primero tiene como fin la programación del Pololu 3pi (ATmega328P) y el segundo tiene como objetivo la simulación del proyecto para verificar que no hay errores.

2.3.1. AVR Studio

Es una herramienta de desarrollo que permite realizar proyectos para microcontroladores. Proporciona una solución fácil para aplicaciones de sistemas embebidos, sin comprometer el rendimiento o el control, además desarrolla rápidamente y despliega aplicaciones complejas.

Originalmente concebido como una herramienta fácil de usar, AVR Studio También se integra con el compilador GCC plug-in, añade soporte para todas

las herramientas de Atmel que apoyan la arquitectura AVR 8-bit, incluyendo la STK500 AVR. La característica principal del **AVR STUDIO** es su fácil manejo, permite visualizar rápidamente que ocurre con los registros, como también así se pueden modificar.

AVR Studio 4 incluye un depurador que permite el control de ejecución con fuente y nivel de instrucción, paso a paso y puntos de interrupción, el registro, la memoria y E / S puntos de vista. Apoyo a la programación completa para los programadores independientes permitiendo crear y modificar librerías existentes para un mejor rendimiento del código.

2.3.2. Proteus 7.7

Es un paquete de software para el diseño de circuitos electrónicos que incluye captura de los esquemas, simulación analógica y digital combinada, además posee una herramienta ARES que se utiliza para el diseño de circuitos impresos. Proteus es un entorno integrado diseñado para la realización completa de proyectos de construcción de equipos electrónicos en todas sus etapas: diseño, simulación, depuración y construcción. El paquete está compuesto por dos programas: ISIS, para la captura y simulación de circuitos; y ARES, para el diseño de PCB’s. También permite simular y depurar el funcionamiento de todo el sistema ejecutando el software paso a paso, insertando puntos de ruptura (breakpoints, que también pueden ser generados por el hardware), mirando el contenido de registros y posiciones de memoria,

etc. y comprobando si la respuesta del hardware es la correcta. También se simulan herramientas electrónicas, como osciloscopios, analizadores lógicos, voltímetros, etc.

2.4. Programador AVR USB

Fig. 2.3 Programador AVR USB

Probablemente uno de los programadores AVR mas pequeños del mundo. El programador AVR USB es un muy compacto, de bajo costo con sistema (ISP) para microcontroladores AVR de Atmel, lo que hace de este dispositivo una solución de programación atractiva para los controladores AVR, compatible con de robots Orangután y Pololu 3Pi.

El programador AVR USB se conecta al puerto USB de su ordenador a través de un cable USB incluido y se comunica con el software de programación, tal como AVR Studio, a través de un puerto COM virtual utilizando el protocolo AVRISPV2/STK500. El programador se conecta al dispositivo de destino mediante un cable ISP de 6 pines.

Otra peculiaridad que hace de este programador tan especial es su función USB a serie, El programador se instala como dos puertos COM virtuales: uno para comunicarse con el software de programación y uno para el adaptador de uso general de USB a serie. Esto significa que usted puede cambiar sin problemas entre la programación de un microcontrolador y la depuración a través del puerto TTL serie sin tener que abrir y cerrar el programa de terminal

CAPÍTULO 3

3. DESCRIPCIÓN E IMPLEMENTACIÓN DEL PROYECTO

3.1. Prueba inicial

Para la realización del proyecto tomamos como consideración dividirlo por fases para así ir diseñando y simulando paso a paso las diferentes partes del mismo. La primera etapa consistía en ver el funcionamiento del pololu con su programa demo el cual ya viene en el robot pololu, que nos permite ver al pololu funcionando, La segunda etapa consistía en encargarnos de manejar el sonar Maxsonar EZ1, verificando los formatos de salida del sonar para poder escoger una de sus salidas y poder trabajar con ella. Las salidas que tiene el sonar son la salida de ancho de pulso, salida de tensión analógica y salida digital serie. La última etapa consistía en la implementación del sonar Maxsonar EZ1 en conjunto con el robot pololu 3pi, dando la señal del sonar al pololu para que el se pueda mover evitando los obstáculos.

3.1.1. Construcción

La construcción del circuito de nuestro proyecto consistió en el la conexión entre el sonar Maxsonar EZ1 y el robot pololu 3pi añadiendo 2 pines en el pololu que son los 5 voltios que necesita el sonar en este caso son vcc y tierra. En esta primera parte del proyecto solo se realizó la prueba del pololu con su demo conectando las baterías y siguiendo las instrucciones en que indicaba su pantalla lcd para verificar que el pololu esté en buen estado.

Fig. 3.1 Imagen inicial del pololu

La siguiente parte de la construcción fue el la conexión de los pines en el sonar Maxsonar EZ1, para el funcionamiento del mismo, ya que necesita 5 voltios de entrada para su funcionamiento y las salidas que el sonar puede entregar.

Fig. 3.2 Conexión de pines en el Maxsonar EZ1

3.2. Descripción del proyecto final

Luego de las conexiones del pololu 3pi como las del sonar y la programación del microcontrolador que tiene el pololu estamos listos para que funcione el prototipo, inicialmente notamos ciertos errores en el movimiento que realizaba el pololu, por motivo de que la primera prueba de comunicación con nuestro sonar Maxsonar EZ1 no la podíamos interpretar muy bien.

Después de varias pruebas realizamos el código final que consiste en el movimiento del pololu por todas partes evitando obstáculos.

El microcontrolador se encarga de hacer trabajar a los demás dispositivos como es la pantalla, la cual va mostrando que hace el pololu. En la pantalla se indica que se tiene que presionar B en el pololu para poder funcionar, avanzando a la siguiente imagen donde muestra la distancia que indica el sonar en ese momento del obstáculo, al presione nuevamente B el pololu

funciona. En la pantalla se mostrara lo que indica el sonar y las velocidades con la cual se esta moviendo el pololu tanto del motor derecho como del motor izquierdo, cuando el pololu se topa con un obstáculo el sonar envía la señal al pololu y el gira para evitar el obstáculo, la pantalla mostrara "buscando exit" ósea donde no exista obstáculo para poder continuar.

3.2.1. Diagrama de bloques

Para la elaboración del proyecto utilizamos el ATmega328p debido a que posee mayor cantidad de memoria, debido a que nuestro pololu puede realizar diferentes aplicaciones las cuales demanda una gran cantidad de líneas de código para ciertos proyectos, además que utilizamos una pantalla LCD y menús que consumen espacio en memoria.

Este proyecto se encarga de almacenar los datos obtenidos por el sensor y analizarlos, de manera que estos son enviados un cable de envía la señal de salida analógica del sonar a nuestro microcontrolador, para luego en una pantalla LCD presentar la posición del obstáculo detectado, además que hemos implementado una pequeña subrutina de inicio de nuestro robot Pololu la cual nos permitirá poder controlar el inicio de la búsqueda de obstáculos para evitarlos de una manera eficaz.

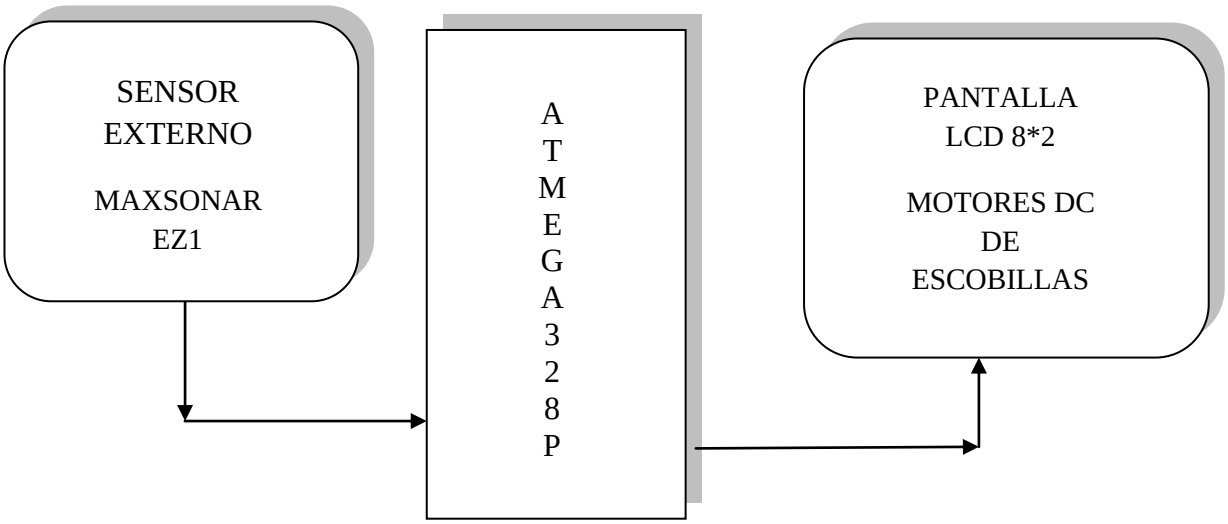
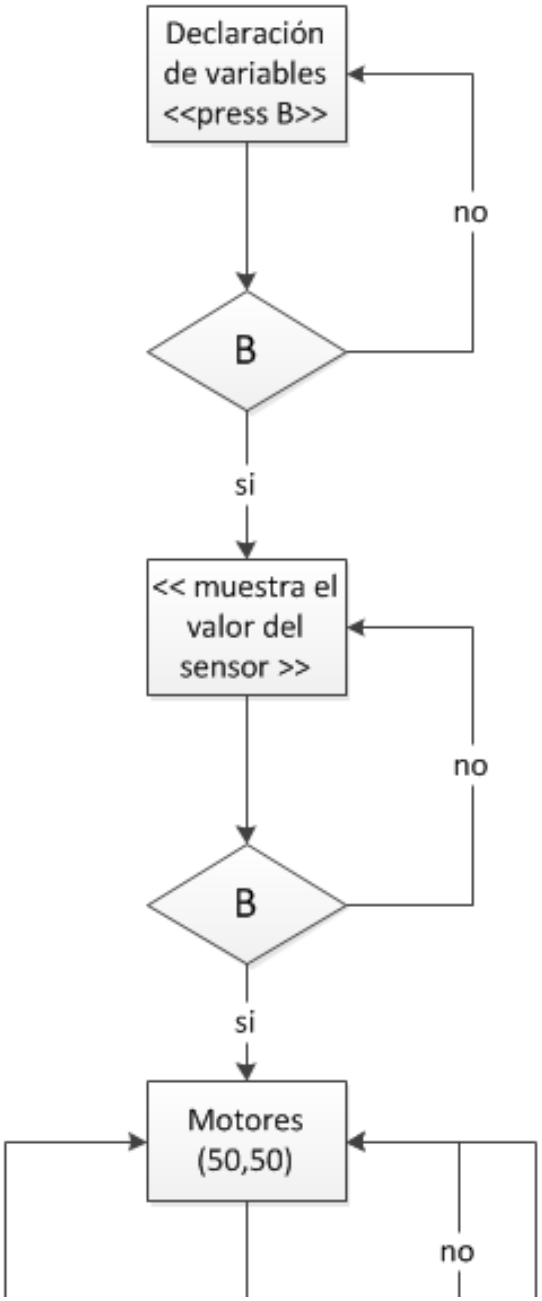
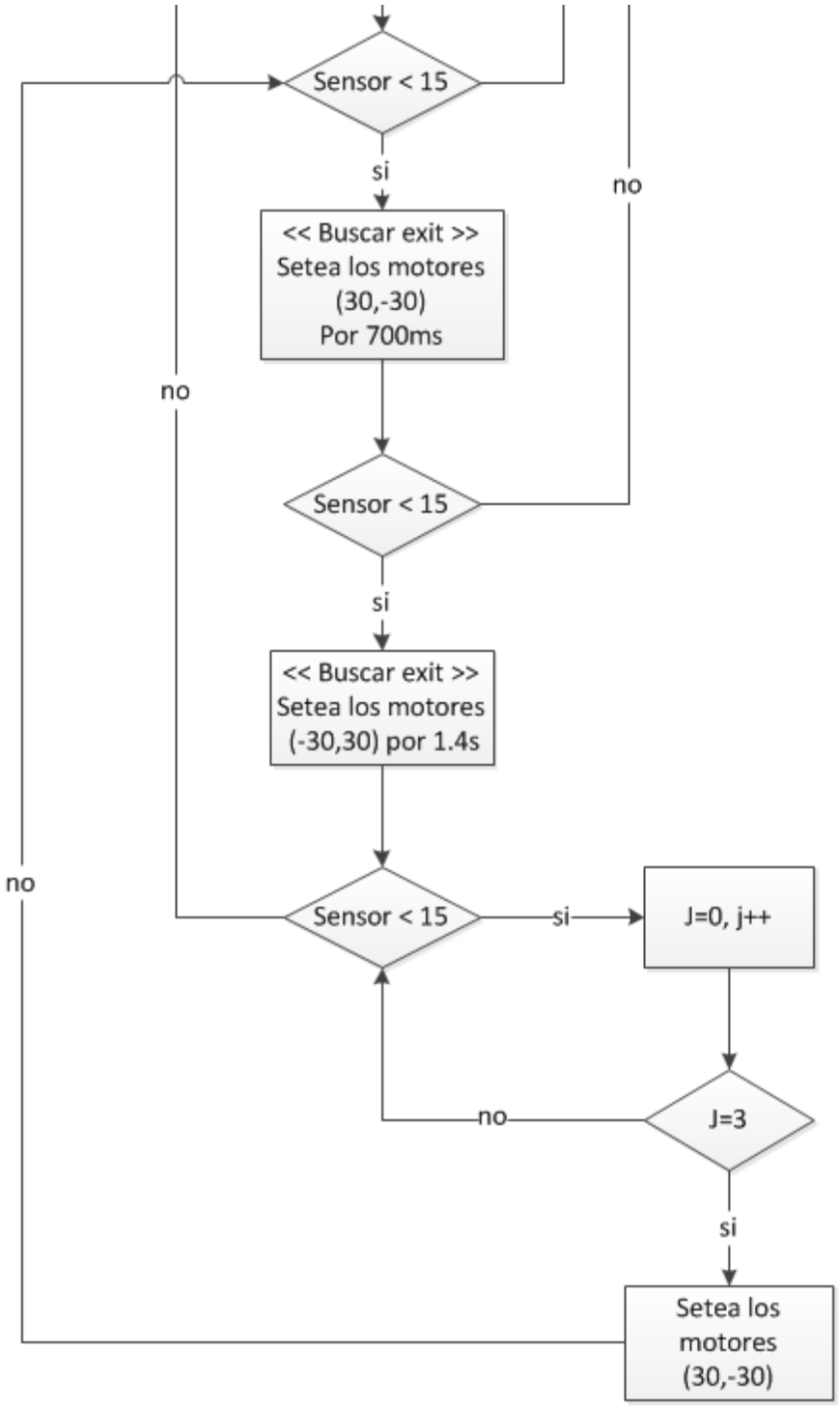


Fig. 3.3 Diagrama de bloques del proyecto

3.3 Diagrama de Flujo




Sensor < 15
si
no
<< Buscar exit >>
Setea los motores
(30,-30)
Por 700ms
no
Sensor < 15
si
<< Buscar exit >>
Setea los motores
(-30,30) por 1.4s
no
Sensor < 15
si
J=0, j++
no
J=3
si
Setea los
motores
(30,-30)

3.4 Programación en AVR STUDIO 4.4

A continuación se explica cada parte del programa realizado para el funcionamiento del microcontrolador.

```
/* Programa:rutina paseo.c        Torres y Carrion   02/2011
*  Descripción:
*  Programa para el 3pi con el sonar Maxsonar Ez1 conectado
*  en ADC7 que le permite esquivar obstáculos
*  Quitar jumper ADC7.
*/
```

En esta parte del código tenemos las librerías que usamos las cuales ayudan a simplificar nuestro programa.

```
//   Includes
#include <pololu/3pi.h>        // especifico 3pi
#include <avr/pgmspace.h>      // variables en memoria de programa
```

Aquí tenemos las contantes que usamos las cuales vamos llamando en nuestro código las cales son de mensajes y música que se ven y se escuchan mientras se ejecuta el programa.

```
// Mensajes de introducción guardados en memoria de programa
const char linea1[] PROGMEM = "Pololu";
const char linea2[] PROGMEM = "obstacle";
// Tonos musicales guardados en memoria de programa
const char hola[] PROGMEM = ">g32>>c32";
const char ir[] PROGMEM = "L16 cdegreg4";
```

Declaramos variables que vamos utilizando en todo el código las cuales las vamos llamando.

```
// Variables generales
unsigned int sens_izq;   // sensor left
int motor_der;         // motor right
int motor_izq;         // motor left
int i;
int j;
float x; // son negativos y positivos variables del motor
float y;
```

Activamos las resistencias de pull-up internas

```
void initialize(){
    // Set PC5 as an input with internal pull-up disabled
    DDRC  &= ~(1<< PORTC5);
    PORTC &= ~(1<< PORTC5);
   emitters_off();
```

Al encender nuestro Pololu muestra los mensajes de nuestras constantes declarados anteriormente y de música.

```
    // Toca music y muestra mensaje de hola
    print_from_program_space(linea1);
    lcd_goto_xy(0,1);
    print_from_program_space(linea2);
```

```
play_from_program_space(hola);
delay_ms(1000);
```

Muestra el voltaje el cual se encuentran las pilas que están puestas cuando el valor de las pilas son inferiores a 4600 muestra un mensaje indicando que es tiempo de cambiarlas.

```
//Muestra el voltaje de la batería y espera botón
 while(!button_is_pressed(BUTTON_B))
 {
    clear();
    print_long(read_battery_millivolts());  // usa ADC6 para batería
    print("mV");
    lcd_goto_xy(0,1);
  if (read_battery_millivolts()<4600){
     print (" !Ahhh¡");
     red_led(1);
  }
  else
       print("Pulsa B"); // indica que hay que presionar B
    delay_ms(100);
}
```

Aquí al presionar B se muestra en pantalla lo que se lee en el sensor.

```
    // Espera botón B para empezar a moverse
    wait_for_button_release(BUTTON_B);
    while(!button_is_pressed(BUTTON_B))
    {
     clear();
     lcd_goto_xy(0,0);
     print ("I ");
       print_long(analog_read(7));  // valor ADC7 sensor <
       lcd_goto_xy(0,1);
       delay_ms(200);
    }
    wait_for_button_release(BUTTON_B);
    clear();
```

Al presionar otra vez B toca música que al finalizar se mueva el Pololu si no hay obstáculo caso contrario ingresa en una función llamada buscando exit encendiendo los led del Pololu

```
    // Toca música y espera a que termine para empezar.
    play_from_program_space(ir);
    while(is_playing());
  }
  void lee_sensores(){
    //  si hay obstáculos
```

```
    if (!analog_is_converting()) sens_izq = analog_read(7);
  }
```

Para ingresar a esta función el sensor debe detectar un obstáculo a una distancia menor a 15cm

```
    void busca_exit(){
    clear();
    lcd_goto_xy(0,0);
     print("buscando");
    lcd_goto_xy(1,1);
     print("exit");
    red_led(1);            // Encienda leds
    green_led(1);
```

En estas condición nuestro robot setea las velocidades de su motores haciéndolo que gire para buscar una salida.

```
    if (sens_izq<15)
    {
        play ("c32");
        set_motors(x,y);
        y=x;
        x=-x; // gira or -30,30
        delay_ms(700);
        lee_sensores();
```

Entra en un lazo for cuando nuestro robot entra en un camino cerrado para buscar la salida

```
for(j=0;j<3;j++)
{
        if(sens_izq<15)
        {
           i=0;
                play ("c32");
      set_motors(x,y); // gira or -40,40
      delay_ms(1400);
      lee_sensores();

    }
        else
        {
           j=10;
        }
        y=x;
        x=-x;
}
if(sens_izq<15)
{
    play ("c32");
```

```
                y=x;
                x=-x;
            set_motors(x,y); // gira or -40,40
            delay_ms(700);
            //sens_izq=30;
        }
        play ("g32");
    }
    red_led(0);          // OK salida encontrada
    green_led(0);        // apaga leds
}

int main(){
    // inicializa 3pi
    initialize();
    // Bucle principal.
    x=30;
    y=-30;

    while(1){
      lee_sensores();
```

Tenemos la condición para que gire en otro sentido y pueda evitar el obstáculo

```
if (sens_izq<15){
   set_motors(0,0);      // stop motors
   y=x;
   x=-x;
   busca_exit();
}
```

En **esta condición se activa siempre y cuando no exista obstáculo en su recorrido seteando los motores a una velocidades constante de 50**

```
if (sens_izq > 15){
   // obstaculo izq gira a derecha ------> para q valla girando
   motor_izq=50; //-sens_izq/10;   // acelera
   motor_der=50; //-sens_izq/10;   // reduce

}

   set_motors(motor_izq, motor_der);   //izq
delay_ms(20);
```

M**uestra en pantalla los valores de las velocidades de los motores**

```
clear();          // Mostrar valores
```

```
            lcd_goto_xy(0,0);
            print_long(sens_izq);
            lcd_goto_xy(0,1);
            print_long(motor_izq);
            lcd_goto_xy(5,1);       // valor motores
            print_long(motor_der);
          }
      }
// end
```

CAPÍTULO 4

4. SIMULACIÓN Y PRUEBAS

4.1. Simulación en Proteus

A través del software de simulación Proteus y todas sus herramientas a utilizar pudimos realizar la simulación de nuestro proyecto.

En este caso la imagen a mostrar fue el atmega 328 con la lcd mostrando la inicialización del robot, ya que Proteus no contiene un sensor Ez1 entre sus elementos se implemento el sensor a través de un potenciómetro para poder simular una variación de la señal de entrada capturada y un pulsador para dar inicio al sistema previamente programado en el Pololu.

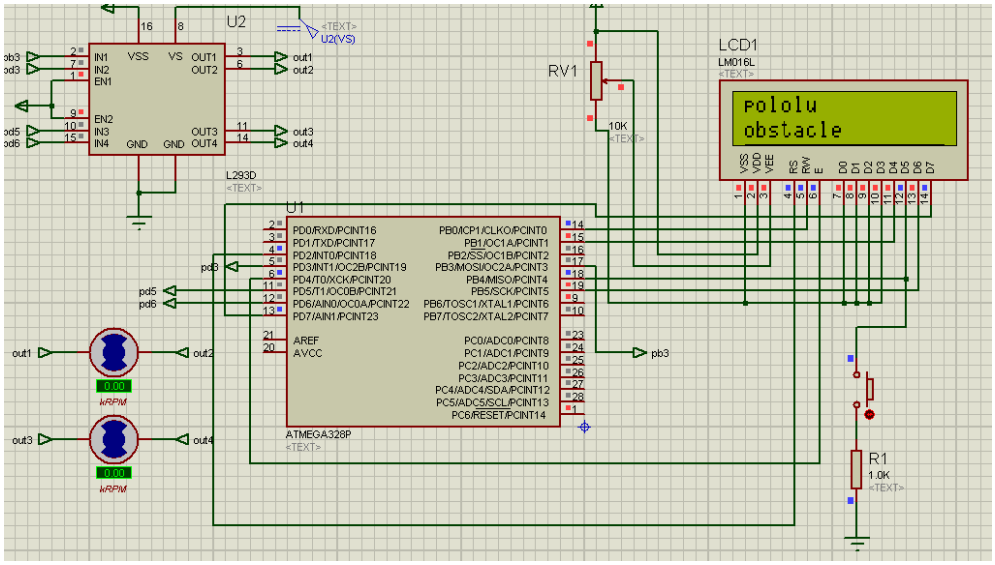


Fig. 4.1 Secuencia de inicialización del proyecto

Inicialmente realizamos la simulación de la interfaz gráfica para luego su posterior implementación. Como nuestro proyecto captura datos del sensor zona Ez1, es un poco complicado manejar con exactitud la manera como se comportara el sensor debido a que nosotros utilizaremos un potenciómetro como medio de señal de entrada para variar y simular nuestro sensor. Tomamos como referencia el proyecto realizado en las clases del seminario, dicho proyecto se encarga de un seguidor de línea que usaba otros tipos de sensores. Después de un riguroso análisis del código, pruebas y mediciones de distancia para evitar colisiones mediante nuestro sensor con esto llegamos a alcanzar las metas planteadas.

La siguiente figura corresponde al valor capturado por el sensor, el cual es mostrado en nuestra lcd como muestra de que se está recibiendo los datos por parte del Atmega 328p y enviando a la lcd de una manera correcta.

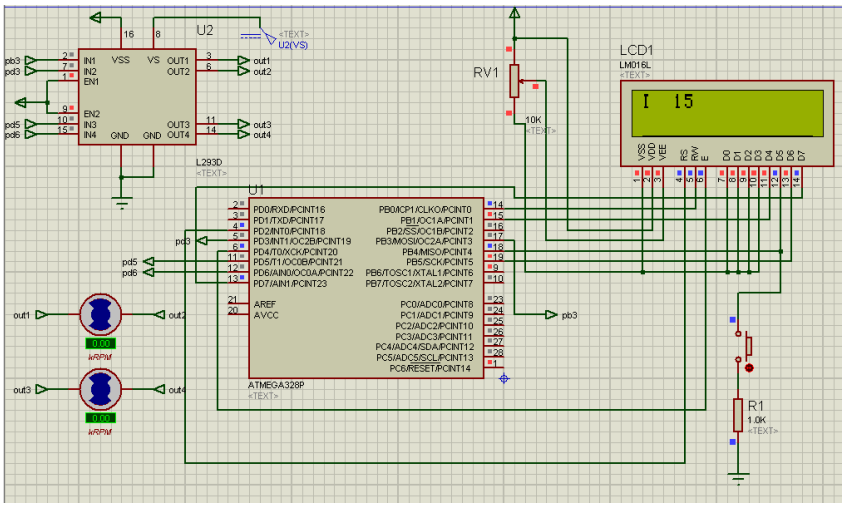


Fig. 4.1.1 Visualización del dato capturado por el sensor

4.2 Simulación del movimiento de los motores

Debido a que para la conexión entre el microcontrolador y los motores se necesita de un driver para controlarlos, en la simulación se usó un L293D que suministra la corriente necesaria para una simulación eficiente. La que permitirá obtener mejores resultados a la hora de realizar nuestras pruebas.

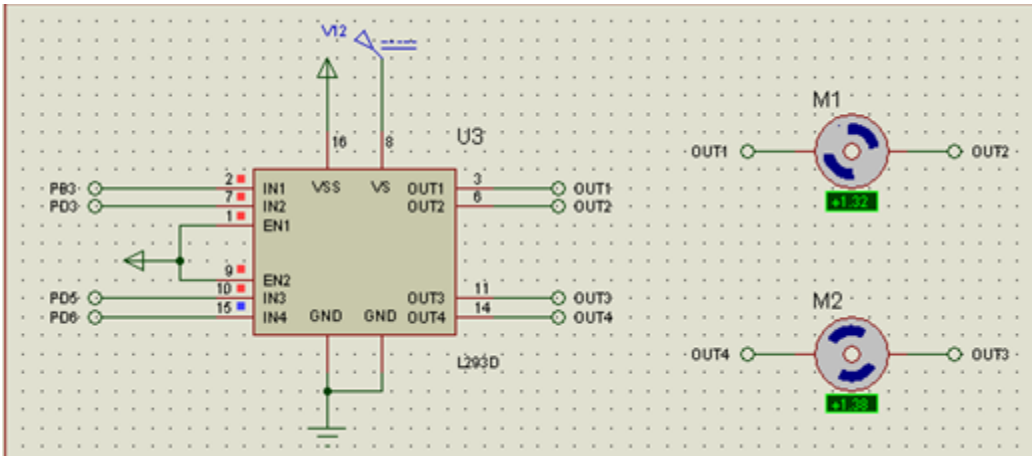


Fig. 4.2.1 Motores ejecutando el desplazamiento hacia adelante

Mediante nuestro programación definimos el momento en el cual hacemos girar a nuestro robot Pololu dependiendo de esto se realizara el giro correspondiente, el giro consiste en girar un motor en sentido horario y el otro motor en sentido anti horario.

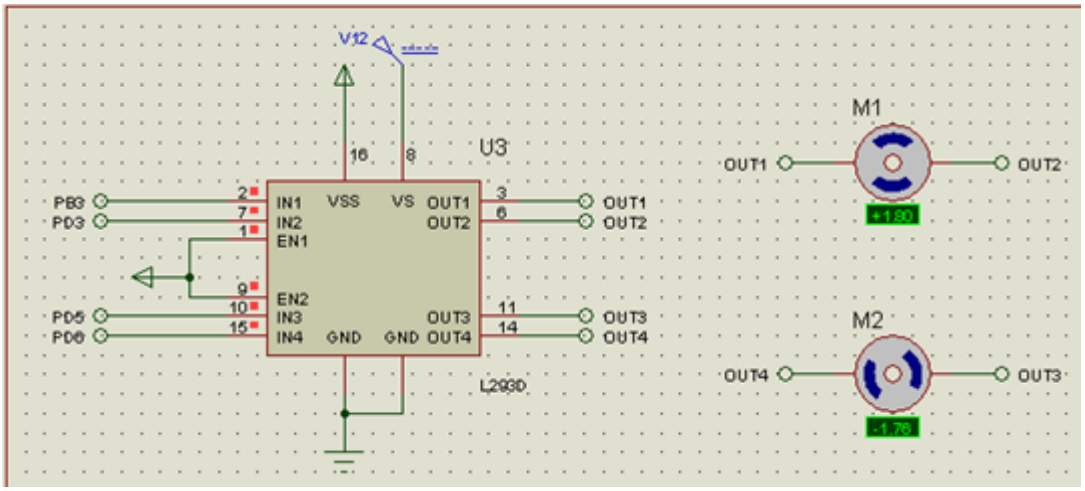


Fig. 4.2.2 Motores ejecutando el giro hacia la derecha

4.3 Pruebas funcionales del robot Pololu

Empezaremos por mostrar las diferentes cualidades de nuestro robot de tal manera que podamos visualizar de que manera hemos logrado realizar nuestros objetivos en la construcción de un robot que es capaz de evitar todo tipo de obstáculos a pesar de solo poseer un solo sensor.

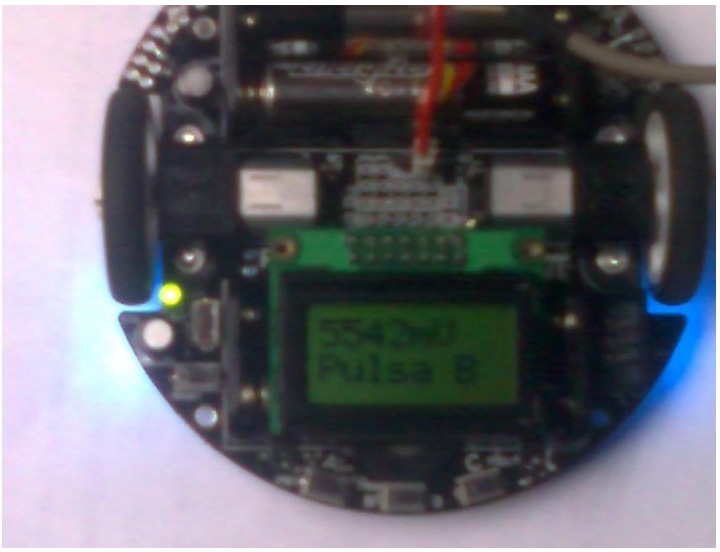


Fig. 4.3.1 Pololu 3pi visualización de voltaje

En esta imagen de nuestro Pololu podemos visualizar en la pantalla lcd el voltaje de las pilas que están energizando el Pololu e indicando que está a la espera de que su pulsador B este activado para poder continuar e iniciar las subrutinas siguientes.

Podemos observar en la siguiente imagen la capacidad de esquivar objetos de diferentes tamaños, a diferente distancia de nuestro Pololu logrado gracias a las subrutinas implementadas

Fig. 4.3.2 Pololu 3pi esquivando obstáculos

Para la siguiente imagen mostramos a nuestro robot en el interior de un camino cerrado en el cual nuestro robot deberá buscar la manera de salir de este camino sin chocar con las paredes de una manera precisa en la cual buscara la salida apropiada para realizar su escape de una forma rápida y segura.

Fig. 4.3.3 Pololu 3pi en un camino cerrado

Nuestro robot Pololu realiza con éxito su escape del camino cerrado sin ningún problema gracias a la ejecución de las subrutinas de una manera precisa

Fig. 4.3.4 Pololu 3pi saliendo del camino cerrado

CONCLUSIONES

1.- Mediante nuestro proyecto hemos logrado implementar un código amigable para poder evitar obstáculos con nuestro robot Pololu 3pi, permitiendo visualizar los datos de diferentes sensores obtenidos en cualquier lugar que queramos tomar muestras de campo, logrando una herramienta portátil de comparación y análisis de muestras.

2.- Los sensores ultrasónicos generan un lóbulo de radiación en su frente que en teórica recoge cualquier variación de cuerpos extraños. Sin embargó por efecto de difracción cuerpos que no cumple con cierta altura no son detectados.

3.- La salida analógica del sonar MaxSonar EZ1 tiene una tensión comprendida entre 0 y 2.55 V que representa el valor de la distancia medida cuyo factor empleado es de 10mV/pulgada. El cual puede detectar un objeto hasta 6.45 metros.

4.- Existe un tiempo entre la lectura del sensor y el cálculo de evasión de obstáculos, en que el robot es incapaz de percibir su entorno por lo que se pueden presentar colisiones.

RECOMENDACIONES

1.- Revisar y entender el manual de especificaciones del Pololu y MaxSonar Ez1 para su buen funcionamiento y de esta manera no cometer errores en la conexión de sus pines.

2.-En la utilización del sensor determinar el rango al cual deseamos que nuestro robot funcione correctamente, dependiendo de esto el sensor emitirá los datos correspondientes, lo importante es visualizar los datos emitidos por el sensor para realizar una correcta programación.

3.-Se recomienda utilizar 2 sonares para obtener un mejor rendimiento al momento de evitar obstáculos, debido a que con un solo sensor se pierde muchos puntos de lectura que si tendríamos con mas sensores para obtener mejores resultados en el momento en el cual el robot realice su recorrido.

4.-Una recomendación que cabe recalcar es de leer detenidamente el datasheet del Atmega 328p, para revisar la configuración de sus pines, y así evitar cometer errores en las conexiones, logrando la prevención de quemar algún componente del proyecto.

BIBLIOGRAFÍA

SITIOS WEBS

1. Ingeniería de Microsistemas Programados S.L, Hoja técnica Sonar MaxSonar EZ1, http://www.msebilbao.com/notas/downloads/Manual%20del%20MaxSonar-EZ1.pdf , 30 de Diciembre del 2010

2. Pololu Corporation, Pololu 3pi Guía de usuario, http://www.pololu.com/file/0J137/Pololu3piRobotGuiaUsuario.pdf, 5 de Enero del 2011

3. Electrónica Estudio, Productos Pololu, http://www.electronicaestudio.com/pololu_productos.htm, 5 de Enero del 2011

4. Pololu Corporation, Pololu – Pololu 3pi Robot, http://www.pololu.com/catalog/product/975, 7 de Enero del 2011

5. Pololu Corporation, Pololu robotics forum - View topic 3pi + eyes , http://forum.pololu.com/viewtopic.php?f=29&t=2110 , 15 de Enero del 2011

6. I.E.S. Juan de la Cierva Fernando Remiro Domínguez, Sensores, Introducción a la micro robótica curso ELE11CM07, http://www.iesjuandelacierva.es/~fremiro/Transparencias%20DPPE/SENSORES.pdf , 15 de Enero del 2011

7. Atmel corporation, Atmel AVR8 and 32 bit – megaAVR, http://www.atmel.com/dyn/products/product_card.asp?part_id=4198, 28de Diciembre del 2010

8. MaxBotix Inc,.LV- MaxSonar- Ez1,Maxbotix, http://www.maxbotix.com/uploads/LV-MaxSonar-EZ1-Datasheet.pdf , 25de Enero del 2011

ANEXOS

ANEXO 1

Librerías utilizadas

Librería Pololu.3pi.h

```
#include <pololu/analog.h>
#include <pololu/buzzer.h>
#include <pololu/time.h>
#include <pololu/motors.h>
#include <pololu/lcd.h>
#include <pololu/leds.h>
#include <pololu/pushbuttons.h>
#include <pololu/serial.h>

#define IR_EMITTERS_OFF 0
#define IR_EMITTERS_ON 1
#define IR_EMITTERS_ON_AND_OFF 2

void pololu_3pi_init(unsigned int line_sensor_timeout);
void pololu_3pi_init_disable_emitter_pin(unsigned int line_sensor_timeout);
void read_line_sensors(unsigned int *sensor_values, unsigned char readMode);
void emitters_on();
void emitters_off();
void calibrate_line_sensors(unsigned char readMode);
void line_sensors_reset_calibration();
void read_line_sensors_calibrated(unsigned int *sensor_values, unsigned char readMode);
unsigned int read_line(unsigned int *sensor_values, unsigned char readMode);
unsigned int read_line_white(unsigned int *sensor_values, unsigned char readMode);

unsigned int *get_line_sensors_calibrated_minimum_on();
unsigned int *get_line_sensors_calibrated_maximum_on();
unsigned int *get_line_sensors_calibrated_minimum_off();
unsigned int *get_line_sensors_calibrated_maximum_off();
```

Librería pgmspace.h

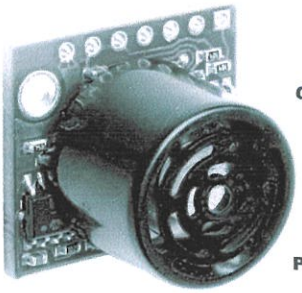
```
#ifndef __PGMSPACE_H_
#define __PGMSPACE_H_ 1

#define __need_size_t
#include <inttypes.h>
#include <stddef.h>
#include <avr/io.h>

#ifndef __ATTR_CONST__
#define __ATTR_CONST__ __attribute__((__const__))
#endif

#ifndef __ATTR_PROGMEM__
#define __ATTR_PROGMEM__ __attribute__((__progmem__))
#endif

#ifndef __ATTR_PURE__
#define __ATTR_PURE__ __attribute__((__pure__))
#endif

/**
   \ingroup avr_pgmspace
   \def PROGMEM

   Attribute to use in order to declare an object being located in
   flash ROM.
 */
#define PROGMEM __ATTR_PROGMEM__

#ifdef __cplusplus
extern "C" {
#endif
```

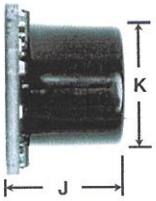
ANEXO 2

Hoja de datos técnicos del sonar MaxSonar EZ1

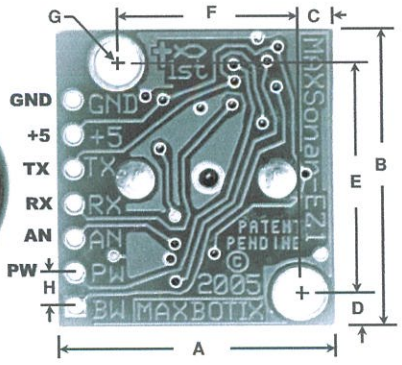
The MaxSonar®-EZ1™ High Performance Sonar Range Finder

The MaxSonar®-EZ1™ offers very short to long-range detection and ranging, in an incredibly small package with ultra low power consumption. The MaxSonar®-EZ1™ detects objects from 0-inches to 254-inches (6.45-meters) and provides sonar range information from 6-inches out to 254-inches with 1-inch resolution. Objects from 0-inches to 6-inches range as 6-inches. The interface output formats included are pulse width output, analog voltage output, and serial digital output.





approximately actual size



weight, 4.3 grams

A	0.785"	19.9 mm	F	0.510"	12.6 mm
B	0.870"	22.1 mm	G	0.124" dia.	3.1 mm dia.
C	0.100"	2.54 mm	H	0.100"	2.54 mm
D	0.100"	2.54 mm	J	0.645"	16.4 mm
E	0.670"	17.0 mm	K	0.610"	15.5 mm

dimensions are nominal

Features

- Continuously variable gain for beam control and side lobe suppression
- Object detection includes zero range objects
- Single 5V supply with 2mA typical current draw
- Readings can occur up to every 50mS, (20-Hz rate)
- Free run operation can continually measure and output range information
- Triggered operation provides the range reading as desired
- All interfaces are active simultaneously
  - Serial, 0 to 5V
    - 9600Baud, 81N
  - Analog (10mV/inch)
  - Pulse width (147uS/inch)
- Learns ringdown pattern when commanded to start ranging
- Designed for protected indoor environments
- Sensor operates at 42KHz
- High output 10V PP square wave sensor drive

Benefits

- Very low cost sonar ranger
- Half the size of other sensors in its class
- Sensor dead zone virtually gone
- No central blind spot
- Quality beam characteristics
- Mounting holes provided on the circuit board
- Lowest power ranger, excellent for multiple sensor or battery based systems
- Can be triggered externally or internally
- Sensor reports the range reading directly, frees up user processor
- Fast measurement cycle
- User can choose any of the three sensor outputs

Beam Characteristics

Sample results for measured beam patterns are shown below on a 12-inch grid. The detection pattern is shown for;

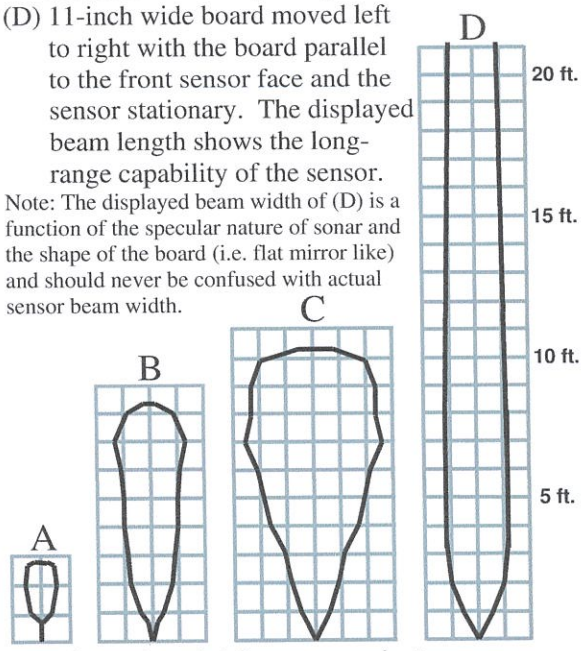
(A) 0.25-inch diameter dowel, note the very narrow beam for close small objects,

(B) 1-inch diameter dowel, dowel, note the long narrow detection pattern,

(C) 3.25-inch diameter rod, note the long controlled detection pattern,

(D) 11-inch wide board moved left to right with the board parallel to the front sensor face and the sensor stationary. The displayed beam length shows the long-range capability of the sensor.

Note: The displayed beam width of (D) is a function of the specular nature of sonar and the shape of the board (i.e. flat mirror like) and should never be confused with actual sensor beam width.



beam characteristics are approximate

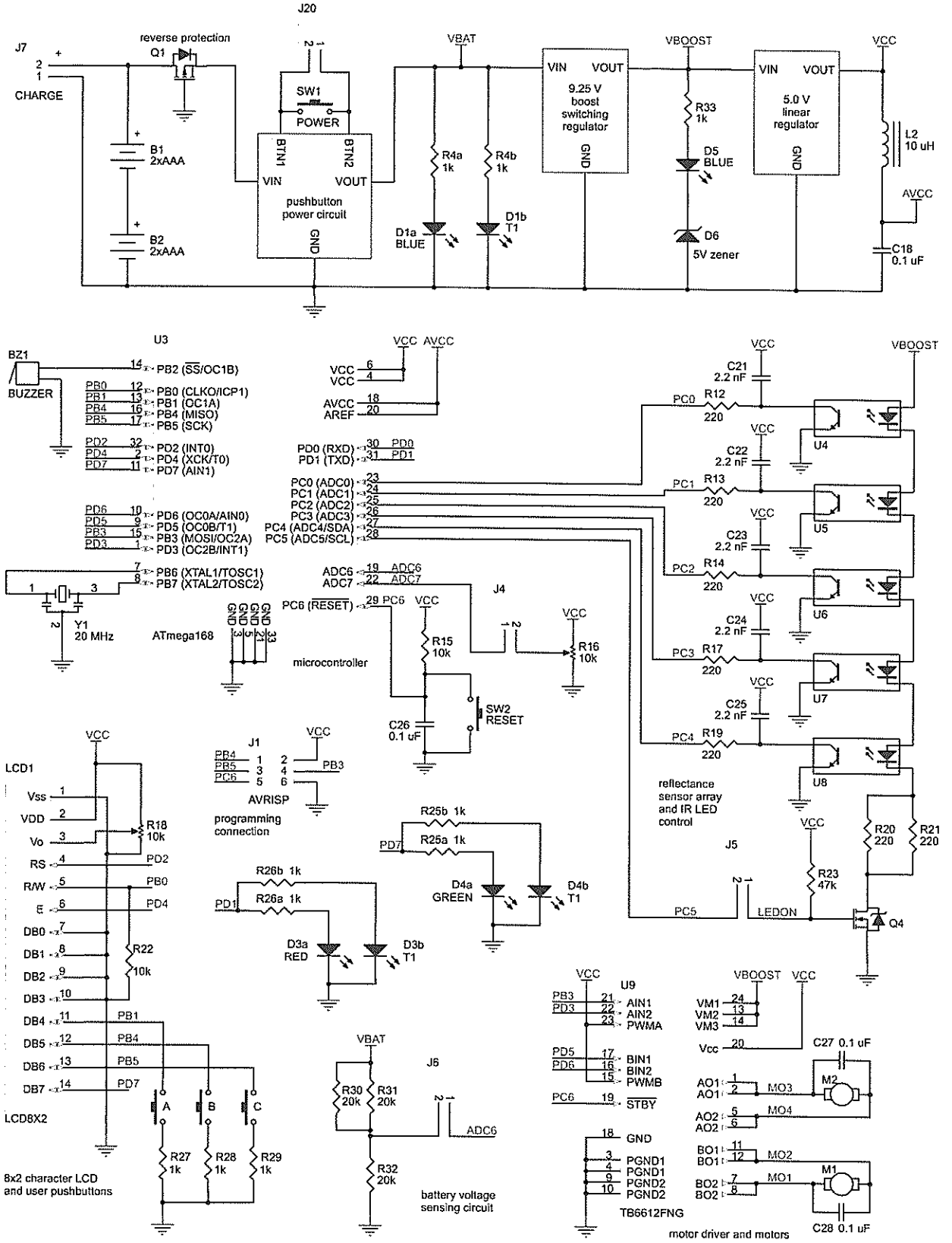
MaxBotix® Inc.

8757 East Chimney Spring Drive, Tucson AZ, 85747 USA
1821 Graydon Avenue, Brainerd, MN, 56401 USA
Email: info@maxbotix.com Web:www.maxbotix.com

ANEXO 3

Hoja de datos técnicos del Pololu 3pi

Pololu 3pi Robot Simplified Schematic Diagram



rrc03a

ANEXO 4

Hoja de datos técnicos del Atmega 328p

Features

- High Performance, Low Power AVR® 8-Bit Microcontroller
- Advanced RISC Architecture
  - 131 Powerful Instructions – Most Single Clock Cycle Execution
  - 32 x 8 General Purpose Working Registers
  - Fully Static Operation
  - Up to 20 MIPS Throughput at 20 MHz
  - On-chip 2-cycle Multiplier
- High Endurance Non-volatile Memory Segments
  - 4/8/16/32K Bytes of In-System Self-Programmable Flash program memory
  - 256/512/512/1K Bytes EEPROM
  - 512/1K/1K/2K Bytes Internal SRAM
  - Write/Erase Cycles: 10,000 Flash/100,000 EEPROM
  - Data retention: 20 years at 85°C/100 years at 25°C(1)
  - Optional Boot Code Section with Independent Lock Bits
    In-System Programming by On-chip Boot Program
    True Read-While-Write Operation
  - Programming Lock for Software Security
- Peripheral Features
  - Two 8-bit Timer/Counters with Separate Prescaler and Compare Mode
  - One 16-bit Timer/Counter with Separate Prescaler, Compare Mode, and Capture Mode
  - Real Time Counter with Separate Oscillator
  - Six PWM Channels
  - 8-channel 10-bit ADC in TQFP and QFN/MLF package
    Temperature Measurement
  - 6-channel 10-bit ADC in PDIP Package
    Temperature Measurement
  - Programmable Serial USART
  - Master/Slave SPI Serial Interface
  - Byte-oriented 2-wire Serial Interface (Philips I^2C compatible)
  - Programmable Watchdog Timer with Separate On-chip Oscillator
  - On-chip Analog Comparator
  - Interrupt and Wake-up on Pin Change
- Special Microcontroller Features
  - Power-on Reset and Programmable Brown-out Detection
  - Internal Calibrated Oscillator
  - External and Internal Interrupt Sources
  - Six Sleep Modes: Idle, ADC Noise Reduction, Power-save, Power-down, Standby, and Extended Standby
- I/O and Packages
  - 23 Programmable I/O Lines
  - 28-pin PDIP, 32-lead TQFP, 28-pad QFN/MLF and 32-pad QFN/MLF
- Operating Voltage:
  - 1.8 - 5.5V
- Temperature Range:
  - -40°C to 85°C
- Speed Grade:
  - 0 - 4 MHz@1.8 - 5.5V, 0 - 10 MHz@2.7 - 5.5.V, 0 - 20 MHz @ 4.5 - 5.5V
- Power Consumption at 1 MHz, 1.8V, 25°C
  - Active Mode: 0.2 mA
  - Power-down Mode: 0.1 µA
  - Power-save Mode: 0.75 µA (Including 32 kHz RTC)

8-bit AVR® Microcontroller with 4/8/16/32K Bytes In-System Programmable Flash

ATmega48A
ATmega48PA
ATmega88A
ATmega88PA
ATmega168A
ATmega168PA
ATmega328
ATmega328P

Summary

Rev. 8271CS–AVR–08/10