ESCUELA SUPERIOR POLITÉCNICA DEL LITORAL

Facultad de Ingeniería en Electricidad y Computación

"Uso del transceptor infrarrojo Pololu Beacon en interfaz con el robot Pololu 3pi en optimización de rutinas de comportamiento coordinado con robot similar."

TESINA DE SEMINARIO

Previa a la obtención del Título de:

Ingeniero en Electricidad con Especialización en Electrónica y Automatización Industrial

Presentado por:

María Leonela Cumbe González.

Vicente Javier Morales Maridueña.

Edison Orlando Pincay Pillasagua

GUAYAQUIL – ECUADOR

AÑO 2011

AGRADECIMIENTO

A Dios.

A nuestros Padres.

A nuestro Profesor Ing. Carlos Valdivieso.

Y todas las personas que de alguna u otra manera nos brindaron su apoyo para la implementación de este proyecto.

DEDICATORIA

A Dios por darme el valor, la fortaleza, la oportunidad de finalizar esta etapa de mi vida de estudios, permitiéndome compartir en este trayecto muchas cosas.

A mi familia y a todas las personas que de alguna manera me han brindado su apoyo por sobre todas las cosas, su comprensión, su amistad, a mis padres especialmente por haber inculcado en mí los valores éticos que tengo, a mis catedráticos que han compartido sus conocimientos.

Edison Orlando Pincay P.

DEDICATORIA

A mi hermano Patricio por haberme dado la oportunidad de alcanzar este reto importante en mi vida.

María Leonela Cumbe G.

DEDICATORIA

A Dios, a mi madre que siempre me ha acompañado en todo momento y por guiarme en toda mi vida, a mi familia por todo el apoyo incondicional que me han brindado en especial a mi padre por haberme dado su apoyo, inculcándome perseverancia con valores éticos y morales con ejemplo de superación para iniciar una vida profesional, a mis amigos y profesores.

Vicente Javier Morales M.

TRIBUNAL DE SUSTENTACIÓN

Ing. Carlos Valdivieso

Profesor de 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 del mismo a la ESCUELA SUPERIOR POLITÉCNICA DEL LITORAL".

(Reglamento de exámenes y títulos profesionales de la ESPOL)

_____

María Leonela Cumbe González.

_____

Vicente Javier Morales Maridueña.

_____

Edison Orlando Pincay Pillasagua.

RESUMEN

La finalidad del presente proyecto es usar el transceptor infrarrojo Pololu Beacon en interfaz con el robot Pololu 3pi en optimización de rutinas de comportamiento coordinado con robot similar para ello se utilizarán varias herramientas de Software tales como un programador para microcontroladores Atmel como lo es el AVR Studio 4 y un simulador de circuitos electrónicos como lo es el Proteus y para la implementación física se utilizará un Pololu 3pi con un par de IR Beacon más un Programador Pololu USB AVR.

En el capítulo uno se describe los antecedentes, descripción del proyecto, descripción del Problema, Aplicación del proyecto y proyectos similares.

El capítulo dos trata de los fundamentos teóricos que afianzan el desarrollo, implementación y optimización del presente proyecto.

En el capítulo tres se mostrará el código de programación pertinente, para luego en el capítulo cuatro culminar con las simulaciones y pruebas del proyecto con sus respectivos análisis.

ÍNDICE GENERAL

ÍNDICE DE TABLAS

ÍNDICE DE FIGURAS

CAPÍTULO I

1. DESCRIPCIÓN GENERAL DEL PROYECTO

1.1. Antecedentes

Antes de comenzar a buscar información sobre el tema del presente proyecto, primero debemos definir el significado de "interfaz y comunicación infrarroja".

Interfaz, punto en el que se establece una conexión entre dos elementos, que les permite trabajar juntos. La interfaz es el medio que permite la interacción entre esos elementos.

En el campo de la informática se distinguen diversos tipos de interfaces que actúan a diversos niveles, desde las interfaces claramente visibles, que

permiten a las personas comunicarse con los programas, hasta las imprescindibles interfaces hardware, a menudo invisibles, que conectan entre sí los dispositivos y componentes dentro de los ordenadores o computadoras.

La Figura 1.1 muestra los tipos de interfaces.

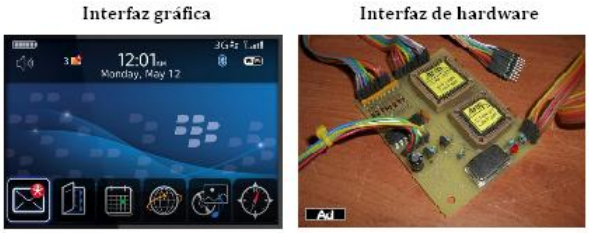


Figura 1.1.- Tipos de Interfaces

La comunicación por infrarrojos utiliza luz infrarroja para transferir datos. La luz infrarroja se utiliza casi universalmente en control remoto para distancia cortas en televisión, vídeo y audio. La comunicación por infrarrojos proporciona una forma rentable de punto a punto de conectar equipos entre sí o con dispositivos y aparatos eléctricos. En la siguiente figura 1.2 se muestra un ejemplo:

Figura 1.2.- Comunicación IR.

Hoy en día existen varias forma de comunicación entre dos máquinas u ordenadores sin las necesidades de un medio específico que los conecten como lo es la comunicación infrarroja, es por esta razón de cómo vamos avanzando en el mundo surge las necesidades de tener una mejor interfaz de comunicación como Wireless, WiFi o GPS, esto hace que la tecnología avance a pasos acelerados, dando confort a los humanos y una mejor interacción entre las máquinas.

Considerando las definiciones anteriormente mencionadas el presente proyecto tiene como finalidad el uso del transceptor infrarrojo Pololu IR Beacon en interfaz con el Pololu 3pi en optimización de rutinas de comportamiento coordinado con robot similar, por el cual se utilizan dos robots; uno como Robot maestro y otro como Robot esclavo, es decir el Robot esclavo debe seguir los movimientos del maestro sin ninguna conexión física que los enlace.

1.2. Descripción del Proyecto.

Para la realización de este proyecto utilizaremos dos transceptores infrarrojos Pololu IR Beacon, uno de ellos deberá realizar la interfaz con el Pololu 3pi al cual lo llamaremos Robot esclavo y el otro Beacon en conjunto con algún Robot Móvil llamado Robot maestro para que emita su orientación o dirección al Robot esclavo. Nos centraremos entonces en programar el Robot esclavo el cual deberá interactuar simultáneamente a los movimientos del Robot maestro.

Los Beacon se utiliza en pares para permitir a un robot la localización de los demás robots. Cada Pololu IR Beacon tiene cuatro emisores de infrarojos que brillan en todas direcciones y cuatro receptores de infrarojos para detectar a otros Pololu IR Beacon con un alcance máximo de 20 pies (6 metros).

En la figura 1.3. muestra una idea básica con relación a nuestro proyecto.

Figura 1.3.- Gato (esclavo) y Ratón (maestro).

Los microcontroladores del Robot esclavo serán programados uno mediante el AVR Studio de Atmel y el otro en PIC-C bajo la plataforma de lenguaje C,

proporcionando al usuario una herramienta para determinar, proveer y realizar distintas tareas o requerimientos de acuerdo a las exigencias del proyecto.

1.3. Descripción del problema.

Conforme avanzamos con la investigación y desarrollo del proyecto surgieron algunas interrogantes las cuales se mencionan a continuación:

- El primer inconveniente, el microcontrolador ATmega 328P del Pololu 3pi no cuenta con las entradas suficientes para realizar una comunicación paralela con el transceptor Pololu IR Beacon.
- Otro problema evitar una colisión entre los robots, esto puede ocurrir al momento que el Robot esclavo alcance la posición del Robot maestro.
- Y por último la parte de montaje de los Pololu IR Beacon, los emisores del Robot maestro deben estar a la misma línea de visualización con los receptores del Robot esclavo, para que no repercuta en la comunicación entre ellos.

1.4. Aplicación del Proyecto.

Una aplicación del proyecto puede ser que el Robot Maestro pueda interactuar con uno o varios esclavos estableciendo un comportamiento simultáneo entre ellos. Recordemos que la comunicación por medio de infrarrojos es más

económica que cualquier otro tipo de comunicación inalámbrica que se tiene en el mercado; pero, no es muy eficiente para proyectos de gran envergadura.

También una aplicación real sería en implementar robots de cargas donde se requiera que los robots esclavos sigan al maestro hasta llevar la carga al punto de destino. Este es un ejemplo de seguridad evitando que el hombre interactúe directamente con la carga, de esta manera se evitaría algún accidente laboral.

1.5. Proyectos similares.

A continuación se muestra tres proyectos con aplicaciones similares utilizando el mismo principio de maestro y esclavo pero con diferentes interfaces de comunicación tales como:

- Beacon localización de Robot.
- Proyecto Comportamientos coordinados en formaciones de robots usando percepción visual y comunicación punto a punto.
- Proyecto Sistemas Robóticos Teleoperados.

1.5.1. Proyecto Beacon localización de Robot.

En este trabajo se utilizaron los Pololu IR Beacon, en la cual un Beacon maestro esta sobre un protoboard como se muestra en la figura 1.4, se lo

coloca en algún lugar como una casa base, mientras que el robot esclavo trate de localizar la ubicación actual de la casa base y llegar a ese lugar; también se debe tener mucho en cuenta que los IR Beacon detectan hasta una distancia máxima de seis metros.

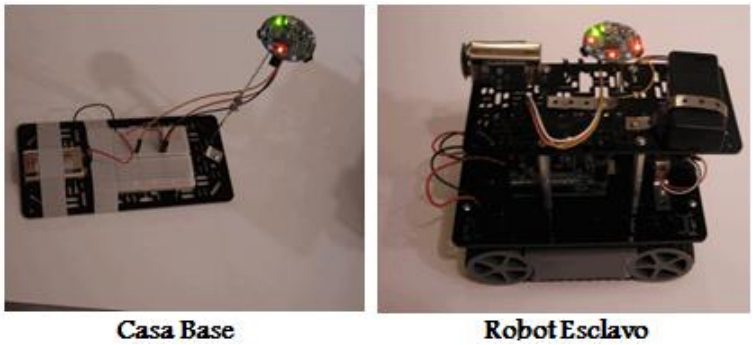


Figura 1.4.- Localización de Casa Base

1.5.2. Proyecto Comportamientos coordinados en formaciones de robots usando percepción visual y comunicación punto a punto.

En este trabajo se presentan varias conductas de coordinación en formaciones de robots con el objetivo de ser utilizadas dentro de tareas de vigilancia en entornos conocidos. Se propone para ello un control distribuido basado en comportamientos que hace uso de percepción local y comunicación restringida, tanto en su ancho de banda como en su alcance. El algoritmo establece y mantiene una red virtual de enlaces de comunicación entre pares de robots próximos en la formación.

Asumimos que el sistema de percepción permite realizar una visualización de un robot y permite detectar marcadores visuales, tanto para planificar la tarea como para identificar el estado de un robot. En la figura 1.5 se muestra la tarea de vigilancia del robot.

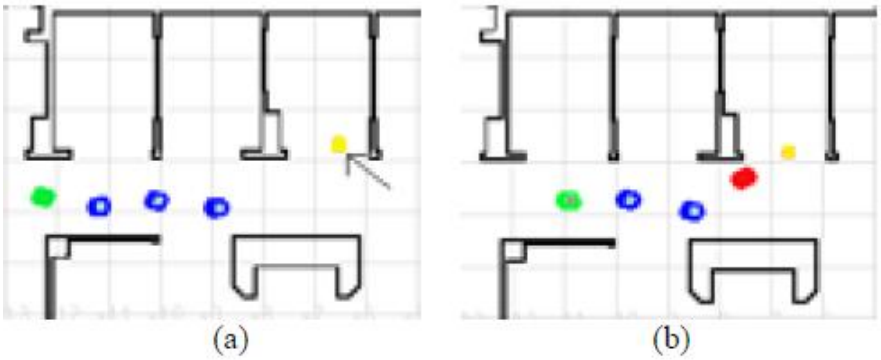


Figura 1.5- Control de tarea de Vigilancia: (a) El líder detecta un punto de vigilancia; (b) se dirige al punto y el resto continúa la marcha con un nuevo líder.

Esto aborda el problema de lograr comportamientos coordinados en un sistema de robots móviles distribuidos. Planteados en un contexto de marcha en formación, con comunicación restringida, y dentro de un entorno de tareas de vigilancia. En un grupo de robots existe coordinación cuando cada robot tiene conocimiento de la presencia de otros robots en el entorno. Se ha conseguido esa coordinación a través de un conocimiento local, tanto a nivel de percepción visual como de comunicación con otros robots. Consideramos además un control distribuido en un grupo de robots homogéneo, en donde la toma de

decisiones se lleva a cabo de una manera autónoma, sin necesidad de que un individuo del grupo o de una unidad central externa organice el trabajo del equipo.

1.5.3. Proyecto Sistemas Robóticos Teleoperados. Los robots teleoperados son aquellos controlados por un usuario a distancia desde una estación remota. Este tipo de manejo supone una ventaja desde el punto de vista de la protección y seguridad del usuario, ya que en caso de realizar trabajos en ambientes inseguros o inestables o con sustancias potencialmente peligrosas, como químicos o explosivos, no se arriesga su integridad física.

Un sistema teleoperado se compone principalmente de una estación de teleoperación, un sistema de comunicación y esclavo, el esclavo puede ser un manipulador o un robot móvil equipado con un manipulador ubicado en un entorno remoto como el robot AndrosWolverine de la empresa Remotec que se ilustra en la figura 1.6. La estación de teleoperación permite controlar al esclavo a distancia por medio del sistema de comunicación, el cual permite transmitir las señales de control hacia el esclavo y, a su vez, recibir señales de información sobre el estado de éste en la estación de teleoperación a través de un canal de

comunicación que puede ser una red de computadores, un enlace de radio frecuencia o microondas. En la figura 1.6 se muestra los componentes de un sistema teleoperado.

Figura 1.6.- Sistemas Teleoperados; (a) Estación de Teleoperación, (b) Robot (esclavo)

CAPÍTULO II

2. FUNDAMENTO TEÓRICO.

En el presente capítulo se describe los fundamentos teóricos básicos de las herramientas de software y hardware utilizados para la implementación de este proyecto.

2.1. Herramientas de Software para el desarrollo del Proyecto.

Los Software requeridos para el desarrollo del proyecto serán descritos a continuación:

AVR Studio 4 de Atmel. Entorno de programación para el microcontrolador ATmega del Pololu 3pi el cual realizará las tareas especificadas por el usuario.

PIC-C. Es un software que nos permite programar en lenguaje C, bajo esta herramienta se desarrolló la programación del microcontrolador 16F886 que se encuentra en el Circuito Moda.

Proteus versión 7.4. Se utilizará para la simulación del proyecto que comprende en interactuar el hardware y software de manera virtual.

En la figura 2.1 se muestra el logotipo de los software a utilizarse.

Figura 2.1.- Requerimientos del proyecto (Software).

2.1.1. AVR Studio 4.

AVR Studio es un entorno de desarrollo integrado (IDE) libre para escribir y depurar aplicaciones AVR en MS entornos Windows. El AVR incluye un ensamblador y simulador de ASM y en combinación con WinAVR, AVR Studio es compatible con C/C++.

Componentes Requeridos.

Con el fin de contar con una plataforma AVR de desarrollo completo se requiere los siguientes componentes:

- GCC: La colección de compiladores de GNU, configurado y compilado para los objetivos de AVR.
- El paquete BinUtils, también con las opciones específicas de AVR habilitadas.
- La biblioteca AVR LibC C
- Un programador (ISP Programmer)

WinAVR. Es un conjunto de herramientas de archivo ejecutable de código abierto de desarrollo de software para la serie de Atmel AVR de microprocesadores RISC alojado en la plataforma Windows. Esto incluye el compilador GNU GCC para C y C + +. WinAVR contiene todas las herramientas para el desarrollo sobre AVR. Incluye el compilador avr-gcc, Programador avrdude, Debugger avr-gdb.

AVRLibc. Es una librería que da acceso a las funciones específicas del AVR, cuyo objetivo es proporcionar una biblioteca de C de alta calidad para su uso con GCC en los microcontroladores Atmel AVR.

Binutils-avr. Es una herramienta que sirve para convertir códigos objeto (object code) a archivos hexadecimales (hex files).

Avrdude. Es el software para manejar el programador.

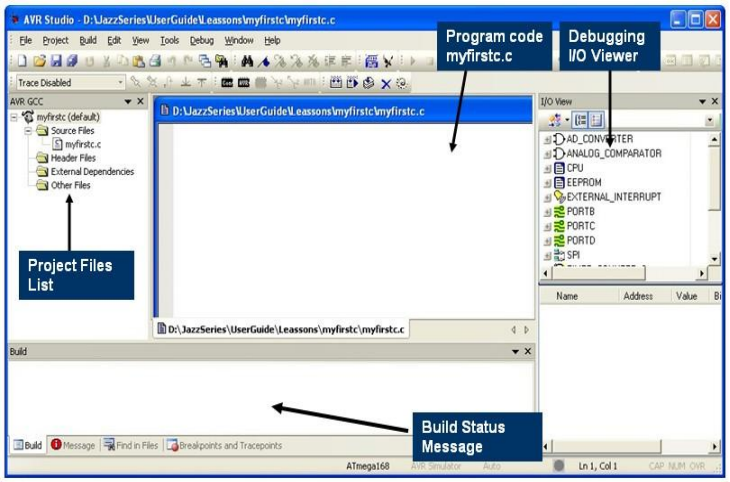


Figura 2.2.- Interfaz gráfica de AVR Studio 4

2.1.2. PIC-C

Inteligente y altamente optimizada compiladores de CCS C contienen operadores estándar C y construido en las bibliotecas de funciones que son específicas a los registros de PIC, proporcionando a los desarrolladores una herramienta poderosa para acceder a las funciones del dispositivo de hardware desde el nivel del lenguaje C. Norma preprocesadores C, los operadores y las declaraciones se pueden combinar con las directivas específicas de hardware y CCS proporciona funciones integradas y las bibliotecas de ejemplo para desarrollar rápidamente aplicaciones que incorpora tecnologías de vanguardia tales como táctil capacitiva, inalámbricas y por cable de comunicación, de movimiento y control de motor y la gestión de la energía.

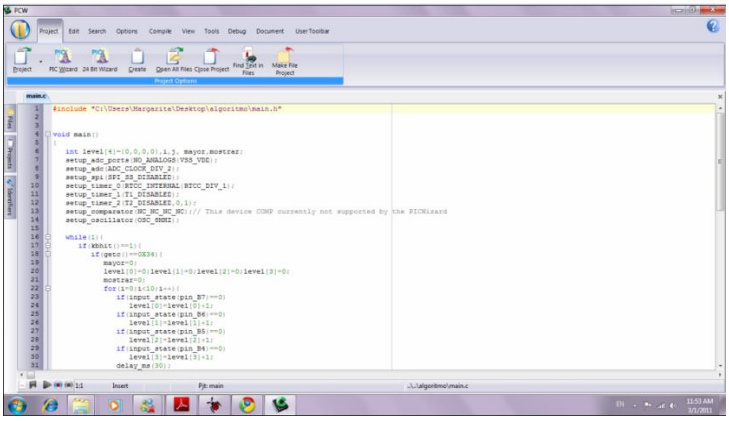


Figura 2.3.- Interfaz Gráfica del PIC C

2.1.3. Proteus versión 7.4

Es una herramienta de software que nos permite elaborar diseños de circuitos electrónicos complejos, donde la cual se incluye la captura de los esquemas y simulación integrando inclusive desarrollos realizados con microcontroladores de varios tipos, es una herramienta de alto desempeño con unas capacidades gráficas impresionantes; sus reconocidas prestaciones lo han convertido en el más popular software de simulación para microcontroladores.

El Programa PROTEUS es una aplicación CAD que se compone de tres módulos básicos:

- **ISIS** (“Esquema Inteligente de Sistema de Entrada”) que es el módulo de captura de esquemas.

- **VSM** (“Modelamiento de Sistema Virtual”) es el módulo de simulación, incluyendo PROSPICE.
- **ARES** (“Modelamiento Avanzado de Enrutamiento”) es el módulo para realización de circuitos impresos (PCB).

PROTEUS, posee entre sus utilidades una variada instrumentación virtual que nos facilita el análisis de circuitos. Estos aparatos se pueden insertar en los circuitos, mostrándonos las medidas en tiempo real según se simula; la apariencia de estos aparatos es semejante a la real, con lo que su manejo es relativamente sencillo, permite simular y depurar el funcionamiento de todo el sistema ejecutando el software paso a paso, insertando puntos de ruptura, verificando el contenido de registros y posiciones de memoria, etc para comprobar si la respuesta del hardware es la correcta.

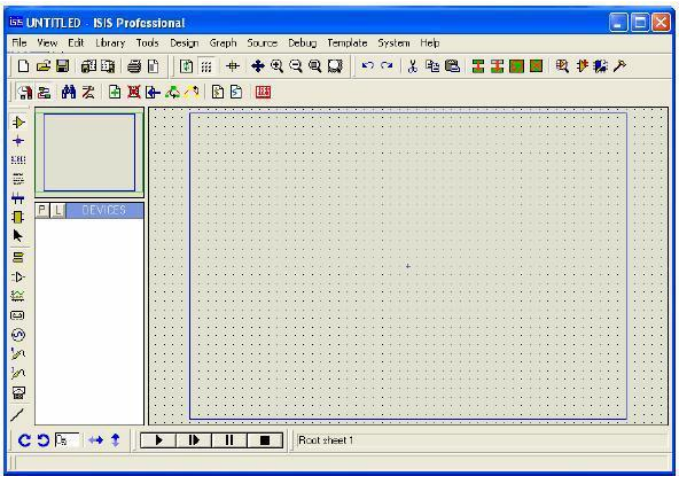


Figura 2.4.- Interfaz gráfica de Proteus

2.2. Herramientas de Hardware para la Implementación del Proyecto.

En la parte de hardware estamos utilizando los módulos detector IR Vishay TSOP34156 junto con el microcontrolador 16F630, estos dispositivos están en el Pololu IR Beacon, en la cual realizará la interfaz por medio del Circuito Moda hacia el Pololu 3pi; para así lograr el objetivo de nuestro proyecto. Para programar al Pololu 3pi utilizaremos un programador Pololu USB AVR y su respectivo cable USB.

En la siguiente figura 2.5 se muestra los requerimientos de Hardware para el presente proyecto.

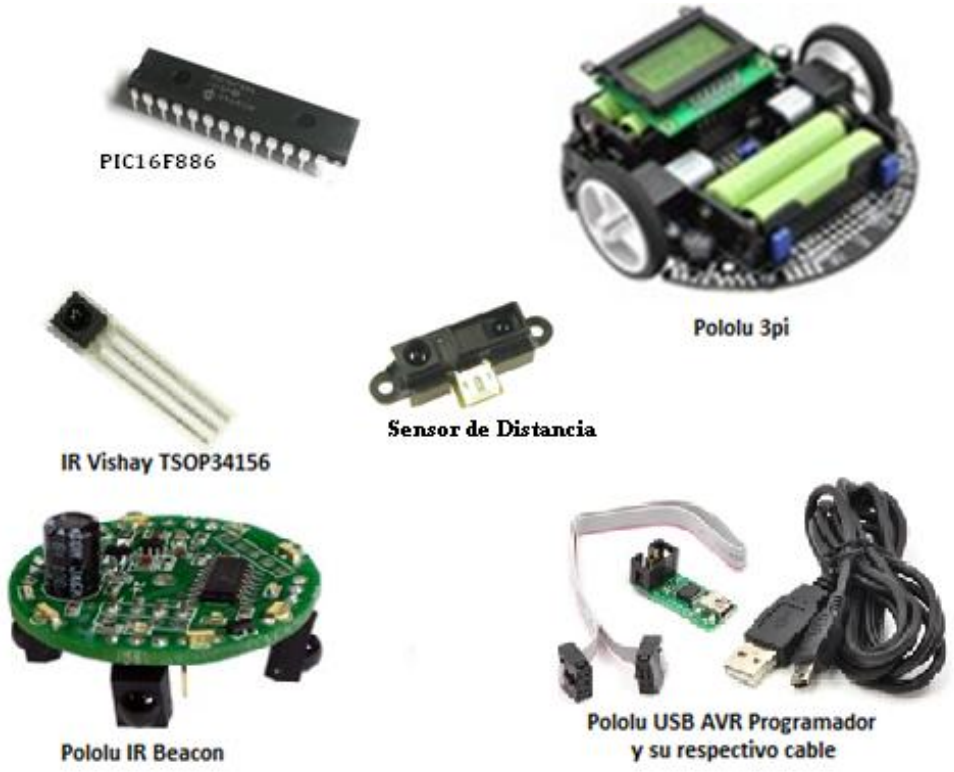


Figura 2.5.- Requerimientos de Hardware

2.2.1. Pololu IR Beacon.

El Pololu IR Beacon es una tarjeta compacta la cual se utiliza en pares para permitir a los robots localizar a otros. Sus especificaciones son las siguientes:

- **Tamaño PCB:** 1.35" diámetro
- **Modulación de Frecuencia IR:** 56 kHz

- **Salida de la Velocidad de actualización:** 20 Hz
- **Rango de Detección:** 0.015 – 6 metros.
- **Voltaje de alimentación:** 6-16 V
- **Dato de Voltaje:** 5 V
- **Número de detectores IR:** 4

Figura 2.6.- Ubicación de los emisores y receptores

Principio de funcionamiento del Pololu IR Beacon. El Pololu IR Beacon funciona de la siguiente manera; realiza la transmisión y recepción de la luz infrarrojas, cada Beacon tiene cuatro emisores y cuatro receptores de infrarrojos los cuales se alternan entre emisores y receptores de infrarrojo para así no confundirse con la reflexión de sus propias transmisión.

Los Beacon tienen cuatro salidas digitales que nos indica cuales de sus cuatro lados detectados del otro Beacon es más fuerte, las cuales

corresponden a los cuatro puntos cardinales. Cuando otro Beacon es detectado, la salida de la dirección adecuada se activa. Este interfaz sencillo permite incluso el control del robot más básico para conectarse al Beacon. Una entrada habilitada permite al Beacon cerrar su transmisión y entrar en un modo de bajo consumo.

A continuación se muestra su diagrama esquemático.

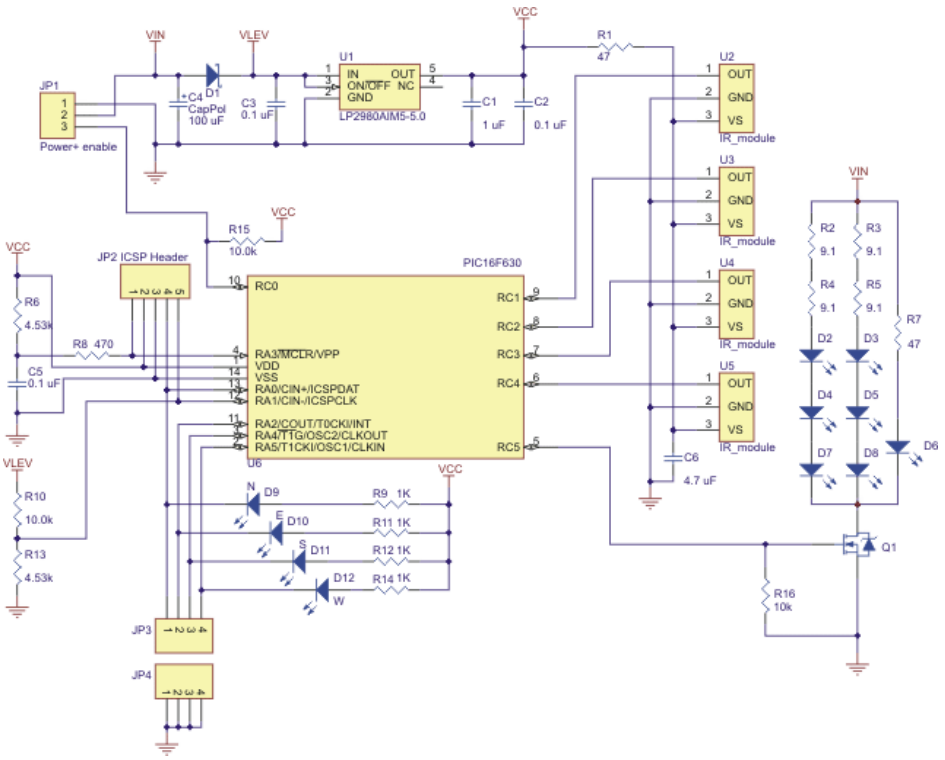


Figura 2.7.- Diagrama Esquemático del IR Beacon.

Módulo detector IR (Vishay TSOP34156). Este dispositivo es el receptor infrarrojo utilizado en Pololu IR Beacon, este sensor cuenta con

un amplio rango de operación de voltaje y permite relativamente altas velocidades de transmisión; el dispositivo esta sintonizado con una frecuencia portadora de 56 KHz. Mostraremos sus especificaciones a continuación:

- **Voltaje de Operación:** 2.7 – 5.5 V
- **Frecuencia portadora (carried):** 56 kHz
- **Máxima velocidad de Datos (Max data rate)**: 4000 bps.

2.2.2. Pololu 3pi.

El Pololu 3pi es un pequeño robot autónomo de alto rendimiento, designado para competiciones de seguimiento de línea y resolución de laberintos.

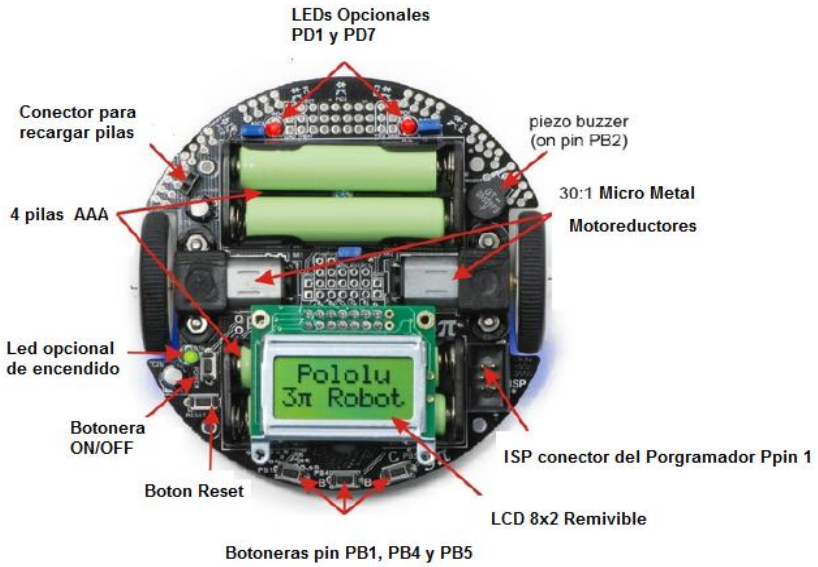


Figura 2.8.- Descripción general del Pololu 3pi

Principio de funcionamiento del Pololu 3pi. El Pololu 3pi contiene un microcontrolador Atmel ATmega 328P, a 20 MHz con 16KB de memoria flash y 1KB de RAM, el doble (32 KB y 2KB) en el ATmega328P y 1KB de EEPROM. En el **Anexo I** se muestra el esquema simplificado del Pololu 3pi.

El uso del ATmega328P 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.

Mover un motor con control de velocidad y dirección, es una característica que tienen los motores, para cambiar de dirección de rotación se debería cambiar la polaridad del voltaje; entonces como no es lógico realizar el cambio de conexión de pilas debemos utilizar el llamado puente H, como veremos en la figura 2.9.

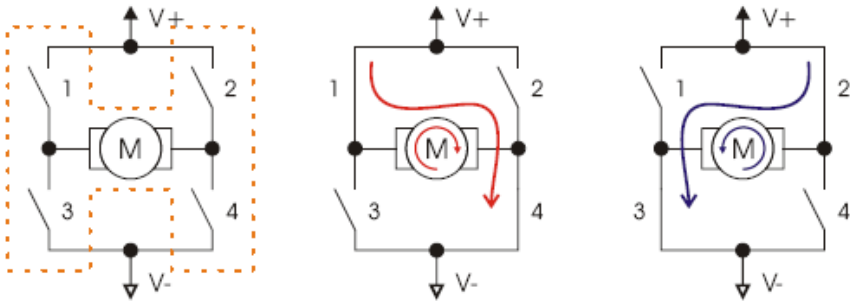


Figura 2.9.- Diagrama del motor DC con puente H

Los cuatro puntos de corte de corriente permiten el cambio de sentido. Se usan puentes H para ambos motores en el Pololu 3pi mediante el chip TB6612FNG conectando las salidas de los puertos del microcontrolador correspondientes a los pines PD5 y PD6 para el motor M1 y para el motor M2 se utilizan los pines de control en PD3 y PB3. Podemos ver su funcionamiento en la siguiente tabla 2.1:

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 2.1.- Funcionamiento de los motores DC.

2.2.3. Microcontrolador ATmega 328P

Este microcontrolador está incorporado en el Pololu 3pi el mismo que será programado para controlar los motores del Pololu 3pi. A continuación se describen las características más relevantes de este dispositivo.

Arquitectura interna

Dentro de su arquitectura interna tenemos; Memoria Flash no volátil de 32 kb, Memoria Sram de 2KB, Memoria EEPROM de 1Kb, posee Port D, C, B, los USART y A/C convertidor.

Características Generales

Tiene una arquitectura RISC, contiene 32 x 8bits Registros de Propósito General de Trabajo, tiene el ALU la cual le permite realizar las operaciones aritméticas y de comparación, las operaciones se realizan a un ciclo por reloj, alta resistencia no-volátiles segmentos de memoria en lo que respecta al espacio de memoria Flash, esta se divide

en dos secciones una de arranque del programa y la otra será la del requerimiento del programa.

En el momento de llamar a las interrupciones y subrutinas, la dirección de retorno del contador de programa (PC) es almacenada en la Pila (Stack).Además contiene 64 direcciones de la CPU para funciones periféricas que son espacio de memoria de entrada y salidas I / O.

En la figura 2.10 se muestra el esquema de los pines del ATmega 328P del Pololu 3pi.

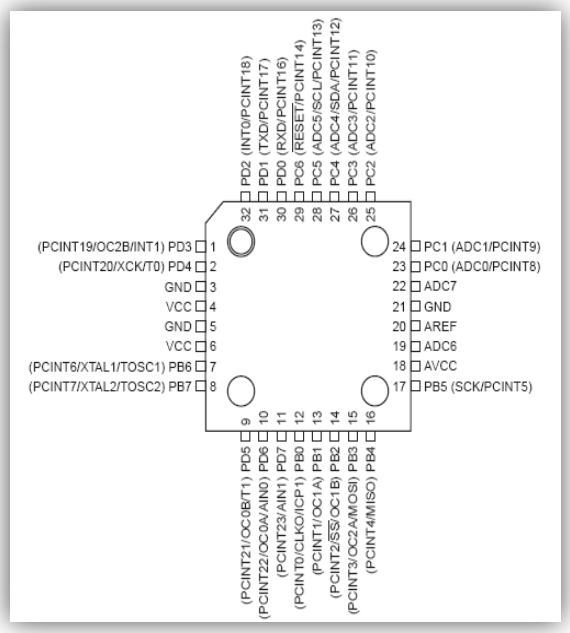


Figura 2.10.- PIC ATmega 328P

En el Anexo II y III se detalla más información sobre las características de este microcontrolador.

2.2.4. Microcontrolador PIC 16F886

Este microcontrolador se encuentra en el Circuito Moda. A continuación se muestra ciertas características de este PIC.

Arquitectura interna. En su arquitectura interna consta de una arquitectura RISC, posee 8 niveles de profundidad en la pila, un rango de frecuencia de 8MHz a 31KHz, convertidor A/D con 10 bit de resolución, temporizadores TMR0, TMR1 y TMR2, modulo de PWM, permite RS-485, RS-232 la cual se está utilizando como comunicación en el proyecto, tiene los Puerto A, B y C.

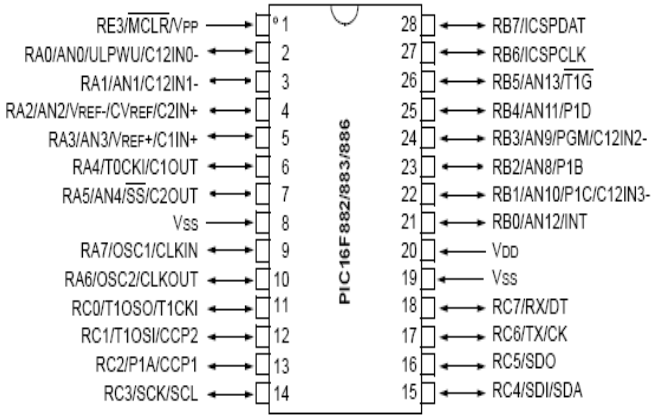


Figura 2.11.- PIC 16F886

En el Anexo IV se detalla más información sobre el microcontrolador.

2.2.5. Pololu USB AVR Programmer

El programador emula un AVRISP v2 en un puerto serie virtual, por lo que es compatible con el software estándar de programación AVR. Tienen dos características adicionales que ayudan a generar y depurar proyectos de aplicación para el monitoreo de señales y los niveles de tensión.

Principio de funcionamiento del Pololu USB AVR Programador. El programador AVR USB se conecta al puerto USB de su ordenador a través de un cable mini-B y se comunica con el software de programación, tales como AVR Studio o AVRDUDE, a través de un puerto COM virtual utilizando el protocolo AVRISPV2/STK500. El programador se conecta al dispositivo de destino a través de un cable de programación de 6 pines ISP.

Figura 2.12.- Pololu USB AVR Programmer

2.2.6. Sensor de Distancia Sharp GP2Y0A21YK0F.

El sensor de distancia Sharp GP2Y0A21YK0F es muy preciso en evitar obstáculos y es muy fácil de usar. Este sensor de infrarrojos es más económico que los ultrasónicos, sin embargo, proporciona un rendimiento mucho mejor que las alternativas de rayos infrarrojos. Interfaz para la mayoría de los microcontroladores es sencilla: la salida analógica puede conectarse a un convertidor de analógico a digital para realizar mediciones a distancia o la salida puede ser conectado a un comparador de umbral de detección. El rango de detección de esta versión es de aproximadamente 20 a 150cm.

El GP2Y0A21YK0F utiliza un conector JST de 3 pines que se conecta a nuestro cable JST de 3 pines. También es fácil soldar tres cables en el sensor en los conectores están montados (véase la imagen inferior a la derecha de la figura 2.13). Al mirar en la parte trasera, las tres conexiones de izquierda a derecha son energía, la tierra, y la señal de salida.

Figura 2.13.- Sensor de Distancia Sharp GP2Y0A21YK0F.

En el Anexo V se detalla más información sobre el microcontrolador.

Características Principales:

- Voltaje de funcionamiento: 4,5 V a 5,5 V
- El consumo promedio actual: 30 mA (típico)
- El rango de medición de distancia: 10 cm a 80 cm (4 "a 32")
- Tipo de salida: tensión analógica
- La salida diferencial de tensión en todo el rango de distancia: 1,9 V (típico).
- Tiempo de respuesta: 38 ± 10 ms

CAPÍTULO III

3. DISEÑO E IMPLEMENTACIÓN DEL PROYECTO

3.1. Diseño preliminar.

Conforme hemos avanzado con la implementación del proyecto se han ido requiriendo más elementos en la parte de hardware como el circuito moda, como circuito de enlace entre transceptor infrarrojo Pololu IR Beacon y el Pololu 3pi, debido a que estos no se pueden conectar directamente entre ellos, las salidas del transceptor IR son cuatro y el microcontrolador del Pololu 3pi nos ofrece tres puertos de entrada por ende no se podría realizar una interfaz de comunicación paralela. La solución es programar el PIC 16F886 para establecer una comunicación serial RS232 con el microcontrolador ATmega 328P del Pololu 3pi. A continuación la figura 3.1 muestra las etapas de interfaz general de los

dispositivos que interactúan en la comunicación.

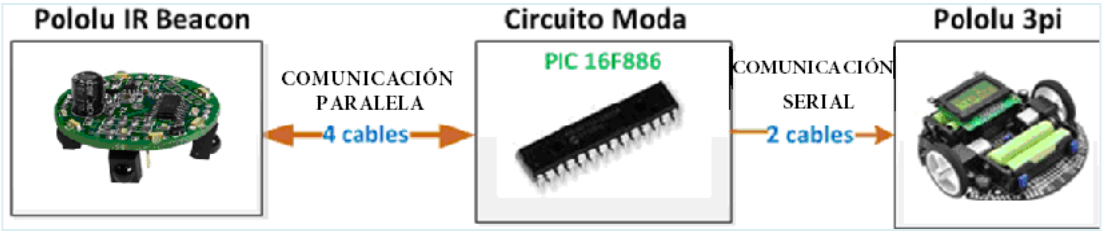


Figura 3.1.- Diseño Preliminar para la interfaz de comunicación.

3.2. Descripción del Proyecto final.

El Robot maestro comprende de un transceptor Pololu IR Beacon montado sobre algún robot móvil como puede ser un seguidor de línea u otro pololu 3pi, este transmitirá la ubicación del mismo para que el Robot esclavo lo recepte por medio de otro Pololu IR Beacon, el cual deberá reaccionar rápidamente a cualquier movimiento del maestro. El robot esclavo comprende un IR Beacon montado sobre un Pololu 3pi del Robot esclavo más una tarjeta electrónica quela hemos llamado Circuito Moda. El Pololu 3pi debe dar el movimiento según la coordenada transmitida por el Robot maestro. Unos inconvenientes al momento de la implantación física se observó que los robo pueden colisionar entre ellos para prevenir esto y evitar que el Robot esclavo de muchas vueltas provocadas por muchas señales infrarrojas, se montó sobre el Robot esclavo un sensor de distancia analógico.

Después de haber realizado las pruebas pertinentes de programación de los microcontroladores y montaje se logró optimizar con alta eficiencia la

implementación del mismo. A continuación la figura 3.2 muestra la implementación del proyecto.

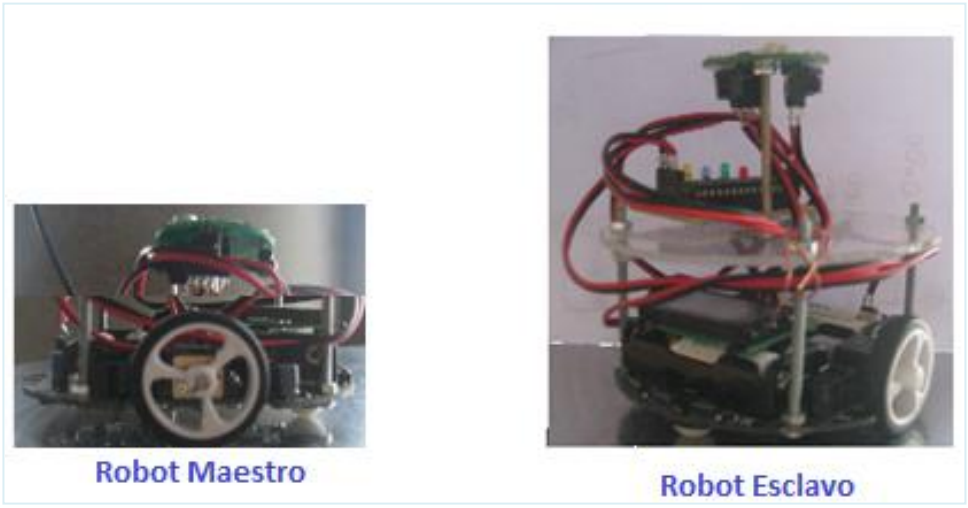


Figura 3.2.- Implementación final del proyecto

3.3. Diagramas de Bloques.

Mostraremos los siguientes diagramas de bloques correspondientes a cada etapa del proyecto.

3.3.1. Diagrama de Bloque del Circuito Moda.

Para un mejor entendimiento se realizó el diagrama de bloque como se indica en la figura siguiente, donde veremos que se realiza una comunicación paralela entre el Pololu IR Beacon y el microcontrolador 16F886 la cual muestra sus salidas conectadas a los LEDs que corresponde a los cuatro puntos cardinales. El diagrama correspondiente se muestra en la figura 3.3.

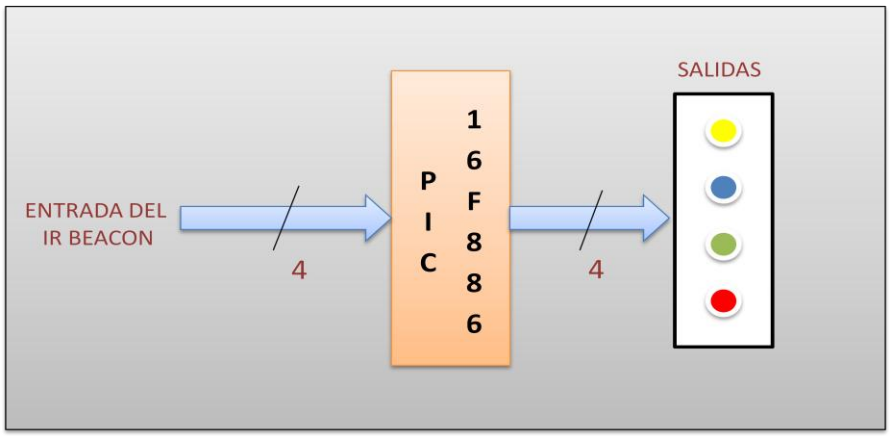


Figura 3.3.- Diagrama de Bloque Circuito Moda.

3.3.2. Diagrama de Bloque del Pololu 3pi.

En esta parte se tomaron muchos parámetros para obtener una buena interfaz de comunicación entre el Pololu 3pi y el Pololu IR Beacon, una de las más importante es la velocidad de transmisión de datos con la que se pretende tener una mejor sincronización evitando perder los datos. En el diagrama de bloque se muestra el criterio de obtención de datos filtrados por el Circuito Moda, y enviar al Pololu 3pi mediante comunicación serial RS232, la orientación correcta para que los motores M1 y M2 correspondientemente se muevan a la dirección procesada por el Circuito Moda. En la figura 3.4 se aprecia el diagrama de bloque.

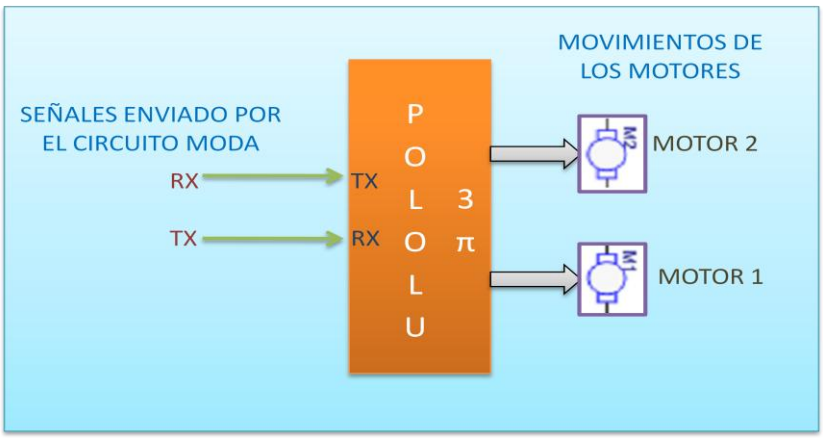


Figura 3.4.- Diagrama de Bloque del Pololu 3pi.

3.4. Algoritmo de Control.

A continuación se describe los algoritmos que sintetiza la idea principal para el desarrollo de la programación.

3.4.1. Algoritmo para Circuito Moda.

En el siguiente diagrama ASM se muestra la lógica utilizada para realizar la programación en el Circuito de Moda. Con este diagrama se pretende realizar el filtrado de las señales no deseables obtenidas en la transmisión y recepción de datos del transceptor Pololu IR Beacon. En la figura 3.5 se muestra su correspondiente diagrama ASM.

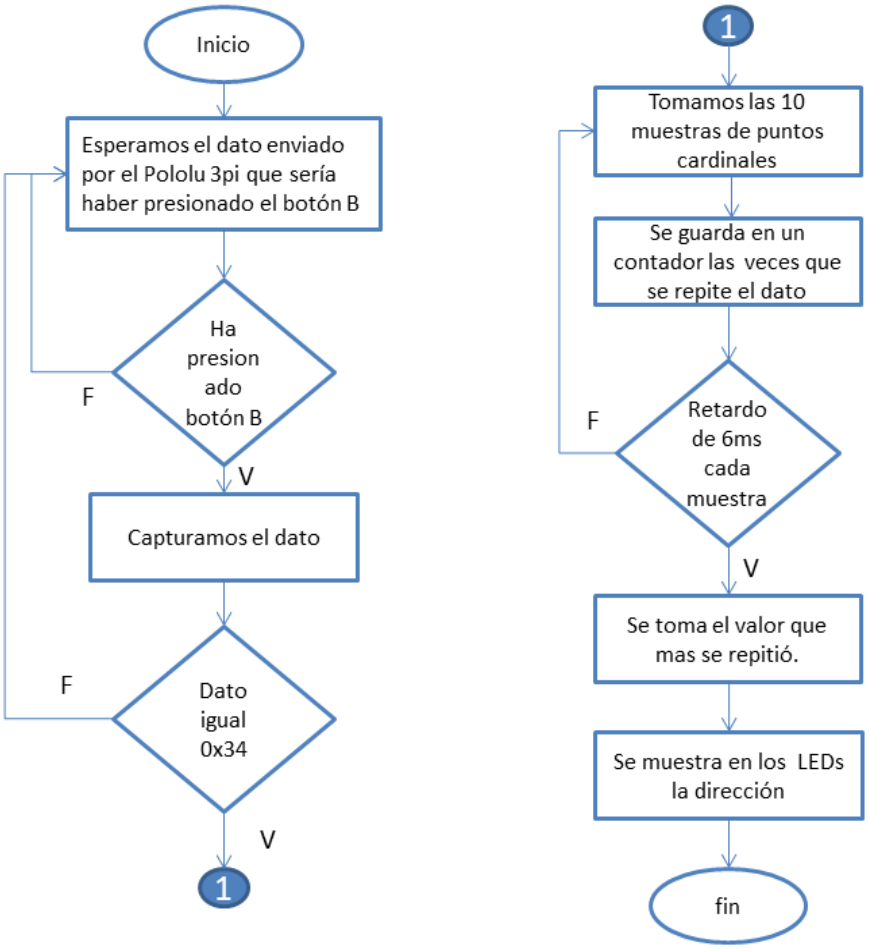


Figura 3.5.- Diagrama ASM del Circuito Moda.

3.4.2. Algoritmo del Pololu 3pi.

Aquí observamos el algoritmo del Pololu 3pi del proyecto donde describe el funcionamiento optimizado de los robots como maestro y esclavo utilizando el microcontrolador PIC 16F886 en interfaz con el Pololu 3pi.

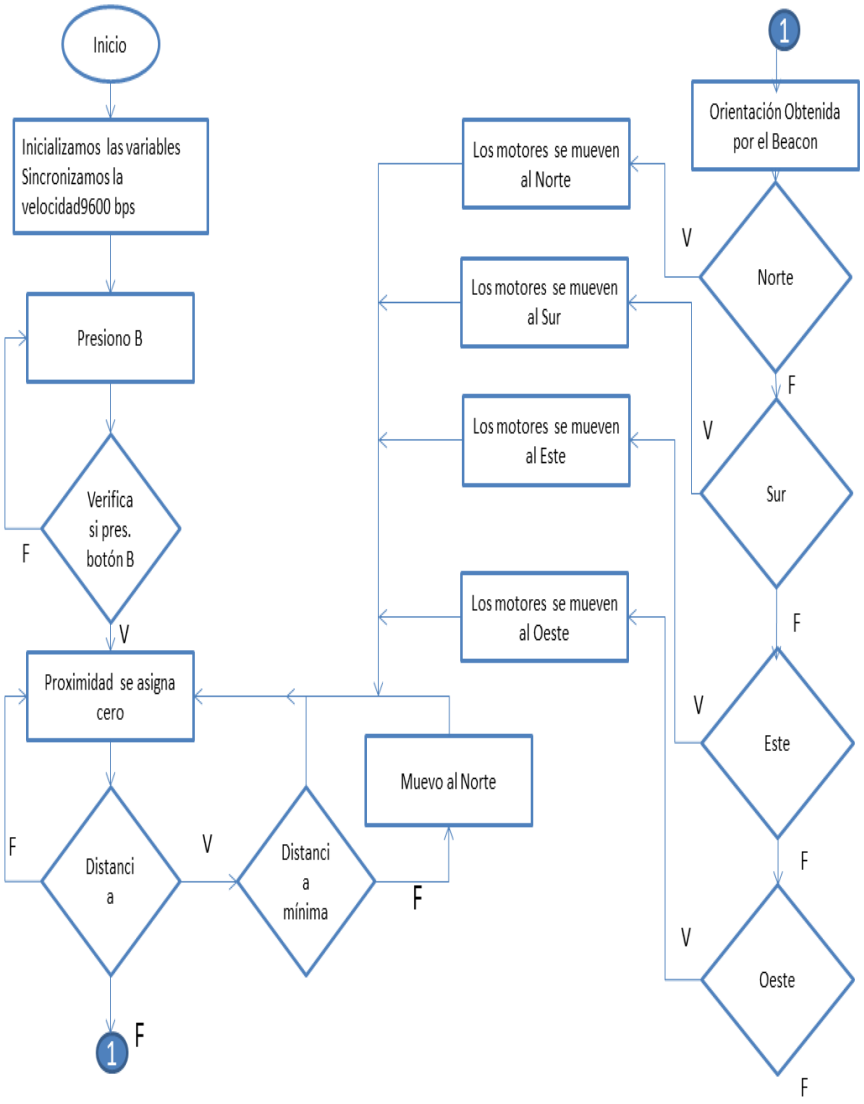


Figura 3.6.- Diagrama ASM del Pololu 3pi.

3.5. Programación del Circuito Moda.

Se programó el PIC 16F886 de Microchip bajo la plataforma de PIC-C de manera que este toma los datos de orientación del Robot maestro recibidos por

el transceptor IR del Robot esclavo para luego ser procesados y validados haciendo el uso de la teoría Moda, además se realizó la interfaz de comunicación USART con el Pololu 3pi para realizar el respectivo movimiento.

A continuación se presenta la programación del código y su respectiva sintaxis.

Programa main.h es donde se genera las especificaciones a utilizarse en el PIC.

```
#include <16F886.h>
#device adc=8

#FUSES NOWDT                    //No Watch Dog Timer
#FUSES HS                       //High speed Osc (> 4mhz for PCM/PCH) (>10mhz for PCD)
#FUSES NOPUT                    //No Power Up Timer
#FUSES NOMCLR                   //Master Clear pin enabled
#FUSES NOPROTECT                //Code not protected from reading
#FUSES NOCPD                    //No EE protection
#FUSES NOBROWNOUT               //No brownout reset
#FUSES IESO                     //Internal External Switch Over mode enabled
#FUSES FCMEN                    //Fail-safe clock monitor enabled
#FUSES NOLVP                    //No low voltage prgming, B3(PIC16) or B5(PIC18) used for I/O
#FUSES NODEBUG                  //No Debug mode for ICD
#FUSES NOWRT                    //Program memory not write protected
#FUSES BORV40                   //Brownout reset at 4.0V
#FUSES RESERVED                 //Used to set the reserved FUSE bits

#use delay(clock=8000000)
#use rs232(baud=9600,parity=N,xmit=PIN_C6,rcv=PIN_C7,bits=8)
```

#include<16F886.h> incluimos la librería del PIC a utilizar.

#device adc=8 utilizamos el dispositivo adc de 8 bits.

#use delay(clock=8000000) usamos un retardo en el reloj de 8 MHz.

#use rs232(baud=9600,parity=N,xmit=PIN_C6,rcv=PIN_C7,bits=8)

Usamos la comunicación RS232 con las siguientes características:

baud = 9600 la velocidad de transmisión de datos a comunicarse.

parity=N paridad ninguna

xmit=PIN_C6 definimos el pin de transmisión

rcv=PIN_C7 definimos el pin de recepción.

bits=8 definimos los bits a utilizarse.

Programa principal del Circuito Moda.

```
        //******Nombre del Proyecto: PROGRAMACION DEL CIRCUITO MODA******//

//******Nombre de los Autores: Leonela Cumbe, Edison Pincay y Vicente Morales******//


#include "C:\Users\leonela\Desktop\algoritmo\main.h"


void main()
{
   int level[4]={0,0,0,0},i,j, mayor,mostrar;
   ////////////picc
   setup_adc_ports(NO_ANALOGS|VSS_VDD);
   setup_adc(ADC_CLOCK_DIV_2);
   setup_spi(SPI_SS_DISABLED);
   setup_timer_0(RTCC_INTERNAL|RTCC_DIV_1);
   setup_timer_1(T1_DISABLED);
   setup_timer_2(T2_DISABLED,0,1);
   setup_comparator(NC_NC_NC_NC);// This device COMP currently not supported by the PICWizard
   setup_oscillator(OSC_8MHZ);
```

#include "C:\Users\leonela\Desktop\algoritmo\main.h" incluimos la dirección donde se encuentra el main.h.

void main() realizamos el procedimiento principal.

int level[4]={0,0,0,0},i, j, mayor, mostrar declaramos a las variable level[4]={0,0,0,0} que es un arreglo de 4, como entero de 8 bits; además también declaramos las variable i, j, mayor y mostrar como entero.

Los setup que se muestran fueron generados al momento de realizar sus características en el main.h.

En esta parte del código se realiza el conteo de las muestra y además su sincronización entre el Pololu 3pi y el Circuito Moda.

```
while(1){//ciclo infinito
   if(kbhit()==1){//retorna 1
      if(getc()==0X34){
         mayor=0;//inicializar
         level[0]=0;level[1]=0;level[2]=0;level[3]=0;
         mostrar=0;
         //obtener datos 10 muestras aplicar moda
         for(i=0;i<10;i++){
            if(input_state(pin_B7)==0)///logica negativa
               level[0]=level[0]+1;/// veces q se repite en las 10 muestras
            if(input_state(pin_B6)==0)
               level[1]=level[1]+1;
            if(input_state(pin_B5)==0)
               level[2]=level[2]+1;
            if(input_state(pin_B4)==0)
               level[3]=level[3]+1;
            delay_ms(6);
         }
```

while(1) un lazo infinito hasta que Pololu 3pi comience a funcionar.

kbhit() ésta función retorna un 1 cuando haya un dato en el buffer y un 0 cuando no tiene dato en buffer.

getc() ésta función se encarga de captura el dato 4 en ASCII enviado por el Pololu 3pi y lo compara con 0x34 en hexadecimal, si esto es verdad entra al lazo for. Podemos mostrar mediante la siguiente tabla que nos permite realizar esta conversión del símbolo 4 a un código hexadecimal.

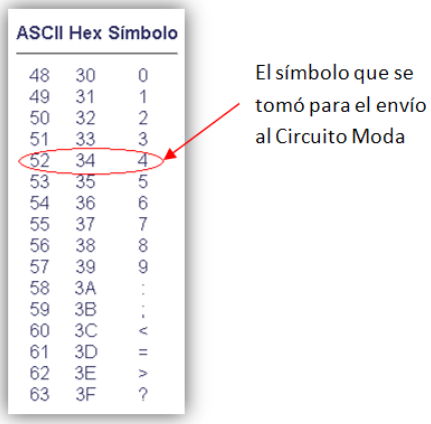


Tabla 3.1.- Código Hexadecimal

Esto se debió hacer ya que la comunicación serial RS232 solamente recibe códigos binarios para entender el símbolo enviado por el Pololu 3pi, se tuvo que realizar la conversión y así se podrán comunicar entre el Pololu 3pi donde se le envía el símbolo y la programación del Circuito Moda donde se recepta y compara ese dato. En el Anexo VI se muestra la tabla completa de la conversión de código ASCII a hexadecimal.

mayor=0 inicializamos la variable.

level[0]=0;level[1]=0;level[2]=0;level[3]=0 inicializamos el arreglo.

mostrar=0 inicializamos la variable.

Se inicializó las variables antes indicada para evitar tomar datos guardados anteriormente que puedan indicar algún dato incorrecto.

for(i=0;i<10;i++) en este lazo es para obtener los datos de las 10 muestras.

input_state() ésta función se encarga verificar el estado del pin de entrada y se compara con 0 debido a que es lógica negativa , los pines de entradas utilizados son: pin_B7, pin_B6, pin_B5 y el pin_B4.

level[]=level[]+1 aquí se lleva el conteo de las 10 muestras obtenidas en cada pin anteriormente mencionada.

delay_ms() ésta función de retardo me indica el tiempo máximo que debe demorar en cada muestra.

A continuación mostraremos la parte principal del código, aquí se realiza la aplicación de la Moda Estadística.

```
//******Àplicacion de la Moda Estadistica******//

 for(j=0;j<4;j++){
   if(level[j]>mayor){
        mostrar=j+1;
        mayor=level[j];
   }
 }
```

for(j=0;j<4;j++) en el lazo for nos permite recorrer los levels.

level[j]>mayor comparamos cada level con mayor si es verdad ingresa a realizar lo indicado.

mostrar=j+1 contamos cada valor de j+1 para ver en los LEDs de salida del PIC del Circuito Moda.

mayor=level[j] el valor que se tiene en level[j] se le asignara a la variable mayor.

Finalmente mostraremos el encendido de los LEDs para visualizar la dirección que tomará el robot.

```
//******Encendidos de led de Orientacion y Envio de datos hacia el Pololu 3pi******//

if(mostrar==1){
   output_high(pin_B0);output_low(PIN_B1);output_low(PIN_B2);output_low(PIN_B3);
   putc(0X20);
}
else if(mostrar==2){
   output_low(pin_B0);output_high(PIN_B1);output_low(PIN_B2);output_low(PIN_B3);
   putc(0X21);
}
else if(mostrar==3){
   output_low(pin_B0);output_low(PIN_B1);output_high(PIN_B2);output_low(PIN_B3);
   putc(0X22);
}
else if(mostrar==4){
   output_low(pin_B0);output_low(PIN_B1);output_low(PIN_B2);output_high(PIN_B3);
   putc(0X23);
}
else{
   output_low(pin_B0);output_low(PIN_B1);output_low(PIN_B2);output_low(PIN_B3);
   putc(0X24);
}
```

mostrar==1 si el valor de mostrar es igual a 1 se encenderá el led correspondiente a esta salida que será 0x20 y las otras tres estarán apagadas.

output_high(pin_B0) ésta función pone un alto a la salida del pin correspondiente y las otras tres tendrán un bajo en sus pines.

putc(0X20) ésta función realiza el envío del dato hacia el Pololu 3pi correspondiente a la orientación oeste.

mostrar==2 si el valor de mostrar es igual a 2 se encenderá el led correspondiente a esta salida que será 0x21 y las otras tres estarán apagadas.

output_high(pin_B1) esta función pone un alto a la salida del pin correspondiente y las otras tres tendrán un bajo en sus pines.

putc(0X21) esta función realiza el envío del dato hacia el Pololu 3pi correspondiente a la orientación norte.

mostrar==3 si el valor de mostrar es igual a 3 se encenderá el led correspondiente a esta salida que será 0x22 y las otras tres estarán apagadas.

output_high(pin_B2) esta función pone un alto a la salida del pin correspondiente y las otras tres tendrán un bajo en sus pines.

putc(0X22) ésta función realiza el envío del dato hacia el Pololu 3pi correspondiente a la orientación este.

mostrar==4 si el valor de mostrar es igual a 4 se encenderá el led correspondiente a esta salida que será 0x23 y las otras tres estarán apagadas.

output_high(pin_B3) esta función pone un alto a la salida del pin correspondiente y las otras tres tendrán un bajo en sus pines.

putc(0X23) ésta función realiza el envío del dato hacia el Pololu 3pi correspondiente a la orientación sur.

3.6. Programación del Pololu 3pi.

Se programó el microcontrolador ATmega 328P en AVR para que controle el movimiento de los motores DC del Pololu 3pi según los datos recibidos por medio de la comunicación serial del Circuito Moda, la cual específica la dirección del Robot maestro.

A continuación se muestra las librerías utilizadas.

Librerías .h exclusivas del pololu/orangután y generales del AVR.

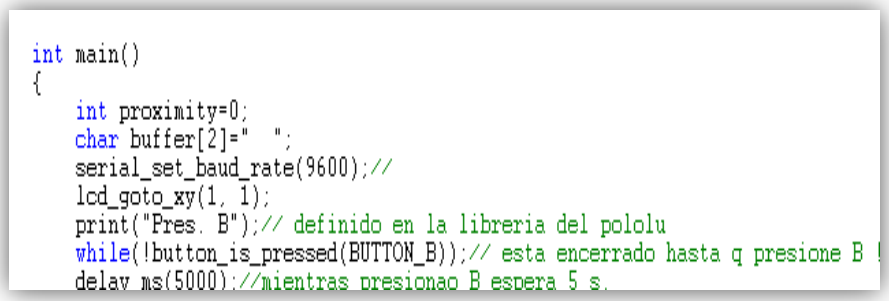
```
//*****Nombre del Proyecto: PROGRAMACION DEL POLOLU 3pi*****//
//***Autores: Leonela Cumbe, Edison Pincay, Vicente Morales***//

#include <pololu/orangutan.h>
#include <avr/io.h>



char dato=0;
```

#include<pololu/orangutan.h> incluimos las librerías del Pololu orangutan para la aplicaciones de las funciones de los motores.

#include<avr/io.h> incluyen las librerías generales de AVR para gcc.

char dato=0 se asigna un cero en la variable caracter dato.

En el siguiente fragmento de programación se tratara de la función principal



```
int main()
{
    int proximity=0;
    char buffer[2]="  ";
    serial_set_baud_rate(9600);//
    lcd_goto_xy(1, 1);
    print("Pres. B");// definido en la libreria del pololu
    while(!button_is_pressed(BUTTON_B));// esta encerrado hasta q presione B
    delay_ms(5000);//mientras presionao B espera 5 s.
```

int main() función principal que nos retorna un entero.

int proximity declaramos a proximity como una variable entera y la inicializamos a cero.

char buffer[2] declaramos una variable arreglo buffer de dos caracter y la inicializamos a cero; en este arreglo se guardaran los datos enviados por el Circuito Moda.

serial_set_baud_rate() en ésta función colocamos la velocidad de transmisión de datos; en donde, se debe tener la misma velocidad de transmisión de datos que en el Circuito Moda, para que los datos que se reciba y transmitan no se pierdan o que aparezca como datos basura.

lcd_goto_xy(1,1) ésta función nos indica en qué posición comenzará la escritura que se mostrara en la pantalla lcd.

print (“Pres. B”) ésta función está declarada en la librería general avr del Pololu y las palabras que están escritas dentro del print se verán en la lcd.

button_is_pressed(BUTTON_B) es una función propia del Pololu que nos indica si el botón_B del Pololu 3pi ha sido presionado. Esta función se encuentra dentro de un lazo while la cual saldrá si la función es cero.

delay_ms() función de retardo de tiempo.

Continuando con la programación, definimos al puerto C para la comunicación del sensor de distancia.

```
while(1)
  {


    DDRC &= ~(1<< PORTC5);
    PORTC &= ~(1<< PORTC5);
    dato=0;
    proximity=0;
    lcd_goto_xy(1, 1);
    buffer[2]="   ";//transmitir serial
```

while (1) entramos a un lazo infinito.

DDRC &= ~(1<< PORTC5) establecemos al PC5 como una entrada.

PORTC &=~ (1<< PORTC5) desactivamos las resistencias de pull up.

dato=0 inicializamos nuevamente la variable.

proximity=0 también la inicializamos para no tener datos guardados anteriormente.

lcd_goto_xy(1,1) ésta función nos indica en qué posición comenzara la escritura que se mostrará en la pantalla lcd.

buffer[2] inicializamos nuevamente esta variable para poder tener los datos seriales transmitidos por el Circuito Moda.

En esta parte del código se agregó los rangos para el sensor de distancia detecte algún objeto u obstáculo en su camino y realice lo que se ha programado.

```
proximity = analog_read(7);// entrada analogica adc7 es una funcion propia del pololu
if(proximity>140){
    print("Objeto");
    if(proximity<500)
        set_motors(-40,-40);
    else
        set_motors(0,0);
}
else{
    serial_send("4",1);// codigo Ascii
    while(serial_receive_blocking(buffer,1, 1000));
    clear();
    dato=buffer[0];
```

analog_read(7) es una función propia del Pololu donde lee la entrada analógica que se tiene en el pin de adc7; en donde, el pin correspondiente a esta ubicación será el 22 del Pololu 3pi, el cual se le asignará a la variable proximity.

proximity>140 comparamos la distancia en donde el sensor ve el objeto u obstáculo.

print (“Objeto”) mostramos en el lcd la palabra contenida dentro de la función print propia del Pololu.

proximity<500 comparamos si la distancia es menor al valor especificado los motores del Pololu 3pi no se detendrán.

set_motors() en ésta función establecemos la velocidad del motor con un máximo de 255.

serial_send() en ésta función envío el código del símbolo 4 por el puerto serie.

serial_receive_blocking(buffer,1, 1000) ésta función recibe una cadena de caracteres que se encuentra en el buffer en un tamaño de un byte durante un segundo.

clear() borra el contenido en la lcd.

dato=buffer[0] guarda la posición cero del buffer en la variable dato.

Esta rutina nos permita mover los motores del Pololu 3pi según la orientación que trasmite vía RS232 el Circuito Moda.

```
            switch(dato)
            {
                case (char)0x20:
                    print("Oeste ");//oeste
                    set_motors(-40,40);
                    break;
                case (char)0x21:
                    print("Norte ");//norte
                    set_motors(-40,-40);
                    break;
                case (char)0x22:
                    print("Este  ");//este
                    set_motors(40,-40);
                    break;
                case (char)0x23:
                    print("Sur   ");//sur
                    set_motors(-40,40);//40 40
                    break;
                case (char)0x24:
                    print("Nada  ");//sur
                    set_motors(0,0);//40 40
                    break;
                default:
                    print("Nada  ");
            }
        }

    }
}
```

Switch**(dato)** conmuta que valor existe en la variable dato y lo deriva a los casos.

case (char) 0x20 convierte y compara si dato es igual a 0x20 esto implica que imprima en la pantalla LCD del Pololu 3pi la orientación “Oeste ”.

set_motors (-40,40) ésta función permite que los motores del Pololu 3pi se muevan a la misma velocidad pero en sentido contrario.

case (char) 0x21 convierte y compara si dato es igual a 0x21 esto implica que imprima en la pantalla LCD del Pololu 3pi la orientación “Norte ”.

set_motors (-40,-40) ésta función permite qué los motores del Pololu 3pi se muevan a la misma velocidad y mismo sentido, es decir con movimiento de

ruedas hacia atrás dando como resultado que el Pololu 3pi se desplace hacia adelante.

case (char) 0x22 convierte y compara si dato es igual a 0x22 esto implica que imprima en la pantalla LCD del Pololu 3pi la orientación “Este ”.

set_motors (40,-40) ésta función permite que los motores del Pololu 3pi se muevan a la misma velocidad pero en sentido contrario.

case (char) 0x22 convierte y compara si dato es igual a 0x23 esto implica que imprima en la pantalla LCD del Pololu 3pi la orientación “Sur ”.

set_motors (-40,40) ésta función permite que los motores del Pololu 3pi se muevan a la misma velocidad pero en sentido contrario, se programó de esta manera ya que tenemos solo un sensor de distancia que se encuentra en la parte posterior del Pololu 3pi controlando que solo se mueva hacia el norte y evita un movimiento hacia atrás.

case (char) 0x24 convierte y compara si dato es igual a 0x22 esto implica que imprima en la pantalla LCD del Pololu 3pi la orientación “Nada ”.

set_motors (0,0) ésta función permite que los motores del Pololu 3pi se queden estáticos.

En el Anexo VII se describe los códigos completos de cada programación.

3.7. Comunicación RS-232

El RS-232 es una interfaz que designa una norma para el intercambio serie de datos binarios entre un DTE (Equipo terminal de datos) y un DCE (Equipo de Comunicación de datos).

La interfaz puede trabajar en comunicación asíncrona o síncrona. La transmisión asíncrona se da lugar cuando el proceso de sincronización entre emisor y receptor se realiza en cada palabra de código transmitido. Esta sincronización se lleva a cabo a través de unos bits especiales que definen el entorno de cada código, mientras que la transmisión síncrona es una técnica que consiste en el envío de una trama de datos (conjunto de caracteres) que configura un bloque de información comenzando con un conjunto de bits de sincronismo y terminando con otro conjunto de bits de final de la trama de datos.

En este caso, el proyecto está dirigido a la transmisión síncrona, ya que los bits de sincronismo tienen la función de sincronizar los relojes existentes tanto en el emisor como en el receptor, de tal forma que estos controlan la duración de cada bit y caracteres.

En la siguiente figura 3.7 se muestra la forma como se realizó la conexión entre el receptor y transmisor entre el PIC y el ATmega 328P que se encuentra internamente en el Pololu 3pi y viceversa.

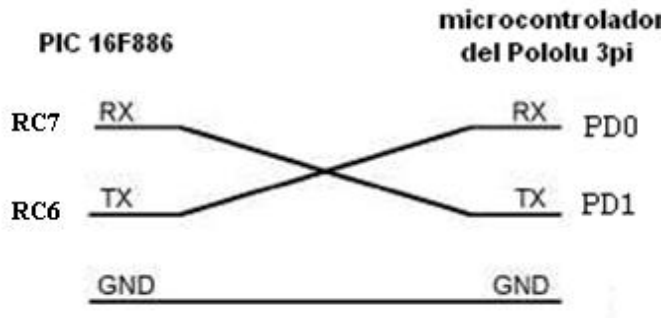


Figura 3.7.-Conexión correcta de los microcontroladores.

CAPÍTULO IV

4. SIMULACIÓN Y PRUEBAS.

4.1. Simulaciones en Proteus.

Antes de comenzar con la implementación física del proyecto utilizamos la herramienta Proteus para que simule el mismo, por medio de esta se observó los tiempos de respuesta y conexión de los pines, además se realizó el diseño del PCB del Circuito Moda a través del Ares que es una aplicación propia del Simulador como se puede ver en la siguiente figura 4.1. en el Anexo VIII se muestra el PCB en 3D.

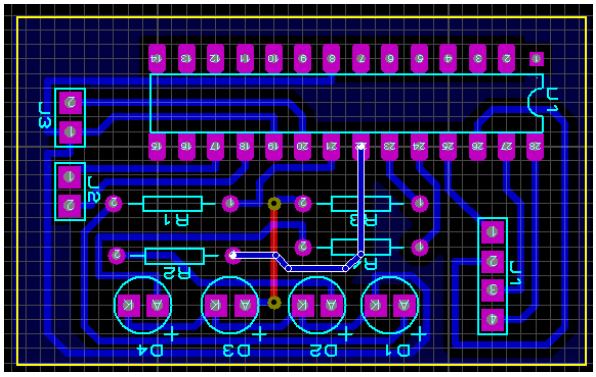


Figura 4.1 PCB Circuito Moda.

A continuación se nombraran y mostraran las simulaciones pertinentes de cada etapa del proyecto.

Simulación del Circuito Moda.

Para realizar la simulación del Circuito Moda en interfaz con el transceptor infrarrojo IR Beacon lo que se hizo es conectar botoneras en los puerto del PB4-PB7 que simule las señales que el IR Beacon debe enviar al PIC16F886 el cual debe realizar el procesamiento de datos para luego ser transmitida vía RS232 al Pololu 3pi. Además se conectaron diodos Leds a la salida de los pines del puerto del PB0 al PB3 del PIC16F886 que reflejan los puntos cardinales Norte, Sur, Este y Oeste, orientación a la cual el Robot esclavo debe actuar.

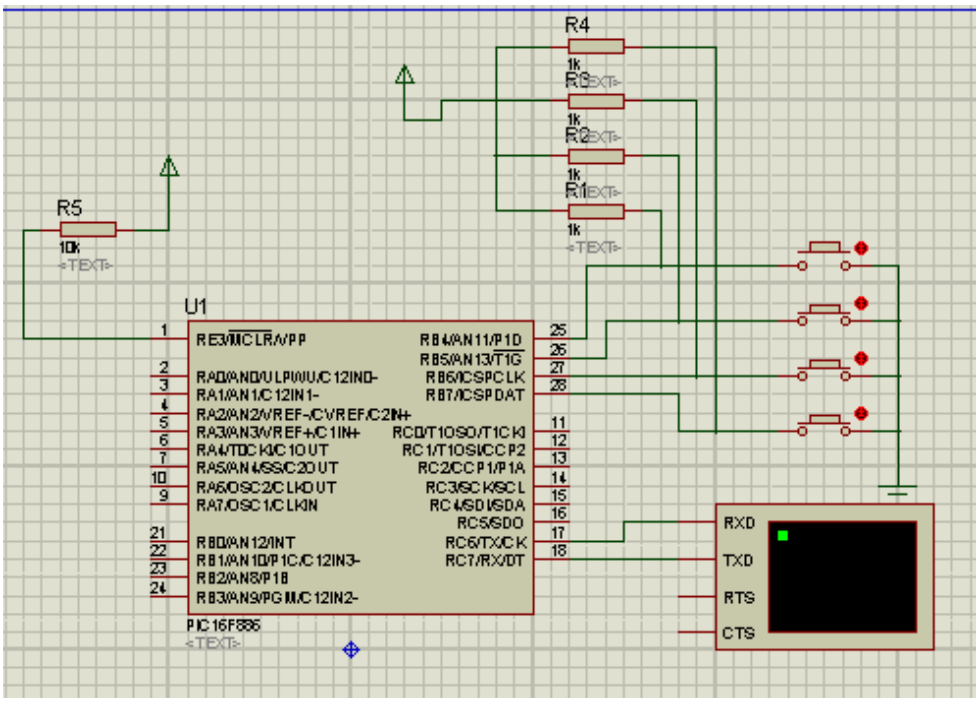


Figura 4.2.- Circuito Moda

Simulación Pololu 3pi

El objetivo de esta simulación es observar cómo interactúan y se procesan los datos entre Circuito Moda y Pololu 3pi, definiendo así la Botonera B del microcontrolador ATmega 328P como un inicio del proceso, este hace la confirmación que se requiere para comenzar la sincronización y comunicación con el PIC 16F886 enviando un código ASCII el número cuatro. Este envió se puede observar por el virtual terminal transformado a código hexadecimal.

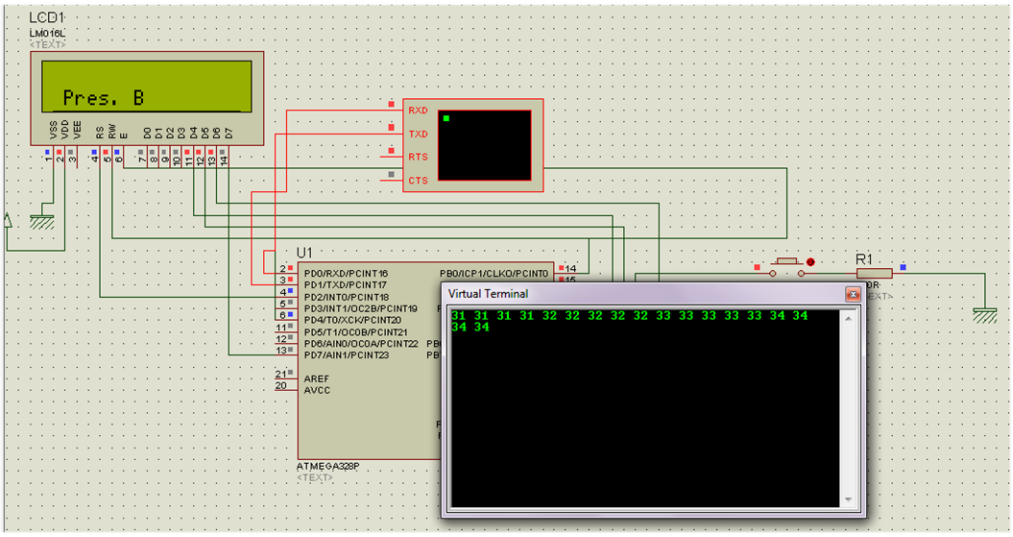


Figura 4.3.- Pololu 3pi

4.2. Pruebas de control de velocidad del motor DC del Pololu 3pi.

Con el fin de optimizar el control de velocidad de los motores del Pololu 3pi se cargaron diferentes códigos de programación. Un ejemplo sencillo es hacer girar los motores de forma manual configurando las botoneras A, B y C del Pololu 3pi como entradas, definiendo así el sentido de giro del motor; hacia adelante botón C, izquierda botón B y derecha botón A. con un retardo de 3 segundos por movimiento.

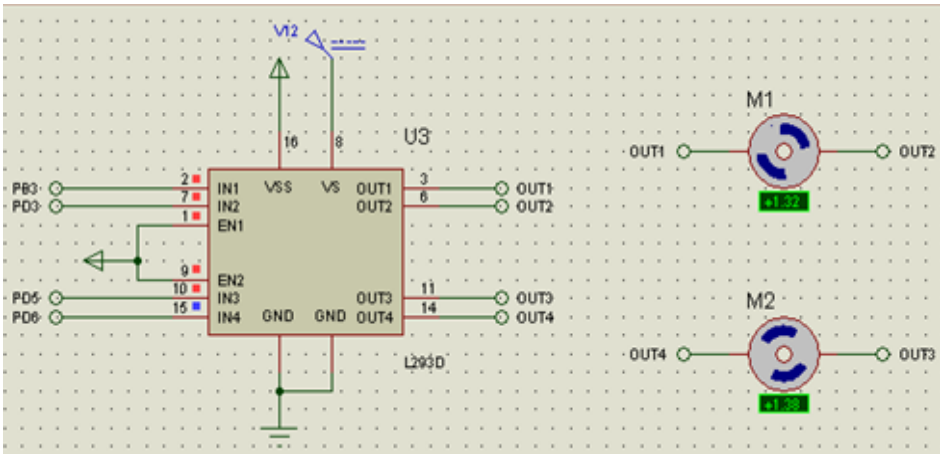


Figura 4.4.- Simulación de Motores DC del Pololu 3pi.

4.3. Pruebas IR Beacon en interfaz con Circuito Moda.

Antes de iniciar a explicar esta etapa del Proyecto primeramente vamos de definir que es la Moda.

En estadística, la Moda es el valor con una mayor frecuencia en una distribución de datos.

Utilizamos Moda ya que el transceptor recepta datos a cada instante y en diferentes posiciones activando así los receptores que indican Norte, Sur, Este u Oeste una y otra vez dando una recepción errónea, entonces con la programación lo que se logra es utilizar los datos reales que son los que más se repiten, de esta manera se obtiene la orientación correcta a la cual el Pololu 3pi esclavo debe seguir. Los cuatro pines de salida del transceptor IR Beacon fueron conectadas hacia los pines del Puerto B del 4 al 7 del PIC 16F886 los

cuales fueron configurados como entradas, también se configuró los pines del Puerto B del 0 al 3 como salidas conectados a cuatro LEDs que indican la orientación del Robot maestro.

En la figura 4.5 se presenta como fue realizada el conexionado de los elementos antes mencionados, los led especifican Norte, Sur, Este y Oeste.

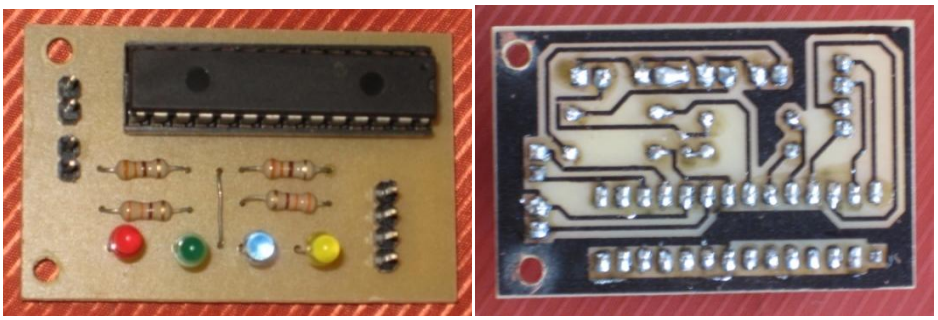


Figura 4.5.- Tarjeta Implementación Circuito Moda

4.4. Pruebas del Circuito Moda en interfaz con Pololu 3pi.

Por medio del Circuito Moda vamos a realizar la interfaz de comunicación USART RS232 configurando el puerto RC7 y RC6 como Receptor (Rx) y Transmisor (Tx) respectivamente del Microchip 16F886 esto permitirá realizar el intercambio de datos con los puerto definidos como PD0 y PD1 que son Recepción y Transmisión respectivamente del Pololu 3pi. La velocidad de los dos microcontroladores están fijadas a 9600 Bps para que estén sincronizados.

4.5. Pruebas Pololu IR Beacon

Se realizaron pruebas de distancias para ver qué sucede con los dos transceptores infrarrojos IR Beacon, cuando están juntos presentan inconvenientes, es decir el Beacon receptor es muy sensible a distancias cortas este activa o recepta datos de forma errónea percibiendo al mismo tiempo todas las direcciones que envía el Beacon transmisor, para nuestro óptima comunicación se realizaron las pruebas para observar a qué distancia esto no afecte al sistema. Entonces a una distancia de 10 cm se eliminó estas fallas.

Figura 4.6.- Pruebas Pololu IR Beacon.

CONCLUSIONES

- Para la realización de nuestro proyecto fue viable utilizar la comunicación infrarroja debido a que se requiere comunicar el Robot maestro y esclavo a distancias cortas menores a seis metros. Además este tipo de comunicación es económica.

- Podemos concluir que mediante nuestro proyecto se ha logrado implementar una interfaz de comunicación sencilla permitiendo a los Pololu IR Beacon y el Pololu 3pi de tal manera que se comuniquen eficientemente, logrando obtener datos enviados, capturados y analizados mediante el circuito implementado.

- Se comprobó mediante experimentos que en la comunicación, transceptores infrarrojos Pololu IR Beacon debe hacerse de una manera específica para que funcione correctamente evitando que tome coordenadas erróneas, para corregir este inconveniente se programó al microcontrolador PIC16F886 bajo los fundamentos teóricos de la Moda estadística con esto se logró obtener la dirección u orientación correcta a la cual el Robot esclavo deba actuar.

- Para el envío de la trama de datos en la comunicación RS232 entre el 16F886 y el ATmega 328P no es necesario utilizar el bit de paridad ya que no existe ninguna

perturbación o ruido cerca del sistema, por ende no se tendría errores en la comunicación de datos. Además estos microcontroladores están conectados por medio un cable a una distancia pequeña más o menos 10 centímetros.

- Luego de haber realizado pruebas con los Robots nos dimos cuenta que al momento que el robot esclavo alcanza la posición del robot maestro los transceptores infrarrojos IR Beacon se alteran evitando todos los emisores a diferentes direcciones, esto provoca que colisionen los robots, por ello fue necesario colocar en el Robot esclavo un sensor de distancia.

- Se puede concluir que el diseño de tarjetas electrónicas con el uso de microcontroladores nos ha facilitado la vida; puesto que estos dispositivos son muy eficientes y se rigen con respecto a sus especificaciones o características de fabricación, gracias a esto hoy en día se han logrado grandes aplicaciones como: la electrónica, robótica, domótica, etc. También podemos palpar la tecnología en nuestra propia casa como es la línea blanca.

- Se puede concluir que el Programa PROTEUS nos permite una excelente visualización del comportamiento del PIC programado dentro del circuito utilizado; a través de la simulación que se puede observar, analizar y mejorar

proyectos relacionado con microntroladores por sus múltiples funciones, por medio de las cuales realizamos un exhaustivo análisis del circuito implementado.

RECOMENDACIONES

- Se recomienda sincronizar la velocidad de transmisión de datos en 9600bps entre los dos microcontroladores para lograr que estos trabajen de forma eficiente y así evitamos fallas al momento de realizar la interfaz de comunicación.

- En el montaje de los Transceptores IR Beacon se deben colocar de tal forma que el emisor este en la misma línea de visualización con el receptor, así logramos que la luz infrarroja incida directamente sobre el receptor evitando que se pierda la señal.

- Es recomendable que para proyectos de gran escala se utilice otro tipo de comunicación inalámbrica como es la Wireless, WiFi o GPS, estas permiten comunicarse a gran distancia.

- Se recomienda realizar las pruebas de los Robots en un lugar donde no incida la luz fluorescente otro tipo de luz similar que puedan alterar o confundir en la comunicación infrarroja, es decir el Beacon puede ver como una señal y actuar de manera aleatoria.

- Se recomienda siempre realizar un Diagrama de flujo que nos permita tener un bosquejo general de la solución, en éste diagrama se deben colocar todos los pasos que vamos a realizar en la ejecución del programa. Este Diagrama debe ser escrito con un lenguaje claro, preciso y conciso para que cualquier persona que no esté familiarizada con los microcontroladores lo pueda entender.

- Se recomienda estar bien informado o familiarizado con los datos teóricos de los diferentes dispositivos que se van a usar en este proyecto puesto que de esta manera se sabrá màs de cada elemento y estaremos menos propensos a errores a la hora de implementarlo o hacer las diferentes pruebas.

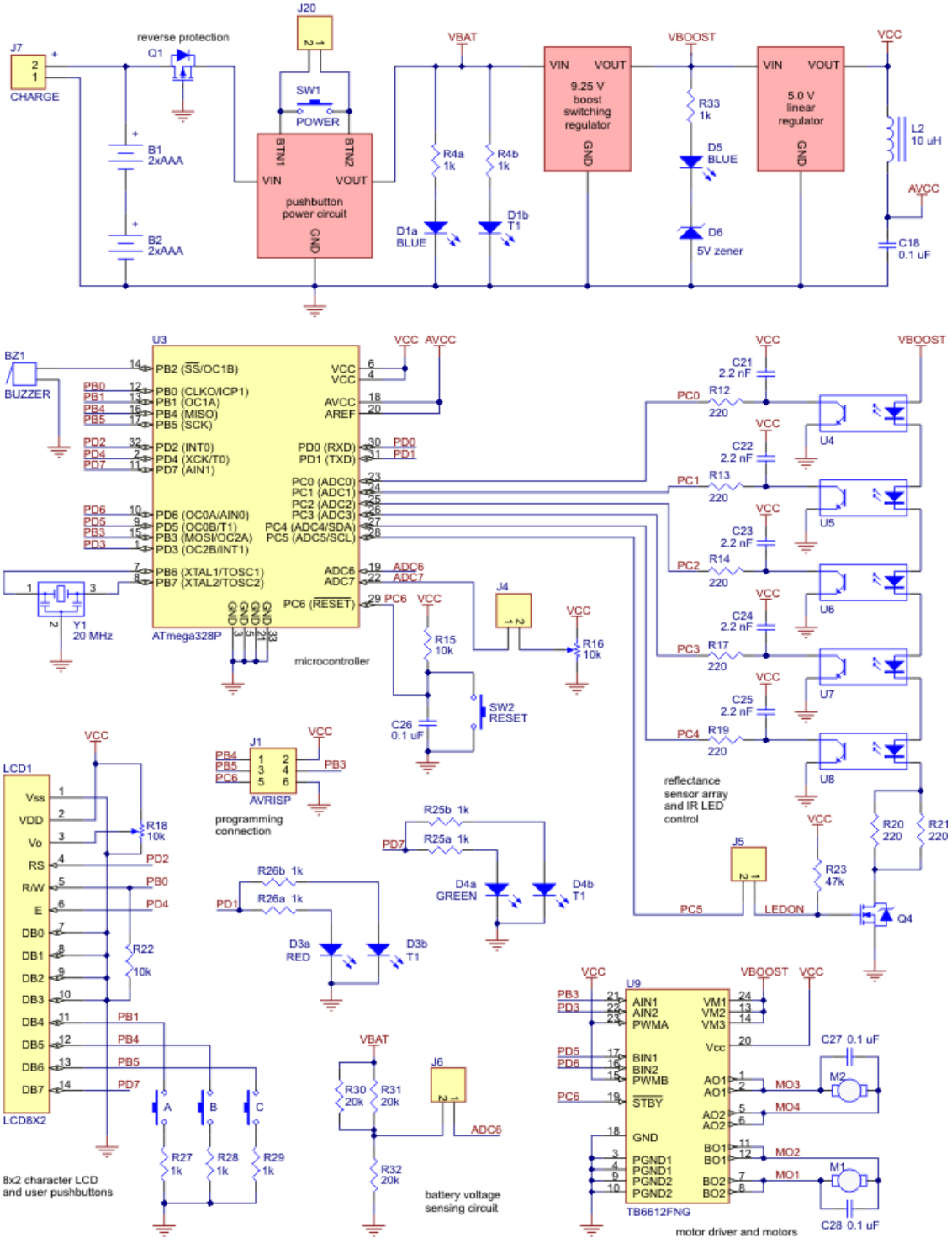
- Se recomienda también tener en cuenta mucho las conexiones en los puntos PD0 (Rx) y PD1 (Tx) del Pololu 3pi con los USART del PIC 16F886 los cuales son pines 17 (Tx) y 18 (Rx), recordar que deben ir cruzados el Rx del microcontrolador PIC 16F886 con el Tx ATmega 328P del Pololu 3pi y el otro de la misma manera, para que se realice de una manera correcta la comunicación serial, la tierra (ground) debe ser común.

- Se recomienda colocar el sensor de distancia paralelo al piso pues un pequeño ángulo de inclinación provoca un error en el funcionamiento del Robot esclavo.

Anexo I

Esquema simplificado del Pololu 3pi.

Esquema simplificado del Pololu 3pi.



Anexo II

Hoja de datos del ATmega 328P.

Anexo III: Tabla de velocidad de transmisión de datos.

Baud Rate (bps)	f_{osc} = 1.0000 MHz				f_{osc} = 1.8432 MHz				f_{osc} = 2.0000 MHz			
	U2Xn = 0		U2Xn = 1		U2Xn = 0		U2Xn = 1		U2Xn = 0		U2Xn = 1	
	UBRRn	Error	UBRRn	Error	UBRRn	Error	UBRRn	Error	UBRRn	Error	UBRRn	Error
2400	25	0.2%	51	0.2%	47	0.0%	95	0.0%	51	0.2%	103	0.2%
4800	12	0.2%	25	0.2%	23	0.0%	47	0.0%	25	0.2%	51	0.2%
9600	6	-7.0%	12	0.2%	11	0.0%	23	0.0%	12	0.2%	25	0.2%
14.4k	3	8.5%	8	-3.5%	7	0.0%	15	0.0%	8	-3.5%	16	2.1%
19.2k	2	8.5%	6	-7.0%	5	0.0%	11	0.0%	6	-7.0%	12	0.2%
28.8k	1	8.5%	3	8.5%	3	0.0%	7	0.0%	3	8.5%	8	-3.5%
38.4k	1	-18.6%	2	8.5%	2	0.0%	5	0.0%	2	8.5%	6	-7.0%
57.6k	0	8.5%	1	8.5%	1	0.0%	3	0.0%	1	8.5%	3	8.5%
76.8k	–	–	1	-18.6%	1	-25.0%	2	0.0%	1	-18.6%	2	8.5%
115.2k	–	–	0	8.5%	0	0.0%	1	0.0%	0	8.5%	1	8.5%
230.4k	–	–	–	–	–	–	0	0.0%	–	–	–	–
250k	–	–	–	–	–	–	–	–	–	–	0	0.0%
Max.[1]	62.5 kbps		125 kbps		115.2 kbps		230.4 kbps		125 kbps		250 kbps	

Anexo IV

Tabla de pines del PIC 16F886.

Anexo V

Sensor de Distancia Sharp GP2Y0A21YK0F.

Anexo VI

Tabla de Conversión ASCII a Hexadecimal

Tabla de Conversión ASCII a Hexadecimal

ASCII	Hex	Símbolo	ASCII	Hex	Símbolo	ASCII	Hex	Símbolo	ASCII	Hex	Símbolo
0	0	NUL	16	10	DLE	32	20	(espacio)	48	30	0
1	1	SOH	17	11	DC1	33	21	!	49	31	1
2	2	STX	18	12	DC2	34	22	"	50	32	2
3	3	ETX	19	13	DC3	35	23	#	51	33	3
4	4	EOT	20	14	DC4	36	24	$	52	34	4
5	5	ENQ	21	15	NAK	37	25	%	53	35	5
6	6	ACK	22	16	SYN	38	26	&	54	36	6
7	7	BEL	23	17	ETB	39	27	'	55	37	7
8	8	BS	24	18	CAN	40	28	(	56	38	8
9	9	TAB	25	19	EM	41	29	)	57	39	9
10	A	LF	26	1A	SUB	42	2A	*	58	3A	:
11	B	VT	27	1B	ESC	43	2B	+	59	3B	;
12	C	FF	28	1C	FS	44	2C	,	60	3C	<
13	D	CR	29	1D	GS	45	2D	-	61	3D	=
14	E	SO	30	1E	RS	46	2E	.	62	3E	>
15	F	SI	31	1F	US	47	2F	/	63	3F	?

ASCII	Hex	Símbolo	ASCII	Hex	Símbolo	ASCII	Hex	Símbolo	ASCII	Hex	Símbolo
64	40	@	80	50	P	96	60	`	112	70	p
65	41	A	81	51	Q	97	61	a	113	71	q
66	42	B	82	52	R	98	62	b	114	72	r
67	43	C	83	53	S	99	63	c	115	73	s
68	44	D	84	54	T	100	64	d	116	74	t
69	45	E	85	55	U	101	65	e	117	75	u
70	46	F	86	56	V	102	66	f	118	76	v
71	47	G	87	57	W	103	67	g	119	77	w
72	48	H	88	58	X	104	68	h	120	78	x
73	49	I	89	59	Y	105	69	i	121	79	y
74	4A	J	90	5A	Z	106	6A	j	122	7A	z
75	4B	K	91	5B	[	107	6B	k	123	7B	{
76	4C	L	92	5C	\	108	6C	l	124	7C	|
77	4D	M	93	5D	]	109	6D	m	125	7D	}
78	4E	N	94	5E	^	110	6E	n	126	7E	~
79	4F	O	95	5F	_	111	6F	o	127	7F	00 1F

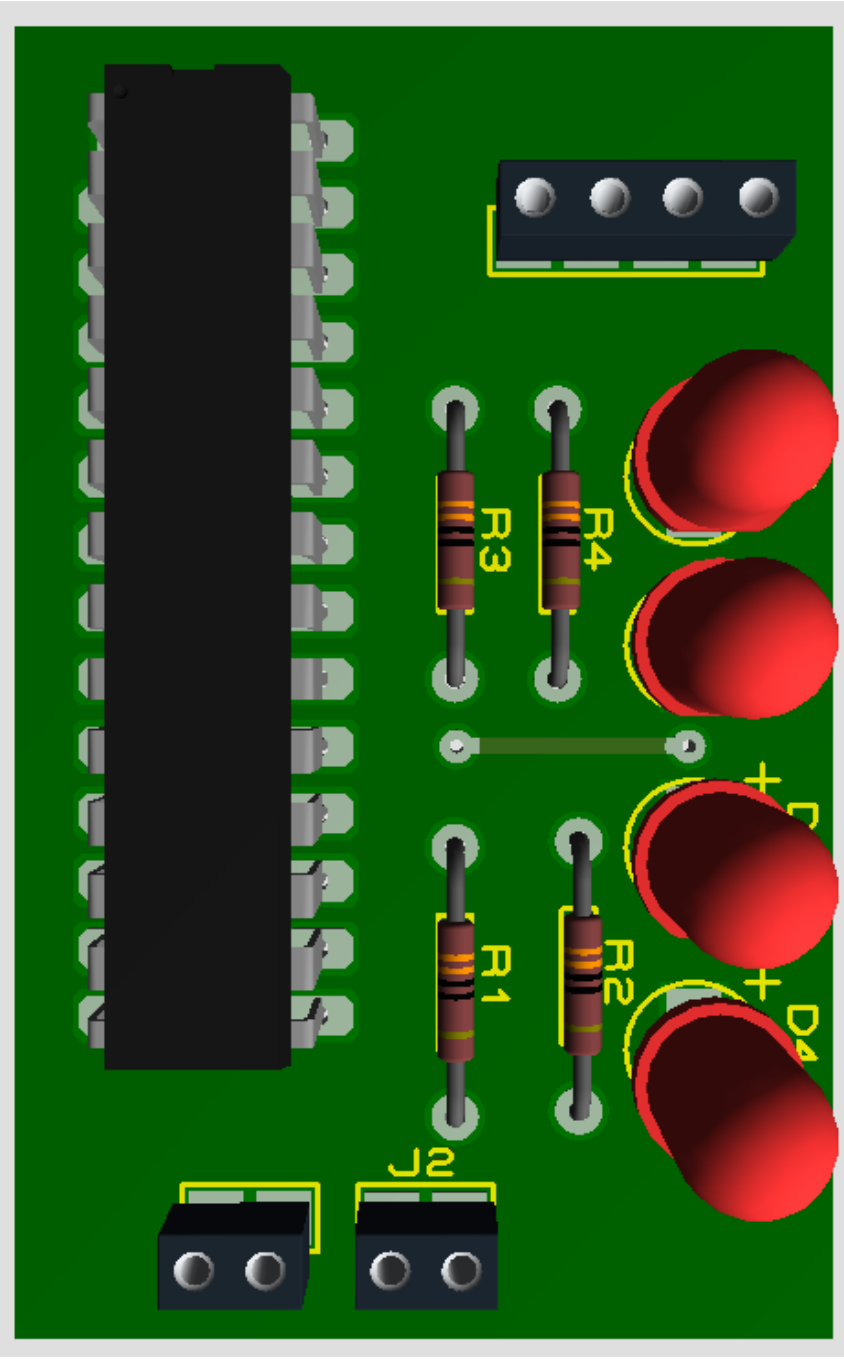
Anexo VII

Programación Total

Anexo VIII

Vista 3D del Circuito Moda

Vista 3D del Circuito Moda



Bibliografía

1. Pololu Robotics & Electronics, Pololu 3pi:

   http://www.pololu.com/catalog/product/975, Fecha de Consulta: 17/01/11.

2. Pololu Robotics & Electronics, Pololu IR Beacon:

   http://www.pololu.com/catalog/product/701, Fecha de Consulta: 17/01/11.

3. Pololu Robotics & Electronics, Librerías de usuario Pololu AVR C/C++:

   http://www.pololu.com/docs/0J20, Fecha de consulta: 18/01/11.

4. Wikipedia, Información general RS232:

   http://es.wikipedia.org/wiki/RS-232, Fecha de Consulta: 20/01/11.

5. Hoja de especificaciones del microcontrolador ATmega 328P:

   http://www.alldatasheet.com/view.jsp?Searchword=ATMEGA328P, Fecha de Consulta: 19/01/11.

6. Hoja de especificaciones del PIC16F886.

   http://www.alldatasheet.com/view.jsp?Searchword=PIC16F886, Fecha de Consulta: 8/02/11.

7. Sharp, Sensor de Distancia;
http://document.sharpsma.com/files/gp2y0a21yk_e.pdf, Fecha de Consulta: 23/02/11.

8. Alexander Cerón Correa, SISTEMAS ROBOTICOS TELEOPERADOSÑ
http://www.umng.edu.co/www/resources/r15_05.pdf, Fecha de Consulta: 14/02/11.

9. ROBOT IS HAPPY, Beacon Locating Robot- Powered;
http://www.robotishappy.com/2009/12/beacon-locating-robot-powered-by-arduino-and-ir-transceiver/. Fecha de Consulta: 15/02/11.

10. Custom Computer Services CCS, Información sobre PIC-C;
http://www.ccsinfo.com/content.php?page=compilers, Fecha de Consulta: 24/02/11.