UNIVERSIDAD TECNOLÓGICA ISRAEL

CARRERA DE ELECTRÓNICA Y TELECOMUNICACIONES

TEMA: ESTUDIO, DISEÑO E IMPLEMENTACIÓN DE UN PROTOTIPO DE ROBOT EXPLORADOR Y ENRUTADOR DE CABLE.

TRABAJO DE GRADUACIÓN PREVIO A LA OBTENCIÓN DEL TÍTULO DE INGENIERO EN ELECTRÓNICA Y TELECOMUNICACIONES

AUTOR: CARDENAS MOGOLLÓN STALIN RODRIGO

TUTOR: ING. MAURICIO ALMINATI

Quito, Febrero 2014

UNIVERSIDAD TECNOLÓGICA ISRAEL

APROBACIÓN DEL TUTOR

En mi calidad de Tutor del Trabajo de Graduación certifico:

Que el trabajo de graduación **"ESTUDIO, DISEÑO E IMPLEMENTACIÓN DE UN PROTOTIPO DE ROBOT EXPLORADOR Y ENRUTADOR DE CABLE"**, presentado por CARDENAS MOGOLLÓN STALIN RODRIGO, estudiante de la Carrera de Electrónica y Telecomunicaciones, reúne los requisitos y méritos suficientes para ser sometido a la evaluación del Tribunal de Grado, que se designe, para su correspondiente estudio y calificación.

Quito D. M., Febrero 2014

TUTOR

Ing. Mauricio Alminati V.

UNIVERSIDAD TECNOLÓGICA ISRAEL

AUTORÍA DE TESIS

El abajo firmante, en calidad de estudiante de la Carrera de Electrónica y Telecomunicaciones, declaro que los contenidos de este Trabajo de Graduación, requisito previo a la obtención del Grado de Ingeniería en Electrónica y Telecomunicaciones, son absolutamente originales, auténticos y de exclusiva responsabilidad legal y académica del autor.

Quito D.M., Febrero 2014

CARDENAS MOGOLLÓN STALIN RODRIGO

CC: 172302068-9

UNIVERSIDAD TECNOLÓGICA ISRAEL

APROBACIÓN DEL TRIBUNAL DE GRADO

Los miembros del Tribunal de Grado, aprueban la tesis de graduación de acuerdo con las disposiciones reglamentarias emitidas por la Universidad Tecnológica Israel para títulos de pregrado.

Quito D.M., Febrero 2014

Para constancia firman:

TRIBUNAL DE GRADO

PRESIDENTE

MIEMBRO 1

MIEMBRO 2

AGRADECIMIENTO

Agradezco a Dios por haberme permitido llegar hasta este punto tan importante de mi vida que es la culminación de mis estudios en la Carrera de Electrónica y Telecomunicaciones. Además, agradezco a mis padres y a mis hermanos que han estado junto a mí en todo momento, con su cariño y dándome fuerzas para conseguir este objetivo tan anhelado.

DEDICATORIA

Va dedicado principalmente a mi familia que han estado cerca y que me han ayudado a superarme cada día, además a todos mis compañeros y amigos, y a los profesores de Electrónica de la UISRAEL que con su apoyo y pasión por la carrera han sabido guiarme a lo largo de mis estudios.

INDICE

ÍNDICE DE TABLAS

ÍNDICE DE FIGURAS

RESUMEN

En el presente documento se muestra el diseño e implementación de un robot explorador y enrutador de cable, este proyecto tiene como finalidad ayudar a técnicos que realizan cableado estructurado, a enrutar cable en lugares en donde no es posible el acceso.

Este prototipo consta de las herramientas necesarias para guiar cable y además monitorear la tarea que se está efectuando en tiempo real.

En el capítulo 1 se detalla los antecedentes del problema que se va a resolver y los objetivos que se tiene que alcanzar.

En el capítulo 2 se muestra toda la teoría requerida para el desarrollo del robot, detallando las características generales de los componentes usados.

En el capítulo 3 se explica el estudio y diseño del sistema, lo que permite el montaje y la implementación del robot, con las explicaciones necesarias.

En el capítulo 4 se muestra el análisis de las pruebas realizadas bajo los parámetros establecidos; además se presenta un análisis FODA del sistema implementado; y finalmente se presenta una lista de costos de los equipos utilizados en el proyecto.

En el capítulo 5 se presentan las conclusiones y recomendaciones sacadas del proyecto.

ABSTRACT

This paper shows the design and implementation of a robot explorer and cable router, this project aims to help technicians performing structured cabling to route wiring in places where access is not possible.

This prototype has the necessary tools to guide cable and also monitor the task that is being performed in real time.

In Chapter 1, the background of the problem to be solved and the objectives that must be achieved is detailed.

In chapter 2 the whole theory required for the development of robot shown, detailing the general characteristics of the components used.

In Chapter 3 the study and design of the system is explained, allowing installation and implementation of the robot, with the necessary explanations.

In Chapter 4 the analysis of the tests performed on the parameters set shown, plus a SWOT analysis of the implemented system is presented, and finally a list of costs for equipment used in the project is presented.

Drawn conclusions and recommendations of the project are presented in Chapter 5.

CAPÍTULO 1

PROBLEMATIZACIÓN

1.1 Antecedentes

Desde sus inicios, la empresa NEWRA Consultancy Cía. Ltda. a jugado un papel importante en las mejores redes de comunicación del país. Se ha creado infraestructura de cableado estructurado que enlaza a personas con la tecnología a través del proceso de evolución. La cartera de soluciones para end-to-end incluye lo que nuestros clientes aspiran para construir un alto rendimiento en redes cableadas e inalámbricas.

Por mucho que la tecnología cambia, el objetivo sigue siendo el mismo: ayudar a los clientes a crear, innovar, diseñar y construir más rápido y mejor. Nunca se deja la conexión y la evolución de las redes para el negocio y la vida diaria.

Los clientes buscan a la empresa NEWRA Consultancy Cía. Ltda. como asesor de confianza que trabaja con ellos para abordar sus retos de negocio más críticos dentro de su centro de datos, Enterprise, y ambientes Industriales. La sólida reputación por la calidad y liderazgo tecnológico, junto con el sólido ecosistema permite ofrecer soluciones integrales que unifican la infraestructura física para ayudar a los clientes a lograr los objetivos operativos y financieros.

Al observar la intensa labor de las empresas del País por mejorar la infraestructura interna de sus redes de comunicación, cada vez que se realiza un diseño para estos requerimientos nunca se termina con todas las

necesidades; sino que al contrario las empresas van creciendo y con ello incrementar sus redes de comunicación.

Cuando una empresa cuenta con toda su infraestructura terminada y todos sus acabados, éstos no permiten a las empresas de servicios de cableado estructurado realizar con comodidad su función para cumplir con las nuevas necesidades del cliente; teniéndose que destruir parte de los acabados para poder realizar esta tarea, lo que implica mayores gastos y tiempo de trabajo de los técnicos.

1.2 Problema Investigado

En todas las edificaciones nuevas o que entran en remodelaciones ya sea por expansión, hoy en día se requiere tener una infraestructura bien implantada para las redes de datos y comunicación interna de una empresa; cada grupo de trabajo que interviene en estos procesos toman decisiones según el requerimiento del cliente y en este caso la empresa que realiza el cableado estructurado necesita primordial atención ya que para realizar su trabajo necesita cierto espacio y tiempo.

Cuando una obra se ha dado por terminada casi siempre surgen necesidades imprevistas de parte del cliente como colocar nuevos puntos de red, para realizar esta nueva petición se hace indispensable destruir parte de los acabados de la edificación ya que técnicos y herramienta necesaria ya no pueden ingresar ni maniobrar en los tumbados donde se tiene que llegar con el cable de red.

Se ha investigado y analizado que los robots en la actualidad ayudan de una forma extraordinaria a resolver ciertos problemas relacionados con la industria

y la ingeniería, mostrando gran versatilidad y fiabilidad a la hora de trabajar en terrenos en donde la mano del hombre no puede llegar fácilmente dando lugar a la intrusión de mecanismos robóticos como exploradores que cuentan con varios parámetro tecnológicos los cuales permiten detallar de una forma aproximada la situación de cualquier elemento del cual se requiere información, estos sistemas robóticos han funcionado de manera alámbrica, mostrando un problema a la hora de explorar largas distancias.

1.2.1 Problema Principal

No existe un sistema robótico que facilite a los técnicos realizar el cableado estructurado en tumbados que se encuentren terminados.

1.2.2 Problemas Secundarios

- No hay estudios sobre las características técnicas de los elementos necesarios para la creación de un robot explorador para cableado estructurado.
- Se carece de un diseño de sistemas robóticos con las características dadas en el proyecto.
- No se ha implementado un sistema robótico para cableado estructurado.

1.3 Justificación

Con este trabajo de graduación se incorporó el conocimiento de varios temas en la rama de Ingeniería Electrónica, Mecánica y Mecatrónica con lo cual se desarrollará nuevas opciones para solucionar problemas o necesidades que van surgiendo en la vida diaria del ser humano.

Mediante este proyecto se obtuvo una base tecnológica para la creación de nuevos robots y satisfacer las necesidades de este tipo de trabajos técnicos.

El presente trabajo ayuda a optimizar el tiempo y los recursos utilizados por las empresas de cableado estructurado y de inmobiliario al realizar cableado en infraestructuras donde los técnicos no pueden llegar, evitando destruir y posteriormente arreglar ciertos acabados en las edificaciones en donde se requieran nuevos puntos de red.

De manera clara en la actualidad los sistemas robóticos son la forma más efectiva de mostrar como la tecnología interactúa con el hombre para satisfacer sus necesidades y abre un mundo de alternativas.

1.4 Objetivos

1.4.1 Objetivo Principal

Estudiar, Diseñar e Implementar un prototipo de robot explorador y enrutador de cable.

1.4.2 Objetivos Específicos

- Estudiar las características técnicas de los elementos necesarios para la creación de un robot explorador para cableado estructurado.
- Diseñar un prototipo de robot explorador para enrutar cable en tumbados de difícil acceso.
- Implementar un robot explorador de acuerdo a los requerimientos del proyecto.

1.5 Metodología

El proyecto se realizó mediante cuatro etapas de investigación, que se describen a continuación:

En la Primera etapa se utilizó los métodos de Análisis y Síntesis para la recopilación de los datos necesarios en la investigación del proyecto.

En la Segunda etapa se tomó en cuenta el método de Modelación que ayudó a realizar el diseño del robot explorador que se basó en la realidad.

En la Tercera y Cuarta etapa se usó el método experimental que permitió verificar el comportamiento del robot explorador para así llegar a ponerlo en marcha.

CAPÍTULO 2

MARCO TEÓRICO

2.1 Arduino

Arduino se define como una plataforma de electrónica abierta para el desarrollo de prototipos basada en hardware y software flexibles y fáciles de usar. Se creó para diseñadores, artistas, aficionados y cualquier usuario que desee crear objetos interactivos u entornos.

Arduino extrae información del entorno por medio de sus pines, de una amplia gama de sensores y tiene la facultad de afectar aquello que lo rodea controlando luces, motores y muchos otros actuadores. El microcontrolador dispuesto en la placa Arduino se programa usando el lenguaje de programación de Arduino. Los proyectos desarrollados con Arduino suelen ejecutarse sin necesidad de conectar a un PC, si bien tienen la posibilidad de realizarlo y comunicar con diferentes tipos de software como: (p.ej. C++, C+, Flash, Processing, MaxMSP).[1]

2.2 Módulos De Transmisión Inalámbrica (XBEE)

Fig.2.1 Módulo Xbee

[1] http://www.arduino.cc/es/

Una de las formas más prácticas de adicionar conectividad inalámbrica a un proyecto es utilizando los módulos Xbee; estos proveen 2 divisiones amigables de comunicación el modo API que expone varias ventajas y modo AT Transmisión serial transparente. Los módulos Xbee pueden programarse desde un PC usando el software X-CTU o también usando un microcontrolador.

Estos dispositivos se pueden comunicar con arquitecturas punto - punto, punto a multipunto o en una red mesh. La elección del módulo Xbee efectivo, se basa en escoger el tipo de antena (chip o conector SMA) y la potencia de transmisión (2 mW para 300 pies o 60 mW hasta 1 milla).

Un módulo Xbee puede ser usado con acopladores Xbee Explorer Serial o Xbee Explorer USB. Los microcontroladores que trabajan con 5V necesitarán de una interfaz (Xbee regulada) para comunicarse con los módulos Xbee. [2]

2.3 Servomotor

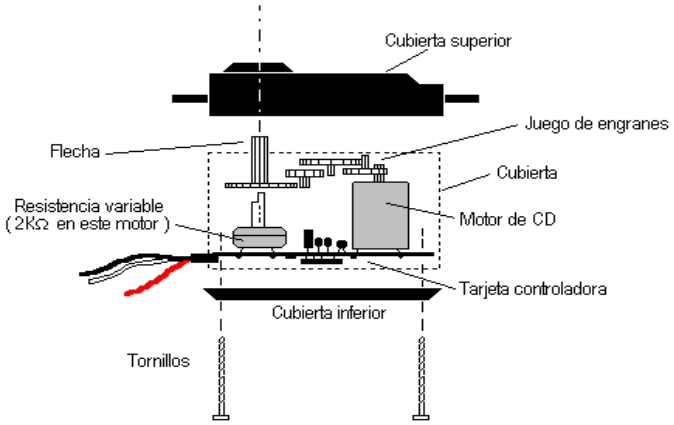

Fig.2.2 Esquema de un Servomotor

Un servomotor es un dispositivo que presenta las mismas características de un motor de corriente continua, con la capacidad de ubicarse en una posición dentro de su rango de operación, y mantenerse estable en dicha posición.

Los servomotores se utilizan frecuentemente en robótica y en sistemas de radio

[2] http://www.xbee.cl/

control, pero su uso no está limitado a éstos. Tienen la posibilidad de ser modificados, obteniendo un motor de corriente continua que ya no tendrá la capacidad de control, pero conservará el torque, velocidad y inercia que caracteriza a este servomotor.

2.4 Visual Basic Express 2008

Fig.2.3 Logo Visual Basic 2008

Es un entorno de desarrollo integrado para sistemas operativos Windows creado y distribuido por Microsoft Corporation. Soporta distintos lenguajes de programación como: Visual C#, Visual C++, ASP.NET, Visual J# y Visual Basic .NET, aunque en la actualidad se ha desarrollado y propuesto extensiones necesarias para muchos más. Es de origen gratuito y proporcionado por la compañía Microsoft Corporation orientándose a estudiantes, principiantes y aficionados.[3]

[3] http://es.wikipedia.org/wiki/Microsoft_Visual_Studio_Express

2.5 Microcontrolador

Fig.2.4 Esquema de un microcontrolador

El microcontrolador es un circuito programable, capaz de realizar órdenes grabadas en su memoria. Está formado de varios bloques funcionales, los cuales realizan una tarea específica. Un microcontrolador incluye en su interior tres unidades funcionales principales de una computadora: Memoria, periféricos de entrada/salida y unidad central de procesamiento.

Algunos microcontroladores usan palabras de 4 bits y operan a velocidad de reloj con frecuencias de 4 kHz, con un consumo mínimo de potencia (mW). Tiene la capacidad de mantenerse funcionando a la espera de un evento como una interrupción, el consumo de energía durante el sueño puede ser sólo nanovatios, lo que hace que sean adecuados para aplicaciones con batería de larga duración. Varios de estos microcontroladores pueden servir para cumplir roles de procesos críticos, donde sea necesario actuar más como un procesador digital de señal, con velocidades de reloj y de consumo de energía más elevados.

2.6 Batería de Litio

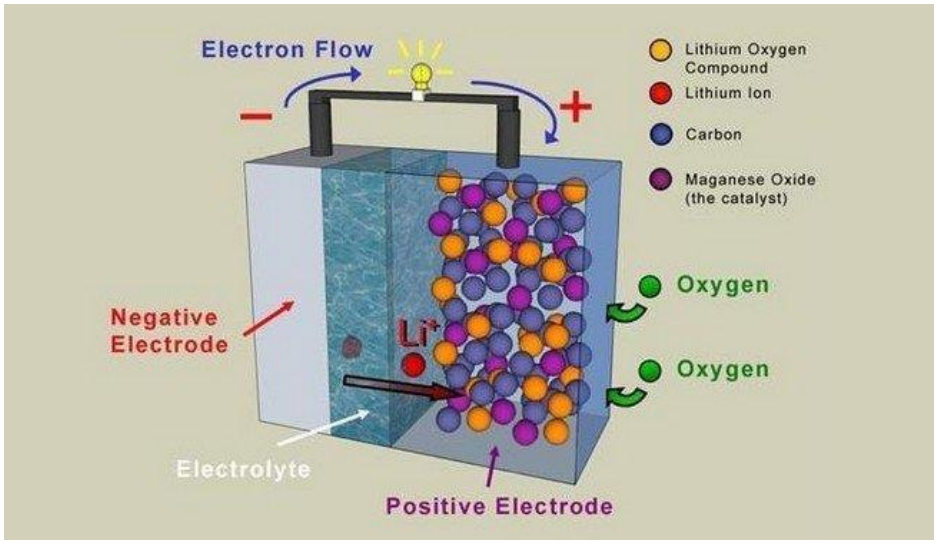

Fig.2.5 Componentes de una batería de litio[4]

Las baterías de ión litio se han convertido en aplicaciones muy populares de una nueva tecnología que se implanta rápidamente en los medios electrónicos de nuestro entorno doméstico y laboral. Ordenadores portátiles, teléfonos móviles, iPods, PDA, herramientas eléctricas se surten de un sistema de acumulación de energía más fiable, con mucha más capacidad. Las baterías de ión litio son el futuro y han llegado para quedarse con nosotros.

Las baterías de ión litio están relacionadas con una reducción del peso de los dispositivos por unidad. Son más ligeras que sus equivalentes de níquel cadmio o de níquel hidruro. Y es que los electrodos de las baterías de ión litio de este metal y de carbono son mucho más ligeros, además, el litio es un metal muy reactivo. Existe una gran cantidad de energía potencial almacenada en sus enlaces atómicos, mucha cantidad de energía en muy poco espacio.

[4] https://www.google.com.ec/search?q=bateria+de+litio

CAPÍTULO 3

ESTUDIO, DISEÑO E IMPLEMENTACIÓN DE UN PROTOTIPO DE ROBOT EXPLORADOR Y ENRUTADOR DE CABLE.

Introducción

En este capítulo se explica el estudio y diseño del sistema, lo que permite el montaje y la implementación del robot, con las explicaciones necesarias.

3.1 Estudio de las características técnicas de los elementos necesarios para la creación de un robot explorador para cableado estructurado.

3.1.1 Arduino Mega 2560

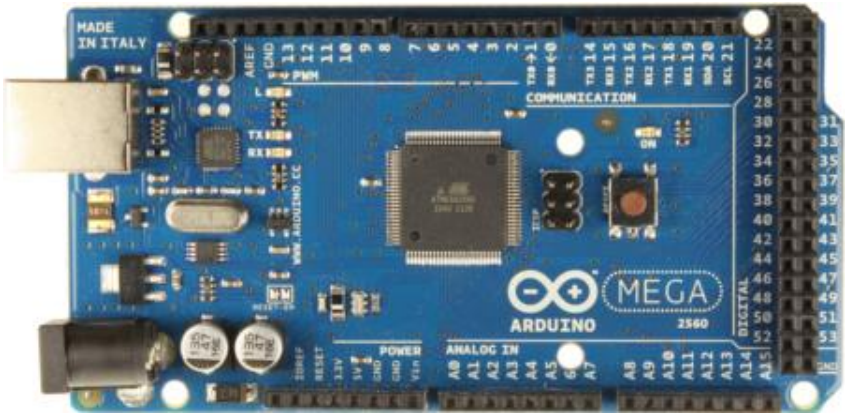

Fig.3.1 Esquema del Arduino Mega 2560

El Arduino Mega 2560 es una placa electrónica basada en el ATmega2560. Tiene 54 pines digitales de entrada / salida, 16 entradas analógicas, 4 UARTs (puertas seriales) , un oscilador de 16MHz , una conexión USB, un conector de alimentación , una cabecera ICSP , y un botón de reinicio .

Contiene lo necesario para apoyar a un microcontrolador, es suficiente con conectarlo a un ordenador (PC) con un cable (USB) o el conector de alimentación con un adaptador AC -DC o batería para empezar.

El Mega 2560 no utiliza un chip controlador USB - to -serial FTDI. En su lugar cuenta con el ATMEGA16U2 programado como convertidor USB a serie.

3.1.1.1 Power

El Arduino Mega puede ser alimentado a través de una conexión USB o con una fuente de alimentación externa. La fuente de alimentación se selecciona automáticamente.

La tarjeta puede funcionar mediante un suministro de 6 V a 20 V. Si la placa se abastece con menos de 7V operará, sin embargo, el pin referente a 5V puede dar menos del nominal y la juntura será inestable. Si se alimenta con más de 12V, el regulador se puede sobrecalentar y por ende dañar la placa. Por lo cual el rango recomendado es de 7 V a 12 V.

3.1.1.2 Comunicación

El Mega 2560 provee una serie de instalaciones para la comunicación con un ordenador, u otros microcontroladores. Proporciona cuatro UART hardware para (5V) de comunicación serie TTL. El software para desarrollo de Arduino tiene incluido un monitor que permite observar los datos de texto simple para ser enviados hacia y desde el tablero. El LED RX y TX de la placa parpadearán cuando los datos se transmiten a través del ATmega8U2/ATmega16U2. Una librería de la biblioteca del Software de Arduino permite la comunicación serial en cualquiera de los pines digitales del Arduino Mega 2560.

Las principales características de la tarjeta Arduino Mega2560 se detallan en la tabla 3.1:

Microcontrolador	ATmega2560
Voltaje de operación	5V
Input Voltage (recomendado)	7V-12V
Input Voltage (Limits)	6V-20V
Digital I/O Pins	54(15 provide PWM output)
Pins de Entradas Análogas	16
DC Corriente I/O Pin	40mA
DC Corriente para 3.3V Pin	50mA
Flash Memory	256 KB of which 8 KB used by bootloader
SRAM	8KB
EEPROM	4KB
Clock Speed	16 MHz

Tabla 3.1 Características eléctricas del Arduino Mega 2560

3.1.2 Servomotor Hitec HS-311

Fig.3.2 Servomotor Hitec HS-311[5]

El servomotor HS-311 es el ideal para aplicaciones de bajo costo, sin embargo, este servo ofrece muchos de los mismos aspectos de servos más costosos. Una pieza de placa de circuitos y engranajes de estrechos garantiza una gran durabilidad. Viene con una serie de algunos cuernos servos y brazos con

[5] https://www.google.com.ec/search?q=servomotor+hitec+hs-311&source=lnms&

accesorios de armaje. [6]

Las características de este servomotor se muestran en la tabla 3.2:

Sistema de Control	+Ancho de Pulso de control 1500usec Neutral
Pulso requerido	3V-5 V pico-pico de onda cuadrada
Tensión de funcionamiento	4.8V a 6 V
Velocidad de funcionamiento(4.8V)	0.19sec/60°
Torsión de parada	42 oz / in(3.0 Kg/cm)
Consumo de corriente	7.4mA/idle, 160 mA sin carga de funcionamiento
Angulo de funcionamiento	45° un impulso lateral
Dirección	Multi-direccional

Tabla.3.2 Características eléctricas del HS-311

En el diagrama de la figura 3.3 se detalla las dimensiones de un servomotor HS-311:

[6] http://www.servocity.com/html/hs-311_standard.html

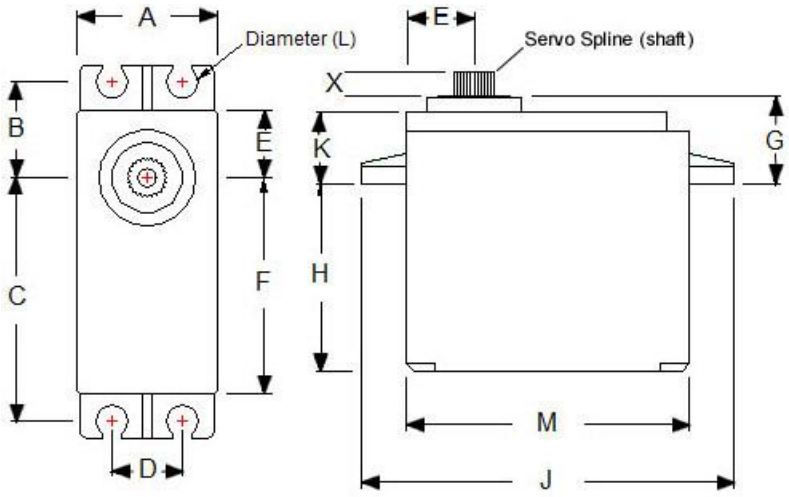

Fig.3.3 Dimensiones del servomotor HS-311[7]

A = 0.780 "(19.82mm)

B = 0.530 "(13.47mm)

C = 1.33 "(33.79mm)

D = 0.400 "(10.17mm)

E = 0.380 "(9.66mm)

F = 1.19 "(30.22 mm)

T = 0.460 "(11.68 mm)

H = 1.05 "(26.67 mm)

J = 2.08 "(52.84mm)

K = 0,368 "(9,35 mm)

L = 0.172 "(4.38mm)

M = 1.57 "(39.88 mm)

X = 0.120 "(3.05 mm)

[7] http://www.servocity.com/html/hs-311_standard.html

3.1.3 Módulo Xbee (Serie 1)

Fig.3.4 Módulo Xbee Serie 1[8]

Zigbee es un protocolo de comunicaciones inalámbrico basado en el estándar de comunicaciones para redes inalámbricas IEEE_802.15.4. Creado por Zigbee Alliance, una organización, teóricamente sin ánimo de lucro. Zigbee admite que dispositivos electrónicos de niveles mínimos de consumo de energía, realicen comunicaciones inalámbricas. Es especialmente útil para redes de sensores en entornos industriales, médicos y, sobre todo, domóticos.

Las comunicaciones que se realizan mediante Zigbee usan la banda libre de 2.4GHz. La comunicación se efectúa a través de una única frecuencia, es decir, de un canal. Normalmente se puede elegir un canal de entre 16 existentes. El alcance depende de la potencia de transmisión del dispositivo así como también del tipo de antenas utilizadas (cerámicas, dipolos, etc.). El alcance standard con antena dipolo en línea de vista es de aproximadamente de 100m y en interiores unos 30m. La velocidad de transmisión de datos de una red diseñada con Zigbee es de hasta 256kbps. Las redes de comunicación Zigbee se pueden formar, teóricamente, hasta con un número valorado de 65535 dispositivos.

[8] https://www.google.com.ec/search?q=xbee+s1

Entre las necesidades principales que satisface el módulo Xbee se encuentran:

- Bajo costo.
- Ultra-bajo consumo de potencia.
- Uso de banda de radio libre y sin licencias.
- Instalación barata y simple.
- Redes extensibles y flexibles.

Al usar el protocolo Zigbee se reemplaza un cable por una comunicación serial inalámbrica, hasta el desarrollo de configuraciones punto a punto, multipunto, peer to peer (nodos conectados entre sí) o redes complejas de sensores. Una conexión típica se muestra en la Figura 3.5, aquí se observa que cada módulo Xbee posee algún tipo de sensor, los cuales entregan los datos para ser enviados usando la red Zigbee a un Centro que administre la información.[9]

Fig.3.5 Conexión Típica Usando Xbee

[9] www.olimex.cl

3.1.4 Android

Android se define como un sistema operativo basado en Linux creado principalmente para dispositivos móviles con pantalla táctil, teléfonos inteligentes y tabletas, inicialmente desarrollado por Android, Inc. Google aporto económicamente y más tarde compró esta empresa en 2005.

El primer móvil con el Sistema Operativo Android fue el HTC Dream y estuvo a la venta en octubre de 2008.[10]

Fig.3.6 Interfaz Android

3.1.5 Batería de Litio (Zippy)

Las baterías Zippy Flightmax proporcionan una plena capacidad de almacenaje de energía y un nivel óptimo de descarga, además de ser las mejores baterías valoradas en el mercado.

[10] http://es.wikipedia.org/wiki/Android

Fig.3.7 Batería de Litio Zippy

En tabla 3.3 se especifica las características principales de esta batería:

Discharge Plug:	Bullet Connector
Capacity:	4000mAh
Voltage:	3S1P / 3 Cell / 11.1v
Discharge:	20C Constant / 30C Burst
Weight:	306g (including wire, plug)
Dimensions:	146x51x22mm
Balance Plug:	JST-XH

Tabla 3.3 Características Batería Zippy[11]

[11] http://www.hobbyking.com/hobbyking/store/uh_viewitem.asp?idproduct=7634

3.2 Diseño del prototipo de robot explorador para enrutar cable en tumbados de difícil acceso.

En la Fig.3.8 se muestra la propuesta de diseño del robot explorador mediante el siguiente diagrama de bloques:

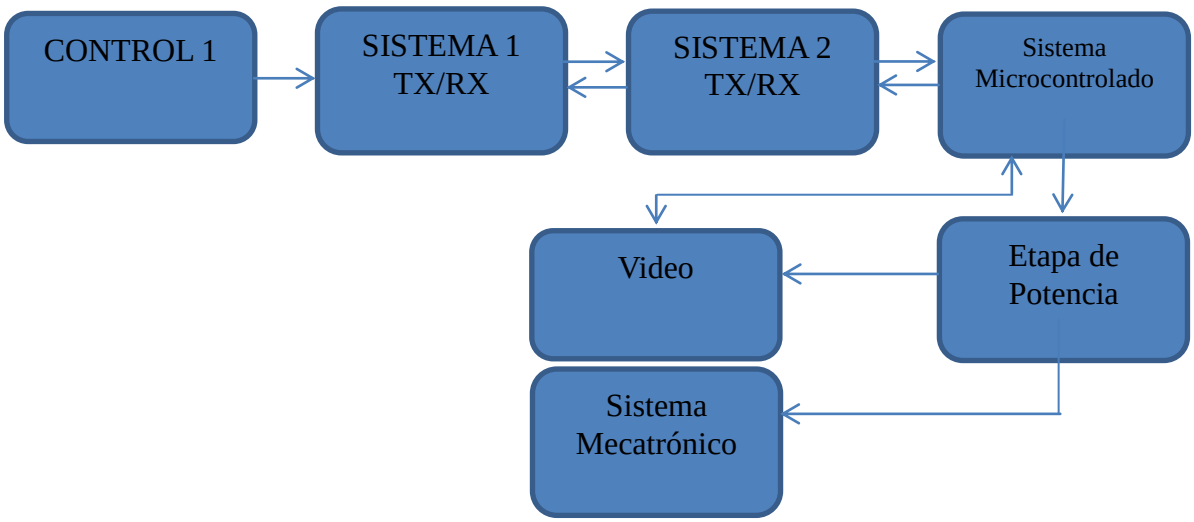

Fig. 3.8 Diagrama de Bloques.

A continuación se describe el funcionamiento del sistema bloque por bloque según el diagrama de bloques de la Fig. 3.8.

3.2.1 Control 1

En este primer bloque del sistema define la interfaz del usuario que ha sido diseñada en el software Visual Basic express 2008, consta de los botones para opción de recurso con el que cuenta el sistema tales como: Direccionamiento de locomoción del robot (adelante-atrás-izquierda-derecha), Giro de rotación de la cámara (horario-anti horario).

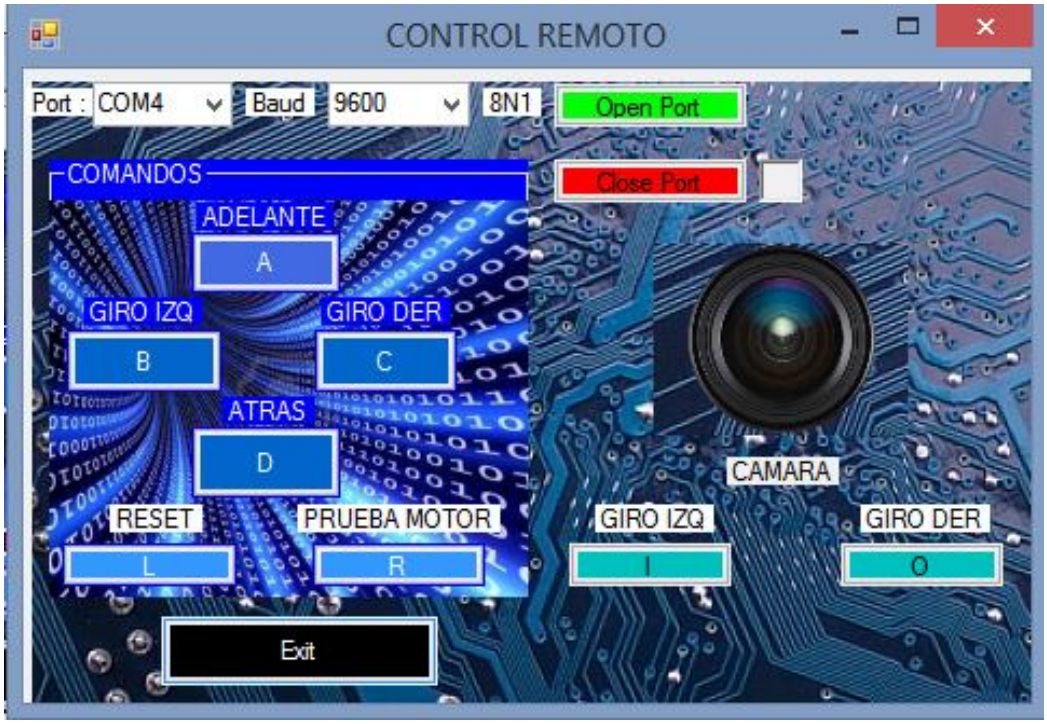

Fig.3.9 Interfaz de usuario en Visual Basic Express 2008.

Dentro de este bloque también se cuenta con la interfaz para el envió de video y control de luz led, mediante el software diseñado por android llamado (IP Webcam). El programa IP Webcam convierte a un dispositivo con sistema operativo “Android” en una cámara capaz de retransmitir las imágenes y video que capta, a través de su Wifi para, de esta forma, poder ver dichas imágenes y video en una pc notebook, etc.

A continuación se detalla los pasos para la instalación de este software y su funcionamiento.

Paso 1. Una vez descargado este software en el dispositivo, se inicia el programa y nos va a abrir toda la configuración, no se presionara nada, se selecciona el botón que dice "STAR SERVER" y automáticamente se abrirá la cámara.

Fig. 3.10 Interfaz IP Webcam

Paso 2. La cámara de dispositivo ya está funcionando a través del programa; en la pantalla se observan 2 botones y una dirección web que dice http://?.?.?.?:?. Se presiona en la opción "HOW DO I CONNECT?" y se selecciona "I´M USING WI-FI ROUTER" como se muestra en la Fig 3.10.

Fig.3.11 Interfaz de conexión IP Webcam

Paso 3. En el buscador de la Pc se escribe la dirección que se muestra en la pantalla del dispositivo "HTTP:// ?.?.?.?:?".

Paso 4. Se presiona la opción "USE JAVA BROWSER PLUGIN" y se enlaza con el dispositivo android por medio de la red Wifi enviando el streaming video.

Fig.3.12 Pagina Web / Servidor Android

Paso 5. En este paso se escoge la opción de panel de control remoto, como se muestra en la Fig.3.5, para poder controlar el led del dispositivo android utilizado. Los parámetros que se muestran son encender y apagar led, en la Fig.3.6 se muestra la interfaz del control.

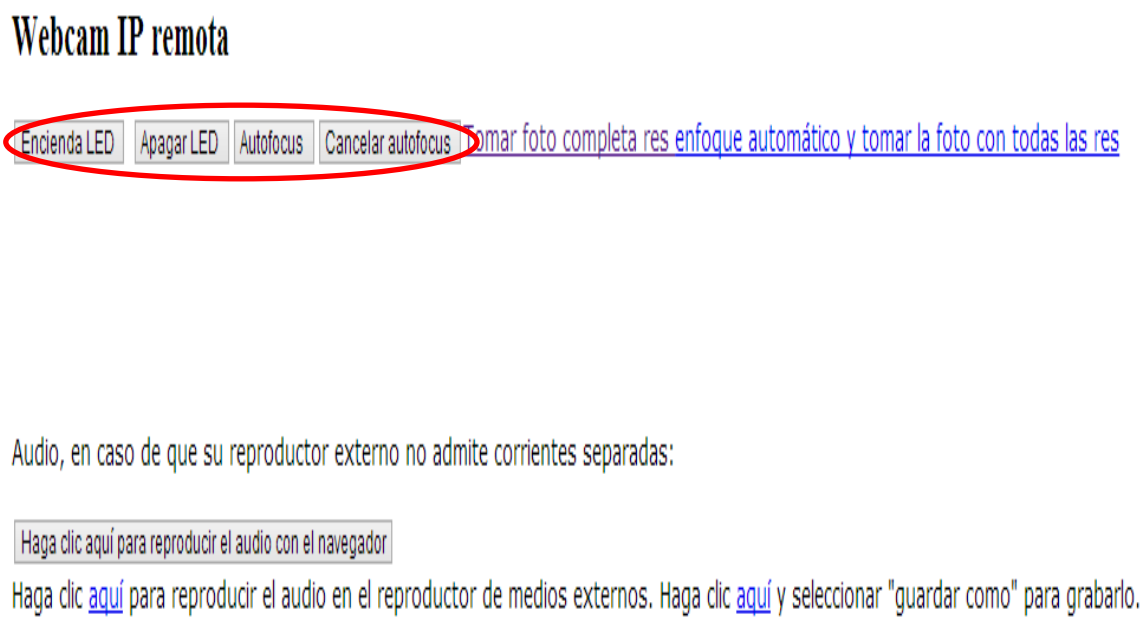

Fig.3.13 Interfaz de Control Remoto

3.2.2 Sistema de Transmisión 1 (Tx)

En este Bloque se desarrolla la comunicación serial que se emplea para enviar los datos al dispositivo inalámbrico (Xbee) por medio del software Visual Basic Express 2008, utilizando un puerto COM de la Pc que en este caso es virtual ya que el Xbee S1 utilizado no tiene un puerto serial, sino un puerto USB para comunicarse, por lo cual se ha instalado un driver FTDI para que se pueda establecer la conexión.

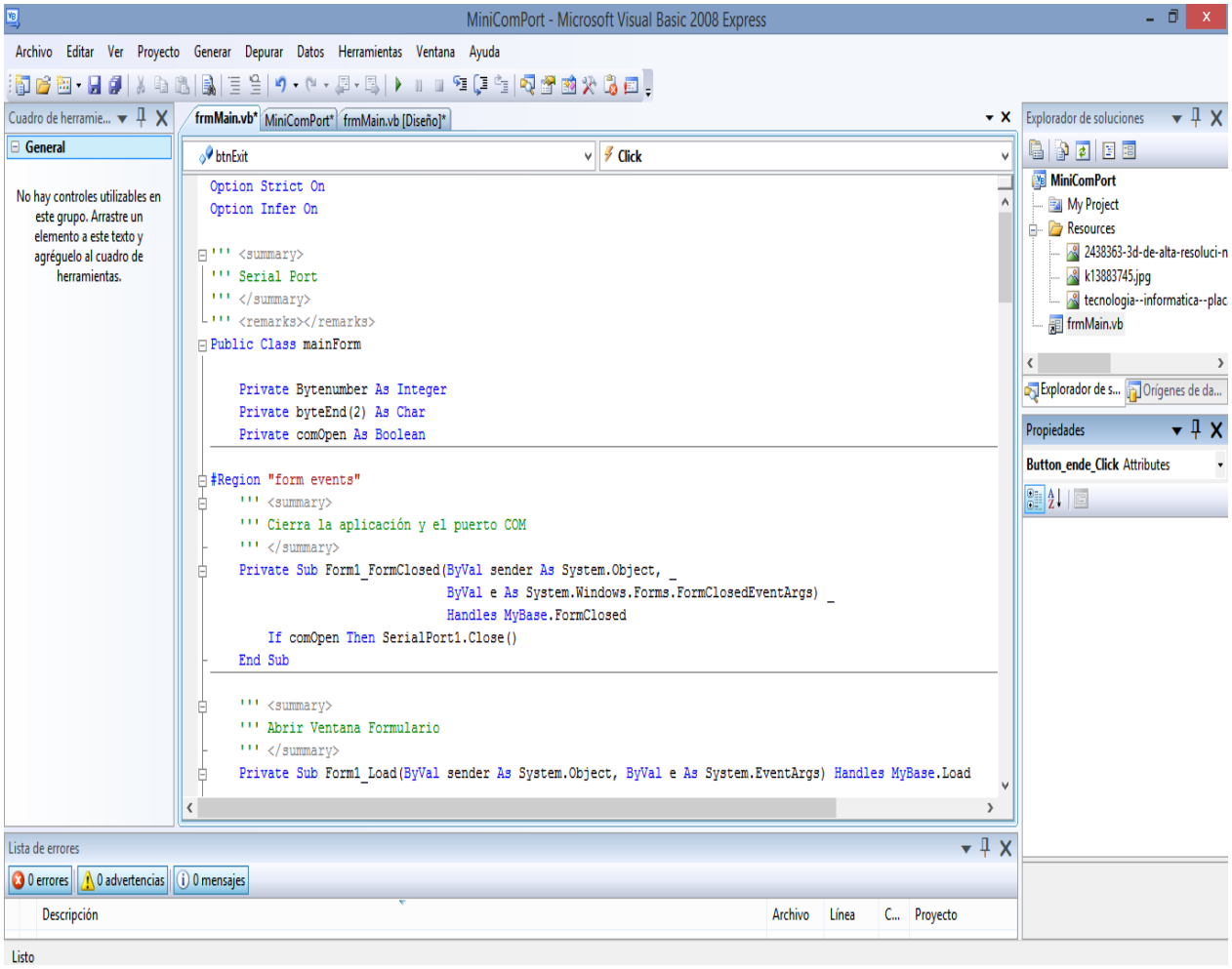

Fig.3.14 Pantalla de programación en VB

Para la programación de los módulos inalámbricos se utilizó el software X-CTU creado por la empresa DIGI, aquí se establecen los parámetros de la red a la que pertenecen los módulos.

En las figuras 3.14 y 3.15 se detallan las características para la configuración del X-CTU.

- **PC Settings**

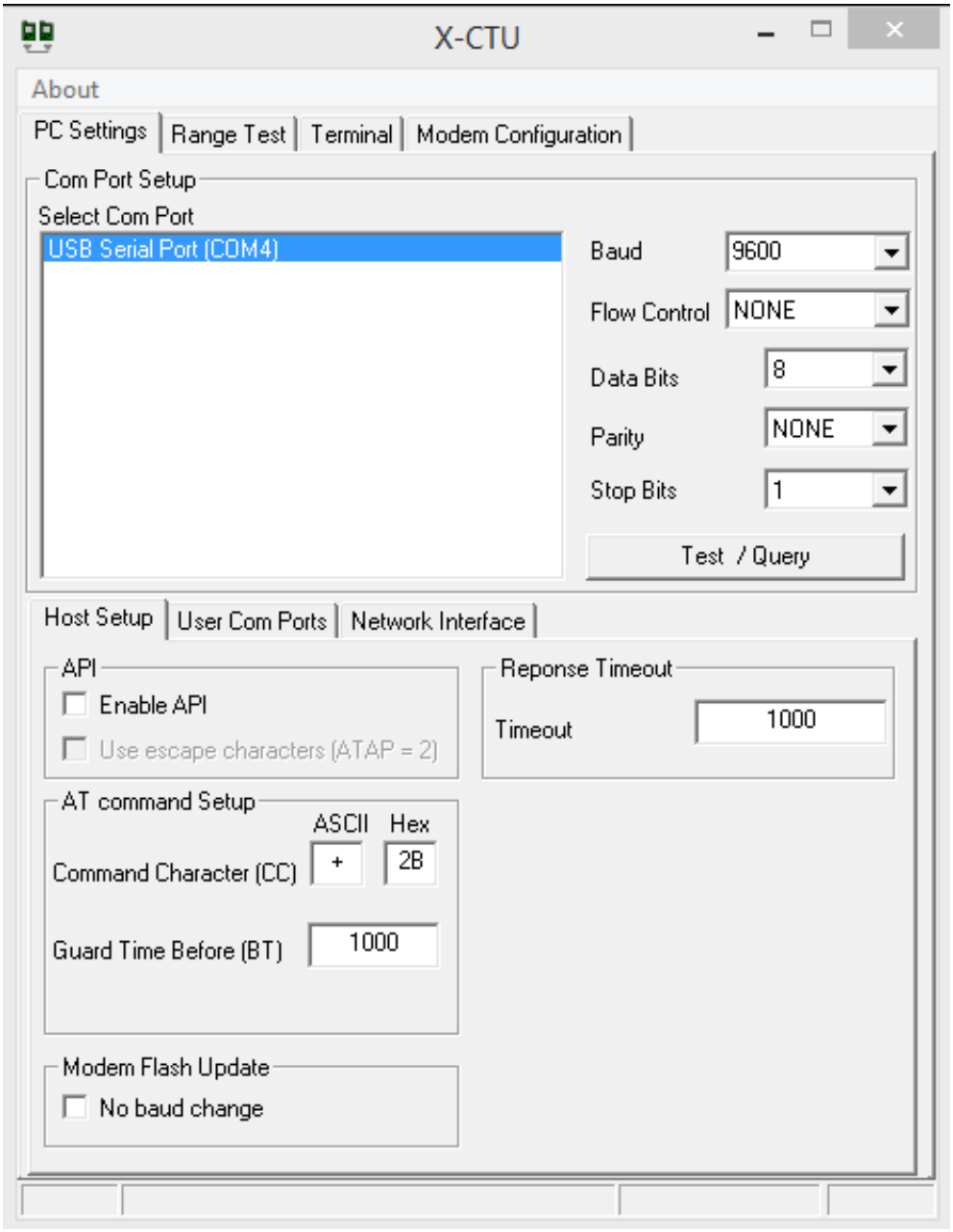

Fig.3.15 Reconocimiento de puerto COM en X-CTU.

- **Modem Configuration**

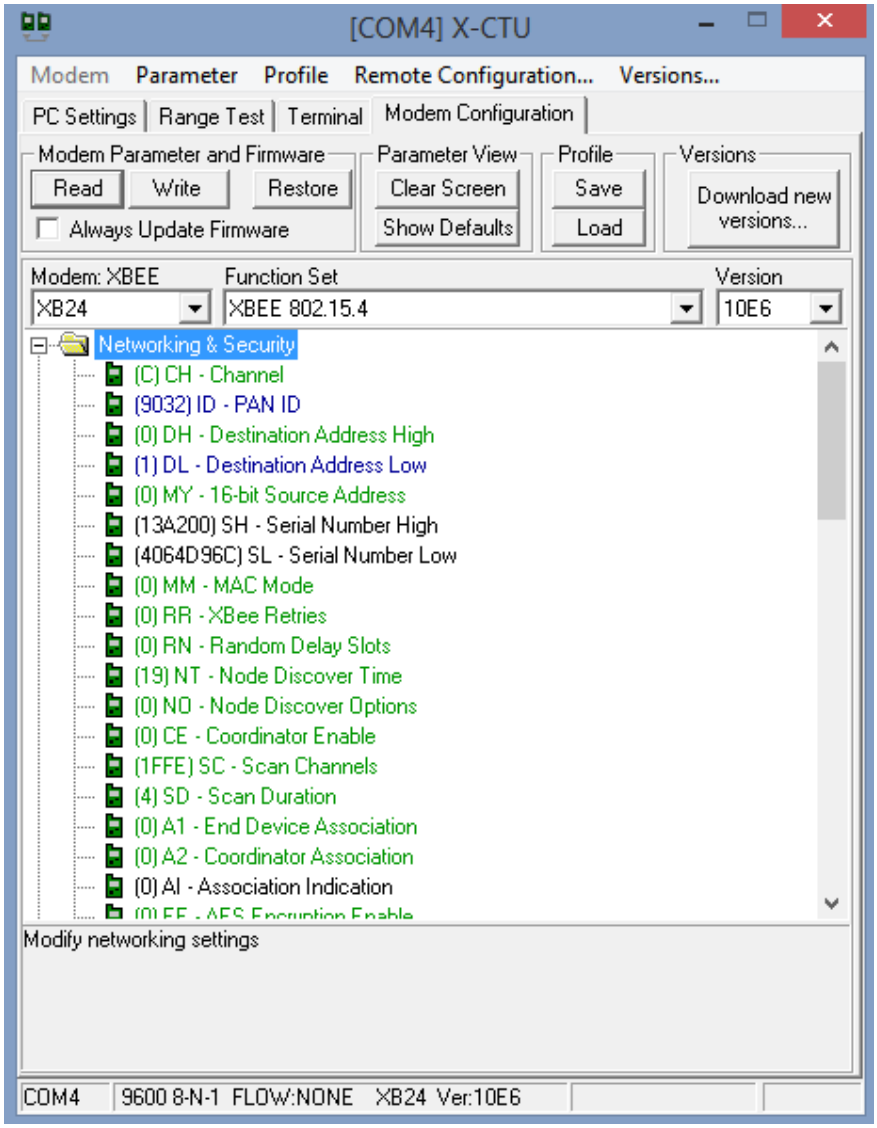

Fig.3.16 Parámetros de configuración del X-CTU

3.2.3 Sistema de Transmisión (Rx)

Aquí se encuentra el módulo de Recepción (Xbee S1) el cual se acopla a un shield para Arduino.

El XBee shield para Arduino permite comunicar al Arduino Mega 2560 de forma inalámbrica usando la plataforma Zigbee, que fue desarrollado por Libelium.

Fig.3.17 Xbee Shield para Arduino Mega 2560

3.2.4 Sistema Microcontrolado

En esta etapa se describe el funcionamiento del Arduino Mega 2560 utilizando el software descargado de forma gratuita desde la página oficial de Arduino[12],mediante la programación en lenguaje C++ se definieron todos los parámetros de los cuales va a estar provisto el robot.

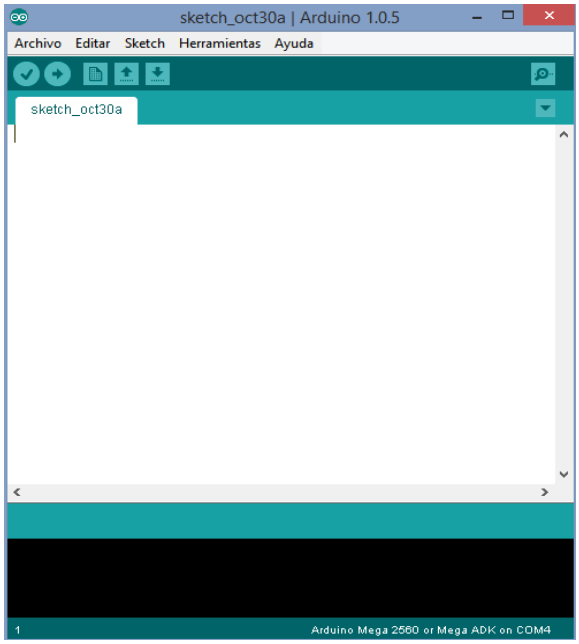

Fig.3.18 Interfaz de software de Arduino

[12] http://www.arduino.cc/es/

3.2.4 Diseño de Hardware

En los diagramas de las figuras se muestra como se ha distribuido los pines en el Arduino Mega2560 y el Xbee S1.

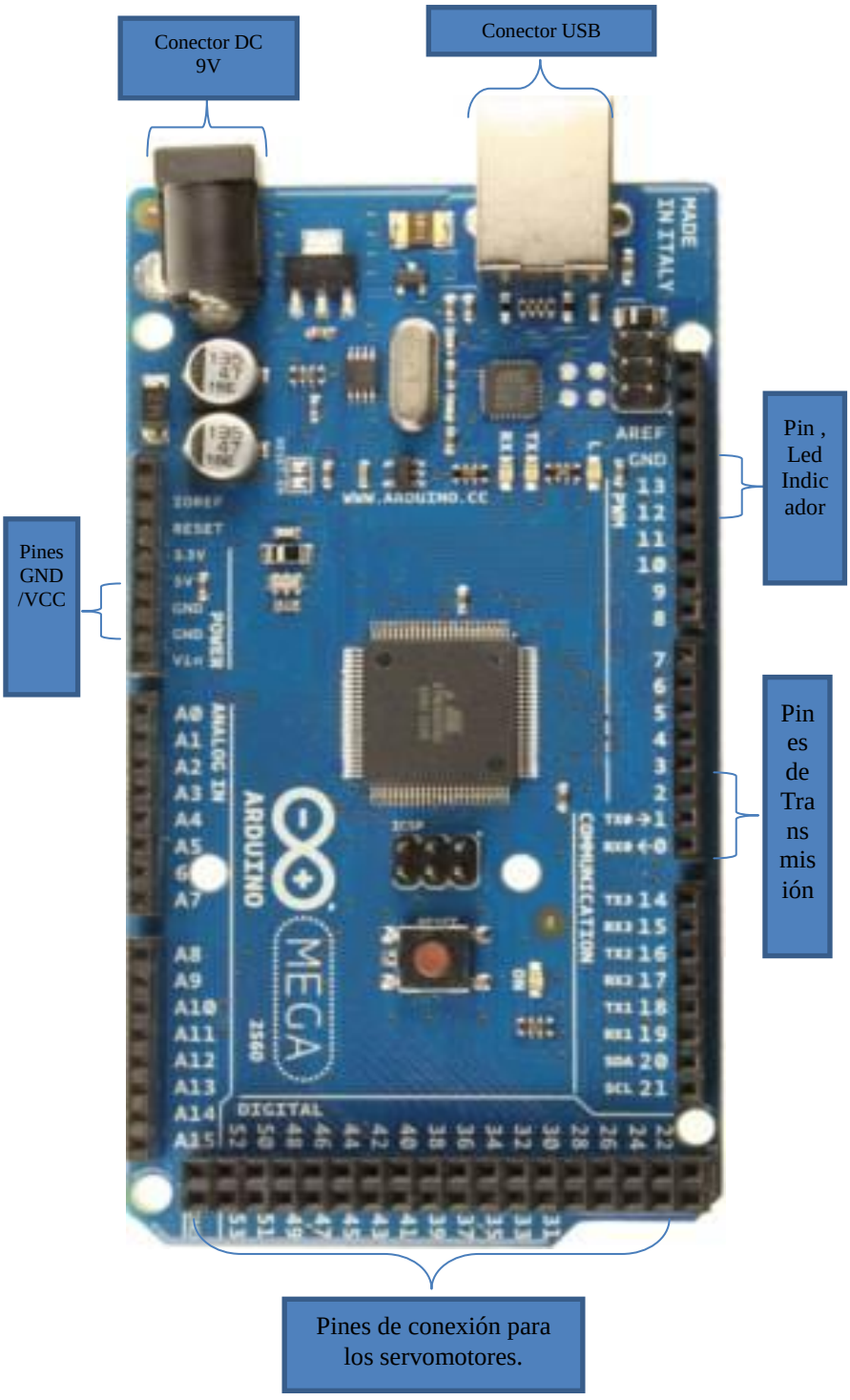

Fig.3.19 Diagrama circuital del Arduino Mega 2560

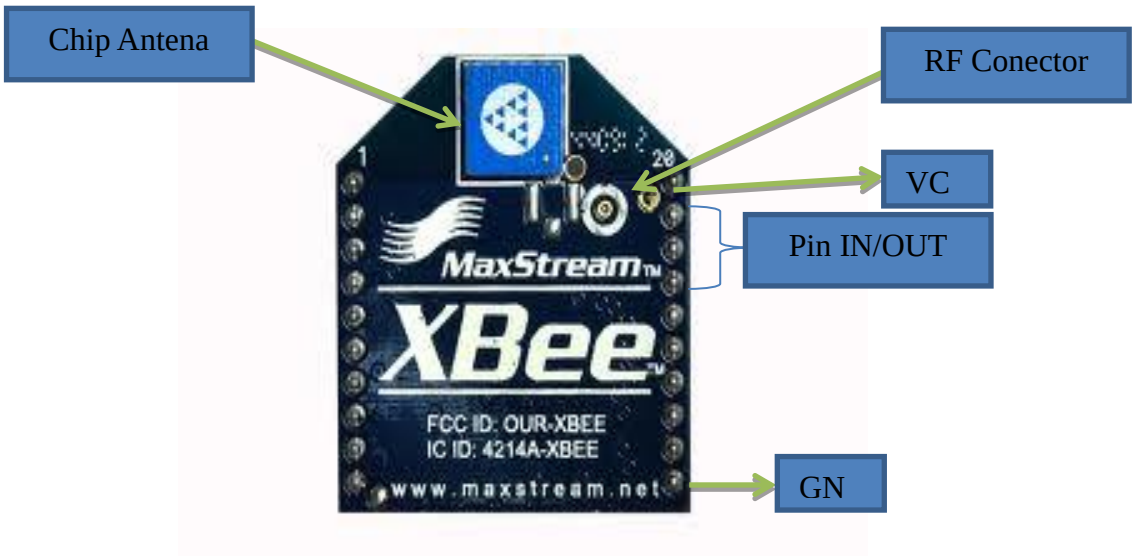

Fig.3.20 Diagrama circuital del Xbee

3.2.5 Diseño de Software

El diseño de software se muestra mediante el siguiente diagrama de flujo (ver Fig.3.20), donde se explica detalladamente el funcionamiento que tendrá el programa y el control remoto, en base a este diagrama se procederá a crear cada función de programa que facilitará el funcionamiento del sistema completo por etapas.

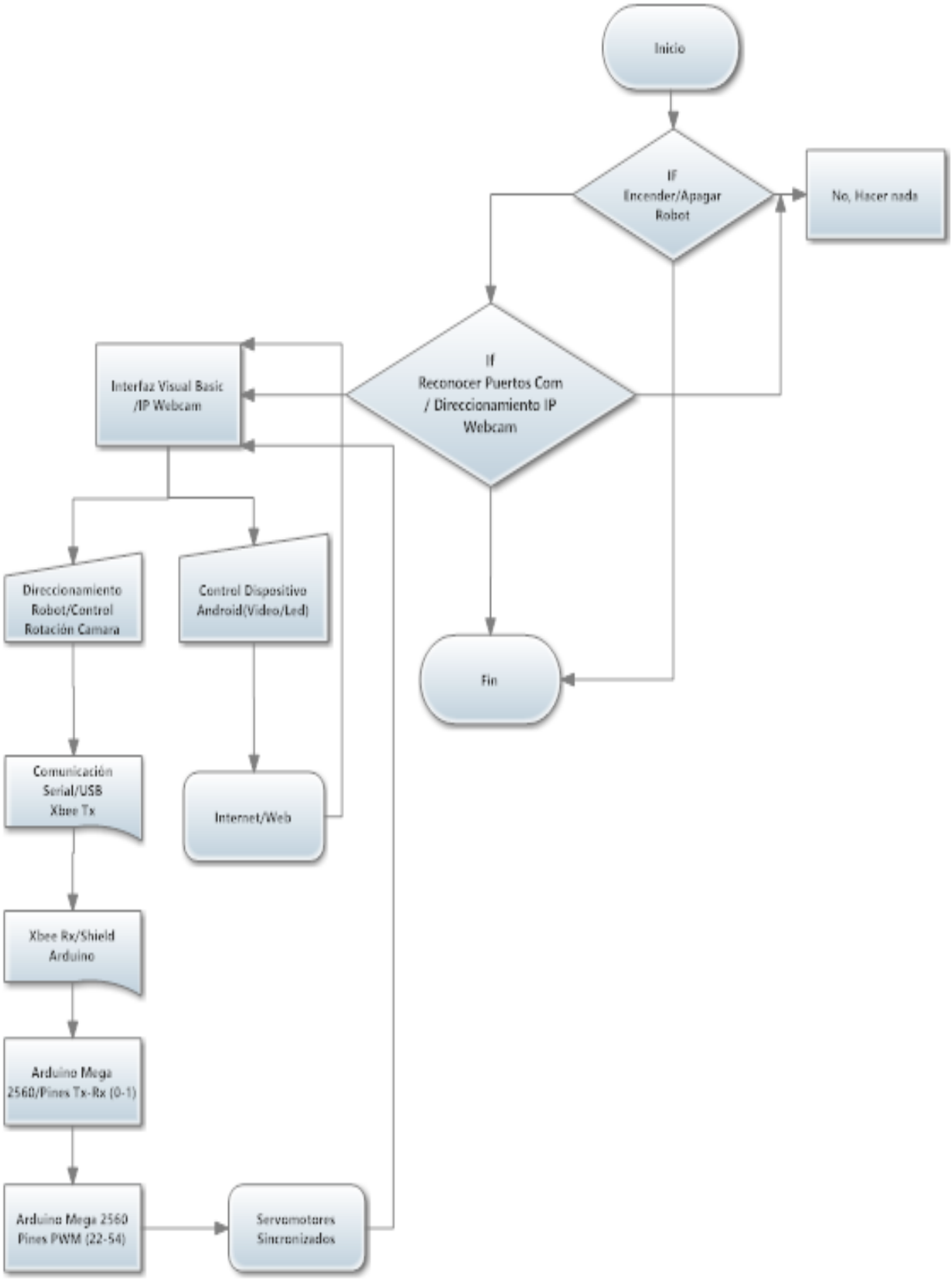

Fig.3.21 Diagrama de Flujo de funcionamiento general del sistema.

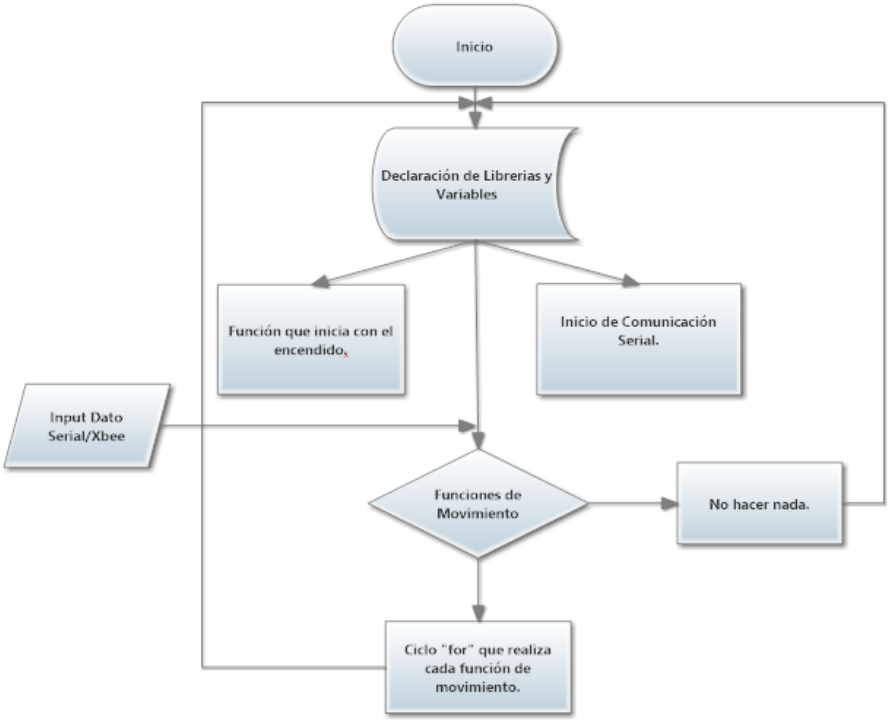

Fig.3.22 Diagrama de Flujo de funcionamiento de la Plataforma Arduino.

3.2.6 Diseño Parte Mecánica

En las Figuras 3.23, 3.24 y 3.25, se muestran las piezas diseñadas para el soporte del robot, tanto para la base superior e inferior del mismo y el soporte individual para cada extensión del armazón.

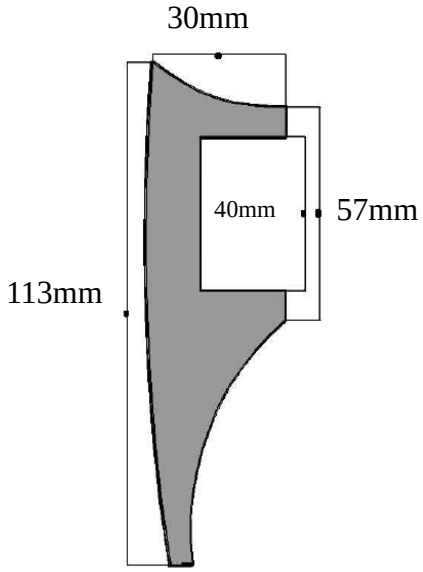

Fig.3.23 Pieza de Soporte de Robot

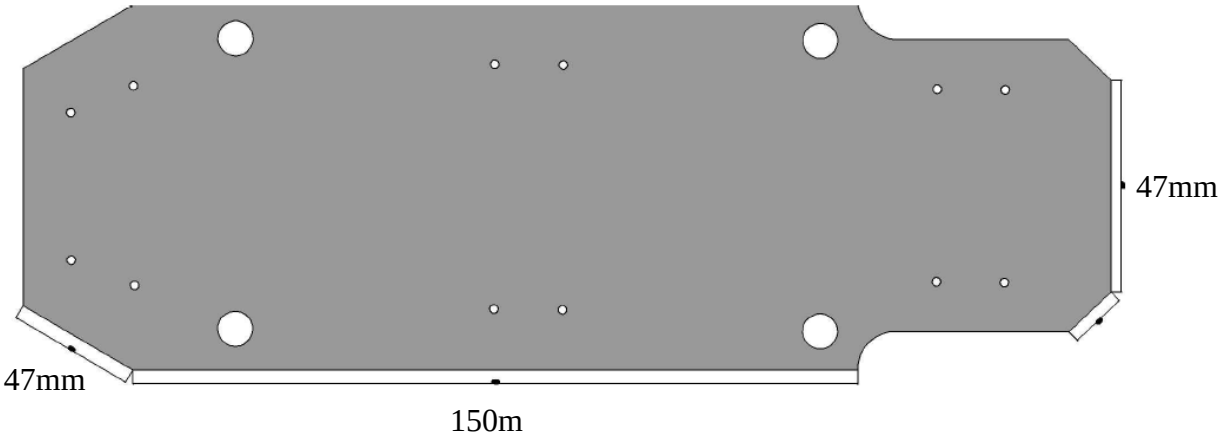

Fig.3.24 Piezas Inferior de ensamble

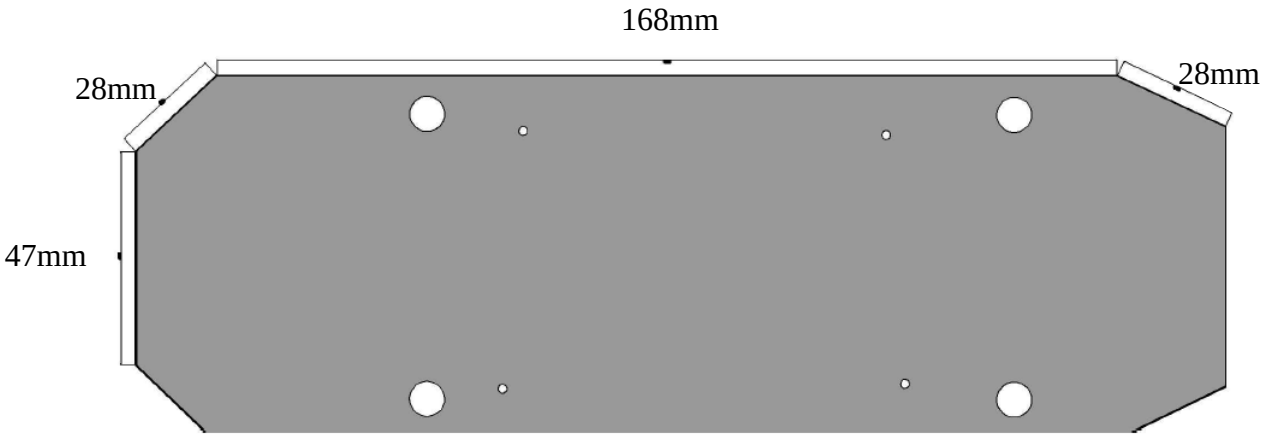

Fig.3.25 Piezas Superior de ensamble

3.3 Montaje del Sistema

3.3.1 Montaje de Hardware

A continuación se muestran los módulos Xbee S1 los cuales se acoplan a los shield para poder establecer la comunicación entre la Pc y el robot.

Fig.3.26 Vista superior Xbee/Shield para Arduino

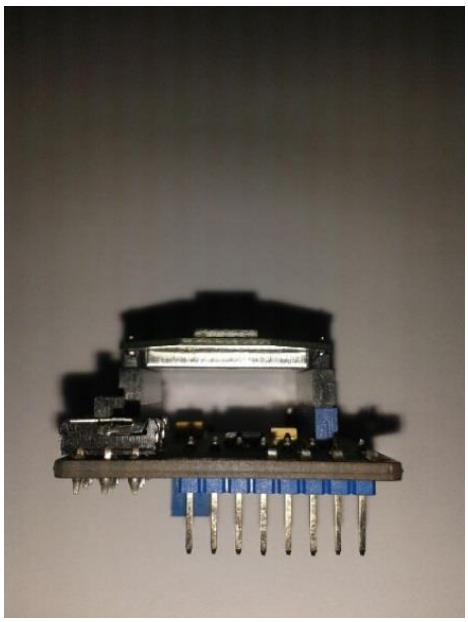

Fig.3.27 Vista frontal de Xbee/Shield para Arduino

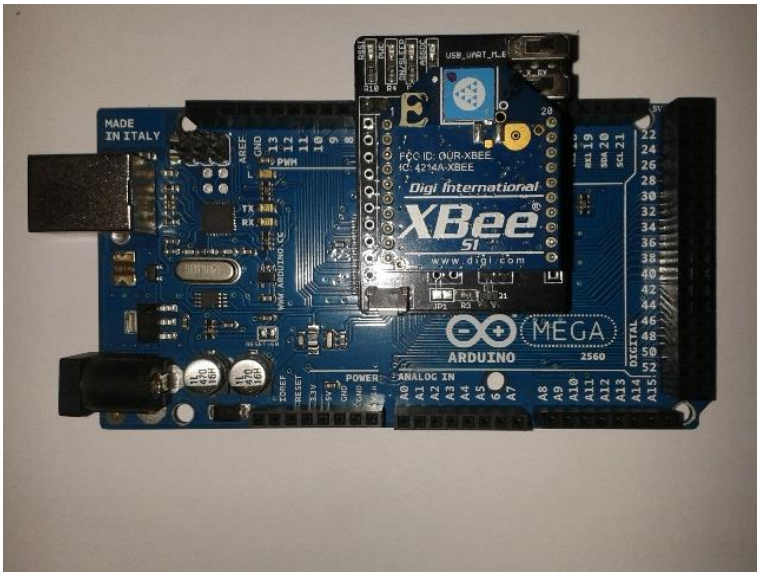

Fig.3.28 Ensamble de Xbee/Shield en Placa Arduino Mega 2560

En la Fig.3.29 se observa cómo se acopla la placa de Arduino Mega 2560 y el módulo Xbee/Rx en la base superior del robot.

Fig.3.29 Montaje de placa Arduino Mega 2560 en pieza superior.

Fig.3.30 Pieza de soporte para dispositivo android

3.3.2 Montaje de Software

- **Plataforma Arduino**

El software utilizado para programar el sistema robótico es la plataforma de Arduino, a continuación se especifica cada parte del programa:

- Inicio del Programa

```
//--------------------------------------------------
//Declaración de puertos de entradas, salidas y variables
//--------------------------------------------------
#include <Servo.h>   //inclusión de librerías para servomotores (PWM)
int led = 13;                //Pin donde se encuentra el LED, salida
char leer;                //Variable donde se almacena la letra
Servo servo1;
Servo servo2;
Servo servo3;
Servo servo4;
Servo servo5;
Servo servo6;
```

```
Servo servo7;
Servo servo8;
Servo servo9;
Servo servo10;
Servo servo11;
Servo servo12;
Servo servo13;
Servo servo14;
Servo servo15;
Servo servo16;
Servo servo17;
Servo servo18;
Servo camservo;
```

- Se asignan los pines de la placa Arduino Mega para cada servo motor y para las salidas y entradas.

```
//-------------------------------------
//Función principal
//-------------------------------------
void setup()                        // Se ejecuta cada vez que el Arduino se inicia
{
  Serial.begin(9600);       //Inicia comunicación serial
  pinMode(led, OUTPUT);  //Configurar el LED como una salida
  servo1.attach(22);
  servo2.attach(24);
  servo3.attach(26);
  servo4.attach(28);
  servo5.attach(30);
  servo6.attach(32);
```

```
    servo7.attach(34);
    servo8.attach(36);
    servo9.attach(38);
    servo10.attach(40);
    servo11.attach(42);
    servo12.attach(44);
    servo13.attach(46);
    servo14.attach(48);
    servo15.attach(50);
    servo16.attach(52);
    servo17.attach(51);
    servo18.attach(53);
    servo19.attach(55);
```

- En esta etapa se declara la función cíclica que se repite en todos los procesos; se crean las funciones para cada interacción del robot y también se escriben los pulsos en grados centígrados gracias a las librerías para servomotores.

```
void loop()                    // Esta función se mantiene ejecutando
{                              //  cuando la placa Arduino se energiza
   leer=Serial.read();         //Lee los datos recibidos por el módulo Xbee
   if (leer=='L' )
{
    resetFunction();
 }
   if (leer=='A' )
  {
    adelFunction();
  }
```

```
   if (leer=='D' )
{
   atrasFunction();
}
   if (leer=='B' )
{
    ghFunction();
}
    if (leer=='C' )
{
   gaFunction();
}
   if (leer=='R' )
{
   pruebaFunction();
}
   if (leer=='I' )
{
   gcizqFunction();
}
   if (leer=='O' )
{
    gcderFunction();
}
}
```

- A continuación se detalla los ciclos de las funciones de cada movimiento.

```
   int resetFunction(){
```

```
digitalWrite(led, HIGH);
camservo.write(90);
servo2.write(60);
servo3.write(100);
servo5.write(60);
servo6.write(100);
servo8.write(120);
servo9.write(80);
delay(500);
servo1.write(90);
servo4.write(90);
servo7.write(90);
delay(500);
servo2.write(90);
servo3.write(90);
servo5.write(90);
servo6.write(90);
servo8.write(90);
servo9.write(90);
delay(500);
servo11.write(60);
servo12.write(100);
servo14.write(120);
servo15.write(80);
servo17.write(120);
servo18.write(80);
delay(500);
servo10.write(90);
servo13.write(90);
servo16.write(90);
```

```
delay(500);
servo11.write(90);
servo12.write(90);
servo14.write(90);
servo15.write(90);
servo17.write(90);
servo18.write(90);
delay(500);
 return(1);
 }
 int adelFunction(){
  digitalWrite(led, HIGH);
servo1.write(90);
servo2.write(60);
servo3.write(100);
servo4.write(90);
servo5.write(60);
servo6.write(100);
servo7.write(90);
servo8.write(120);
servo9.write(80);
servo10.write(90);
servo11.write(90);
servo12.write(90);
servo13.write(90);
servo14.write(90);
servo15.write(90);
servo16.write(90);
servo17.write(90);
servo18.write(90);
```

```
 delay(200);
servo1.write(70);
servo2.write(60);
servo3.write(100);
servo4.write(70);
servo5.write(60);
servo6.write(100);
servo7.write(110);
servo8.write(120);
servo9.write(80);
servo10.write(90);
servo11.write(90);
servo12.write(90);
servo13.write(90);
servo14.write(90);
servo15.write(90);
servo16.write(90);
servo17.write(90);
servo18.write(90);
delay(200);
servo1.write(70);
servo2.write(90);
servo3.write(90);
servo4.write(70);
servo5.write(90);
servo6.write(90);
servo7.write(110);
servo8.write(90);
servo9.write(90);
servo10.write(90);
```

```
servo11.write(90);
servo12.write(90);
servo13.write(90);
servo14.write(90);
servo15.write(90);
servo16.write(90);
servo17.write(90);
servo18.write(90);
delay(200);
servo1.write(70);
servo2.write(90);
servo3.write(90);
servo4.write(70);
servo5.write(90);
servo6.write(90);
servo7.write(110);
servo8.write(90);
servo9.write(90);
servo10.write(90);
servo11.write(60);
servo12.write(100);
servo13.write(90);
servo14.write(120);
servo15.write(80);
servo16.write(90);
servo17.write(120);
servo18.write(80);
delay(200);
servo1.write(110);
servo2.write(90);
```

```
servo3.write(90);
servo4.write(110);
servo5.write(90);
servo6.write(90);
servo7.write(70);
servo8.write(90);
servo9.write(90);
servo10.write(70);
servo11.write(60);
servo12.write(100);
servo13.write(110);
servo14.write(120);
servo15.write(80);
servo16.write(110);
servo17.write(120);
servo18.write(80);
delay(200);
servo1.write(110);
servo2.write(90);
servo3.write(90);
servo4.write(110);
servo5.write(90);
servo6.write(90);
servo7.write(70);
servo8.write(90);
servo9.write(90);
servo10.write(70);
servo11.write(90);
servo12.write(90);
servo13.write(110);
```

```
servo14.write(90);
servo15.write(90);
servo16.write(110);
servo17.write(90);
servo18.write(90);
 delay(200);
 return(1);
}
int atrasFunction() {
digitalWrite(led, HIGH);
servo1.write(90);
servo2.write(90);
servo3.write(90);
servo4.write(90);
servo5.write(90);
servo6.write(90);
servo7.write(90);
servo8.write(90);
servo9.write(90);
servo10.write(90);
servo11.write(60);
servo12.write(100);
servo13.write(90);
servo14.write(120);
servo15.write(80);
servo16.write(90);
servo17.write(120);
servo18.write(80);
delay(200);
servo1.write(90);
```

```
servo2.write(90);
servo3.write(90);
servo4.write(90);
servo5.write(90);
servo6.write(90);
servo7.write(90);
servo8.write(90);
servo9.write(90);
servo10.write(110);
servo11.write(60);
servo12.write(100);
servo13.write(70);
servo14.write(120);
servo15.write(80);
servo16.write(70);
servo17.write(120);
servo18.write(80);
delay(200);
servo1.write(90);
servo2.write(90);
servo3.write(90);
servo4.write(90);
servo5.write(90);
servo6.write(90);
servo7.write(90);
servo8.write(90);
servo9.write(90);
servo10.write(110);
servo11.write(90);
servo12.write(90);
```

```
servo13.write(70);
servo14.write(90);
servo15.write(90);
servo16.write(70);
servo17.write(90);
servo18.write(90);
 delay(200);
servo1.write(90);
servo2.write(60);
servo3.write(100);
servo4.write(90);
servo5.write(60);
servo6.write(100);
servo7.write(90);
servo8.write(120);
servo9.write(80);
servo10.write(110);
servo11.write(90);
servo12.write(90);
servo13.write(70);
servo14.write(90);
servo15.write(90);
servo16.write(70);
servo17.write(90);
servo18.write(90);
 delay(200);
servo1.write(110);
servo2.write(60);
servo3.write(100);
servo4.write(110);
```

```
servo5.write(60);
servo6.write(100);
servo7.write(70);
servo8.write(120);
servo9.write(80);
servo10.write(70);
servo11.write(90);
servo12.write(90);
servo13.write(110);
servo14.write(90);
servo15.write(90);
servo16.write(110);
servo17.write(90);
servo18.write(90);
delay(200);
servo1.write(110);
servo2.write(90);
servo3.write(90);
servo4.write(110);
servo5.write(90);
servo6.write(90);
servo7.write(70);
servo8.write(90);
servo9.write(90);
servo10.write(70);
servo11.write(90);
servo12.write(90);
servo13.write(110);
servo14.write(90);
servo15.write(90);
```

```
servo16.write(110);
servo17.write(90);
servo18.write(90);
delay(200);
 return(1);
}
int ghFunction()
{
digitalWrite(led, HIGH);
servo1.write(90);
servo2.write(60);
servo3.write(100);
servo4.write(90);
servo5.write(60);
servo6.write(100);
servo7.write(90);
servo8.write(120);
servo9.write(80);
servo10.write(90);
servo11.write(90);
servo12.write(90);
servo13.write(90);
servo14.write(90);
servo15.write(90);
servo16.write(90);
servo17.write(90);
servo18.write(90);
 delay(500);
servo1.write(70);
servo2.write(60);
```

```
servo3.write(100);
servo4.write(70);
servo5.write(60);
servo6.write(100);
servo7.write(70);
servo8.write(120);
servo9.write(80);
servo10.write(90);
servo11.write(90);
servo12.write(90);
servo13.write(90);
servo14.write(90);
servo15.write(90);
servo16.write(90);
servo17.write(90);
servo18.write(90);
delay(500);
servo1.write(70);
servo2.write(90);
servo3.write(90);
servo4.write(70);
servo5.write(90);
servo6.write(90);
servo7.write(70);
servo8.write(90);
servo9.write(90);
servo10.write(90);
servo11.write(90);
servo12.write(90);
servo13.write(90);
```

```
servo14.write(90);
servo15.write(90);
servo16.write(90);
servo17.write(90);
servo18.write(90);
delay(500);
servo1.write(70);
servo2.write(90);
servo3.write(90);
servo4.write(70);
servo5.write(90);
servo6.write(90);
servo7.write(110);
servo8.write(90);
servo9.write(90);
servo10.write(90);
servo11.write(60);
servo12.write(100);
servo13.write(90);
servo14.write(120);
servo15.write(80);
servo16.write(90);
servo17.write(120);
servo18.write(80);
delay(500);
servo1.write(110);
servo2.write(90);
servo3.write(90);
servo4.write(110);
servo5.write(90);
```

```
servo6.write(90);
servo7.write(70);
servo8.write(90);
servo9.write(90);
servo10.write(110);
servo11.write(60);
servo12.write(100);
servo13.write(70);
servo14.write(120);
servo15.write(80);
servo16.write(70);
servo17.write(120);
servo18.write(80);
delay(500);
servo1.write(110);
servo2.write(90);
servo3.write(90);
servo4.write(110);
servo5.write(90);
servo6.write(90);
servo7.write(70);
servo8.write(90);
servo9.write(90);
servo10.write(110);
servo11.write(90);
servo12.write(90);
servo13.write(70);
servo14.write(90);
servo15.write(90);
servo16.write(70);
```

```
servo17.write(90);
servo18.write(90);
 delay(500);
 return(1);
}
int pruebaFunction()
{
  digitalWrite(led, HIGH);
    for (int x = 1 ; x<2;x++)
 {
servo1.write(90);
servo2.write(150);
servo3.write(130);
servo4.write(90);
servo5.write(150);
servo6.write(130);
servo7.write(90);
servo8.write(30);
servo9.write(50);
servo10.write(90);
servo11.write(150);
servo12.write(130);
servo13.write(90);
servo14.write(30);
servo15.write(50);
servo16.write(90);
servo17.write(30);
servo18.write(50);
 delay(1000);
servo1.write(90);
```

```
 servo2.write(90);
 servo3.write(90);
 servo4.write(90);
 servo5.write(90);
 servo6.write(90);
 servo7.write(90);
 servo8.write(90);
 servo9.write(90);
 servo10.write(90);
 servo11.write(90);
 servo12.write(90);
 servo13.write(90);
 servo14.write(90);
 servo15.write(90);
 servo16.write(90);
 servo17.write(90);
 servo18.write(90);
  delay(1000);
}
  return(1);
 }
  int gcizqFunction()
 {
    if (pos == 90)
 {
    for(pos = 90; pos > 50; pos -= 1)
 {
     camservo.write(pos);
     delay(50);
}
```

```
}
 Else
{
  for(pos == 129;pos > 50;pos -=1)
{
    camservo.write(pos);
    delay(50);
 }
 }
  delay(500);
    return(1);
 }
 int gcderFunction()
{
   if (pos == 90)
{
    for(pos = 90; pos < 130; pos += 1)
{
    camservo.write(pos);
    delay(50);
}
}
 Else
{
  for (pos == 51 ; pos < 130;pos +=1)
{
    camservo.write(pos);
    delay(50);
 }
 }
```

```
    delay(500);
      return (1);
    }
```

- **Visual Basic 2008 Express**

El lenguaje utilizado para la interfaz de control se define en Visual Basic 2008 Express, esta programación se la realizo en lenguaje C++, ya que el compilador soporta varios lenguajes; a continuación se detalla cada parte:

```
Option Strict On
Option Infer On
''' <summary>
''' Puerto Serial
''' </summary>
''' <remarks></remarks>
Public Class mainForm

    Private Bytenumber As Integer
    Private byteEnd(2) As Char
    Private comOpen As Boolean

---------------------------------------------------------------------
#Region "form events"
    ''' <summary>
    ''' Cierra la aplicación y el puerto COM
    ''' </summary>
    Private Sub Form1_FormClosed(ByVal sender As System.Object, _
```

```
                    ByVal e As
System.Windows.Forms.FormClosedEventArgs) _
                    Handles MyBase.FormClosed
        If comOpen Then SerialPort1.Close()
    End Sub
```

```
    ''' <summary>
    ''' Abrir Ventana Formulario
    ''' </summary>
    Private Sub Form1_Load(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles MyBase.Load
        ' lee puertos COM disponibles :
        Dim Portnames As String() = System.IO.Ports.SerialPort.GetPortNames
            If Portnames Is Nothing Then
                MsgBox("There are no Com Ports detected!")
                Me.Close()
            End If
        cboComPort.Items.AddRange(Portnames)
        cboComPort.Text = Portnames(0)
        cboBaudRate.Text = "9600"
 End Sub
```

```
 ''' <summary>
 ''' Abrir puerto Com
 ''' </summary>
```

```
Private Sub btnComOpen_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles btnComOpen.Click

    ' Parametros del dispositivo
    With SerialPort1
        .ParityReplace = &H3B   ";"cuando se produce un error de paridad
        .PortName = cboComPort.Text
        .BaudRate = CInt(cboBaudRate.Text)
        .Parity = IO.Ports.Parity.None
        .DataBits = 8
        .StopBits = IO.Ports.StopBits.One
        .Handshake = IO.Ports.Handshake.None
        .RtsEnable = False
        .ReceivedBytesThreshold =
        .NewLine = vbCr  '
        SerialPort.readLine
        .ReadTimeout = 10000

    End With
    ' comprueba si el dispositivo ha sido desactivado:
    Try
        SerialPort1.Open()
        comOpen = SerialPort1.IsOpen
        Catch ex As Exception
        comOpen = False
```

```
            MsgBox("Error Open: " & ex.Message)
            picOpen.BackColor = Color.Red
        End Try
        If comOpen Then
            picOpen.BackColor = Color.Green
            cboComPort.Enabled = False
            cboBaudRate.Enabled = False
        End If
End Sub
```

```
    ''' <summary>
    ''' Cerrar Puerto Com
    ''' </summary>
    Private Sub Button_Close_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles btnComClose.Click
        If comOpen Then
            ' Limpiar entrada de buffer
            SerialPort1.DiscardInBuffer()
            SerialPort1.Close()
        End If
        comOpen = False
        picOpen.BackColor = Color.Red
        cboComPort.Enabled = True
        cboBaudRate.Enabled = True
End Sub
```

```
    ''' <summary>
    ''' Cerrar Aplicacion
    ''' </summary>
    Private Sub Button_ende_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles btnExit.Click
        If comOpen Then
            ' clear input buffer
            SerialPort1.DiscardInBuffer()
            SerialPort1.Close()
        End If
        comOpen = False
        Me.Close()
 End Sub
```

```
    ''' <summary>
    ''' enviar tecla del panel de control al puerto com
    ''' </summary>
    ''' <param name="sender">return key name</param>
    Private Sub Tasten_Click(ByVal sender As System.Object, _
                             ByVal e As System.EventArgs) _
                             Handles Button8.Click, _
                                     Button6.Click, Button5.Click, _
```

```
                    Button4.Click, Button2.Click, ButtonR.Click, _
                    ButtonL.Click, ButtonI.Click, ButtonO.Click
 Dim key As String = CType(sender, Button).Text
 If comOpen Then SerialPort1.Write(key)
End Sub
#End Region
End Class
```

3.3.3 Montaje Parte Mecánica

En las figuras a continuación se muestra como se armó cada parte del robot:

La placa Arduino se ubicó en la base superior de cabeza para poder reducir al máximo el tamaño del robot y se cambió la altura entre las bases.

Fig.3.31 Ensamble Base superior con placa Arduino/tornillos/Led indicador

La base que sirve para que la cámara del dispositivo android gire se acoplo con un servo motor mediante un pegamento especial.

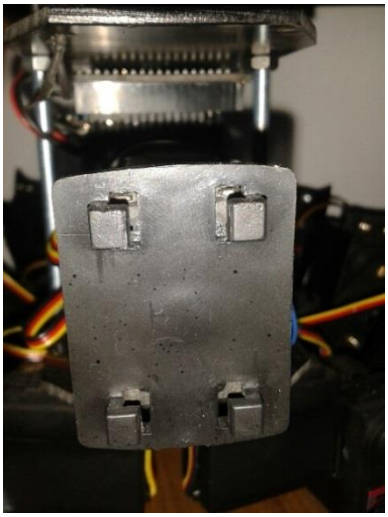

Fig.3.32 Base para soporte de cámara

En la figura 3.33 se muestra la base y el soporte para el dispositivo android.

Fig.3.33 Soporte de dispositivo Android.

Para la estructura de cada extremidad se utilizó acoples metálicos y plásticos; en los engranajes centrales de cada servo motor.

Fig.3.34 Montaje de Extremidad individual.

Los acoples que se usan para montar los servo motores son igualmente de plástico y se atornillan a la base inferior del robot.

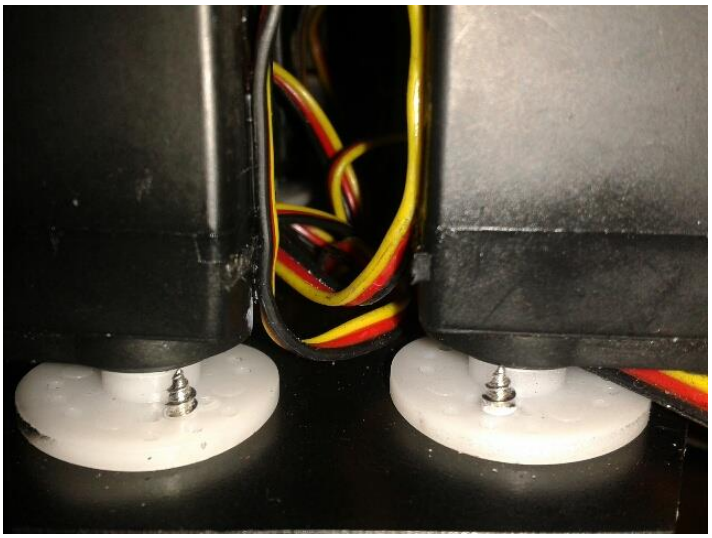

Fig.3.35 Ensamble de Servomotores en base Inferior.

3.4 Implementación

3.4.1 Implementación de Hardware

A continuación se muestra cada una de las partes del robot:

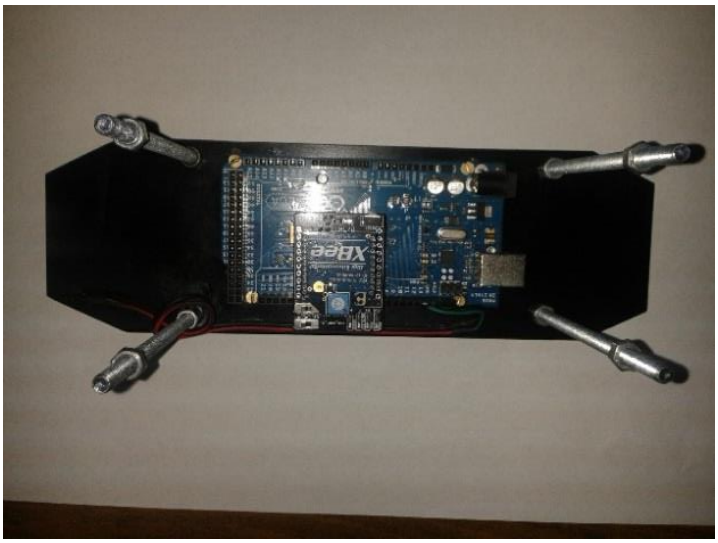

Fig.3.36 Placa implementada en la base superior.

En la figura 3.37 se observa que el pin de GND de la placa Arduino debe ir conectado a GND de cada una de las fuentes, ya que al usar PWM los circuitos son muy sensibles a las caídas de tensión.

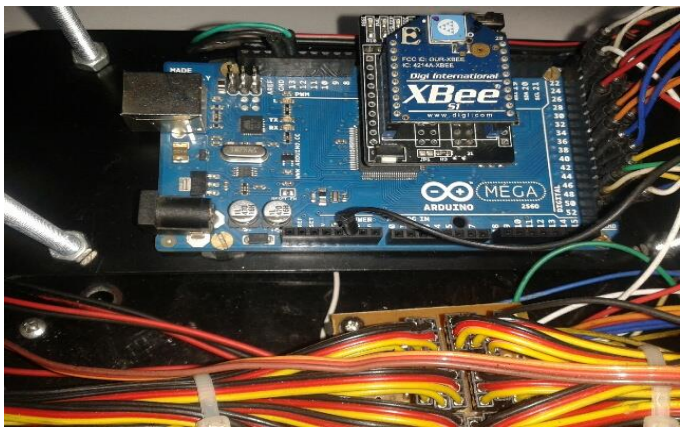

Fig.3.37 Conexión de cableado en placa Arduino

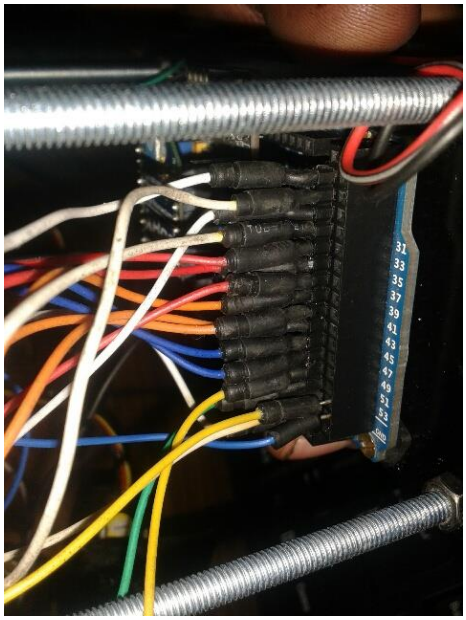

Fig.3.38 Conexión de pines PWM en la placa Arduino

3.4.2 Implementación Mecánica

Para montar el motor con el soporte del dispositivo android se usó un acople de plástico tipo estrella con tornillos en la base inferior.

Fig.3.39 Acople del servomotor de dirección de cámara a la base.

La implementación de todo el sistema se observa en las figuras (3.40, 3.41, 3.42). Aquí se muestra las partes más relevantes para que el funcionamiento del robot.

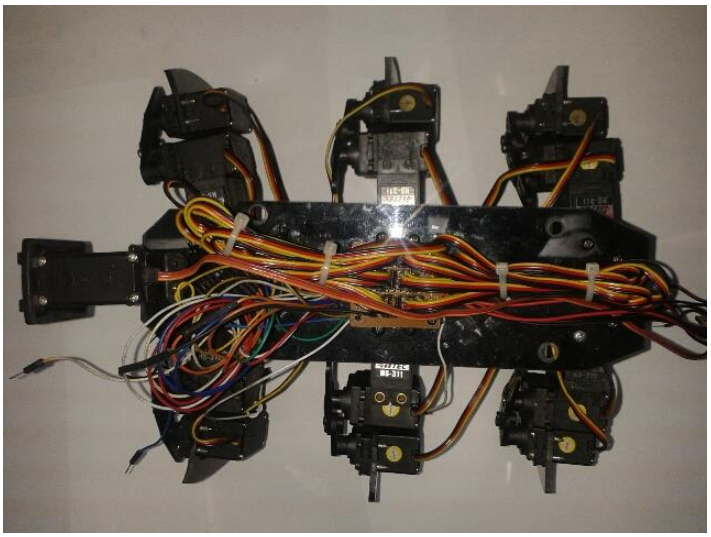

Fig.3.40 Implementación total de servomotores.

Fig.3.41 Implementación de soporte para dispositivo android.

En la figura 3.42 y 3.43 se muestra la vista superior y lateral del sistema.

completo.

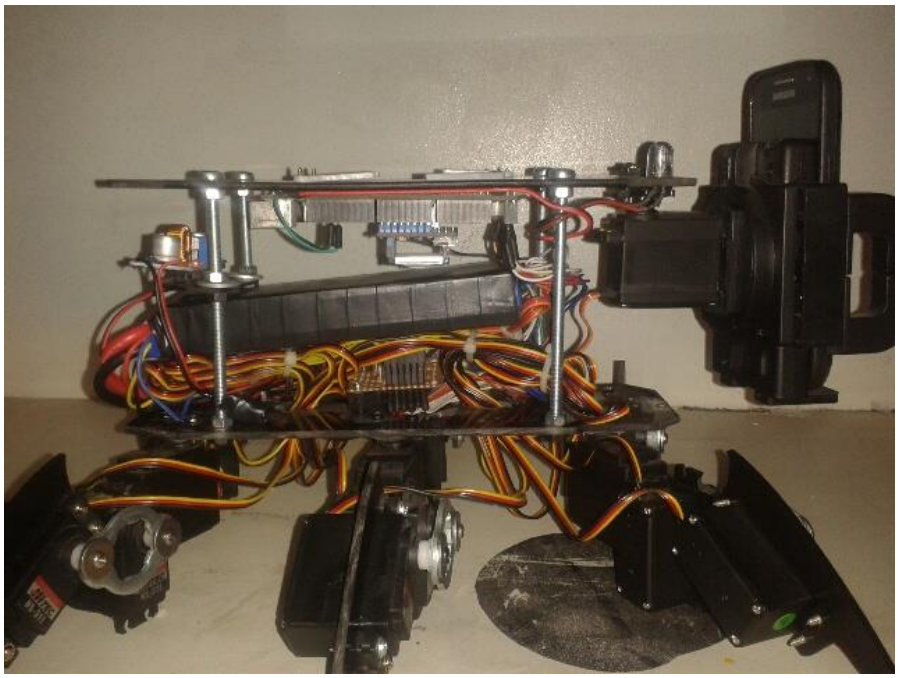

Fig.3.42 Vista lateral del Robot

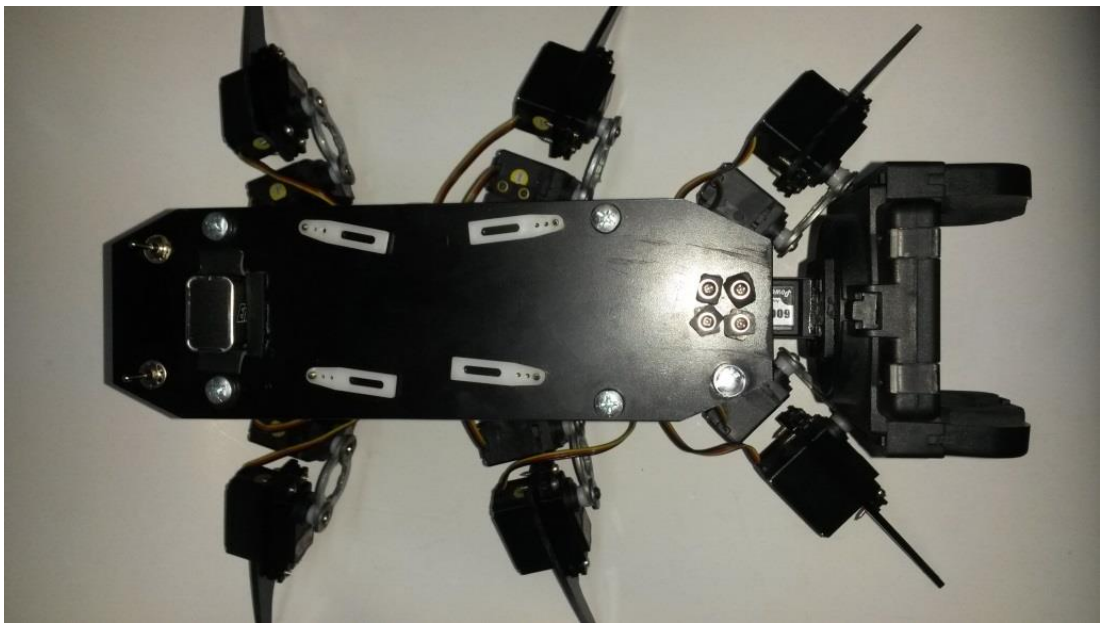

Fig.3.43 Robot Implementado Completamente.

En la siguiente figura se muestra la implementación de los switch de encendido y el regulador de voltaje, también la colocación de las baterías en su lugar.

Fig.3.44 Vista Posterior del Robot.

CAPÍTULO 4

RESULTADOS Y COSTOS

Introducción

En este capítulo se muestra el análisis de las pruebas realizadas bajo los parámetros establecidos; además se presenta un análisis FODA del sistema implementado; y finalmente se presenta una lista de costos de los equipos utilizados en el proyecto.

4.1. Pruebas de funcionamiento

4.1.1. Pruebas de validación de Hardware

N°	Prueba	Activación	Funcionamiento	Observación
1	Placa Arduino	✓	Funciona con normalidad.	Sin observación.
2	Led Indicador	✓	Funciona con normalidad.	Sin observación.
3	Módulo Xbee	✓	Funciona con normalidad.	Este módulo se energiza con 3.3V,
4	Shield Arduino/Xbee	✓	Funciona con normalidad.	Para la conexión con la Pc, es necesario conmutar los switch en USB, ya que este permite comunicación serial y paralelo.
5	Mov. 1 Servo	✓	Funciona con	El movimiento se da por un

	motor HS-311		normalidad.	pulso de 1500us, en este caso con la plataforma Arduino se programa mediante grados. Estos servomotores, al iniciar su funcionamiento se posicionan en 90°, es decir, en el centro.
6	Mov. 2 Servo motores HS-311	✓	Funciona con normalidad.	Al conectar dos motores se observó que la fuente utilizada no tiene la suficiente corriente para la requerida por estos servomotores.
7	Mov. 3 Servo motores HS-311	✓	Funciona con normalidad.	Para completar el movimiento de una extremidad se conectaron 3 motores, por lo cual se cambió la fuente de alimentación.
8	Mov. Extremidad #1	✓	Funciona con normalidad.	Sin observación.
9	Mov. Extremidad #2	✓	Funciona con normalidad.	Sin observación.
10	Mov. Extremidad #3	✓	Funciona con normalidad.	Sin observación.
11	Mov. Extremidad	✓	Funciona con	Sin observación.

	#4		normalidad.	
12	Mov. Extremidad #5	✓	Funciona con normalidad.	Sin observación.
13	Mov. Extremidad #6	✓	Funciona con normalidad.	Sin observación.
14	Mov. Sistema Completo	✓	Su función es irregular.	El movimiento del sistema, es irregular es inapropiado para los requerimientos por el peso total. Fallo servomotor intermedio ya que el torque debe ser mayor para soportar el peso. Se cambió de serie del servomotor del hs-311 al hs-600 con un torque de 7Kg/cm.
15	Comunicación Pc-Xbee	✓	Funciona con normalidad.	Al conectar el módulo Xbee/Shield por medio de un cable USB, a la Pc y mediante el Software X-CTU se verifico los parámetros de funcionamiento y de conexión.
16	Comunicación Xbee-Xbee	✓	Funciona con normalidad.	Cuando se envió el primer dato Serial, la comunicación se verifico, encendiendo el led indicador.
17	Dispositivo	✓	Funciona con	El dispositivo usado es el

	Android		normalidad.	Samsung Mini-S3. El software es compatible con cualquier dispositivo android.
18	Cámara	✓	Funciona con normalidad.	La cámara del teléfono usado es de 5 Mega Pixeles.

Tabla 4.1 Pruebas de Hardware

4.1.2 Pruebas de validación de software

N°	**Prueba**	**Funcionamiento**	**Observación**
1	Exportar la Librería para servomotores en Arduino.	Funciona con normalidad.	No todos los pines de la placa son PWM, con lo cual funciona un servo motor para su movimiento. Con esta librería todos pines digitales se convierten en salidas PWM.
2	Reconocimiento de puerto COM con Arduino.	Incompatible con la versión de Windows.	El puerto COM no se reconoce por falta del driver FDTP. Para simular el puerto USB como serial.

Tabla 4.2 Pruebas de Software

4.1.3. Pruebas de Operatividad

4.1.3.1 Pruebas de Motricidad

Prueba	Funcionalidad	Observación
1	Extremidad #1 Extremidad #2 Extremidad #3 Extremidad #4 Extremidad #5 Extremidad #6	De acuerdo con el modelo del servomotor utilizado, se tiene un movimiento de 0° a 180°, con variación según la necesidad en 1°.
2	Primera secuencia de motricidad entre extremidades 1, 2, 3.	Se observó que una sóla fuente no abastece con la corriente suficiente a 9 servo motores, sino a seis servomotores por lo cual se utiliza 3 fuentes separadas para todas las extremidades.
3	Segunda secuencia de motricidad con las 6 extremidades.	Existe una latencia o retardo al momento de realizar el movimiento de todos los servomotores.
4	Conexión inalámbrica, prueba de distancias para el control desde la PC.	Se observó cierto retardo al enviar la señal, pero no es relevante en general.
5	Conexión de la cámara y envió de video.	Existe un retardo de 1s en el envió del video.

Tabla 4.3 Pruebas de Motricidad

4.1.3.2 Pruebas de Destreza

Prueba	Funcionalidad	Observación
1	Trayectoria hacia adelante.	Existe un desbalance entre los ángulos programados para cada extremidad y el soporte armado para cada uno.
2	Trayectoria hacia atrás.	Existe un desbalance entre los ángulos programados para cada extremidad y el soporte armado para cada uno.
3	Trayectoria hacia derecha.	Existe un desbalance entre los ángulos programados para cada extremidad y el soporte armado para cada uno.
4	Trayectoria hacia izquierda.	Existe un desbalance entre los ángulos programados para cada extremidad y el soporte armado para cada uno.
5	Giro anti horario.	Sin Observación.
6	Giro horario.	Sin Observación.
7	Trayectoria en semicircunferencia.	El tiempo en direccionar el mecanismo, es considerable para un operario que no ha tenido una capacitación previa.

Tabla 4.4 Pruebas de Destreza

4.2 Análisis de Resultados

4.2.1 Pruebas de Hardware

Prueba1: La placa Arduino no tuvo ninguna observación.

Prueba2: El led indicador no mostro ninguna observación.

Prueba3: Al energizar el módulo Xbee, se observó en las especificaciones técnicas que necesita un voltaje máximo de 3.3V, por lo que gracias al shield acoplador, al conectar a la PC y a la placa Arduino no se detectó ninguna anomalía.

Prueba4: Los módulos Xbee deben ser configurados para comunicación por USB, estos dispositivos también se pueden comunicar mediante datos serie o paralelo, pero en este caso se ha utilizado USB, ya que la PC usada posee sólo puertos USB.

Prueba5: Al realizar la prueba de movimiento el servomotor HS-311, se dio un pulso de 1500us, en este caso con la plataforma Arduino se programa mediante grados. Estos servomotores, al iniciar su funcionamiento se posicionan en 90°, es decir, en el centro.

Prueba 6 y 7: Al energizar 3 servomotores y probar su funcionamiento en secuencia se observó que pierden torque, por esto se cambió la fuente de alimentación por una que provee más corriente.

Prueba 8, 9, 10, 11, 12, 13: Cada extremidad del robot al cambiar de fuentes funcionan con normalidad.

Prueba 14: Cada servomotor tiene que operar soportando un torque determinado, al poner en funcionamiento todo el sistema se observó que los servos intermedios no responden a ciertos movimientos. Cuando se cambió el servo por un modelo más adecuado para soportar el torque requerido funciono con normalidad.

Prueba 15: Al conectar el módulo Xbee mediante el cable USB e iniciar el Programa X-CTU de DIGI, se comprobó cada parámetro que se configuró para la comunicación serial y se verifico que la conexión se realizó sin ninguna interrupción.

Prueba 16: Para poder verificar el envió de datos mediante el puerto serial (COM), se envió un bit como carácter, el cual respondió encendiendo el led indicador; iniciando así la comunicación para cada función que controla el sistema.

Prueba 17, 18: Las funciones del dispositivo android que se requieren operan de manera correcta.

4.2.2 Pruebas de Software

Prueba 1: Una de las facilidades que proporciona la plataforma de Arduino y sus utilidades es que existen librerías que modifican el funcionamiento de estas, es el caso de pines con PWM, que en la Arduino 2560 solo especifica 14, pero mediante la librería (#include <Servo.h>) se convierte todos los pines digitales en pines que envían pulsos PWM.

Prueba 2: Al no poseer un puerto serial en la PC, hubo que simular un puerto serial y esto se lo realizó mediante un driver FDTP, el cual simula puertos USB como si fueran serial (COM).

4.2.3 Pruebas de Motricidad

Prueba1: Mediante el software y los servos utilizados se pudó variar el movimiento de cada extremidad del robot de 0° a 180° en paso de 1 grado sin ninguna novedad.

Prueba2: La fuente usada en un principio no abasteció con la corriente necesaria para 3 extremidades, sino solo para 2; por lo que se cambió de fuente y se energizó correctamente.

Prueba3: Por la velocidad del procesamiento del microcontrolador usado, y por la cantidad de instrucciones que se deben procesar, se pudo verificar que hubo un retardo casi imperceptible al momento de

poner en funcionamiento al robot.

Prueba4: Al iniciar la conexión con el envió de los datos seriales al robot, se pudo verificar que existe cierto retraso en el proceso de realizar la función correspondiente, pero en general no es un factor determinante en el funcionamiento del sistema completo.

Prueba5: Él envió del video se realizó exitosamente mediante el servidor del programa IPWEBCAM con conexión WIFI.

4.2.4 Pruebas de Destreza

Prueba1, 2, 3, 4, 7: La sincronización de cada servo y de cada extremidad en conjunto es el aspecto más importante al momento de poner en funcionar todo el sistema, en las primeras pruebas se observo que no todos los motores no tienen la misma precisión, por lo cual se realizó una prueba con cada motor por separado y por consiguiente cada extremidad, graduando cada motor al número de grados que debía girar para que no exista desbalance en soporte que debía mantener en cada secuencia de movimiento.

4.3 Matriz FODA

Fortalezas	Oportunidades
• Los materiales utilizados para la construcción del robot son fáciles de reemplazar y de bajo costo. • Los programas diseñados, pueden ser modificados para acoplar el sistema de acuerdo a la necesidad del operario. • La placa electrónica para la comunicación inalámbrica usa un mínimo de energía.	• De acuerdo a la necesidad que se presente, el cliente puede hacer uso de este dispositivo para otras aplicaciones. • Puede producirse en serie ya que no tiene ningún componente especial. • El sistema puede utilizarse como un fiscalizador de cableado estructurado.
Debilidades	**Amenazas**
• El material utilizado no soporta condiciones de terreno extremas. • La corriente utilizada por los motores tiene un nivel considerable por lo cual el tiempo de duración de las baterías es limitado. • No siempre se tendrá conexión a internet por lo que la cámara no funcionará en todos los casos.	• Puede ser copiado ya que se usa materiales sencillos. • No posee protección para ambientes hostiles, es el caso de descargas eléctricas. • Por rubros adicionales acerca del servicio de este sistema, puede no requerirse.

Tabla 4.5 Matriz FODA.

4.4. Costos del Proyecto

En la Tabla 4.6. Se muestran los costos de los equipos y elementos utilizados y así también la mano de obra requerida para la implementación.

Equipos	Costo
19 servomotores	285.00
Placa Arduino Mega 2560	70.00
2 módulos Xbee S1	60.00
2 Shield Arduino/Xbee	40.00
Smartphone Samsung Galaxy Mini S4	350.00
Cable USB	3.00
Componentes electrónicos	50.00
Materiales de piezas acrílicas de construcción	40.00
Mano de Obra	300.00
Total:	**$ 1198.00**

Tabla 4.6 Costos del Proyecto.

CAPÍTULO 5

CONCLUSIONES Y RECOMENDACIONES

5.1. Conclusiones

- Se aprovechó de las ventajas que proveen varias plataformas de programación y de diseños de placas electrónicas.

- El sistema diseñado permite que el prototipo trabaje eficientemente y que el usuario opere con facilidad el sistema.

- Los sistemas inalámbricos son de gran utilidad, ya que usados de manera correcta y eficiente logran el desarrollo de nuevos procesos.

- Se desarrolló un sistema para facilitar ciertos trabajos en cableado estructurado, lo que genera un interés adquisitivo de esta herramienta por parte de las empresas que realizan este trabajo.

- Se estructuró el sistema de control remoto, mediante drivers que pueden ser usados por varios programas o plataformas de comunicación.

- El uso de la plataforma Arduino se debe a que el control de servomotores es sencillo y requiere programación de alto nivel, lo cual facilita la programación para las secuencias de giro(ángulo) que deben realizar cada uno de los servomotores.

5.2. Recomendaciones

- Para lograr un adecuado desempeño del sistema, se sugiere que las fuentes de energía estén en un rango mínimo operativo del 30% de su capacidad.

- Para el desarrollo de nuevos robots que faciliten el trabajo del ser humano no es necesario la creación de nuevas tecnologías, sino aplicar o fusionar las ya existentes.

- Se puede agregar nuevas funcionalidades para mejorar el uso del sistema.

- Al escoger los componentes que incluya un proyecto, verificar las características de los mismos, para poder medir mediante los parámetros de diseño la correspondencia de cada requerimiento.

- Se puede implementar proyectos muy factibles mediante las placas electrónicas de la plataforma Arduino.

Web grafía

- (2009, 12). Puerto Paralelo Pic 16F877A. BuenasTareas.com. Recuperado de: htmlhttp://www.neoteo.com/servomotores-el-primer-paso-hacia-tu-robot
- (2012, 6). Librería Servo. Arduino.com. Recuperado de: http://arduino.cc/es/Reference/Servo
- (2010, 5). Adafruit 16-Channel 12-bit PWM/Servo Driver - I2C interface - PCA9685. Adafruit.com. Recuperado de: www http://www.adafruit.com/products/815
- (2008, 2). Knob. Arduino.com. Recuperado de: http://arduino.cc/es/Tutorial/Knob
- (2011, 4). Guía para Arduino. tdrobótica.co. Recuperado de: http://www.tdrobotica.co/tutoriales/arduino
- (2011, 12). Mauricio Velandia. Arduino avanzado T00AA - Control de un servomotor con PWM. Tdrobótica.co. Recuperado de: http://www.tdrobotica.co/tutoriales/arduino/312-arduino-avanzado-t00aa-control-de-un-servomotor-con-pwm
- (2009, 6). A. Tamayo. Comunicación Serial. galaxi0.wordpress.com. Recuperado de: http://galaxi0.wordpress.com/el-puerto-serial/
- (2012, 5). Redes Inalámbricas. technohall.com. Recuperado de: http://technohall.com/2012/05/tips-para-implementar-redes-inalambricas-de-sensores-con-xbee/
- (2009, 5). Módulos de Transmisión Inalámbrica. Xbee.cl. Recuperado de: http://www.xbee.cl/
- (2012, 2). XBee & XBee-PRO OEM RF Module Antenna Considerations. Digi.com. Recuperado de: http://ftp1.digi.com/support/images/XST-AN019a_XBeeAntennas
- (2010, 1). Guia de Usuario, xbee Serie 1. Mci electronics. Recuperado de: http://www.olimex.cl/pdf/Wireless/ZigBee/XBee-Guia_Usuario

- (2009, 7). Batería de Litio. Wikipedia.com. Recuperado de: https://www.google.com.ec/search?q=bateria+de+litio