UNIVERSIDAD CATÓLICA
DE SANTIAGO DE GUAYAQUIL

FACULTAD DE EDUCACIÓN TÉCNICA PARA EL DESARROLLO
CARRERA DE INGENIERÍA EN TELECOMUNICACIONES

TÍTULO:

DISEÑO E IMPLEMENTACION DE UN ROBOT MOVIL SOCCER UTILIZANDO LA TARJETA ARDUINO NANO Y CONTROLADO MEDIANTE BLUETOOTH.

AUTOR:
KARLA ERENIA JACOME GUERRERO

Previa la obtención del Título
INGENIERO EN TELECOMUNICACIONES

TUTOR:
M. Sc. Fernando Palacios Meléndez

Guayaquil, Ecuador
2016

UNIVERSIDAD CATÓLICA
DE SANTIAGO DE GUAYAQUIL

FACULTAD DE EDUCACIÓN TÉCNICA PARA EL DESARROLLO
CARRERA DE INGENIERÍA EN TELECOMUNICACIONES

CERTIFICACIÓN

Certificamos que el presente trabajo fue realizado en su totalidad por el Sr. **Karla Erenia Jácome Guerrero** como requerimiento parcial para la obtención del título de INGENIERO EN TELECOMUNICACIONES.

TUTOR

M. Sc. Fernando Palacios Meléndez

DIRECTOR DE CARRERA

M. Sc. Miguel A. Heras Sánchez.

Guayaquil, a los 14 del mes de Marzo del año 2016

UNIVERSIDAD CATÓLICA
DE SANTIAGO DE GUAYAQUIL

FACULTAD DE EDUCACIÓN TÉCNICA PARA EL DESARROLLO
CARRERA DE INGENIERÍA EN TELECOMUNICACIONES

DECLARACIÓN DE RESPONSABILIDAD

Yo, **Karla Erenia Jácome Guerrero**

DECLARÓ QUE:

El trabajo de titulación "**Diseño e implementación de un robot móvil soccer utilizando la tarjeta ARDUINO NANO y controlado mediante bluetooth.**" previa a la obtención del Título de Ingeniero en Telecomunicaciones, ha sido desarrollado respetando derechos intelectuales de terceros conforme las citas que constan al pie de las páginas correspondientes, cuyas fuentes se incorporan en la bibliografía. Consecuentemente este trabajo es de nuestra autoría.

En virtud de esta declaración, me responsabilizo del contenido, veracidad y alcance científico del Trabajo de Titulación referido.

Guayaquil, a los 14 del mes de Marzo del año 2016

EL AUTOR

KARLA ERENIA JACOME GUERRERO

UNIVERSIDAD CATÓLICA
DE SANTIAGO DE GUAYAQUIL

FACULTAD DE EDUCACIÓN TÉCNICA PARA EL DESARROLLO
CARRERA DE INGENIERÍA EN TELECOMUNICACIONES

AUTORIZACIÓN

Yo, **Karla Erenia Jácome Guerrero**

Autorizamos a la Universidad Católica de Santiago de Guayaquil, la publicación, en la biblioteca de la institución del Trabajo de Titulación: "**Diseño e implementación de un robot móvil soccer utilizando la tarjeta ARDUINO NANO y controlado mediante bluetooth**", cuyo contenido, ideas y criterios es de mi exclusiva responsabilidad y autoría.

Guayaquil, a los 14 del mes de Marzo del año 2016

EL AUTOR

KARLA ERENIA JACOME GUERRERO

DEDICATORIA

A mis padres Carlos Jácome y Nancy Guerrero.
A mis hermanas Nancy, Katherine, Joyce.
A mi sobrino Alejandro.

EL AUTOR

KARLA ERENIA JACOME GUERRERO

AGRADECIMIENTO

Quiero agradecer primero a Dios que con sus bendiciones y su protección me ha dado la fuerza para salir adelante y superar todas las dificultades que en tenido en el trayecto de mi vida.

A mis padres que con su amor, su ejemplo y su total apoyo han sabido guiarme en todo momento, han sido mi motor para poder alcanzar todas mis metas.

A toda mi familia que ha estado presente en todo momento, a mis profesores, compañeros y amigos de la universidad.

Al Ing. Edwin Palacios por su apoyo y guía al realizar este trabajo.

EL AUTOR

KARLA ERENIA JACOME GUERRERO

Índice General

Índice de Figuras

Capítulo 2

Capítulo 3

Índice de Tablas

Capítulo 2

Capítulo 3

Resumen

Durante los años de formación de Ingeniería en Telecomunicaciones, los estudiantes decidimos en el año 2014 incursionar en participaciones de Concursos de Robótica y formar el Club de Robótica “ROBOFET”, en la que a través de proyectos de tutoría y de trabajos de titulación hemos incrementado las categorías para participaciones nacionales. El propósito del presente trabajo de titulación, es contribuir e incentivar a los estudiantes en el desarrollo de nuevos prototipos de robots móviles autónomos y controlados a través de dispositivos de comunicaciones inalámbricas, tales como Xbee y Bluetooth. Para el desarrollo del robot soccer se hizo una búsqueda de prototipos y en base a esta información, se logró realizar un diseño funcional, tanto en la parte microelectrónica y de la interfaz de comunicación, que en este proyecto se utiliza la tecnología Bluetooth. El comportamiento del robot soccer durante las pruebas realizadas, cumplió con las expectativas que se propusieron en este trabajo. No existieron inconvenientes en la comunicación mediante tecnología Bluetooth para controlar y operar el robot.

CAPÍTULO 1: INTRODUCCIÓN

1.1. Antecedentes.

A nivel nacional los concursos de robótica que se desarrollan cada año, va tomando mayor fuerza. Las Instituciones de Educación Superior (IE) cuentan con sus respectivos Clubes de Robótica, especialmente en aquellas donde se ofrecen carreras, tales como, Ingeniería en Telecomunicaciones, Electrónica, Electricidad, entre otras relacionadas a la Ciencia de la Ingeniería.

Existen algunos eventos que desarrollan cada año, en eventos nacionales, están:

a. CER (Concurso Ecuatoriano de Robótica): este evento se desarrolla cada año en Universidades Públicas o Privadas, de acuerdo a la designación que se realiza en reunión de todos los docentes representantes de los Clubes de Robótica de las IES inscritas en el CER. En diciembre 2015 se desarrolló el XI CER en el campus de la Universidad Católica de Santiago de Guayaquil con 22 categorías entre ellas Robot Soccer o Futbol que es el tema titulación propuesto, en la misma participaron 21 Universidades entre públicas y privadas. Cabe indicar que para participar en el CER no hay cobro de inscripciones.
b. UMEBOT: es otro concurso nacional de robótica, impulsado por estudiantes del club de robótica de la Escuela Politécnica Nacional

(EPN) pero se cobra una cuota de inscripción en todas las categorías. También, existe la categoría de Soccer.

También, se realizan concursos internacionales de robóticas organizados por representantes ecuatorianos. Los concursos internacionales que son realizados en nuestro país, son: Robot Games Zero Latitud, y Riotronics. El primero es realizado por CAD Metronics una organización sin fines de lucro, que tiene acreditaciones para participaciones en concursos de robótica internacional, en México, Colombia, Japón, entre otros países.

1.2. Justificación del Problema.

El Club de Robótica "ROBOFET" de las Carreras de Ingeniería en Telecomunicaciones y Electrónica en Control y Automatismo de la Facultad de Educación Técnica para el Desarrollo de la Universidad Católica de Santiago de Guayaquil, hasta la presente ha participado en pocas categorías, tales como, Megasumo, Minisumo y Creatividad.

El Club ROBOFET no ha tenido participaciones en la categoría de Soccer, y el propósito es disponer de un robot controlado inalámbricamente mediante tecnología Bluetooth. Para el diseño del robot soccer, se considerará los reglamentos y dimensiones para su correcta implementación y posterior homologación en los futuros concursos de robótica que ROBOFET desee participar.

1.3. Definición del Problema.

Necesidad de realizar el diseño e implementación de un robot móvil no autónomo utilizando el microcontrolador Atmega 328 incorporado en la tarjeta Arduino Nano y que se controlado mediante tecnología Bluetooth.

1.4. Objetivos del Problema de Investigación.

1.4.1. Objetivo General.

Diseñar e Implementar un Robot Móvil Soccer utilizando la tarjeta Arduino Nano y controlado mediante la tecnología Bluetooth.

1.4.2. Objetivos Específicos.

- Describir los fundamentos teóricos de los microcontroladores y de hardware abierto.
- Diseñar el circuito esquemático utilizando el microcontrolador Atmega 328 integrado en la tarjeta Arduino Nano.
- Integrar el dispositivo de comunicación Bluetooth para controlar inalámbricamente al Robot Soccer.
- Evaluar el funcionamiento del Robot Soccer y del dispositivo inalámbrico Bluetooth.

1.5. Hipótesis.

La utilización del microcontrolador atmega328 integrado en la tarjeta Arduino nano y articulado con la tecnología bluetooth, permiten implementar el Robot Soccer. El Diseño e Implementación del Robot Soccer permitirá

evidenciar el conocimiento adquirido en las asignaturas de Microcontroladores, Diseño Electrónico Digital y Comunicaciones Inalámbricos en la Carrera de Ingeniería en Telecomunicaciones, también será de gran utilidad para futuras participaciones en los Concursos de Robótica que se realizarán en este año 2016.

1.6. Metodología de Investigación.

El presente proyecto de titulación de carácter académico busca utilizar varios tipos de estudio aplicables a proyectos tecnológicos de orden académico, tanto teóricos como sistemáticos. El empleo de métodos sistemáticos, que está enfocado en la modelación del objeto en el que se trabaja mediante la identificación de sus elementos, su funcionamiento y su relación, de los cuales se determina su dinámica y su estructura.

CAPÍTULO 2: FUNDAMENTACIÓN TEÓRICA

2.1. Sistemas Microcontroladores.

Los microcontroladores se crean para resolver los problemas económicos y de tamaño de los microprocesadores. Está estructurado por un solo circuito integrado, las memorias (de datos y programa) y unidades de entrada/salida, esto hace que sea muy útil para varias aplicaciones. En la figura 2.1 se muestra el sistema basado en microcontroladores, pero perteneciente a la familia de los PICs de la empresa Microchip Technology.

Figura 2. 1: Sistema basado en microcontroladores.

Fuente: (*Electrónica & microcontroladores PIC*. USERSHOP)

Existen en el mercado otros dispositivos microcontroladores, tales como Atmega (Atmel), 8051 (Dallas Semiconductor), ARM Cortex-M4 (Texas Instruments), entre otros. En la tabla 2.1 se muestra el listado de fabricantes/proveedores de los diferentes microcontroladores que existen en el mercado mundial.

Tabla 2. 1: Lista de fabricantes de microcontroladores.

Fabricantes/Proveedores	Microcontroladores
Atmel	Series AT89, AT90, ATtiny, ATmega, Series ATxmega
Dallas Semiconductor	Familia 8051, Familia MAXQ RISC, Familia Secure Micros
ELAN Microelectronic Corp	EM78PXXX Low pin-Count, Tipo GPIO, Tipo EM78PXXXN ADC
EPSON Semiconductor	Familia SIC6x (4-bit), Familia SIC88 (8-bit), Familia SIC17 (16-bit), Familia SIC33 (32-bit)
Freescale Semiconductor	68HC05, 68HC08, 68HC11 (8-bit), 68HC12, 68HC16 (16-bit), 683XX, MCF5xxx, M-core, MPC500, MPC860 (32-bit)
Holtek	HT48FXX flash I/O , HT48RXX I/O, HT46RXX A/D
Intel	MCS-51, 8xC251 (8-bit), MCS-96, Intel MCS-296
Microchip Technology	PIC10, PIC12, PIC16, PIC18 series (8-bit), PIC24, dsPIC (16-bit), PIC32MX series
NXP Semiconductor	ARM7/LPC2000, ARM9/LPC3000, ARM Cortex-M0/LPC800,LPC1100, LPC1200, ARM Cortex-M3/LPC1300, LPC1700, LPC1800
National Semiconductor	COP400, COP8, SC/MP, CR16
ST Microelectronics	ST6, ST7, STM8, uPSD (8-bit), ST10 (16-bit), ST20, ARM7/STR7, ARM9/STR9, ARM Cortex-M0/STM32 F0, ARM Cortex-M3/STM32 F1, F2
Texas Instruments	TMS370 (8-bit), MSP430 (16-bit), TMS 320, ARM Cortex-R4/TMS570 (32-bit)
Zilog	Zilog eZ8, Zilog eZ80, Zilog Z16
Maxim Integrated	8051, ARM922T, MAXQ20, MAXQ30, MIPS4kSD
NEC	78K, 75X, 17K, MPD78C14, V25, V850

Fuente: (Kumar P., Das, & Das, 2013)

2.2. Microcontroladores especializados y de aplicación general.

Según los circuitos de interfaz de entrada y salida, los microcontroladores se clasifican en:

- Microcontroladores especializados.

Estos microcontroladores están fabricados con un software y hardware para un área específica de tecnología, a la vez estos microcontroladores se

dividen en comerciales y en los de aplicación específica. Los microcontroladores especializados comerciales tienen circuitos de interfaz configurables y periféricos internos. Estos beneficios ayudan a un mejor funcionamiento de las placas electrónicas como por ejemplo lectores, discos compactos, etc.

- Microcontroladores de aplicación general.

Estos microcontroladores se los conoce por tener unos circuitos de interfaz de entrada y salida, como por ejemplo puerto serie, puerto paralelo. Y también bloques funcionales determinados como lo son los temporizadores, relojes analógicos, etc. Para que estos microcontroladores tengan un buen funcionamiento deben de tener circuitos programables, es decir que pueden ser modificados mediante la ejecución de su programa. Como sabemos uno de los microcontroladores más conocidos y utilizados son los PIC.

2.3. Microcontroladores PIC

Los microcontroladores abarcan un área fundamental de la electrónica aplicada, ya que son los que abren el campo de los procesadores digitales en los diferentes artefactos u objetos industriales. Sin embargo sabemos que dichos artefactos son parte de un mundo lleno de complejidad sea en el ámbito de programación o sistemas, ya que tenemos que tener en cuenta cual es el grado de fiabilidad de dichos objetos; hay que ser siempre precavidos en sus diseños en cada uno de sus sistemas.

A través de los años nos hemos dado cuenta que la tecnología en lo que electrónica ha avanzado en un 100%, por ejemplo los autos actualmente utiliza un sistema de microcontroladores en los cuales permite que tenga un buen funcionamiento, así como sabemos que se han mezclado del uso de hardware y el software. Con el bus funcionamiento en conjunto de estos dos, nos podemos dar cuanta que la tecnología está haciendo un buen avance en el mundo de los microcontroladores y sus tarjetas.

Como para conocimiento de nosotros un microprocesador es un circuito integrado digital monolítico que comprende todos los fundamentos de un procesador digital secuencial síncrono programable de una estructura Harvard o Princeton. Otro nombre con el cual se lo conoce es microcomputador integrado o empotrado en ingles se la conoce como EMBEDDED PROCESOR estos circuitos están dirigidos a la comunicación y tareas de control.

Debemos tener en cuenta que los microcontroladores a pesar de su tamaño tienen funciones vitales para los artefactos electrónicos más que todo en el campo industrial. Al momento de emplear energía y velocidad son muy ahorrativas para que las aplicaciones puedan ejecutarse sin ningún problema. Los PIC contienen diferentes programas de seguridad que se los conoce como safety y los programas que tienen mecanismo de almacenamiento que son los security.

Los microcontroladores se los emplean en los diferentes objetos electrónicos como las cocinas, teléfonos, circuitos de control de red, controlador de un brazo de robot. En todos los controladores están formados por sistemas integrados MSI y LSI. (Mandado P & Lopez M, 2007).

2.3.1. Estructura de los Microcontroladores PIC

Los PIC tienen una arquitectura: de memorias de datos y de programas por separado. La memoria de programa se estructura en 12, 14 o 16 bits y la de datos en 8 bits. Existen algunos registros de funciones especiales (SFR), que permiten el acceso a varios dispositivos de entrada y salida, estos microcontroladores tienen algo memoria EEPROM que sirve para almacenar datos.

Figura 2. 2: Microcontrolador PIC.

Fuente (*Electrónica & microcontroladores PIC*. USERSHOP)

Los microcontroladores PIC tienen entre 33 y 77 instrucciones las cuales son del mismo tamaño 12, 14 o 16 bits, también tienen un registro de trabajo y registro de la memoria de datos, la transferencia de datos se la efectúa entre

estos dos registros y en los PIC de gama alta se realizan estas transferencias directas.

Todos estos microcontroladores tienen la técnica de segmentado también llamado pipeline, las instrucciones se realizan en un solo ciclo, correspondiente a cuatro ciclos del oscilador principal del microcontrolador, menos las transferencias de control que tienen dos ciclos de instrucción (Perez F, 2007). Algunos de los PIC de 8 bits son PIC12, PIC16, PIC18.

La pila de los PIC está de manera independiente y tiene una profundidad limitada, también tienen un temporizador y un cierto número de bits que sirven para configurar el dispositivo. La afinidad entre estos microcontroladores hace que con pocos cambios se puedan portar de uno hacia otro.

2.3.2. Arquitectura de los Microcontroladores PIC.

La arquitectura de un microcontrolador depende de la aplicación para la que está construido. Por ejemplo, algunos diseños incluyen el uso de más de una memoria RAM, ROM y la funcionalidad de dispositivos de E/S integradas en el chip. En la figura 2.3 se muestra la arquitectura general de los microcontroladores. La arquitectura de un microcontrolador típico es complejo y puede incluir lo siguiente (EngineersGarage, 2015):

1. Una CPU, que van desde simples procesadores de 4 bits hasta los procesadores complejos de 64 bits.
2. Periféricos tales como temporizadores, contadores y watchdog.

3. Memoria RAM (memoria volátil) para el almacenamiento de datos. Los datos se almacenan en forma de registros, y los registros de propósito general almacenan información que interactúa con la unidad lógica aritmética (ALU).
4. Memorias ROM, EPROM, EEPROM o memoria flash para el almacenamiento de programas y parámetros de funcionamiento.
5. Capacidades de programación.
6. Serial de E/S, como puertos serie.
7. Un generador de reloj para el resonador, cristal de cuarzo o de temporización del circuito RC.
8. Convertidor Analógico/Digital.
9. Puertos seriales.
10. Bus de datos para transmitir información.

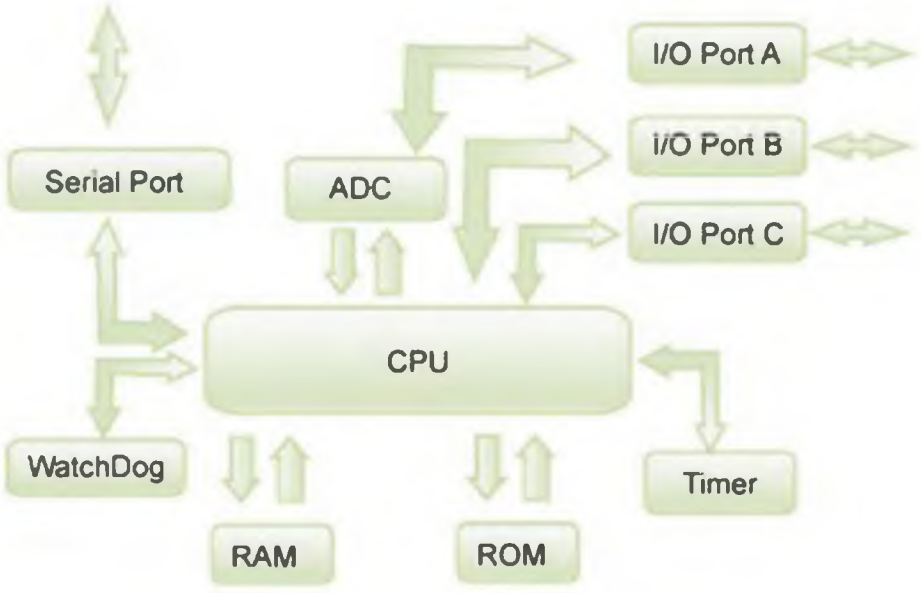

Figura 2. 3: Arquitectura básica general de microcontroladores PIC.

Fuente: (EngineersGarage, 2015)

En la figura 2.4 se muestra la arquitectura de un microcontrolador PIC de la familia de 8 bits.

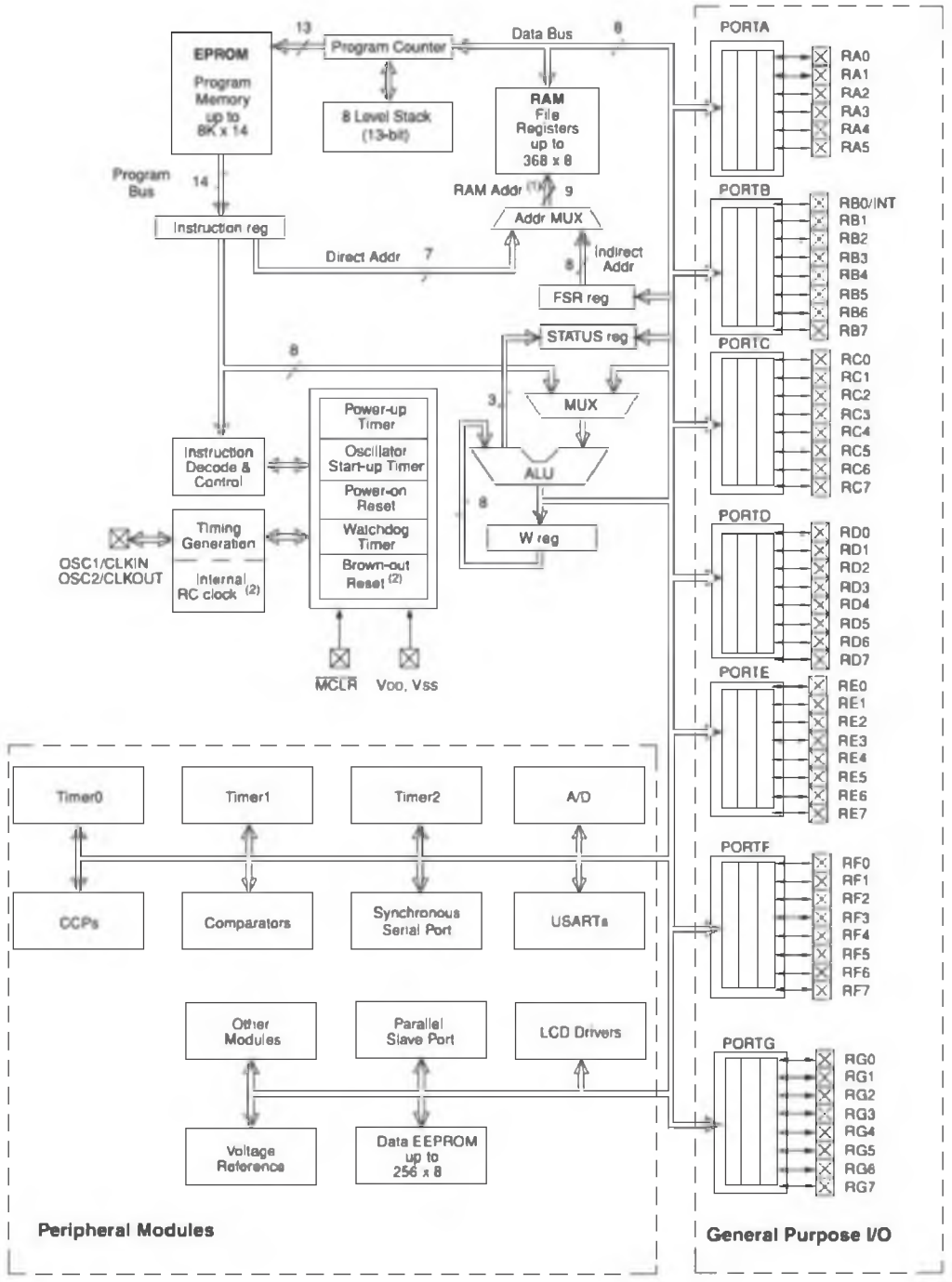

Figura 2. 4: Arquitectura de un microcontrolador PIC de 8 bits.

Fuente: (Verle, 2009)

2.4. Familia de los controladores ARDUINO

Arduino es una plataforma de creación de prototipos de código abierto fácil de usar tanto en hardware y software. Las placas Arduino (véase la figura 2.5) tienen la capacidad de leer las entradas y lo convierten en salida. Mediante el tablero se hace él envió de instrucciones al microcontrolador. Arduino se lo ha utilizado para muchos proyectos que pueden utilizar instrumentos científicos complejos y cotidianos, y esto va a seguir evolucionando a través del tiempo. Para dar las instrucciones al microcontrolador este utiliza un lenguaje C y el software Arduino.

Figura 2. 5: Familia Arduino.

Fuente: (Digital, 2015)

Arduino es una herramienta fácil de usar se la diseño para que pueda ser utilizada por estudiantes principiantes. Desde que se empezó a comercializar la placa ARDUINO se la empezó a adaptar a las necesidades que los usuarios tenían que variaban desde simples tablas de 8 bits a los productos que se utilizaban en el ámbito científico. Arduino es recomendable para todo tipo de proyectos, ya que, además de su bajo costo sus placas son de código libre, y esto permite que las personas puedan adecuarlo de acuerdo a lo que necesitan. Su software se está desarrollando continuamente debido a que como es abierto los usuarios de todo el mundo lo utilizan para proyectos científicos y esto hace que vaya abriéndose paso en múltiple áreas. (Arduino, s.f.).

Arduino puede ser ejecutado en Mac, Linux y Windows, en la educación se los profesores lo utilizan para el aprendizaje de programación y robótica, así como también en diseños científicos básicos de física o química, gracias a su bajo precio.

Para usuarios que están interesados en diferentes sistemas, este ofrece algunas ventajas, que son:

- Asequible – Es decir que Arduino es mucho más económico en relación a otras plataformas. Un ejemplo la versión más barata del módulo Arduino puede ser ensamblado a mano, e incluso los módulos premontados. (Arduino, s.f.)

- Multiplataforma – En general los microcontroladores se limitan solo a Windows, pero Arduino puede ejecutarse en diferentes sistemas como por ejemplo Linux, OS, etc. (Arduino, s.f.)
- Simple, entorno de programación clara – En la programación el IDE de Arduino es muy sencillo de utilizar y esta es una gran ventaja para maestros que enseñan a como programar, es más cómodo para los estudiantes ya que se familiarizan con su funcionamiento. (Arduino, s.f.)
- Código abierto y software extensible – Su software puede extenderse a través de bibliotecas como C++, los usuarios aprenden la parte técnica para poder utilizar diferentes herramientas para sus proyectos como agregar código AVR-C directamente en sus programas de Arduino. (Arduino, s.f.)
- Código abierto y hardware ampliable – Las personas que son diseñadores experimentados de circuitos, utilizan esta herramienta pueden crear una versión propia del módulo, la cual pueden mejorarla ampliarla o modificarla para así poder economizar costos y también algunos los hacen para entender su funcionamiento. (Arduino, s.f.)

2.5. Software de Programación

En el software se tiene una consola de texto, una barra de herramientas, un editor para texto, el área de mensajes y también una serie de menús que son fáciles de usar. Para poder comunicarse con estas herramientas se necesita cargar los programas que están en el hardware Arduino. En la figura 2.6 se muestra la ventana del software de programación Sketch de Arduino.

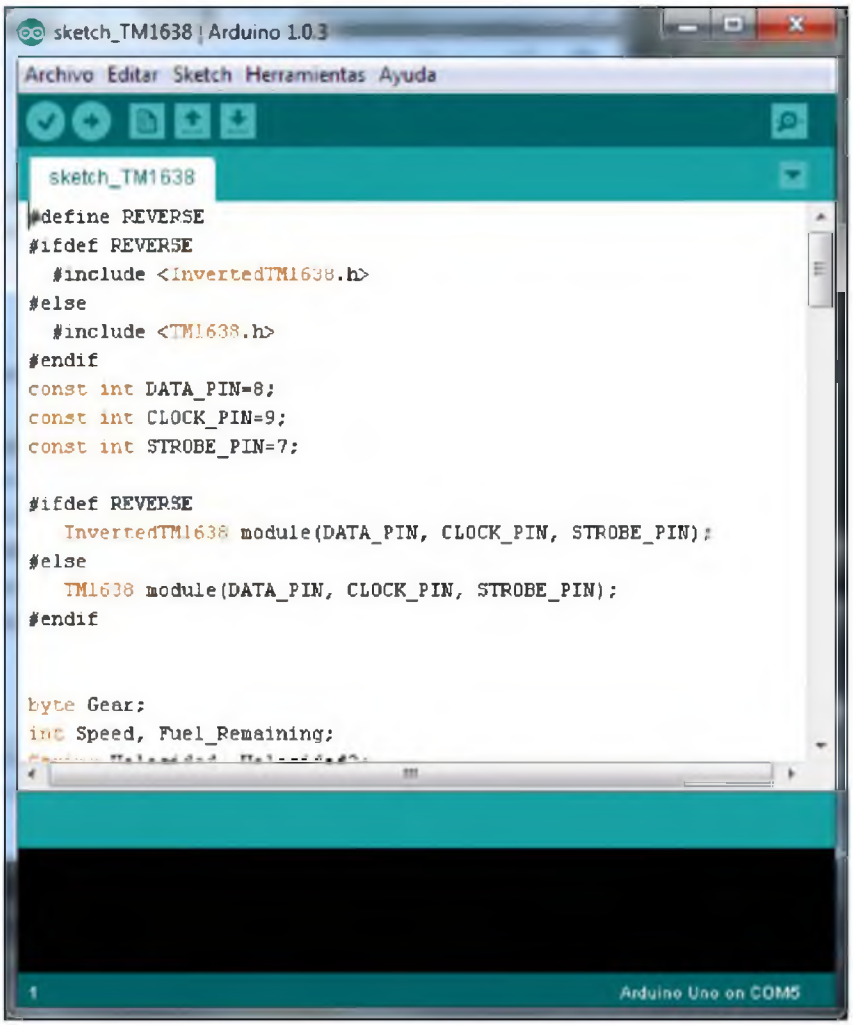

Figura 2. 6: Ventana de Programación Arduino.

Fuente: (Arduino, s.f.)

2.5.1 Apuntes de escritura

Los bocetos son programas escritos que utilizan arduino. Estos son dibujos que están escritos en el editor de texto y al guardarlos se quedan con la extensión de archivo .ino. En el editor se tiene varias funciones como cortar, pegar, buscar, etc. Los mensajes de error y demás datos se muestran en la salida de texto. Dentro de la ventana en la esquina inferior derecha se muestra el tablero configurado y puerto serie.

En los botones de la barra de herramientas existen funciones que permiten comprobar y cargar programas, crear, abrir y guardar bocetos, y abrir el monitor serie. Estas funciones son fáciles de utilizar ya que también tienen la información que ayuda a los usuarios a manejar cada una de estas opciones para poder llevar a cabo sus proyectos. A continuación, en la tabla2.2 podemos ver cuáles son y para qué sirven cada uno de estos botones.

	*Verificar* los controles su código de errores de compilación ella.
	*Subir* compila el código y lo sube a la placa configurada. Ver la posibilidad de subir los detalles a continuación. Nota: Si está utilizando un programador externo con su junta, puede mantener pulsada la tecla "Shift" en su ordenador cuando se utiliza este icono. El texto cambiará a "Subir el uso de programador"
	*Nuevo* Crea un nuevo dibujo.
	*Abrir* Presenta un menú con todos los bocetos en su cuaderno de dibujo. Al hacer clic en uno que se abrirá en la ventana actual sobrescribir su contenido. Nota: debido a un error en Java, este menú no se desplaza; si es necesario abrir un boceto al final de la lista, utilice el Archivo |Sketchbook menú en su lugar.
	*Guardar* Guarda el dibujo.
	*Serial Monitor* Abre el monitor serie .

Tabla 2. 2: Botones barra de herramienta.

Fuente. (Arduino, s.f.)

2.5.2 Comandos

A continuación en las tablas 2.3, 2.4, 2.5, 2.6 y 2.7 se detallan los comandos adicionales que se encuentran dentro del menu de herramientas: Archivo, Editar, Bosquejo, Herramientas, Ayuda. Estos menús son importantes para los proyectos que se llevan a cabo con esta herramienta.

MENU	DESCRIPCION
ARCHIVO	*Nueva: Crea una nueva instancia del editor, con la estructura mínima expresión de un boceto que ya están en su lugar.*
	Abrir *Permite cargar un archivo de la navegación boceto por las unidades y carpetas de la computadora.*
	Abrir reciente: Proporciona una breve lista de las más recientes bocetos, listo para ser abierto
	Sketchbook *Muestra los bocetos actuales dentro de la estructura de carpetas cuaderno de dibujo; al hacer clic en cualquier nombre se abre el dibujo correspondiente en una nueva instancia del editor.*
	Ejemplos: Cualquier ejemplo proporcionado por el software de Arduino (IDE) o la biblioteca aparece en este elemento de menú.
	Cerrar: Cierra la instancia del software de Arduino de la que se hace clic.
	Guardar: Guarda el dibujo con el nombre actual. Si el archivo no ha sido nombrado antes, un nombre se proporcionará en una ventana "Guardar como ...".
	Guardar como: Permite guardar el dibujo actual con un nombre diferente.
	Configurar página: Se muestra la ventana de configuración de página para la impresión.
	Imprimir: Envía el dibujo actual a la impresora de acuerdo con la configuración definida en Configuración de página.
	Preferencias: Abre la ventana de preferencias donde algunos ajustes de la IDE se pueden personalizar, como el idioma de la interfaz IDE.
	Salir: Cierra todas las ventanas del IDE. Los mismos dibujos se abren cuando fue elegido Quit se volverán a abrir automáticamente la próxima vez que inicie el IDE.

Tabla 2. 3: Menú ARCHIVO.

Fuente. (Arduino, s.f.)

MENU	DESCRIPCION
EDITAR	*Deshacer / Rehacer: Vuelve de uno o más pasos que hizo durante la edición; cuando vuelva, puede seguir adelante con Rehacer*
	Copiar: Duplica el texto seleccionado en el editor y lo coloca en el portapapeles.
	Copia para el foro: Copia el código de su boceto en el portapapeles en una forma adecuada para su publicación en el foro, con coloreado de sintaxis.
	Copiar como HTML: Copia el código de su boceto en el portapapeles como HTML, adecuado para incrustar en páginas web.
	Pega: Pone el contenido del portapapeles en la posición del cursor, en el editor
	Seleccione Todos: Selecciona y resalta todo el contenido del editor.
	Comentario / Descomente: Pone o quita el marcador de comentario // al principio de cada línea seleccionada.
	Aumentar / Reducir sangría: suma o resta un espacio al principio de cada línea seleccionada, moviendo el texto de un espacio a la derecha o la eliminación de un espacio al principio.
	Encuentra: Abre la ventana Buscar y reemplazar, donde puede especificar el texto a buscar en el interior del boceto actual de acuerdo con varias opciones
	Buscar siguiente: Resalta la siguiente aparición - en su caso - de la cadena especificada como el elemento de búsqueda en la ventana Buscar, en relación con la posición del cursor.
	Consulte más: destacado de la aparición anterior - en su caso - de la cadena especificada como el elemento de búsqueda en la ventana Buscar en relación con la posición del cursor.

Tabla 2. 4: Menú EDITAR

Fuente. (Arduino, s.f.)

MENU	DESCRIPCION
BOSQUEJO	*Verificar / Compilar: Comprueba su boceto de errores de compilación ella: se informará el uso de memoria para el código y las variables en el área de la consola.*
	Sube. Compila y carga el archivo binario en el tablero configurado a través del puerto configurado.
	Cargar mediante programador: Esto sobrescribirá el gestor de arranque en el tablero; tendrá que utilizar herramientas> Burn Bootloader para restaurarla y poder subir al puerto serie USB de nuevo.
	Compilado de exportación binario: Guarda un archivo hex que podrán ser conservados como archivo o se envía a la tarjeta usando otras herramientas.
	Mostrar Sketch Folder: Abre la carpeta de dibujos actual.
	Incluir Biblioteca: Añade una biblioteca para su dibujo mediante la inserción de instrucciones #include al comienzo de su código Desde este punto de menú puede acceder al Administrador de Biblioteca e importar nuevas bibliotecas de archivos zip.
	Agregar archivo: Añade un archivo de origen en el boceto El nuevo archivo aparece en una nueva pestaña en la ventana de dibujo. Los archivos pueden ser retirados del boceto usando el menú de la ficha accesible clic en el icono pequeño triángulo debajo del monitor serie uno en el lado derecho o la barra de herramientas.

Tabla 2. 5: Menú BOSQUEJO

Fuente. (Arduino, s.f.)

MENU	DESCRIPCION
HERRAMIENTAS	*Auto Format: Formatea el código muy bien: es decir, por lo que los guiones de apertura y cierre de llaves se alinean, y que las declaraciones dentro de llaves están más sangrado.*
	Archivo de croquis. Archives una copia del dibujo actual en formato zip. El archivo se coloca en el mismo directorio que el boceto.
	Fijar codificación y Recargar: Corrige posibles discrepancias entre el editor de mapas carbón codificación y otros sistemas operativos mapas de carbonilla
	Serial Monitor: Abre la ventana del monitor de serie e inicia el intercambio de datos con cualquier placa conectada en el puerto seleccionado en ese momento. Esto por lo general se restablece el tablero, si la placa es compatible Restablecer sobre la abertura del puerto serie.
	Junta. Seleccione la junta que está utilizando. Véase más abajo para las descripciones de las distintas juntas
	Puerto: Este menú contiene todos los dispositivos serie (real o virtual) en su máquina. Se debe actualizar de forma automática cada vez que se abre el menú de herramientas de alto nivel.
	Programador: Para seleccionar un programador hardware al programar una junta o chip y no usar la conexión USB-serie a bordo. Normalmente no necesitará esto, pero si usted está quemando un gestor de arranque de un nuevo microcontrolador, que va a utilizar esto.
	Burn Bootloader: Los elementos de este menú le permiten grabar un gestor de arranque en el microcontrolador en una placa Arduino. Este comando también establece los fusibles adecuados.

Tabla 2. 6: Menú HERRAMIENTAS

Fuente. (Arduino, s.f.)

MENU	DESCRIPCION
AYUDA	*Encuentra en la Referencia:* Esta es la única función interactiva del menú Ayuda: selecciona directamente la página correspondiente en la copia local de la referencia para la función o comando bajo el cursor.
	Bloc de dibujo: El software de Arduino (IDE) utiliza el concepto de un cuaderno de dibujo: un lugar estándar para almacenar los programas (o esquemas). La primera vez que ejecute el software de Arduino, se creará automáticamente un directorio para su cuaderno de dibujo. Puede ver o cambiar la ubicación de la ubicación del cuaderno de dibujo en Preferencias de diálogo. Es posible que todavía .pde abierto llamado archivos en la versión 1.0 y posteriores, el software cambiará automáticamente el nombre de la extensión a .ino.
	Pestañas, varios archivos, y Compilación: Le permite administrar bocetos con más de un archivo (cada uno de los cuales aparece en su propia pestaña). Estos pueden ser normales archivos de código de Arduino (sin extensión visible), archivos de C (extensión .c), archivos de C ++ (.cpp), o archivos de cabecera (.h).
	Subiendo: Antes de subir el boceto, es necesario seleccionar los elementos correctos de la Herramientas> Junta y Herramientas> Puerto menús. Cuando se carga un boceto, que está utilizando el Arduino gestor de arranque, un pequeño programa que se ha cargado en el microcontrolador en su tablero. Se le permite cargar código sin necesidad de utilizar ningún hardware adicional.
	Bibliotecas: Las bibliotecas proporcionan funcionalidad adicional para su uso en bocetos, por ejemplo, trabajar con el hardware o manipular los datos. Para utilizar una biblioteca en un boceto, selecciónelo de la Boceto> Importar biblioteca menú. Esto insertará una o más # include declaraciones en la parte superior del boceto y compile la biblioteca con su dibujo. Debido a que las bibliotecas se cargan en el tablero con su dibujo, que aumentan la cantidad de espacio que ocupa. Si un boceto ya no necesita una biblioteca, sólo tiene que borrar sus # include declaraciones de la parte superior de su código.
	Hardware de terceros: Soporte para hardware de terceros puede ser añadido a la ferretería de su directorio de bloc de dibujo. Plataformas instaladas puede incluir definiciones de mesa (que aparecen en el menú bordo), bibliotecas del núcleo, bootloaders, y las definiciones de programador. Para instalar, crear el hardware de directorio, descomprimir la plataforma de terceros en su propio subdirectorio. (No utilice "Arduino" como nombre de subdirectorio o tendrá que reemplazar la plataforma integrada de Arduino.) Para desinstalar, sólo tiene que borrar su directorio.
	Preferencias: Algunas preferencias se pueden configurar en el diálogo de preferencias. El resto se puede encontrar en el archivo de preferencias, cuya ubicación se muestra en el diálogo de preferencias.

Tabla 2. 7: Menú AYUDA

Fuente. (Arduino, s.f.)

2.5.3 Ayuda de idioma

Desde la versión 1.0.1, el software de Arduino (IDE) ha sido traducido a más de 30 idiomas diferentes. Por defecto, las cargas de IDE en el idioma seleccionado por el sistema operativo, tal como se muestra en la figura 2.7. (Nota: en Windows y Linux, posiblemente, esto está determinado por la configuración local que controla los formatos de idioma y fecha, y no por el idioma del sistema operativo)

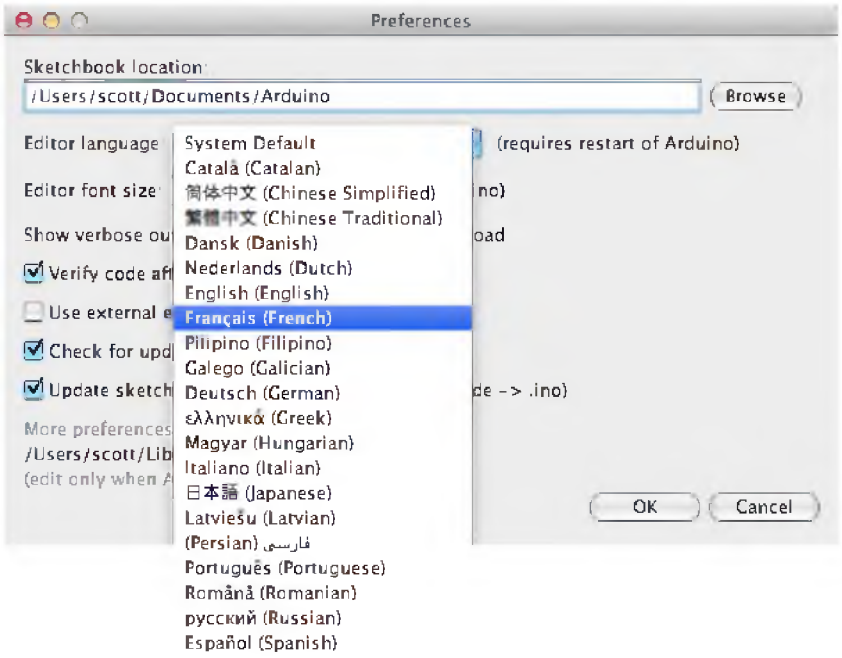

Figura 2. 7: Menú de Idiomas. Software Arduino.

Fuente. (Arduino, s.f.)

Si desea cambiar el idioma manualmente, inicie el software de Arduino y abra el Preferencias ventana. Al lado del Editor de idiomas hay un menú desplegable de idiomas soportados actualmente. Seleccione su idioma preferido en el menú, y reiniciar el software para utilizar el idioma seleccionado. Si no se admite el idioma del sistema operativo, el software de Arduino (IDE) será por defecto Inglés.

Puede devolver el software a su configuración predeterminada de elegir su lenguaje basado en su sistema operativo mediante la selección predeterminada del sistema desde el Editor de idiomas desplegable. Esta configuración se aplicará cuando se reinicia el software de Arduino (IDE). Del mismo modo, después de cambiar la configuración de su sistema operativo, debe reiniciar el software de Arduino (IDE) para actualizarlo al nuevo idioma por defecto.

2.5.4 Juntas

La selección placa tiene dos efectos: establece los parámetros empleados en la compilación y la carga de bocetos; y los conjuntos y la configuración del archivo y de fusibles utilizados por el comando quemadura gestor de arranque. Algunas de las definiciones de mesa sólo se diferencian en el segundo, por lo que incluso si usted ha estado cargando con éxito con una selección particular que usted querrá comprobar que antes de grabar el gestor de arranque.

Arduino Software (IDE) incluye el soporte incorporado, todos ellos basados en el AVR Core. El Administrador de Juntas incluido en la instalación estándar permite añadir soporte para el creciente número de nuevas placas basadas en diferentes núcleos como Arduino Due, Arduino Cero, Edison, Galileo y así sucesivamente.

Estas son algunas de las juntas en la que Arduino Software IDE incluye el soporte.

- *Arduino Yun*

  Un ATmega32u4 trabajando a 16 MHz con auto-reset, 12 Analog In, 20 de E / S digitales y 7 PWM.

- *Arduino / Genuino Uno*

  Un ATmega328 trabajando a 16 MHz con reajuste automático, 6 analógica, 14 de E / S digitales y 6 PWM.

- *Arduino Duemilanove o Diecimila w /* ATmega168

  Un ATmega168 trabajando a 16 MHz con auto-reset.

- *Arduino Nano w /* ATmega328

  Un ATmega328 trabajando a 16 MHz con auto-reset. Tiene ocho entradas analógicas.

- *Arduino / Genuino 2560 mega*

  Un Atmega2560 trabajando a 16 MHz con auto-reset, 16 Analog In, 54 E / S digitales y 15 PWM.

- *Arduino Mega*

  Un ATmega1280 trabajando a 16 MHz con auto-reset, 16 Analog In, 54 E / S digitales y 15 PWM.

- *Arduino Mega ADK*

  Un Atmega2560 trabajando a 16 MHz con auto-reset, 16 Analog In, 54 E / S digitales y 15 PWM.

- *Arduino Leonardo*

  Un ATmega32u4 trabajando a 16 MHz con auto-reset, 12 Analog In, 20 de E / S digitales y 7 PWM.

- *Arduino Micro*

  Un ATmega32u4 trabajando a 16 MHz con auto-reset, 12 Analog In, 20 de E / S digitales y 7 PWM.

- *Arduino Esplora*

  Un ATmega32u4 trabajando a 16 MHz con auto-reset.

- *Arduino Mini w /* ATmega328

  Un ATmega328 trabajando a 16 MHz con reajuste automático, 8 Analog In, 14 de E / S digitales y 6 PWM.

- *Arduino Ethernet*

  Equivalente a Arduino UNO con un escudo de Ethernet: Un ATmega328 trabajando a 16 MHz con reajuste automático, 6 analógica, 14 de E / S digitales y 6 PWM.

- *Arduino Fio*

  Un ATmega328 funcionando a 8 MHz con auto-reset. Equivalente al Arduino Pro o Pro Mini (3,3 V, 8 MHz) w /ATmega328 , 6 analógica, 14 de E / S digitales y 6 PWM.

- *Arduino BT w /* ATmega328

  ATmega328 trabajando a 16 MHz . El gestor de arranque quemado (4 KB) incluye códigos para inicializar el módulo de a bordo del bluetooth, 6 analógica, 14 de E / S digitales y 6 PWM ..

- LilyPad *Arduino* *USB*

  Un ATmega32u4 funcionando a 8 MHz con reajuste automático, 4 Analog In, 9 E / S digitales y 4 PWM.

- LilyPad *Arduino*

  Un ATmega168 o ATmega132 funcionando a 8 MHz con reajuste automático, 6 analógica, 14 de E / S digitales y 6 PWM.

- *Arduino Pro o Pro Mini (5 V, 16* MHz) *w /* ATmega328

  Un ATmega328 trabajando a 16 MHz con auto-reset. Equivalente al Arduino Duemilanove o Nano w / ATmega328; 6 analógica, 14 de E / S digitales y 6 PWM.

- *Arduino GN o más años w /* ATmega168

  Un ATmega168 trabajando a 16 MHz *sin* auto-reset. Compilación y carga es equivalente al Arduino Duemilanove o Diecimila w / ATmega168, pero el gestor de arranque quemado tiene un tiempo de espera más lento (y parpadea el pasado 13 LED tres veces al reiniciar); 6 analógica, 14 de E / S digitales y 6 PWM.

- *Control de Robot Arduino*

  Un ATmega328 trabajando a 16 MHz con auto-reset.

- *Robot Arduino Motor*

  Un ATmega328 trabajando a 16 MHz con auto-reset.

- *Arduino Gemma*

  Un ATtiny85 funcionando a 8 MHz con reajuste automático, 1 analógica, 3 E / S digitales y 2 PWM.

2.6. Robótica

La robótica ha estado presente en muchas áreas desde la parte industrial, investigación científica hasta hospitales y casas automatizadas desde ya 50 años. Esta área sigue en constante desarrollo y abarca distintas disciplinas como la electrónica, la física, la matemática, programación e inteligencia artificial.

La robótica estudia el desarrollo de sistemas mecánicos llamados robots manipuladores que son diseñados para distintas aplicaciones en diferentes campos científicos, comerciales, industriales y domésticos.

Los robots han tenido una buena aceptación en el área industrial, ya que se han convertido en un elemento primordial en el proceso de automatización industrial, estos han traído muchos beneficios a este campo como el incremento de la productividad, mejorar la calidad del producto y también una reducción de costos.

En el área domestica también han sido de gran ayuda, haciendo que muchos hogares sean automatizados y por esto hacer la vida cotidiana cómoda.

En la investigación científica se siguen desarrollando prototipos de robots que ayuden en la medicina, manufactura, espaciales, agricultura, juegos, etc.

Los robots también se han desarrollado para el campo universitario, en el cual ha incentivado la investigación y desarrollo de diferentes tipos de robots que cumplan distintas disciplinas como los robots humanoides, para batallas, para carreras, bailarines, de futbol, voladores, etc.

2.7. Competencias en concursos de Robótica

2.7.1. CER

Antecedentes:

En el año 2005 se realizó en el país el primer concurso nacional de robótica, CER por sus siglas, un evento que se realiza una vez por año en diferentes establecimientos de educación superior, en el que se reúnen estudiantes de varias universidades y escuelas politécnicas para participar en las categorías de robótica ya establecidas.

Cada año se van omitiendo o agregando más categorías al concurso, dependiendo mucho de la organización y gestión de aquello.

Entre ellas están:

- Batalla de robot
- Batalla de robot simulada
- Mini batalla de robot
- Pelea de bípedos
- Creatividad Lego
- Robot transformer

- Robot trepador
- Robot acuático
- Robot laberinto
- Mini, micro, mega sumo y mega sumo RC
- Carrera de humanoides
- Robot bailarín
- Robot volador
- Programación industrial
- Robot futbol
- Seguidor de línea

Cada uno se realiza en un escenario diferente, aunque en ciertos casos se comparten dichos escenarios. Cada universidad se inscribe con su respectivo club de robótica encabezado por representantes que son docentes de sus instituciones. Cada grupo asume su gasto propio de viáticos, comida, hotel para los participantes, la universidad anfitriona únicamente proporciona el escenario, realización del evento, premios, entre otros.

En el año 2014 nosotros como estudiantes de la Universidad Católica de Santiago de Guayaquil, con el afán de representar dignamente a nuestra institución, y gracias al apoyo de catedráticos de nuestra facultad, tuvimos la iniciativa propia para participar en este evento que se lo realizo en la Universidad Nacional de Chimborazo, UNACH, en la ciudad de Riobamba los días 17, 18 y 19 de noviembre de ese año.

En dicho concurso que se realizó el año pasado, claramente no pudimos participar en todas las categorías ya que somos un grupo joven, pero si representamos y dejamos muy en alto el nombre de la institución para ser nuestra segunda participación. Las categorías en las que participamos fueron: Mega sumo, mini sumo, creatividad lego, robot laberinto, robot futbol, en las que mini sumo pudimos llegar a posicionarnos en un 4to lugar a nivel nacional, siendo la novedad para muchos de los clubes que han participado anteriormente.

En la reunión dada dentro de los directivos organizadores del X CER 2014, se anticipa la designación de la realización para el próximo año, así nuestra universidad fue elegida para este evento. Gracias a ello nuestra universidad captará estudiantes de distintos puntos del país para desarrollar una competencia académica, y compartir experiencias en el área correspondiente.

Luego de aquella experiencia realizada en la Universidad Nacional de Chimborazo, también pudimos participar en un concurso llamado UMEBOT realizado en la ciudad de Quito para la fecha del 5 de diciembre del 2014 en la Escuela Politécnica Nacional, participando en una nueva categoría, siendo este un evento de nivel internacional. En nuestro grupo pudimos desarrollar un nuevo robot para la categoría de seguidor de línea velocista para participar en el evento en Quito.

Recientemente el club de robótica Robofet participó en el torneo internacional realizado en Cali, Colombia, ya que pudo obtener una invitación por parte de aquella organización. Con la suficiente autogestión y apoyo de un docente de nuestra facultad, pudimos ir a este país para representar a nuestra universidad, siendo jóvenes aun en participaciones pudimos hacer posible que nuestro grupo se haga conocer y así también nuestra prestigiosa universidad.

OBJETIVOS DEL CONCURSO ECUATORIANO DE ROBOTICA:

General: Promover y fomentar la competitividad del concurso ecuatoriano de robótica entre los estudiantes de pregrado de las instituciones de educación superior del Ecuador que ven en el desarrollo de la robótica un recurso tecnológico en el campo doméstico e industrial.

Específicos:

Propiciar un espacio de discusión e investigación acerca de la robótica.

Demostrar los conocimientos fundamentales de los sistemas electrónicos digitales y de microcontroladores para el diseño y creación de robots.

Organizar a futuro concursos de robótica anuales.

2.7.2. ROBOT GAMES ZERO LATITUD

Se fundó el 5 de abril del 2014, es un torneo Internacional de Robótica celebrado en Ecuador, en esta competencia se pone a prueba las capacidades en Micro Electrónica, Electrónica, Mecatrónica y Robótica,

aplicando programación en todo tipo de tarjeta de control, diseño y mecánica, es el evento más representativo a nivel Nacional e Internacional dando la oportunidad de darse a conocer los nuevos prototipos Ecuatorianos a nivel mundial.

Robot Games Zero Latitudº cuenta con 3 categorías generales, que a su vez se clasifica en las 24 categorías diferentes disponibles.

CATEGORÍA JUNIOR

1. Seguidor de Línea (Junior)
2. Libre con materiales reciclados (Junior)

CATEGORÍA MUJERES

3. Seguidor de Línea velocidad (Mujeres)
4. Mini Sumo (Mujeres)

CATEGORÍA TODO PÚBLICO

5. Seguidor de Línea Velocidad
6. Seguidor de Línea con Obstáculos
7. Natcar
8. Laberinto
9. Soccer
10. Hockey
11. Carrera de balance
12. Carrera de voladores

13. Nano Sumo

14. Micro Sumo

15. Mini Sumo

16. Mega Sumo

17. Sumo RC

18. Batalla 1Lb

19. Batalla 12Lb

20. Batalla 30Lb

21. Batalla 60Lb

22. Batalla 120Lb

23. Pelea de Humanoides

24. Impacto Tecnológico

2.7.3. UMEBOT

Es una competencia que se realiza en la Politécnica Nacional, su fin es incentivar el desarrollo de nueva tecnología en el país.

Sus categorías son:

1.- Seguidor de línea

2. Robot sumo

3. Robot batalla

4. Humanoides

5. Carrera de voladores

6. Robot insecto

7. Balacin

8. Robot soccer

9. Robot laberinto

10. Robot escalador

11. Categoría libre

2.8. Reglas generales para la categoría robot soccer

ROBOTS

La superficie máxima de un robot es de 18 centímetros cuadrados. La altura se limita a 15 cm sin o 21.5 cm con sistema de visión local. Los robots deben ser marcados por una pelota de ping-pong de color en la parte superior de ellos, de modo que "amigos y enemigos" pueden ser distinguidos. Todos los robots participantes tienen que encajar en un cilindro de 18 cm de diámetro. Para los robots rectangulares esto restringe su longitud diagonal de 18 cm.

Equipo

Un equipo formado por no más de 5 robots. Uno de ellos puede ser un encargado de la meta que se le permite retener el balón durante un periodo de tiempo de 15 segundos.

Campo

El campo tiene el tamaño de una mesa de ping pong. Está pintado de verde con paredes blancas y una serie de líneas blancas en él.

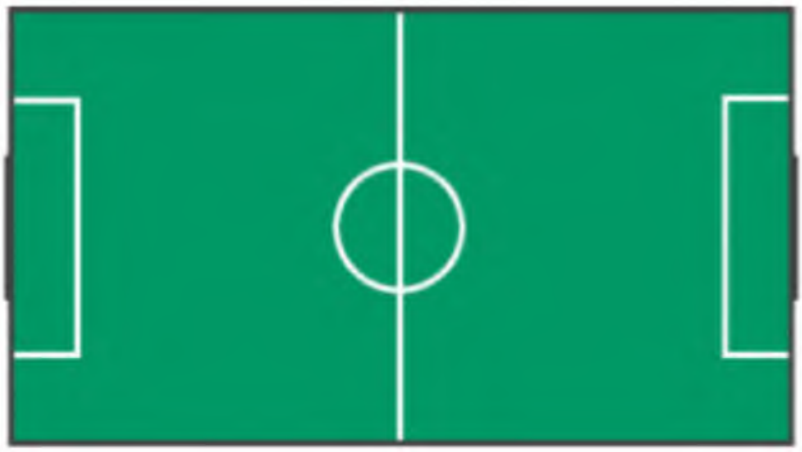

Figura 2. 8: Diseño de la cancha para robots soccer.

Fuente (*ROBOT SOCCER. Birgit Graf*)

Objetivos

Los objetivos son 50 cm de ancho, cada meta y el área del campo dentro de la portería está pintada en un color especial (uno azul y el otro amarillo).

Pelota

Se utiliza una pelota de ping pong.

Duración del Juego

El juego consiste en primer tiempo, descanso y la segunda mitad, teniendo cada uno de ellos de 10 minutos.

Sistema de comunicación

Se permite el uso de un sistema de comunicación inalámbrica que conecta robots y un ordenador o una aplicación situados fuera del campo.

Sistema de Visión Global

Se permite el uso de una cámara mundial en la parte superior del campo. Las discusiones actuales oscilan en torno a si cada equipo se le permitirá su propio sistema de cámara o se instalará una cámara central.

Faltas.

Faltas son llamados cuando un robot ataca intencionalmente a otro robot, cuando más de un robot de defensa entra en la zona de la defensa intencionadamente o cuando un robot trata de mantener la bola o a otro robot. Son el resultado de una patada o sin multas.

Realiza un tiro libre.

El balón será puesto en el más cercano de las posiciones de tiro libre (equivalentes a posiciones de la bola libre), se permite que un robot del equipo contra el cual se haya cometido la falta de pie directamente detrás de la pelota y juegue con ella.

Penal

El balón será puesto en la posición de tiro penal que se encuentra a una distancia de 45 cm del objetivo. Se permite a un robot del equipo contra el cual se haya cometido la falta de pie directamente detrás de la pelota y juega con él. Todos los otros robots, excepto el robot que jugará el balón y el portero contrario no se les permiten moverse hasta que el penal se ha jugado.

Saque de banda

Un tiro se realiza si la pelota se juega en las paredes del campo.

2.9. Mecánica de los Robots Soccer

El robot está equipado con dos ruedas motrices a cada lado del robot. Para evitar que el robot se vuelque hay dos ruedas de soporte que pueden moverse libremente en la parte delantera y posterior de la misma. Siete sensores binarios de proximidad de infrarrojos, así como un sistema de parachoques en forma de un tubo de silicio se colocan en todo el robot.

Un ejemplo de robot soccer puede ser el EyeBot II que se lo utilizo para la competencia RoboCup, esta competencia es solo para robots soccer. El objetivo principal de la competencia es que el juego sea lo más similar a un partido de fútbol humano.

Figura 2. 9: EyeBot II.

Fuente (*ROBOT SOCCER. Birgit Graf*)

Vehículo EyeBot II se construyó con el fin de mejorar el rendimiento del jugador de fútbol del robot y crear una plataforma para cumplir con las tareas más complejas que el vehículo EyeBot I. Para cumplir con las regulaciones de RoboCup para la pequeña liga robot de tamaño que el robot se ha reducido a un tamaño de 10 x 15 cm. Para cumplir con las restricciones de altura del tablero EyeBot está montada sobre el robot en un ángulo. Para atrapar la pelota de fútbol, la parte delantera de este robot está diseñado en una curva.

El tamaño de la zona curvada tubo que calcular fuera de la regla de que al menos dos tercios de superficie de la bola deben estar siempre fuera de la envolvente convexa alrededor del robot. Con el balón tiene un diámetro de aproximadamente 4,5 cm de la profundidad de esta curva se limita a 11 mm.

CAPÍTULO 3: DESARROLLO

3.1. Descripción de elementos del robot soccer

En la experimentación y aplicación de elementos para la construcción de robots para el concurso ecuatoriano de robótica CER 2015, es necesario el conocimiento de todos aquellos, como su funcionamiento y su estructura física para aplicarlos.

Entre los elementos que se utilizaron para la aplicación de este robot de soccer, se necesitaron los siguientes elementos que a continuación serán descritos uno a uno.

3.1.1. Puente H

El puente H, es un dispositivo electrónico que proporciona el control de la velocidad y sentido de giro de los motores, siendo controlado por una señal PWM a su entrada y este dependiendo de su valor puede hacer variar la velocidad y para cambiar su giro se cambia la polaridad del motor. En la figura 3.1 se muestra el circuito esquemático para puentes H.

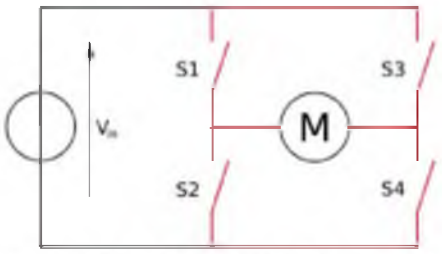

Figura 3. 1: Circuito de Puente H

Fuente. El Autor.

Este dispositivo electrónico se constituye básicamente por 4 interruptores que pueden ser mecánicos o por elementos como transistores o diodos, en la siguiente imagen demuestra un esquemático básico del funcionamiento, este si los interruptores S1 y S4 están cerrados, la dirección de la corriente hace girar el motor de un sentido, al contrario si S3 y S2 están cerrados cambia la dirección de la corriente.

El driver o puente H utilizado en el robot soccer es el modelo Tb6612fng (véase la figura 3.2), con un costo de 16 dólares en el mercado nacional, a continuación mostraremos la imagen correspondiente a este controlador.

Figura 3. 2: Puente H

Fuente. El Autor.

En la figura 3.3 se muestra el diagrama esquemático del puente H Tb6612fng. Este controlador, es conectado a los motores para entregar la corriente suficiente, y se conectan a los pines de la tarjeta Arduino Nano. Para las entradas AIN1 y AIN2 corresponden a la entrada del motor 1, y BIN1 y BIN2 corresponden a la entrada del segundo motor.

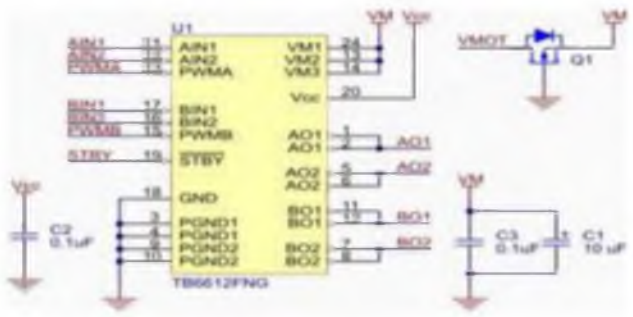

Figura 3. 3: Esquemático de Puente H

Fuente. El Autor.

3.1.2. Modulo Bluetooth HC-05

El modulo que se utilizó para la comunicación entre el robot soccer y el dispositivo móvil con sistema Android fue el HC-05 (véase la figura 3.4), que por sus características se decidió utilizarlo en nuestro proyecto.

Figura 3. 4: Modulo bluetooth HC-05

Fuente. El Autor.

El módulo HC-05, es fácil para usar el protocolo serial (SSP), diseñado para una conexión inalámbrica eficiente sin pérdida de datos. Esta tarjeta funciona en la banda de 2.4 GHz con una velocidad de 3 Mbps que es la

versión 2.0+EDR del bluetooth, utiliza tecnología CMOS y sus dimensiones son 12.7mm x 27mm.

3.1.3. Ruedas Omnidireccionales

Para un robot de estas características, es necesario que su movimiento en la cancha de soccer sea lo más eficaz posible, ya que en dicha competición los participantes deben realizar ciertos movimientos que es imposible realizarlos con ruedas comunes.

Para ello fue necesaria la adquisición de ruedas omnidireccionales importadas porque en nuestro país no se encuentra disponible en el mercado. A continuación presentaremos en la figura 3.6 lo correspondiente a este dispositivo físico que se adquirió para la aplicación del robot.

Figura 3. 5: Ruedas omnidireccionales

Fuente. El Autor.

Las medidas y diseño se presentaran a continuación en la figura 3.6

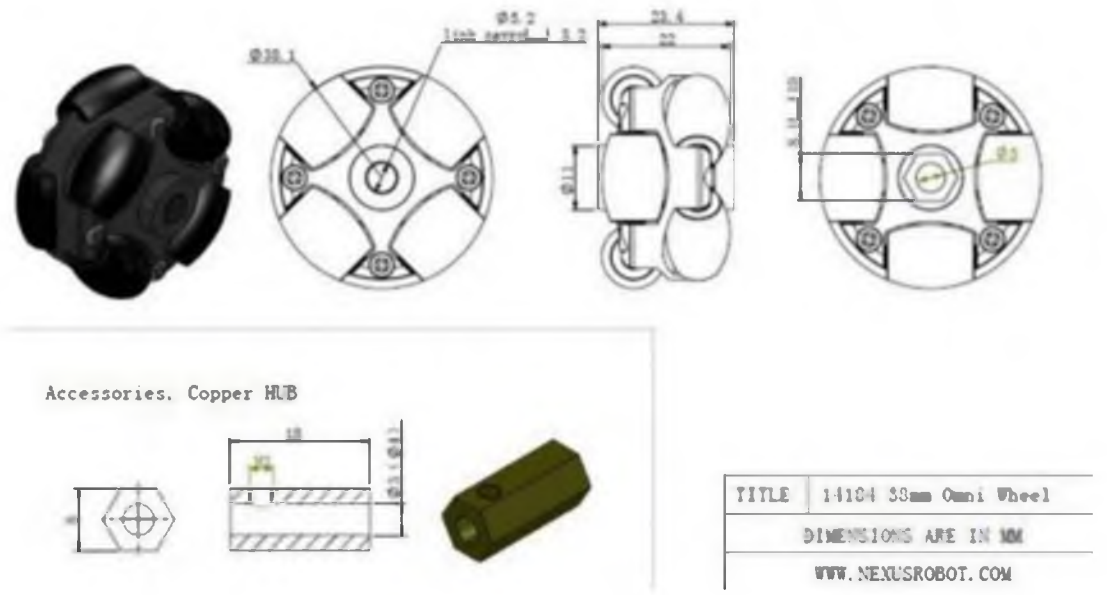

Figura 3. 6: Medida y Diseño de las Ruedas omnidireccionales.

Fuente. El Autor.

3.1.4. Batería Lipo

La batería de polímero de litio (LIPO), es uno de los componentes necesarios e importantes para el desarrollo y aplicación de cualquier robot. Hemos considerado tres grandes características de este tipo de batería para ser usado actualmente en los robots que se construyen.

1.- Estas baterías tienen gran capacidad de carga en un tamaño mínimo.

2.- Son muy ligeras y se encuentran en casi cualquier tipo de tamaño y forma.

3.- Este tipo de batería tienen una alta tasa de descarga, es decir, puede alimentar dispositivos que requieran mayor potencia en menor tiempo.

Figura 3. 7: Batería Lipo

Fuente. El Autor.

Las baterías Lipo se los pueden clasificar por tres grandes aspectos que poseen como son por la cantidad de celdas que se encuentran en una batería, su capacidad de carga, y la tasa de descarga. En la tabla 3.1 tenemos el número de celdas en la batería:

1 celda x 3.7 voltios (1S) = 3,7 voltios
2 celdas x 3,7 voltios (2S) = 7,4 voltios
3 celdas x 3,7 voltios (3S) = 11,1 voltios
4 celdas x 3,7 voltios (4S) = 14,8 voltios
5 celdas x 3,7 voltios (5S) = 18,5 voltios
6 celdas x 3,7 voltios (6S) = 22,2 voltios

Tabla 3. 1: Numero de Celdas

Fuente. El Autor.

3.1.5. Capacidad de carga:

Esta característica indica cuanta energía puede mantener en la batería y este viene dado en mAh, el cual es la medida en miliamperios que se puede mantener en un determinado tiempo.

Así por ejemplo, tenemos una batería de lipo de 1000 mAh, quiere decir que se puede descargar en una hora con una carga de 1000 miliamperio conectado a esta, si conectamos una carga menor puede durar mucho más tiempo esta carga.

3.1.6. La tasa de descarga (C):

La tasa de descarga "C" en una batería de Lipo es muy importante conocerla para saber su posible aplicación, dependiendo mucho este valor y su variación se puede utilizar en diferentes tipos de prototipos. Siendo esta característica directamente proporcional a la velocidad que entrega la mayor potencia en el menor tiempo posible, para muchos prototipos dependiendo su funcionalidad es donde se realiza la elección del tipo de batería a utilizar.

Por ejemplo, si se tiene un dispositivo volador (DRONE) esta característica deberá ser lo más alto posible para que este pueda funcionar con el mejor rendimiento, caso contrario en prototipos terrestres móviles no es necesario que su característica "C" sea muy elevado.

3.2. Diseño Esquemático

Para el diseño de nuestra tarjeta que estará montada en el robot soccer, utilizamos varias herramientas, entre ellas están Fritzing y Proteus. Utilizamos el programa Fritzing (véase la figura 3.8) para poder realizar un diseño esquemático previo, esta herramienta cuenta con una gran variedad de

componentes en sus librerías, facilitando así el uso su montaje. Entre los componentes que utilizamos son: Arduino, Driver TB6612, regulador de voltaje LM7805, Switch, Modulo Bluetooth HC-05, y partes necesarias como borneras, jumpers, entre otros.

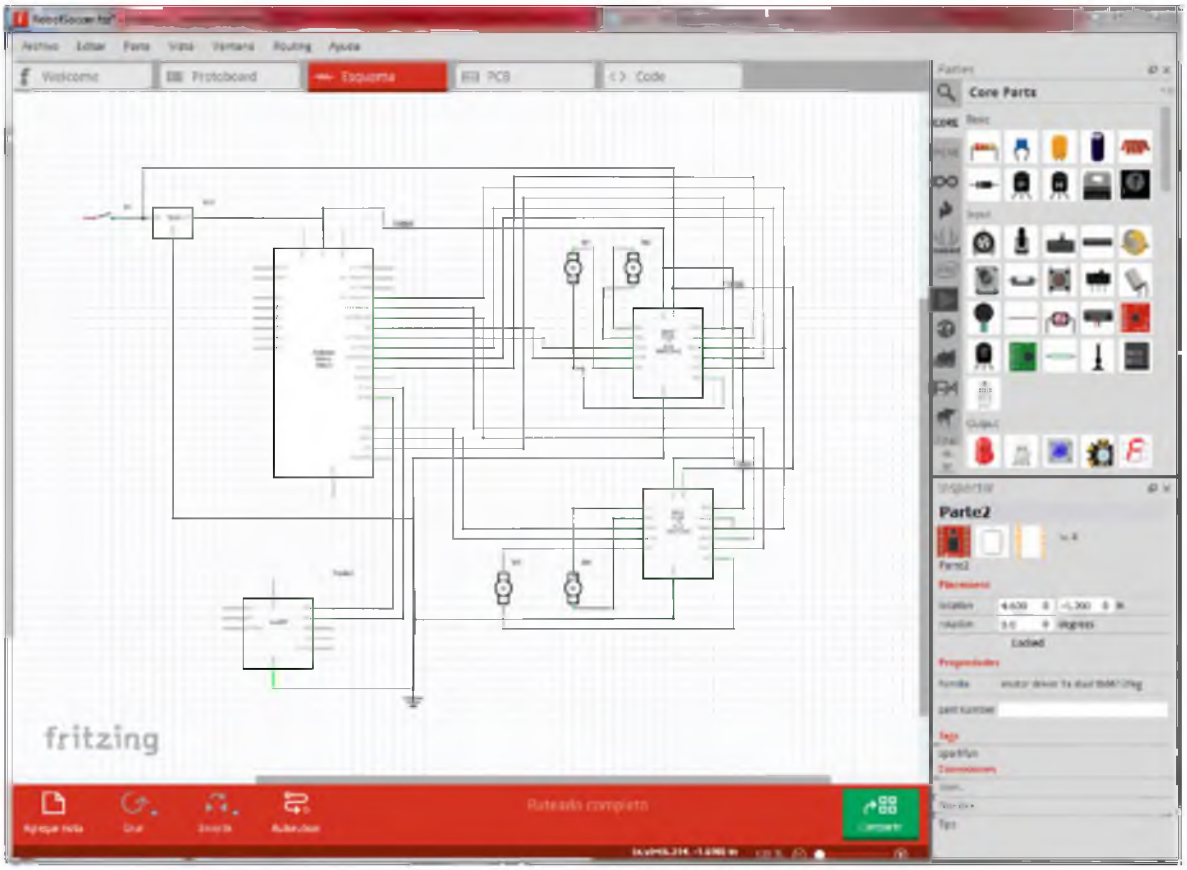

Figura 3. 8: Realización del diseño de la tarjeta.

Fuente. El Autor.

En la figura 3.9 se muestra el esquemático final realizado en la herramienta Fritzing, el motivo por el que se utilizó esta herramienta es por su facilidad de montaje y su gran variedad de componentes. El único componente que no pudimos acceder en la herramienta fue encontrar un módulo bluetooth HC-05, pero sin embargo para cuestiones de diseño utilizamos un

componente bluetooth genérico que representaría simplemente la comunicación entre este y la tarjeta arduino.

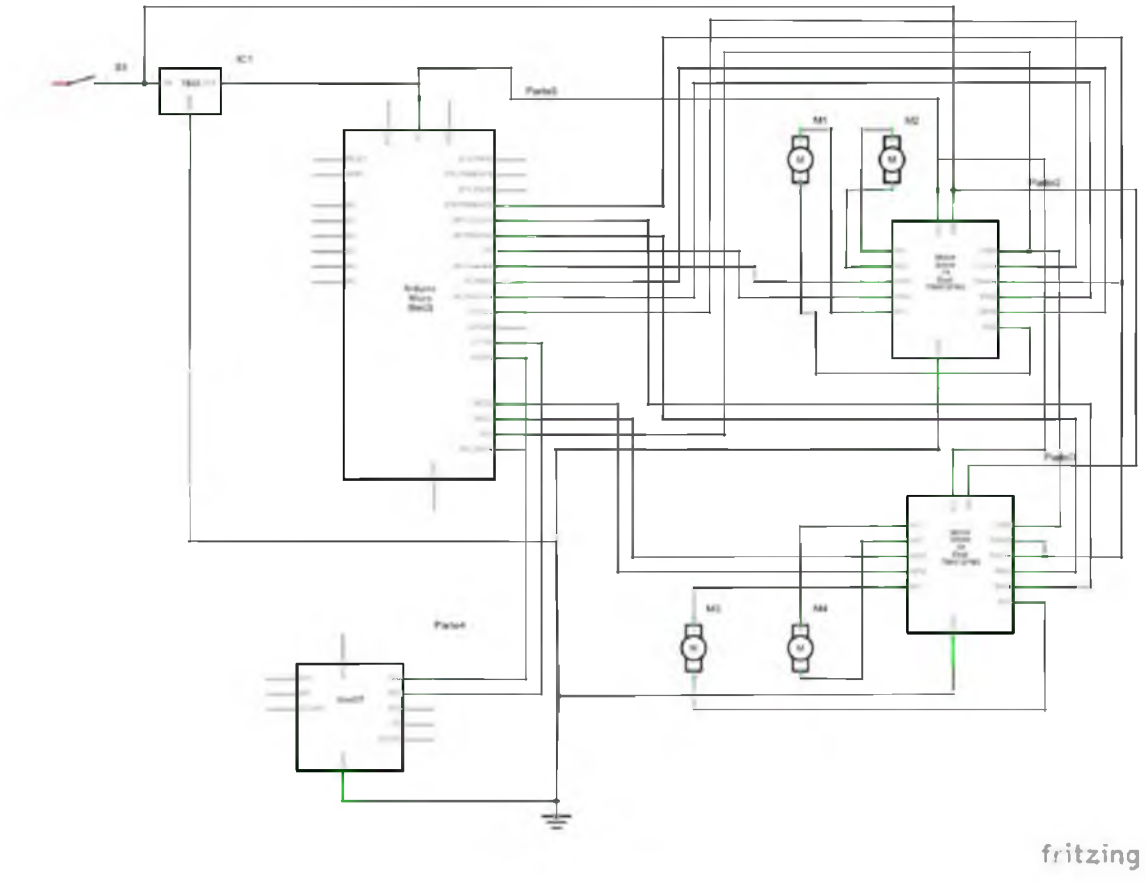

Figura 3. 9: Esquemático final de la tarjeta.

Fuente. El Autor.

3.3. Diseño PCB en Ares de Proteus.

Su pre diseño en Fritzing nos ayuda a tener una idea de cómo deben ir montados los componentes a la placa final, gracias a este pudimos diseñar en Proteus el PCB. En la figura 3.10 se muestra la ventana de trabajo de Ares de Proteus. Mientras que en la figura 3.11 se muestra el diseño final de la PCB.

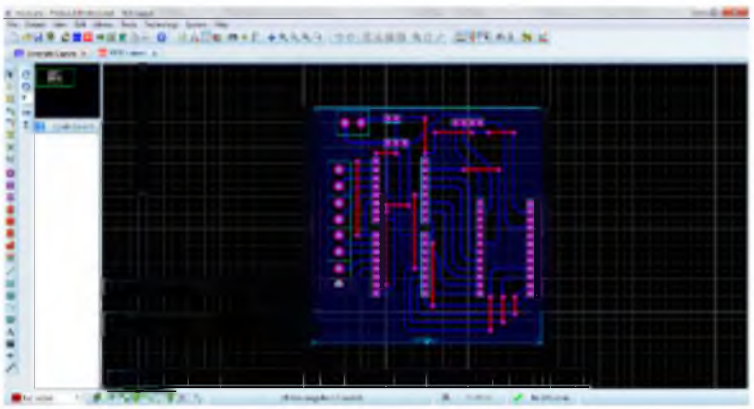

Figura 3. 10: Diseño de la placa PCB para robot soccer en Proteus.

Fuente. El Autor.

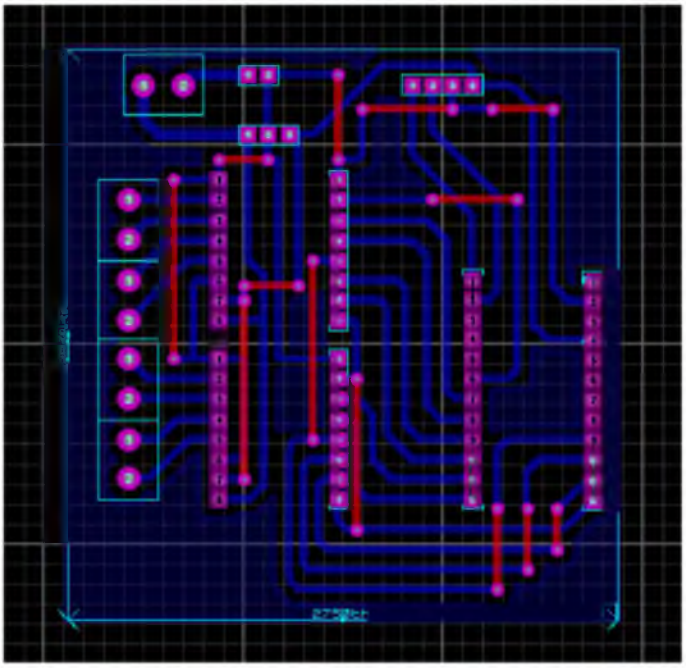

Figura 3. 11: Diseño final de la placa PCB del robot soccer

Fuente. El Autor.

Como se dijo anteriormente es difícil encontrar el componente Bluetooth HC-05 en ciertas librerías de los programas, así que la solución fue utilizar conectores de espadines hembra para su montaje así como para la

placa arduino y el regulador de voltaje LM7805. La borneras donde irán conectados los motores si fue fácil de encontrar, y esos se los coloco en el extremo izquierdo de la placa para facilitar su conexión y evitar que los cables tengan longitudes diferentes así también una mala distribución de estos dañando la estética del robot.

3.4. Programación del Robot Soccer.

3.4.1. Algoritmo de programación del Robot Soccer

Para la programación del robot soccer, utilizaremos el ide de arduino ya que en la parte física estamos utilizando una placa de desarrollo arduino micro pro. La explicación del funcionamiento del programa lo presentaremos por bloques, su estructuración y desarrollo.

La figura 3.12 nos demuestra el llamado de la librería que utilizaremos, este será "SoftwareSerial.h" que no es más que un conjunto de instrucciones que nos ayudarán a la comunicación entre el modulo bluetooth HC-05 y la placa Arduino Nano, este a su vez se conectará inalámbricamente a un dispositivo móvil con sistema operativo Android que posee una aplicación para el envío de caracteres. Además definiremos un retardo (DELAY) con el valor de 27, este se lo utilizara más adelante.

Otra de definiciones que haremos en el programa es asignar un tipo de variable a cada valor descrito en el mismo, así tenemos por ejemplo MA1 de tipo entero con valor de 16, motores de tipo char con valor de 120.

```
#include <SoftwareSerial.h>
#define DELAY 27
int MA1 = 16;
int MA2 = 14;
int MB1 = 9;
int MB2 = 8;
int MC1 = 6;
int MC2 = 7;
int MD1 = 5;
int MD2 = 4;
int pwm1 = 10; //MOTORES
int pwm2 = 3; //DISPARADOR
int enable = 15;
int vmotores = 120;
int vdisparador = 200;
char dato = 0;
char dataIn = 'S'; //Carácter o datos procedente del teléfono
char determinant; //Se utiliza en la función check, almacena el carácter recibido desde el teléfono.
char det; //Se utiliza en la función loop, almacena el carácter recibido desde el teléfono..
int velocity = 0; //Almacena la velocidad basada en el carácter enviado por el teléfono.
```

Figura 3. 12: Encabezado del programa para el robot soccer

Fuente. El Autor.

A continuación en la figura 3.13, podemos observar que escribimos la función “void setup”, en el que se definirán los puertos de entradas y salidas utilizando la sentencia “pinMode” y su respectiva descripción del puerto con su configuración, hacemos esto para todos los puertos que utilizaremos en el hardware construido previamente.

```
void setup() {
  Serial1.begin(9600); //Serial.begin(9600);
  pinMode(MA1, OUTPUT); pinMode(MA2, OUTPUT); pinMode(MB1, OUTPUT);
  pinMode(MB2, OUTPUT); pinMode(MC1, OUTPUT); pinMode(MC2, OUTPUT);
  pinMode(MD1, OUTPUT); pinMode(MD2, OUTPUT); pinMode(pwm1, OUTPUT);
  pinMode(pwm2, OUTPUT); pinMode(enable, OUTPUT);
  char det;
  analogWrite(pwm1, velocity); analogWrite(pwm2, vdisparador); digitalWrite(enable, HIGH);
}
```

Figura 3. 13: Descripción de la función void setup para la configuración de puertos de Arduino Nano.

Fuente. El Autor.

La función “void check” que se presenta en las figuras 3.14, 3.15, 3.16 y 3.17, es la que recibirá los datos por el puerto serial que está comunicándose vía bluetooth gracias al módulo HC-05 y dependiendo de su valor designara una función extra para efectuar los movimientos del robot soccer. En pocas palabras este es el que dará las ordenes, y estas órdenes están declaradas en la siguiente función que describiremos más delante de nuestra documentación.

```
void check() {
  analogWrite(pwm1, velocity); analogWrite(pwm2, vdisparador);

  if (Serial1.available())
  {
    dataIn = (char)Serial1.read();
    if (dataIn == 'F') {
    adelante();
    }
    else if (dataIn == 'B') {
      atras();
    }
    else if (dataIn == 'L') {
      izquierda();
    }
    else if (dataIn == 'R') {
      derecha();
    }
    else if (dataIn == 'S') {
      detener();
    }
```

Figura 3. 14: Primera parte de la descripción de la función “void check”

Fuente. El Autor.

Las sentencias condicionales son las que comparan los valores recibidos por el puerto serial y llamaran a la función respectiva. Por ejemplo en el primer “if” comparamos la variable “dataIn” si es igual a la letra ‘F’, si es

correcto se procederá a llamar a la función “adelante” caso contrario seguirá comparando hasta encontrar con la coincidencia respectiva.

```
else if (dataIn == '0') {
  velocity = 25;vdisparador=255;
}
else if (dataIn == '1') {
  velocity = 45;vdisparador=255;
}
else if (dataIn == '2') {
  velocity = 60;vdisparador=255;
}
else if (dataIn == '3') {
  velocity = 85;vdisparador=255;
}
else if (dataIn == '4') {
  velocity = 110;vdisparador=255;
}
else if (dataIn == '5') {
  velocity = 135;vdisparador=255;
}
```

Figura 3. 15: Segunda parte de la descripción de la función “void check”

Fuente. El Autor.

```
else if (dataIn == '6') {
  velocity = 140;vdisparador=230;
}
else if (dataIn == '7') {
  velocity = 165;vdisparador=250;
}
else if (dataIn == '8') {
  velocity = 180;vdisparador=250;
}
else if (dataIn == '9') {
  velocity = 200;vdisparador=250;
}
else if (dataIn == 'q') {
  velocity = 250;vdisparador=250;
}
else if (dataIn == 'U') {
 digitalWrite(MD1, LOW);digitalWrite(MD2, LOW);
}
else if (dataIn == 'u') {
digitalWrite(MD1, LOW); digitalWrite(MD2, LOW);
}
```

Figura 3. 16: Tercera parte de la descripción de la función “void check”

Fuente. El Autor.

```
    else if (dataIn == 'W') {
    disparadora();
    }
    else if (dataIn == 'w') {
    disparadora();
    }
    else if (dataIn == 'V') {
    disparadorb();
    }
    else if (dataIn == 'v') {
    disparadorb();
    }
    else if (dataIn == 'G') {
    izquierda1();
    }
    else if (dataIn == 'I') {
    derecha1();
    }
     else if (dataIn == 'J') {
    izquierda2();
    }
    else if (dataIn == 'H') {
    derecha2();
    }
  }
}
```

Figura 3. 17:Cuarta parte de la descripción de la función “void check”

Fuente. El Autor.

Todas las órdenes que asignan cada comparador “if” de las figuras anteriores no serían posible si no se desarrollan las funciones correspondientes, para ello escribimos el código de cada función como son “adelante”, “atrás”, “derecha”, “izquierda”, y demás como se podrán ver en las figuras 3.18 y 3.19. Cada función opera con la sentencia “digitalWrite” que en su sintaxis se describe con el pin del puerto a utilizar y su salida en alto o bajo (HIGH o LOW), gracias a que llevamos un esquema previo de nuestro robot

soccer podemos decirle que puerto accionar y cuales no según lo que deseemos.

Por ejemplo si queremos que el robot soccer se traslade en dirección hacia delante es necesario llamar a la función “adelante” y este accionara los motores que se encuentran uno paralelamente al otro y que se encuentran conectados en el puente H o drives correspondiente.

Vale recalcar que se escribió una función extra para un disparador B en caso de que en un momento se desea modificar el hardware y asignar otro motor.

```
void adelante() {
  digitalWrite(MA1, HIGH); digitalWrite(MA2, LOW);
  digitalWrite(MB1, HIGH); digitalWrite(MB2, LOW);
  digitalWrite(MC1, LOW); digitalWrite(MC2, LOW);
}

void atras() {
  digitalWrite(MA2, HIGH); digitalWrite(MA1, LOW);
  digitalWrite(MB2, HIGH); digitalWrite(MB1, LOW);
  digitalWrite(MC1, LOW); digitalWrite(MC2, LOW);
}

void derecha() {
  digitalWrite(MA1, LOW); digitalWrite(MA2, HIGH);
  digitalWrite(MB2, LOW); digitalWrite(MB1, HIGH);
  digitalWrite(MC2, LOW); digitalWrite(MC1, HIGH);
}

void izquierda() {
  digitalWrite(MA2, LOW); digitalWrite(MA1, HIGH);
  digitalWrite(MB1, LOW); digitalWrite(MB2, HIGH);
  digitalWrite(MC1, LOW); digitalWrite(MC2, HIGH);
}

void derecha1() {
  digitalWrite(MA2, LOW); digitalWrite(MA1, LOW);
  digitalWrite(MB2, LOW); digitalWrite(MB1, HIGH);
  digitalWrite(MC2, LOW); digitalWrite(MC1, HIGH);
}
```

Figura 3. 18: Funciones para movimientos “adelante”, “atrás”, “derecha”, “izquierda”, “derecha1”.

Fuente. El Autor.

```
void derecha2() {
  digitalWrite(MB2, LOW); digitalWrite(MB1, LOW);
  digitalWrite(MA1, LOW); digitalWrite(MA2, HIGH);
  digitalWrite(MC1, LOW); digitalWrite(MC2, HIGH);
}

void izquierda1() {
  digitalWrite(MB2, LOW); digitalWrite(MB1, LOW);
  digitalWrite(MA2, LOW); digitalWrite(MA1, HIGH);
 digitalWrite(MC2, LOW); digitalWrite(MC1, HIGH);
}

void izquierda2() {
  digitalWrite(MA2, LOW); digitalWrite(MA1, LOW);
  digitalWrite(MB1, LOW); digitalWrite(MB2, HIGH);
  digitalWrite(HC2, LOW); digitalWrite(MC1, HIGH);
}


void detener() {
  digitalWrite(MA1, LOW); digitalWrite(MA2, LOW);
  digitalWrite(MB1, LOW); digitalWrite(MB2, LOW);
  digitalWrite(MC1, LOW); digitalWrite(MC2, LOW);
}

void disparadora(){
  digitalWrite(MD1, HIGH); digitalWrite(MD2, LOW);
}
void disparadorb(){
  digitalWrite(MD2, HIGH); digitalWrite(MD1, LOW);
}
```

Figura 3. 19: Funciones para movimientos “derecha2”, “izquierda1”, “izquierda2”, “detener”, “disparadora”, “disparadorb”.

Fuente. El Autor.

Finalmente después de haber escrito todo el código necesario para el funcionamiento del robot soccer, es necesario declarar la función principal (void loop), que se muestra a continuación en la figura 3.20. Este simplemente dará como salida en alto a la variable declarada anteriormente “enable” para un led, es lo que hace la función digitalWrite que se encuentra en la primera línea dentro de la función void loop.

A continuación es necesario un llamar la sentencia while para que se repita la función check, el cual esta descrita anteriormente.

```
void loop()
{
   digitalWrite(enable, HIGH);
  while (true) {
    check();
  }
}
```

Figura 3. 20: Función principal del programa, Void Loop.

Fuente. El Autor.

CAPÍTULO 4: CONCLUSIONES Y RECOMENDACIONES.

4.1. Conclusiones.

Para el presente trabajo de titulación, hemos concluido lo siguiente:

El conocimiento del uso de microcontroladores es fundamental para la aplicación de cualquier tipo de robot y otras áreas como automatización y comunicaciones ya que consiste de un conjunto de componentes que dependiendo de su forma de configurar se pueden utilizar de diferentes maneras.

Utilizamos la placa de desarrollo de arduino ya que es muchísimo más accesible en sus costos y estos tienen ya implementado todo un sistema que permite funcionar al microcontrolador de manera eficiente ahorrando así para la persona que desee construir un robot tiempo y dinero.

La plataforma que utiliza la placa de desarrollo arduino para su programación y configuración posee su propio IDE, elegimos este ya que su programación es en lenguaje C y cuenta con un sinfín de librerías para muchos tipos de sensores y otros dispositivos, esto facilita en el aprendizaje y aplicación.

La estructura de nuestro robot soccer fue diseñado de tal manera que cumpla con los requisitos que solicitan en el reglamento de esta categoría, su

forma cilíndrica otorga una fuerte base para los componentes electrónicos y una gran movilidad en el campo de aplicación.

4.2. Recomendaciones.

Entre las recomendaciones para este trabajo de titulación presentamos que:

La construcción de robots para otras categorías que existen en las competencias nacionales e internacionales tales como Hockey, mini batalla de 1 y 10 libras, pueden ser aplicados a partir de ciertas modificaciones de nuestro diseño.

El uso eficiente de baterías de Lipo para este tipo de estructuras y diseño electrónico es necesario no disminuir el tamaño y potencia que entrega ya que disminuirá el tiempo de operación de nuestro robot en la competencia, tampoco es recomendado utilizar unas de mayor tamaño por su peso disminuirían así la movilidad del dispositivo.

REFERENCIAS BIBLIOGRÁFICAS

Arduino. (s.f.). *arduino*. Obtenido de www.arduino.cc: https://www.arduino.cc/en/Guide/Introduction

Digital, F. (20 de febrero de 2015). *Fabrica Digital*. Obtenido de http://fabricadigital.org/lesson/introduccion-a-arduino/

EngineersGarage. (12 de Diciembre de 2015). *EngineersGarage insiring creations*. Obtenido de http://www.engineersgarage.com/microcontroller

Kumar P., M., Das, B., & Das, G. (2013). An Overview of Microcontroller Unit: From Proper Selection To Specific Application. *International Journal of Soft Computing and Engineering (IJSCE)*, 228-231.

Mandado P, E., & Lopez M, E. (2007). *Microcontroladores PIC. Ssitema de Autoaprendizaje.* Barcelona.

Perez F, E. (2007). *Microcontroladores: fundamentos y aplicaciones con PIC.* Marcombo.

Verle, M. (2009). *PIC Microcontrollers - Programming in C.* MikroElektronika.

Presidencia de la República del Ecuador

Plan Nacional de Ciencia, Tecnología, Innovación y Saberes

SENESCYT
Secretaría Nacional de Educación Superior, Ciencia, Tecnología e Innovación

DECLARACIÓN Y AUTORIZACIÓN

Yo, **KARLA ERENIA JACOME GUERRERO**, con C.C: # 0201951431 autor/a del trabajo de titulación: **"Diseño e implementación de un robot móvil soccer utilizando la tarjeta ARDUINO NANO y controlado mediante bluetooth"** previo a la obtención del título de **INGENIERO EN TELECOMUNICACIONES** en la Universidad Católica de Santiago de Guayaquil.

1.- Declaro tener pleno conocimiento de la obligación que tienen las instituciones de educación superior, de conformidad con el Artículo 144 de la Ley Orgánica de Educación Superior, de entregar a la SENESCYT en formato digital una copia del referido trabajo de titulación para que sea integrado al Sistema Nacional de Información de la Educación Superior del Ecuador para su difusión pública respetando los derechos de autor.

2.- Autorizo a la SENESCYT a tener una copia del referido trabajo de titulación, con el propósito de generar un repositorio que democratice la información, respetando las políticas de propiedad intelectual vigentes.

Guayaquil, 14 de marzo de 2016

f. _____

Nombre: KARLA ERENIA JACOME GUERRERO.
C.C: 0201951431

Presidencia de la República del Ecuador

Plan Nacional de Ciencia, Tecnología, Innovación y Saberes

SENESCYT
Secretaría Nacional de Educación Superior, Ciencia, Tecnología e Innovación

REPOSITORIO NACIONAL EN CIENCIA Y TECNOLOGÍA

FICHA DE REGISTRO DE TESIS/TRABAJO DE TITULACIÓN

TÍTULO Y SUBTÍTULO:	Diseño e implementación de un robot móvil soccer utilizando la Tarjeta ARDUINO NANO y controlado mediante bluetooth.		
AUTOR(ES) (apellidos/nombres):	JACOME GUERRERO KARLA ERENIA		
REVISOR(ES)/TUTOR(ES) (apellidos/nombres):	PALACIOS MELENDEZ EDWIN FERNANDO		
INSTITUCIÓN:	Universidad Católica de Santiago de Guayaquil		
FACULTAD:	Facultad de Educación Técnica para el Desarrollo		
CARRERA:	Ingeniería en Telecomunicaciones		
TITULO OBTENIDO:	Ingeniero en Telecomunicaciones		
FECHA DE PUBLICACIÓN:	14 de marzo de 2016	**No. DE PÁGINAS:**	72
ÁREAS TEMÁTICAS:	Sistemas de Información, Desarrollo de Sistemas		
PALABRAS CLAVES/ KEYWORDS:	PROYECTO DE TITULACION, ROBOT SOCCER, ROBOTICA		

RESUMEN/ABSTRACT (150-250 palabras):
Durante los años de formación de Ingeniería en Telecomunicaciones, los estudiantes decidimos en el año 2014 incursionar en participaciones de Concursos de Robótica y formar el Club de Robótica "ROBOFET", en la que a través de proyectos de tutoría y de trabajos de titulación hemos incrementado las categorías para participaciones nacionales. El propósito del presente trabajo de titulación, es contribuir e incentivar a los estudiantes en el desarrollo de nuevos prototipos de robots móviles autónomos y controlados a través de dispositivos de comunicaciones inalámbricas, tales como Xbee y Bluetooth. Para el desarrollo del robot soccer se hizo una búsqueda de prototipos y en base a esta información, se logró realizar un diseño funcional, tanto en la parte microelectrónica y de la interfaz de comunicación, que en este proyecto se utiliza la tecnología Bluetooth. El comportamiento del robot soccer durante las pruebas realizadas, cumplió con las expectativas que se propusieron en este trabajo. No existieron inconvenientes en la comunicación mediante tecnología Bluetooth para controlar y operar el robot.

ADJUNTO PDF:	☒ SI	☐ NO
CONTACTO CON AUTOR/ES:	**Teléfono:** 0996913036	E-mail: ka_er92@hotmail.com
CONTACTO CON LA INSTITUCIÓN: COORDINADOR DEL PROCESO DE UTE	**Nombre:** Palacios Meléndez Edwin Fernando **Teléfono:** 0968366762 **E-mail:** edwin.palacios@cu.ucsg.edu.ec	

SECCIÓN PARA USO DE BIBLIOTECA	
Nº. DE REGISTRO (en base a datos):	
Nº. DE CLASIFICACIÓN:	
DIRECCIÓN URL (tesis en la web):	