INSTITUTO TECNOLOGICO DE TUXTLA GUTIERREZ

Reporte Final de Residencia Profesional

Control de un Robot Móvil usando un Celular con Tecnología Bluetooth

Ing. Francisco Ramón Sánchez Rodríguez

Julio César Zebadúa Vázquez
Miguel Jair Pedraza Toledo

Residencia Profesional Agosto- Diciembre 2011

ÍNDICE

INTRODUCCIÓN

A medida que la tecnología móvil evoluciona, se hace posible la implementación de dispositivos más sofisticados llegando a abarcar tantas aplicaciones o recursos que se hacen parte fundamental de nuestro trabajo y vida.

Hoy en día podemos encontrar dispositivos celulares que corren potentes sistemas operáticos, que integran sensores de posición, rotación, aceleración, táctiles, termómetros, así como módulos WIFI, Bluetooth, puertos USB, MODEM y cámaras de alta resolución. Se han convertido en pequeños computadores potentes y de reducido tamaño que ayudan a resolver y administrar nuestro trabajo de día a día, siendo dispositivos que muchas veces la limitante depende del uso al que deseemos darle.

El uso de robots industriales y comerciales junto con los sistemas de diseño asistidos por computadora (CAD), y los sistemas de fabricación asistidos por computadora (CAM), son la última tendencia.

La construcción de robots de batalla o de resolución de problemas ha permitido a los estudiantes de la escuela de Ingeniería Electrónica participar en los diferentes eventos de robótica a nivel nacional e internacional, de esta manera se desarrollan nuevas tendencias de diseño, mejores forma de procesamiento de variables y una evolución en la inteligencia del robot, así como generar nuevas aplicaciones que ayuden al ser humano en las distintas áreas donde pueden desempeñar un papel laboral o de apoyo.

El dispositivo Bluetooth posibilita la transmisión de voz y datos entre diferentes dispositivos mediante un enlace por radiofrecuencia segura y globalmente libre a una frecuencia de 2,4 a 2,8 GHz. Los principales objetivos que se pretenden conseguir con la implementación del dispositivo es conseguir una transmisión más fiable, por utilizar un total de 79 frecuencias con intervalos de un MHz, lo cual nos permite una mayor seguridad y robustez y de esta manera no tener interferencias con otros robot o sistemas que usan este mismo medio de comunicación, además de facilitar la sincronización con las etapas de servomecanismos eléctrico-electrónicas del robot.

El avance de las Telecomunicaciones nos permite la utilización del protocolo de banda base (canales simples por vía) el cual combina la conmutación de circuitos y de paquetes para asegurar que los paquetes no lleguen fuera de orden.

Hoy en día estamos inmersos en una serie de problemas tanto de carácter político-social como naturales, es por eso que la robótica busca ser una alternativa para resolver problemas o generar apoyo a ciertas situaciones que la humanidad vive. Un robot de apoyo a desastres o catástrofes es una forma de ayudar a la humanidad donde la integridad de las dos partes sea más segura.

ANTECEDENTES

Por siglos el ser humano ha construido máquinas que imitan las partes del cuerpo humano. Los antiguos egipcios unieron brazos mecánicos a las estatuas de sus dioses. Estos brazos fueron operados por sacerdotes, quienes clamaban que el movimiento de estos era inspiración de sus dioses igualmente, los griegos construyeron estatuas que operaban con sistemas hidráulicos, los cuales se utilizaban para fascinar a los adoradores de los templos.

Durante los siglos XVII y XVIII en Europa fueron construidos muñecos mecánicos muy ingeniosos que tenían algunas características de robots. Jacques de Vauncansos construyó varios títeres de tamaño humano a mediados del siglo XVIII. Esencialmente se trataban de robots mecánicos diseñados para un propósito específico: la diversión. En 1805, Henri Maillardert construyó una muñeca mecánica que era capaz de hacer dibujos, en la cual, una serie de levas eran el sistema principal de control para que esta pudiese escribir y dibujar. Hubo otras invenciones mecánicas durante la revolución industrial, muchas de las cuales estaban dirigidas al sector de la producción textil, entre ellas se puede citar la hiladora giratoria de Hargreaves (1770), la hiladora mecánica de Crompton (1779), el telar mecánico de Cartwright (1785), el telar de Jacquard (1801), entre otros[1].

El desarrollo en la tecnología, donde se incluyen las computadoras electrónicas, los actuadores de control retroalimentados, transmisión de potencia a través de engranajes, y la tecnología en sensores, han contribuido a facilitar la construcción de mecanismos autómatas para desempeñar tareas dentro de la industria. Son varios los factores que intervienen para que se desarrollaran los primeros robots en la década de los cincuenta; la investigación en inteligencia artificial desarrolló maneras de emular el procesamiento de información humana con computadoras electrónicas y proporcionó una variedad de mecanismos para probar sus teorías.

Una obra checoslovaca publicada en 1917 por Karel Kapek, denominada Rossum's Universal Robots, dio lugar al término robot. La palabra checa 'Robota' significa servidumbre o trabajador forzado, en el momento en que se tradujo al inglés se convirtió en el término "robot". Entre los escritores de ciencia ficción, Isaac Asimov contribuyó con varias narraciones relativas a robots, comenzó en 1939, a él se atribuye el acuñamiento del término Robótica. La imagen de robot que aparece en su obra es el de una máquina bien diseñada y con una seguridad garantizada que actúa de acuerdo con tres principios.

[1] LUI, Jiming, Wu, Jianbing. Multi-Agent Robotics System. United State of America. 2001. CRC.

Estos principios fueron denominados por Asimov las Tres Leyes de la Robótica, y son:

- Un robot no puede actuar contra un ser humano o, mediante la inacción, que un ser humano sufra daños.
- Un robot debe obedecer las órdenes dadas por los seres humanos, salvo que estén en conflictos con la primera ley.
- Un robot debe proteger su propia existencia, a no ser que esté en conflicto con las dos primeras leyes.

La Robótica es una ciencia aplicada, que surgió como tal hacia 1960. Han transcurrido pocos años y el interés que ha despertado desborda cualquier previsión; el auge de la Robótica y la imperante necesidad de su implantación en numerosas instalaciones industriales, requiere el concurso de un buen número de especialistas en la materia. Los robots utilizados en la industria hacia la década de los ochenta, se dedicaban al montaje y labores de inspección. En dicha época, la fabricación de automóviles llegaba al 58%[2] de la industria mundial, siguiendo en importancia las empresas constructoras de maquinaria eléctrica y electrónica. La incorporación al mundo del trabajo del robot, introduce el nuevo vocablo de "sistema de fabricación flexible", cuya principal característica consiste en la facilidad de adaptación de este a diferentes tareas de producción. Las células flexibles de producción se ajustan a necesidades del mercado y están constituidas básicamente por grupos de robots, controlados por un ordenador. Las células flexibles disminuyen el tiempo del ciclo de trabajo en la fabricación de un producto y liberan a las personas de trabajos desagradables y monótonos, lo que dará lugar a la fábrica totalmente automatizada.

FECHAS RELEVANTES[3]

- El inicio del control de robots se puede considerar asociado a la idea de Denavit y Hartenberg en 1955 de incluir sistemas de referencia solidarios a los distintos eslabones de una forma sistemática.

- El primer robot industrial se construyó en 1958 (J. F. Engelberger) y la primera patente de robots la obtuvo George Devol en 1961.

- Hasta 1965 los robots se controlaban en lazo abierto.

[2]RENTERIA, Arantxa, Robótica Industrial - Fundamentos y Aplicaciones. McGraw-Hill Interamericana Febrero 2001.
[3]NIKE B. Saeed, Introduction to Robotics Analysis, System, Applications. United State of America. 2001, 349p

- Hasta 1975 se hicieron avances en control mediante realimentación. El control más utilizado es el "Control independiente de las uniones" que se basa en suponer que las ecuaciones dinámicas están desacopladas y que las interacciones entre los eslabones no son más que perturbaciones o imperfecciones del modelo. Se exploró también el área de control mediante par calculado.

- Hasta 1993 los controles utilizados en los robots industriales se basan sobre todo en conceptos clásicos o en los controles antes mencionados. En entornos académicos empiezan a aparecer los primeros robots que utilizan los conceptos de control robusto y control adaptativo.

- Salvo contadas excepciones, los robots comerciales no han cumplido las expectativas que sobre ellos se tenían. De hecho, no es arriesgado decir que las capacidades de los robots en 1993 se parecían bastante a las que tenían en 1983. A finales del 1997 se podía decir que aún no se habían llegado a cumplir las expectativas fijadas en 1983 para 1988.

- En la actualidad, en ambientes de investigación, han evolucionado muchos conceptos de diseño mecánico tales como sistemas capaces de manipular objetos (herramientas con formas de mano), sensores (especialmente visión), estructuras muy ligeras o estructuras flexibles, sin embargo, aún no se puede decir que hayan llegado de forma masiva al robot comercial.

- Lo mismo ocurre con los conceptos de control, sobre todo en tareas que requieren una interacción compleja con el ambiente. Se puede decir que las tendencias actuales en control de robots incluyen el control robusto, control adaptativo, control híbrido fuerza/posición, robots autónomos (exploración espacial o movimientos en ambientes peligrosos), más avances en sistemas de visión artificial, robots con forma humana y capaces de andar (de los que ya hay las primeras versiones) y control basado en niveles de comportamiento.

La motivación para la elaboración de un sistema de control de un robot móvil por medio de un celular usando tecnología Bluetooth, surge por la necesidad de implementar nuevas formas de control, de corrección de algunos problemas y la necesidad de hacer sistemas más completos, eficientes y portables. Hoy en día el uso del celular funge como una herramienta necesaria para la humanidad, puesto que es un medio que independientemente de servir para comunicarnos, es utilizado como herramientas de trabajo, diversión y almacenamiento de datos.

La tecnología móvil posee tantos recursos que muchas veces no son usados a la par del potencial que estos pueden presentar, es por eso que este proyecto de residencia implementa el uso de algunos de esos recursos para el control de dispositivos robóticos móviles que pueden desempeñar numerosas aplicaciones para la humanidad.

La tecnología Bluetooth hoy presenta uno de los medio más usados para la comunicación, transferencia de datos o interconexión de diversos dispositivos, además que un 90% de la telefonía móvil incluye un módulo Bluetooth. De esta manera este puede ser una alternativa para el uso de esta tecnología, más allá del uso simple que presenta en nuestro celular.

Este tipo de sistema aporta muchas ventajas contra el uso de otros sistemas, puesto que el celular es un dispositivo compacto y portable, con diversidad de sensores y módulos integrados, que pueden compactar mucho un sistema y no tener una multitud de circuitos que controlen independientemente, de esta manera el celular es un medio actualizable y muchas veces universal que puede reducir costos, tamaño y facilidad de actualización de software cuando se requiera, puesto que una aplicación puede ser corrida en diversos modelos y tipos de celulares que le dan esa universalidad de no depender de hardware especifico.

Este proyecto de residencia sirve como generador de información y conocimiento para implementar control de sistemas móviles o de comunicación para otros sistemas usando la tecnología Bluetooth y un lenguaje de programación que se adapte al sistema operativo del celular.

JUSTIFICACIÓN

La Robótica es una ciencia que nos permite el desarrollar aplicaciones mediante la fusión de diferentes tecnologías como la electrónica, mecánica y telecomunicaciones entre otras.

El proyecto planteado tiene como fin el desarrollo de un robot móvil controlado mediante un dispositivo celular usando el protocolo Bluetooth, el cual nos permitirá poder realizar la implementación de las diferentes tecnologías para su aplicación en el área de la robótica y telemetría que servirá como base de introducción para el uso de esta tecnología.

Mediante el desarrollo del vehículo móvil nos podemos dar cuenta cómo avanza día a día la tecnología con lo que respecta a la programación, electrónica, telecomunicaciones, mecánica y la robótica, los mismos que combinan sus diferentes etapas para brindar un correcto funcionamiento y para esto nos valemos de los conceptos y laboratorios aprendidos en el transcurso de la carrera así como los recursos disponibles en Internet.

Mediante la realización del proyecto planteado se pretende lograr un óptimo desempeño en el control de las direcciones del móvil, además de servirles como base para futuras implementaciones de proyectos similares y afines a este. Con la utilización del dispositivo Bluetooth podremos reducir los problemas relacionados con la interferencia de las telecomunicaciones obteniendo una mayor eficiencia y fidelidad del robot.

La creación del robot móvil nos ayuda a entender el funcionamiento y adaptabilidad de nuestro sistema a un mecanismo al cual podemos asignarle una tarea específica y demostrar que la implementación de este método de control es eficiente para el manejo de dicho sistema.

Con esto se fomenta la creación de nuevos grupos que se sientan identificados con el desarrollo de aplicaciones similares en el campo de la robótica, también de la realización de investigaciones de nuevas aplicaciones y su presentación en los diversos usos a los que este se puede aplicar.

Además, este es un proyecto investigativo ya que nos ayuda a vincularnos y a conocer un poco más a fondo las diferentes tecnologías a utilizar.

OBJETIVOS

General

Desarrollar un sistema de control por medio de una aplicación instalada en un celular para el control de las direcciones de un robot móvil, a efecto de maximizar el uso de los recursos que estos dispositivos ofrecen.

Específicos

- Crear una interfaz gráfica para el control de movimiento del robot.
- Establecer comunicación entre el dispositivo móvil y el robot por medio de una red inalámbrica.
- Elaborar un sistema que permita el acceso al control del robot de forma fácil y rápida.
- Determinar el lenguaje de programación que se adapte a la necesidad y diseño de nuestro proyecto.
- Desarrollar el sistema utilizando las nuevas plataformas de mercado.
- Construcción de un robot móvil utilizando plataformas nuevas de desarrollo.

PLANTEAMIENTO DEL PROBLEMA

En la actualidad, los dispositivos móviles aportan diferentes ventajas: sirven como medio de comunicación, son portables, con mayores recursos, con acceso a la red y por último, es posible desarrollar aplicaciones de acuerdo a las necesidades de cada usuario y cargarlas en estos dispositivos.

En este concepto está fundamentado el presente proyecto de residencia profesional; el cuál, además de mostrar la forma en que pueden diversificarse las funcionalidades de los dispositivos móviles, expondrá las ventajas de compartir servicios en un ambiente distribuido de tal manera que estos se encuentren accesibles para cualquier tipo de cliente, sean inalámbricos o no.

El proyecto consiste en la implementación de un sistema donde se integren dispositivos móviles para controlar robots y demostrar que este tipo de sistemas cumple con los requerimientos y características que se necesitan para realizar este tipo de trabajos.

En la actualidad se observan métodos de control por medio de radiofrecuencia, en la cual son muy fáciles de recibir interferencias, aunado que si el robot posee sensores en los cuales se requiera más ancho de banda que el utilizado, se tengan que usar otros controles o receptores adicionales, de la misma forma, si el robot es controlado por medio de Internet, por medio de WIFI o un protocolo TCP/IP, existe la problemática de fallos de conexión al igual que de pérdidas por cobertura.

Los robots han avanzado para resolver problemas en las diferentes áreas, así como la forma en la que son controlados, pero muchos métodos presentan ventajas y desventajas dependiendo del área en la que operan, así como la necesidad de utilizar grandes equipos o diversos equipos para el control de éste, o al grado de requerir un computador.

La implementación de este método de control intenta ser una opción más en la diversidad de métodos, con las ventajas de mayor seguridad de recursos y con la intención de innovar al ir adoptando nuevas tecnologías para nuevos métodos de control de sistemas robóticos.

PROBLEMAS A RESOLVER

Para crear un sistema de control por medio de un teléfono celular, usando el protocolo Bluetooth, debemos resolver los siguientes problemas:

- Determinar los recursos que se necesitaran para realizar el proyecto.
- Obtener el alcance máximo que tendrá nuestro sistema y características del Bluetooth.
- Cotizar gastos para la fabricación de hardware y software.
- Obtención de recursos bibliográficos.
- Determinar las características del celular.
- Diseñar el robot móvil tomando en cuenta, forma, locomoción y dirección.
- Determinar el software que mejor se adapte a nuestras necesidades.

ALCANCES

Para un planteamiento claro de las áreas que abarca el proyecto de desarrollo y para definir con mayor detalle y precisión las diferentes capacidades que conformarán su funcionalidad, se han identificado los aspectos que serán tomados en cuenta en el diseño y desarrollo del mismo.

A continuación se presenta un listado de dichos aspectos, con el cual se describen los alcances del proyecto de desarrollo.

- Se usara un protocolo de comunicación Bluetooth para el envío de datos y recepción de éstos.
- Entorno gráfico para el usuario que permita el manejo de las acciones.
- Uso de la plataforma Arduino para la creación del robot móvil.
- Uso de la plataforma Android para la implementación del sistema de control.
- Generación de recursos informativos para la creación de nuevas plataformas.
- Nuevos conocimientos para el desarrollo de aplicaciones bajo estas plataformas.

LIMITACIONES

- El sistema será implementado para obtención de resultados informativos.
- El sistema estará limitado a trabajar bajo la plataforma Android.
- El diseño y construcción del hardware dependerá directamente de los recursos de la institución.
- Las modificaciones, así como la estructura del sistema, estará delimitada por la institución que requiere el proyecto.

PROYECCIÓN SOCIAL

Mediante el desarrollo del sistema piloto para el control de un robot móvil por medio de un celular usando Bluetooth, se pretende el uso de esta tecnología para sistemas más complejos o interacción con diversas aplicaciones en la cual utilizar los recursos que poseen los celulares para la migración de plataformas de baja y media tecnología a una de alta tecnología, la cual permita el desarrollo de nuestra sociedad.

Con el sistema de control por medio de un celular se logrará que áreas empresariales, médicas y militares opten por el uso de celulares como medio de trabajo, puedan contar con una herramienta económica y segura la cual les refleje el ahorro de otras herramientas y espacio.

Poder hacer sistemas telemétricos en los cuales, toda la información recabada por el celular sea envía por un medio GSM, sistemas embebidos en los cuales la información recabada en campo sea procesada y analizada dentro del celular y se envíe por un medio de comunicación.

Lograr que empresas tanto nacionales como internacionales ejecuten proyectos en cualquier tipo de rubro, teniendo como herramienta la tecnología de dispositivos móviles, la cual traerá grandes avances y beneficios dentro de ellas.

Así como también, conseguir que estudiantes, desarrolladores o cualquier persona interesada en este tipo de tecnología, puedan tomar como base de motivación este trabajo para poder desarrollar aplicaciones que implementen este tipo de tecnologías.

UNIDAD 1

MARCO TEÓRICO

1.1 Lenguajes De Programación

La generalización en los últimos años de teléfonos móviles, Smartphone, PDAs, etc. Ha generado una importante necesidad de aplicaciones para este tipo de dispositivos. Las prestaciones de los dispositivos móviles se incrementan día a día, posibilitando la implementación de aplicaciones muy interesantes.

Hoy en día estos dispositivos tienen la capacidad de portar cámaras, acceso a internet por medio de antenas WIFI, conexión Bluetooth, GPS, sensores de orientación, aceleradores 2D y 3D, envío de mensajes, sintonizadores de TV, reproducción de video y sonido, editores de texto, entre una gran cantidad de aplicaciones y recursos en los cuales pueden ampliar su utilidad más que para lo que principalmente fueron hechos.

Sin embargo el tamaño de estos y el ahorro energético llevan a usar tipos de procesadores y memorias de bajo consumo, frecuencia y velocidad de cómputo, también al procesar gráficos, sonidos y conexiones inalámbricas, entre otras cosas, consumen batería, esto la desgasta y hoy en día el tamaño y peso de los dispositivos es importante.

Los sistemas operativos y lenguajes de programación dependen mucho del tipo de móvil y compañía que los fabrica, es decir, existen una diversidad de S.O. Que dependen de las aplicaciones que realizará el dispositivo, mercado al que va dirigido, precio y soporte a desarrolladores, además, en ocasiones, las propias empresas fabricantes de dispositivos móviles, crean sus S.O. con referencia a un modelo, familia de dispositivo o adopta algún otro perteneciente a una empresa, de esta manera puede ser privado o de uso público.

En esta unidad hablaremos de los Sistemas Operativos actualmente más usados, así como de los lenguajes de programación que se utilizan para el desarrollo de aplicaciones, y dependerá del usuario así como de la aplicación final. El lenguaje de programación que decidamos tomar va relacionado con el S.O. de nuestro móvil, puesto que muchos restringen la posibilidad de correr algunos lenguajes.

1.1.1 java 2 Micro Edition (J2me)

- Java Catálogo de Productos

La tecnología Java es un entorno de programación orientado a objetos creado por Sun Microsystems ahora perteneciente a ORACLE. El objetivo principal era ofrecer un entorno para desarrollar aplicaciones independientes de la plataforma siguiendo la consigna "escribir una vez, ejecutar en cualquier lugar". Hoy en día Java es uno de los entorno de software más populares en desarrollo. Ofrece la oportunidad de crear aplicaciones empresariales de servidor y las computadoras de escritorio, así como aplicaciones para dispositivos pequeños y móviles. Para familiarizarse con la tecnología J2ME para el desarrollo de aplicaciones móviles, vamos a empezar con la exploración de la familia Java en general en primer lugar. Como se muestra en la Figura 1.1, Java se ha dividido en cuatro ediciones distintas:

- **J2EE** se destina principalmente para la construcción de aplicaciones distribuidas de la empresa con énfasis en el desarrollo del lado del servidor y aplicaciones web.
- **J2SE** es para escritorio convencional de desarrollo de aplicaciones (PC).
- **J2ME** es un subconjunto de J2SE destinadas a los dispositivos integrados que no puede apoyar una total implementación de J2SE.
- **Java Card** ofrece un entorno para desarrollar aplicaciones que se ejecutan en tarjetas inteligentes.

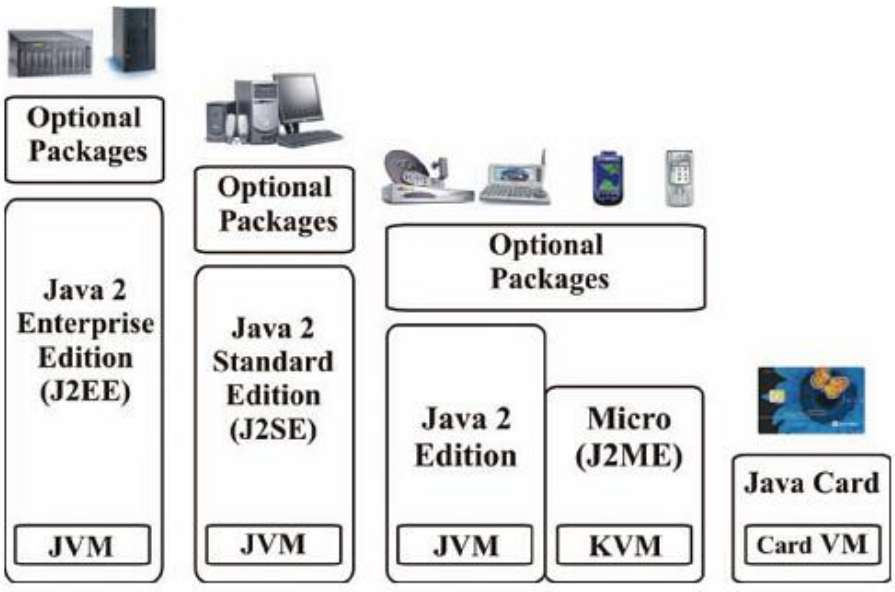


Figura 1.1 Arquitectura de la Familia JAVA

Esta división se ha hecho para ofrecer una solución para las diferentes necesidades y circunstancias. Sería ineficiente, por ejemplo, para usar una implementación completa de J2EE, que incluye paquetes adecuados para una aplicación de empresa, si solo una máquina de escritorio se utiliza. Además, la plena aplicación de J2SE no puede ser apoyada por un dispositivo de capacidad pequeña, que tiene menos recursos que una máquina de escritorio.

- Plataforma J2ME

J2ME es parte de la familia Java (Figura 1.1), la intención de construir aplicaciones que se ejecutan en plataformas de batería impulsado, como teléfonos móviles, PDA's y otros dispositivos integrados. Dado que todos estos dispositivos tienen diferentes capacidades en términos de memoria, el tiempo de procesamiento, el hardware y la pantalla, se han agrupado de acuerdo a sus fines y funciones en los grupos de dispositivos.

Dentro de cada configuración, los diferentes perfiles se crean con el fin de clasificar el tipo de dispositivo. J2ME no es un lenguaje nuevo, pero se ha adaptado la tecnología existente para el funcionamiento de Java en dispositivos embebidos, eliminando las partes de J2SE que no pueden correr y la adición de nuevas características que se necesitan. Sin embargo, la plataforma J2ME incluye una amplia gama de tipos de dispositivos: TV set-top box (SKY, DISH, MIGO, etc), TV por Internet, PDA's, Ebook Reader, teléfonos móviles, etc. Todos estos dispositivos tienen diferentes capacidades, tales como la interfaz de usuario, el presupuesto de la memoria y el poder de procesamiento. Por el momento dos configuraciones existen en la plataforma J2ME:

- CLDC, para dispositivos con:
  - Interfaz de usuario muy sencilla.
  - Bajo nivel de presupuesto de la memoria (160 Kb a 512 Kb) para Java.
  - Comunicación inalámbrica, dirigido a ancho de banda bajo.
  - 16 bits o procesadores de 32 bits
  - Potencia limitada, a menudo funcionamiento de la batería.
  - Ejemplos: teléfonos móviles, buscapersonas, PDAs nivel de entrada.

Estos dispositivos de aquí se conocen como dispositivos de menor capacidad.

- CDC, para dispositivos con:
  - Amplia gama de capacidades de interfaz de usuario.
  - Memoria del presupuesto en el rango de 16.2 MB para Java.
  - Conectividad a algún tipo de red.
  - 16 bits o 32 bit procesadores.
  - Ejemplos: televisores a Internet, TV set-top boxes, PDA.

En la Figura 1.2 mostraremos los dispositivos que ejecutan esta configuración de Java a medida y otros dispositivos capaces.

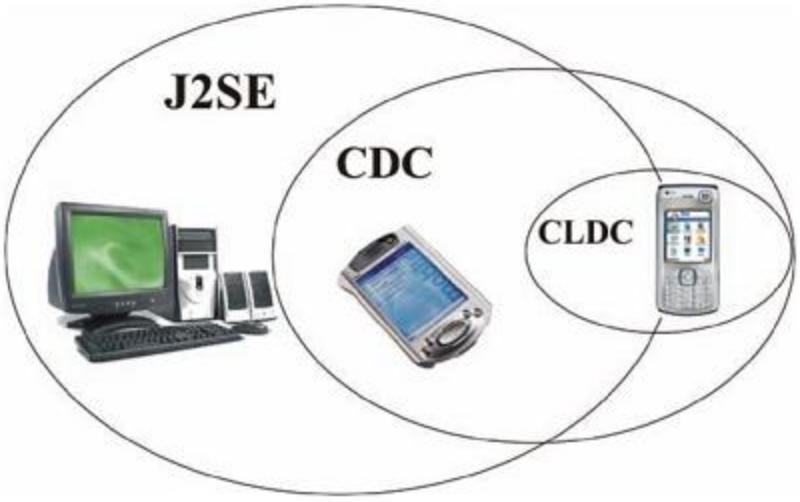


Figura 1.2

La relación entre CDC, CLDC, y J2SE se muestra en la Figura 1.2: todas las funcionalidades principales de los CDC y CLDC se heredan de J2SE. Además, todas las características implementadas en el CLDC se implementan en el CDC, con el fin de lograr la compatibilidad hacia arriba entre las dos configuraciones de J2ME. La plataforma J2ME se ejecuta en la parte superior del sistema operativo de los teléfonos móviles habilitados para Java y consta de dos versiones diferentes: una para los dispositivos más capaces y uno por menos dispositivos capaces, como se muestra en la Figura 1.3.

Cada uno de ellos incluye:

Configuración: Este es el conjunto básico de bibliotecas de clases y la máquina virtual que interpreta y ejecuta las aplicaciones. La máquina virtual puede ser un completo JVM o un subconjunto de este. Para CLDC y CDC las bibliotecas de clases son en su mayoría un subconjunto de la API de J2SE, además de las API para J2ME específicas tales como la formación bruta de capital.

Perfiles: Los perfiles se complementan con la configuración más específica de alto nivel de API's para un grupo de dispositivos. Las API's proporcionadas por un perfil que normalmente consiste en la aplicación del ciclo de vida del modelo, interfaz de usuario, el almacenamiento, red de apoyo específico, y mucho más. Hay que tener en cuenta que los perfiles se pueden superponer unos encima de otros. La configuración con uno o más perfiles crea un entorno J2ME entorno completo.

Paquetes opcionales: Existen API's que amplían los recursos de ejecución y control, tales como conectividad de base de datos, métodos de comunicación, gráficos, etc. Un paquete opcional de configuración puede ser independiente (es decir, el apoyo tanto de CLDC y CDC) o puede ser dirigido a una configuración específica.

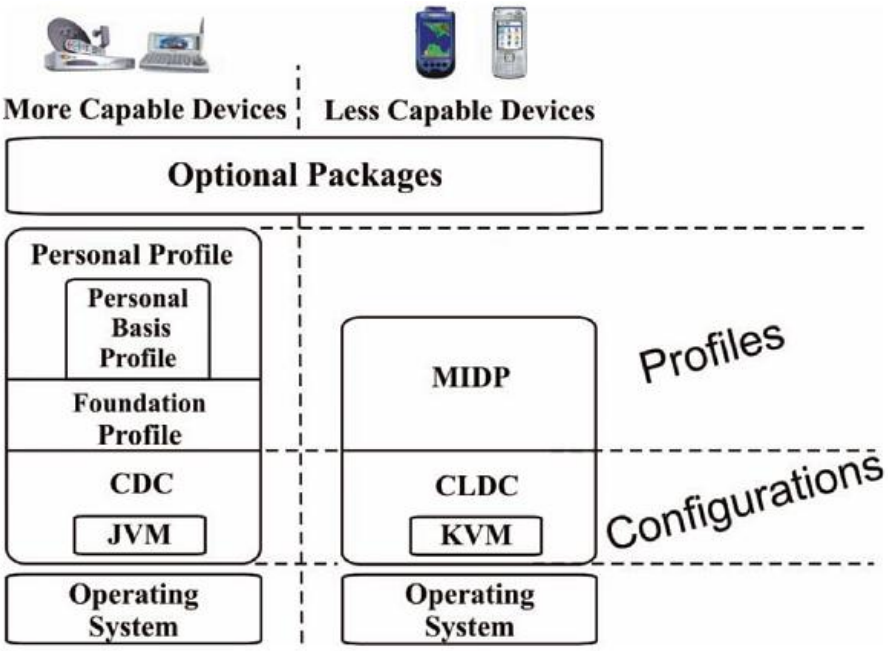


Figura 1.3

Desafortunadamente los paquetes opcionales pueden hacer el despliegue de la aplicación difícil, ya que podría terminar con una larga lista de paquetes opcionales para apoyar su solicitud.

1.1.2 Android

Android es un sistema operativo basado en GNU/Linux diseñado originalmente para dispositivos móviles, tales como teléfonos inteligentes, pero que posteriormente expandió su desarrollo para soportar otros dispositivos tales como tablets, reproductores MP3, netbooks, PCs, televisores, lectores de e-books, etc. Fue desarrollado inicialmente por Android Inc., una firma comprada por Google en 2005. Es el principal producto de la Open Handset Alliance, un conglomerado de fabricantes y desarrolladores de hardware, software y operadores de servicio. Las unidades vendidas de teléfonos inteligentes con Android se ubican en el primer puesto en los Estados Unidos, en el segundo y tercer trimestres de 2010, con una cuota de mercado de 43,6% en el tercer trimestre.

Android tiene una gran comunidad de desarrolladores escribiendo aplicaciones para extender la funcionalidad de los dispositivos. A la fecha, se han sobrepasado las 250.000 aplicaciones disponibles para la tienda de aplicaciones oficial de Android: Android Market, sin tener en cuenta aplicaciones de otras tiendas no oficiales para Android, como pueden ser la App Store de Amazon o la tienda de aplicaciones de Samsung Android Market, que es la tienda de aplicaciones en línea administrada por Google, aunque existe la posibilidad de obtener software externamente. Los programas están escritos en el lenguaje de programación Java. No obstante, no es un sistema operativo libre de malware, aunque la mayoría de ellos son descargados de sitios de terceros.

Android es una pila de software para dispositivos móviles que incluye las aplicaciones de un sistema operativo, middleware y la clave. El SDK de Android proporciona las herramientas y APIs necesarios para comenzar a desarrollar aplicaciones en la plataforma Android usando el lenguaje de programación Java.

Características:

- Marco de aplicación que permite la reutilización y sustitución de los componentes
- Máquina virtual optimizada para dispositivos móviles
- Navegador integrado basado en el código abierto WebKit motor
- Gráficos optimizados impulsado por una costumbre de la biblioteca de gráficos 2D, gráficos 3D basado en la especificación OpenGL ES 1.0 (aceleración de hardware opcional)
- SQLite para almacenamiento de datos estructurados
- Apoyo a los medios de audio comunes, videos y formatos de imagen (MPEG4, H.264, MP3, AAC, AMR, JPG, PNG, GIF)
- GSM de telefonía (dependiente del hardware)
- Bluetooth, EDGE, 3G y WiFi (dependiente del hardware)
- Cámara, GPS, brújula y acelerómetro (dependiente del hardware)
- Entorno de desarrollo completo que incluye un emulador de dispositivos, herramientas para la depuración, la memoria y de perfiles de rendimiento, y un plugin para el IDE de Eclipse.

1.1.3 App Inventor

Una aplicación proporcionada por Google que permite a cualquiera crear aplicaciones de software para el sistema operativo Android (OS) Se utiliza una interfaz gráfica, muy similar Scratch y la interfaz de usuario StarLogo TNG, que permite a los usuarios arrastrar y soltar objetos visuales para crear una aplicación que puede ejecutarse en el sistema Android, que funciona en muchos dispositivos móviles. La aplicación se puso a disposición a través de solicitud el 12 de julio de 2010, y luego publicada el 15 de diciembre de 2010.

Está hecha para personas no familiarizadas con la programación de computadoras. El razonamiento es que si los jóvenes desean entrar al desarrollo de aplicaciones para satisfacer sus propias necesidades e instalarlos en sus propios teléfonos, será más probable que el uso de los teléfonos sea con más frecuencia, o cambiar al sistema operativo Android si no están usando un teléfono que funciona con el sistema.

En la creación de App Inventor para Android, Google se basó en investigaciones previas significativas en informática educativa y el trabajo realizado dentro de Google en entornos de desarrollo en línea.

El editor de bloques utiliza la biblioteca de Open Blocks de Java para crear lenguajes de programación visual por bloques. Open Blocks se distribuye por el Massachusetts Institute of Technology Program 's Scheller para la creación de programas educacionales y deriva de la investigación de tesis de maestría por Ricarose Roque. El profesor Eric Klopfer y Daniel Wendel del Programa Scheller apoyó la distribución de Open Blocks bajo la licencia MIT. Open Blocks es programación visual y está estrechamente relacionado con el StarLogo TNG, un proyecto de la MIT Media Laboratory 's grupo Lifelong Kindergarten .

App Inventor del Editor de bloques

El compilador que traduce el lenguaje visual de bloques para la implementación de Android utiliza la Kawa lenguage Framework y la estructura del lenguaje Kawa, desarrollada por Per Bothner y distribuido como parte del sistema operativo GNU de la Free Software Foundation.

Google App Inventor se encuentra disponible para su uso por cualquier persona que tenga una cuenta de Google.

Para empezar a crear proyectos en App Inventor debemos configurar nuestro equipo:

Paso 1: Requisitos del sistema:

Ordenador y Sistema Operativo

- Macintosh (con procesador intel): Mac OS X 10.5, 10.6, 10.7
- Windows: Windows XP, Windows Vista, Windows 7
- GNU/Linux: Ubuntu 8+, Debian 5+

Navegador

- Mozilla Firefox 3.6 o superior
- Apple Safari 5.0 o superior
- Google Chrome 4.0 o superior
- Microsoft Internet Explorer 7 o superior

El equipo debe de ejecutar Java 6 (también conocido como Java 1.6) Puede descargar JAVA desde http://java.com/es/.

Paso 2: Instalación de librerías de App inventor

Windows XP, Vista, 7:

Recomendamos que realice la instalación desde una cuenta que tenga privilegios de administrador en su equipo. Esto instalará el software para todos los usuarios de la máquina. Si usted no tiene privilegios de administrador, la instalación debería funcionar, pero la aplicación de App inventor sólo podrá utilizarse a partir de la cuenta que se usó cuando instaló el software.

1. Descarga e instala este instalador: AppInventor_Setup_Installer_v_1_1.exe (~92 MB)

2. Si una vez instalado, empieza a utilizar Appinventor web y java le pregunta la ubicación del software es esta:

- 32 bits : C: \ Archivos de programa \ Appinventor \ comandos para la Appinventor.
- 64 bits : C: \ Archivos de programa (x86) \ Appinventor \ comandos para la Appinventor.

Mac OS X:

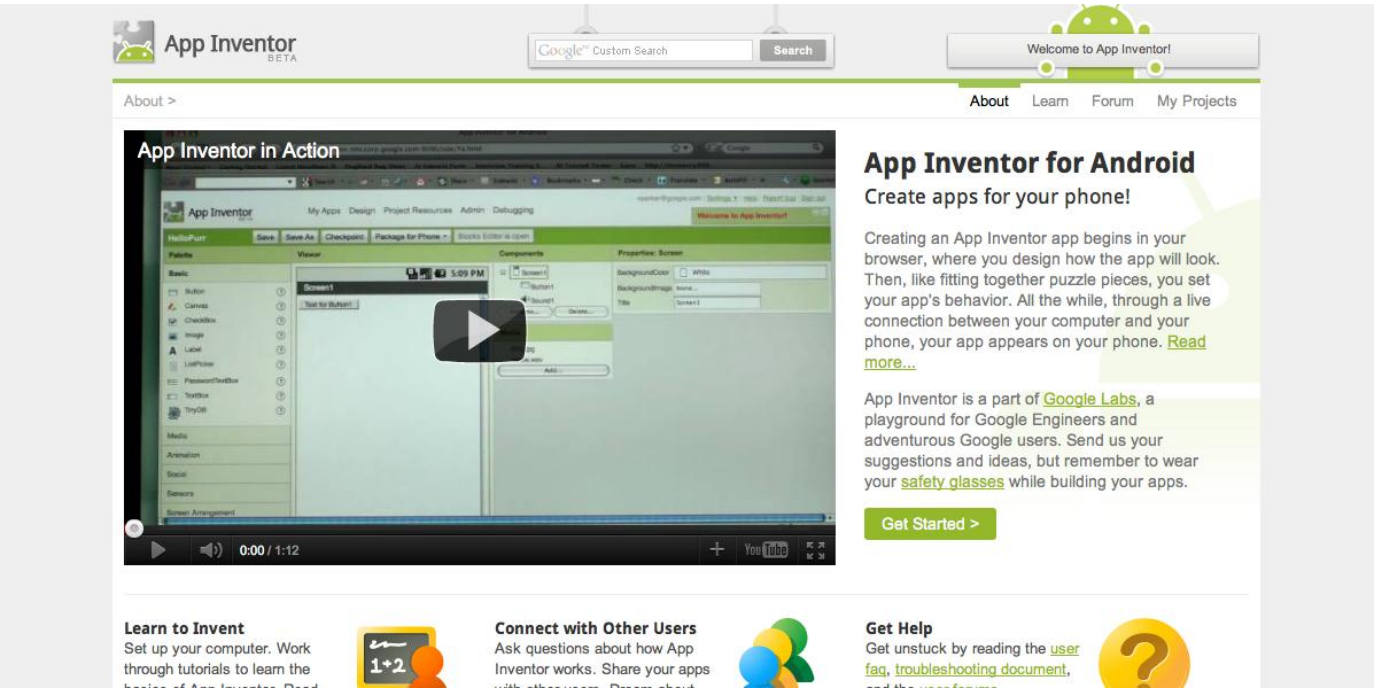
Descarga e instala este instalador: AppInventor_Setup_Installer_v_1_1.dmg (~92 MB)

GNU/Linux:

Descarga e instala este instalado: AppInventor_Setup_Installer_v_1_1_all.deb (~92 MB)
Si no puedes instalar el .deb descarga e instala este instalador:
AppInventor_Setup_Installer_v_1_1.tar.gz (~92 MB)

Paso 3: Empieza a utilizar App Inventor!!!

Teniendo una cuenta Google para poder iniciar automáticamente o crear una cuenta Google, dirígete al siguiente link: http://www.appinventorbeta.com/about/



1.2 Protocolo De Comunicación

1.2.1 Bluetooth

Bluetooth es una especificación industrial para Redes Inalámbricas de Área Personal (WPANs) que posibilita la transmisión de voz y datos entre diferentes dispositivos mediante un enlace por radiofrecuencia en la banda ISM de los 2,4 GHz. Los principales objetivos que se pretenden conseguir con esta norma son:

- Facilitar las comunicaciones entre equipos móviles y fijos...
- Eliminar cables y conectores entre éstos.
- Ofrecer la posibilidad de crear pequeñas redes inalámbricas y facilitar la sincronización de datos entre equipos personales.

Los dispositivos que con mayor frecuencia utilizan esta tecnología pertenecen a sectores de las telecomunicaciones y la informática personal, como PDA, teléfonos móviles, computadoras portátiles, ordenadores personales, impresoras o cámaras digitales.

Se denomina Bluetooth al protocolo de comunicaciones diseñado especialmente para dispositivos de bajo consumo, con una cobertura baja y basada en transceptores de bajo costo. Gracias a este protocolo, los dispositivos que lo implementan pueden comunicarse entre ellos cuando se encuentran dentro de su alcance. Las comunicaciones se realizan por radiofrecuencia de forma que los dispositivos no tienen que estar alineados y pueden incluso estar en habitaciones separadas si la potencia de transmisión lo permite.

Estos dispositivos se clasifican como "Clase 1", "Clase 2" o "Clase 3" en referencia a su potencia de transmisión, siendo totalmente compatibles los dispositivos de una clase con los de las otras. La tabla. I.I muestra las diferencias entre cada una de las clases de

Bluetooth:

Clase	Potencia máxima permitida (mW)	Potencia máxima permitida (dBm)	Rango (aproximado)
Clase 1	100 mW	20 dBm	~100 metros
Clase 2	2.5 mW	4 dBm	~25 metros
Clase 3	1 mW	0 dBm	~1 metro

En la mayoría de los casos, la cobertura efectiva de un dispositivo de clase 2 se extiende cuando se conecta a un transceptor de clase 1. Esto es así gracias a la mayor sensibilidad y potencia de transmisión del dispositivo de clase 1, es decir, la mayor potencia de transmisión del dispositivo de clase 1 permite que la señal llegue con energía suficiente hasta el de clase 2. Por otra parte la mayor sensibilidad del dispositivo de clase 1 permite recibir la señal del otro pese a ser más débil.

Topología de las redes Bluetooth

A diferencia de otras tecnologías LAN inalámbricas, como IEEE 802.11 (Wi-Fi), diseñadas para dispositivos que se hallen dentro o en los alrededores de un mismo edificio, los dispositivos que utilicen las redes PAN inalámbricas IEEE 802.15, incluyendo Bluetooth, podrán comunicarse en cualquier parte del mundo de forma stand-alone, incluso a bordo de un barco o avión y sin necesidad de utilizar equipo hardware adicional, como puntos de acceso. Cuando un dispositivo Bluetooth está dentro del radio de cobertura de otro, pueden establecer un enlace entre ellos. Hasta ocho unidades Bluetooth pueden comunicarse entre ellas y formar lo que se denomina una Piconet o Picorred.

La unión de varias piconets se denomina Scatternet o Red Dispersa lo cual se muestra en la figura 1.4

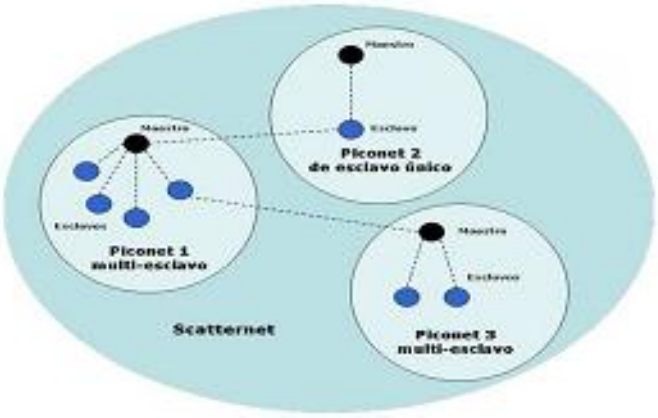


Fig. 1.4 Scatternet o Red Dispersa

Los dispositivos dentro de una piconet juegan dos papeles: maestro o esclavo. En todas las piconets sólo puede haber una unidad maestro, que normalmente es quien inicia la conexión, el resto de unidades Bluetooth en la piconet se denominan esclavos.

Cualquier dispositivo puede realizar las funciones de maestro y esclavo, pero un mismo dispositivo únicamente puede ser maestro de una piconet cuyo reloj y patrón de saltos se utilizan para sincronizar a todos los demás dispositivos esclavos.

La topología Bluetooth se puede describir como una estructura de piconets múltiples. Dado que la especificación Bluetooth soporta tantas conexiones punto a punto como punto a multipunto, se pueden establecer y enlazar varias piconets en forma de scatternet. Las piconets pertenecientes a una misma scatternet no están coordinadas y los saltos de frecuencia suceden de forma independiente, es decir, todos los dispositivos que participan en la misma piconet se sincronizan con su correspondiente tiempo de reloj y patrón de saltos determinado. Aunque no se permite la sincronización de diferentes piconets, los dispositivos pueden participar en diferentes piconets gracias a una multiplexación por división de tiempo (TDM).

Esto permite a un dispositivo participar de forma secuencial en diferentes piconets, estando activo en sólo una piconet cada vez.

Principio de Funcionamiento

Las redes de Bluetooth transmiten datos por medio de ondas de radio de baja potencia.
Se comunica en una frecuencia de entre 2.402 GHz y 2.480 GHz, para ser exactos. Estas frecuencias se han dado por una decisión conjunta a nivel internacional para su uso en dispositivos industriales, científicos y médicos. Los monitores de control de bebés, las puertas de garaje automáticas, y las nuevas generaciones de teléfonos inalámbricos, hacen uso de las frecuencias en la banda ISM. Asegurarse de que Bluetooth y estos otros dispositivos no se interfieren entre ellos, ha sido una parte crucial en el proceso de diseño.

Una de las maneras en las que Bluetooth evita interferir con otros sistemas, es enviando señales muy débiles de más o menos 1 milivatio. Por comparación, los teléfonos móviles más potentes pueden transmitir una señal de 3 vatios. Esta baja potencia limita el rango de acción de un dispositivo Bluetooth, a unos 10 metros, atajando las posibilidades de interferencias entre tu sistema informático y tu teléfono móvil o televisión.

1.3 Modulo RN-41

La RN-41 soporta múltiples perfiles de Bluetooth, está certificada por completo, y es simple en diseño, por lo que es una completa solución integrada Bluetooth. Con su alto rendimiento en un chip como la antena y soporte para Bluetooth Enhanced Data Rate (EDR), la RN-41 ofrece hasta 3 Mbps de velocidad de datos a distancias de 100 metros. La RN-41 es el producto perfecto para los ingenieros para la implementación de comunicación Bluetooth a sus productos sin tener que gastar mucho tiempo y dinero para desarrollar hardware con Bluetooth y software específicos.

La RN-41 puede ser configurada de diferentes modos:

HCI Mode: En este modo, la pila de Bluetooth se está ejecutando en un procesador externo (no en la RN-41).

Hay dos posibles opciones de interfaz de hardware:

- UART: Esto se llama HCI en H4. La velocidad de transmisión en el que la RN-41 se comunica con el procesador es fijo y tiene que ser programado (en la RN-41) durante el proceso de actualización del firmware. Los clientes necesitan para especificar esta el pedido.

- USB: En este modo, la RN-41 es la interfaz con el procesador externo utilizando una interfaz USB. La RN-41 actúa como un esclavo USB (no es un host USB).

La principal ventaja del modo de HCI es que permite a los clientes ejecutar perfiles personalizados en su procesador. También proporciona altas velocidades de datos (hasta 3 Mbps).

HID Mode: el cual sirve para dar soporte para la creación o aplicación de teclados, mouses u otros dispositivos que usen la Human Interface Device.

Características

- Totalmente cualificado para la clase 1 Bluetooth 2.1 + EDR módulo.
- Bluetooth SIG calificado.
- UART (SPP o HCI) y USB (sólo HCI).
- Pila Embebida integrada (procesador central no es necesario).
- Compatible con Bluetooth de enlace de datos para iPhone / iPad / iPod Touch
- Compatible con el perfil HID para la fabricación de accesorios tales como teclados, ratones, dispositivos de señalización.
- Modo de bajo consumo programables.
- Comunicaciones seguras, encriptación de 128 bits.

- Corrección de errores en la entrega de paquetes garantizados. Auto-discovery/pairing no requiere ninguna configuración de software (el reemplazo de cables instantáneos).

1.4 Arduino UNO

Arduino es una plataforma electrónica abierta para la creación de prototipos basada en software y hardware flexibles y fáciles de usar.

Arduino puede tomar información del entorno a través de sus pines de entrada de toda una gama de sensores y puede afectar aquello que le rodea controlando luces, motores y otros actuadores. El microcontrolador en la placa Arduino se programa mediante el lenguaje de programación Arduino (basado en Wiring) y el entorno de desarrollo Arduino (basado en Processing). Los proyectos hechos con Arduino pueden ejecutarse sin necesidad de conectar a un ordenador, si bien tienen la posibilidad de hacerlo y comunicar con diferentes tipos de software (p.ej. Flash, Processing, MaxMSP).

El Arduino Uno posee:

14 entradas/salidas digitales, de los cuales 6 pueden ser usados como salidas PWM
Posee 6 entradas analógicas
Los pin 0 y 1 pueden funcionar como RX y TX serial.
Un oscilador de crystal de 16 MHz
Conector USB
Un jack de poder
Un conector ICSP
Botón de Reset

El Arduino UNO posee todo lo que se necesita para manejar el controlador, simplemente se conecta a un computador por medio del cable USB o se puede alimentar utilizando una batería o un adaptador AC-DC. Si se conecta por USB, la alimentación externa no es necesaria.

Algunas características son:

Microcontroller	ATmega328
Operating Voltage	5V
Input Voltage (recommended)	7-12 V
Input Voltage (limits)	6-20 V
Digital I/O Pins	14 (of which 6 provide PWM output)
Analog Input Pins	6
DC Current per I/O Pin	40 mA
DC Current for 3.3V Pin	50 mA

Flash Memory	32 KB (of which 0.5 KB used by bootloader)
SRAM	2 KB
EEPROM	1 KB
Clock Speed	16 MHz

Las Entradas analógicas son de 10 bits, por lo que entregan valores entre 0 y 1023. El rango de voltaje está dado entre 0 y 5 volts, pero utilizando el pin AREF, este rango se puede variar a algún otro deseado.

1.5 Plataforma Para el Hardware

1.5.1 Energía

Una de las partes más importantes en el diseño de sistemas es que la energía es el parteaguas de todo proyecto, recordemos que todas las cosas que se mueven necesitan alguna forma de energía y, como ejemplo simple, cuando un objeto cae su fuente de energía es la gravedad. Por eso el poder, trabajo y energía están más ligados de lo que podemos pensar. Una de las experiencias que me ha dejado es que cuando decidimos realizar algún proyecto casi nunca consideramos que el suministro de energía debe de ser un tema de consideración, y más cuando esta está generada por algún medio portátil como baterías o celdas fotovoltaicas; cuántas veces hemos alimentado nuestro circuito y en un lapso de tiempo se para, observamos que nuestra expectativa de vida no fue la que esperamos, esto es porque nunca consideramos el consumo que éste tiene en un lapso de tiempo y si nuestra fuente puede proporcionar ese desgaste energético.

1.5.1.1 Energía Mecánica

La energía mecánica es la suma de energía potencial y cinética de un objeto. La energía potencial es la cantidad de energía que se almacena en un objeto en reposo. La energía cinética es la que tiene un objeto debido a su movimiento. Tomando un ejemplo de un libro; una pelota parada en la cima de una colina, tiene una energía potencial igual a su peso multiplicada por la altura de la colina.

Energía Potencial = Peso x altura

Si la bola pesa dos libras, y el cerro se encuentra a 20 metros de altura, la energía potencial es de 40 ft-lb. Si se empuja y empieza a rodar por la colina, la energía potencial se convierte en energía cineteca.

Energía Cinética = ½ x masa x Velocidad^2

Mientras la pelota está rodando por la colina, la energía potencial está disminuyendo (porque está perdiendo altura), mientras que la energía cinética es cada vez mayor (porque va más rápido). Al llegar al pie de la montaña la pelota no tiene energía potencial porque se ha convertido en energía cinética.

Nota: Este es uno de los tantos factores físicos que afectan a nuestro robot y debe de darse cuenta que dependiendo de nuestro diseño este puede ir necesitando mayores puntos a analizar, puesto que las leyes físicas están presentes en toda actividad de la tierra.

1.5.1.2 Cálculos

Primero hay que analizar al robot, este tiene una masa y realiza acciones que uno le asigna, a menudo podemos calcular la energía que se necesitará. Por ejemplo, si sabemos que el robot pesa 1.5k (con baterías incluidas) y tiene que subir una escalera de 6 metros 2 veces al día, se puede determinar la energía potencial que se necesita para que el robot suba la escalera, con la formula PE = m X g X h:

$$PE = 1.5\ kg * 9.8\ {}^{m}/_{s^2}\ * 6\ m = 88.2\ joules$$

Si un joules es igual a .000278 watt-hora, 88.2 joules equivale a 0.0245196. Como datos de referencia coloco una tabla de relación de algunas baterías recargables (Tabla 1.1) con su relación watt-hora. Quiero aclarar que existen muchas otras tecnologías de baterías disponibles en el mercado, por eso recomendamos investigar más posibilidades y prestaciones para nuestro proyecto.

Tabla 1.1

Tipo de Batería	Watt-hora	Material
D	**4.0**	**Nicad**
D	**7.8**	**NMH**
AA	**0.6**	**Nicad**
AA	**1.8**	**NMH**
AAA	**1.2**	**Nicad**

Hoy en día existen baterías más eficientes, de menor tamaño y precio un poco razonables si consideramos las prestaciones que nos dará, un ejemplo de ello son las baterías de Polímero de Litio (Lipo), o las de Ion Litio (Li-Ion), estas baterías podrían considerarse como un avance de generación puesto que brindan mayor energía con respecto al peso y de soportar mayor cantidad de cargas y descargas que las anteriores mencionadas. En los temas siguientes se muestra más casos de análisis al momento de escoger una fuente de energía.

1.5.1.3 Medidas empíricas

Otra forma de saber cuál sería el consumo energético, se basa en realizar un modelo de prueba de nuestro sistema, de esta forma podemos observar mejor su requerimiento, pero esto en ocasiones implica más costos de elaboración. Existen diversos programas que son capaces de simular tanto sistemas eléctricos como mecánicos pero sabemos que muchos de los casos no pueden ser simulados, pero esto nos brinda resultados cercanos que podemos considerar en nuestro diseño.

1.5.1.4 Baterías

Las baterías son el medio de suministro energético más utilizado en los circuitos móviles o robótica, y en este tema mostramos consideraciones que nos ayudarán a escoger la batería que se adapte mejor a nuestro requerimiento y diseño. Como nuestro proyecto posee características de movilidad por medio inalámbrico, hemos requerido de una batería y estas son las consideraciones que hemos tomado.

1.5.1.4.1 Energía de las baterías

Todas las baterías, sin importar su tipo, se miden en amperio-hora, es decir, la corriente (medida en amperios o miliamperios) multiplicado por el tiempo (hora) que la corriente fluye en la batería. Es necesario tomar este dato (Tema 1.2.2) ya que nos sirve de referencia del tiempo que la batería suministrará carga para nosotros. Esto es fácil al momento de querer saber el tiempo que nuestro circuito funcionará, tomando un ejemplo, supongamos que nuestro carro una vez terminado consumirá toda la circuitería 1 Amper, y la batería que sirve de fuente entrega una corriente de 2 Amper-hora, esto significa que nuestro carro andará por 2 horas, pero si este consumiera 2 Amper esto nos limita a que la batería pueda alimentarlo durante 1 hora, esta misma analogía para el cálculo del tiempo es igual una vez sabiendo el consumo. Vemos entonces porque es importante este dato en nuestro diseño.

El tema de las baterías es muy extenso y no es nuestro objetivo proporcionarlo en este manual, pero podemos adelantarle que las baterías tienen una eficiencia y está ligada por el tipo de materia que la componen, además de poseer una resistencia interna que causa

que su eficiencia no sea del 100% y que el valor de esta depende de la calidad, marca y precio. El fabricante proporciona información sobre esto, y como dato importante, todas las baterías poseen una gráfica que marca la relación de descarga y tiempo, esto significa que las baterías trabajan mejor por lapsos de tiempo ya que permiten que recuperen su carga y tiempo para enfriamiento. Entonces al considerar una batería debemos observar la temperatura de trabajo, densidad de energía, cargas y descargas, voltaje, corriente, capacidad y hasta su vida útil.

1.5.2 Locomoción

Esta es una parte importante del proyecto sin menospreciar a las anteriores mencionadas, pero esta juega un papel especial en el robot, porque para deslizarse y saltar, en la mecánica de un robot, existen diversas formas de lograrlo y el análisis de la locomoción nos ayudara a entender esto.

La locomoción se descompone en dos partes: la que realiza el apoyo sobre el medio en el que se espera que se desplace el robot y la que permite su propulsión. Esta última incluye los motores y los mecanismos que permiten el desplazamiento como son las llantas, rieles, orugas, patas, etc. Sabemos que dependiendo del medio en el que se desplace (agua, tierra, aire) será el mecanismo que elijamos.

En nuestro proyecto, al considerar que se desplazara, sobre superficies terrestres y con regularidad media, optamos por el uso de las llantas o ruedas.

1.5.3 Dirección

Ahora nos centremos en la dirección de nuestro móvil. Este punto está ligado a la forma que tendrá el robot, considerando la forma o el número de llantas, así como la libertad de navegación y el tipo de trabajo que debe realizar, durante la investigación pude observar que existen varios arreglos de ruedas de las cuales un diseñador puede elegir, entre estas se encuentran (ver Figura 1.5)

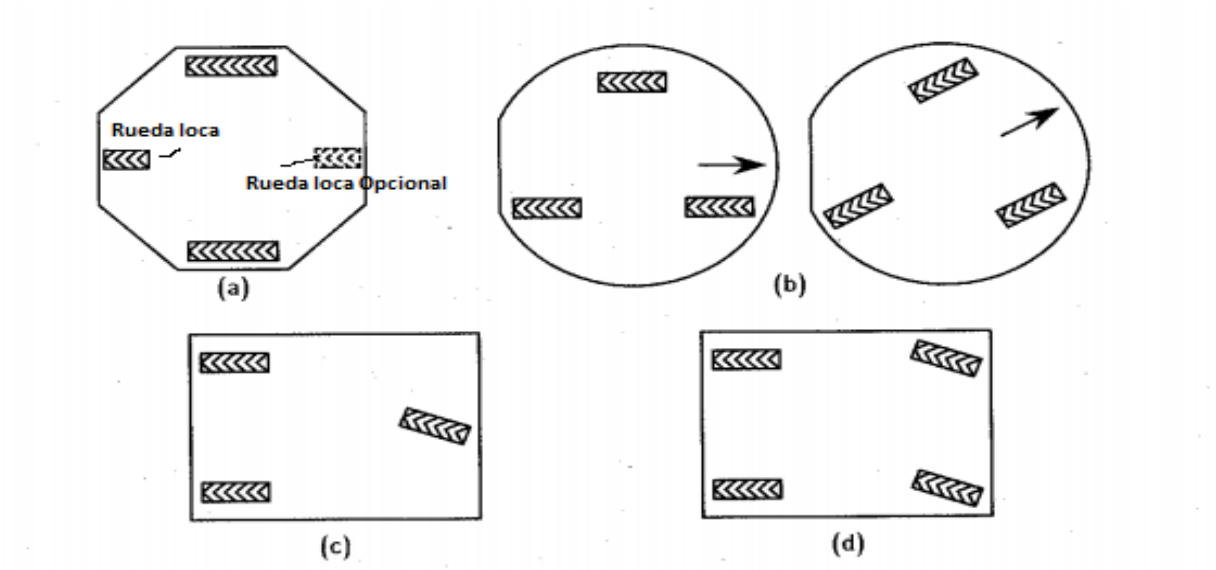


*Figura 1.5., observemos que entre los arreglos se encuentra **(a)** un arreglo con ruedas locas y tracción lateral, **(b)** un arreglo el cual usa tres ruedas y la tracción dependerá del torque y maniobrabilidad que requiramos, **(c)** el arreglo bicicleta que depende de una tracción trasera y una rueda delantera y el **(d)** un arreglo tipo automóvil, ese basa en tener tracción trasera y un eje que posee dos ruedas.*

Basándonos en nuestro requerimiento, decidimos optar por el arreglo automóvil, ya que considerando el peso, volumen y adaptaciones del proyecto, observamos que este medio proporciona mayor soporte al peso, estabilidad en terreno irregular y adaptaciones al chasis.

Es claro que estos son modelos base y que se prestan para modificaciones o adaptabilidad en el diseño y forma de nuestro robot, es por eso que debemos jugar un poco con el diseño y observar su desempeño.

1.5.4 Manejo

Ya tocamos el tema de la dirección pero falta el análisis del manejo, recopilando información notamos que éste está profundamente ligado con la dirección y ésto es así porque depende del arreglo de ruedas, lo que nos permite configurar el método de manejo.

Como se observa, es posible configurar diversas formas de manejo en un mismo diseño, cada una contiene un punto fuerte y uno débil, ya que cada uno se amolda al tipo de robot que vamos a diseñar. Otras formas son, ver Figura 1.6

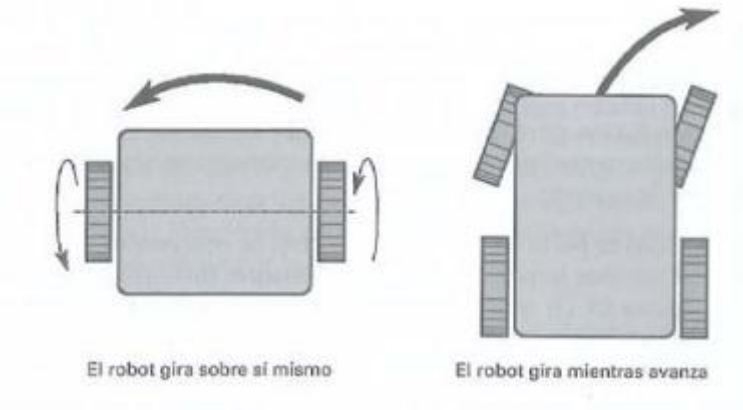


Figura 1.6

Se observa que el cambio de dirección se consigue variando la velocidad de los motores asociados a cada una de las ruedas laterales, o variando la orientación de las ruedas.

Si tomamos como ejemplo el arreglo **(a)** o **(b)** de la Figura 1.5, podemos observar que la tracción estaría más relacionada a las llantas laterales o las dos traseras, esto repercute en nuestro diseño de la siguiente forma, si analizamos bien la forma en que este giraría observamos que, si colocamos la tracción a las dos ruedas sin posibilidad de control independiente, esto causa que el móvil abarque más área de la superficie al realizar un giro, ya que las dos ruedas generan un vector de dirección, pero es caso contrario si

nosotros implementamos que la tracción se encuentre de igual forma en las llantas traseras pero con la capacidad de manipularlas independientemente, estos nos genera dos métodos más, uno con el cual podamos apagar una llanta y girar otra, esto causa que el móvil pueda girar en su propio eje, pero generando arrastre; y la segunda es girar una rueda al sentido contrario que la otra obteniendo un giro más suave y evitando lo más posible el lastre del hule en la superficie.

Al optar por un método tipo automóvil, nuestro manejo está ligado a tener una tracción trasera y no necesariamente control independiente, esto es porque en las ruedas delanteras se encontrara un sistema que proporcione dirección, este diseño se fortalece por que posee mayor estabilidad en terrenos irregulares, equilibrio, soporte de peso, tamaño, por mencionar algunos. Pero en giro este posee un área mayor (Figura 1.7), como se explicó en el modelo anterior.

El tema del área de giro es importante porque este nos delimitaría hasta qué grado nuestro robot puede acceder a áreas con espacios reducidos o la velocidad de giro, respuesta rápida y maniobras difíciles. En nuestro caso queremos un robot móvil que funcione en terrenos más escarpados y con una dirección fiable si con el tiempo pensamos adaptarle nuevos sistemas.

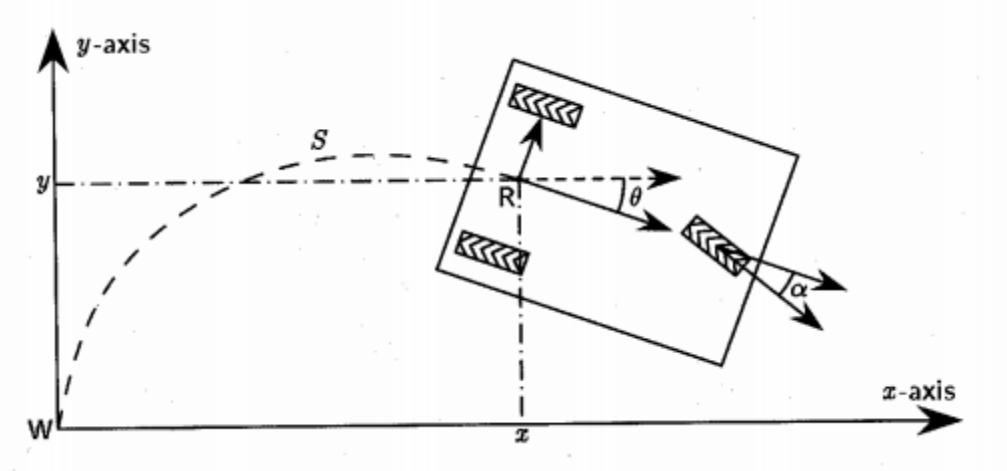


Figura 1.7

En la figura se muestra el recorrido que debe realizar el móvil para cumplir una vuelta. Recordar que cada modelo tiene una trayectoria, para ejercer un giro es indispensable el diseño porque de esta forma podemos deducir cual cumple con el requerido, en la Figura 1.8 ponemos una comparación de dos configuraciones de rueda y manejo.

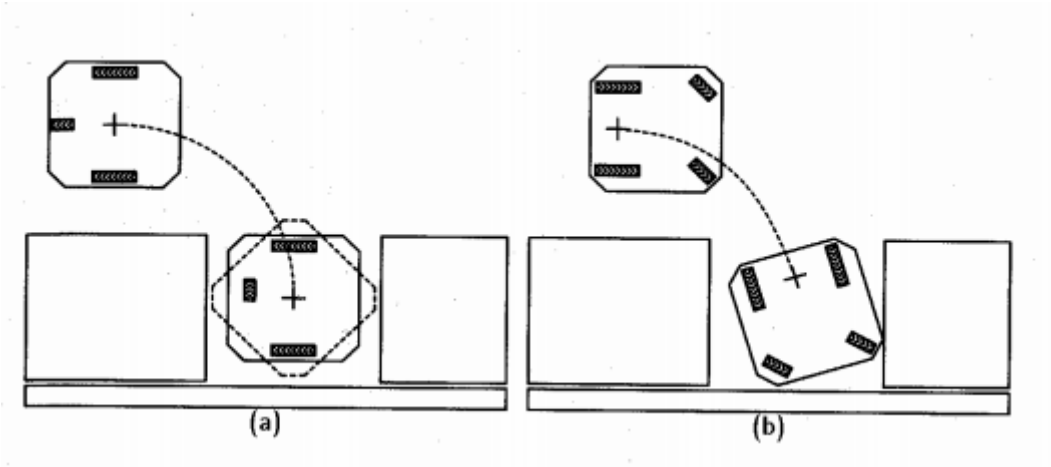


Figura 1.8

1.5.5 Motores

Los motores son simplemente dispositivos que toman la energía eléctrica y la convierten en energía mecánica. La mayoría de los motores convierten la energía de un campo magnético con bobinas, pero unos pocos no utilizan bobinas para hacer este trabajo. La energía que ellos utilizan puede provenir de una fuente de CA o CD y en función del tipo de energía que usan es como se clasifica.

Aquí pretendemos dar una idea de consideración al construir un robot, y como lo hemos hecho en los temas anteriores, solo marcaremos algunos puntos importantes y teoría practica para analizar; los motores son una materia de estudio y como tal, cada modelo y tipo contiene una extensa información y fórmulas que nos detallan su funcionamiento y operación.

1.5.5.1 Motores de CD

Este tipo de motores es el más común en aplicaciones en la robótica puesto que su alimentación puede estar debida a una batería y no posiblemente a un extenso cable como seria en los tipos CA, además vienen en muchos estilos diferentes. En los motores de CA tienen menos estilos porque su arquitectura intenta aprovechar el movimiento existente (onda) de la alimentación de CA.

En general, los motores de CD tienen imanes permanentes en el estator y el rotor de las bobinas (al revés a los motores de CA). Sin embargo, desde CD no tienen movimiento a causa de la forma de onda de la señal, es por eso que la electrónica del motor debe de hacer este cambio de polaridad. Esto puede suceder de diferentes maneras.

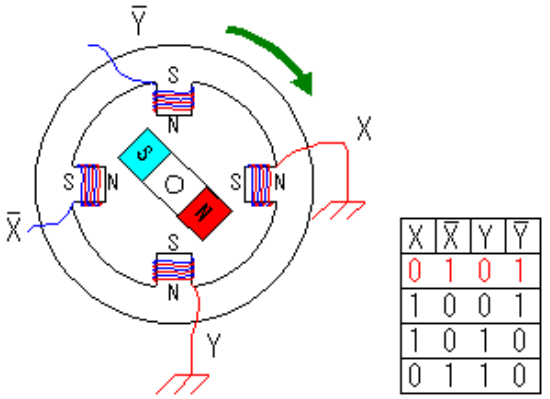
Figura 1.9 Ejemplo de un motor de CD

1.5.5.2 Motor a Pasos

Este tipo de motores se puede utilizar para la locomoción, movimiento, dirección y control de posicionamiento. Estos motores son únicos porque se pueden controlar por medio de circuitos digitales, esto les brinda mucha aceptación en la robótica o aplicaciones electrónicas. Además son ideales para el control de forma lineal o rotativa.

Cuando se aplica energía a un motor eléctrico estándar, su rotor empieza a girar sin problema, la velocidad y la posición del rotor del motor es una función de la tensión, la carga del motor y tiempo. Un posicionamiento del rotor no es posible. Un motor paso a paso, sin embargo, se ejecuta en una secuencia de pulsos eléctricos a las bobinas del motor. Cada pulso de una bobina hace girar el rotor de una cantidad exacta predeterminada. Los movimientos incrementales del rotor a menudo son llamados pasos. De ahí el nombre de motor paso a paso.

No todos los motores paso a paso giran el eje (rotor) en la misma cantidad por paso. Estos son fabricados con diferentes grados de rotación por el paso (o pulso). Los grados óptimos en cada paso dependerán de la aplicación en particular. Cuando se compra un motor de este tipo, en su hoja de especificaciones explican claramente el grado de rotación. Usted puede encontrar una gran variedad de motores, con diferentes grados por paso que van desde una fracción de grado (es decir 0.72 grados) a muchos grados (es decir 22.5 grados).



X	X̄	Y	Ȳ
0	1	0	1
1	0	0	1
1	0	1	0
0	1	1	0

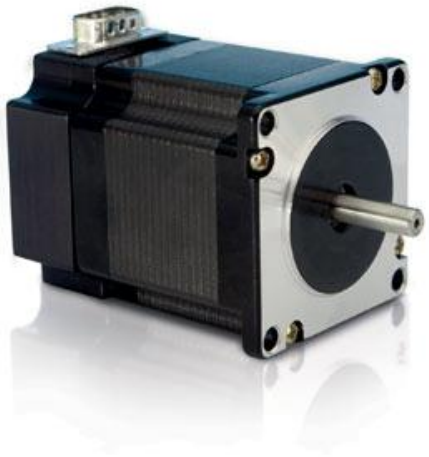


Figura 1.10 Estructura del motor

Figura 1.11 Motor a Pasos

1.5.5.3 Servomotor

Un servomotor (también llamado servo) es un dispositivo similar a un motor de corriente continua que tiene la capacidad de ubicarse en cualquier posición dentro de su rango de operación, y mantenerse estable en dicha posición. Está conformado por un motor, una caja reductora y un circuito de control.

La corriente que requiere depende del tamaño del servo. Normalmente el fabricante indica cual es la corriente que consume. La corriente depende principalmente del par, y puede exceder un amperio si el servo está enclavado, pero no es muy alto si el servo está libre moviéndose todo el tiempo.

Los servomotores hacen uso de la modulación por ancho de pulsos (PWM) para controlar la dirección o posición de los motores de corriente continua. La mayoría trabaja en la frecuencia de los cincuenta hercios, así las señales PWM tendrán un periodo de veinte milisegundos. La electrónica dentro del servomotor responderá al ancho de la señal modulada. Si los circuitos dentro del servomotor reciben una señal de entre 0,5 a 1,4 milisegundos, este se moverá en sentido horario; entre 1,6 a 2 milisegundos moverá el servomotor en sentido antihorario; 1,5 milisegundos representa un estado neutro para los servomotores estándares. A continuación se exponen un ejemplo de control (Figura 1.7):

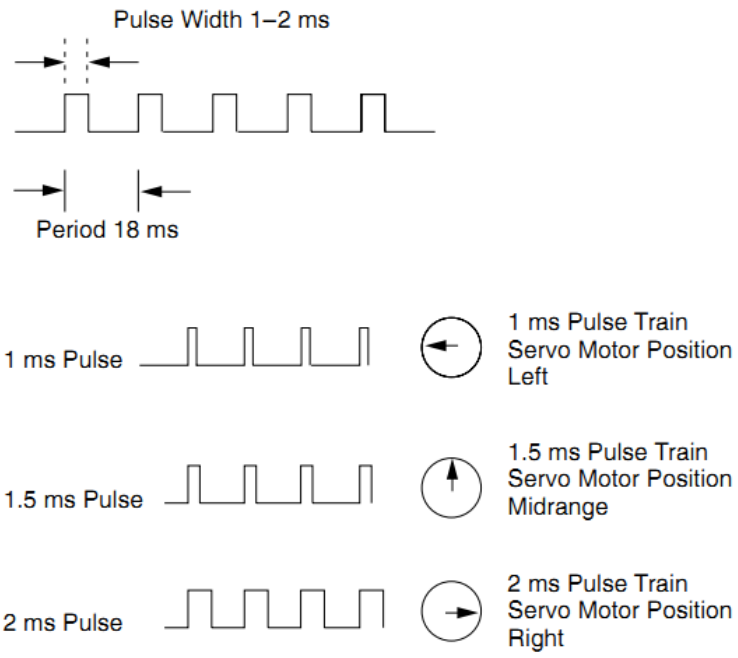


Figura 1.12 ejemplo de control de un servomotor

Figura 1.13 Servomotor

1.5.5.4 Motor de CA

Se denomina motor de corriente alterna a aquellos motores eléctricos que funcionan con corriente alterna. Existe una gran variedad de motores de CA, entre ellos tres tipos básicos: el universal, el síncrono y el de jaula de ardilla. Este tema no será profundizado puesto que nuestro móvil no utilizara este tipo de motores, sin embargo son utilizados en la robótica.

1.5.6 Puente H

Un Puente H es un circuito electrónico que permite a un motor eléctrico DC girar en ambos sentidos, avance y retroceso. Son ampliamente usados en robótica y como convertidores de potencia. Los puentes H están disponibles como circuitos integrados, pero también pueden construirse a partir de componentes discretos, ya que la interfaz de potencia para motores de CC, PUENTE H, es un sistema que permite controlar motores en rangos entre 12 y 30 voltios y con consumos de hasta 30 amperios por medio de relés de potencia. La figura 1.14 nos muestra la estructura básica de un puente H.

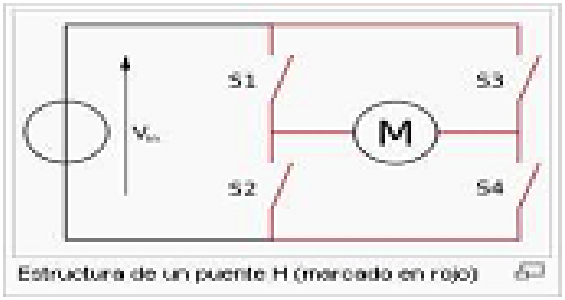


Fig. 1.14.- Estructura básica de un Puente H

El término "puente H" proviene de la típica representación gráfica del circuito. Un puente H se construye con 4 interruptores (mecánicos o mediante transistores). Cuando los interruptores S1 y S4 (ver primera figura) están cerrados (y S2 y S3 abiertos) se aplica una tensión positiva en el motor, haciéndolo girar en un sentido. Abriendo los interruptores S1 y S4 (y cerrando S2 y S3), el voltaje se invierte, permitiendo el giro en sentido inverso del motor.

Con la nomenclatura que estamos usando, los interruptores S1 y S2 nunca podrán estar cerrados al mismo tiempo, porque esto cortocircuitaría la fuente de tensión. Lo mismo sucede con S3 y S4. La figura 1.15 muestra los estados de un Puente H:

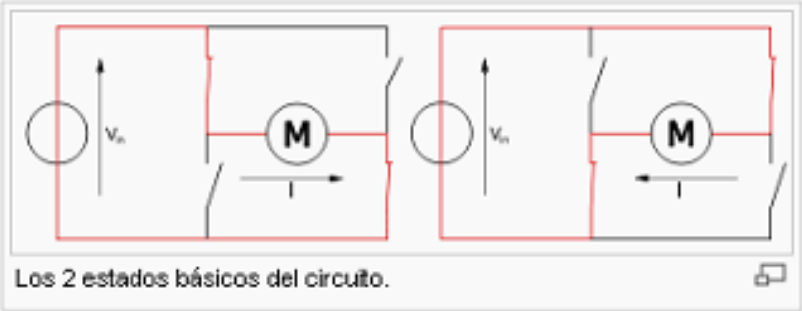


Fig. 1.15 Estados de un Puente H

Aplicaciones

Como hemos dicho el puente H se usa para invertir el giro de un motor, pero también puede usarse para frenarlo (de manera brusca), al hacer un corto entre las bornas del motor, o incluso puede usarse para permitir que el motor frene bajo su propia inercia, cuando desconectamos el motor de la fuente que lo alimenta.

En la Tabla 1.2 se muestra un cuadro de resumen de las diferentes acciones que posee un Puente H:

S1	S2	S3	S4	Resultado
1	0	0	1	El motor gira en *avance*
0	1	1	0	El motor gira en *retroceso*
0	0	0	0	El motor se detiene bajo su inercia
0	1	0	1	El motor frena (*fast-stop*)

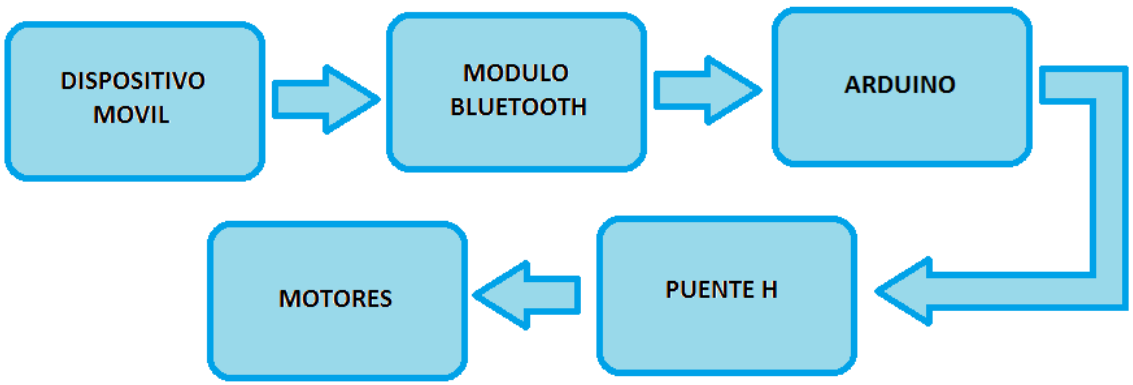
Tabla. 1.2.- Resumen de Acciones de un Puente H

UNIDAD 2

DISEÑO Y CONSTRUCCIÓN

2.1 Diseño

La temática del proyecto aborda la conectividad entre dos dispositivos y la forma de interactuar entre ellos. La idea es demostrar que no importa qué tipo de dispositivo se use, con tal de que posea interfaces que nos permitan un flujo de datos, un método de programación y un equipo que actúe como centro de control de datos. Esto hace posible que podamos enviar datos y así sacarle provecho a esta mezcla de tecnologías. En este caso se trabajó con microcontrolador ATMEL 328 bajo una placa de desarrollo ARDUINO, un módulo de comunicación Bluetooth y un celular.



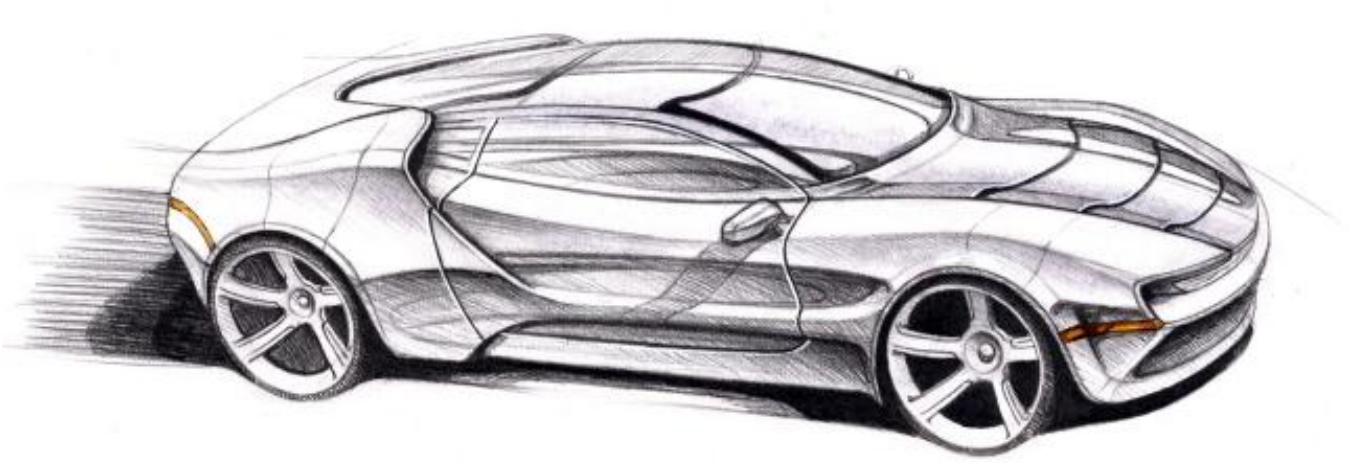
El equipo central de control se basa en un dispositivo celular con sistema operático Android, el cual por medio de una aplicación se llevara el manejo de las direcciones del robot, el celular posee un módulo Bluetooth ya integrado, por ese medio se usara el soporte de App inventor para su control.

El modulo Bluetooth RN-41 posee salida serial TTL el cual será conectado a la placa de desarrollo de ARDUINO, por medio de sus pines TX/RX. El RN-41 posee un Firmware interno que convierte del protocolo Bluetooth a serial, de esta forma solo debe considerarse el envio de cadenas de bits, dicho de otras palabras el módulo funciona como un Puerto Serial COM.

Después de analizar los tipos de locomoción y dirección que se pueden implementar en el diseño del robot móvil, observamos que todos se adaptan a nuestro requerimiento, sin embargo en estos casos debemos, al elegir un método, considerar otros aspectos influyentes y, en este caso, el presupuesto.

Finalmente podemos controlar el robot usando el equipo de telefonía celular. Este es un robot con funciones básicas (grados de libertad), hablamos de movimientos básicos como son los movimientos similares a los de un auto de juguete, ya que la idea central del proyecto es probar la conectividad entre estos dispositivos.

2.1.1 Robot Móvil



La estructura es simple basado en las funciones que éste tendrá. Una base sólida como única estructura del robot, la cual llevara los circuitos electrónicos, baterías y antenas sobre ella, y por debajo de ésta los motores de tracción. Tomando como medida la las placas electrónicas, motores y ruedas.

Se consideró el siguiente diseño; un sistema de 3 ruedas, de las cuales 2 estarán conectadas a motorreductores con las siguientes características:

Escala	Torque	Velocidad	Peso	Consumo S/Carga	Consumo con Carga
1:220	3.6KgF*cm	36 RPM	32gr	75mA	670mA

Para el sistema de dirección del robot móvil implementaremos el método de giro a la inversa. Este modo funciona de la siguiente manera: para hacer el movimiento de desplazamiento hacia adelante los dos motores girarán a la misma velocidad, de la misma forma seria para el sistema de reversa.

Caso contrario para los giros hacia la izquierda y derecha. Usualmente se usa el método de parar el motor del lado al que nos queremos dirigir, de esta forma se genera lastre y dependiendo del terreno puede o no tener mucha libertad, esto se explicó en el apartado de Manejo en la unidad xx. Para evitar este tipo de problemas implementaremos nosotros el giro a la inversa, es decir si queremos que gire a la derecha el motor del lado derecho girara anti horario y el de la izquierda al sentido horario.

El robot móvil no llevara mucho hardware de control, ya que la placa ARDUINO simplifica mucho hardware. Es por eso que se pensó en 3 módulos a diseñar:

- La fuente de alimentación, la cual suministrará energía tanto a la placa de control de motores como al ARDUINO.
- Control de motores: la cual se basa en un Circuito Integrado L293B (puente H) en configuración básica y circuitos de protección.
- ARDUINO.

Se calculó que la plataforma principal seria de madera liviana (Triplay) y llevaría las siguientes dimensiones:

Alto: 17cm
Ancho: 12cm
Espesor: 5mm

De esta manera se cumple con los requerimientos del robot móvil (fig. 2.1).

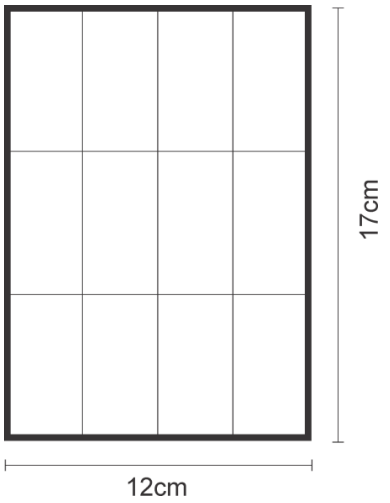
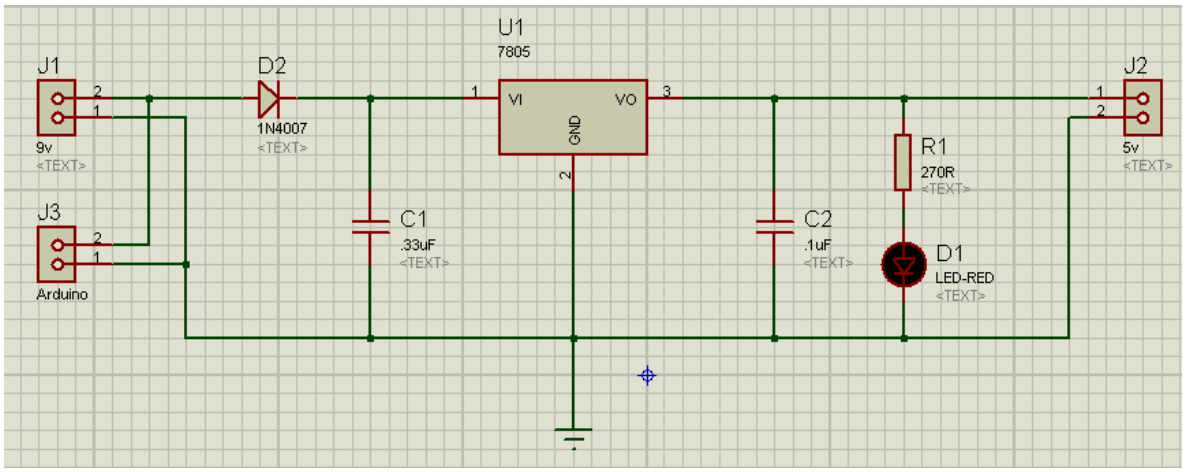


Fig. 2.1

Para la fuente de poder requirió de dos tipos de voltaje, el cual uno seria de 5vcd y otro de 9vcd, esto se debe a que los motores trabajan en un rango de 5vcd y la placa ARDUINO al poseer una entrada tipo Jack el cual deben de ingresar de 9 a 12vd, puesto que en el interior se encuentra un regulador a 5vcd.



El circuito está basado en:

- UI 7805CT
- C1 Capacitor cerámico de .33uF
- C2 Capacitor cerámico de .1uF
- D2 Diodo 1N4007
- D1 LED Rojo
- R1 Resistencia 270 ohm
- J1,2,3 Conectores T-Blocks

Para el diseño del puente H la configuración básica del L293B, el cual lleva un arreglo de diodos rectificadores para la inversión y protección de la señal. En la imagen 2.2, se muestra el diagrama de nuestro circuito.

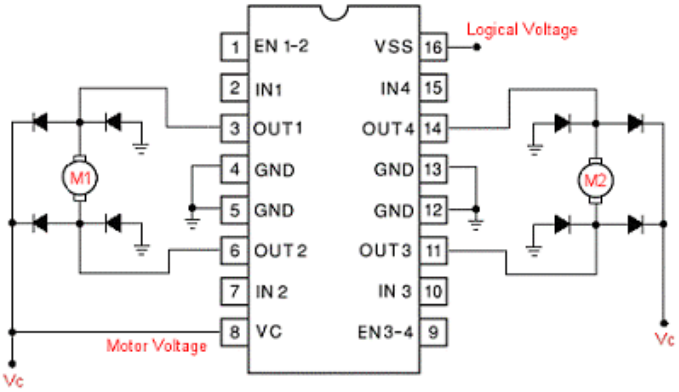


Fig.2.2

La placa de control de motores tendrá los siguientes componentes:

- 8 Diodos 1N4007
- L293B (puente H)
- T-Blocks como conectores

De esta manera el diagrama final podría considerarse como el de la figura 2.3.

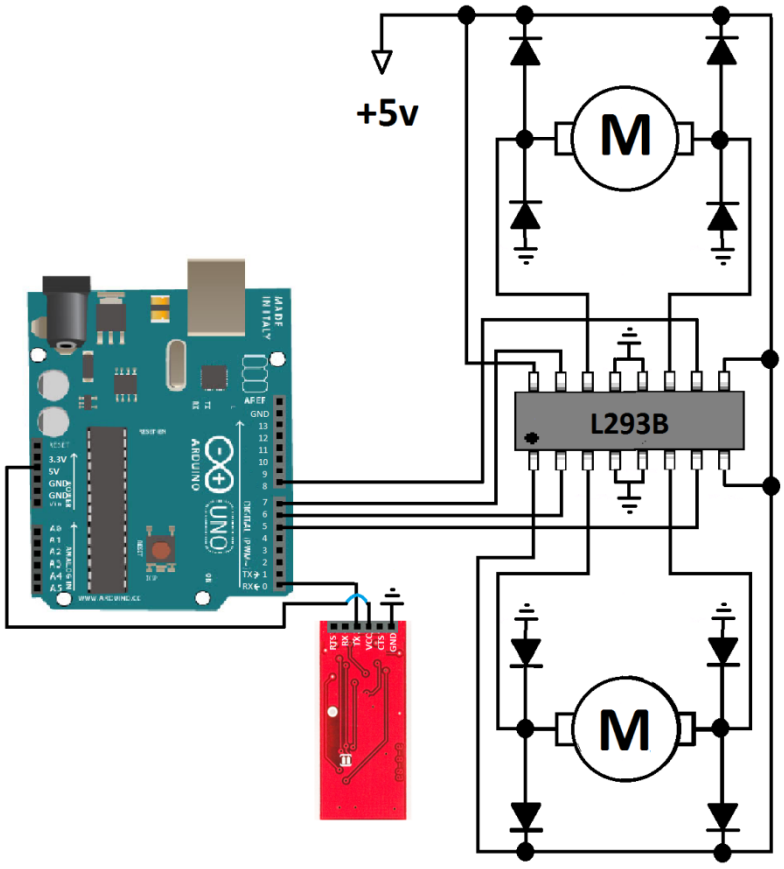
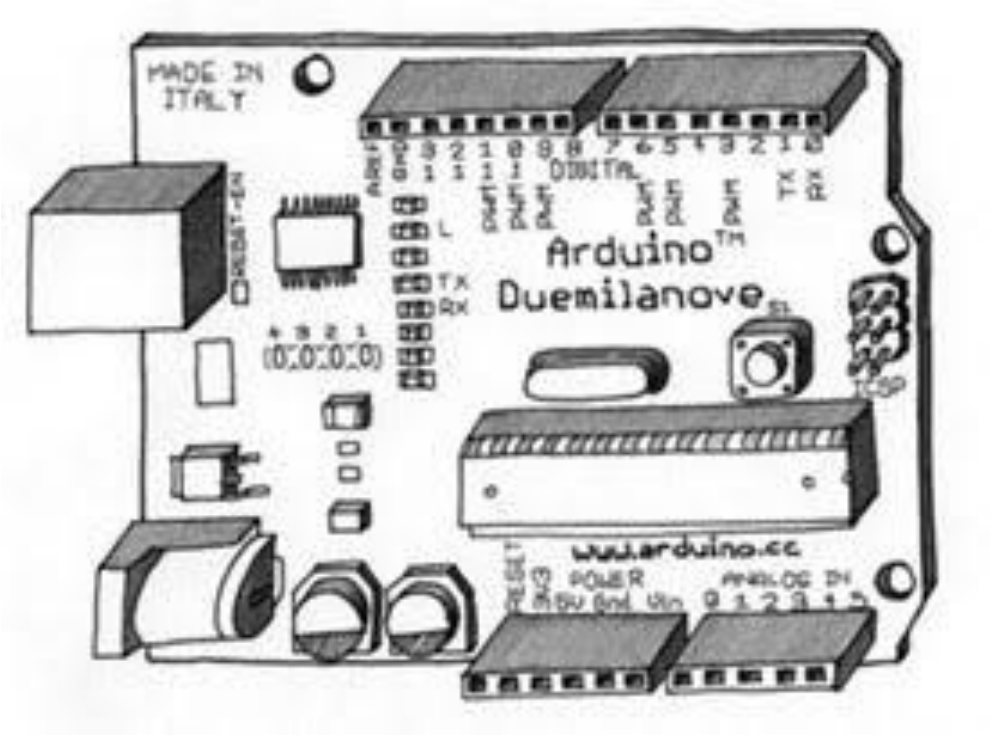


Figura 2.3

2.1.2 Código Arduino



Todo el sistema de control y procesamiento de información se encuentra en el microcontrolador, de esta forma le damos un giro universal al control del Robot móvil. El Arduino llevará una programación en la cual analizará variables que estará recibiendo de la comunicación por Bluetooth y dependiendo de esta llevará a cabo una acción. Esto nos da la libertad de poder controlar el robot desde otro entorno o plataforma que cuente con un módulo Bluetooth, un ejemplo de esto podría ser que desde la PC y por medio de una terminal mandar dichas variables de control, o realizar algún software por medio de otra aplicación dedicada a esta tarea.

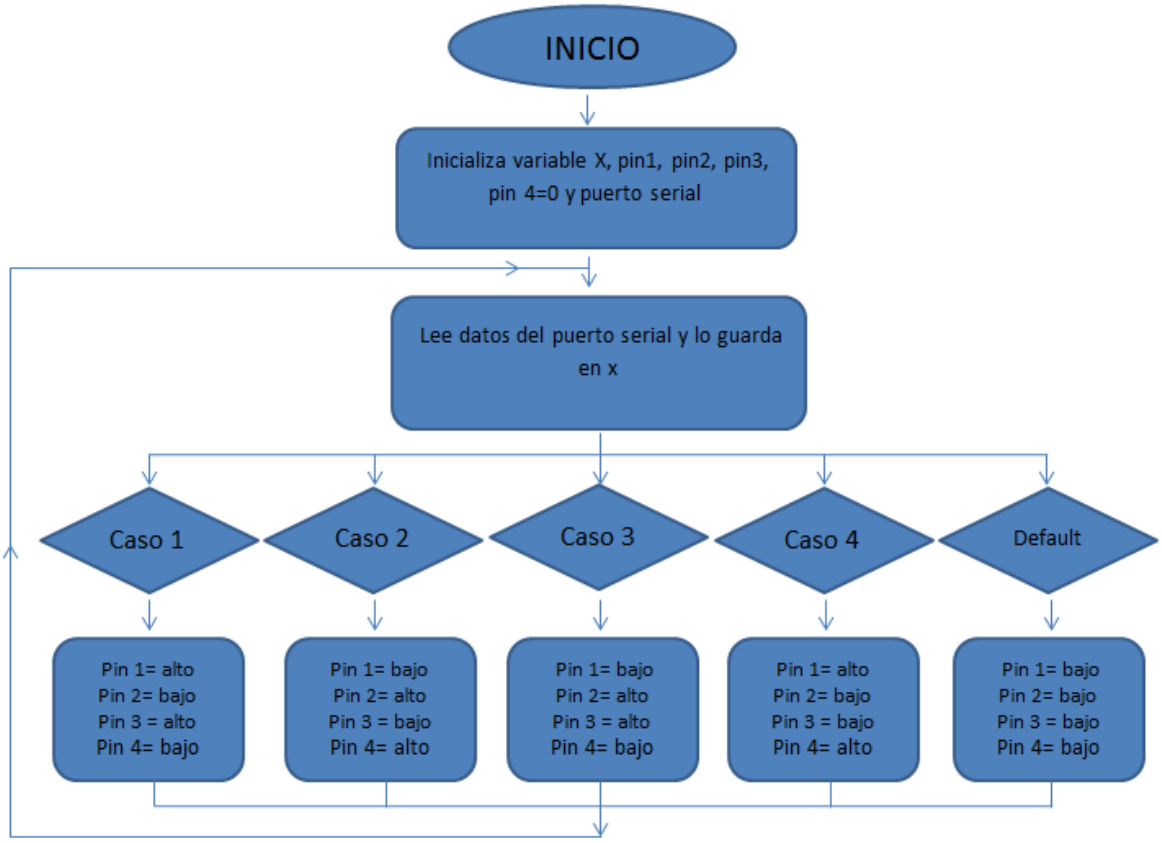


Figura 2.4

Una vez que tenemos la estructura del programa (Fig. 2.4) debemos de desarrollarlo en el ARDUINO IDE. A continuación se explican los recursos, librerías y sintaxis que se usaron en el programa.

int

Integers (Números enteros) son el principal tipo de datos para almacenar números, y guardan valores de **2 bytes**. Esto produce un rango entre *-32,768* hasta *32,767* (valor mínimo de -2^15 y un valor máximo de (2^15) - 1).
Variables tipo **Int**, almacenan números negativos con una técnica llamada Complemento a dos. El *bit* más alto, a veces llamado como *"sign" bit*, indica que el número es negativo. Se invierte el valor de cada uno de los *bits*, es decir, se realiza el complemento a uno, y se suma 1 al número obtenido.
La placa Arduino, se encarga de tratar con números negativos por tí, para que las operaciones aritméticas trabajen de manera transparente y en la forma esperada.

Ejemplo

int ledPin = 13;
Sintaxis
  int var = val;
 var - nombre de la variable *int*.
 val - valor asignado a dicha variable.

setup()

La función setup() se establece cuando se inicia un programa –sketch (compilador de programación). Se emplea para iniciar variables, establecer el estado de los pines, inicializar librerías, etc. Esta función se ejecutará una única vez después de que se conecte la placa Arduino a la fuente de alimentación, o cuando se pulse el botón de reinicio de la placa.

Ejemplo:

```
int buttonPin = 3;

void setup()
{
  Serial.begin(9600);
  pinMode(buttonPin, INPUT);
}
```

pinMode(pin, modo)

Configura el pin especificado para comportarse como una entrada o una salida. Mira la descripción de pines digitales para más información.

Parámetros

pin: el numero del pin que se desea configurar
modo: INPUT (Entrada) o OUTPUT (Salida)

Loop()

Luego de crear la función setup(), la cual inicializa y prepara los valores inciales, la función loop() hace justamente lo que su nombre sugiere, por lo tanto se ejecuta consecutivamente, permitiéndole al programa variar y responder. Úsala para controlar de forma activa la placa Arduino.

Serial

Se utiliza para la comunicación entre la placa Arduino y un ordenador u otros dispositivos. Todas las placas Arduino tienen al menos un puerto serie (también conocido como UART o USART): **Serial**. Se comunica a través de los pines digitales 0 (RX) y 1 (TX), así como con el ordenador mediante USB. Por lo tanto, si utilizas estas funciones, no puedes usar los pines 0 y 1 como entrada o salida digital.

Puedes utilizar el monitor del puerto serie incorporado en el entorno Arduino para comunicarte con la placa Arduino**. Haz clic** en el botón del monitor de puerto serie en la barra de herramientas y selecciona la misma velocidad en baudios utilizada en la llamada a begin().

Funciones

begin()
end()
available()
read()
flush()
print()
println()
write()
Serial

Serial.begin (velocidad)

Establece la velocidad de datos en bits por segundo (baudios) para la transmisión de datos en serie. Para comunicarse con el computador, utilice una de estas velocidades: 300, 1200, 2400, 4800, 9600, 14400, 19200, 28800, 38400, 57600 o 115200. Sin embargo, puedes especificar otras velocidades - por ejemplo, para comunicarte a través de los pines 0 y 1 con un componente que requiere una velocidad de transmisión en particular.

Serial.available()

Devuelve el número de bytes (caracteres) disponibles para ser leídos por el puerto serie. Se refiere a datos ya recibidos y disponibles en el buffer de recepción del puerto (que tiene una capacidad de 128 bytes).

Serial.read()

Lee los datos entrantes del puerto serie.

digitalWrite(pin, valor)

Escribe un valor HIGH o LOW hacia un pin digital.

Si el pin ha sido configurado como OUTPUT con pinMode(), su voltaje será establecido al correspondiente valor: 5V (o 3.3V en tarjetas de 3.3V) para HIGH, 0V (tierra) para LOW.
Si el pin es configurado como INPUT, escribir un valor de HIGH con digitalWrite() habilitará una resistencia interna de 20K conectada en pullup (ver el tutorial de pines digitales). Escribir LOW invalidará la resistencia. La resistencia es suficiente para hacer brillar un LED de forma opaca, si los LEDs aparentan funcionar, pero no muy iluminados, esta puede ser la causa. La solución es establecer el pin como salida con la función pinMode().

NOTA: El pin digital número 13 es más difícil de usar que los otros pines digitales porque tiene un LED y una resistencia adjuntos, los cuales se encuentran soldados a la tarjeta, y la mayoría de las tarjetas se encuentran así. Si habilitas la resistencia interna en pullup, proporcionará 1.7V en vez de los 5V esperados, porque el LED soldado en la tarjeta y resistencias bajan el nivel de voltaje, significando que siempre regresará LOW. Si debes usar el pin número 13 como entrada digital, usa una resistencia externa conectada a pulldown.

Parametros
pin: el número de pin
valor: HIGH o LOW

Sentencia switch / case

Como las sentencias **if**, **switch...case** controla el flujo de programas permitiendo a los programadores especificar diferentes códigos que deberían ser ejecutados en función de varias condiciones. En particular, una sentencia switch compara el valor de una variable con el valor especificado en las sentencias case. Cuando se encuentra una sentencia case cuyo valor coincide con dicha variable, el código de esa sentencia se ejecuta.
La palabra clave **break** sale de la sentencia switch, y es usada típicamente al final de cada caso. Si una sentencia break, la sentencia switch continuaría ejecutando las siguientes expresiones ("falling-through") hasta encontrar un break, o hasta llegar al final de la sentencia switch.

Ejemplo

```
 switch (var) {
   case 1:
    //hacer algo cuando sea igual a 1
    break;
   case 2:
    //hacer algo cuando sea igual a 2
    break;
   default:
    // si nada coincide, ejecuta el "default"
    // el "default" es opcional
```

Conociendo los recursos que se usaran para la programación del Arduino, empezamos a escribir el código final que hará el control de direcciones. Así como el de considerar la siguiente información sobre el Modulo Bluetooth, la hoja de datos del RN-41 especifica que la trasmisión del módulo es de 9600 baudios, sin embargo la comunicación del módulo con el Procesador es de 115200 baudios.

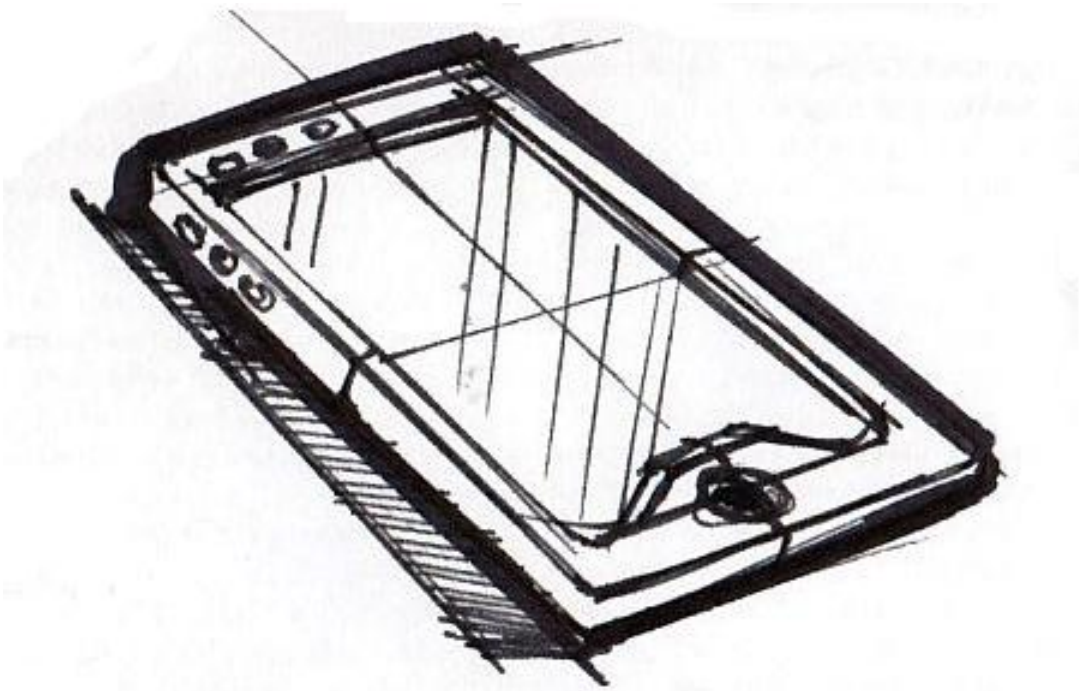
Código del Arduino:

```
//creación de variables
int led1 = 5;
int led2 = 6;
int led3 = 7;
int led4 = 8;
int var=0;

void setup() //inializacion de variables y configuracion de pines
{
       pinMode(led1, OUTPUT); //configuración de pines como salidas
       pinMode(led2, OUTPUT);
       pinMode(led3, OUTPUT);
       pinMode(led4, OUTPUT);
       Serial.begin(115200);    // abre el puerto serie a 115200 bps
}

void loop() //ciclo infinito
{
          if (Serial.available() > 0) // inicia rutina solamente cuando recibe datos por el serial
  {
   var = Serial.read(); //lee los datos del serial y lo guarda en una variable
   digitalWrite(led1, LOW); // manda estados bajos a los pines de salidas
   digitalWrite(led2, LOW);
   digitalWrite(led3, LOW);
   digitalWrite(led4, LOW);
}

     switch(var)// menu de rutinas
       {

case '1'://si la variable var es 1 hace lo siguiente
            {
              digitalWrite(led1, HIGH);// manda un estado alto al pin asignado
              digitalWrite(led2, LOW);// manda un estado bajo al pin asignado
              digitalWrite(led3, HIGH);
              digitalWrite(led4, LOW);
              break; //sale del menu
            }

case '2':// si la variable var es 2 hace lo siguiente
            {
              digitalWrite(led1, LOW);
              digitalWrite(led2, HIGH);
              digitalWrite(led3, LOW);
              digitalWrite(led4, HIGH);
              break;
            }
```

```
 case '3':// si la variable var es 3 hace lo siguiente
            {
              digitalWrite(led1, LOW);
              digitalWrite(led2, HIGH);
              digitalWrite(led3, HIGH);
              digitalWrite(led4, LOW);
              break;
            }
 case '4':// si la variable var es 4 hace lo siguiente
            {
              digitalWrite(led1, HIGH);
              digitalWrite(led2, LOW);
              digitalWrite(led3, LOW);
              digitalWrite(led4, HIGH);
              break;
            }

case '0':// si la variable var es 0 hace lo siguiente
            {
              digitalWrite(led1, LOW);
              digitalWrite(led2, LOW);
              digitalWrite(led3, LOW);
              digitalWrite(led4, LOW);
              break;
            }
       }
```

// fin del Programa

2.1.3 Código Celular



Para el diseño del código del celular debemos analizar qué información se necesita para que podamos controlar el Robot móvil, además de saber que recursos del celular vamos a usar. Nuestro modelo de celular es un SAMSUNG GALAXY Ace (GT-S5830). App Inventor nos facilita muchas cosas y una de ellas es que adapta los recursos a los dispositivos móviles.

App Inventor al ser totalmente gráfico sólo requiere que inicies sesión y hagas tu primer proyecto el cual después de eso te enviará a una ventana donde podrás incluir y diseñar tu aplicación.

En la figura 2.5 se muestra la página de creación de proyectos.

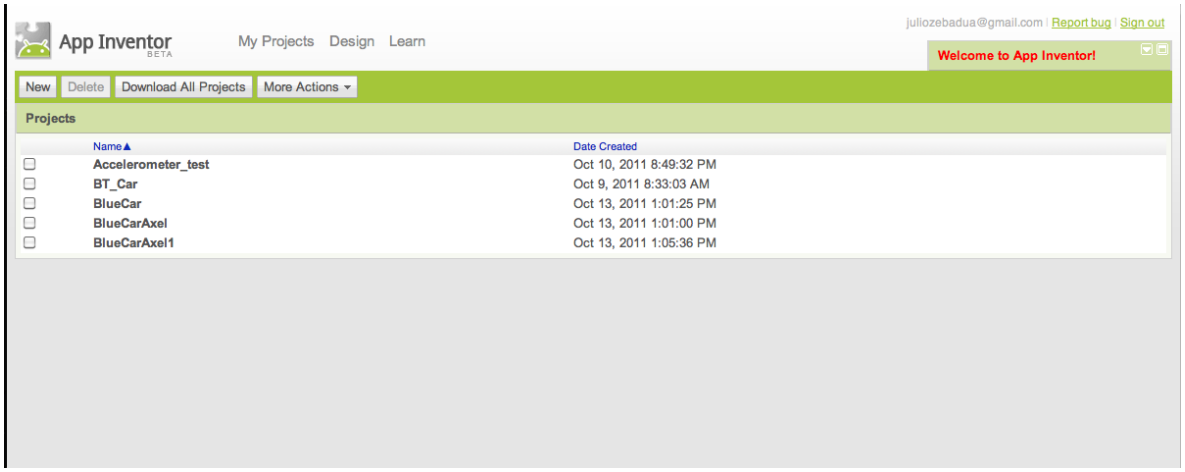


Fig. 2.5

Una vez creado el proyecto te mostrará una nueva página donde podrás diseñar la aplicación (Figura 2.6).

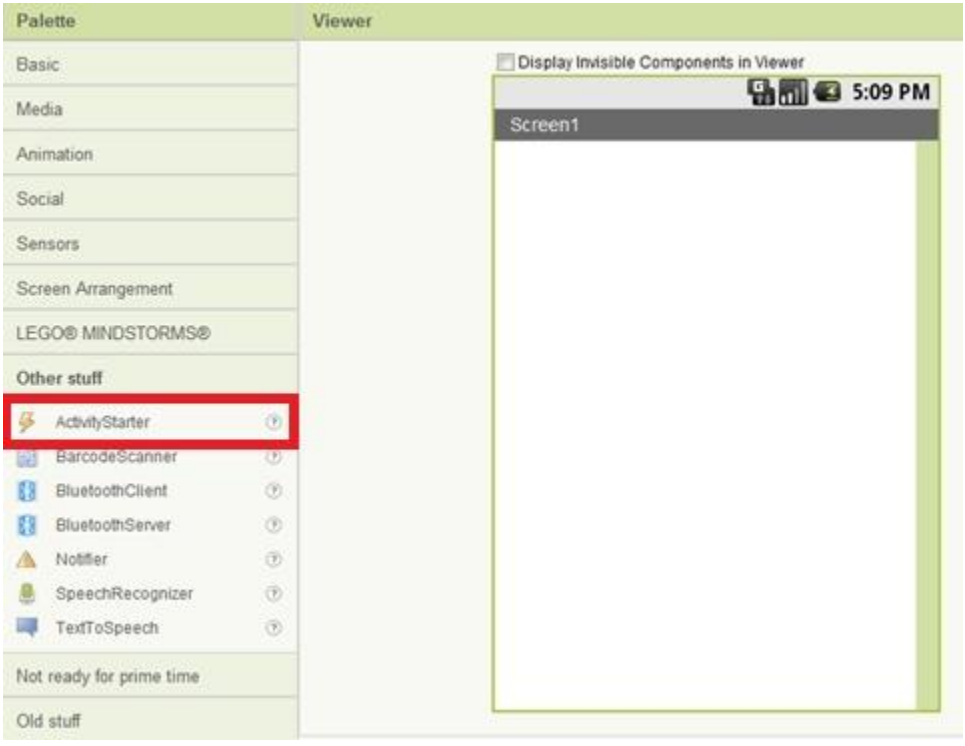


Fig. 2.6

Nuestra forma de programar la aplicación está basada por medio de bloques que dependen de los recursos que deseemos usar y estos bloques contienen funciones la cual les asigna una operación.

Primero que nada y antes de empezar a programar nuestra aplicación debemos hacer la parte gráfica de nuestra aplicación, colocar los botones, imágenes, texto y los recursos que gráficamente queremos mostrar al usuario.

Sólo basta ir a la paleta de la izquierda y escoger las funciones que queremos agregar a nuestra aplicación, las funciones se encuentran organizadas por carpeta y dependiendo del tipo de ésta es donde se encontrarán. Después sólo basta arrastrar hasta la pantalla de nuestro teléfono para colocarlo.

En la figura 2.7 se muestra el diseño de nuestra aplicación.

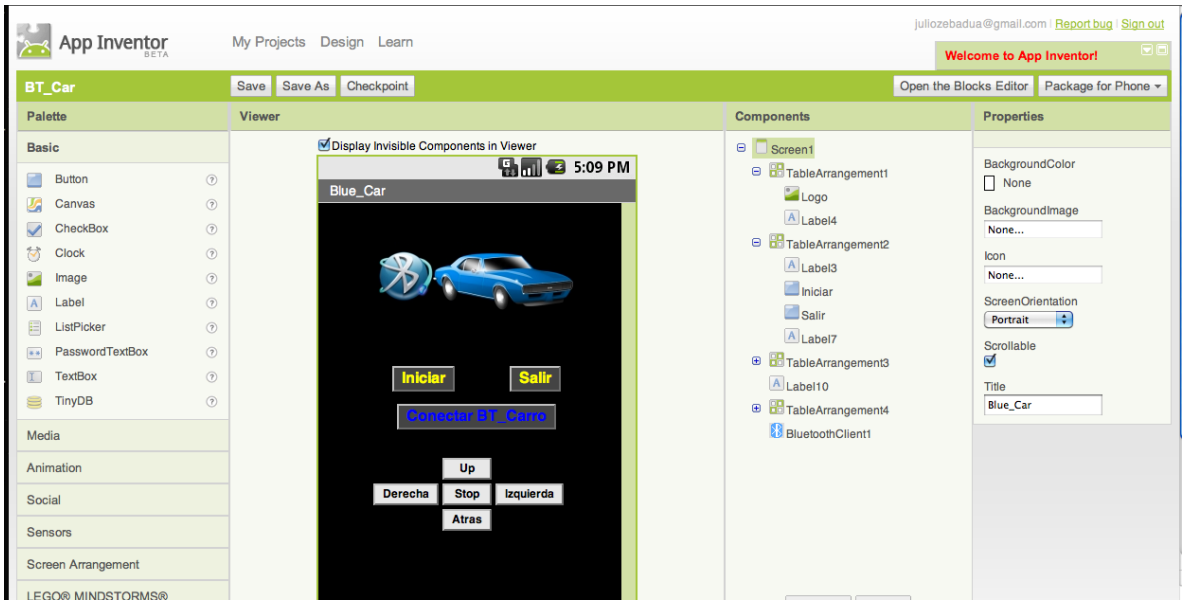


Fig. 2.7

Una vez que hemos creado nuestro entorno gráfico, debemos proceder a unir los bloques crear las rutinas y estructurar nuestro programa. Solo basta en abrir el editor de bloques para realizar estas operaciones (Figura 2.8).

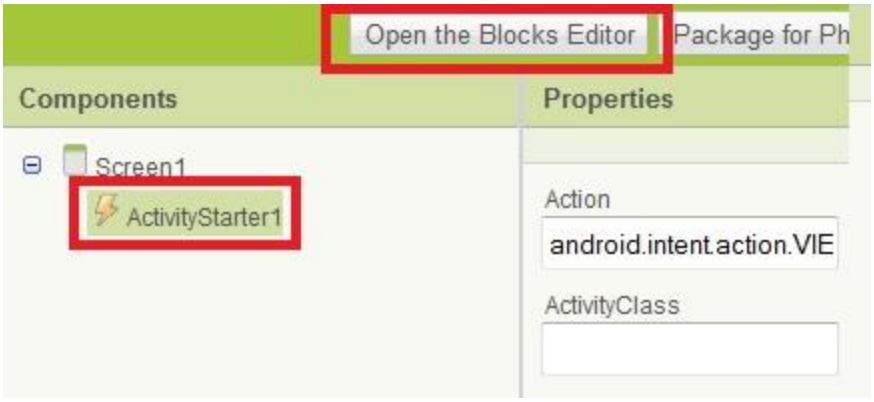


Fig. 2.8

Una vez abierto nuestro editor, nos encontraremos con nuestro menú a la izquierda, donde se encontrarán nuestros recursos que decidimos colocar a nuestro proyecto, tales como los módulos, sensores, botones así como sentencias, rutinas, operaciones y todo lo necesario para estructurar nuestro programa. En la figura 2.9 se encuentra nuestro programa del control del robot móvil.

Ver imagen 2.9.

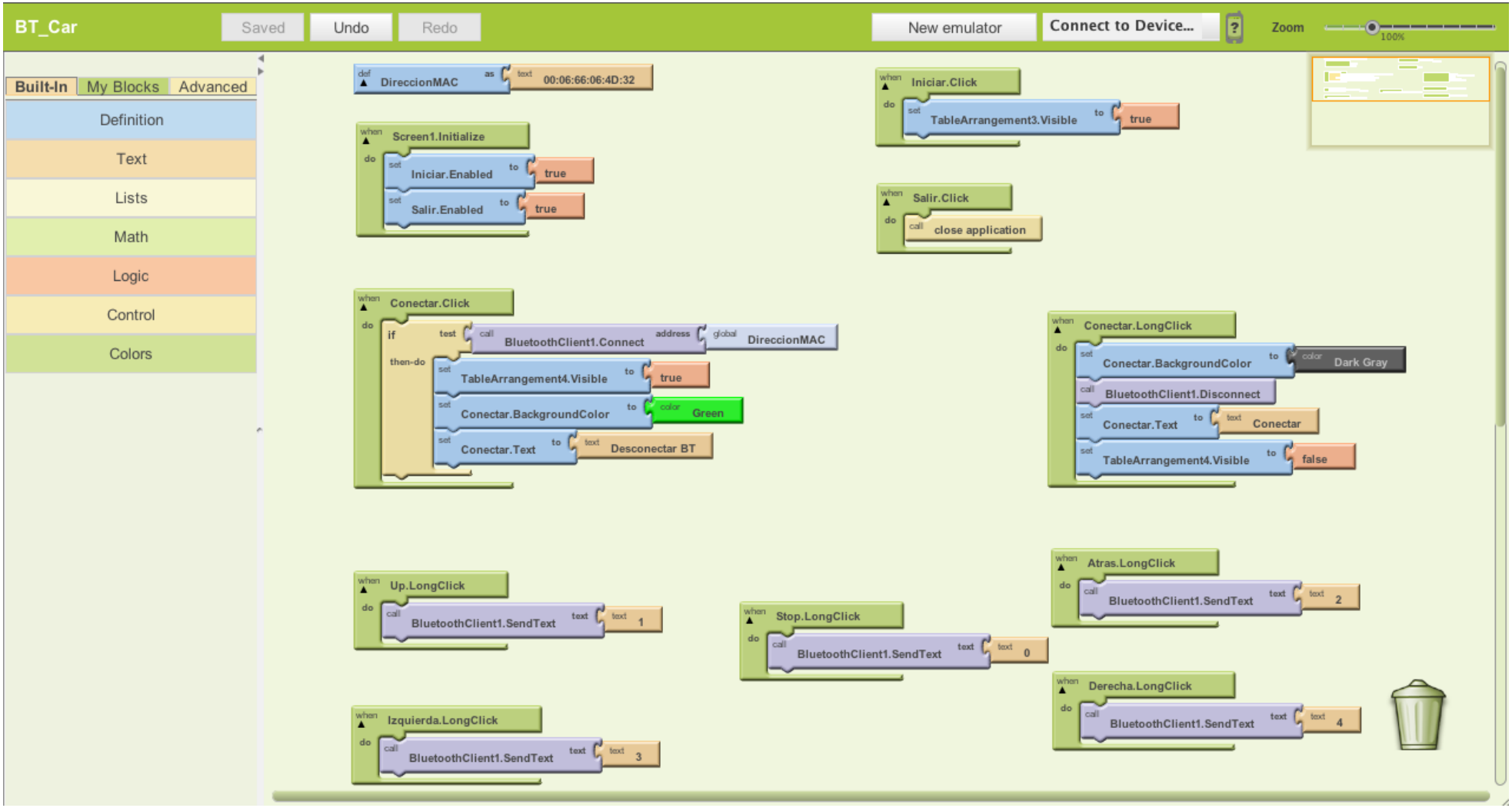

BT_Car
Saved
Undo
Redo
New emulator
Connect to Device...
Zoom
100%
Built-In
My Blocks
Advanced
Definition
Text
Lists
Math
Logic
Control
Colors
def DireccionMAC as text 00:06:66:06:4D:32
when Screen1.Initialize do set Iniciar.Enabled to true set Salir.Enabled to true
when Iniciar.Click do set TableArrangement3.Visible to true
when Salir.Click do call close application
when Conectar.Click do if test call BluetoothClient1.Connect address global DireccionMAC then-do set TableArrangement4.Visible to true set Conectar.BackgroundColor to color Green set Conectar.Text to text Desconectar BT
when Conectar.LongClick do set Conectar.BackgroundColor to color Dark Gray call BluetoothClient1.Disconnect set Conectar.Text to text Conectar set TableArrangement4.Visible to false
when Up.LongClick do call BluetoothClient1.SendText text text 1
when Stop.LongClick do call BluetoothClient1.SendText text text 0
when Atras.LongClick do call BluetoothClient1.SendText text text 2
when Izquierda.LongClick do call BluetoothClient1.SendText text text 3
when Derecha.LongClick do call BluetoothClient1.SendText text text 4

Figura 2.9

Como podemos observar, la estructura del programa es la Siguiente:

Hacemos una variable con el nombre de **DireccionMAC**, esta contendrá la dirección MAC de nuestro modulo Bluetooth, esto nos servirá puesto que será ejecutada en el botón **Conectar**. Debemos de inicializar nuestra pantalla y con esta activar nuestros dos botones **Iniciar** y **Salir**, en la Pestaña **My Blocks**, sacamos nuestra función **Screen1.Inicialize**, dentro de este bloque colocaremos dos bloques más los cuales serán **Iniciar.Enable** y **Salir.Enable** seguido de los bloques lógicos **True**.

Luego debemos de declarar la función de estos botones, dentro de los bloques **Iniciar** y **Salir** le asignaremos la forma de activación llamada **Click**; con la función **Click** obtendremos lo siguiente, cuando presionemos el botón por un instante de tiempo simulando una pulsación, éste activara el botón y las declaraciones que ahí se encuentren, en este caso un tabla oculta, se usó esta tabla para poder colocar nuestros botones a nuestra disposición, si no se hace este tipo de arreglo, App inventor coloca los botones por default en forma de escalera sin posibilidad de colocarlo a nuestra necesidad.

El botón **Salir** tiene asignado un bloque **Close.Aplication** localizado en las funciones de **Control**, esto hará que al hacer click a ese botón cierre la aplicación en nuestro celular.

Una vez que hayamos hecho click al botón **Iniciar** y nos muestre nuestro arreglo, en nuestra pantalla aparecerá el Botón **Conectar**, el cual posee los siguientes bloques bajo un condicional **if**:

- **BluetoothClient1.Conect**: el cual conecta el modulo Bluetooth a nuestro robot por medio de la dirección MAC asignada.
- **TableArrangement4.Visible**: el cual muestra la tabla que contiene los botones de dirección.
- Seguido de dos bloques cuya función es cambiar el color y texto del botón conectar, esto con la finalidad de hacer que ese mismo botón sirva de desconexión del módulo con la aplicación, entender que no salimos de la aplicación.

Es por eso que declaramos de nuevo el botón **Conectar** pero ahora asignándole la función de **LongClick**, de esta manera aseguramos que si por accidente pasamos el dedo por ahí o pulsamos no desconectara nuestro dispositivo, hasta que presionemos el botón que ahora se llama **Desconectar** por un instante de tiempo.

Si observamos hacemos todo lo contario en el bloque **Conectar.Click**, con esto tenemos una función que nos proporcionara la opción de desconectar nuestro celular del robot sin salir de la aplicación.

Una vez conectado el celular y el robot móvil, se mostrarán otros arreglos de tablas donde se encuentran los bloques del botón **Up**, **Atrás**, **Derecha**, **Izquierda**.

Dentro de estos bloques encontramos la función **BluetoothClient1.SendText**, la cual se encarga de enviar información de carácter string, seguido de un bloque tipo texto, cuyo valor colocado en el bloque es el que envía por la comunicación Bluetooth.

UNIDAD 3

PRUEBAS Y RESULTADOS

Las pruebas se realizaron de manera independiente para cada una de las etapas, de esta manera se obtuvieron los mejores resultados. La etapa de control es la encargada de gestionar los movimientos del robot. El control del robot se realiza mediante la inserción de caracteres recibidos en la comunicación Bluetooth, de esta manera se ingresan códigos al Arduino, lo que permite realizar una comparación efectiva para ejecutar las instrucciones.

Tabla de verdad

Carácter	Acción
0	Parar
1	Adelante
2	Atrás
3	Izquierda
4	Derecha

La tabla de verdad indica los códigos que se ingresaron al Arduino, de esta manera dependiendo del botón que se haya presionado en el dispositivo móvil este enviará un carácter con un valor numérico, dependiendo del valor el Arduino ejecutará su respectiva acción. La placa de la etapa de control funciona con una batería de 9v a 1 Ampere/h.

3.1 Etapa Del Puente H

Una vez comprobada la etapa de control, se conecta al puente H para verificar su funcionamiento para comprobar los máximos y mínimos consumos de corriente, de esta forma podemos saber el tiempo máximo de funcionamiento dependiendo del tipo de consumo que éste realice. Sabiendo que el consumo de un motor con su respectiva caja de engranaje es de:

Motorreductor	Consumo Mínimo	Consumo Nominal	Consumo Máximo
	65mA	150mA	670mA

Tabla 3.1. Consumo de Corrientes

La alimentación del puente H de control de movimiento del robot y armas se lo realiza con una batería de 7.2v 300mA/h de Níquel-Cadmio, por lo cual, mediante pruebas realizadas se obtuvo diferentes tiempos de funcionamiento, el consumo de las baterías se puede expresar por tiempos cortos, medios o prolongados como se muestra a continuación.

	Carga de la Batería	Consumo	Minutos
Consumo Mínimo	300mA	75mA	3.5 horas
Consumo Nominal	300mA	150mA	1.70 horas
Consumo Máximo	300mA	670mA	20 minutos

Tabla 3.2.- Tiempo de funcionamiento del robot

Con la tabla anterior tenemos datos aproximados de durabilidad de las baterías cuando el robot está en funcionamiento, de esta manera nos permite llevar el control de tiempos para posibles cambios o tiempos de operatividad.

3.2 Bluetooth

Se realizaron pruebas inalámbricamente con el robot a una distancia aproximada de 4 – 6 metros. Las cuales aprobó satisfactoriamente evitando el común cruce de radiofrecuencias que podrían existir en la misma área de operación. De igual forma se llevaron pruebas de alcance máximo llegando a un radio de 16 metros sin ningún problema de comunicación. Para lo cual se muestra una tabla de consumo de corrientes y voltajes del módulo Bluetooth:

Modulo Bluetooth	**Consumo Mínimo**	**Consumo Máximo**
Consumo de Voltaje	3.3v	5.0v
Consumo de Corriente	25mA	400mA

Tabla 3.3. Consumo de corrientes del Módulo Bluetooth

3.3 Pruebas De La Aplicación Del Celular

Una vez instalado en el dispositivo móvil se debe encender primero el modulo Bluetooth antes de correr la aplicación, de esta manera se evita el error de reconocimiento de Antena.

La aplicación no presenta algún informe de error, o desconfiguración y todos los botones como sus funciones corren sin ningún problema.

INSTITUTO TECNOLOGICO DE TUXTLA GUTIERREZ

SUBDIRECCIÓN ACADÉMICA
DEPARTAMENTO DE INGENIERIA ELECTRICA Y ELECTRONICA
SEGUIMIENTO DE PROYECTO DE RESIDENCIAS PROFESIONALES

ALUMNO: Zebadúa Vázquez Julio César — No. DE CONTROL: 07270354

NOMBRE DEL PROYECTO: Control de un Robot Móvil Usando un Celular con Tecnología Bluetooth. — EMPRESA: Instituto Tecnológico de Tuxtla Gutiérrez

ASESOR EXTERNO: Dr. Madain Pérez Patricio — ASESOR INTERNO: Ing. Francisco Ramón Sánchez

PERIODO DE REALIZACIÓN: Agosto- Diciembre 2011

		SEMANAS														
ACTIVIDAD		**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**
Recopilación de información	P	■	■													
	R	x	x	x												
Diseño de prototipo del Robot Móvil	P			■	■	■										
	R				x	x	x									
Diseño del aplicación del celular	P						■	■	■							
	R						x	x	x	x	x					
Construction de los Prototipos	P									■	■					
	R									x	x	x				
Pruebas de resultados	P											■	■			
	R												x	x		
Corrección de diseño y bugs de los sistemas	P													■	■	
	R														x	
Correcciones y entrega del proyecto de residencia	P															■
	R															x

OBSERVACIONES: **REPORTE FINAL 14-15 DE DIC. 2011**		**19 y 20 DE SEPTIEMBRE 2011**	**17 y 18 DE OCTUBRE 2011**	**14 y 15 NOVIEMBRE 2011**
ENTREGA DE REPORTES	Docente	Ing. Francisco Ramón Sánchez		
	Alumno	Zebadúa Vázquez Julio César		
	Jefe Depto.	Ing. Vicente de León Orozco		

INSTITUTO TECNOLOGICO DE TUXTLA GUTIERREZ

SUBDIRECCIÓN ACADÉMICA
DEPARTAMENTO DE INGENIERIA ELECTRICA Y ELECTRONICA
SEGUIMIENTO DE PROYECTO DE RESIDENCIAS PROFESIONALES

ALUMNO: Pedraza Toledo Miguel Jair — No. DE CONTROL: 07270336

NOMBRE DEL PROYECTO: Control de un Robot Móvil Usando un Celular con Tecnología Bluetooth. — EMPRESA: Instituto Tecnológico de Tuxtla Gutiérrez

ASESOR EXTERNO: Dr. Madain Pérez Patricio — ASESOR INTERNO: Ing. Francisco Ramón Sánchez

PERIODO DE REALIZACIÓN: Agosto- Diciembre 2011

		SEMANAS														
ACTIVIDAD		**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**
Recopilación de información	P	■	■													
	R	x	x	x												
Diseño de prototipo del Robot Móvil	P			■	■	■										
	R				x	x	x									
Diseño del aplicación del celular	P						■	■	■							
	R						x	x	x	x	x					
Construction de los Prototipos	P									■	■					
	R									x	x	x				
Pruebas de resultados	P											■	■			
	R												x	x		
Corrección de diseño y bugs de los sistemas	P													■	■	
	R														x	
Correcciones y entrega del proyecto de residencia	P															■
	R															x

OBSERVACIONES: **REPORTE FINAL 14-15 DE DIC. 2011**		**19 y 20 DE SEPTIEMBRE 2011**	**17 y 18 DE OCTUBRE 2011**	**14 y 15 NOVIEMBRE 2011**
ENTREGA DE REPORTES	Docente	Ing. Francisco Ramón Sánchez		
	Alumno	Pedraza Toledo Miguel Jair		
	Jefe Depto.	Ing. Vicente de León Orozco		

CONCLUSIONES

La creación de nuevas tecnologías y la implementación de estas en nuestra vida cotidiana por medio de los dispositivos digitales que usamos, es parte fundamental de la evolución que la humanidad tiene. Hoy en día los sistemas automatizados y controlados representan una mano de obra considerada. La carrera digital es una fuente de desarrollo para instituciones y países que desean una superación económica, productiva y educacional, con la existencia de nuevas plataformas y de la necesidad de evolucionar nuestras formas de trabajo y nuevas aplicaciones para resolver situaciones, hace que sea necesaria la creación o implementación de nuevas formas de control de los sistemas digitales.

Este proyecto de residencia profesional busca demostrar que un celular puede funcionar para controlar y diseñar nuevas formas de control y justificar que es un medio óptimo para el control de sistemas específicos y que dependiendo de la aplicación puede considerarse una forma eficiente de desarrollo, dotado de módulos de comunicación Bluetooth puede ser usado para la interconexión de diversos dispositivos y controlar como verdadero mando a distancias convirtiendo un medio pensado en la comunicación e interconexión de usuarios, de herramientas personales, en una plataforma de control y en una verdadera herramienta de trabajo.

La conclusión de este proyecto de residencia se da a conocer en los siguientes puntos:

- El uso de la plataforma Arduino como herramienta de control, es fundamental a causa de la reducción de circuitería, líneas de código como la facilidad y soporte que este brinda.

- Android es un sistema operativo open source, el cual nos brinda herramientas para desarrolladores, además de no tener tantas restricciones como los demás entornos, de esta manera remarcamos que se convierte en una alternativa de trabajo si se desea tener el mayor control y libertad en el diseño de aplicaciones o implementación de nuestros diseños electrónicos.

- El proyecto se diseñó utilizando diferentes lenguajes de programación entre ellos lenguaje C y Lenguaje Java.

- Podemos decir que se ha logrado un nivel básico de comunicación entre distintos sistemas, lo que demuestra que es posible la comunicación entre microcontroladores y celulares usando módulos que permitan la interacción, además de tener una comunicación completamente sin cables.

RECOMENDACIONES

- Para la construcción se recomienda la utilización de materiales reciclados los cuales son muy fáciles de conseguir como por ejemplo: para la estructura, motores de CC y cajas de engranajes que se encuentran en carros para niños, los cuales abaratan los costos de la construcción.

- Con los circuitos electrónicos se recomienda tratar en lo posible de separar el sistema. De esta manera saber en qué parte del circuito estamos trabajando y si existe alguna avería tener una forma más fácil para darle solución, además nos ayuda a ubicar los circuitos de una manera más fácil dentro de la estructura del robot.

- Antes de poner en funcionamiento el robot verificar que los elementos estén conectados y ubicados de la manera correcta.

- Es recomendable revisar las conexiones de entrada y salida del Arduino. De la misma forma se recomienda verificar las conexiones de las baterías y su conexión hacia los motores para que estos tengan el giro coordinado para su movimiento.

- Se debe tener cuidado con el manejo del módulo Bluetooth ya que es un componente de fácil destrucción por su tamaño.

- La aplicación del celular la podemos utilizar en cualquier equipo de telefonía cuyo sistema operativo sea Android.

- Para la alimentación del circuito debemos utilizar un voltaje máximo de 9 o el sugerido en la hoja de datos de cada circuito regulador de voltaje.

- Verificar que antes de conectar el robot al celular, debe de estar encendido el modulo Bluetooth de éste, de otra forma la aplicación marcará error.

- La información recabada en la contruccion de este proyecto, es una base para futuras aplicaciones o actualizaciones, el uso del Bluetooth como forma de comunicación es una gama amplia de posibilidades en el diseño e implementaciones.

- Nuestro robot puede ser equipado con nuevo sistemas, como brazos, sensores de proximidad, GPS y herramientas que le brinden autonomía y diversidad de aplicación.

FUENTES DE INFORMACIÓN

LIBROS:

DAVID WOLBER, HAL ABELSON, ELLEN SPERTUS & LIZ LOONEY. App Inventor Create Your Own Android Apps. O'REILLY. Primera Edición. CANADA 2011.

OWEN BISHOP (B.S.). Robot Builder's Cookbook: Build and Design Your Own Robots. Newnes. Primera edición, 2007.

DAVID COOK. Robot Building for Beginners (Technology in Action). Apress. Segunda Edición, 2009.

MASSIMO BANZI. Getting Started with Arduino (Make: Projects). Make. Primera Edición, 2008.

MICHAEL MCROBERTS. Beginning Arduino . Apress. Primera Edición, 2010.

ALBERT S. HUANG. Bluetooth Essentials for Programmers. Cambridge University Press. Primera Edición, 2007.

PÁGINAS DE INTERNET:

-Recursos y Soporte Appinventor
http://www.appinventorbeta.com/about/

-Compra, recursos Bluetooth RN-41
http://www.sparkfun.com/products/582

-Recursos e información sobre Arduino
http://www.arduino.cc/

ANEXOS

Class 1 Bluetooth® Module

Features

- Fully qualified Bluetooth 2.1/2.0/1.2/1.1 module
- Bluetooth v2.0+EDR support
- Postage stamp sized form factor, 13.4mm x 25.8 mm x2mm
- Low power *(30mA connected,, <10mA sniff mode)*
- UART (SPP or HCI) and USB (HCI only) data connection interfaces.
- Sustained SPP data rates - 240Kbps (slave), 300Kbps (master)
- HCI data rates - 1.5Mbps sustained, 3.0Mbps burst in HCI mode
- Embedded Bluetooth stack profiles included *(requires no host stack)*: GAP, SDP, RFCOMM and L2CAP protocols, with SPP and DUN profile support.
- Bluetooth SIG Qualified, End Product Listing
- Castellated SMT pads for easy and reliable PCB mounting
- Class 1 high power amplifier with on board ceramic RF chip antenna.
    - Certifications: FCC, ICS, CE
    - Environmentally friendly, RoHS compliant

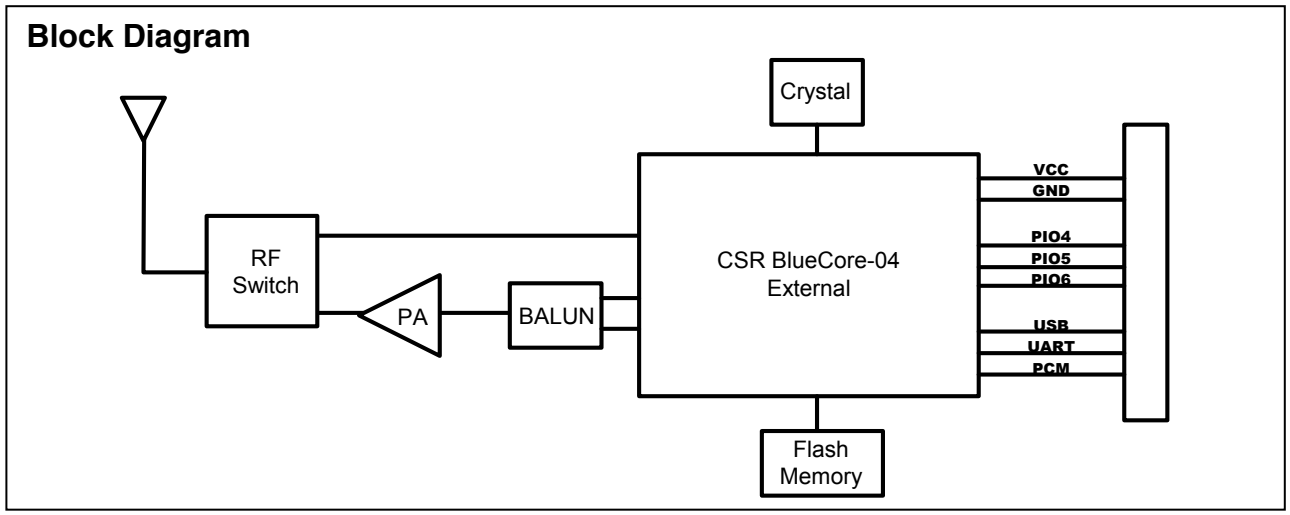
Applications

- Cable replacement
- Barcode scanners
- Measurement and monitoring systems
- Industrial sensors and controls
- Medical devices
- Asset tacking

Description

The RN41 is a small form factor, low power, highly economic Bluetooth radio for OEM's adding wireless capability to their products. The RN41 supports multiple interface protocols, is simple to design in and fully certified, making it a complete embedded Bluetooth solution. With its high performance on chip antenna and support for Bluetooth® Enhanced Data Rate (EDR), the RN41 delivers up to 3 Mbps data rate for distances to 100M.. The RN41 is the perfect product for engineers wanting to add wireless capability to their product but don't want to spend significant time and money developing Bluetooth specific hardware and software.

Block Diagram



Overview

- Baud rate speeds: 1200bps up to 921Kbps, non-standard baud rates can be programmed.
- Class 1 radio, 330' (100m) distance, 15dBm output transmitter, -80dBm typical receive sensitivity
- Frequency 2402 ~ 2480MHz,
- FHSS/GFSK modulation, 79 channels at 1MHz intervals
- Secure communications, 128 bit encryption
- Error correction for guaranteed packet delivery
- UART local and over-the-air RF configuration
- Auto-discovery/pairing requires no software configuration (instant cable replacement).
- Auto-connect master, IO pin (DTR) and character based trigger modes

Environmental Conditions

Parameter	Value
Temperature Range (Operating)	-40 °C ~ 85 °C
Temperature Range (Storage)	-40 °C ~ 85 °C
Relative Humidity (Operating)	≤90%
Relative Humidity (Storage)	≤90%

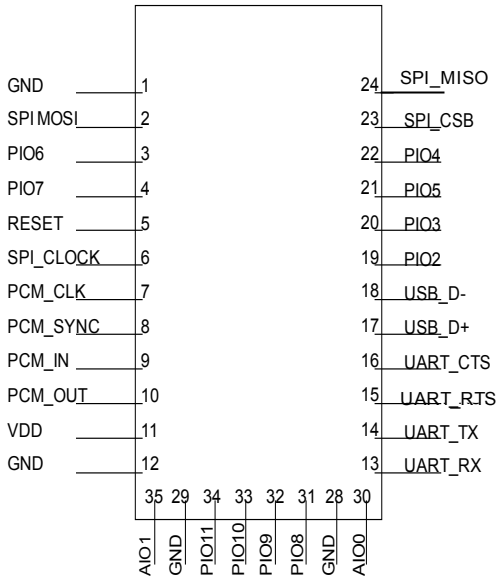
Electrical Characteristics

Parameter	Min	Typ.	Max.	Unit
Supply Voltage (DC)	3.0	3.3	3.6	V
RX Supply Current		35	60	mA
TX Supply Current		65	100	mA
Average power consumption				
Standby/Idle (default settings)		25		mA
Connected (normal mode)		30		mA
Connected (low power Sniff)		8		mA
Standby/Idle (Deep sleep enabled)	250uA	2.5		mA

Radio Characteristics

Parameter	Freq. (GHz)	Min	Typ	Max	Bluetooth Specification	Units
Sensitivity @ 0.1%BER	2.402	-	-80	-86	≤ -70	dBm
	2.441	-	-80	-86		dBm
	2.480	-	-80	-86		dBm
RF Transmit Power	2.402	15.0	16.0		≤ 20	dBm
	2.441	15.0	16.0			dBm
	2.480	15.0	16.0			dBm
Initial Carrier Frequency Tolerance	2.402	-	5	75	75	kHz
	2.441	-	5	75		kHz
	2.480	-	5	75		kHz
20dB bandwidth for modulated carrier		-	900	1000	≤ 1000	kHz
Drift (Five slots packet)		-	15	-	40	kHz
Drift Rate		-	13	-	20	kHz
$\Delta f1_{avg}$ Max Modulation	2.402	140	165	175	>140	kHz
	2.441	140	165	175		kHz
	2.480	140	165	175		kHz
$\Delta f2_{avg}$ Min Modulation	2.402	140	190	-	115	kHz
	2.441	140	190	-		kHz
	2.480	140	190	-		kHz

Pin Description

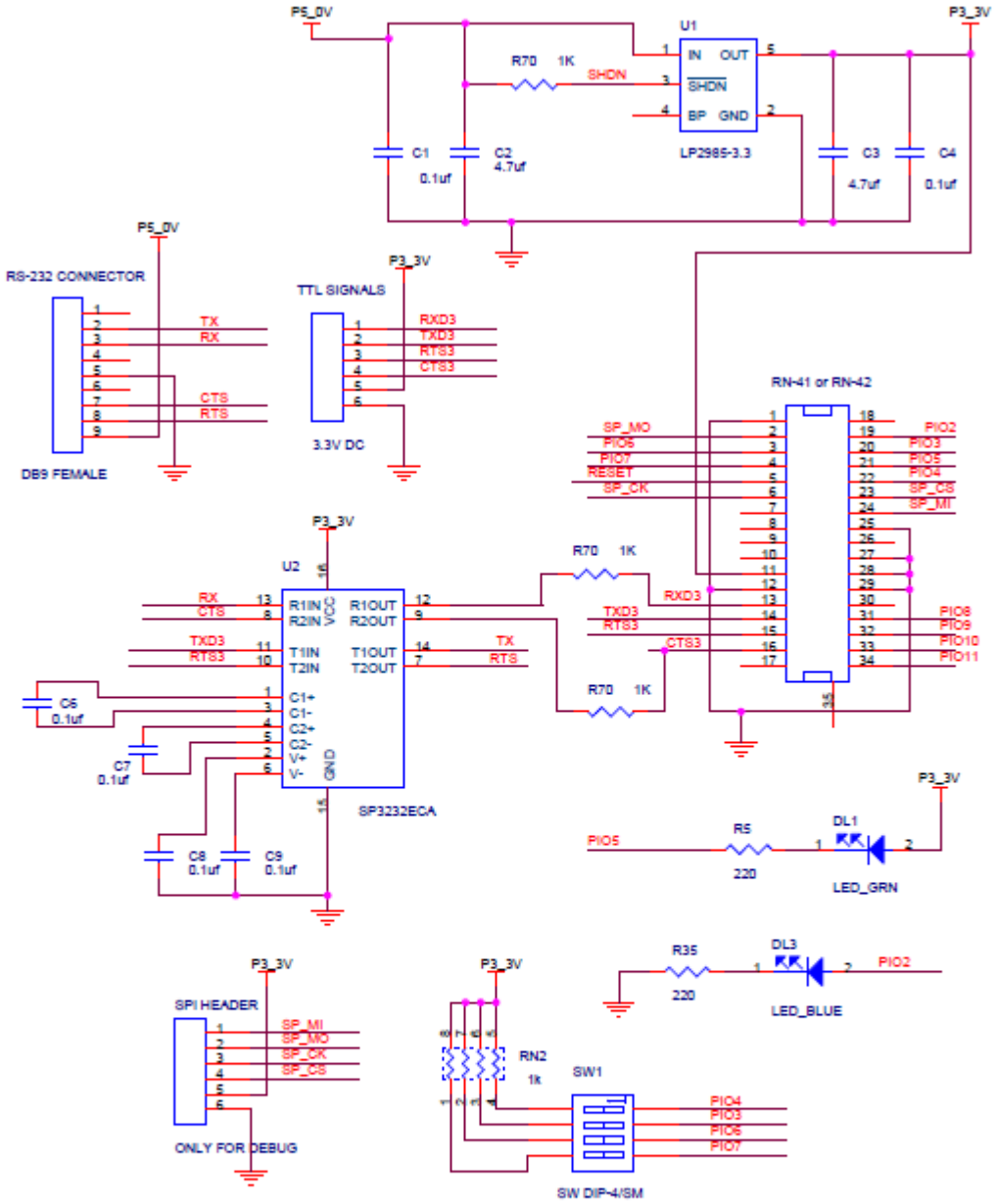


Pin	Name	Description	Default
1	GND		
2	SPI MOSI	Programming only	No Connect
3	PIO6	Set BT master (HIGH=auto-master mode)	Input to RN41with weak pulldown
4	PIO7	Set Baud rate (HIGH = force 9600, LOW = 115K or firmware setting)	Input to RN41 with weak pulldown
5	RESET	Active LOW reset	Input to RN41 with 1K pullup
6	SPI_CLK	Programming only	No Connect
7	PCM_CLK	PCM interface	No Connect
8	PCM_SYNC	PCM interface	No Connect
9	PCM_IN	PCM interface	No Connect
10	PCM_OUT	PCM interface	No Connect
11	VDD	3.3V regulated power input	
12	GND		
13	UART_RX	UART receive Input	Input to RN41
14	UART_TX	UART transmit output	High level output from RN41
15	UART_RTS	UART RTS, goes HIGH to disable host transmitter	Low level output from RN41
16	UART_CTS	UART CTS, if set HIGH, disables transmitter	Low level input to RN41
17	USB_D+	USB port	Pull up 1.5K when active
18	USB_D-	USB port	
19	PIO2	Status, HIGH when connected, LOW otherwise	Output from RN41
20	PIO3	Auto discovery = HIGH	Input to RN41 with weak pulldown
21	PIO5	Status, toggles based on state, LOW on connect	Output from RN41
22	PIO4	Set factory defaults	Input to RN41 with weak pulldown
23	SPI_CSB	Programming only	No Connect
24	SPI_MISO	Programming only	No Connect
25	GND		
26	NC	RF pad keep all traces and planes clear.	
27-29	GND		
30	AIO0	Optional analog input	Not Used
31	PIO8	Status (RF data rx/tx)	Output from RN41
32	PIO9	IO	Input to RN41 with weak pulldown
33	PIO10	IO (remote DTR signal)	Input to RN41 with weak pulldown
34	PIO11	IO (remote RTS signal)	Input to RN41 with weak pulldown
35	AIO1	Optional analog input	Not Used

Digital I/O Characteristics

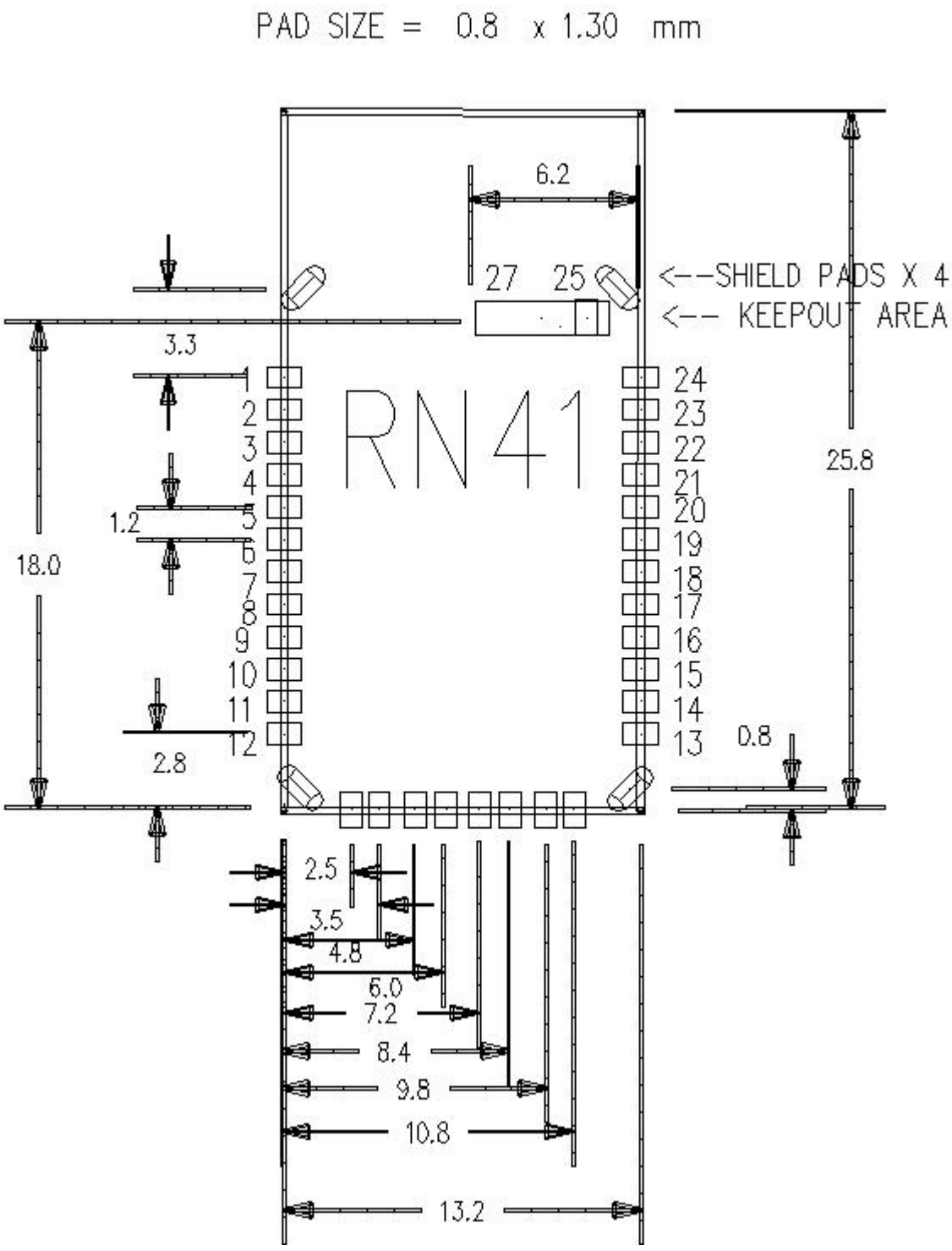
2.7V ≤ VDD ≤ 3.0V	Min	Typ.	Max.	Unit
Input logic level LOW	-0.4	-	+0.8	V
Input logic level HIGH	0.7VDD	-	VDD+0.4	V
Output logc level LOW	-	-	0.2	V
Output logic level HIGH	VDD-0.2	-	-	V
All I/O's (except reset) default to weakpull down	+0.2	+1.0	+5.0	uA

Typical Application Circuit



SIGNAL	FUNCTION
PIO2	CONNECT- HIGH
PIO3	AUTO PAIRING
PIO4	FACTORY RESET
PIO5	STATUS LED
PIO6	AUTO MASTER/DTR
PIO7	BAUDRATE

Module Dimensions



Design Concerns

1. ***Reset circuit***. RN-41 contains a 1k pullup to VCC, the polarity of reset on the RN41 is ACTIVE LOW. A power on reset circuit with delay is OPTIONAL on the reset pin of the module. It should only be required if the input power supply has a very slow ramp, or tends to bounce or have instability on power up. Often a microcontroller or embedded CPU IO is available to generate reset once power is stable. If not, there are many low cost power supervisor chips available, such as MCP809, MCP102/121, and Torex XC61F.

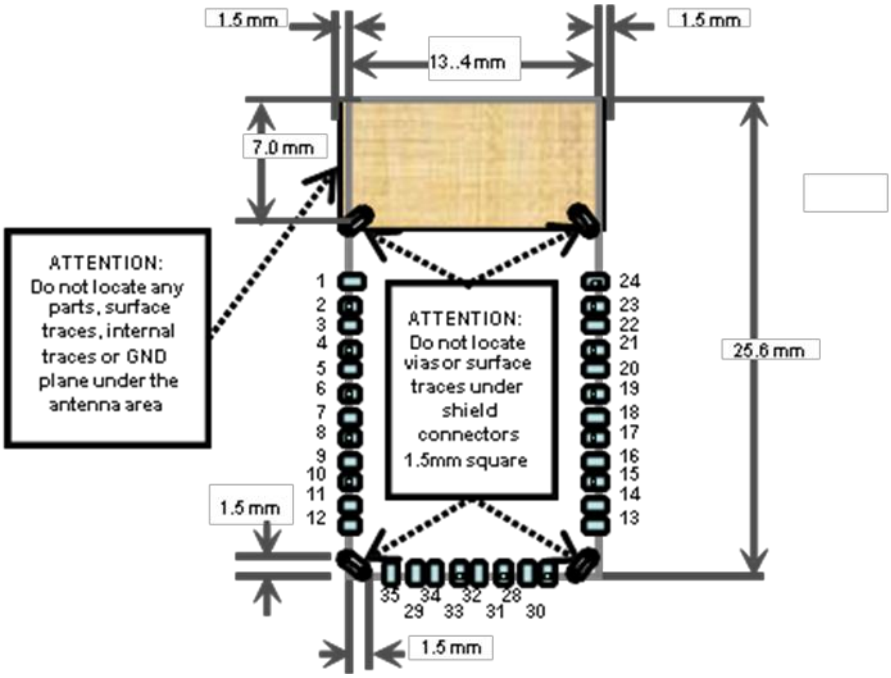
2. ***Factory reset PIO4.*** It is a good idea to connect this pin to a switch, or jumper, or resistor, so it can be accessed. This pin can be used to reset the module to FACTORY DEFAULTS and is often critical in situations where the module has been mis-configured. To set Factory defaults start HIGH, then toggle times.

3. ***Connection status.*** PIO5 is available to drive an LED, and blinks at various speeds to indicate status. PIO2 is an output which directly reflects the connection state, it goes HIGH when connected, and LOW otherwise.

4. **HCI mode.** The RN41 module must be loaded with special firmware to run in HCI mode. When in HCI mode the standard SPP/DUN applications are disabled.

5. ***Using SPI bus for flash upgrade.*** While not required, this bus is very useful for configuring advanced parameters of the Bluetooth modules, and is required for upgrading the firmware on modules. The suggested ref-design shows a 6pin header which can be implemented to gain access to this bus. A minimum-mode version could just use the SPI signals (4pins) and pickup ground and VCC from elsewhere on the design.

6. ***Minimizing Radio interference.*** When laying out the carrier board for the RN41 module the areas under the antenna and shielding connections should not have surface traces, GND planes, or exposed vias. (See diagram to right) For optimal radio performance the antenna end of RN41 module should protrude 5mm past any metal enclosure.



7. ***Soldering Reflow Profile.***
   - Lead-Free Solder Reflow
   - Temp: 230 degree C , 30-40 seconds, Peak 250 degree C maximum.
   - Preheat temp: 165 +- 15 degree C, 90 to 120 seconds.
   - Time: Single Pass, One Time

Compliance Information

Category	Country	Standard
Radio	USA	FCC CFR47 Part 15 C, para 15.247
	FCC ID:	T9J-R41-1
	EUROPE	EN 300 328-1
		EN 300 328-2 2.4GHz
	CANADA	IC RSS-210 low power comm. device
	IC Canada ID:	6514A-RN411
EMC	USA	FCC CFR47 Part 15 subclass B
	EUROPE	EN 55022 Class B radiated
		EN61000-4-2 ESD immunity
		EN61000-4-3 radiated field
		EN61000-4-6 RF immunity
		EN61000-4-8 power magnetic immunity
Bluetooth	LISTED	B013180
Environmental	RoHS	RoHS compliant

Ordering Information

Part Number	Description
RN-41	Standard Application firmware (SPP/DUN Master and Slave)
RN-41-H	HCI firmware (HCI over H4 UART)
RN-41-U	USB firmware (HCI over USB port, slave device at 12Mbps rate)
For other configurations, contact Roving Networks directly.	

Visit http://www.rovingnetworks.com/buynow.php for current pricing and a list of distributors carrying our products.

The Bluetooth trademark and logo are registered trademarks and are owned by the Bluetooth SIG, Inc. All other trademarks are property of their respective owners.

Roving Networks reserves the right to make corrections, modifications, and other changes to its products, documentation and services at any time. Customers should obtain the latest relevant information before placing orders and should verify that such information is current and complete.

Roving Networks assumes no liability for applications assistance or customer product design. Customers are responsible for their products and applications using Roving Networks components. To minimize the risks associated with customer products and applications, customers should provide adequate design and operating safeguards.

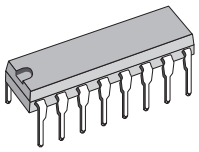
Roving Networks products are not authorized for use in safety-critical applications (such as life support) where a failure of the Roving Networks product would reasonably be expected to cause severe personal injury or death, unless officers of the parties have executed an agreement specifically governing such use.

SGS-THOMSON MICROELECTRONICS

L293B
L293E

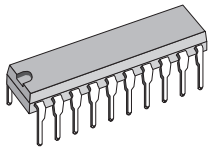
PUSH-PULL FOUR CHANNEL DRIVERS

- OUTPUT CURRENT 1A PER CHANNEL
- PEAK OUTPUT CURRENT 2A PER CHANNEL (non repetitive)
- INHIBIT FACILITY
- HIGH NOISE IMMUNITY
- SEPARATE LOGIC SUPPLY
- OVERTEMPERATURE PROTECTION



DIP16

ORDERING NUMBER : L293B



POWERDIP (16 + 2+ 2)

ORDERING NUMBER : L293E

DESCRIPTION

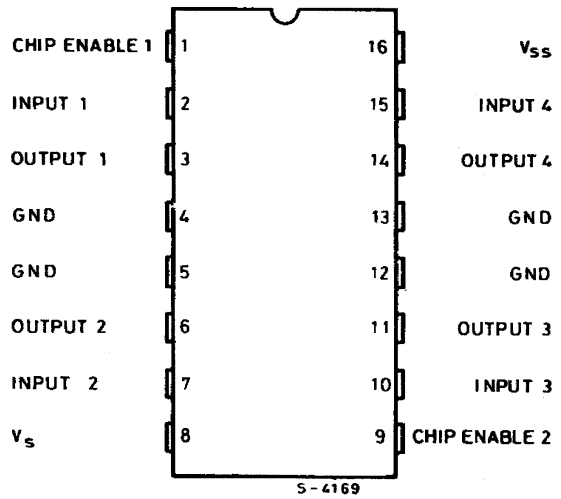
The L293B and L293E are quad push-pull drivers capable of delivering output currents to 1A per channel. Each channel is controlled by a TTL-compatible logic input and each pair of drivers (a full bridge) is equipped with an inhibit input which turns off all four transistors. A separate supply input is provided for the logic so that it may be run off a lower voltage to reduce dissipation.

Additionally, the L293E has external connection of sensing resistors, for switchmode control.

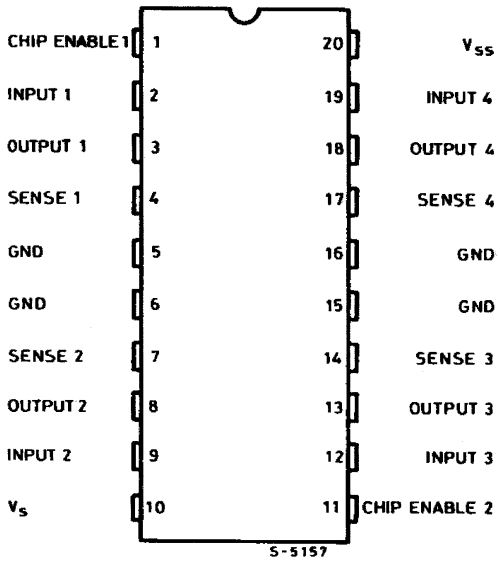
The L293B and L293E are package in 16 and 20-pin plastic DIPs respectively ; both use the four center pins to conduct heat to the printed circuit board.

PIN CONNECTIONS

DIP16 - L293B

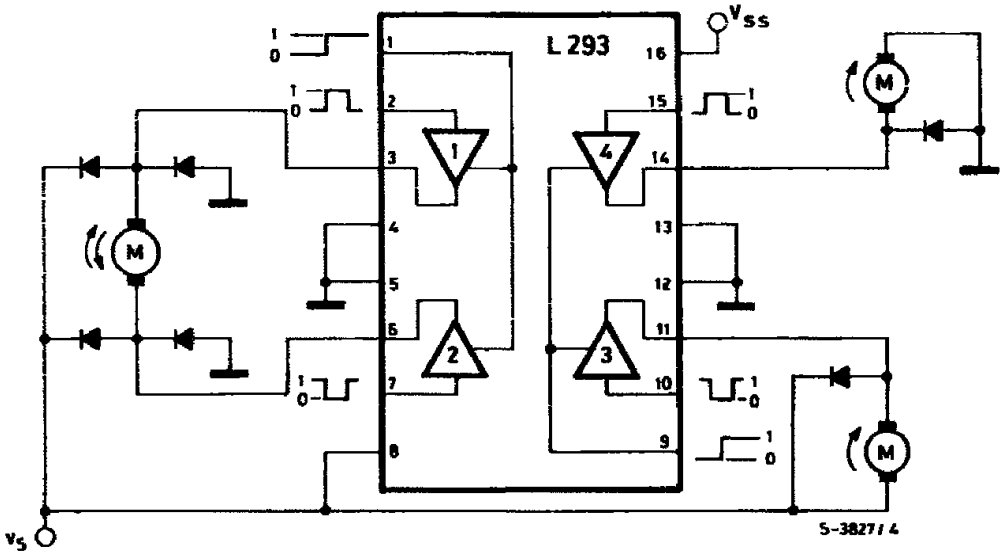


POWERDIP (16+2+2) - L293E

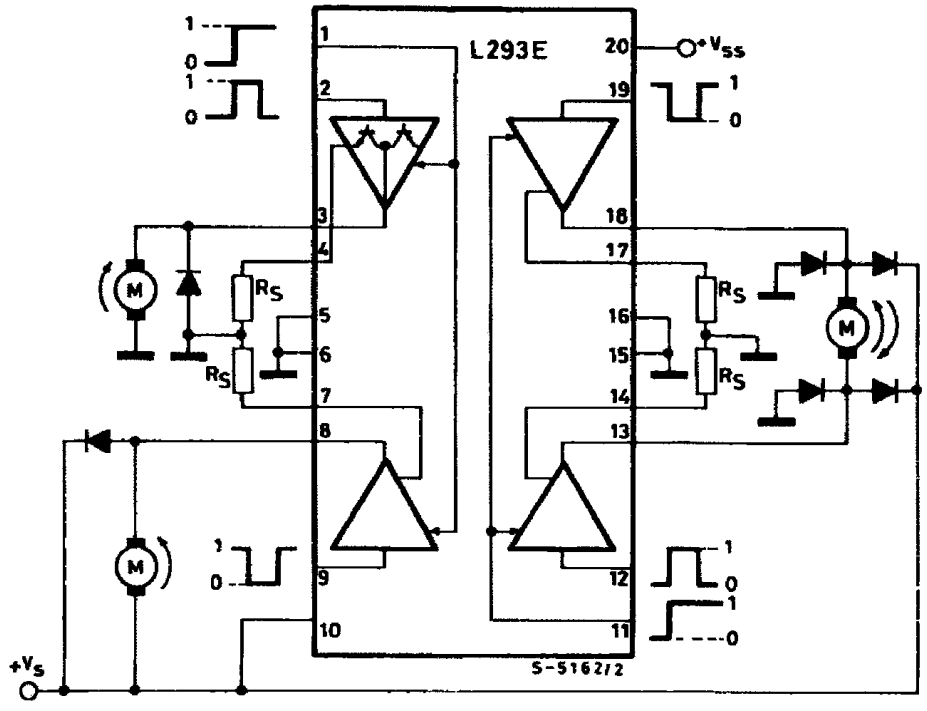


BLOCK DIAGRAMS

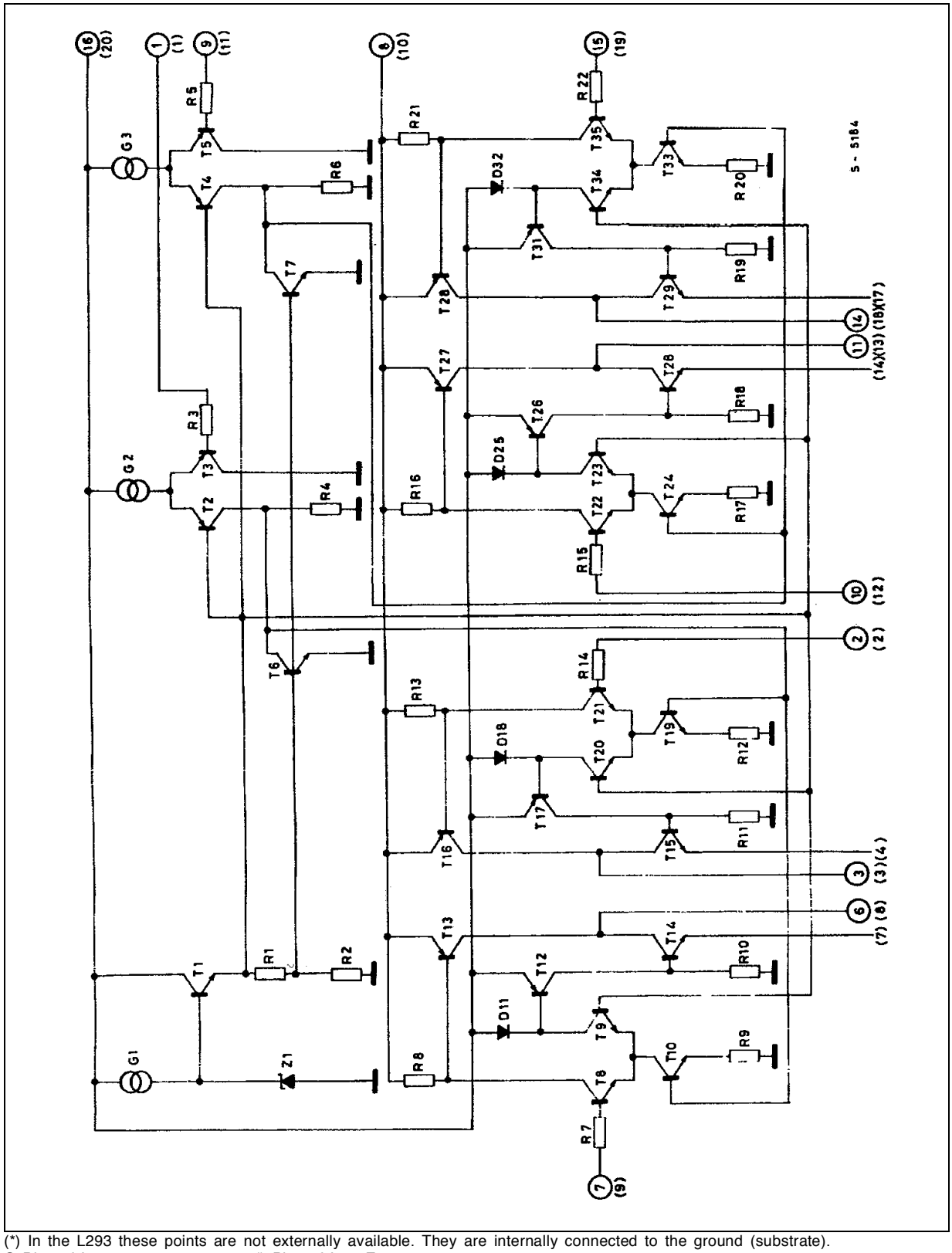
DIP16 - L293B



POWERDIP (16+2+2) - L293E



SCHEMATIC DIAGRAM



(*) In the L293 these points are not externally available. They are internally connected to the ground (substrate).
O Pins of L293 () Pins of L293E.

ABSOLUTE MAXIMUM RATINGS

Symbol	Parameter	Value	Unit
V_s	Supply Voltage	36	V
V_{ss}	Logic Supply Voltage	36	V
V_i	Input Voltage	7	V
V_{inh}	Inhibit Voltage	7	V
I_{out}	Peak Output Current (non repetitive t = 5ms)	2	A
P_{tot}	Total Power Dissipation at $T_{ground\text{-}pins}$ = 80°C	5	W
T_{stg}, T_j	Storage and Junction Temperature	–40 to +150	°C

THERMAL DATA

Symbol	Parameter		Value	Unit
$R_{th\ j\text{-}case}$	Thermal Resistance Junction-case	Max.	14	°C/W
$R_{th\ j\text{-}amb}$	Thermal Resistance Junction-ambient	Max.	80	°C/W

ELECTRICAL CHARACTERISTICS

For each channel, V_S = 24V, V_{SS} = 5V, T_{amb} = 25°C, unless otherwise specified

Symbol	Parameter	Test Conditions	Min.	TYp.	Max.	Unit
V_s	Supply Voltage		V_{ss}		36	V
V_{ss}	Logic Supply Voltage		4.5		36	V
I_s	Total Quiescent Supply Current	V_i = L I_o = 0 V_{inh} = H V_i = H I_o = 0 V_{inh} = H V_{inh} = L		2 16	6 24 4	mA
I_{ss}	Total Quiescent Logic Supply Current	V_i = L I_o = 0 V_{inh} = H V_i = H I_o = 0 V_{inh} = H V_{inh} = L		44 16 16	60 22 24	mA
V_{iL}	Input Low Voltage		-03.		1.5	V
V_{iH}	Input High Voltage	$V_{SS} \leq 7V$ $V_{ss} > 7V$	2.3 2.3		V_{ss} 7	V
I_{iL}	Low Voltage Input Current	V_{il} = 1.5V			-10	µA
I_{iH}	High Voltage Input Current	$2.3V \leq V_{IH} \leq V_{ss} - 0.6V$		30	100	µA
V_{inhL}	Inhibit Low Voltage		-0.3		1.5	V
V_{inhH}	Inhibit High Voltage	$V_{SS} \leq 7V$ $V_{ss} > 7V$	2.3 2.3		V_{ss} 7	V
I_{inhL}	Low Voltage Inhibit Current	V_{inhL} = 1.5V		-30	-100	µA
I_{inhH}	High Voltage Inhibit Current	$2.3V \leq V_{inhH} \leq V_{ss} - 0.6V$			±10	µA
V_{CEsatH}	Source Output Saturation Voltage	I_o = -1A		1.4	1.8	V
V_{CEsatL}	Sink Output Saturation Voltage	I_o = 1A		1.2	1.8	V
V_{SENS}	Sensing Voltage (pins 4, 7, 14, 17) (**)				2	V
t_r	Rise Time	0.1 to 0.9 V_o (*)		250		ns
t_f	Fall Time	0.9 to 0.1 V_o (*)		250		ns
t_{on}	Turn-on Delay	0.5 V_i to 0.5 V_o (*)		750		ns
t_{off}	Turn-off Delay	0.5 V_i to 0.5 V_o (*)		200		ns

* See figure 1
** Referred to L293E

TRUTH TABLE

V_i (each channel)	V_o	V_{inh} (∞)
H	H	H
L	L	H
H	X (°)	L
L	X (°)	L

(*) High output impedance
(**) Relative to the considerate channel

Figure 1 : Switching Timers

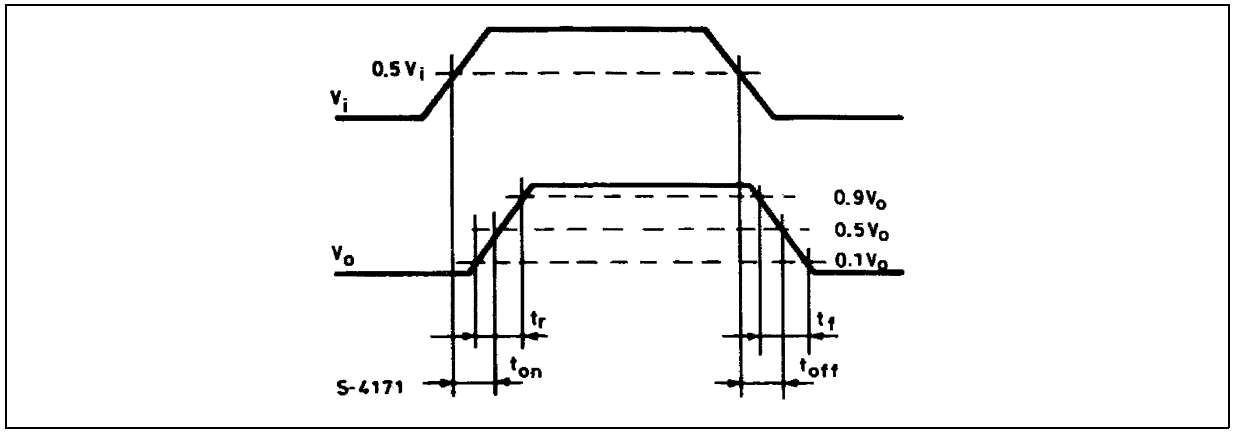


Figure 2 : Saturation voltage versus Output Current

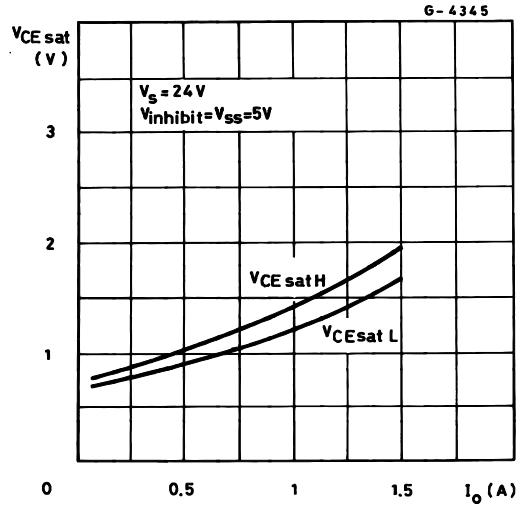


Figure 3 : Source Saturation Voltage versus Ambient Temperature

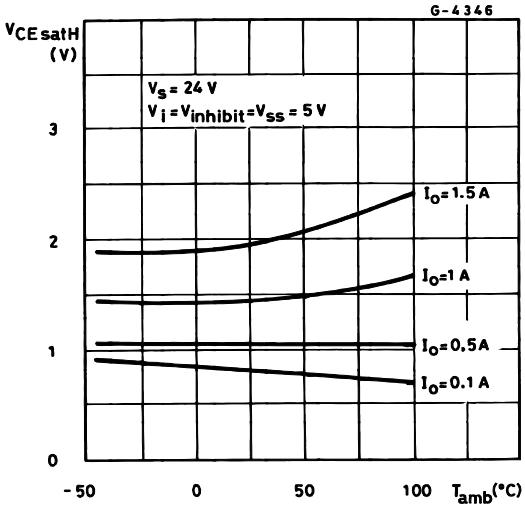


Figure 4 : Sink Saturation Voltage versus Ambient Temperature

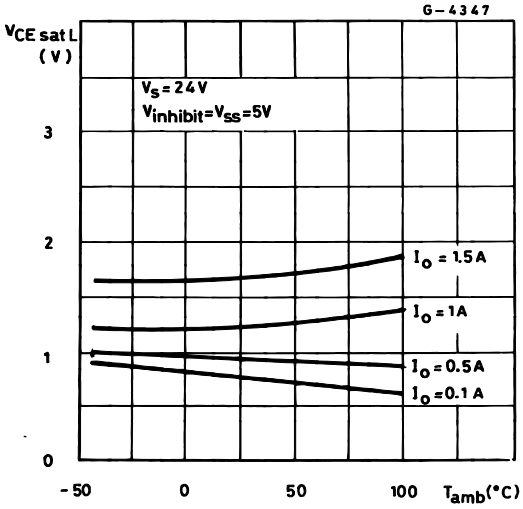


Figure 5 : Quiescent Logic Supply Current versus Logic Supply Voltage

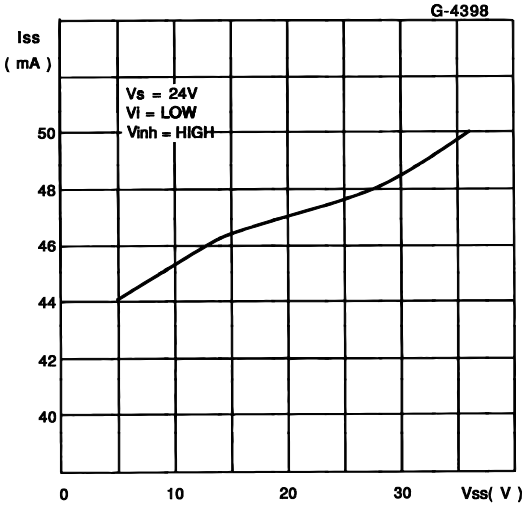


Figure 6 : Output Voltage versus Input Voltage

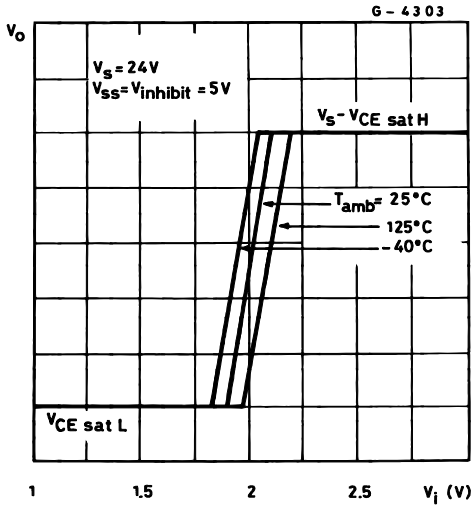
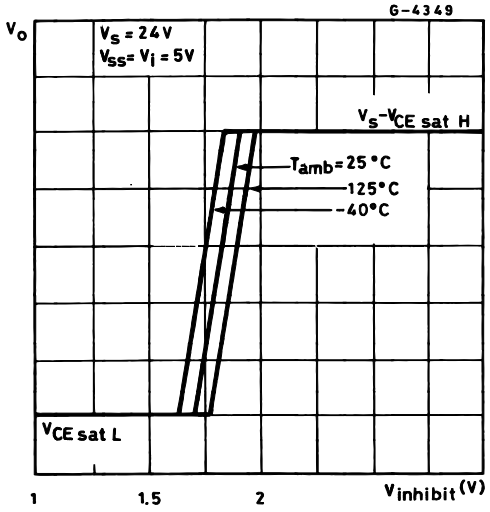
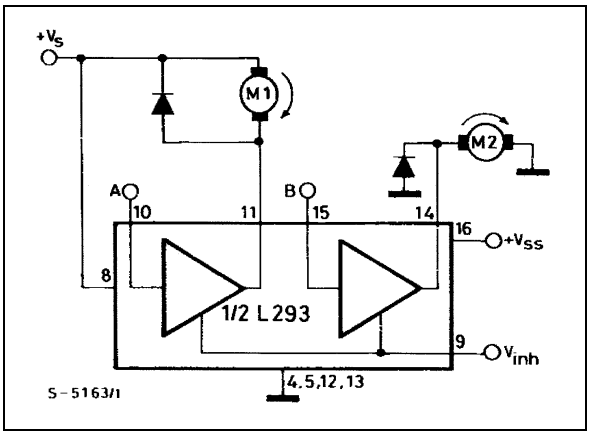


Figure 7 : Output Voltage versus Inhibit Voltage



APPLICATION INFORMATION

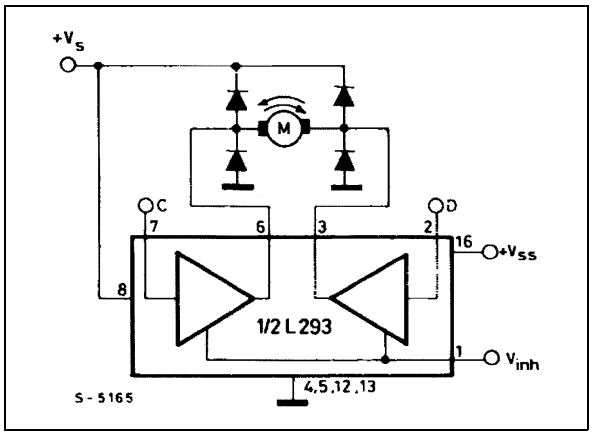
Figure 8 : DC Motor Controls (with connection to ground and to the supply voltage)



V_{inh}	A	M1	B	M2
H	H	Fast Motor Stop	H	Run
H	L	Run	L	Fast Motor Stop
L	X	Free Running Motor Stop	X	Free Running Motor Stop

L = Low H = High X = Don't Care

Figure 9 : Bidirectional DC Motor Control



Inputs	Function	
V_{inh} = H	C = H ; D = L	Turn Right
	C = L ; D = H	Turn Left
	C = D	Fast Motor Stop
V_{inh} = L	C = X ; D = X	Free Running Motor Stop

L = Low H = High X = Don't Care

Figure 10 : Bipolar Stepping Motor Control

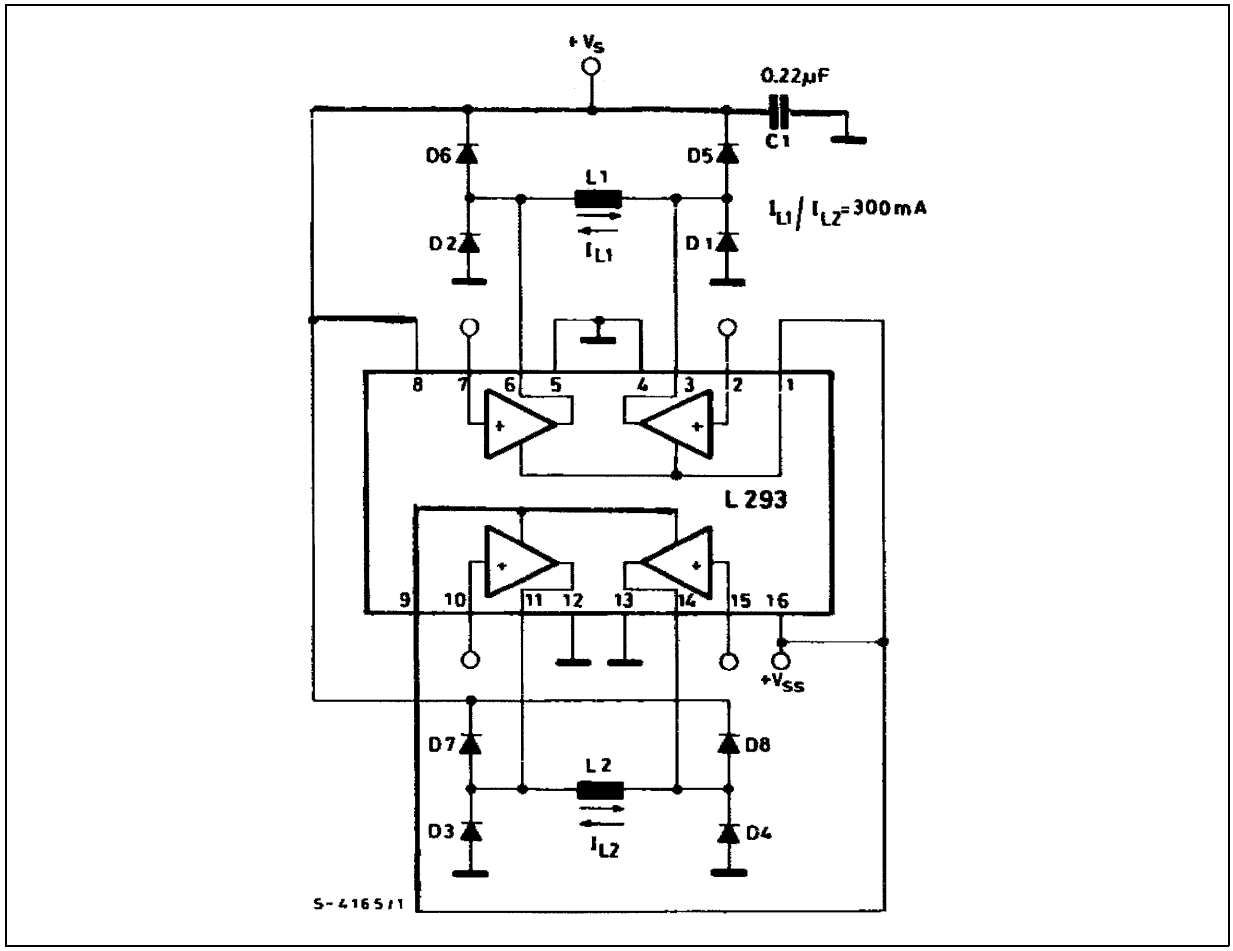


Figure 11 : Stepping Motor Driver with Phase Current Control and Short Circuit Protection