
*"Responsabilidad con pensamiento positivo"*

# UNIVERSIDAD TECNOLÓGICA ISRAEL

# TRABAJO DE TITULACIÓN

**CARRERA:** Electrónica Digital y Telecomunicaciones

**TEMA:** Diseño e implementación de un control de bloqueo para un automóvil a través de la bomba de gasolina utilizando tecnología Arduino.

**AUTOR:** Mauro Javier Peralta Coraquilla.

**TUTOR:** Mg. Wilmer Albarracín.

**2015**

UNIVERSIDAD TECNOLÓGICA ISRAEL

# APROBACIÓN DEL TUTOR

En mi calidad de Tutor del Trabajo de Titulación certifico:

Que el Trabajo de Titulación "**DISEÑO E IMPLEMENTACIÓN DE UN CONTROL DE BLOQUEO PARA UN AUTOMÓVIL A TRAVÉS DE LA BOMBA DE GASOLINA UTILIZANDO TECNOLOGÍA ARDUINO**", presentado por el señor Mauro Javier Peralta Coraquilla, CI 1718474750, estudiante de la carrera de Electrónica Digital y Telecomunicaciones, reúne los requisitos y méritos suficientes para ser sometido a la evaluación del Tribunal de Grado, que se designe, para su correspondiente estudio y certificación.

Quito D.M. febrero del 2015

TUTOR

Ing. Wilmer Albarracín.

UNIVERSIDAD TECNOLÓGICA ISRAEL

# AUTORÍA DEL TRABAJO DE TITULACIÓN

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

Quito D.M. febrero del 2015

Mauro Javier Peralta Coraquilla

CI. 1718474750

UNIVERSIDAD TECNOLÓGICA ISRAEL

# APROBACIÓN DEL TRIBUNAL DE GRADO

Los miembros del Tribunal de Grado, aprueban el trabajo de titulación de acuerdo con las disposiciones reglamentarias emitidas por la Universidad Tecnológica Israel para Títulos de Pregrado.

Quito D.M febrero del 2015

Para constancia firman:

_____

PRESIDENTE

_____ _____

MIEMBRO 1 MIEMBRO 2

# AGRADECIMIENTO

En esta parte quiero agradecer de manera muy especial a todas las personas que siempre creyeron en mí y me brindaron todo su apoyo, A mis maestros que me impartieron con sabios conocimientos durante toda esta etapa.

# DEDICATORIA

Este trabajo quiero dedicar con mucho amor a mis padres, ya que ellos son el pilar que sostiene toda mi vida. A mi padre Wilson Peralta que es un ejemplo a seguir para mi futuro y a mi madre Olga Coraquilla que es la persona más extraordinaria de todo este mundo, ellos son quienes más anhelaron que llegue esta etapa de mi vida para sentirse muy orgullosos de su hijo Mauro Peralta, todo esto gracias a DIOS que es el todo poderoso.

**ÍNDICE DE CONTENIDOS**

## ÍNDICE DE FIGURAS

## ÍNDICE DE TABLAS

## INTRODUCCIÓN

Con el avance de la tecnología en el siglo XXI, la frontera de la disciplina y fortaleza de la ingeniería Electrónica se ha convertido en una de las más importantes en el mundo, tal es así que algunos productos en el mercado están fabricados con componentes que tienen una interdependencia de la Electrónica, la Mecánica, Ingeniería Eléctrica y los sistemas computarizados de control inteligentes, esto ha llevado a que los fabricantes de estos productos integren en sus actividades a verdaderos Ingenieros en el campo de la Electrónica, Electricidad, Mecánicos y de sistemas computarizados.

La "implementación de un control de bloqueo para un automóvil a través de la bomba de gasolina utilizando tecnología Arduino", es un proyecto que está dirigido a la comunidad en general que por las condiciones cotidianas de sus labores, sus actividades o planes diarios personales, tienen que dejar sus automóviles en lugares que no cuentan con vigilancia, por lo que es necesario para ellos tener una seguridad tecnológica instalada desde el interior de su auto controlada con un dispositivo móvil, y así realizar cualquier actividad durante el día sin preocupación de que su automóvil sea robado.

En la actualidad la delincuencia en robos de automóviles ha crecido significativamente a nivel nacional y se cuenta con una limitada seguridad aplicada a un control de bloqueo a la bomba de gasolina mediante un dispositivo móvil que tenga sistema Android programado con tecnología Arduino.

A pesar de los grandes avances y beneficios de la seguridad para automóviles, existen limitaciones en el estudio para realizar un prototipo electrónico que se permita realizar un control de bloqueo utilizando tecnología Arduino, ya que carece de un diseño específico para controlar la bomba de gasolina.

Actualmente, la comunidad en general requieren de una tranquilidad al momento de dejar sus automóviles para realizar sus actividades diarias; sin embargo, el país no cuenta con los suficientes parqueaderos ni lugares de estacionamiento con vigilancia durante el día, sin tomar en cuenta que en algunos casos dejamos los autos durante la noche sin tener ninguna clase de seguridad y exponiendo a robo.

Al tratarse de un proyecto de ejecución, se han planteado los propósitos y alcances de la investigación, los mismos que permiten realizar un análisis de la problemática planteada que es la limitación de un control de bloqueo para un automóvil a través de la bomba de gasolina utilizando tecnología Arduino, para lo cual es preciso dar soluciones.

Se dividió en fases para sistematizar el proceso del proyecto y poder cumplir con nuestra meta:

La primera fase permitió conocer a fondo la tecnología Arduino, la cual se va a utilizar. Fue muy importante ya que se conoce la programación implementada en un sistema Android con todos los tipos de seguridad que requería nuestro proyecto.

La segunda fase, en cambio, sirvió para fusionar la parte electrónica con la parte mecánica, ya que después de tener listo el prototipo electrónico comenzamos la instalación en la bomba de gasolina del automóvil para completar nuestro proyecto.

En la siguiente fase se pudo verificar el funcionamiento del control de bloqueo, y los beneficios para la población, validando la metodología empleada en el desarrollo del Trabajo de Titulación.

La ejecución de este proyecto de investigación tecnológico, permite dar soluciones a la problemática que ocurre en nuestra sociedad, facilitando a la población una seguridad tecnológica implementada para disminuir el robo de autos.

# OBJETIVOS

## OBJETIVO PRINCIPAL:

Implementar un control de bloqueo para un automóvil a través de la bomba de gasolina utilizando tecnología Arduino.

## OBJETIVOS ESPECÍFICOS:

- Estudiar todos los componentes electrónicos para la creación del control de bloqueo a través de la bomba de gasolina.
- Diseñar un circuito electrónico específico para poder realizar el proyecto.
- Implementar un control de bloqueo a través de la bomba de gasolina para un automóvil utilizando tecnología Arduino, teniendo en cuenta los requerimientos del proyecto.
- Validar el funcionamiento del control de bloqueo implementado.

# CAPÍTULO I

# FUNDAMENTACIÓN TEÓRICA

## 1.1 INTRODUCCIÓN

En este capítulo se relataran fundamentos relacionados con las partes importantes que conformaron el prototipo y su funcionamiento, que se han organizados de acuerdo a investigaciones de varios expertos y de la persona que desarrollo este proyecto.

El control de bloqueo de un automóvil se ha implementado de acuerdo a un intensivo análisis acorde a los beneficios y a los avances tecnológicos de nuestra actualidad, para lo cual se pone en consideración los siguientes conceptos que permitirán al lector relacionarse con el contexto del proyecto.

## 1.2 MARCO TEÓRICO

### 1.2.1 Arduino

La tecnología Arduino es una plataforma de hardware libre que está basada en una placa que cuenta con un microcontrolador y un entorno de desarrollo, está diseñada para facilitar el uso de la electrónica en proyectos multidisciplinares. Arduino UNO es posiblemente la placa más conocida y documentada. (Arduino UNO)

“Arduino es una plataforma de electrónica abierta a la creación de, arduino puede tomar información del entorno a través de sus pines de entrada.” (Arduino, Arduino, 2014)

El hardware de Arduino consiste en una placa con un microcontrolador Atmel AVR y puertos de entrada/salida.4 entre los microcontroladores más utilizados tenemos el Atmega168, Atmega328, Atmega1280, Atmega8 por su bajo precio que permiten el desarrollo de varios diseños. En cambio el software es un entorno de desarrollo que crea el lenguaje de programación Processing/Wiring y el cargador de arranque que es ejecutado en la placa. (wikipedia, s.f.)

También utilizamos para desarrollar objetos interactivos autónomos o puede ser conectado a software tal como Adobe Flash, Processing, Max/MSP, Pure Data. Podemos realizar las placas a mano o adquirirlas, podemos descargar gratuitamente el entorno de desarrollo integrado libre Las placas se pueden montar a mano o adquirirse. El entorno de desarrollo integrado libre. (wikipedia, s.f.)

Podemos adquirir información del entorno mediante sus entradas analógicas y digitales, puede controlar luces, motores y otros actuadores. Mediante el lenguaje de programación en Arduino (basado en Wiring) y el entorno de desarrollo Arduino (basado en Processing)

podemos programar el microcontrolador que se encuentra en la placa. Los proyectos hechos con Arduino pueden ejecutarse sin necesidad de conectar a un computador. (wikipedia, s.f.)

### 1.2.2 Beneficios y Desventajas

Con el pasar del tiempo y las exigencias tecnológicas de nuestra actualidad, los proyectos programados en placas electrónicas se han ido modificando las funcionalidades, características y lenguajes de programación para desempeñar mejor su control electrónico, sin embargo existen ciertas limitantes que podrían ser tomadas como desventajas al momento de decidir trabajar con esta tecnología o sino optar por otra. (Investigador, 2015)

En el siguiente cuadro de beneficios y desventajas mostraremos por qué la utilización de Arduino.

| BENEFICIOS | DESVENTAJAS |
|---|---|
| El software de Arduino funciona en varios sistemas operativos como: Windows, Linux, Macintosh OSX, por otra parte la mayoría de los entornos para Microcontroladores están limitadas solamente para Windows. | Su Programación es un poco compleja para si no se tiene conocimientos |
| Para Arduino las placas son más asequibles en comparación de otras plataformas de Microcontroladores. | Falta de salidas de potencia, por lo que sin un potenciómetro es incompatible con motores. |
| El software de Arduino está publicado con licencia libre, y preparado para poder ser ampliado por experimentados programadores. | |
| | |

Tabla 1.1: Beneficios y desventajas de la tecnología Arduino

(Fuente: Investigador)

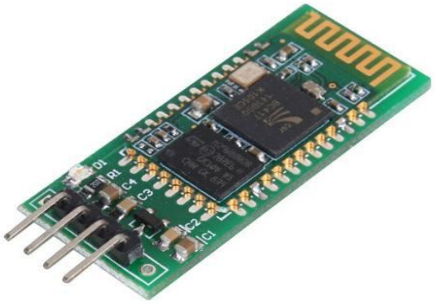
### 1.2.3 Comunicación serial

Se utiliza para comunicarse entre la placa arduino y un dispositivo “Las placas Arduino tienen un puerto serial (también conocido como UART o USART). Este se comunica a través de los dos pines de conexión RX (pin 0) y TX (pin 1).” (Arduino, Arduino, 2014)

### 1.2.4 El Modulo Bluetooth HC-06

Es el elemento que va a estar instalado en el prototipo para dar conectividad con el sistema Android. Sus dimensiones son tan pequeñas como un conector USB y su precio es muy económico. En este pequeño dispositivo encontramos todo lo necesario para conexiones Wireless, el cual va a estar impreso en el prototipo que instalaremos en la bomba de gasolina del auto y así realizara la comunicación con el Sistema. (Investigador, 2015)

Figura1.1: Modulo Bluetooth



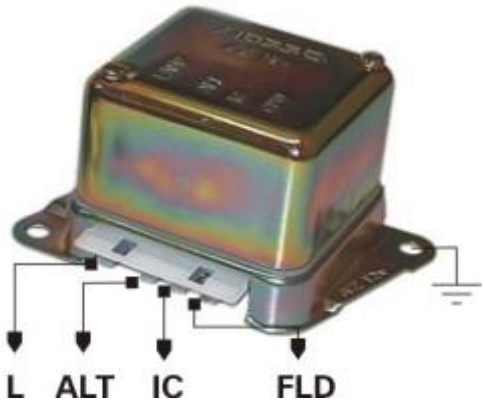
(Fuente: Plataforma Garagem, pág. 236)

### 1.2.5 Fuente de Poder

La fuente de poder es un elemento importantísimo para la realización del proyecto. En el automóvil tenemos 12v la cual vamos a estabilizar a 9v q es de Arduino, una fuente de alimentación es un dispositivo el cual convierte la tensión alterna en una o varias tensiones continuas, que alimentan los distintos circuitos del aparato electrónico al que se conecta. (Investigador, 2015)

En vehículos actuales se usa generalmente 12 V. Constituida por placas (+) de Peróxido de Plomo y placas (-) de Plomo esponjoso. Separadas entre ellas por aisladores y sumergidas en ácido sulfúrico. La reacción química del ácido sobre el plomo origina una serie de elementos resultantes, de todos ellos recogemos la electricidad del tipo corriente continua para el funcionamiento de distintos sistemas y aparatos. (Rojas, 2008)

Figura 1.2: Fuente de poder




(Fuente: Scanning & Mobility)

### 1.2.6 Bomba de Gasolina

La bomba de gasolina del automóvil es un elemento automotriz esencial para el buen rendimiento del motor, se encargada de hacer que el sistema de inyección reciba de manera constante el combustible a través de los rieles de los inyectores que mediante succión extraen el líquido del tanque el cual permite la circulación del combustible hacia los inyectores y produce el movimiento de los pistones. (Rojas, 2008)

Por lo particular, se trata de bombas eléctricas que tienden a estar instaladas en el interior del depósito de combustible. El amperaje con el cual funciona la bomba de gasolina de un automóvil es de 12 V y se acciona a través del relé de la bomba. (Rojas, 2008)

Figura1.3: Bomba de gasolina

(Fuente: Garagem 2008, pág. 87)

## 1.3 APLICACIONES

En función de la programación con tecnología Arduino realizaremos la conexión con Android.

### 1.3.1 Sistema Android

Android es una plataforma de software creada para dispositivos móviles con tecnología avanzada, tiene un Sistema Operativo y aplicaciones. (wikipedia, s.f.)

Es una recopilación de herramientas, aplicaciones vinculadas a una repartición Linux para dispositivos móviles. El sistema Android por sí solo es un sistema de código abierto, gratuito y no requiere de pagos de licencias. (wikipedia, s.f.)

Para dispositivos móviles es una plataforma de código abierto y desarrollada por Open Handset Alliance, Los primeros teléfonos con Sistema Android salió en el año 2008 creado por las compañías LG, Motorola y HTC. (wikipedia, s.f.)

### 1.3.2 CARACTERÍSTICAS DE ANDROID CON OTROS SISTEMAS OPERATIVOS

En el siguiente cuadro mostraremos algunas características del Sistema Android con otros Sistemas Operativos. (Investigador, 2015)

| N° | CARACTERISTICAS |
|---|---|
| Aplicaciones | Incluyen un cliente de correo electrónico, programa de SMS, calendario, mapas, navegador, contactos y otros. Todas ellas escritas en Java. |
| Marco de trabajo de aplicaciones | Los desarrolladores tienen acceso completo a los mismos APIs del framework usados por las aplicaciones base. La arquitectura está diseñada para simplificar la reutilización de componentes; cualquier aplicación puede publicar sus capacidades y cualquier otra aplicación puede luego hacer uso de esas capacidades. |
| | Incluye un conjunto de bibliotecas de C/C++ usadas por varios componentes del sistema. |

Tabla 1.2: Características Android

(Fuente: Investigador)

### 1.3.3 ANDROID EN CELULARES

Es una plataforma creada para desempeñar su funcionalidad para celulares, parte principal del dispositivo, se encuentra físicamente en el procesador, es de ahí donde enviamos las órdenes que permiten al Smartphone aplicar el sin número de tareas, programas y aplicaciones que el celular lo requiera. (wikipedia, s.f.)

### 1.3.4 BENEFICIOS Y DESVENTAJAS DEL SISTEMA ANDROID EN CELULAR

En el siguiente cuadro analizaremos las principales funcionalidades que nos ofrece el Sistema Android. (Investigador, 2015)

| BENEFICIOS | DESVENTAJAS |
|---|---|
| Es un sistema en Linux, abierto y libre, su licencia no tiene costo, por lo tanto, es totalmente gratuito. | Android por ser un sistema que salió recientemente, puede ocasionar algún mínimo problema |
| Su principal uso fue hecho para Smartphone de gama media-alta y con pantalla táctil, nos brinda rapidez de reacción y una excelente exactitud en el manejo del teclado virtual | Su velocidad puede ser por momentos menor si la memoria del móvil llena de programas y archivos |
| Para Android Marquet existe infinidad de aplicaciones y siempre siguen creando nuevos contenidos para el celular, lo excelente de utilizar Android es que no tiene ninguna clase de restricciones | Al momento de querer descargar aplicaciones nos exige que cerremos otras. |
| Accede a juegos normalmente y visualiza videos ya que resuelve cualquier web en flash. | |

Tabla 1.3 Características de Android

(Fuente: Investigador)

# CAPÍTULO II

## DIAGNÓSTICO DEL PROBLEMA ESTUDIADO Y BREVE DESCRIPCIÓN DEL PROCESO INVESTIGATIVO REALIZADO PARA LA IMPLEMENTACIÓN DE UN CONTROL DE BLOQUEO CON TECNOLOGIA ARDUINO.

En este capítulo se expone la importancia del proyecto en la solución de una de las problemáticas de la sociedad con respecto al uso de seguridades electrónicas para evitar el robo de automóviles. Se describe el análisis realizado sobre los inconvenientes que posee la población por la falta de un control de bloqueo para un automóvil a través de la bomba de gasolina utilizando tecnología Arduino.

Para el desarrollo de sistema se propuso un objetivo general plateado en la Introducción del presente informe con la finalidad de elaborar un sistema que permita bloquear la bomba de gasolina de un automóvil a través de un control (Smartphone) con tecnología Arduino.

Los objetivos específicos fueron propuestos con la finalidad de investigar los componentes necesarios para su implementación y ejecutar el proyecto.

La idea a defender o hipótesis del presente proyecto se plantea de la siguiente forma: si se implementa un control de bloqueo para un automóvil a través de la bomba de gasolina utilizando tecnología Arduino, disminuirá los robos de automóviles. Teniendo como variable dependiente: el control de bloqueo para un automóvil a través de la bomba de gasolina utilizando tecnología Arduino y como variable dependiente: disminución de robos de automóviles.

Para el presente proyecto se van a utilizar ciertas tecnologías para la aplicación del producto en nuestra sociedad, estas tecnologías permitirán automatizar un control de bloqueo a través de la bomba de gasolina, utilizando la tecnología arduino como la parte principal, al momento de programar insertaremos un código, el cual tendremos que ingresar a nuestro Smartphone a través del Sistema Android utilizando la aplicación Blue term.

El presente trabajo de titulación está dividido en 4 etapas (Estudio, Diseño, Implementación y Validación) en las cuales se utilizarán los siguientes métodos de investigación:

En la Primera Etapa se utilizará los métodos de Análisis y Síntesis para la adjuntar todos los datos para poder desarrollar el proyecto.

En la Segunda Etapa se tomara en cuenta el diseño del proyecto utilizando el método de modelación.

En la Tercera y Cuarta Etapa se usará el método experimental que va a permitir verificar el comportamiento del proyecto para así llegar a ponerlo en marcha.

Una vez que se han analizado las soluciones de diseñar un prototipo electrónico especifico con tecnología Arduino, es necesario detectar la aceptación de las personas tienen auto, a través de una encuesta aplicada el 09 de Enero del 2015 en las calles de la cuidad a 30 habitantes entre hombres y mujeres mayores de 18 años. Se plantearon las siguientes preguntas:

UNIVERSIDAD TECNOLÓGICA ISRAEL

CARRERA DE ELECTRÓNICA Y TELECOMUNICACIONES

ENTREVISTA A LA COMUNIDAD DE LA CIUDAD DE QUITO

Nombre: …….

Sexo: …….

Fecha: …….

1. ¿Utiliza usted un dispositivo móvil con Sistema Android (celular, Tablet)?

SI ☐

NO ☐

2.- ¿En qué tipo de dispositivo le gustaría tener implementado un control de bloqueo tecnológico para evitar el robo de su auto?

Celular ☐

Tablet ☐

Control de Alarma ☐

3.- ¿Al momento de estacionar su auto a las afueras de su trabajo, casa, etc., Ud. cuenta con vigilancia mientras realiza sus actividades?

SI ☐

NO ☐

4.- ¿A qué parte del auto piensa Ud. que se debería bloquear utilizando seguridad electrónica?

Bomba de gasolina ☐

Switch de encendido ☐

Alarma del auto ☐

A continuación se presentan los resultados de la encuesta.

Pregunta 1

Utiliza usted un dispositivo móvil con Sistema Android (celular, Tablet).

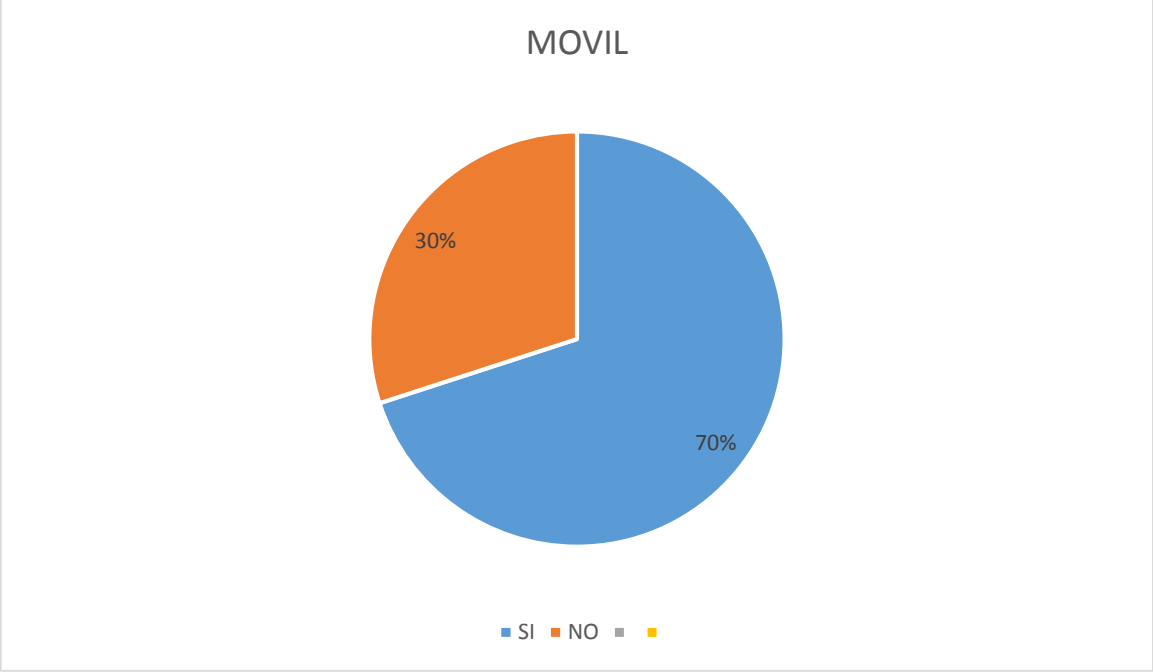
Respuestas obtenidas:

| N° | Respuesta | N° de Habitante | % |
|---|---|---|---|
| 1 | SI | 21 | 70% |
| 2 | NO | 9 | 30% |

Tabla 2.1: Tabulación de pregunta 1

(Fuente: Investigador, Enero 2015)

Figura 2.1: Tabulación de pregunta 1




(Fuente: Investigador, Enero 2015)

Análisis:

Las respuestas de los habitantes encuestados demuestran que la mayoría de ellos tienen un móvil con Sistema Android, ya que su precio no es muy elevado y accesible para adquirirlo. Un 70% si cuentan con este tipo de celulares y el otro 30% restante no.

Pregunta 2

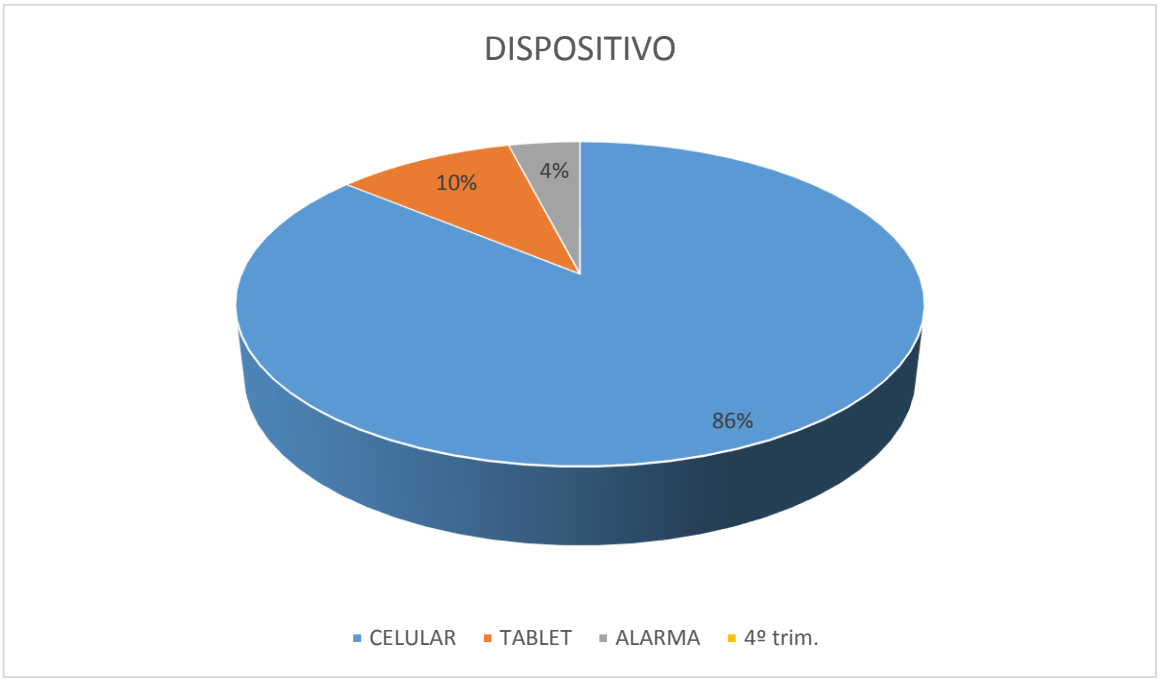
En qué tipo de dispositivo le gustaría tener implementado un control de bloqueo tecnológico para evitar el robo de su auto.

| N° | DISPOSITIVOS | N° DE HABITANTES | % |
|---|---|---|---|
| **1** | Celular | 26 | 86.6% |
| **2** | Tablet | 3 | 10% |
| **3** | Control de Alarma | 1 | 3.33% |

Tabla 2.2: Tabulación de pregunta 2

(Fuente: Investigador, Enero 2015)

Figura 2.2: Tabulación de pregunta 2




(Fuente: Investigador, Enero 2015)

Análisis:

Este análisis demuestra que los habitantes se inclinan por utilizar su celular para controlar el bloqueo de su auto.

Pregunta 3

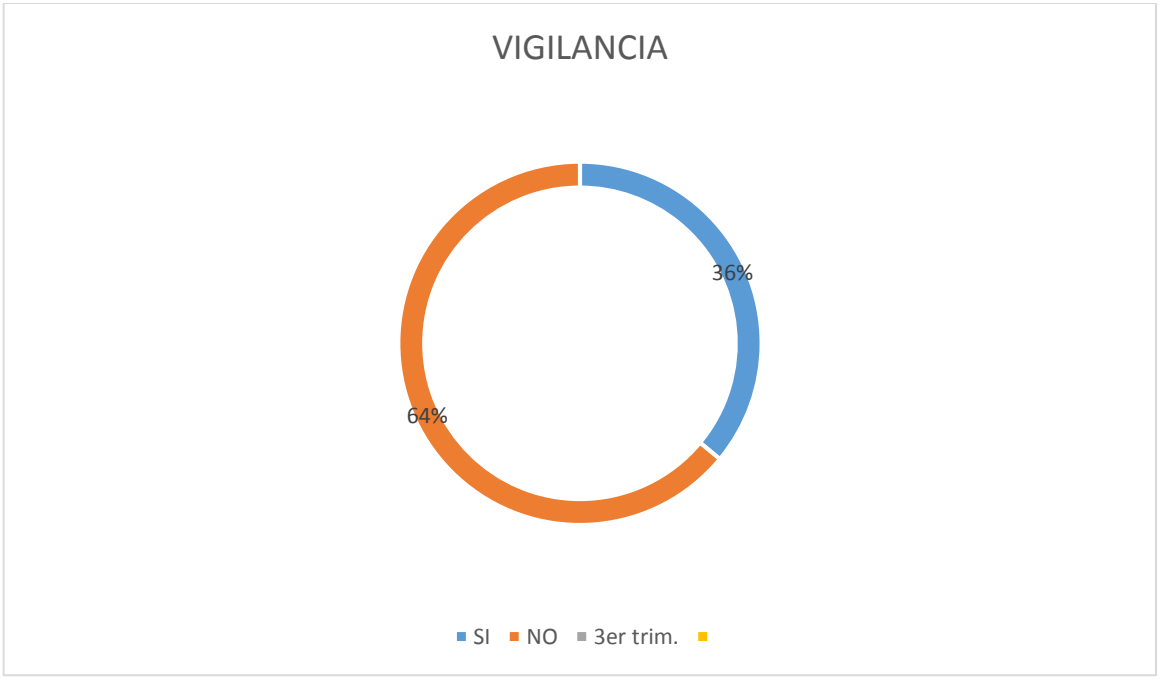
Al momento de estacionar su auto a las afueras de su trabajo, casa, etc., Ud. cuenta con vigilancia mientras realiza sus actividades.

| N° | RESPUESTA | N° DE HABITANTES | % |
|---|---|---|---|
| **1** | SI | 11 | 36% |
| **2** | NO | 19 | 64% |

Tabla 2.3: Tabulación de pregunta 3

(Fuente: Investigador, Enero 2015)

Figura 2.3: Tabulación de pregunta 3




(Fuente: Investigador, Enero 2015)

Análisis:

Las personas encuestadas por lo general dejan su auto a las afueras de sus localidades y la mayoría que es un 63.3% no cuenta con una seguridad que se encargue de la vigilancia permanente.

Pregunta 4

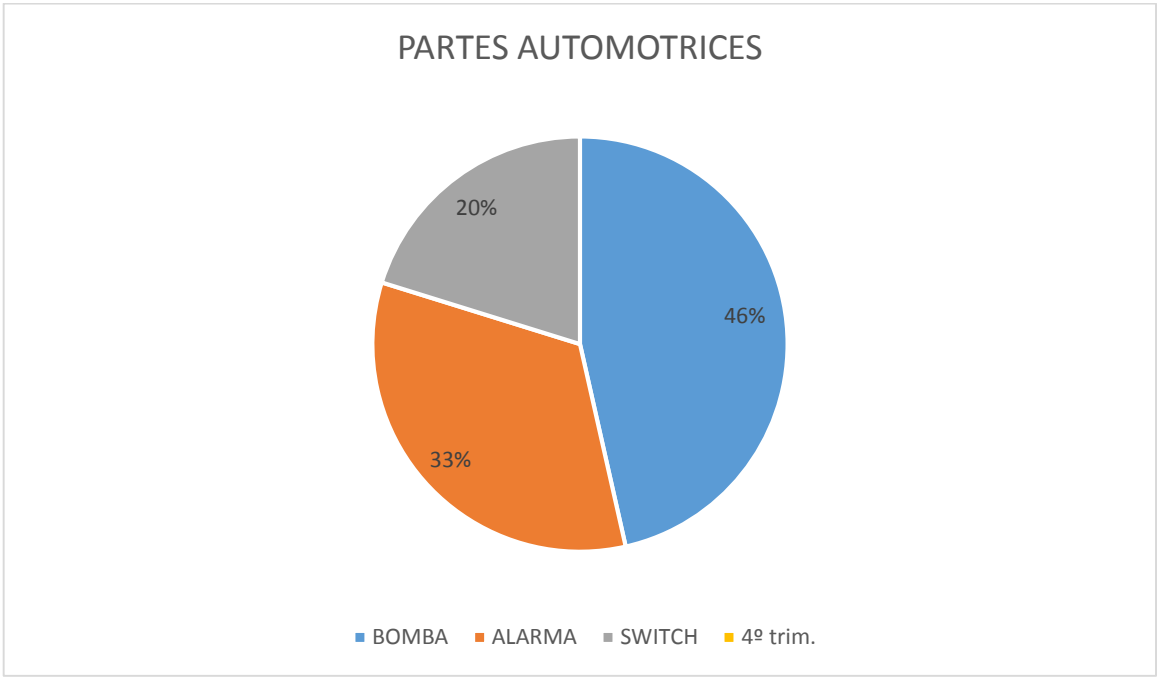
A que parte del auto piensa Ud. que se debería bloquear utilizando seguridad electrónica.

| N° | PARTE AUTOMOTRIZ | N ° DE HABITANTES | % |
|---|---|---|---|
| **1** | Bomba de gasolina | 14 | 46.6% |
| **2** | Switch de encendido | 10 | 33.3% |
| **3** | Alarma del auto | 6 | 20% |

Tabla 2.4: Tabulación de pregunta 4

(Fuente: Investigador, Enero 2015)

Figura 2.4: Tabulación de pregunta 4




(Fuente: Investigador, Enero 2015)

Análisis:

El 46% de personas prefieren optar por implementar la seguridad electrónica en la bomba de gasolina, ya que al momento de bloquear esta parte del auto, queda totalmente inmovilizado, ya que por ningún motivo arrancara el carro porque no absorbe gasolina el motor.

# CAPÍTULO III

# PRESENTACIÓN DE LOS RESULTADOS

## 3.1 PROPUESTA DE SOLUCIÓN DEL PROBLEMA

En este capítulo se relata la ejecución del proyecto, se realizan los objetivos propuestos de forma correcta, se aplica diversos métodos y técnicas de investigación, se realiza programaciones e instalaciones para la funcionalidad del control de bloqueo.

Se analiza la tecnología que va a ser utilizada teniendo en cuenta su programación, precio, funcionamiento y sus respectivas características sobre ella, debe cumplir con todos los requerimientos tecnológicos que el proyecto lo requiera.

Se continua con el estudio de todos los elementos que van a ser utilizados en nuestro prototipo electrónico.

Se realiza el diseño del circuito electrónico con todos los requerimientos propuestos, para dar funcionamiento a nuestro proyecto tecnológico.

Una vez elaborado el diseño, se implementa el prototipo electrónico de acuerdo a los parámetros establecidos y a los recursos tecnológicos disponibles. Se realiza varias pruebas de funcionamiento al momento de programar y realizar el circuito impreso, descubriendo bondades y limitantes de la tecnología utilizada, así como los procedimientos al momento de la instalación en el automóvil, se verifica su correcto funcionamiento al momento de involucrarnos en la parte mecánica la cual sería nuestra última etapa del proyecto.

Se realiza un análisis presupuestario comparativo, para obtener las diferencias en costos en cuanto a componentes electrónicos y mecánicos, con los requerimientos de la implementación del proyecto y así justificar su inversión.

## 3.2 Diseño

En el Diseño Electrónico se mostrara a continuación todos los diagramas y etapas que se utilizó para realizar el proyecto:

- Etapas de Funcionamiento
- Diagrama de Bloques
- Diagrama Esquemático
- Diagrama pcb
- Diagrama screen

### 3.2.1 Etapas de Funcionamiento

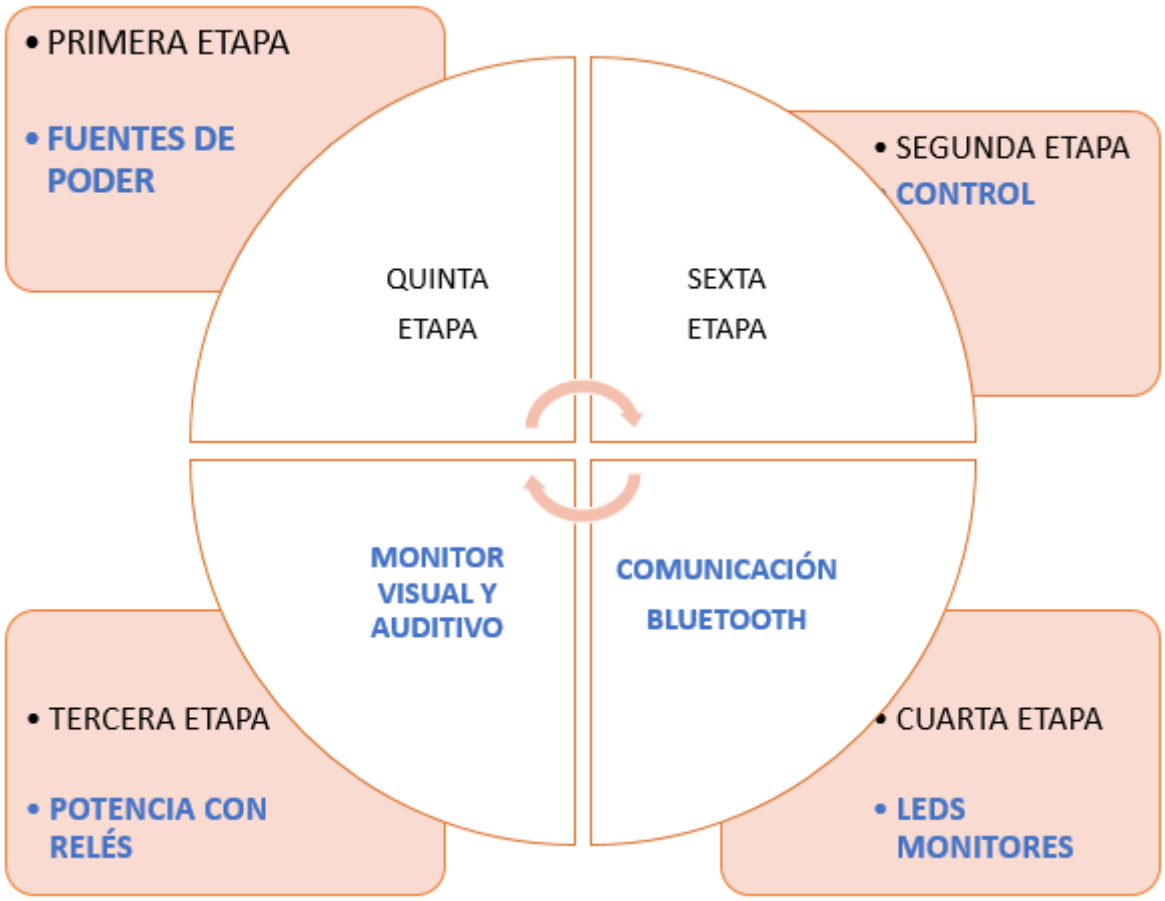


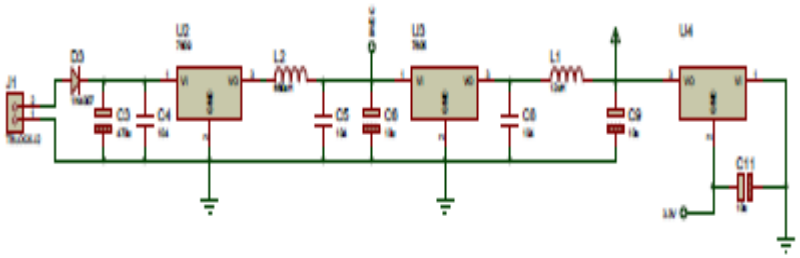

Tabla 3.1: Etapas de Funcionamiento

(Fuente: Investigador, Enero 2015)

**Primera Etapa**

Contiene todo lo relacionado a fuentes de poder, Se estabiliza los voltajes, de 12 v estabilizamos a 9v, 5v y 3v.

Figura 3.1: Fuentes de Poder

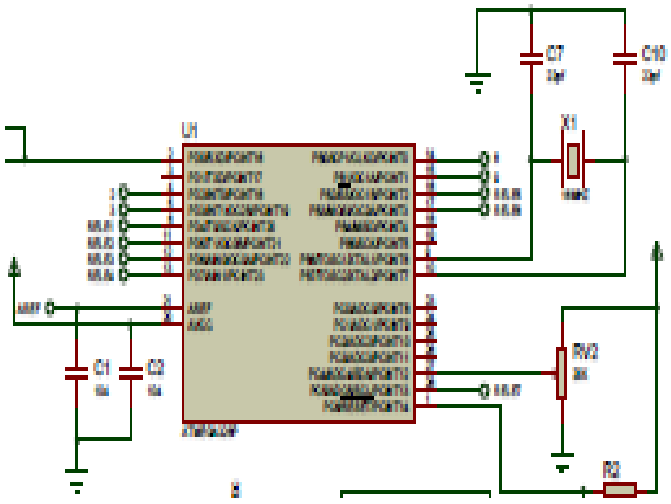


(Fuente: Investigador, Enero 2015)

**Segunda Etapa**

Contiene el Control Electrónico del circuito, el reloj que es la frecuencia de trabajo que tiene el PIC.

Figura 3.2: Control del Sistema

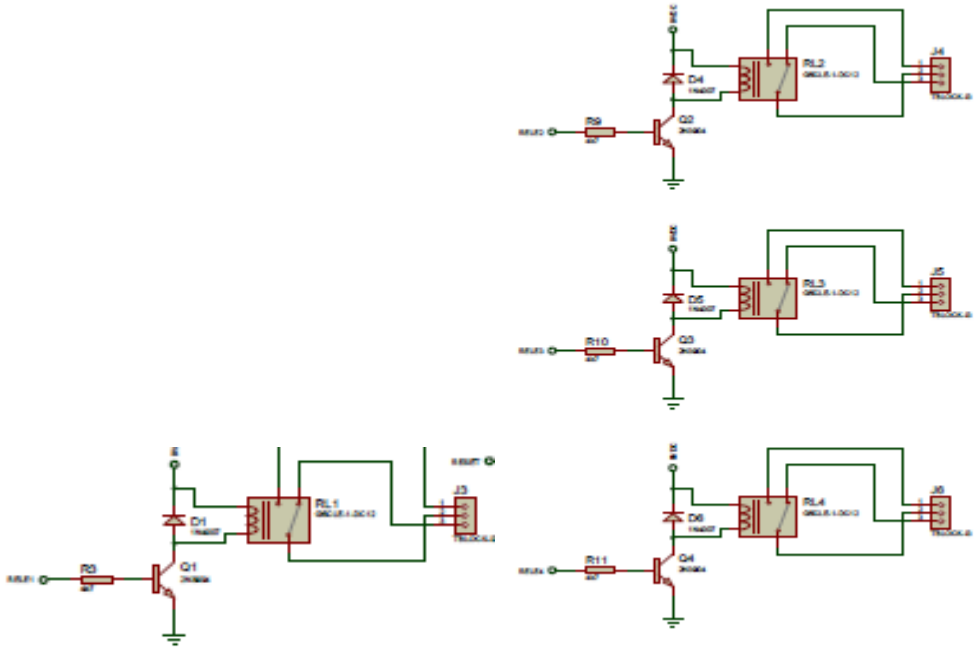


(Fuente: Investigador, Enero 2015)

### Tercera Etapa

Contiene la potencia con relé, es decir las salidas que tenemos para conectar con la parte mecánica que es el auto.

Figura 3.3: Relés de Salida

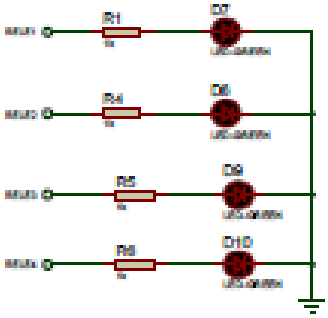


(Fuente: Investigador, Enero 2015)

### Cuarta Etapa

Contiene los leds monitores que funcionan al momento que se activan los relés, es decir los leds se encienden cada vez q el relé este activado.

Figura 3.4: Leds Monitores

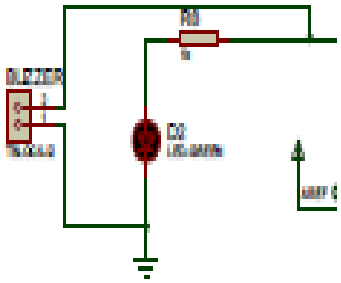


(Fuente: Investigador, Enero 2015)

**Quinta Etapa**

Contiene el indicador monitor visual y auditivo que emite un sonido al momento de conectar el bluetooth del móvil con el prototipo.

Figura 3.5: Monitor Visual-Auditivo

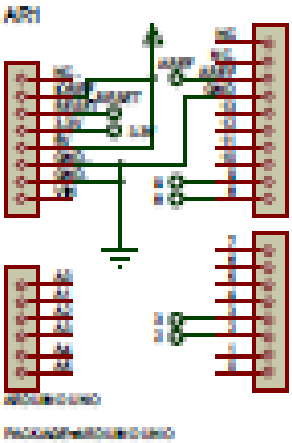


(Fuente: Investigador, Enero 2015)

**Sexta Etapa**

Contiene la comunicación donde existe un sócalo en el cual conectamos el bluetooth.

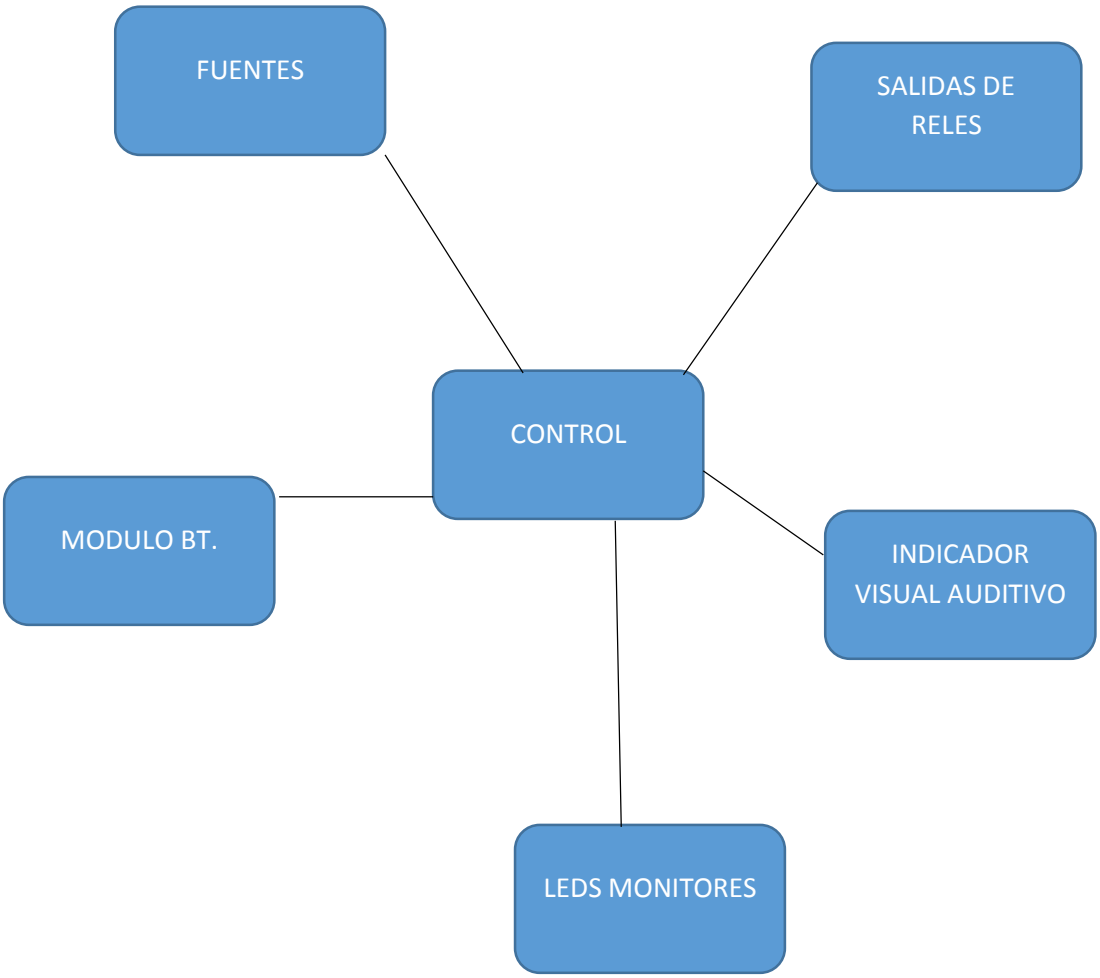
Figura 3.6: Modulo Bluetooth



(Fuente: Investigador, Enero 2015)

### 3.2.2 Diagrama de Bloques

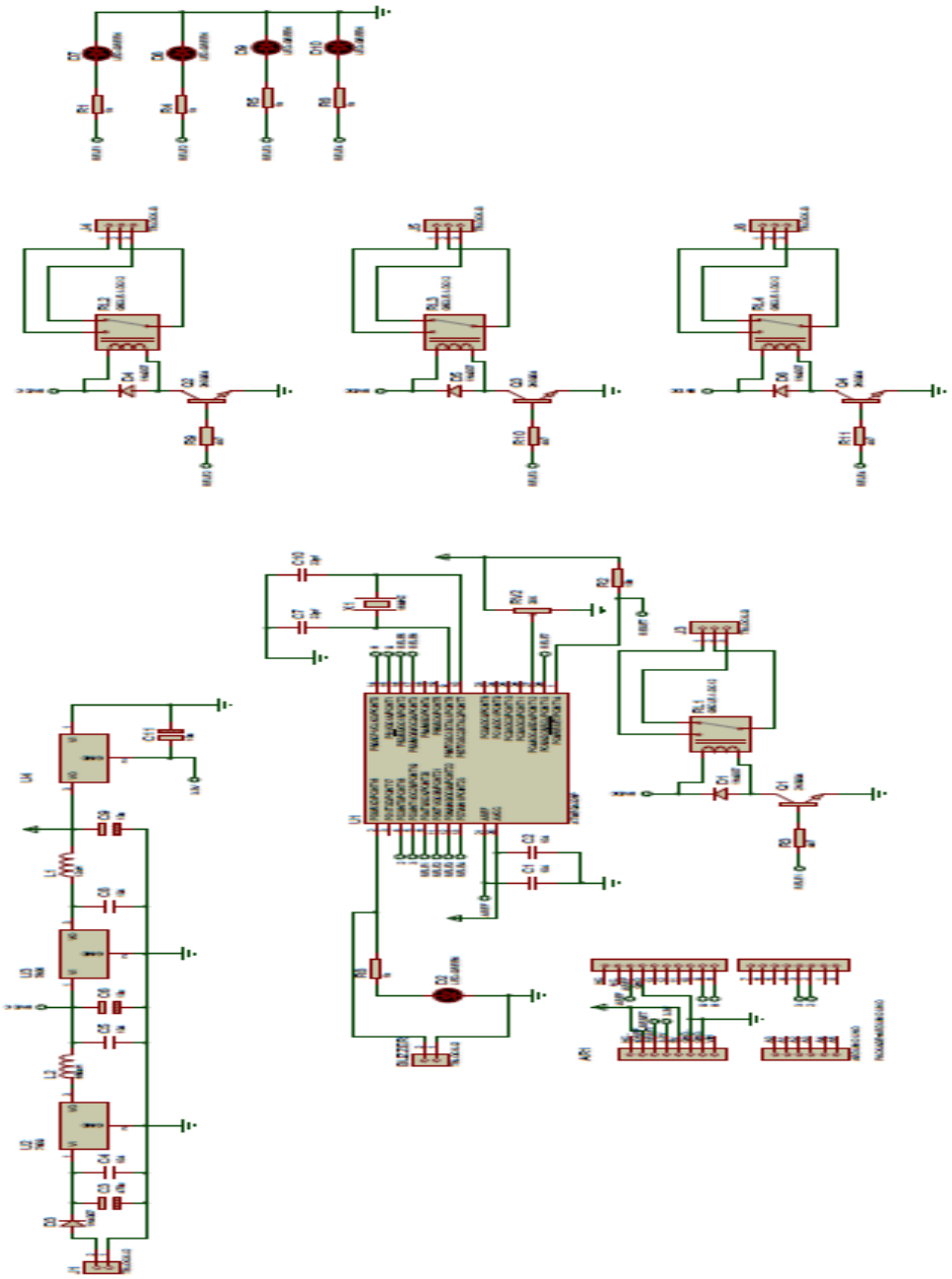
Figura 3.7: Diagrama de Bloques




(Fuente: Investigador, Enero 2015)

### 3.2.3 Diagrama Esquemático

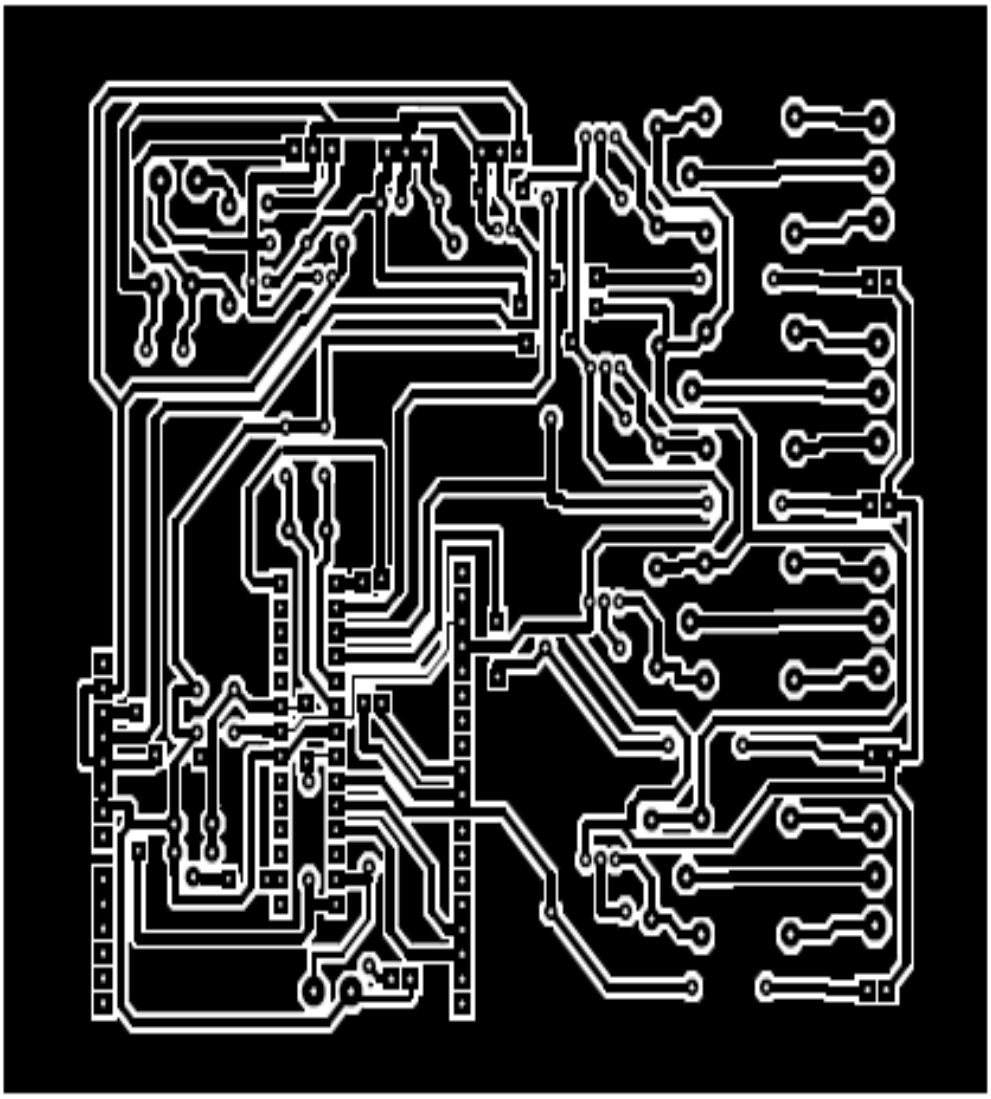
Figura 3.8: Diagrama Esquemático



(Fuente: Investigador, Enero 2015)

### 3.2.4 Diagrama Pcb

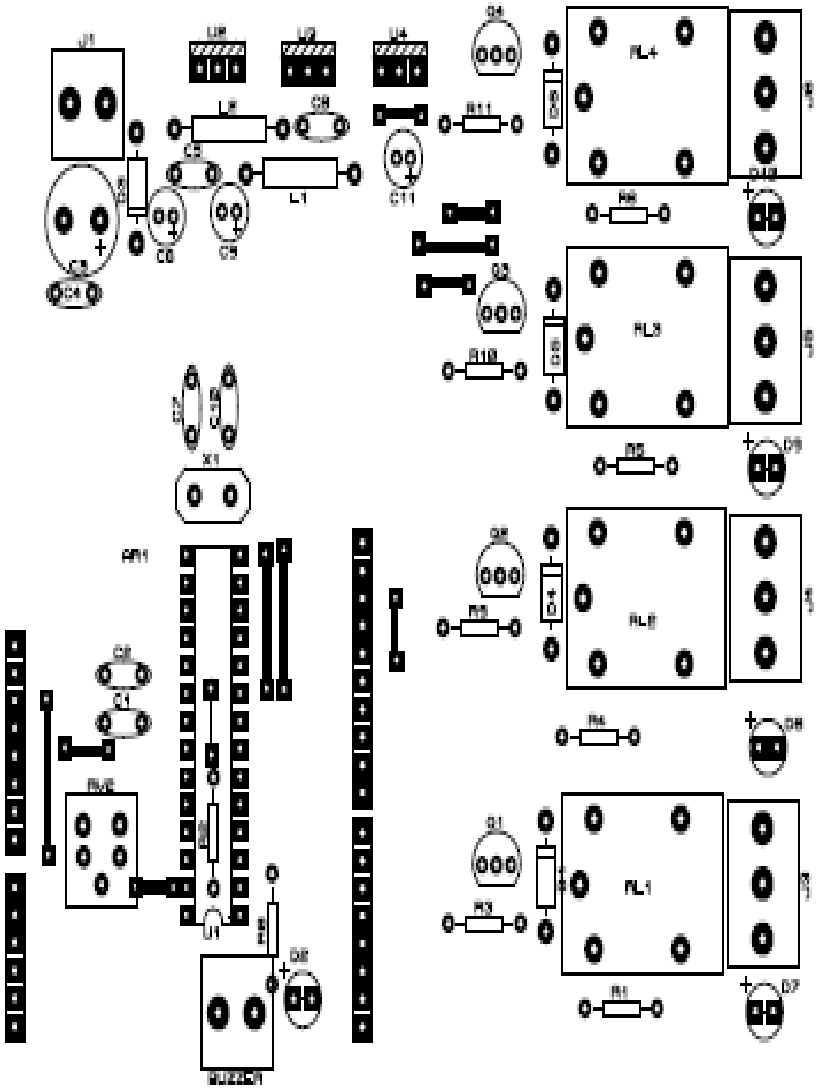
Figura 3.9: Diagrama pcb



(Fuente: Investigador, Enero 2015)

### 3.2.5 Diagrama Screen

Figura 3.10: Diagrama screen




(Fuente: Investigador, Enero 2015)

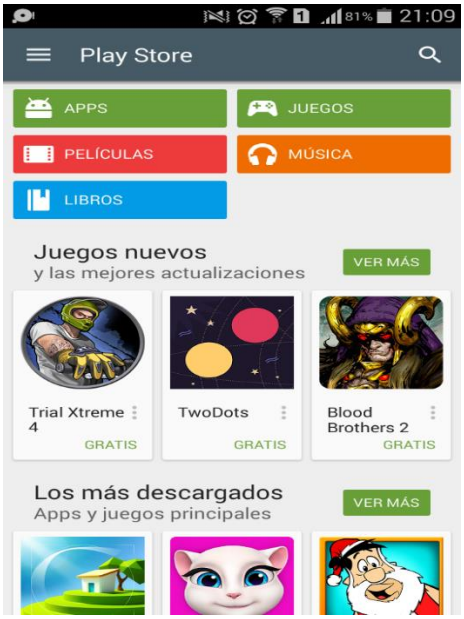
### 3.3 PROCEDIMIENTO

#### 3.3.1 Descarga de Android en el Dispositivo Móvil

A continuación se mostrara los pasos que se realizó para la obtención de la aplicación Blue term.

- Ingresamos a google play que es la tienda de aplicación de Android.

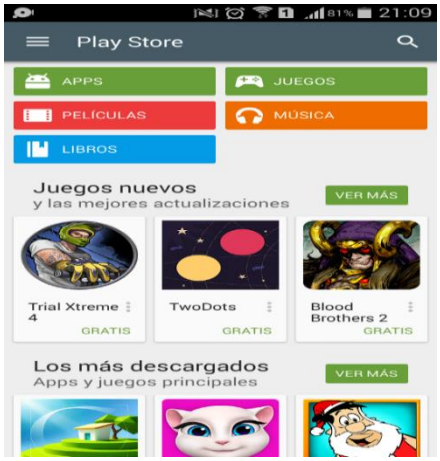
Figura 3.11: Google play




(Fuente: Investigador, Enero 2015)

- Se realiza la búsqueda de la aplicación en cuestión (Blue term)

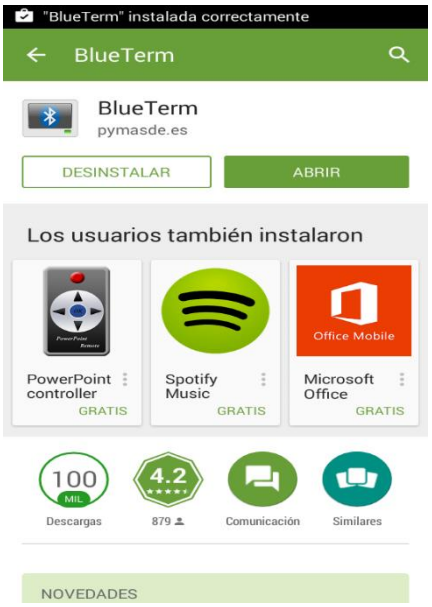
Figura 3.12: Digitación de Blue term




(Fuente: Investigador, Enero 2015)

- Aparece la opción de descarga para la búsqueda realizada

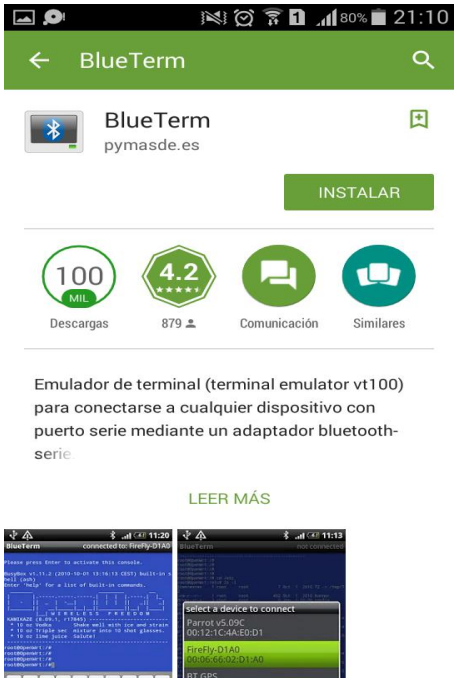
Figura 3.13: Pantalla de descarga




(Fuente: Investigador, Enero 2015)

- Se realiza la instalación de la aplicación en el móvil

Figura 3.14: Instalación de Blue term




(Fuente: Investigador, Enero 2015)

- Una vez ya instalado la aplicación, se inicia por primera vez. Para poder ingresar a la aplicación este solicita encender el bluetooth del dispositivo móvil.

Figura 3.15: Activación de Blue term


(Fuente: Investigador, Enero 2015)

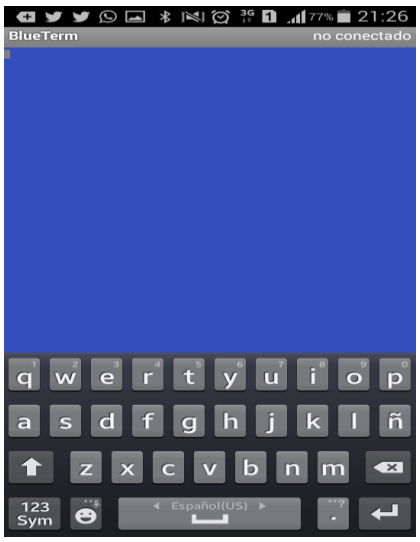
- Luego de ingresar a la aplicación, aparece una ventana dándonos a conocer las características técnicas.

Figura 3.16: Características Técnicas


(Fuente: Investigador, Enero 2015)

- A continuación se muestra como la aplicación realiza la conexión con el dispositivo que se desee conectar.

Figura 3.17: Conexión con Arduino

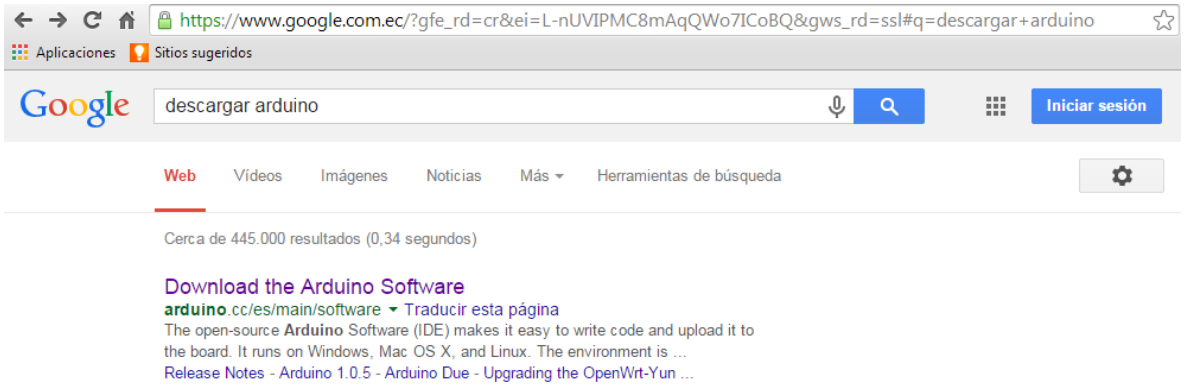



(Fuente: Investigador, Enero 2015)

### 3.3.2 Descarga de Plataforma Arduino

Se mostrara la descarga de Arduino en el Sistema Operativo Windows y su respectiva programación.

Figura 3.18: Búsqueda en Google




(Fuente: Investigador, Enero 2015)

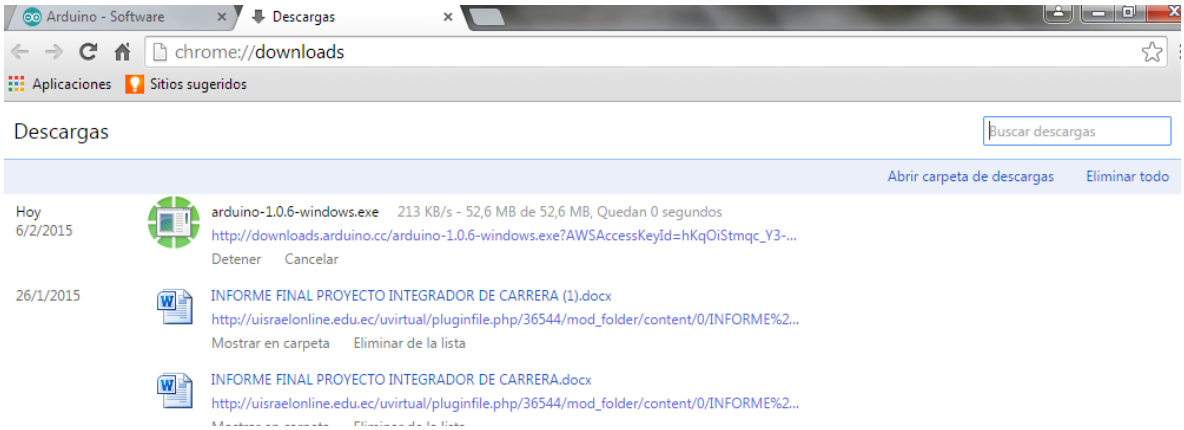
- Ingresamos a la plataforma de Arduino para descargar el software.

Figura 3.19: Plataforma Arduino


(Fuente: Investigador, Enero 2015)

- Descargamos el software

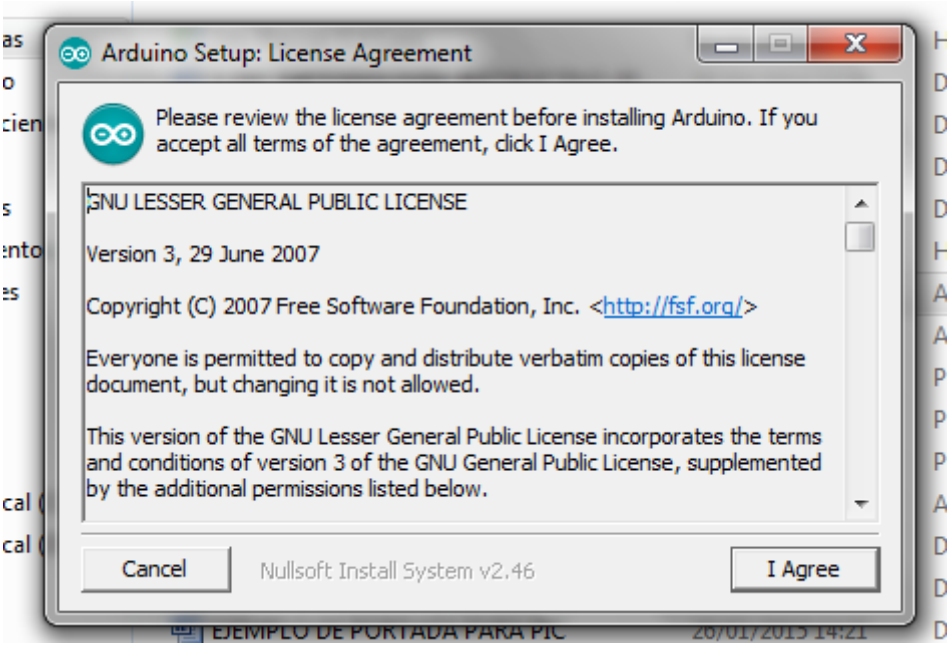
Figura 3.20: descarga




(Fuente: Investigador, Enero 2015)

- Aceptación de licencia de Arduino

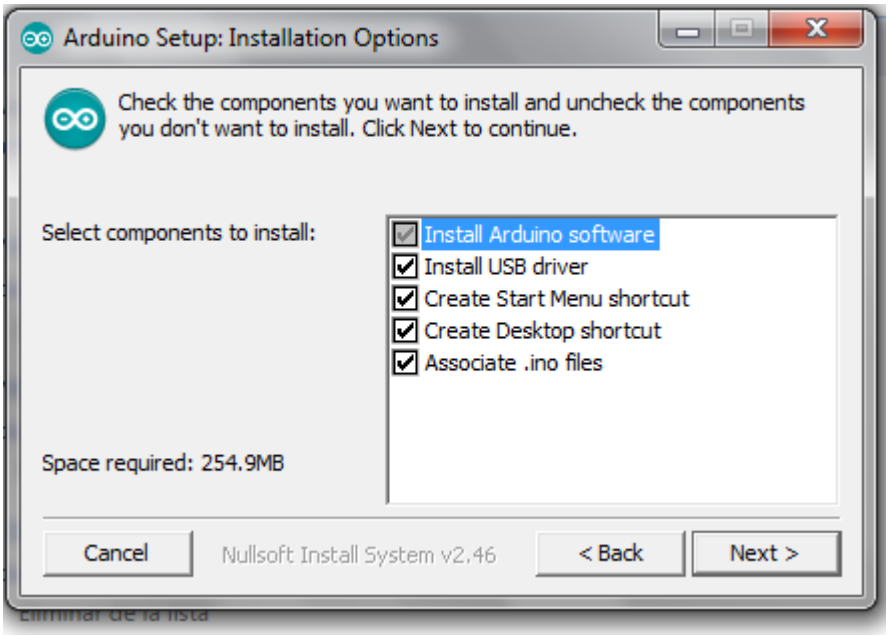
Figura 3.21: Licencia




(Fuente: Investigador, Enero 2015)

- Instalación en Windows

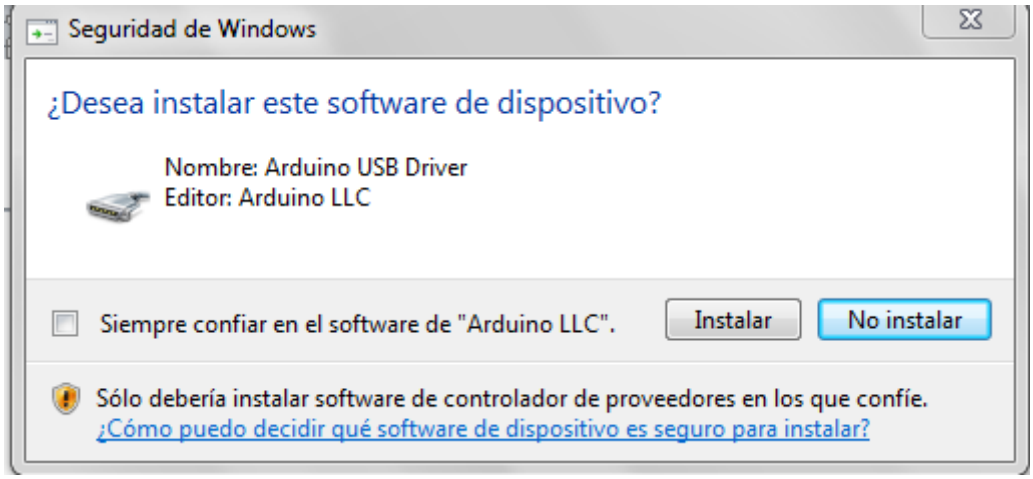
Figura 3.22: Opciones de instalación




(Fuente: Investigador, Enero 2015)

- Aceptación de software

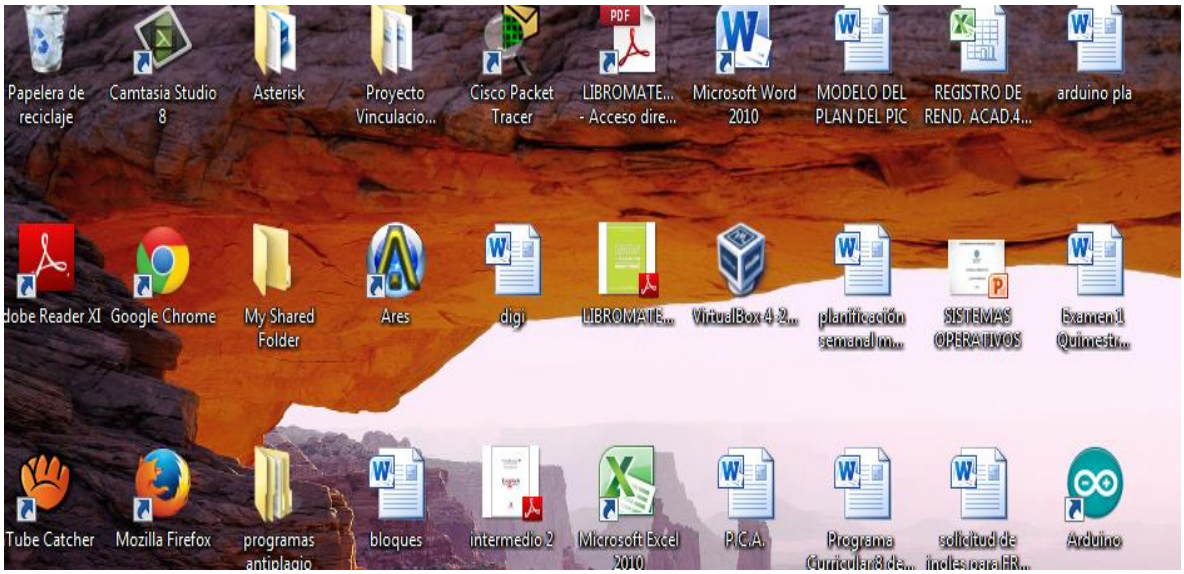
Figura 3.23: Confirmación de Instalación




(Fuente: Investigador, Enero 2015)

- Tenemos Arduino instalado en nuestro escritorio

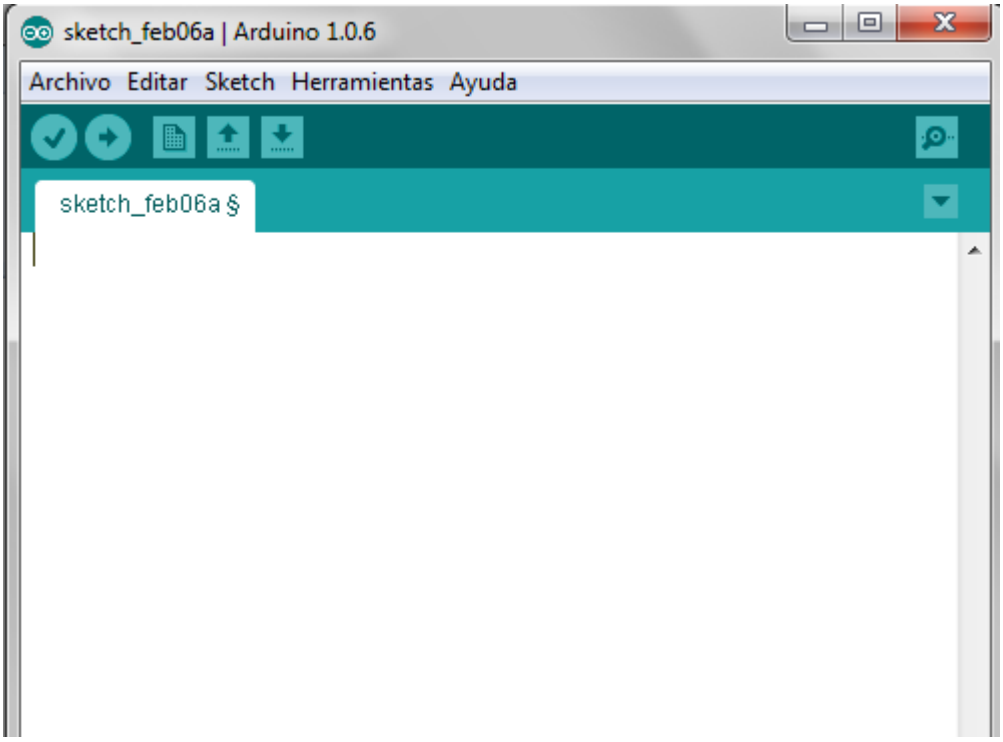
Figura 3.24: Icono de Arduino




(Fuente: Investigador, Enero 2015)

- Pantalla de programación de Arduino

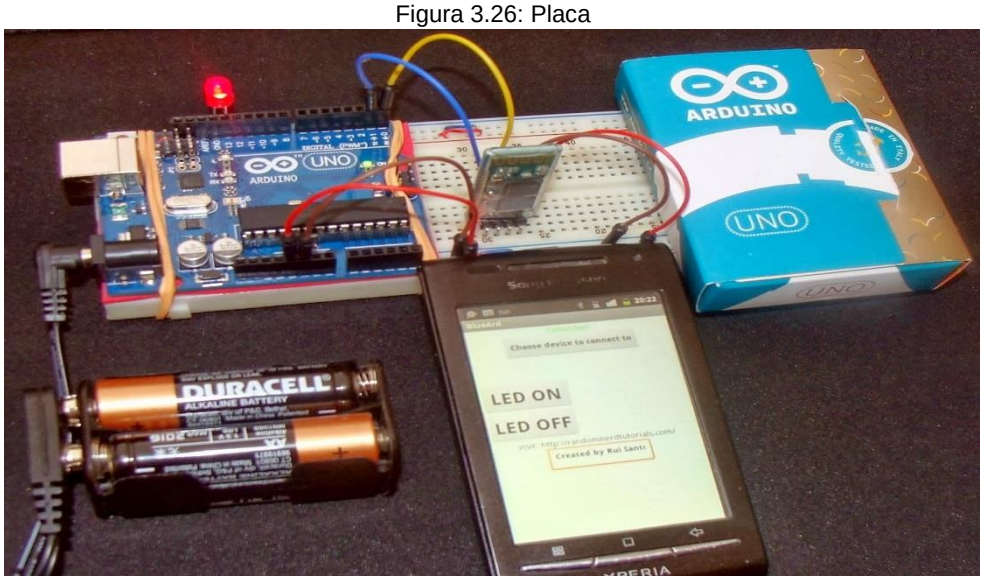
Figura 3.26: Programa




(Fuente: Investigador, Enero 2015)

- Placa Arduino conectado a Sistema Android

Figura 3.26: Placa




(Fuente: Investigador, Enero 2015)

La programación que realizamos en Arduino para realizar el proyecto se encuentra en la parte de ANEXOS la cual se encuentra en la parte final de nuestro informe.

### 3.4 DIFERENCIAS ENTRE LAS SEGURIDADES DE PARTES AUTOMOTRICES DEL AUTO

A continuación se muestra un cuadro con las dos partes que más se utiliza para implementar una seguridad electrónica.

| N° | PARTES AUTOMOTRICES | CARACTERISTICAS |
|---|---|---|
| 1 | Bomba de Gasolina | En esta parte del auto es mucho más recomendable implementar una seguridad electrónica ya que bloquea totalmente al auto dejándolo inmóvil, ya que no va a permitir que se transmita gasolina al motor y va impedir totalmente su encendido |
| 2 | Alarma | Las seguridades implementadas en la alarma del auto son métodos que ya son utilizados por nuestra sociedad, como en el caso del CHEVY que ingresamos códigos en el control de la alarma para poder encender el auto, pero existe el peligro de desactivar el código y encender el auto. |

Tabla 3.2: Diferencias de partes automotrices

(Fuente: Investigador, Enero 2015)

### 3.5 ANÁLISIS ECONÓMICO

En el siguiente se cuadro se detalla los elementos más importantes que conformaron nuestra placa, poniendo en consideración su precio.

| ELEMENTO | CANTIDAD | VALOR UNITARIO | TOTAL |
|---|---|---|---|
| **Arduino** | 1 | 40 | 40 |
| **PIC ATMEGA 328** | 1 | 27 | 27 |
| **Bluetooth HC-06** | 1 | 19 | 38 |
| **regulador de 3 voltios LM111** | 2 | 19 | 38 |

Tabla 3.3: Análisis económico

(Fuente: Investigador, Enero 2015)

### 3.6 PRUEBAS DE FUNCIONAMIENTO

Para la validación del funcionamiento del proyecto se ha realizado dos tipos de pruebas que son:

Prueba 1: Alcance del bluetooth

Prueba 2: Comportamiento del automóvil al momento de bloquear la bomba de gasolina a través del control (Smartphone)

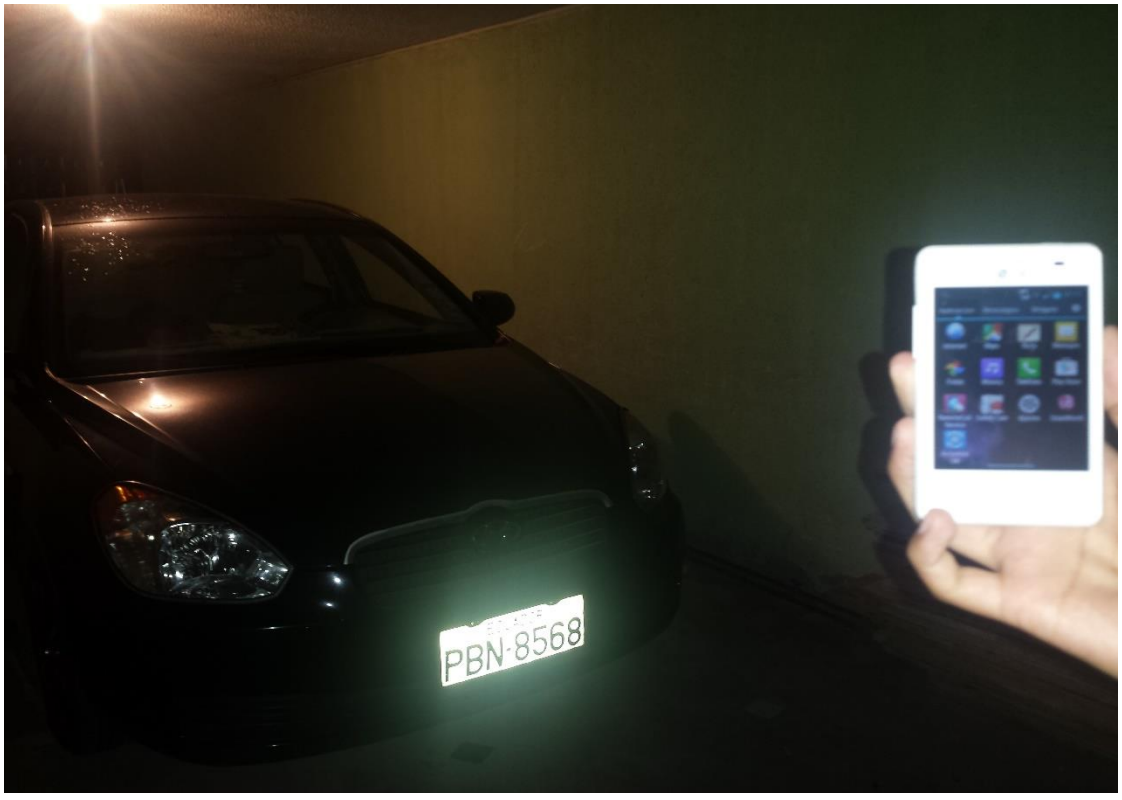
Prueba 3: Comportamiento del automóvil al momento de accionar los tres procesos que son:

- Poner en contacto
- Activar sistema de accesorios (plumas, radio, luces, etc.)
- Encendido del auto

Prueba 1

La prueba consiste en verificar el alcance que tiene el bluetooth de nuestro dispositivo móvil al conectar con el prototipo instalado en el auto.

Figura 3.27: prueba 1




(Fuente: Investigador, Enero 2015)

En esta prueba el dispositivo tuvo un alcance de 9 a 10m

Prueba 2

Esta prueba consiste en comprobar que el bloqueo este activado y de esa manera no pueda encender el auto.

Figura 3-28: Prueba 2

(Fuente: Investigador, Enero 2015)

Se comprobó que el prototipo instalado en la bomba de gasolina funciono correctamente y el auto quedo completamente inmóvil.

Prueba 3

Esta prueba consiste en realizar los tres pasos antes mencionados utilizando nuestro dispositivo móvil.

Figura 3.29: Prueba 3


(Fuente: Investigador, Enero 2015)

Se pudo comprobar que se activaron todos los pasos mencionados y el auto pudo encender.

Se puso en contacto, después se activaron lo accesorios y prendió el auto.

## CONCLUSIONES

- A lo largo de la Investigación, se encontró varios dispositivos móviles para realizar el proyecto, en cuanto al alcance de la señal bluetooth se pudo comprobar que depende de las características del equipo.
- Se puede trabajar directo con el voltaje de la batería del automóvil 14.5VDC debido a que la placa posee reguladores de voltaje.
- Las diferentes pruebas realizadas determinan que no existen cruces de señal al momento del envió de datos seriales por bluetooth.
- Durante el transcurso del proyecto se logró entablar la comunicación serial entre Arduino y un Dispositivo móvil con Sistema Android para la creación del control de bloqueo a través de la bomba de gasolina.
- En base al software del diseño electrónico de nombre Proteus, se diseñó el esquema y la placa del circuito impreso.
- Implementamos un control de bloqueo a través de la bomba de gasolina para un automóvil utilizando tecnología Arduino, teniendo en cuenta los requerimientos del proyecto.
- Una vez ya conectado la bomba de gasolina del auto hacia la etapa de salida instalada en la placa a través del relé, se pudo validar el funcionamiento del proyecto.
- Las encuestas demostraron que la mayoría de personas tienen un sistema móvil con Android, y se encuentran muy interesados en este tipo de seguridad.

## RECOMENDACIONES

- Se recomienda incorporar la placa electrónica en una caja impermeable para evitar que ingrese polvo y agua debido a que se instaló en una parte del automóvil donde se podría derramar gasolina.
- Mantener la clave que viene por defecto o anotar en algún lugar seguro la nueva clave que le dará al dispositivo bluetooth, ya que esta es nuestra seguridad primordial para poder bloquear y encender nuestro auto.
- Mantener en un lugar seguro nuestro dispositivo móvil, ya que es nuestro control de bloqueo del auto.

## BIBLIOGRAFÍA


Arduino. (2014). Arduino. Recuperado el MAYO de 2014, de http://www.arduino.cc/

Margolis, M. (2011). Arduino Cookbook. United States of America: O'Reilly Media, Inc...

Carlos Reyes (2011) Programación en Basic E. Reyes Ecuador

Juan Penagos (2010) Ecuador Como programar en lenguaje C los Microcontroladores, de Penagos- Plaza

Arduino, T. (28 de Diciembre de 2013). Taller Arduino. Recuperado el 2014, de http://tallerarduino.com/tag/teclado

Leonardo Rojas (2008) Ecuador, Mecánica Automotriz, de INACAP

Plataforma Arduino, Recuperado el Mayo de 2014, de http://www.arduino.cc/es/pmwiki.php?n=

# ANEXOS

# ANEXO 1
# PROGRAMACIÓN

```
#include <LiquidCrystal.h>
#include <SoftwareSerial.h> //Software Serial Port

#define uint8 unsigned char
#define uint16 unsigned int
#define uint32 unsigned long int

#define RxD 2 // This is the pin that the Bluetooth (BT_TX) will transmit to the Arduino (RxD)
#define TxD 3 // This is the pin that the Bluetooth (BT_RX) will receive from the Arduino (TxD)

#define DEBUG_ENABLED 1

LiquidCrystal lcd(13, 1, 12, 17, 16, 15, 14);//1 desabilitado



int valor1=0;
int valor2=0;
int valor3=0;
int valor4=0;
int valor5=0;
int valor6=0;
int valor7=0;

int bandera1=0;
int bandera2=0;
int bandera3=0;
int bandera4=0;
int bandera5=0;
int bandera6=0;
int bandera7=0;
```

```
int indicador=0;
int rele1=4;
int rele2=5;
int rele3=6;
int rele4=7;


int Clkpin = 9; //RGB LED Clock Pin (Digital 9)
int Datapin = 8; //RGB LED Data Pin (Digital 8)
int dato; //Creamos una variable de nombre dato
SoftwareSerial blueToothSerial(RxD,TxD);

/*----------------------SETUP----------------------------*/
void setup() {
 //Serial.begin(9600); // Allow Serial communication via USB cable to computer (if required)
 pinMode(RxD, INPUT); // Setup the Arduino to receive INPUT from the bluetooth shield on Digital Pin 6
 pinMode(TxD, OUTPUT); // Setup the Arduino to send data (OUTPUT) to the bluetooth shield on Digital Pin 7

 pinMode(rele1,OUTPUT); // Use onboard LED if required.
 pinMode(rele2,OUTPUT); // Use onboard LED if required.
 pinMode(rele3,OUTPUT); // Use onboard LED if required.
 pinMode(rele4,OUTPUT); // Use onboard LED if required.


 pinMode(indicador,OUTPUT); // Use onboard LED if required.

 setupBlueToothConnection(); //Used to initialise the Bluetooth shield

 pinMode(Datapin, OUTPUT); // Setup the RGB LED Data Pin
 pinMode(Clkpin, OUTPUT); // Setup the RGB LED Clock pin
```

```
 lcd.clear();
 lcd.begin(16,2);

rele1,LOW;
rele2,LOW;
rele3,LOW;
rele4,LOW;



indicador,LOW;
pantalla();
}

/*----------------------LOOP----------------------------*/
void loop() {


 char recvChar;
 while(1){
 if(blueToothSerial.available()){//check if there's any data sent from the remote bluetooth shield
 recvChar = blueToothSerial.read();
 //Serial.print(recvChar); // Print the character received to the Serial Monitor (if required)



 if(recvChar=='a'){
 Send32Zero(); // begin
 DataDealWithAndSend(255, 0, 0); // first node data
 Send32Zero(); // send to update data
 bandera1=1;


   if(valor1==1 && bandera1==1){
   digitalWrite(rele1,LOW);
   valor1=0;
```

```
    delay(100);
    bandera1=0;
    beep();
    pantalla();
    }

     if(valor1==0 && bandera1==1){
    digitalWrite(rele1,HIGH);
    valor1=1;
    delay(100);
    bandera1=0;
    beep();
    pantalla();
}

 }




 if(recvChar=='b'){
 Send32Zero(); // begin
 DataDealWithAndSend(255, 0, 0); // first node data
 Send32Zero(); // send to update data
 bandera2=1;

    if(valor2==1 && bandera2==1){
    digitalWrite(rele2,LOW);
    valor2=0;
    delay(100);
    bandera2=0;
    beep();
```

```
  pantalla();
  }

   if(valor2==0 && bandera2==1){
  digitalWrite(rele2,HIGH);
  valor2=1;
  delay(100);
  bandera2=0;
  beep();
  pantalla();
 }

}
```

```
if(recvChar=='c'){
Send32Zero(); // begin
DataDealWithAndSend(255, 0, 0); // first node data
Send32Zero(); // send to update data
bandera3=1;

  if(valor3==1 && bandera3==1){
  digitalWrite(rele3,LOW);
  valor3=0;
  delay(100);
  bandera3=0;
  beep();
  pantalla();
  }
```

```
    if(valor3==0 && bandera3==1){
   digitalWrite(rele3,HIGH);
   valor3=1;
   delay(100);
   bandera3=0;
   beep();
   pantalla();
 }

}
```

```
if(recvChar=='d'){
Send32Zero(); // begin
DataDealWithAndSend(255, 0, 0); // first node data
Send32Zero(); // send to update data
bandera4=1;

   if(valor4==1 && bandera4==1){
   digitalWrite(rele4,LOW);
   valor4=0;
   delay(100);
   bandera4=0;
   beep();
   pantalla();
   }

    if(valor4==0 && bandera4==1){
   digitalWrite(rele4,HIGH);
```

```
    valor4=1;
    delay(100);
    bandera4=0;
    beep();
    pantalla();
  }

 }
```

```
 }
```

```
 //You can use the following code to deal with any information coming from the Computer
(serial monitor)
 if(Serial.available()){
 recvChar = Serial.read();

 //This will send value obtained (recvChar) to the phone. The value will be displayed on the
phone.
 blueToothSerial.print(recvChar);
 }
 }
}
```

```
//The following code is necessary to setup the bluetooth shield ------copy and paste----------
---
void setupBlueToothConnection()
{
 blueToothSerial.begin(38400); //Set BluetoothBee BaudRate to default baud rate 38400
 blueToothSerial.print("\r\n+STWMOD=0\r\n"); //set the bluetooth work in slave mode
 blueToothSerial.print("\r\n+STNA=SeeedBTSlave\r\n"); //set the bluetooth name as
"SeeedBTSlave"
 blueToothSerial.print("\r\n+STOAUT=1\r\n"); // Permit Paired device to connect me
 blueToothSerial.print("\r\n+STAUTO=0\r\n"); // Auto-connection should be forbidden here
 delay(2000); // This delay is required.
 blueToothSerial.print("\r\n+INQ=1\r\n"); //make the slave bluetooth inquirable
 //Serial.println("The slave bluetooth is inquirable!");
 delay(2000); // This delay is required.
 blueToothSerial.flush();
}


//The following code snippets are used update the colour of the RGB LED-----copy and
paste------------
void ClkProduce(void){
 digitalWrite(Clkpin, LOW);
 delayMicroseconds(20);
 digitalWrite(Clkpin, HIGH);
 delayMicroseconds(20);
}

void Send32Zero(void){
 unsigned char i;
 for (i=0; i<32; i++){
 digitalWrite(Datapin, LOW);
 ClkProduce();
 }
```

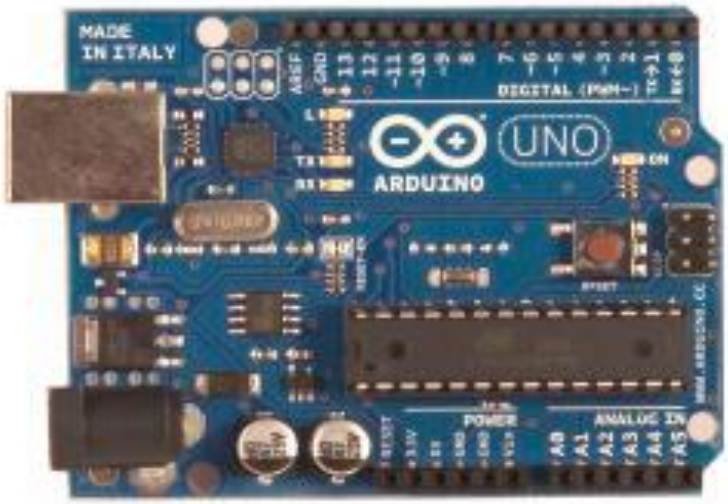
```
}

uint8 TakeAntiCode(uint8 dat){
 uint8 tmp = 0;
 if ((dat & 0x80) == 0){
 tmp |= 0x02;
 }

 if ((dat & 0x40) == 0){
 tmp |= 0x01;
 }

 return tmp;
}

// gray data
void DatSend(uint32 dx){
 uint8 i;
 for (i=0; i<32; i++){
 if ((dx & 0x80000000) != 0){
 digitalWrite(Datapin, HIGH);
 } else {
 digitalWrite(Datapin, LOW);
 }

 dx <<= 1;
 ClkProduce();
 }
}

// data processing
void DataDealWithAndSend(uint8 r, uint8 g, uint8 b){
```

```
 uint32 dx = 0;

 dx |= (uint32)0x03 << 30; // highest two bits 1,  flag bits
 dx |= (uint32)TakeAntiCode(b) << 28;
 dx |= (uint32)TakeAntiCode(g) << 26;
 dx |= (uint32)TakeAntiCode(r) << 24;

 dx |= (uint32)b << 16;
 dx |= (uint32)g << 8;
 dx |= r;

 DatSend(dx);
}



void beep()
{
 digitalWrite(indicador,HIGH); //Turn off the onboard Arduino LED
 delay(100); // This delay is required.
 digitalWrite(indicador,LOW); //Turn off the onboard Arduino LED
}

void pantalla(){
 lcd.setCursor(0, 0);
 lcd.print(" a b c d e f g  ");
 lcd.setCursor(1, 1);
 lcd.print(valor1);
 lcd.setCursor(3, 1);
 lcd.print(valor2);
 lcd.setCursor(5, 1);
 lcd.print(valor3);
```

```
 lcd.setCursor(7, 1);
 lcd.print(valor4);
 lcd.setCursor(9, 1);
 lcd.print(valor5);
 lcd.setCursor(11, 1);
 lcd.print(valor6);
 lcd.setCursor(13, 1);
 lcd.print(valor7);
}
```

# ANEXO 5
# Datasheet arduino

# Arduino UNO




CE

## Product Overview

The Arduino Uno is a microcontroller board based on the ATmega328 (datasheet). It has 14 digital input/output pins (of which 6 can be used as PWM outputs), 6 analog inputs, a 16 MHz crystal oscillator, a USB connection, a power jack, an ICSP header, and a reset button. It contains everything needed to support the microcontroller; simply connect it to a computer with a USB cable or power it with a AC-to-DC adapter or battery to get started. The Uno differs from all preceding boards in that it does not use the FTDI USB-to-serial driver chip. Instead, it features the Atmega8U2 programmed as a USB-to-serial converter.

"Uno" means one in Italian and is named to mark the upcoming release of Arduino 1.0. The Uno and version 1.0 will be the reference versions of Arduino, moving forward. The Uno is the latest in a series of USB Arduino boards, and the reference model for the Arduino platform; for a comparison with previous versions, see the index of Arduino boards.

## Index

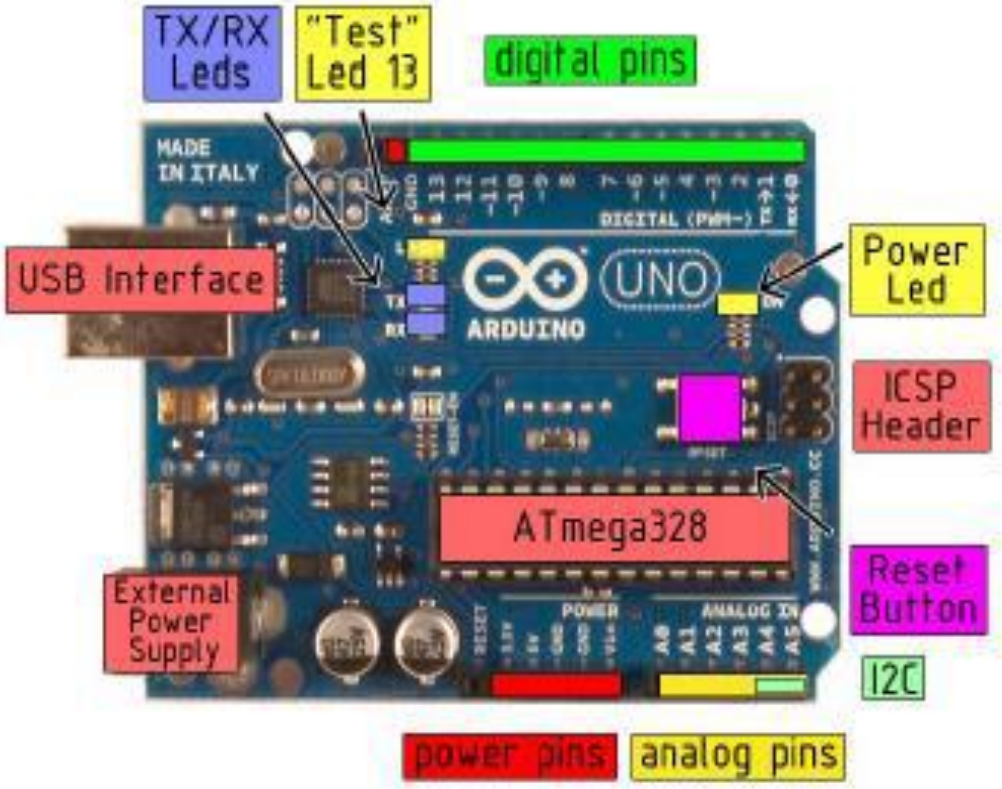
# Technical Specification

EAGLE files: arduino-duemilanove-uno-design.zip Schematic: arduino-uno-schematic.pdf

## Summary

| | |
|---|---|
| Microcontroller | ATmega328 |
| Operating Voltage | 5V |
| Input Voltage (recommended) | 7-12V |
| Input Voltage (limits) | 6-20V |
| 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 |

## the board



## Power

The Arduino Uno can be powered via the USB connection or with an external power supply. The power source is selected automatically.

External (non-USB) power can come either from an AC-to-DC adapter (wall-wart) or battery. The adapter can be connected by plugging a 2.1mm center-positive plug into the board's power jack. Leads from a battery can be inserted in the Gnd and Vin pin headers of the POWER connector.

The board can operate on an external supply of 6 to 20 volts. If supplied with less than 7V, however, the 5V pin may supply less than five volts and the board may be unstable. If using more than 12V, the voltage regulator may overheat and damage the board. The recommended range is 7 to 12 volts.

The power pins are as follows:

- **VIN.** The input voltage to the Arduino board when it's using an external power source (as opposed to 5 volts from the USB connection or other regulated power source). You can supply voltage through this pin, or, if supplying voltage via the power jack, access it through this pin.
- **5V.** The regulated power supply used to power the microcontroller and other components on the board. This can come either from VIN via an on-board regulator, or be supplied by USB or another regulated 5V supply.
- **3V3.** A 3.3 volt supply generated by the on-board regulator. Maximum current draw is 50 mA.
- **GND.** Ground pins.

## Memory

The Atmega328 has 32 KB of flash memory for storing code (of which 0,5 KB is used for the bootloader); It has also 2 KB of SRAM and 1 KB of EEPROM (which can be read and written with the EEPROM library).

## Input and Output

Each of the 14 digital pins on the Uno can be used as an input or output, using pinMode(), digitalWrite(), and digitalRead() functions. They operate at 5 volts. Each pin can provide or receive a maximum of 40 mA and has an internal pull-up resistor (disconnected by default) of 20-50 kOhms. In addition, some pins have specialized functions:

- **Serial: 0 (RX) and 1 (TX).** Used to receive (RX) and transmit (TX) TTL serial data. TThese pins are connected to the corresponding pins of the ATmega8U2 USB-to-TTL Serial chip .
- **External Interrupts: 2 and 3.** These pins can be configured to trigger an interrupt on a low value, a rising or falling edge, or a change in value. See the attachInterrupt() function for details.
- **PWM: 3, 5, 6, 9, 10, and 11.** Provide 8-bit PWM output with the analogWrite() function.
- **SPI: 10 (SS), 11 (MOSI), 12 (MISO), 13 (SCK).** These pins support SPI communication, which, although provided by the underlying hardware, is not currently included in the Arduino language.

- **LED: 13.** There is a built-in LED connected to digital pin 13. When the pin is HIGH value, the LED is on, when the pin is LOW, it's off.

The Uno has 6 analog inputs, each of which provide 10 bits of resolution (i.e. 1024 different values). By default they measure from ground to 5 volts, though is it possible to change the upper end of their range using the AREF pin and the analogReference() function. Additionally, some pins have specialized functionality:

- I²C: **4 (SDA) and 5 (SCL)**. Support I²C (TWI) communication using the Wire library.

There are a couple of other pins on the board:

- **AREF**. Reference voltage for the analog inputs. Used with analogReference().
- **Reset**. Bring this line LOW to reset the microcontroller. Typically used to add a reset button to shields which block the one on the board.

See also the mapping between Arduino pins and Atmega328 ports.

## Communication

The Arduino Uno has a number of facilities for communicating with a computer, another Arduino, or other microcontrollers. The ATmega328 provides UART TTL (5V) serial communication, which is available on digital pins 0 (RX) and 1 (TX). An ATmega8U2 on the board channels this serial communication over USB and appears as a virtual com port to software on the computer. The '8U2 firmware uses the standard USB COM drivers, and no external driver is needed. However, on Windows, an ".inf file is required..

The Arduino software includes a serial monitor which allows simple textual data to be sent to and from the Arduino board. The RX and TX LEDs on the board will flash when data is being transmitted via the USB-to-serial chip and USB connection to the computer (but not for serial communication on pins 0 and 1).

A SoftwareSerial library allows for serial communication on any of the Uno's digital pins.

The ATmega328 also support I2C (TWI) and SPI communication. The Arduino software includes a Wire library to simplify use of the I2C bus; see the documentation for details. To use the SPI communication, please see the ATmega328 datasheet.

## Programming

The Arduino Uno can be programmed with the Arduino software (download). Select "Arduino Uno w/ ATmega328" from the **Tools** > **Board** menu (according to the microcontroller on your board). For details, see the reference and tutorials.

The ATmega328 on the Arduino Uno comes preburned with a bootloader that allows you to upload new code to it without the use of an external hardware programmer. It communicates using the original STK500 protocol (reference, C header files).

You can also bypass the bootloader and program the microcontroller through the ICSP (In-Circuit Serial Programming) header; see these instructions for details.

The ATmega8U2 firmware source code is available . The ATmega8U2 is loaded with a DFU bootloader, which can be activated by connecting the solder jumper on the back of the board (near the map of Italy) and then resetting the 8U2. You can then use Atmel's FLIP software (Windows) or the DFU programmer (Mac OS X and Linux) to load a new firmware. Or you can use the ISP header with an external programmer (overwriting the DFU bootloader).

## Automatic (Software) Reset

Rather than requiring a physical press of the reset button before an upload, the Arduino Uno is designed in a way that allows it to be reset by software running on a connected computer. One of the hardware flow control lines (DTR) of the ATmega8U2 is connected to the reset line of the ATmega328 via a 100 nanofarad capacitor. When this line is asserted (taken low), the reset line drops long enough to reset the chip. The Arduino software uses this capability to allow you to upload code by simply pressing the upload button in the Arduino environment. This means that the bootloader can have a shorter timeout, as the lowering of DTR can be well-coordinated with the start of the upload.

This setup has other implications. When the Uno is connected to either a computer running Mac OS X or Linux, it resets each time a connection is made to it from software (via USB). For the following half-second or so, the bootloader is running on the Uno. While it is programmed to ignore malformed data (i.e. anything besides an upload of new code), it will intercept the first few bytes of data sent to the board after a connection is opened. If a sketch running on the board receives one-time configuration or other data when it first starts, make sure that the software with which it communicates waits a second after opening the connection and before sending this data.

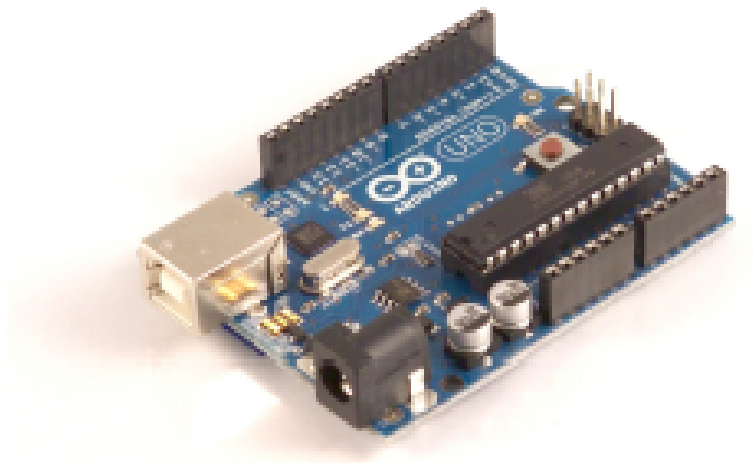
The Uno contains a trace that can be cut to disable the auto-reset. The pads on either side of the trace can be soldered together to re-enable it. It's labeled "RESET-EN". You may also be able to disable the auto-reset by connecting a 110 ohm resistor from 5V to the reset line; see this forum thread for details.

## USB Overcurrent Protection

The Arduino Uno has a resettable polyfuse that protects your computer's USB ports from shorts and overcurrent. Although most computers provide their own internal protection, the fuse provides an extra layer of protection. If more than 500 mA is applied to the USB port, the fuse will automatically break the connection until the short or overload is removed.

## Physical Characteristics

The maximum length and width of the Uno PCB are 2.7 and 2.1 inches respectively, with the USB connector and power jack extending beyond the former dimension. Three screw holes allow the board to be attached to a surface or case. Note that the distance between digital pins 7 and 8 is 160 mil (0.16"), not an even multiple of the 100 mil spacing of the other pins.



# How to use Arduino

Arduino can sense the environment by receiving input from a variety of sensors and can affect its surroundings by controlling lights, motors, and other actuators. The microcontroller on the board is programmed using the Arduino programming language (based on Wiring) and the Arduino development environment (based on Processing). Arduino projects can be stand-alone or they can communicate with software on running on a computer (e.g. Flash, Processing, MaxMSP).

Arduino is a cross-platoform program. You'll have to follow different instructions for your personal OS. Check on the Arduino site for the latest instructions. *http://arduino.cc/en/Guide/HomePage*

## Linux Install

## Windows Install

## Mac Install

Once you have downloaded/unzipped the arduino IDE, you can Plug the Arduino to your PC via USB cable.

## Blink led

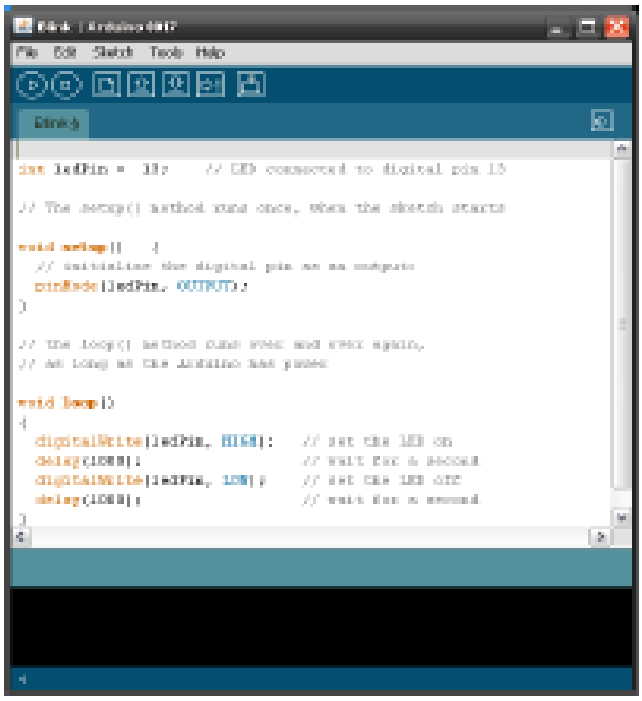
Now you're actually ready to "burn" your first program on the arduino board. To select "blink led", the physical translation of the well known programming "hello world", select

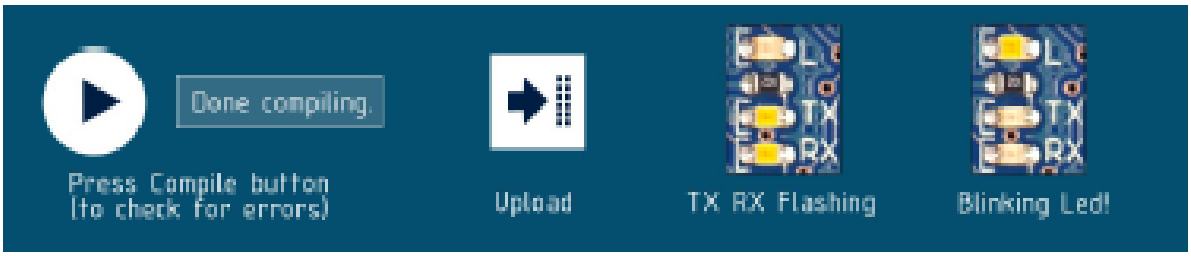
**File>Sketchbook>**
**Arduino-0017>Examples>**
**Digital>Blink**

Once you have your skecth you'll see something very close to the screenshot on the right.

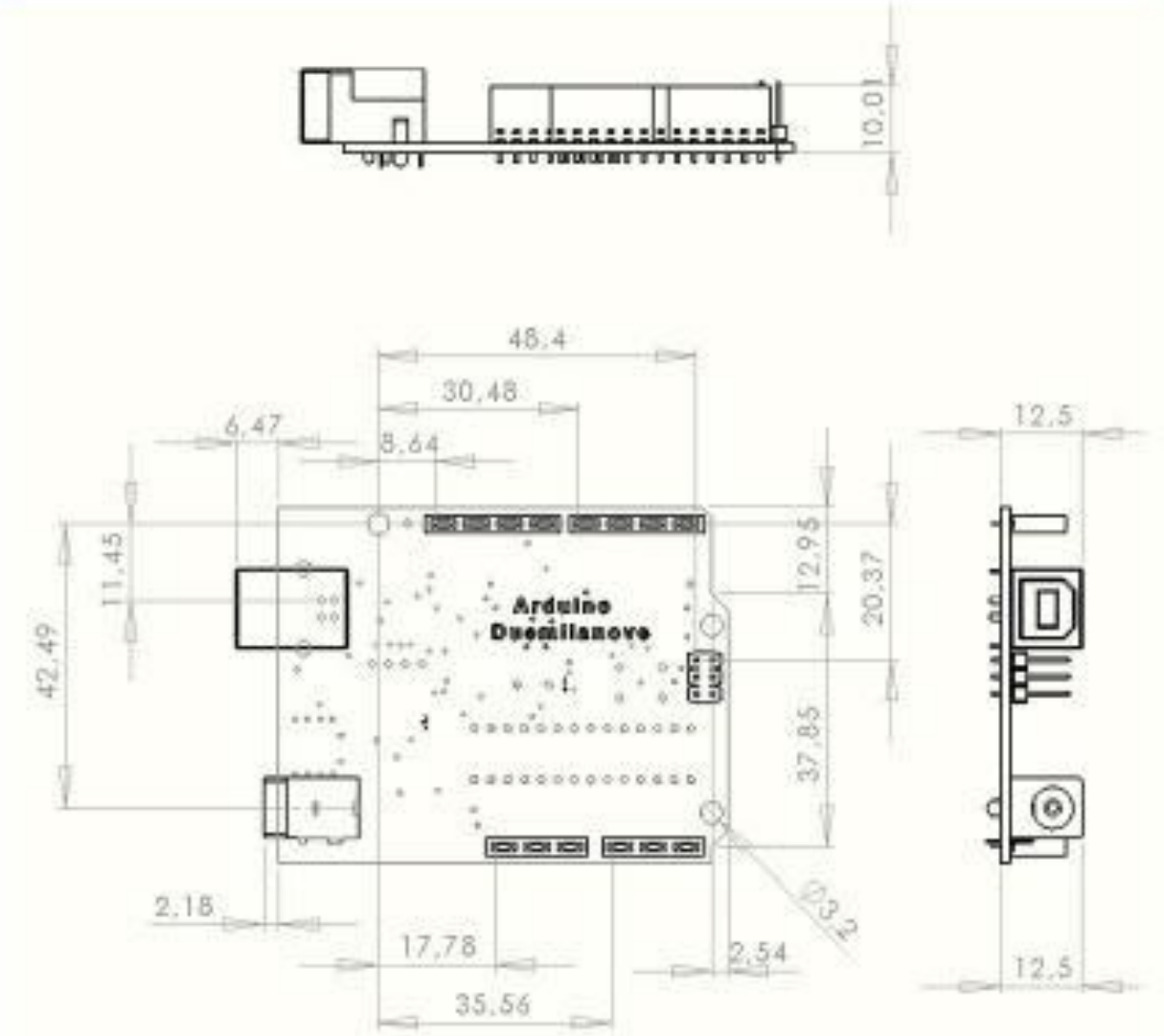
In **Tools>Board** select

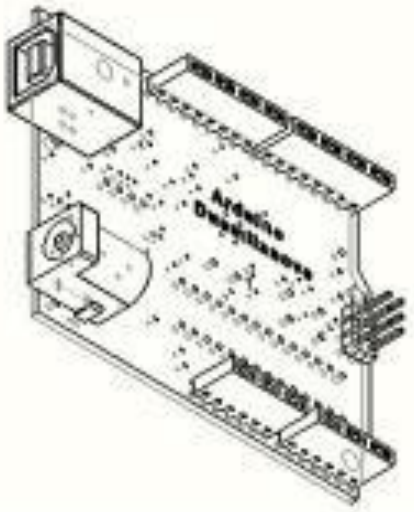
Now you have to go to
**Tools>SerialPort**
and select the right serial port, the one arduino is attached to.



## Dimensioned Drawing



# Terms & Conditions

## 1. Warranties

1.1 The producer warrants that its products will conform to the Specifications. This warranty lasts for one (1) years from the date of the sale. The producer shall not be liable for any defects that are caused by neglect, misuse or mistreatment by the Customer, including improper installation or testing, or for any products that have been altered or modified in any way by a Customer. Moreover, The producer shall not be liable for any defects that result from Customer's design, specifications or instructions for such products. Testing and other quality control techniques are used to the extent the producer deems necessary.

1.2 If any products fail to conform to the warranty set forth above, the producer's sole liability shall be to replace such products. The producer's liability shall be limited to products that are determined by the producer not to conform to such warranty. If the producer elects to replace such products, the producer shall have a reasonable time to replacements. Replaced products shall be warranted for a new full warranty period.

1.3 EXCEPT AS SET FORTH ABOVE, PRODUCTS ARE PROVIDED "AS IS" AND "WITH ALL FAULTS." THE PRODUCER DISCLAIMS ALL OTHER WARRANTIES, EXPRESS OR IMPLIED, REGARDING PRODUCTS, INCLUDING BUT NOT LIMITED TO, ANY IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE

1.4 Customer agrees that prior to using any systems that include the producer products, Customer will test such systems and the functionality of the products as used in such systems. The producer may provide technical, applications or design advice, quality characterization, reliability data or other services. Customer acknowledges and agrees that providing these services shall not expand or otherwise alter the producer's warranties, as set forth above, and no additional obligations or liabilities shall arise from the producer providing such services.

1.5 The Arduino™ products are not authorized for use in safety-critical applications where a failure of the product would reasonably be expected to cause severe personal injury or death. Safety-Critical Applications include, without limitation, life support devices and systems, equipment or systems for the operation of nuclear facilities and weapons systems. Arduino™ products are neither designed nor intended for use in military or aerospace applications or environments and for automotive applications or environment. Customer acknowledges and agrees that any such use of Arduino™ products which is solely at the Customer's risk, and that Customer is solely responsible for compliance with all legal and regulatory requirements in connection with such use.

1.6 Customer acknowledges and agrees that it is solely responsible for compliance with all legal, regulatory and safety-related requirements concerning its products and any use of Arduino™ products in Customer's applications, notwithstanding any applications-related information or support that may be provided by the producer.

## 2. Indemnification

The Customer acknowledges and agrees to defend, indemnify and hold harmless the producer from and against any and all third-party losses, damages, liabilities and expenses it incurs to the extent directly caused by: (i) an actual breach by a Customer of the representation and warranties made under this terms and conditions or (ii) the gross negligence or willful misconduct by the Customer.

## 3. Consequential Damages Waiver

In no event the producer shall be liable to the Customer or any third parties for any special, collateral, indirect, punitive, incidental, consequential or exemplary damages in connection with or arising out of the products provided hereunder, regardless of whether the producer has been advised of the possibility of such damages. This section will survive the termination of the warranty period.

## 4. Changes to specifications

The producer may make changes to specifications and product descriptions at any time, without notice. The Customer must not rely on the absence or characteristics of any features or instructions marked "reserved" or "undefined." The producer reserves these for future definition and shall have no responsibility whatsoever for conflicts or incompatibilities arising from future changes to them. The product information on the Web Site or Materials is subject to change without notice. Do not finalize a design with this information.

# Enviromental Policies

The producer of Arduino™ has joined the Impatto Zero® policy of LifeGate.It. For each Arduino board produced Is created / looked after half squared Km of Costa Rica's forest's.

Anexo 2

# Módulo Datasheet Bluetooth HC-06

HC-06 Bluetooth module datasheet and configuration with Arduino

In a previous post I shared my notes on  how to connect an Arduino to an Android phone using the the popular and cheap HC-06 Bluetooth module. In that example I used the Bluetooth module with its default settings.

That works fine, but some applications may require changing the communication speed (Baud rate), the pairing code, the module name etc. For example, I am trying to set-up a way to program my Arduino Uno and Arduino Pro Mini wirelessly, over Bluetooth. This requires changing the baud rate of the module from the default 9600 to 115200, or 57600, to match the default sketch upload speed for these Arduino boards.

Also, if things are not working, you may want to restore the settings back their defaults and start troubleshooting from there.

There are multiple versions of the module floating around, with different firmware and breakout boards, but the general functionality should match the HC-06 Bluetooth module datasheet.

Step 1: Hook up the HC-06 Bluetooth module to the Arduino

Connect the HC-06 Ground (GND) pin to ground (duh!).

Connect the HC-06 VCC pin to 5v.

Connect the HC-06 TX/TXD pin to Arduino digital pin 4.

Connect the HC-06 RX/RXD pin to Arduino digital pin 2.

It is recommended to use a level shifter, voltage regulator (or a voltage divider, like in my set-up below) to protect the Bluetooth module RX pin. It is designed for 3.3v operation, while the Arduino digital pins work on 5 volts. You do not need the LED on the Arduino pin 13 that I have on my set-up below.

Arduino-JY-MCU-Bluetooth-AT-Commands

Step 2: Upload the Arduino HC-06 configuration sketch

The Arduino sketch below will allow you to configure your HC-06 module using the Arduino IDE serial monitor. The Arduino will act as a middleman between the Bluetooth module and your computer. It will communicate with your PC over the built in serial connection through the USB cable, and with the HC-06 Bluetooth module over pins 4 and 2, using the Software Serial library.

The Software Serial library comes pre-installed with the latest version of the Arduino IDE. It has been developed to allow setting up serial communication on (almost any) digital pin of the Arduino, using software to replicate Arduino's native serial support. See the SoftwareSerial library page for more details on its features and limitations.

```
#include <SoftwareSerial.h>

SoftwareSerial mySerial(4, 2); // RX, TX


String command = ""; // Stores response of the HC-06 Bluetooth device



void setup() {

  // Open serial communications:

  Serial.begin(9600);

  Serial.println("Type AT commands!");


  // The HC-06 defaults to 9600 according to the datasheet.

  mySerial.begin(9600);

}


void loop() {

  // Read device output if available.

  if (mySerial.available()) {

    while(mySerial.available()) { // While there is more to be read, keep reading.
```

```
    command += (char)mySerial.read();
   }

   Serial.println(command);
   command = ""; // No repeats
  }

  // Read user input if available.
  if (Serial.available()){
   delay(10); // The delay is necessary to get this working!
   mySerial.write(Serial.read());
  }
}
```

Credit: The code above is based on this article by user Ecno92

Step 3: HC-06 Bluetooth module configuration using AT commands

The HC-06 configurations options are covered in section 9 of the module's datasheet.

The default settings are:

Name / ID: linvor

Baud rate: 9600

Pairing code / password: 1234

No parity check

Testing communication with the module:

Open the Arduino Serial monitor. make sure you have selected the correct port and Baud rate of 9600

You should see the text: Type AT commands! . If not, something is wrong and you need to re-check your set-up

Type AT in the Arduino IDE Serial monitor input field and press the Send button. You should see the response: OK. Now you are ready to change the module's settings!

Type AT+VERSION without spaces in the Arduino IDE Serial monitor input field and press the Send button. You should get a response that will have the module name and version, like: OKlinvorV1.8

One of the more useful attributes is the Bluetooth baud rate. You can set that by sending a command like AT+BAUD4, where the last number (4 in this case) is defining the rate as follows:

AT+BAUD1———1200

AT+BAUD2———2400

AT+BAUD3———4800

AT+BAUD4———9600 (Default)

AT+BAUD5———19200

AT+BAUD6———38400

AT+BAUD7———57600

AT+BAUD8———115200

AT+BAUD9———230400

AT+BAUDA———460800

AT+BAUDB———921600

AT+BAUDC———1382400

If you enter AT+BAUD4 you should receive a response OK9600. Do not set a rate above 115200, as you will not be able to communicate with the module through your Arduino at that speed.

Refer to section 9 of the module's datasheet for all other available configuration options. In the video below I change the Baud rate of the Bluetooth module from 9600 to 115200.

Modulo Datasheet Microcontrolador ATMEGA