

# TABLERO DIGITAL POR PLATAFORMA ARDUINO

Obtención de Datos en un Tablero Digital para un Vehículo Convencional a través de la  
Plataforma de Arduino.

Francisco Miguel Calderón Pérez

Universidad Internacional SEK

Nota de Autor

Francisco Miguel Calderón Pérez, Facultad de Ingeniería Mecánica,

Universidad Internacional SEK;

Cualquier correspondencia concerniente a este trabajo puede dirigirse a:

[franco290489@hotmail.com](mailto:franco290489@hotmail.com)

## Índice de Contenidos

Introducción .....	1
Justificación .....	3
Objetivos .....	5
Objetivo general .....	5
Objetivos específicos.....	5
Marco Teórico .....	6
Método .....	12
Resultados .....	39
Conclusiones .....	42
Recomendaciones .....	43
Anexos .....	44
Referencias.....	60

## Índice de Figuras

Figura 1: Componentes Básicos Arduino .....	8
Figura 2. Tablero Convencional Vehículo.....	10
Figura 3. Microcontrolador Arduino Mega.....	12
Figura 4. Diagrama funcionamiento Arduino Envío y Recepción de Datos.....	13
Figura 5. Rectificador de Señal.....	14
Figura 5.1. Gráfico explicativo de Rectificador de Señal .....	14
Figura 6. Puente H .....	17
Figura 6. 1. Gráfico explicativo de Puente H.....	17
Figura 7. Tarjeta de Relay .....	19
Figura 7. 1. Gráfico explicativo de Tarjeta de Relay .....	20
Figura 8. Encoder de Pulsos Ópticos .....	21
Figura 8. 1. Gráfico explicativo de Encoder de Pulsos Ópticos .....	21
Figura 9. Sensor de pulsos Magnético para el velocímetro .....	23
Figura 10. Placa Arduino shield Bluetooth.....	24

## TABLERO DIGITAL POR PLATAFORMA ARDUINO

Figura 11. Método Programación App Inventor .....	27
Figura 12. Esquema de conexión Arduino y Bluetooth .....	28
Figura 13. Shield Bluetooth Arduino .....	29
Figura 14. Creación de la pantalla AppInventor .....	29
Figura 15. Barra de componentes para la pantalla .....	30
Figura 16. Montaje de la Conexión Shield Bluetooth.....	35
Figura 17. Simulación de conexión del sistema.....	38
Figura 18. Aplicación Desde el Smartphone.....	40
Figura 19. Pantalla aplicación Velocímetro Android .....	41

**Índice de Tablas**

Tabla 1. Elección de Microcontrolador para Controlar el Sistema. ....	12
Tabla 2. Descripción de Componentes para el Rectificador de Señal. ....	15
Tabla 3. Descripción de Componentes para el Puente H. ....	18
Tabla 4. Combinaciones para Avance y Retroceso del giro del motor eléctrico.....	19
Tabla 5. Descripción de Componentes para la Tarjeta de Relay. ....	20
Tabla 8. Descripción de Componentes para el Encoder de Pulsos Ópticos.....	22

## TABLERO DIGITAL POR PLATAFORMA ARDUINO

**Índice de Diagramas**

DFD 1. Diagrama de Flujo de la Programación del Arduino .....	25
---	----

**Índice de Anexos**

Programación del Velocímetro Digital.....	44
Programación del Tacómetro (Numero de Revoluciones RPM) .....	46
Controlador Maestro (Arduino Mega) .....	48
Programación Aplicación Android (app inventor).....	52
Resultados de encuestas realizadas: .....	55



**Declaración Juramentada**

Yo, FRANCISCO MIGUEL CALDERON PEREZ, con cédula de identidad 171428526-7, declaro bajo juramento que el trabajo aquí desarrollado es de mi autoría, que no ha sido previamente presentado para ningún grado o calificación profesional; y, que se ha consultado las referencias bibliográficas que se incluyen en este documento. A través de la presente declaración, cedo mis derechos de propiedad intelectual correspondientes a este trabajo, a la UNIVERSIDAD INTERNACIONAL SEK, según lo establecido por la Ley de Propiedad Intelectual, por su reglamento y por la normativa institucional vigente.

FRANCISCO MIGUEL CALDERON PEREZ

C.I.: 1714285267

# TABLERO DIGITAL POR PLATAFORMA ARDUINO

## **Resumen**

Como resultado de este trabajo de fin de carrera se obtuvo datos de un tablero digital para un vehículo convencional a través de la plataforma de Arduino por medio de circuitos electrónicos. Se lo realiza recibiendo las señales de un vehículo antiguo los cuales van a ser procesadas en el microcontrolador Arduino y mostrar los valores obtenidos en una aplicación que se diseñó en el programa App inventor para la plataforma Android.

## **Abstract**

As a result of this Final Career Project it's getting data into a digital board to a conventional vehicle through the Arduino platform. It is performed Getting data from a old vehicle which will be processed in the microcontroller Arduino to finally show the values obtained in an application which is created in the android platform.

## **Introducción**

La electrónica en el mundo actual, ha logrado adelantos tecnológicos para facilitar la vida a las personas interviniendo en muchos ámbitos. Uno de los sectores que se ha enfocado es el área automotriz, por ende se tiene mejoras en sus sistemas electrónicos para mejorar el desempeño del vehículo brindando ayudas al manejo y al confort de los pasajeros y brindando una mejora en la experiencia de funcionamiento del mismo y con un amplio progreso en el futuro.

Las circunstancias con las que nos encontramos hoy en el campo de los microcontroladores tienen sus raíces en el desarrollo de la tecnología de los circuitos integrados. Este desarrollo ha hecho posible contener cientos de miles de transistores en un solo chip. Ése era uno de los requisitos previos para la producción de los microprocesadores. Así es cómo se desarrolló el primer chip que contenía una microcomputadora, o lo que después se llegaría a conocer como un microcontrolador. (UASLP", 2013)

En el año 1969, un equipo de ingenieros japoneses de la compañía BUSICOM llegó a Estados Unidos con una idea, ellos deseaban usar para sus proyectos pocos circuitos integrados. Esta solución presumía que la función del circuito integrado se determinaría por un programa almacenado en él. Eso significaba que la configuración sería más simple, pero que requeriría mucho más memoria de lo que requería el proyecto (UASLP", 2013)

Arduino fue inventado en el año 2005 por el entonces estudiante del instituto IVRAE Massimo Banzi, quien, en un principio, pensaba en hacer Arduino por una necesidad de aprendizaje para los estudiantes de computación y electrónica del mismo instituto, ya que en ese entonces, adquirir una placa de micro controladores eran bastante caro y no ofrecían el soporte

adecuado; no obstante, nunca se imaginó que esta herramienta se llegaría a convertir en años más adelante en el líder mundial de tecnologías DIY (Do It Yourself). (Arduinodhtics, 2008)

Tiempo después, se integro al "Team Arduino" el estudiante español David Cuartielles, experto en circuitos y computadoras, quien ayudó Banzi a mejorar la interfaz de hardware de esta placa, agregando los micro controladores necesarios para brindar soporte y memoria al lenguaje de programación para manipular esta plataforma. Para la producción en serie de la primera versi[on se tomaron en cuenta algunas consideraciones: Economía (no > a 30 Euros), debía ser Plug and Play, utilizaron el color azul para marcar una diferencia con las placas convencionales, trabajar en todas las plataformas Mac, Windows y Linux. (Arduinodhtics, 2008)

Se desarrollara un sistema electrónico a través de un microcontrolador para implementar un tablero digital obteniendo datos como la velocidad, numero de revoluciones del motor, sensores y actuadores a automóviles de más de 30 años de antigüedad que se consideran como Clásicos, Con este microcontrolador se obtiene un monitoreo de datos importantes al momento de obtención de los datos del estado del funcionamiento del vehículo. Existen dispositivos electrónicos en los automóviles que permiten receptar datos de velocidad del automóvil, nivel de temperatura, medidor de combustible, de revoluciones, voltaje de batería, kilometraje, etc., Sin embargo estos medidores no han sido implementados todos en una sola aplicación pero funcionan a través de una conexión a internet y GPS lo cual dificulta la accesibilidad a los usuarios y el monitoreo no es exacto.

Para el desarrollo del tablero se creara una aplicación de la plataforma de software libre Android para obtener una visualización de los medidores que se necesiten implementar con los datos del tablero qu vas a ser procesados a través del microcontrolador para monitoreo del funcionamiento y muestra en pantalla.

**Justificación**

En el Ecuador existen grupos de personas que tienen afición por los vehículos clásicos y estos no cuentan con sistemas tecnológicos de monitoreo como el tablero de instrumentos debido a que sus señales son analógicas y en muchos de los casos ya no tienen funcionamiento.

Se realizó una investigación a personas que cuentan con vehículos clásicos a través de una encuesta online y se obtuvo como datos finales de la muestra de 44 elementos entre los cuales indicaron que 32.4 % no tiene funcionamiento su tablero de instrumentos el 70.3 % estaría dispuesto a implementar el sistema de monitorio del tablero digital, y que 84.1 % disponen de un dispositivo android para su funcionamiento. Con estos valores se comprende que los vehículos tienen fallas en su funcionamiento y al implementar este proyecto van a tener un correcto funcionamiento y será de gran ayuda por llegar a mejorar uno de los sistemas de monitoreo más importantes, obteniendo datos reales esta implementación proporcionará un adecuado monitoreo y una mejora de equipamiento del vehículo.



## **Objetivos**

### **Objetivo general**

Construir e implementar un tablero de instrumentos digital a través de la programación de un microcontrolador y visualización de los datos a través de un Smartphone o Tablet para vehículos de varios años de antigüedad para los dueños que quieren adaptar un sistema más moderno para monitoreo a tiempo real.

### **Objetivos específicos**

- Encontrar los tipos señales de los sensores y medidores que presentan los diferentes vehículos.
- Construir un tablero didáctico para la simulación del tablero digital para el envío y recepción de datos desde el microcontrolador.
- Crear el diseño del hardware del microcontrolador y la aplicación android para el control y visualización de los valores a obtener.



## Marco Teórico

En los últimos años, la tecnología ha mejorado facilitando en muchos ámbitos al trabajo de las personas permitiendo de una manera rápida y confiable para obtener datos precisos y de manera continua a través de placas electrónicas que han sido programadas para realizar tareas específicas, para el proyecto que se va a construir vamos a crear un sistema de monitoreo de datos relevantes de un automóvil.

Para la implementación del sistema a diseñar es necesario el uso de un microcontrolador que permita procesar datos necesarios que van a ser programados que cumplan con las necesidades del proyecto. Valdés y Pallas lo describen: “Un microcontrolador es un circuito integrado que permite ser programado según se lo requiera y que es capaz de ejecutar las órdenes grabadas en su memoria según se lo establezca en su programación” (Fernando E. Valdés Pérez, 2007, pág. 11) Además un microcontrolador combina los recursos fundamentales disponibles en un microcomputador, es decir, la unidad central de procesamiento (CPU), la memoria y los recursos de entrada y salida, en un único circuito integrado (Itphonenet, 2013). Este proyecto se desarrolla para obtener datos de un vehículo en un tablero digital con el fin de visualizar valores y datos referenciales de manera actualizada, ya que los automotores de años anteriores a los 70 no cuentan con un sistema que permita visualizar estos datos.

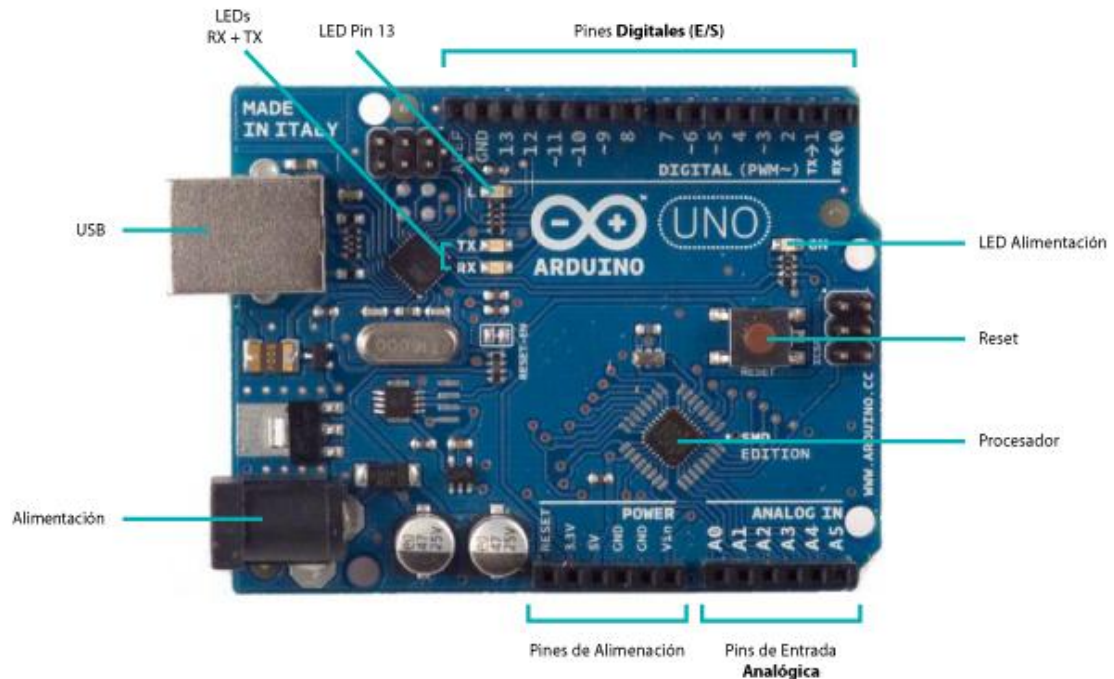
La totalidad de los indicadores y luces de testigos suelen agruparse formando, en conjunto, el cuadro o tablero de instrumentos (López, 2011, pág. 360).

Cada uno de los elementos que lo integran tiene encomendada una función determinada y dispone de su propio circuito eléctrico lo que supone una gran cantidad de conexiones y cableado entre los cuales están velocímetro, cuenta revoluciones, indicador de nivel de aceite, indicador de nivel de gasolina, indicador de temperatura del motor etc. (Alonso, 2006).

En el tablero de instrumentos se encuentran todos estos indicadores y estos datos van a ser obtenidos por el microcontrolador para procesarlos con el uso de Arduino, que es una plataforma de software libre que usa una placa con un microcontrolador y un entorno de desarrollo para su programación, por ello, facilita el uso de la electrónica en proyectos basándose en la programación que se diseñará para esta cada necesidad. Se puede decir que es de una de las plataformas más complejas y variables que pueden existir.

En la página web oficial de Arduino se describe al producto como una plataforma de electrónica abierta (Open Source) para la creación de prototipos basada en software y hardware flexibles y fáciles de usar. Estos módulos se crearon para artistas, diseñadores, aficionados y cualquier persona que esté interesado en crear entornos u objetos interactivos. (Arduino, Arduino Web Oficial, 2006). Se Puede usar al Arduino para crear objetos interactivos, leyendo datos de una gran variedad de interruptores y sensores y controlar multitud de tipos de luces, motores y otros actuadores físicos. Los proyectos de Arduino pueden ser autónomos o comunicarse con un programa (software) que se ejecute en tu ordenador (ej. Flash, Processing, MaxMSP).

La placa se la puede montar uno mismo o comprarla ya lista para usar, y el software de desarrollo es abierto y se lo puede descargar gratis. (Arduino M., 2014)



**Figura 1: Componentes Básicos Arduino**

(Arduino M., 2014)

El microcontrolador se programa para procesar datos dependiendo de las necesidades del proyecto en el que se desea usar la placa. Existe gran variedad de fabricantes y versiones disponibles. El funcionamiento de la placa microcontrolador de Arduino se basa en una interfaz de entrada, que puede estar directamente unida a los “periféricos” que es una denominación para designar al aparato o dispositivo auxiliar conectado a la unidad central de procesamiento de la computadora.

También se consideran periféricos a las unidades o dispositivos de Hardware a través de los cuales la computadora se comunica con el exterior, igual a los sistemas que almacenan o archivan la información, sirviendo de memoria auxiliar de la memoria principal), o conectarse a ellos por puertos. (Gomez, 2007) de la misma manera cuenta con una interfaz de salida, que es la que encargada de llevar la información procesada a los periféricos que se encargan de mostrar los datos finales , por ejemplo, una pantalla que muestra visualmente o un altavoz encargado a través de sonidos mostrar lo que se quiere transmitir.

En todo vehículo resulta necesario el panel de instrumentos, la presencia de ciertos indicadores o señales de control en el tablero, para que permitan al conductor mantener visibles los datos del vehículo y cumplir con los reglamentos de seguridad vigentes. El panel de instrumentos cuenta con varios indicadores y testigos entre los cuales comprende: el indicador de velocidad del vehículo, el tacómetro o cuenta revoluciones, medidores de temperatura del motor, indicador de combustible que son datos que se encuentran en forma de señales analógicas o digitales o una combinación de ambos. Además existe una serie de testigos luminosos de simbología normalizada, como por ejemplo el testigo de presión de aceite, de carga de la batería, de indicadores de fallos intermitente, entre otros. Aunque varían el modo de operar y la cantidad de estos indicadores de un vehículo a otro en general se pueden clasificarse en cuatro grupos:

- Instrumentos para el control de los índices de funcionamiento técnico del coche.
- Instrumentos para indicar los índices de circulación vial.
- Señales de alarma.
- Señales de alerta.

En esta clasificación para nuestro proyecto solo se tendrá en cuenta las dos primeras ya que contamos con indicadores y actuadores de funcionamiento técnico del coche como las luces y bocina, también de Instrumentos para indicar los índices de circulación vial como la velocidad los otros dos grupos no se los tomo en cuenta debido a que las señales de alerta y alarma nos indican un mal funcionamiento , posibles desgastes o niveles bajos de líquidos y se tiene que implementar mas sensores y aumenta el valor del proyecto.

Los Indicadores y Testigos muestran varios datos y señales del auto tales como la velocidad a la que está rodando el automóvil "Velocímetro" el numero de vueltas de giro del motor " Tacómetro" la distancia de kilómetros recorridos, cantidad de combustible en el tanque, temperatura del liquido refrigerante del motor etc. También cuenta con formas luminosas que demuestran una falla en motor o el vehículo capaces de llamar la atención del conductor, para evitar problemas de seguridad de manejo o mecánicos graves: por ejemplo el testigo de Reserva de combustible, presión de aceite, carga de la batería, temperatura excesiva del refrigerante, nivel de líquido de frenos, calentadores en los motores diesel, fallos del motor etc.



**Figura 2. Tablero Convencional Vehículo**  
(Automotriz, 2012)

Para la creación de la aplicación en la plataforma android existen varios programas entre los cuales Appmachine, AppInventor, android studio, java y muchos más. App Inventor es una herramienta útil de programación, con la ventaja adicional de permitir el desarrollo de aplicaciones para dispositivos móviles que usen el sistema operativo Android. (David Wolber, 2011). Una característica interesante es que el desarrollo de la aplicación es en Web y los proyectos se guardan en línea. El App Inventor consta de dos segmentos principales: un módulo Web y el editor de bloques de Android. El módulo en Web que se mencionaba, donde aparte de ser el punto de entrada tenemos acceso a nuestros proyectos y, una vez abierto un proyecto, podemos entrar a la sección de diseño de nuestra aplicación. Esta sección es donde podemos añadir los componentes y configurarlos apropiadamente. Si se trata de componentes visuales, entonces definimos también el diseño de la interfaz. El segmento del editor de bloques que es ahí es donde los bloques se conectan como piezas de Lego, formando la lógica de la aplicación. Para los familiarizados con desarrollo de aplicaciones mediante componentes visuales verán que es un concepto bastante similar. (APPIinventor, 2011).

Con estas herramientas es el correcto desarrollo del proyecto con sus datos recibidos de los sensores y actuadores del vehículo que van a ser procesados a través de la programación del microcontrolador y mostrados en pantalla por la aplicación creada para arduino en la plataforma App inventor.

Método

Escogimiento del sistema

Características/Importancia		Arduino Mega			Arduino Uno			Arduino Leonardo		
		Valor	Puntaje	Total	Valor	Puntaje	Total	Valor	Puntaje	Total
Entradas y Salidas Digitales	30%	0.30	5	1.5	0.30	3	0.9	0.30	4	1.2
Pines Analogicos	15%	0.15	5	0.75	0.15	3	0.45	0.15	4	0.6
Multitarea	20%	0.20	4	0.8	0.20	3	0.6	0.20	3	0.6
Procesador	15%	0.15	4	0.6	0.15	3	0.45	0.15	3	0.45
Costos	20%	0.20	3	0.6	0.20	5	1	0.20	4	0.8
				4.25				3.4		
Puntaje										
5 Maximo - 1 minimo										

Tabla 1. Elección de Microcontrolador para Controlar el Sistema.

El desarrollo del proyecto para el diseño del tablero digital consta de una placa microcontrolador en este caso de Arduino Mega, Se escogió este microcontrolador como placa principal por mejores prestaciones que otras placas asi se procesa mejor las conexiones a los testigos medidores y actuadores que se requieran implementar en la pantalla de visualización.

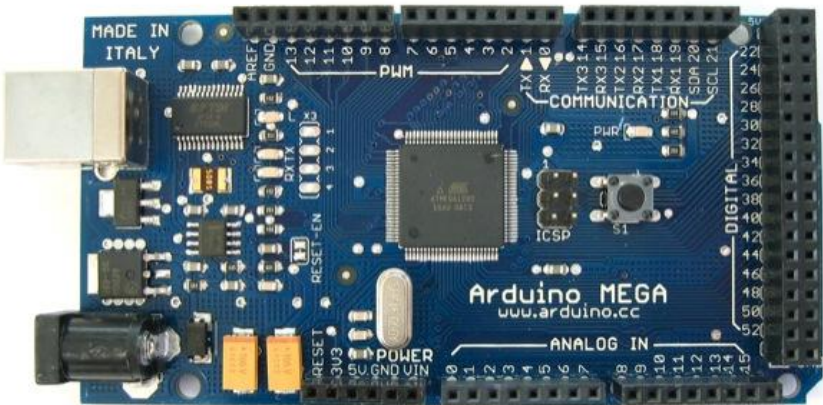
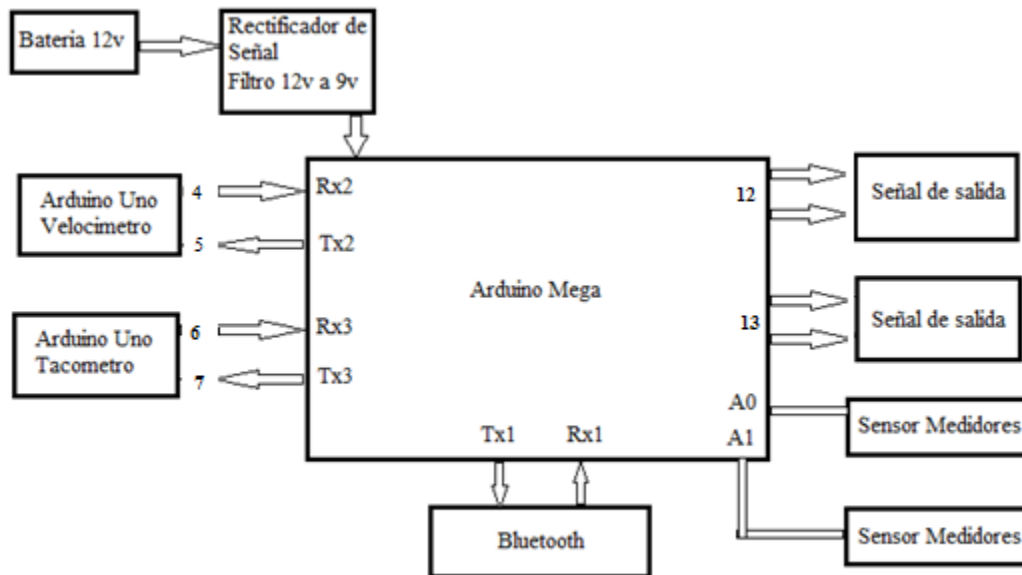


Figura 3. Microcontrolador Arduino Mega  
(Arduino, 2015)

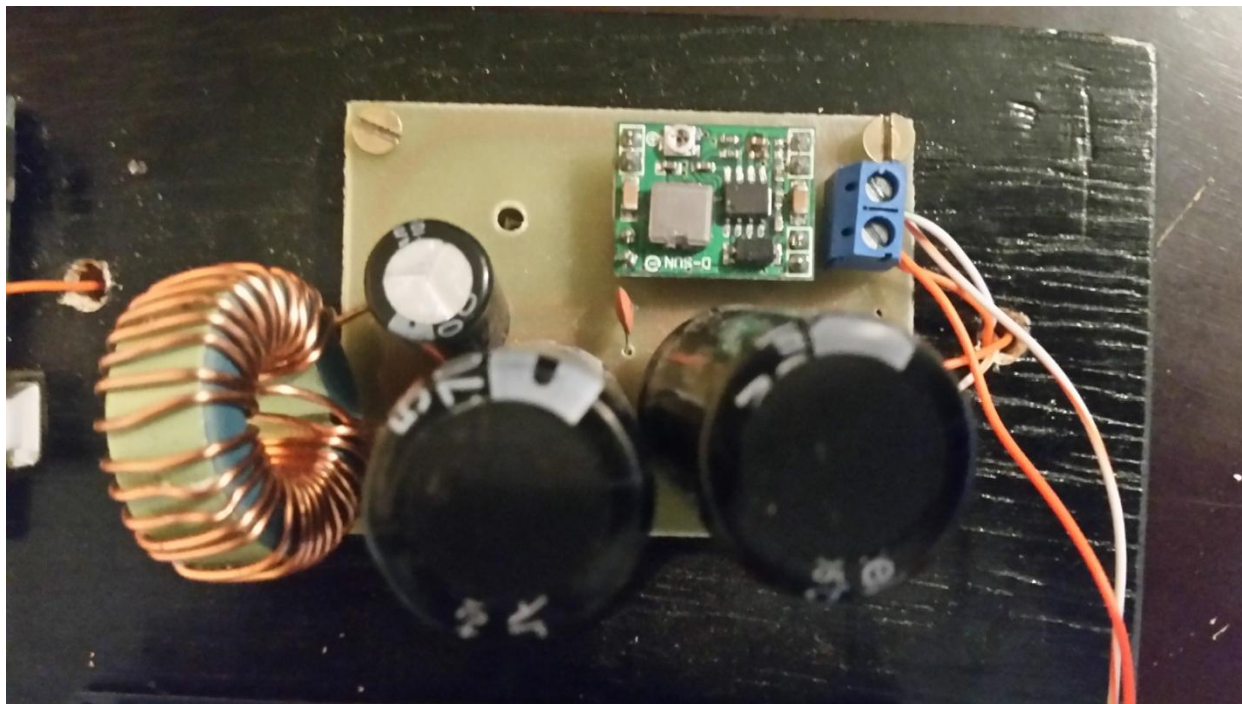
## Funcionamiento

El diagrama de funcionamiento de las placas del microcontrolador se realiza desde la fuente de energía de 12V que entrega la batería del automóvil de prueba para lo que se incorpora un rectificador de señal y un regulador de voltaje para entrada al Arduino.

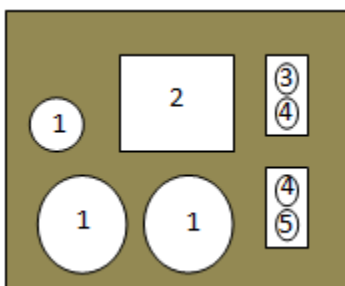


**Figura 4. Diagrama funcionamiento Arduino Envío y Recepción de Datos**



**Rectificador de Señal Filtro 12v a 9v****Figura 5. Rectificador de Señal**

Mediante el siguiente gráfico y tabla se indica cómo se conecta el rectificador de señal:

**Figura 5.1. Gráfico explicativo de Rectificador de Señal**

ID.	Nombre	Descripción
1	Capacitores	Filtro tipo $\pi$ . Elimina el ruido eléctrico del vehículo
2	Regulador Integral Variable	Switching de 36V conmutado. Regula la salida del voltaje. Trabaja de 0 a 2V menos que la fuente y mantiene el voltaje estable.
3	Entrada de Voltaje	12V desde la batería del vehículo.
4	Tierra	Conexión a masa.
5	Salida de Voltaje	9V regulados o señal.

**Tabla 2. Descripción de Componentes para el Rectificador de Señal.**

Se requería rectificar la onda de voltaje, mantenerla estable y eliminar el ruido eléctrico, por tal razón se seleccionó esta fuente ya que regula la entrada de voltaje a la tarjeta debido a las recomendaciones del fabricante que muestra que las placas de Arduino trabajan a 5V óptimamente y tienen una entrada de voltaje recomendada de 7V a 12V teniendo límites máximos de 20V pero con este voltaje que no aseguran un funcionamiento óptimo ya que la batería del vehículo nos entrega 12V se necesita un voltaje estable de máximo 9V. Además elimina el ruido eléctrico causado por el alternador generado en el rotor entre cada polo.

La fuente que se implementó consta de los siguientes materiales:

- Capacitores electrolíticos de 4700 micro faradios (mF.) a 50V.
- Bobina Toroidal.
- Capacitores cerámicos de 220 nano faradios a 50 voltios.
- Regulador de salida de voltaje..

Se implemento el capacitor eléctrico de 4700 micro faradios (mF.) a 50V ya que el voltaje que nos entrega el auto de Corriente continua varia hasta llegar 15V y el fabricante nos indica que el primer capacitor de 470(mF.) debe ser menor al segundo con esto se sobredimensiona la capacidad para evitar daños en el sistema.

La Bobina toroidal se la escogió por que al ser una bobina se crea un campo y atravez del cable enrollado logramos suprimir ruidos en el caso del automóvil causado por algunos elementos como el alternador, en el diámetro interior del bobinado se crea un campo magnetico para anularlo el cual es un filtro de paso bajo y mantiene mas plana la señal evitando el ruido eléctrico y los rizados máximos en la senal que siempre existen en la onda.

## **Puente H**

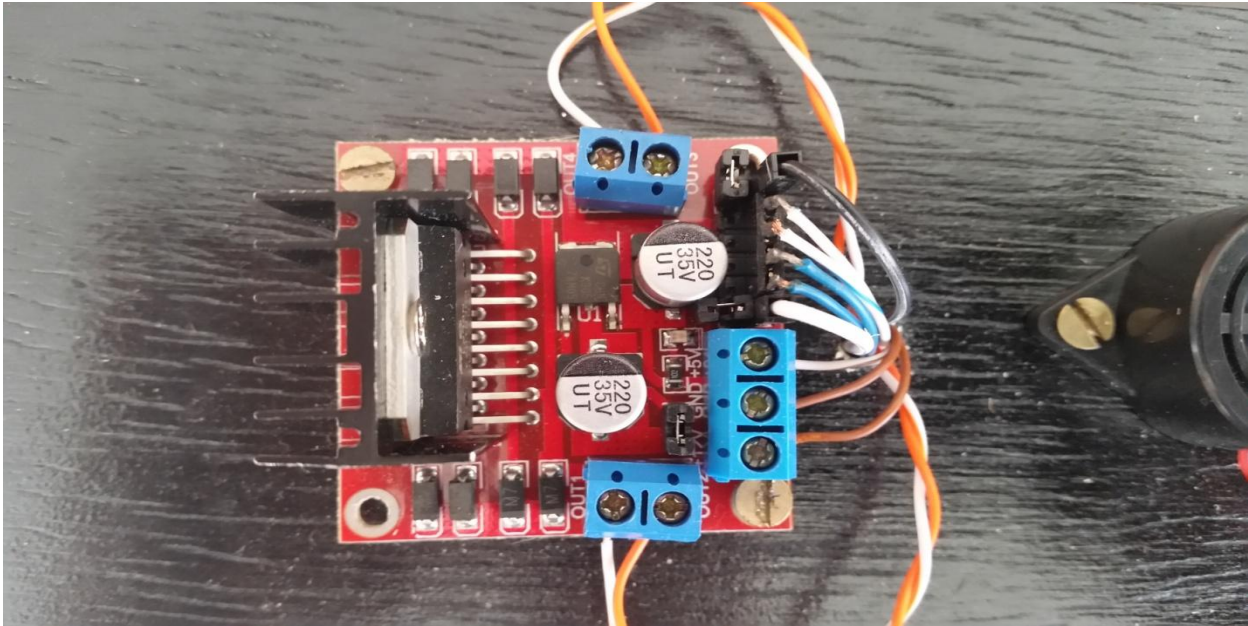


Figura 6. Puente H

Mediante el siguiente gráfico y tabla se indica cómo se conecta el puente H:

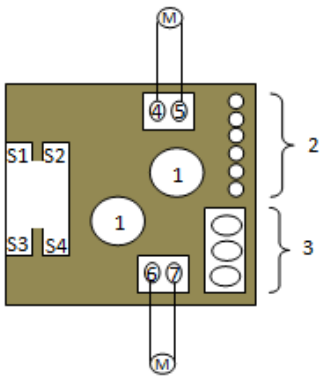


Figura 6. 1. Gráfico explicativo de Puente H

ID.	Nombre	Descripción
1	Capacitores	Almacenan energía eléctrica.

		Se detalla de arriba hacia abajo:
		2.1. PWM1
		2.2. IN4
2	Bornera	2.3. IN3
		2.4. IN2
		2.5. IN1
		2.6. PWM2
		Se detalla de arriba hacia abajo:
		3.1. 5V
3	Bornera	3.2. Tierra
		3.3. 12V
4	Out 1	Envía señal S1 a puente H
5	Out 2	Envía señal S2 a puente H
6	Out 3	Envía señal S3 a puente H
7	Out 4	Envía señal S4 a puente H

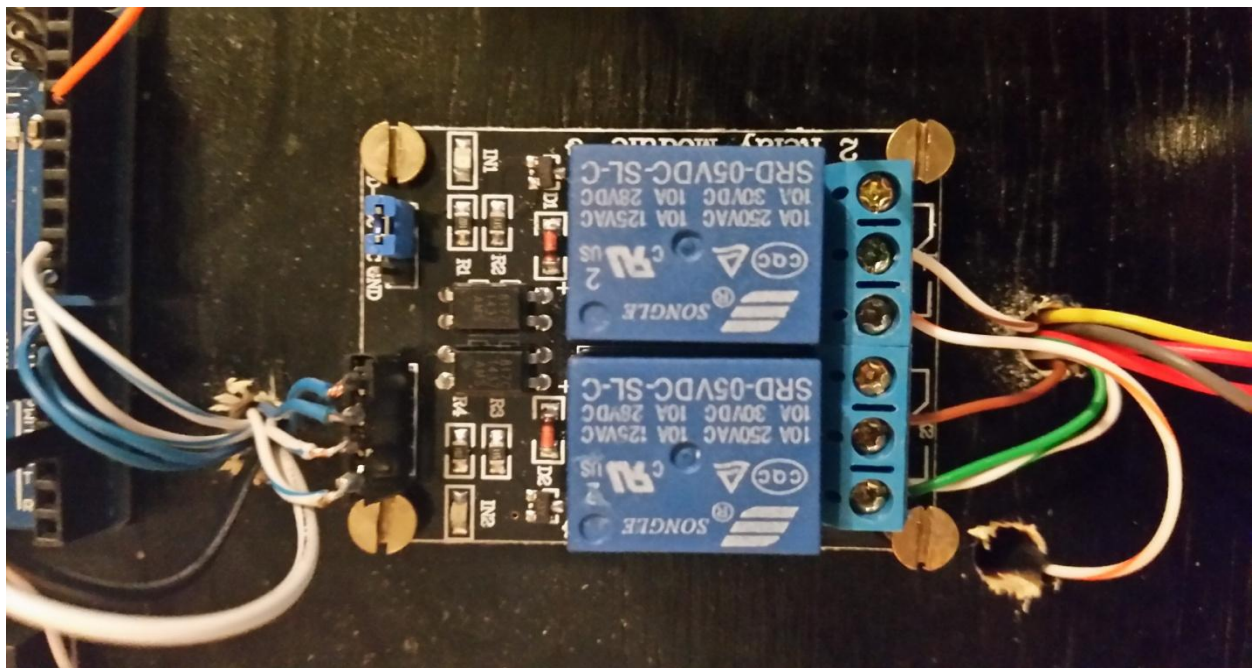
**Tabla 3. Descripción de Componentes para el Puente H.**

Para el Puente H se selecciono el integrado L298d que nos sirve para la alimentación digital de microcontroladores indicado según el fabricante y el numero de amperios que tiene a cada una de sus salidas , permite que los motores eléctricos usados en la maqueta giren en ambos sentidos, avance y retroceso. podíamos usar un ingrado l293 pero la salida de Amperios son muy bajas para la necesidad del proyecto, Tambien se utiliza como convertidor de potencia para variar la velocidad de giro mediante una variación de voltaje. El Puente H trabaja de la siguiente manera:

S1	S2	S3	S4	Resultado
1	0	0	1	Gira en avance
0	1	1	0	Gira en retroceso
0	0	0	0	El motor se detiene
1	0	1	0	El motor frena

**Tabla 4. Combinaciones para Avance y Retroceso del giro del motor eléctrico.**

### Tarjeta de Relay



**Figura 7. Tarjeta de Relay**

Mediante el siguiente gráfico y tabla se indica cómo se conecta la tarjeta de Relay:

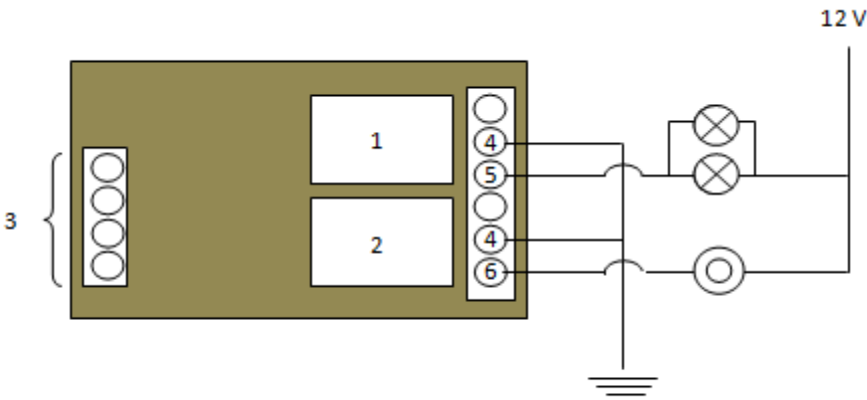


Figura 7. 1. Gráfico explicativo de Tarjeta de Relay

ID.	Nombre	Descripción
1	Relay 1	Controla la carga de las luces
2	Relay 2	Controla la carga del buzzer
		Se detalla de arriba hacia abajo:
		3.1. Tierra
3	Bornera	3.2. In1
		3.3. In2
		3.4. 5V
4	Tierra	Conexión a masa.
5	Switch Luces	Enciende o Apaga las luces
6	Switch Buzzer	Enciende o Apaga el buzzer

Tabla 5. Descripción de Componentes para la Tarjeta de Relay.

El módulo de relay para Arduino cuenta con salidas digitales para controlar las cargas más grandes de dispositivos electrónicos como las luces y/o el buzzer.

### Sensor de pulsos Ópticos



**Figura 8. Encoder de Pulsos Ópticos**

Mediante el siguiente gráfico y tabla se indica cómo se conecta el encoder de pulsos ópticos:



**Figura 8. 1. Gráfico explicativo de Encoder de Pulsos Ópticos**

ID.	Nombre	Descripción
-----	--------	-------------



---

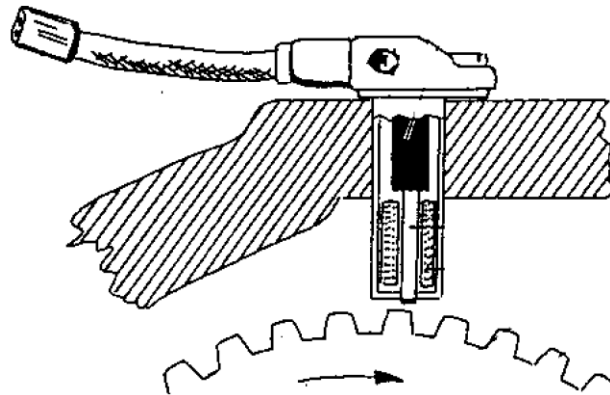
1	Emisor	Sensor opto interruptor
2	Receptor	Sensor opto interruptor
		Se detalla de arriba hacia abajo:
		3.1. 5V
3	Bornera	3.2. Tierra
		3.3. A0
		3.4. D0

---

**Tabla 8. Descripción de Componentes para el Encoder de Pulsos Ópticos.**

Este sensor se utiliza para medir principalmente la velocidad a la que gira una rueda por medio de un contador óptico entre el emisor y el receptor.

}Las señales que se procesan en el microcontrolador se transfieren de diferente manera, varias son tomadas desde el tablero original del automóvil que cuentan con señales digitales las cuales pueden ingresar para ser procesadas por el microcontrolador y las señales como el velocímetro que al ser una analógica con cadena de giro teniendo que implementar un sensor de pulsos magnéticos en el cual existe un emisor y un receptor a través de imanes en la rueda del vehículo el cual da una cantidad de pulsos por cada giro de los emisores en la rueda y con la distancia y una unidad de tiempo.



**Figura 9. Sensor de pulsos Magnético para el velocímetro**

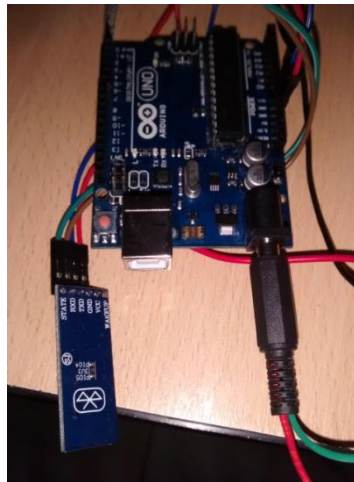
Para la medición del velocímetro y número de revoluciones se implementó un circuito eléctrico de dos placas construidas con las partes básicas de un de Arduino uno por cuestión de gastos los materiales para la construcción de la placa son los siguientes:

- Cristales de 16Mhz.
- Capacitores Cerámicos 22pf.
- Resistencias de 10 k y 1k.
- Led de encendido y funcionamiento.
- Microprocesadores atmega 128 o 368.

Estas placas extras de procesadores genéricos Atmega que tienen la función de trabajar como Arduinos conjuntamente para que los valores recepten y envíen los datos sin interrupciones y funcionen los dos a la vez ya que si se realiza la transferencia de datos sin estos trabajarían por turnos las señales y se perdería valores en el transcurso de cambio de cada uno.

Para los medidores en el tablero se realiza un acondicionamiento de voltaje de 0 a 5 voltios con lo cual los valores que se obtendrán como el medidor de combustible medidor de temperatura de batería etc., los valores se los va a interpretar en este rango hasta la visualización final. Para los actuadores el envío de datos desde el Arduino se envía una señal de voltaje controlador por un relé hacia los actuadores que se quieran implementar entre los cuales se tendrá las luces, bocina, arranque, bloqueo central etc.

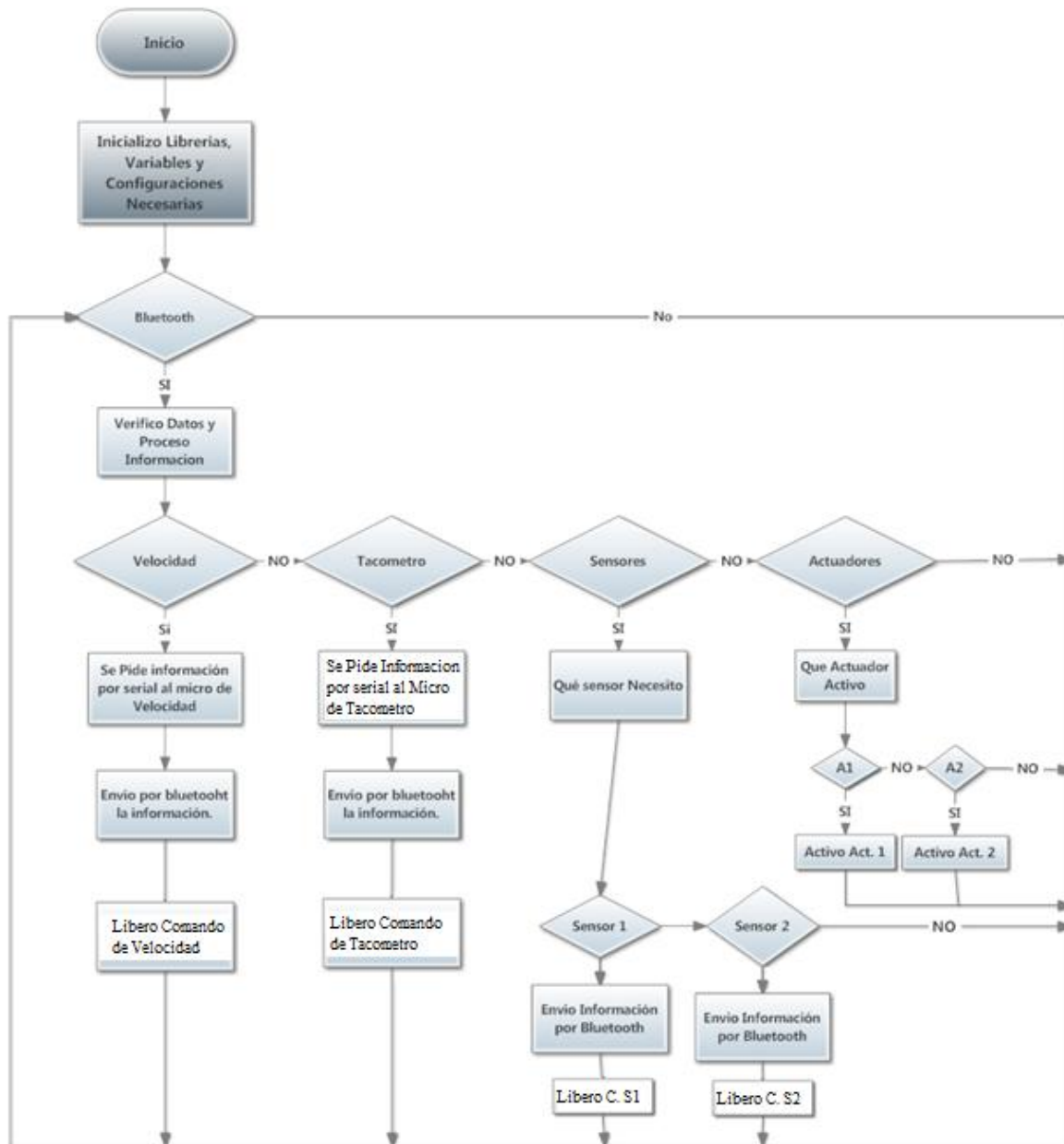
Estos valores visualizados en la aplicación Android tiene un tipo de comunicación con el Arduino por medio de un shield bluetooth que se comunica por puertos seriales que cuenta con envío y recepción de datos para una conexión inalámbrica para facilidad del usuario final.



**Figura 10. Placa Arduino shield Bluetooth**

## Programación

### Arduino



**DFD 1. Diagrama de Flujo de la Programación del Arduino**

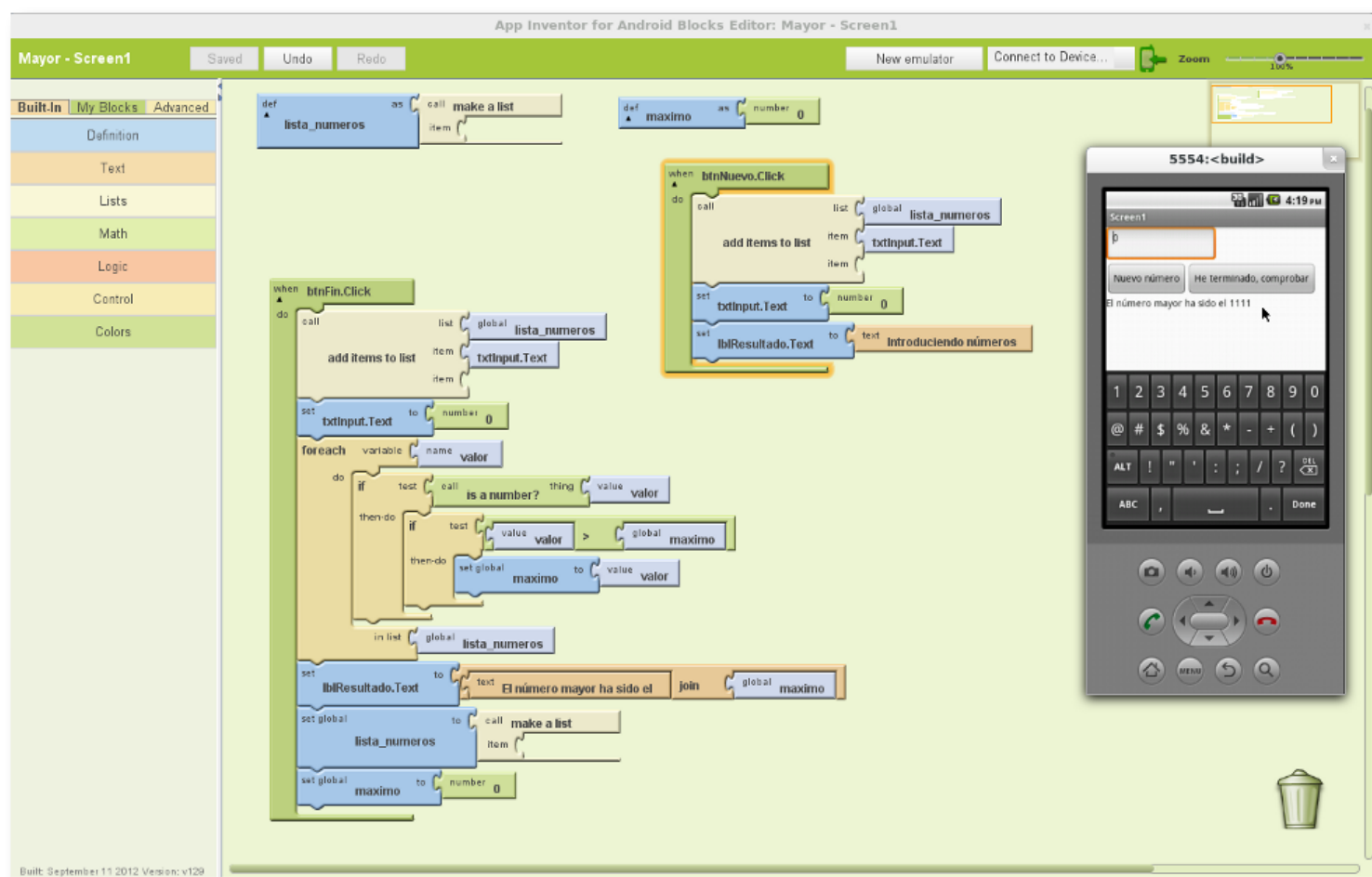
Para programar la placa es necesario descargarse de la página web de Arduino el entorno de desarrollo (IDE). Se dispone de versiones para Windows y para MAC. La estructura básica de programación de Arduino es bastante simple y divide la ejecución en dos partes: setup y loop. Setup() constituye la preparación del programa y loop() es la ejecución. En la función Setup() se incluye la declaración de variables y se trata de la primera función que se ejecuta en el programa. Esta función se ejecuta una única vez y es empleada para configurar el pinMode (p. ej. si un determinado pin digital es de entrada o salida) e inicializar la comunicación serie. La función loop() incluye el código a ser ejecutado continuamente (leyendo las entradas de la placa, salidas, etc.)

### **Aplicación**

La aplicación está programada por Google App Inventor que es una plataforma desarrollada por Google Labs la cual permite a los usuarios crear aplicaciones de software para el sistema operativo Android de una forma simple y rápida, de forma visual y a partir de un conjunto de herramientas básicas utilizando por un lado un editor gráfico para colocar botones, dibujos, textos, deslizadores, etc., El usuario se encarga de ir enlazando una serie de bloques de eventos para crear la aplicación. El sistema es gratuito y se puede descargar fácilmente de la web. Aunque muchas veces por su simplicidad se han dado casos que no es válida para crear aplicaciones de mucha complejidad esto se debe a que es una plataforma para crear aplicaciones para principiantes y no para desarrolladores profesionales con esto el nivel de aplicaciones creada son mucho mas sencillas y básicas.

Google App Inventor, cuenta con un incremento importante en el número de aplicaciones para Android debido a dos grandes factores: la simplicidad de uso, que facilita la aparición de un

gran número de estas diseñadas con fines específicos; y el Android Market, el centro de distribución de Android donde cualquier usuario puede distribuir sus creaciones libremente (AppInventor, 2015) lo que ayuda a diseñar de una manera sencilla la aplicación para el proyecto.

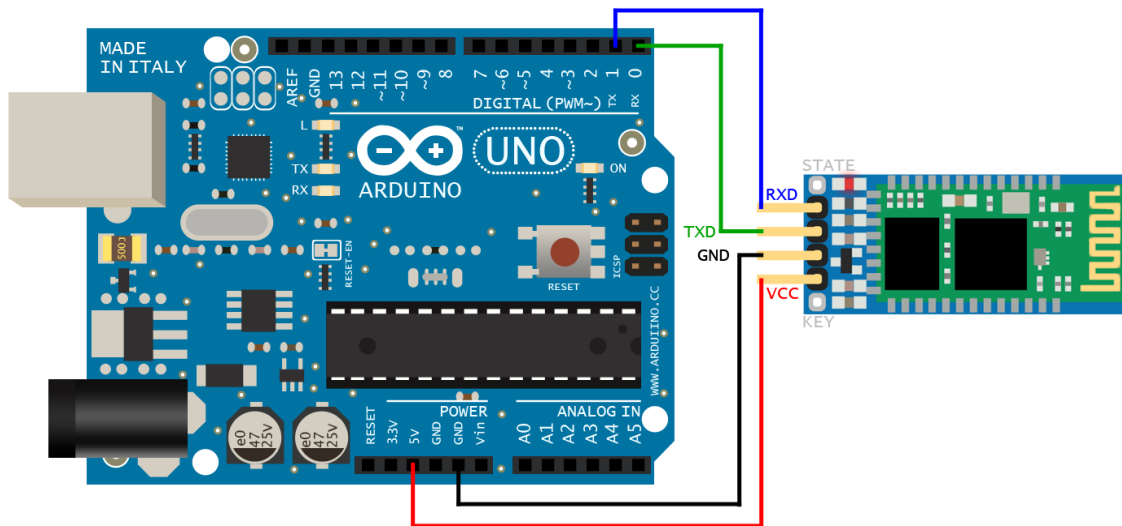


**Figura 11. Método Programación App Inventor**  
(España & Ministerio de Educacion, 2013)

En google App inventor se trabaja en varias secciones la una es la parte grafica y la otra es la ventana de desarrollo en la cual se puede implementar conexiones inalámbricas en su

entorno de programación en este caso utiliza una conexión a través de bluetooth con esto se evita conexiones utilizando cables por medio de un shield bluetooth conectado al Arduino maestro que envía y recibe los datos procesados del microcontrolador

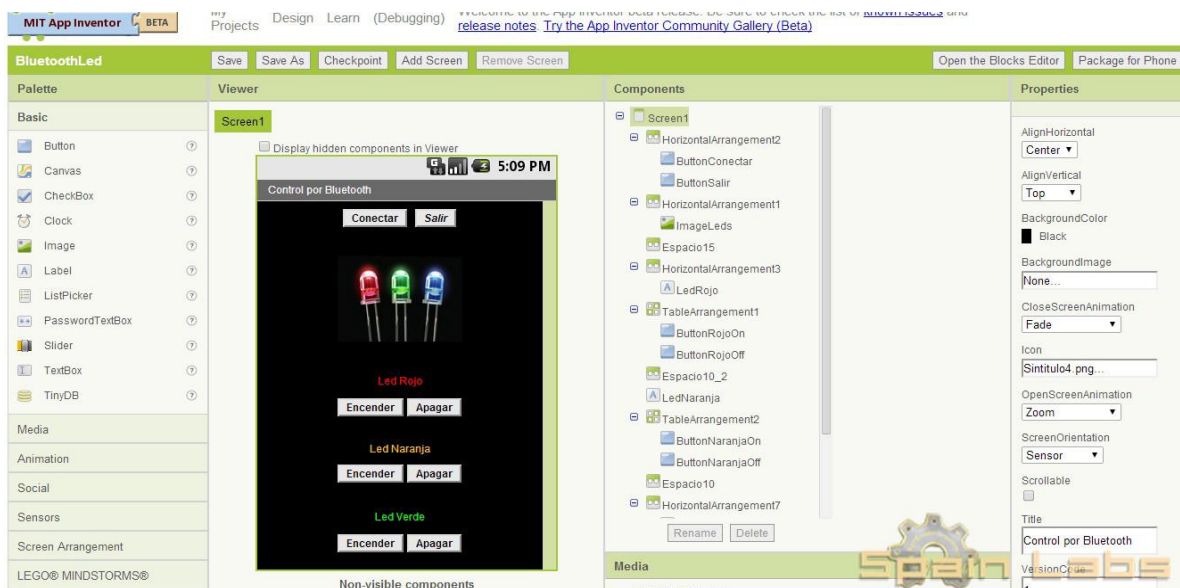
La programación se realiza en el editor de bloques se tiene un ejemplo guía para entender el método de la creación y la conexión bluetooth así como trabajo en conjunto con el microcontrolador Arduino:



**Figura 12. Esquema de conexión Arduino y Bluetooth**



**Figura 13. Shield Bluetooth Arduino**  
(Armdudroid, 2015)



**Figura 14. Creación de la pantalla AppInventor**  
(España & Ministerio de Educación, 2013)



La creación cuenta con una barra de componentes de formas, imágenes y entradas de texto para cubrir las necesidades que requiere en este caso solo encender una serie de leds por separado.



**Figura 15. Barra de componentes para la pantalla**

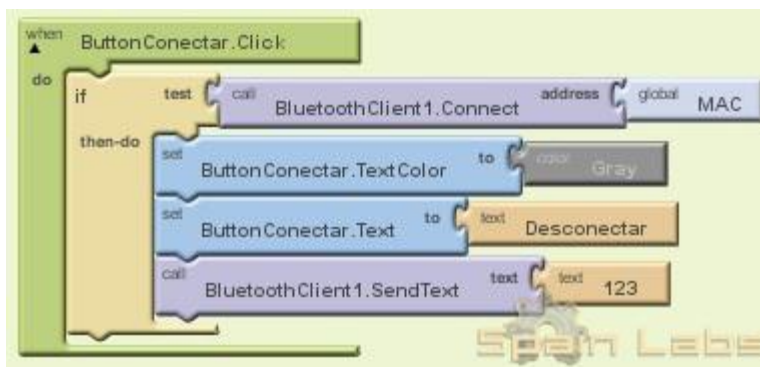
(España & Ministerio de Educacion, 2013)

Se empieza declarando la variable "MAC", asignándole la dirección de nuestro módulo bluetooth (se puede saber la dirección por ejemplo con la app Bluetooth Terminal - Link).

En el campo de texto donde pone "DIRECCION MAC", se tendrá que poner la dirección tal cual aparece en el móvil, algo así: XX: XX: XX: XX: XX: XX



Se sigue ahora con el botón para conectar con el módulo desde la aplicación se realiza una comprobación de la MAC del módulo para ver si está funcionando.

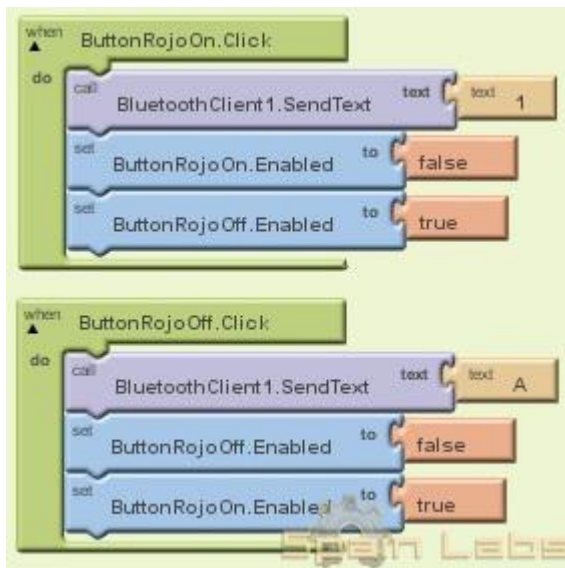


Primero, al pulsar el botón de encender "on", se tiene que:

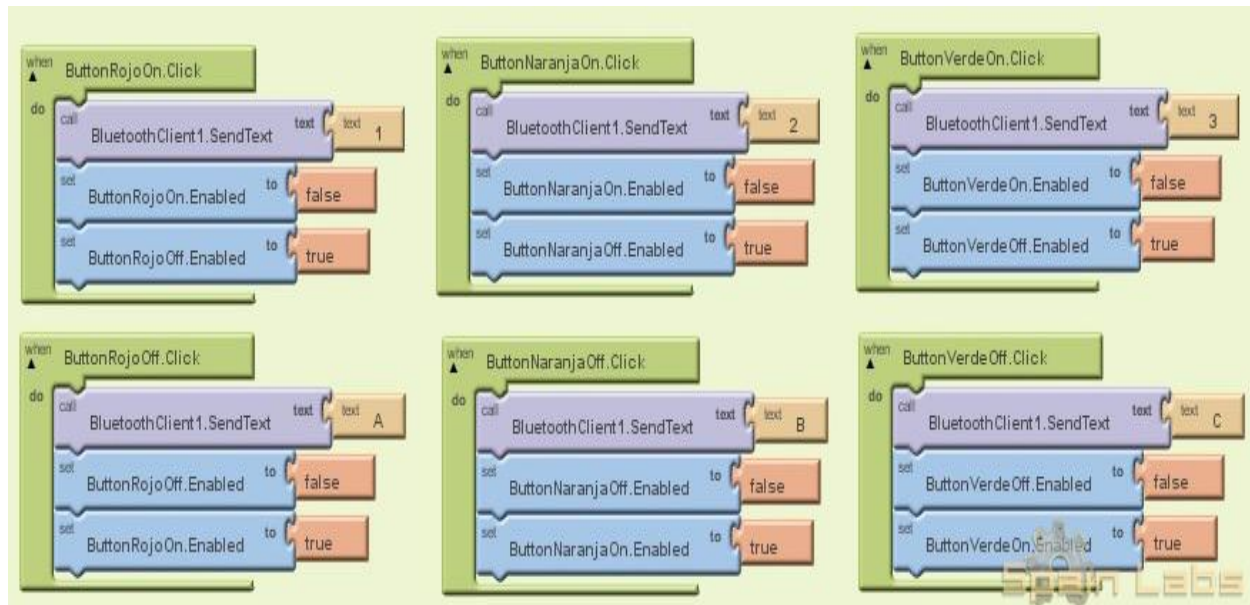
- Se envía un "1", que se usará para encender en este caso el led rojo con el Arduino.
- Desactivamos el botón de encender para este led.
- Activamos el botón de apagado (previamente en el diseño lo marcamos para que esté desactivado en el inicio).

Segundo, una vez desactivado el botón de encender y activado el de apagar "off", si pulsamos este último:

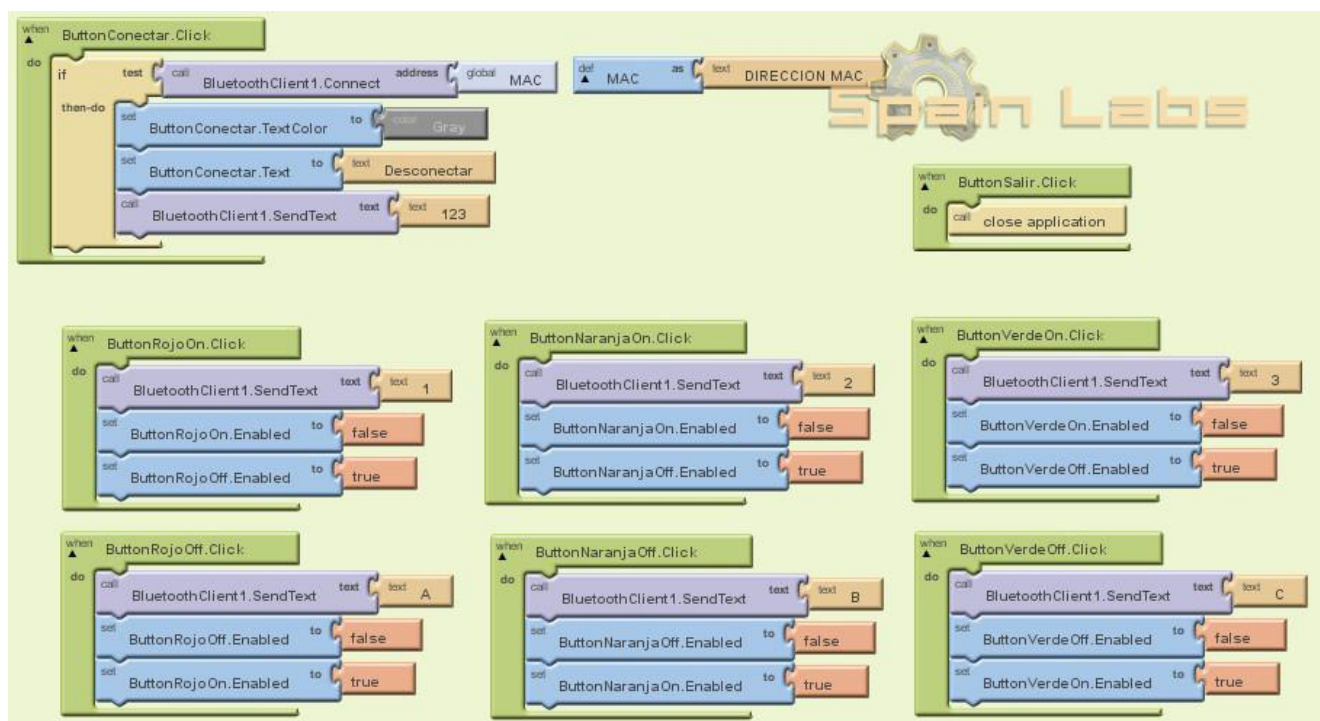
- Se envía una "A", que usaremos para apagar en este caso el led rojo con el arduino.
- Desactivamos el botón de apagar para este led.
- Activamos nuevamente el botón de encendido.



Las cadenas de caracteres que se envían al Arduino, pueden ser cualquier texto, solo hay que poner luego lo mismo en la parte del código donde corresponda. De esta forma quedaría para los 3 leds:



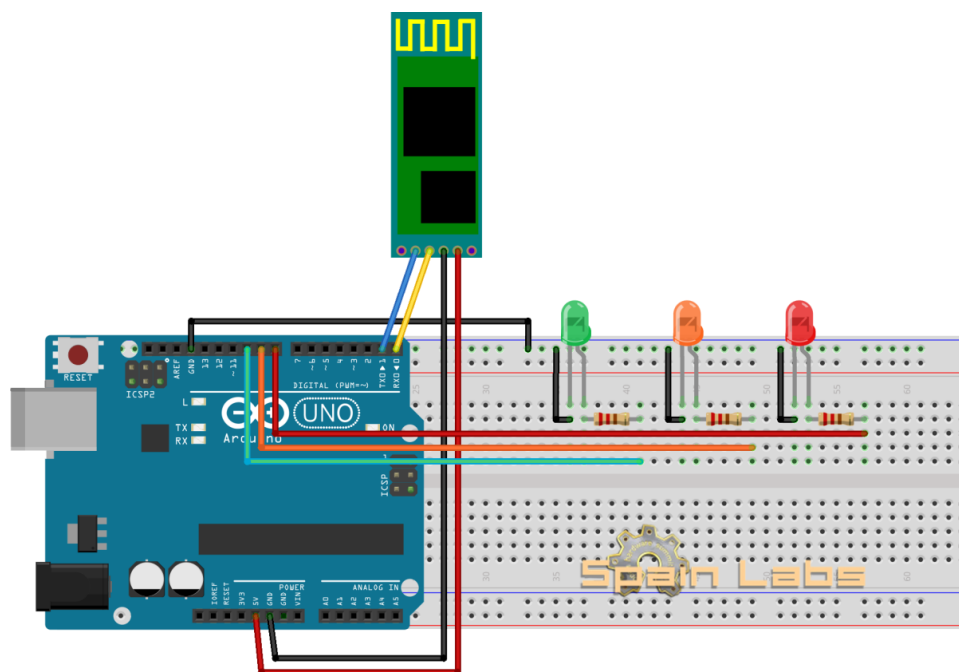
Y este sería el resultado final, añadiéndole al botón "salir" la acción de cerrar la aplicación:



**Código Arduino Para Este Ejemplo:**

```
1 //Ejemplo de control de 3 LEDs mediante un módulo bluetooth y dispositivo Android.
2 String dato;
3 int rojo= 8;
4 int naranja= 9;
5 int verde= 10;
6
7 void setup() {
8   Serial.begin(9600);
9
10  //Salidas digitales
11  pinMode(rojo, OUTPUT);
12  pinMode(naranja, OUTPUT);
13  pinMode(verde, OUTPUT);
14 }
15
16 void loop() {
17   while (Serial.available()) {
18     delay(10);
19     if (Serial.available() > 0) {
20       char c = Serial.read();
21     }
22   }
23   if (dato.length() > 0) {
24     if(dato == "1") {
25       Serial.println("Encender led rojo");
26       digitalWrite(rojo, HIGH);
27     }
28     if(dato == "A") {
29       Serial.println("Apagar led rojo");
30       digitalWrite(rojo, LOW);
31     }
32     if(dato == "2") {
33       Serial.println("Encender led naranja");
34       digitalWrite(naranja, HIGH);
35     }
36     if(dato == "B") {
37       Serial.println("Apagar led naranja");
38       digitalWrite(naranja, LOW);
39     }
40     if(dato == "3") {
41       Serial.println("Encender led verde");
42       digitalWrite(verde, HIGH);
43     }
44     if(dato == "C") {
45       Serial.println("Apagar led verde");
46       digitalWrite(verde, LOW);
47     }
48     dato=""; //
49   }
50 }
```

### Conexión Shield Bluetooth Arduino



**Figura 16. Montaje de la Conexión Shield Bluetooth**

La programación se la realiza de una manera correcta haciendo pruebas de conexión de las placas del microcontrolador y la aplicación mediante bluetooth corrigiendo errores que se presentaron en las pruebas.

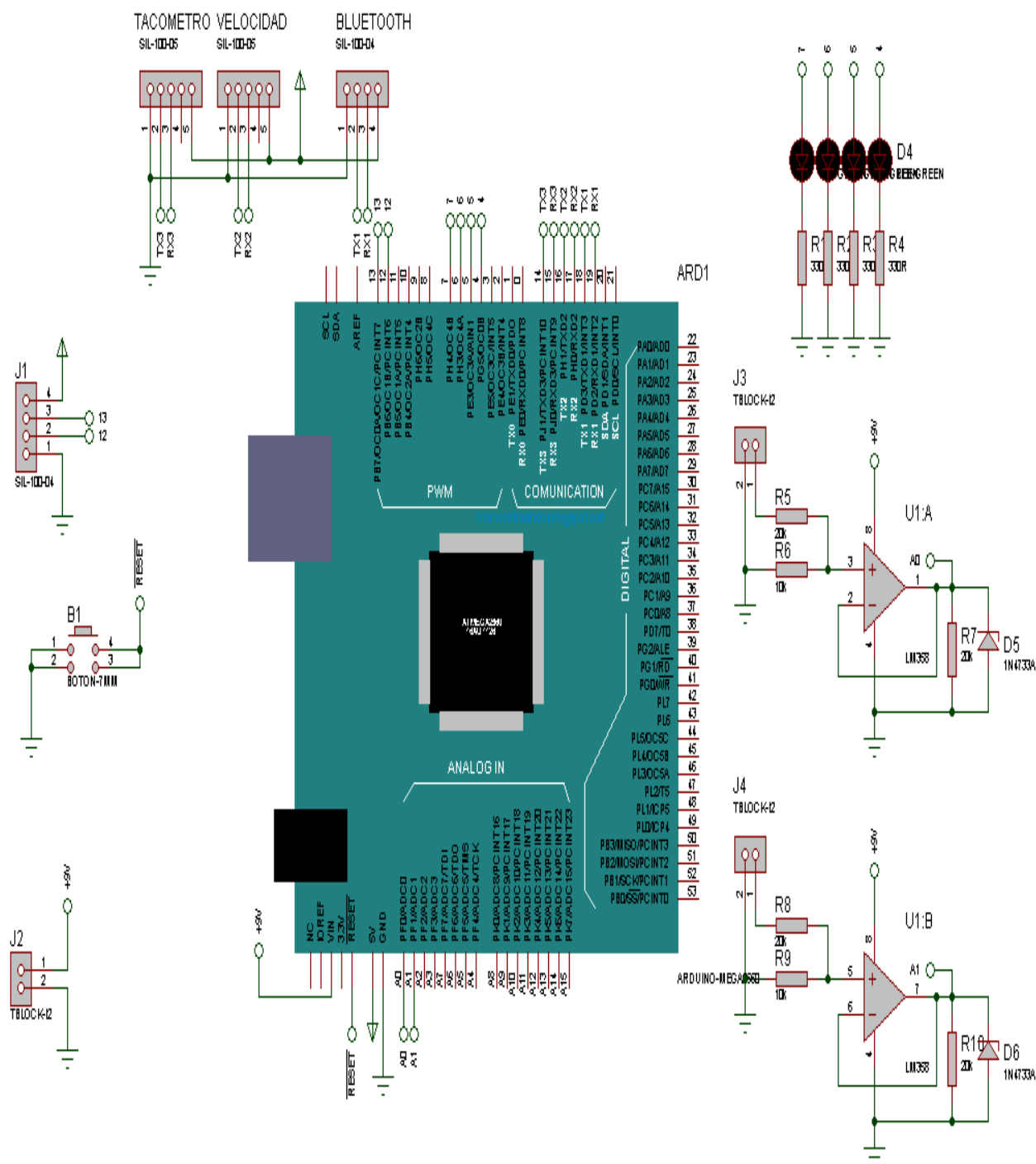
**Diseño del Hardware:**

Se realizo un grafico de la simulación por etapas y conexiones de cada uno ya que cada Arduino tiene la función de calcular durante una unidad de tiempo que en este caso es el segundo, la cantidad de pulsos generados por las señales y pulsos del velocímetro y el tacómetro.

En cambio el controlador maestro que es el Arduino mega sirve para comandar los datos de los sensores los cuales cuentan con una placa de acondicionamiento de señal debido que ingresan al Arduino señales de mayores de 12 voltios y se requiere hacer un reacondicionamiento de señal.

Los actuadores pueden ser el encendido de las luces, bloque central de seguros, bocina etc. Son controlados por el Arduino el cual envía un voltaje y con una conexión por medio de un relé se acondiciona la señal para que envíen la señal lo necesario para actuar. Esto a su vez establece las comunicaciones seriales para que los demás Arduinos devuelvan sus valores de sus respectivos sensores y enviar los datos al número serial que maneja el bluetooth para que la aplicación se encargue de mostrar el resultado en la Tableta o Smartphone.

Las placas trabajando conjuntamente tienen una respuesta muy rápida y efectiva en los medidores al igual que los actuadores que se comunican con la aplicación por medio de bluetooth de manera ininterrumpida





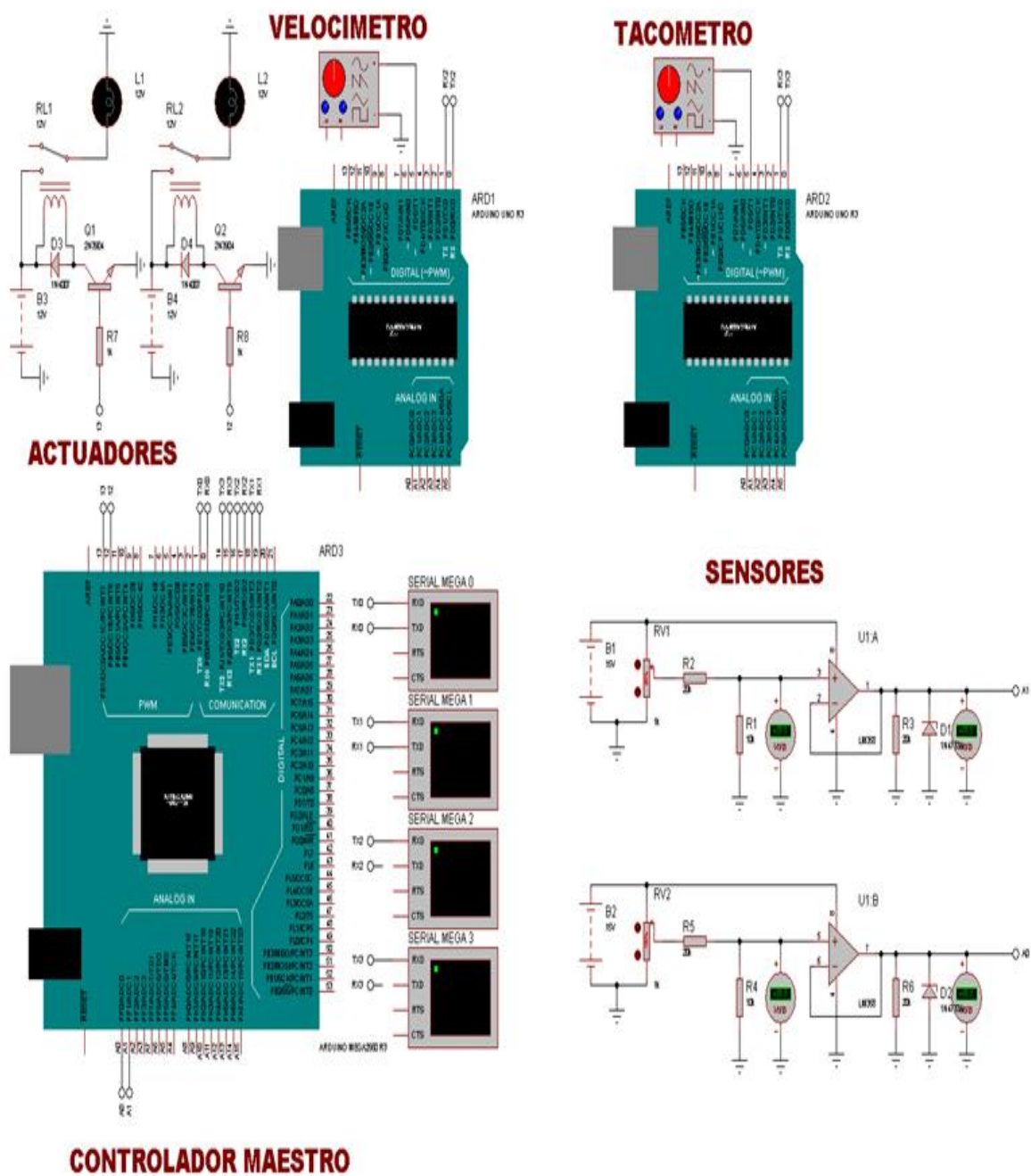


Figura 57. Simulación de conexión del sistema

### **Resultados**

La transmisión de datos usando varios procesadores tiene una respuesta muy rápida y no se pierden los datos al momento de recibir cada uno de los valores que han sido procesados por el microcontrolador.

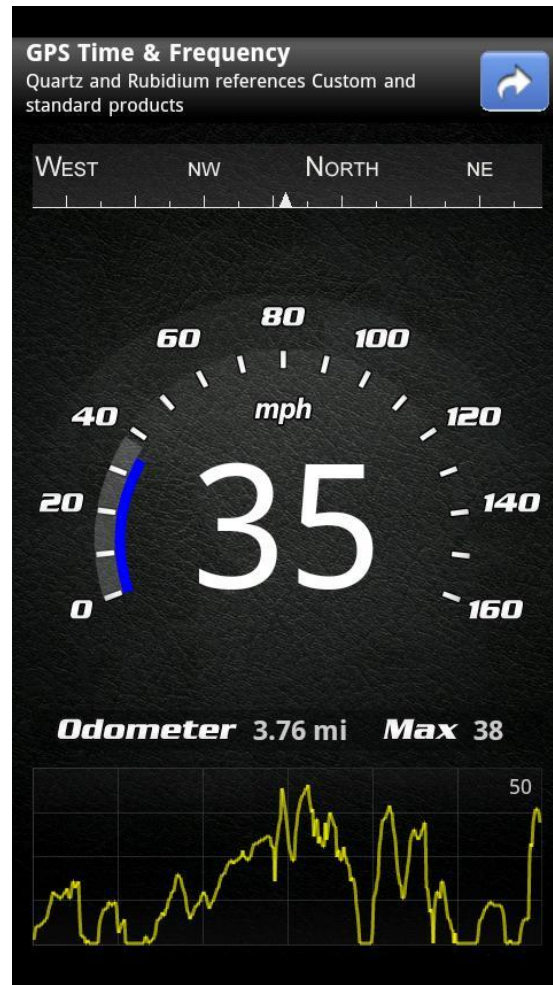
La placa del microcontrolador trabaja de una manera constante sin interrupciones en su funcionamiento gracias al acoplamiento de la fuente y regulador de voltaje que evita que se sobrealimente y haya pérdida de lectura si la placa se resetea y pierde momentos de procesamiento de datos para la impresión de pantalla.

La pantalla está diseñada para la recepción y envío de datos los más importantes en este caso son el velocímetro y tacómetro los cuales se manejan independientes cada uno con una placa y procesador, dos medidores de sensores acondicionados en los cuales se puede leer medidor de combustible, medidor de batería medidor de temperatura , nivel de aceite y dos actuadores para envío de datos en los cuales se puede incorporar encendido de luces , bocina, bloqueo central etc. .



**Figura 68. Aplicación Desde el Smartphone**

En la actualidad existen aplicaciones Android en las cuales tienen medidores de velocidad estas aplicaciones no solo sirven para cuando vas en el automóvil ya que la puedes usar para correr, ir en bici, etc. Disponen de muchas funciones como el poder ver la distancia recorrida y hacerte gráficos con la velocidad que has llevado. la interfaz es muy sencilla de usar. tan solo tenemos que encender el GPS y correr la aplicación. Se puede configurar si deseas verlo en millas por hora o kilómetros por hora.



**Figura 19. Pantalla aplicación Velocímetro Android**  
(AppStore,2015)

Al usar la aplicación mediante un satélite GPS la respuesta no instantánea se pierden valores y la aplicación no tiene una secuencia sucesiva de datos correctamente en la aplicación comandada por un microcontrolador los valores nunca se pierden y los datos son mucho más exactos.

## Conclusiones

Después de analizar los datos se encontró que la señal análoga no es interpretable mediante el uso de un Arduino por lo que se debió modificar el tipo de señal por medio de un convertidor obteniendo datos digitales apropiados para la medición.

Al conectar el microcontrolador a las entradas de señal provenientes del vehículo varios datos a ser procesados para lecturas de los medidores del tablero se perdían valores y esto se debe a un problema de recepción ya que el microcontrolador tiene un procesador de datos que trabaja en sincronización uno a la vez y al recibir los valores de cada uno de los medidores se pierde lectura de los otros, esto sucede en fracciones de tiempo y al momento de imprimir en pantalla los valores finales no se los mantiene estables por lo cual se incrementó varios microcontroladores para evitar interrupciones en la lectura de cada señal con esto se tiene mediciones en tiempo real gracias de manera interrumpida y con una velocidad de transferencias de datos de igual manera.

Para evitar la incoherencia de datos hay que revisar los valores en manuales o fichas técnicas del auto con esto se tiene un correcto valor sobre los datos a procesar en el microcontrolador y evitar daños en el sistema eléctrico del auto.

### **Recomendaciones**

Se recomienda trabajar con el microcontrolador al voltaje sugerido por el desarrollador ya que si no es así el microcontrolador se reseteara y no tiene una secuencia de recepción y envío de datos lo cual hace perder la información y no funciona adecuadamente, se recomienda regularlo y acondicionarlo para el ingreso de la fuente de energía a 12 voltios.

También se recomienda usar una señal estable al reducir ruidos eléctricos, para evitar interrupciones al momento de la transferencia de datos y así obtenerlos de una manera más estable y confiable teniendo valores más exactos y sin variaciones constantes de señal.

Revisar los valores que nos entrega cada señal del automóvil antes de conectarlos con la placa para hacer el acondicionamiento de señal al voltaje indicado ya que hay señales en el automóvil que tienen gran cantidad de voltaje y si no se regula de la manera correcta se podría llegar a circuitar y quemar el microcontrolador.

Seleccionar adecuadamente los cables de recepción de señal de los medidores para tener una lectura exacta y no perder tiempo ya que muchos de estos se alimentan a 12v pero no tienen variación y se confunde el cable de señal con el cable alimentación.

## Anexos

### Anexo 1:

#### Programación del Velocímetro Digital:

```
#include <FreqCount.h>
String inputString = "";
boolean stringComplete = false;
unsigned long data=0;
void setup()

{

Serial.begin(9600);
  FreqCount.begin(1000);
  inputString.reserve(200);

}

void loop() {
  if (FreqCount.available()) {
    unsigned long count = FreqCount.read();
    data = count;

  }

  if (stringComplete) {
    //Serial.println(inputString);
    inputString.trim();
    if(inputString=="velocidad"){
      data = data/4;
      data = data*1.23;
      data = data*60;
      data= data/1000;
      Serial.print("V: ");
      Serial.print(data);
      Serial.println("Km/h");
    }

  }

  inputString = "";
  stringComplete = false;
```

```
    }  
}  
  
void serialEvent() {  
    while (Serial.available()) {  
        char inChar = (char)Serial.read();  
        inputString += inChar;  
        if (inChar == '\n' || inChar == '\r') {  
            stringComplete = true;  
  
        }  
    }  
}
```



**Anexo 2:****Programación del Tacómetro (Numero de Revoluciones RPM)**

```
#include <FreqCount.h>
String inputString = "";
boolean stringComplete = false;
unsigned long data=0;
void setup()

{

  Serial.begin(9600);
  FreqCount.begin(1000);
  inputString.reserve(200);

}

void loop() {
  if (FreqCount.available()) {
    unsigned long count = FreqCount.read();
    data = count;

  }

  if (stringComplete) {
    //Serial.println(inputString);
    inputString.trim();
    if(inputString=="revoluciones"){
      data = map(data,0,60,0,1800);
      Serial.print("R: ");
      Serial.print(data);
      Serial.println("RPM");

    }

    inputString = "";
    stringComplete = false;

  }
}
```

```
void serialEvent() {  
  while (Serial.available()) {  
    char inChar = (char)Serial.read();  
    inputString += inChar;  
    if (inChar == '\n' || inChar == '\r') {  
      stringComplete = true;  
    }  
  }  
}
```

**Anexo 3:****Controlador Maestro (Arduino Mega):**

El controlador maestro que es comandado por el Arduino mega maneja las seriales mediante evento de interrupción que cuando el bluetooth recibe un dato este a su vez lo valida y luego según si es válido o no este envía una petición a los Arduinos de velocidad y tacómetro, de ahí tiene el control directo de los actuadores y de los sensores que de igual manera se piden desde el bluetooth:

```
String inputString = "";
String inputString1 = "";
String inputString2 = "";
String inputString3 = "";

boolean stringComplete = false;
boolean stringComplete1 = false;
boolean stringComplete2 = false;
boolean stringComplete3 = false;

void setup() {

  Serial.begin(9600);
  Serial1.begin(9600);
  Serial2.begin(9600);
  Serial3.begin(9600);

  inputString.reserve(200);
  inputString1.reserve(200);
  inputString2.reserve(200);
  inputString3.reserve(200);

  pinMode(13, OUTPUT);
  pinMode(12, OUTPUT);
  digitalWrite(13,0);
  digitalWrite(12,0);

}
```

```
void loop() {  
  if (stringComplete) {  
    Serial.println(inputString);  
    inputString = "";  
    stringComplete = false;  
  
  }  
  
  if (stringComplete1) {  
    //Serial1.println(inputString1);  
    inputString1.trim();  
    if(inputString1=="velocidad"){  
      Serial2.println("velocidad");  
  
    }  
  
    if(inputString1=="revoluciones"){  
      Serial3.println("revoluciones");  
  
    }  
  
    if(inputString1=="aON"){  
      digitalWrite(13,1);  
      //Serial1.println("a on");  
  
    }  
  
    if(inputString1=="aOFF"){  
      digitalWrite(13,0);  
      //Serial1.println("a off");  
  
    }  
  
    if(inputString1=="bON"){  
      digitalWrite(12,1);  
      //Serial1.println("b on");  
  
    }  
  
    if(inputString1=="bOFF"){  
      digitalWrite(12,0);  
      //Serial1.println("b off");  
  
    }  
  
    if(inputString1=="S1"){
```

```
float s1 = analogRead(A0);
s1 = (s1*5.00)/1024;
Serial1.print("S1: ");
Serial1.println(s1);

}

if(inputString1=="S2"){
float s2 = analogRead(A1);
s2 = (s2*5.00)/1024;
Serial1.print("S2: ");
Serial1.println(s2);

}

inputString1 = "";
stringComplete1 = false;

}

if (stringComplete2) {
Serial2.println(inputString2);

inputString2.trim();
String aux2=inputString2;
if(inputString2.substring(0,3)=="V: "){
Serial1.println(aux2);
}

inputString2 = "";
stringComplete2 = false;

}

if (stringComplete3) {
Serial3.println(inputString3);

inputString3.trim();
String aux3=inputString3;
if(inputString3.substring(0,3)=="R: "){
Serial1.println(aux3);
}

inputString3 = "";
stringComplete3 = false;
```

```
    }  
  }  
  
  void serialEvent() {  
    while (Serial.available()) {  
      char inChar = (char)Serial.read();  
      inputString += inChar;  
      if (inChar == '\n' || inChar == '\r') {  
        stringComplete = true;  
  
      }  
    }  
  }  
  
  void serialEvent1() {  
    while (Serial1.available()) {  
      char inChar1 = (char)Serial1.read();  
      inputString1 += inChar1;  
      if (inChar1 == '\n' || inChar1 == '\r') {  
        stringComplete1 = true;  
  
      }  
    }  
  }  
  
  void serialEvent2() {  
    while (Serial2.available()) {  
      char inChar2 = (char)Serial2.read();  
      inputString2 += inChar2;  
      if (inChar2 == '\n' || inChar2 == '\r') {  
        stringComplete2 = true;  
  
      }  
    }  
  }  
  
  void serialEvent3() {  
    while (Serial3.available()) {  
      char inChar3 = (char)Serial3.read();  
      inputString3 += inChar3;  
      if (inChar3 == '\n' || inChar3 == '\r') {  
        stringComplete3 = true;  
  
      }  
    }  
  }  
}
```

#### Anexo 4

##### Programación Aplicación Android (app inventor):

En el desarrollo de la aplicación la programación se la realiza de la siguiente manera , el código es desde la aplicación app inventor es visual y consta de un proceso de eventos los cuales van representados por cada figura conectada

Con esto tenemos desde la conexión de la aplicación al dispositivo android por medio del bluetooth hasta cada salida de pantalla de cada medidor y actuador variando simplemente las salidas programadas desde el Arduino para su pantalla que también es programada para establecer los parámetros y el orden de cada medidor para su visualización final .

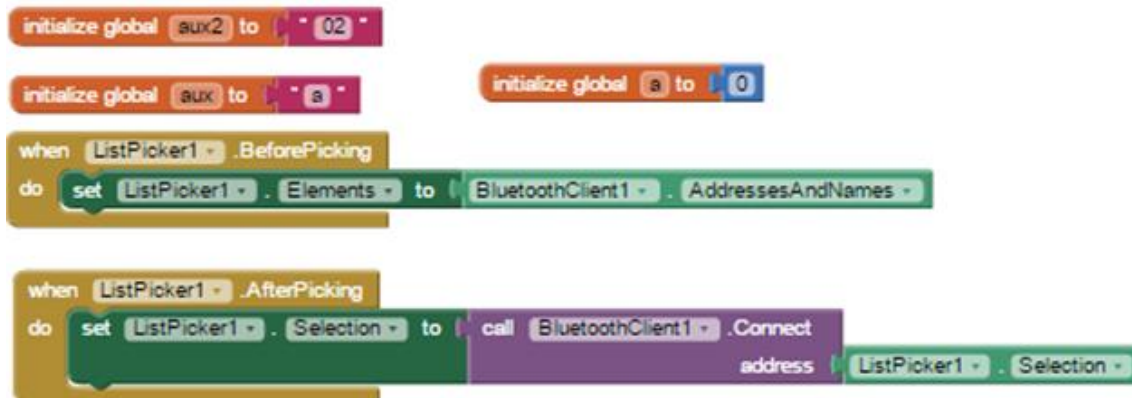


Figura 15. Diagrama conexión AppInventor Bluetooth

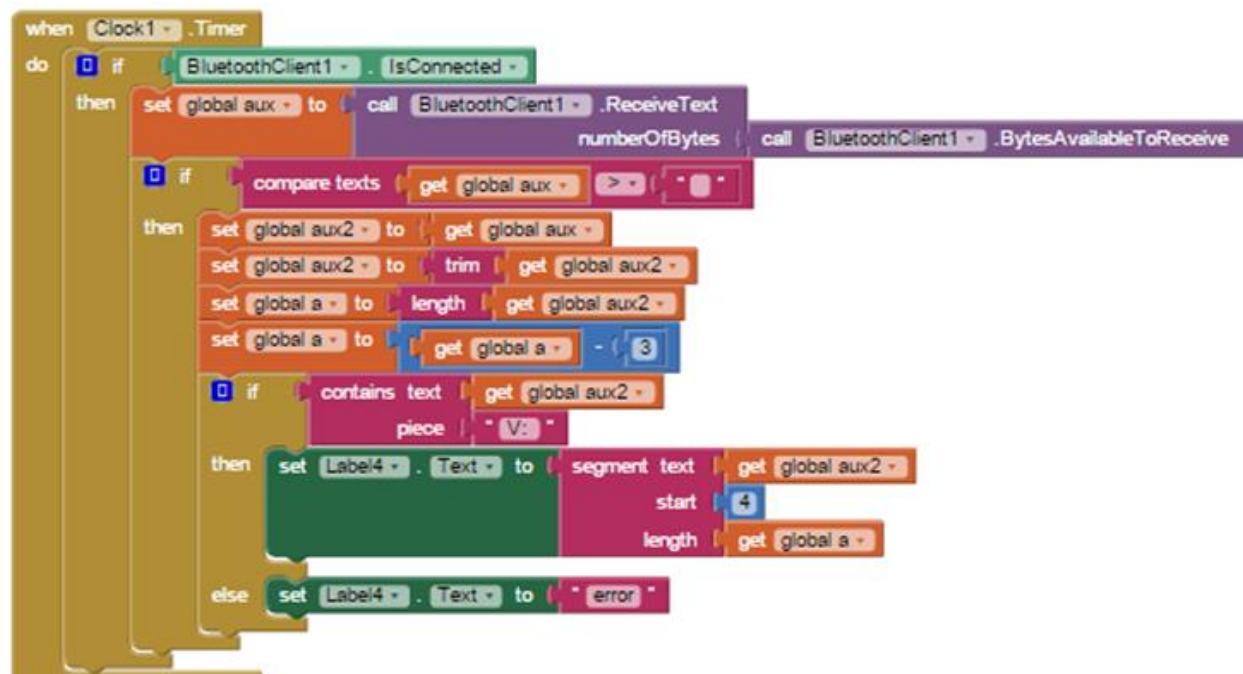


Figura 16. Cuadro de Programación App Inventor

El código de programación de la aplicación en App inventor es el mismo ya que solo es salida de impresión de pantalla en este caso solo se dirige a cada bloque de impresión designada en la parte visual al cambiar el modulo que recibe desde la programación del Arduino que es el que se encarga de la correcta interpretación de los valores que van a ser representados en el cuadro de impresión de pantalla, que se implemento en la parte visual de la aplicación.



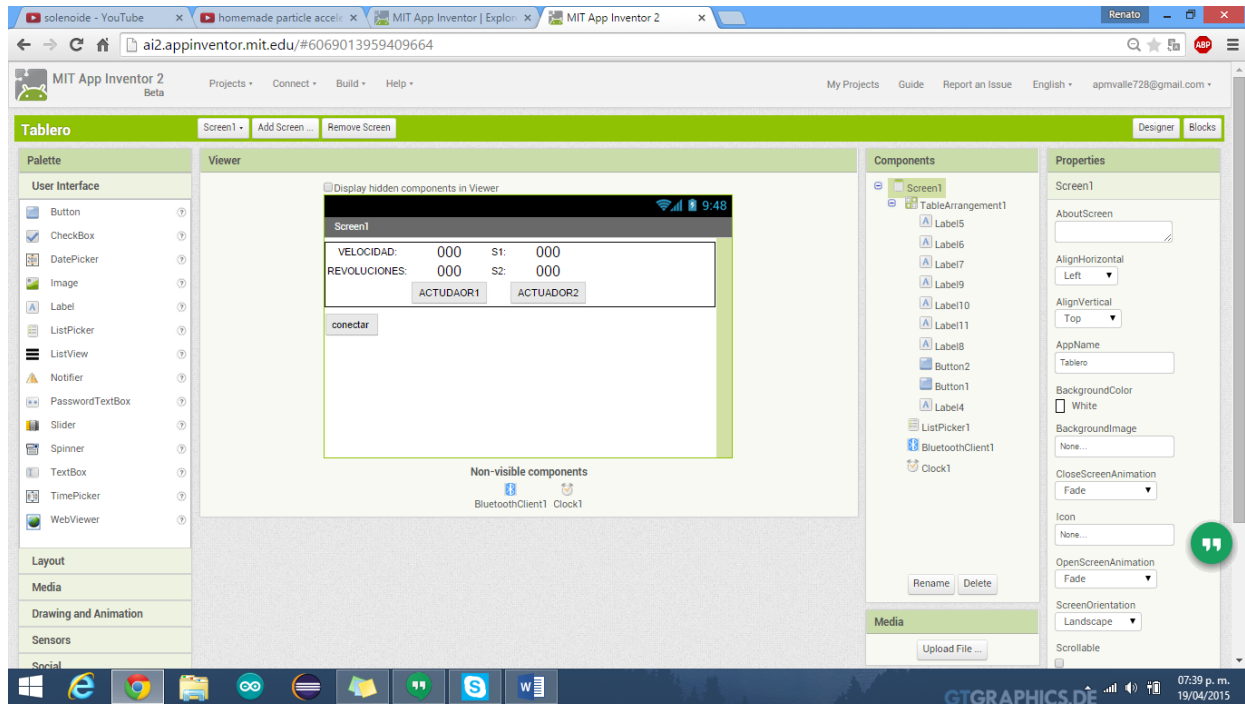


Figura 17. Programación Parte Visual App Inventor

## Anexo 5

Resultados de encuestas realizadas: <https://www.onlineencuesta.com/results/127202b-a11733e>

1. Nombres y apellidos del entrevistado:	2. Cédula de identidad:	3. Dispone de un dispositivo móvil con sistema Android:	4. Marca del vehículo:	5. Modelo del vehículo:	6. Placa:	7. Año de fabricación:	8. Qué tipo de señales tiene el tablero de su vehículo:	9. En el tablero de su vehículo, los indicadores funcionan correctamente:	10. Estaría dispuesto a invertir para que se pueda visualizar de manera distinta los indicadores del tablero (mediante su dispositivo android):	11. Cuánto estaría dispuesto a invertir por dicho cambio:	12. Estaría dispuesto a cambiar el sistema de señales de su vehículo a señales digitales:	
Geovanni Guerra	401199286	si	Jeep	Willya	PVW115	1977	Analogico	si	si	0-100	si	
francisco saa	1702972967	si	GMC	Sierra	tba-0895	1954	Analogico	si	si	100-200	si	
Pedro Klac	1715968070	si	Chevrolet	Nova	PGO875	1975	Analogico	si	no	0-100	no	
vibiana valdivieso	1708929466	si	vw	escarabajo	pfg-532	1975	Analogico	si	si	0-100	si	
Carlos Vinueto	1709764920	si	Ford	Maverick	PBW387	1977	Analogico	si	si	100-200	si	
Abrahan Proaño	1705616355	si	mini	austin 1000	pck-163	1977	Analogico	si	no	0-100	no	
Luis Palacio	1702181932	si	Ford	Bronco	TBD 073	1980	Analogico	no	si	0-100	si	
Juan nicanor ramirez	1000234755	si	ford		350	1976	Analogico	no	si	100-200	si	
vinicio peña	100026863	si	Ford		250	ABJ-793	1973	Analogico	si	200-300	si	
Jose Javier Peñaherrera Rodriguez	1701175810	si	ford		250	MBW 556	1986	Analogico	no	100-200	si	
Luis Pacheco	602290058	si	vw	escarabajo	Pbf 666	1972	Analogico	si	no	0-100	no	
Edwin Ramiro Peñafiel Hinojosa	400579934	si	Ford	Courier	cba-230	1976	Analogico	no	si	100-200	si	
Mónica Judith Matute Barrera	30166570	si	toyota	corola	xbe-034	1980	Analogico	si	si	100-200	si	
Diego Muñoz	1710145093	si	Jeep	Willys	PTH 893	1977	Analogico	si	no	0-100	no	
Freddy Tapia Camino	170895674	si	Datsun	120V	XBE - 0372	1980	Analogico	si	si	0-100	si	
fernando Velastegui	1713275138	si	Subaru	A22	pdf698	1980	Analogico	si	no	0-100	no	
Patricio Jarrin Barba	1701321554	si	Toyota	Hiace	PCM-406	1978	Analogico	no	si	100-200	si	
Oscar Orlando Benavides	4011177908	si	Chevrolet	bel air	PLC 735	1956	Analogico	si	si	0-100	si	
Gabriel Cozzarelli Vazquez	909659229	si	Mini	Austin	PB5-817	1978	Analogico	no	no	0-100	no	
Juan Naranjo	1801424340	si	Opel	GT	PBX0351	1970	Analogico	no	si	0-100	si	
Marco Andre Atiencia Vera	1719372490	no										
Diego Fernandez	1703109270	si	BMW		1602	Pgb0835	1975	Analogico	si	si	100-200	si
Geraldo Steven Castro Guerrero	1723793483	si	Datsun		1500	pcn-066	1975	Analogico	si	si	0-100	si
Freddy Vinicio Gutierrez	703925354	si	datsun		1200	GF5 - 215	1974	Analogico	no	si	0-100	no
Miguel Calderón Alava	1301706253	no										
Irma Pérez Malats	1704132545	no										
Donald Ian Cass	1105630261	si	Toyota	Land cruiser	TBL-937	1979	Analogico	no	si	100-200	si	
Carlota Freddy	402491448	si	BMW		520	GGH-478	1973	Analogico	si	no	0-100	no
Carlos Freire	1801771245	si	Benz	280sle	PHC0611	1968	Analogico	si	si	0-100	si	
Luis Roberto Guadalucho	1709491219	si	jeep	CJ5	pdt 089	1975	Analogico	si	no	0-100	no	
Edgar Augusto Bolaños Lima	400063723	si	jeep	comando	pfn-453	1975	Analogico	no	no	0-100	no	
Luis Garcia	102598745	si	Toyota	stout	XBM-376	1980	Analogico	si	no	0-100	no	
Francisco Calderon	1714285267	si	volswagen	Escarabajo	PHM0714	1966	Analogico	si	si	200-300	si	
Viviana Calderón Pérez	1714285275	no										
Irma Calderón	1714285259	no										
Mauricio Kranevitter	26941141	si	Wolsvagen	Voyage	Onc376	2014	Digital	si	si	200-300	si	
Andres Gabriel Sanchez ordoñez	1712909207	si	Bmw	323i	gcb599	1985	Analogico	no	no	0-100	si	
Julio Cesar Coronel	1718836255	si	BMW		320	GCS453	1978	Analogico	si	si	0-100	no
Francisco Soto	1714190848	si	suzuki	samurai	pju-015	1984	Analogico	no	si	100-200	si	
Pedro Pérez	1800761049	si	Chevrolet	Condor	TBC - 0173	1980	Analogico	si	si	0-100	si	
Juan Gallardo	500651260	si	BMW		525	pvw - 0321	1980	Analogico	si	100-200	si	
Segundo Casillas	1704411584	no										
Diego Chiriboga	1701620260	no										
Manuel Luzuriaga	1802170579	si	BMW		733	PJP - 0374	1984	Analogico	si	si	0-100	si

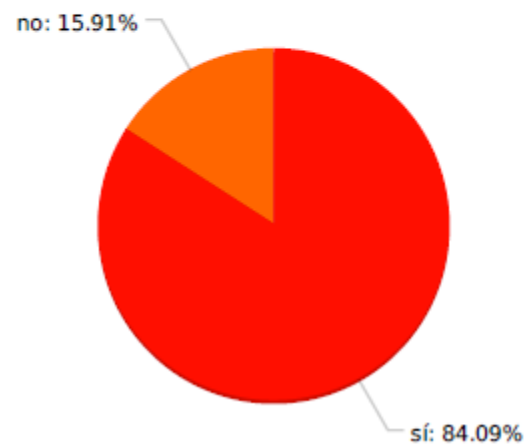
A continuación se adjuntan los indicadores de las preguntas que tienen tabulación:

3. Dispone de un dispositivo móvil con sistema Android: \*

Número de participantes: 44

37 (84.1%): sí

7 (15.9%): no

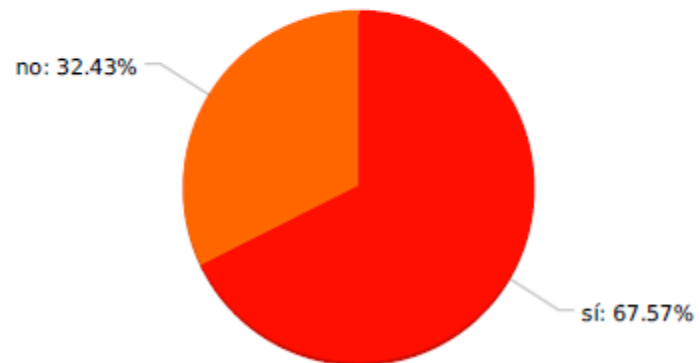


9. En el tablero de su vehículo, los indicadores funcionan correctamente: \*

Número de participantes: 37

25 (67.6%): sí

12 (32.4%): no

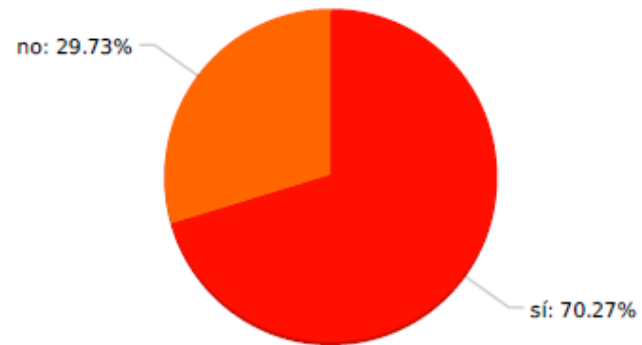


10. Estaría dispuesto a invertir para que se pueda visualizar de manera distinta los indicadores del tablero (mediante su dispositivo android): \*

Número de participantes: 37

26 (70.3%): sí

11 (29.7%): no



## 11. Cuánto estaría dispuesto a invertir por dicho cambio: \*

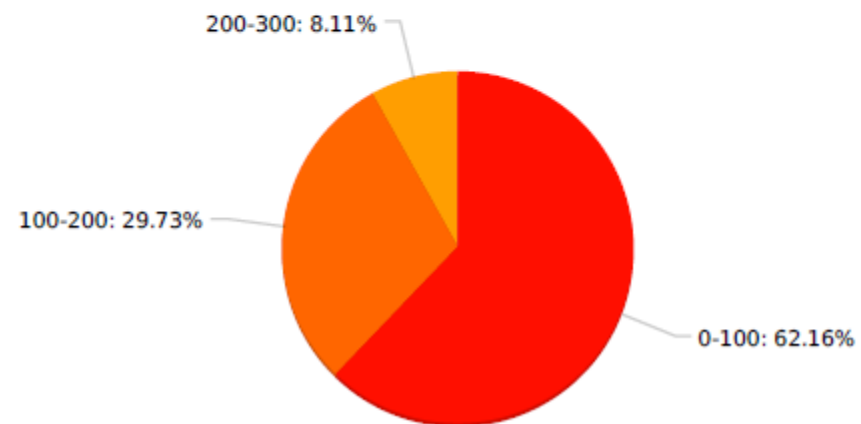
Número de participantes: 37

23 (62.2%): 0-100

11 (29.7%): 100-200

3 (8.1%): 200-300

- (0.0%): 300-400



### Referencias

Arduino, Arduino Web Oficial, (2006)

Recuperado de:

<http://www.arduino.cc/en/Main/arduinoBoardMega>

Arduino, Arduino Web Oficial, (2006)

Recuperado de:

<http://www.arduino.cc/>

Ivan Cuadriello Automotriz, (2012) Elementos amovibles Panel de Instrumentos

Recuperado de:

<http://ivancuadriello-eaf.blogspot.com.ar/2012/03/elementos-amovibles-panel-de.html>

Educacion, (2013) Uso de AppInventor en la asignatura de tecnologías de la información y la comunicación

Recuperado de:

<http://recursostic.educacion.es/observatorio/web/en/software/programacion/1090-uso-de-appinventor-en-la-asignatura-de-tecnologias-de-la-comunicacion-y-la-informacion>

Automovil,(2013)\_*Panel de Instrumentos del Automóvil*

Recuperado de:

<http://www.sabelotodo.org/automovil/panelinstru.html>

AppInventor, (2015). MIT App Inventor Beta.

Recuperado de:

<http://beta.appinventor.mit.edu/>

Juan Camilo Diaz (2014 Colegio La Victoria I.E.D Arquitectura del PC.

Recuperado de:

[http://issuu.com/juancamilomartinez1/docs/arquitectura\\_del\\_pc](http://issuu.com/juancamilomartinez1/docs/arquitectura_del_pc)

APPIinventor, (2011) Apuntes para un aprendiz de programador.

Recuperado de:

[http://www.utm.mx/edi\\_anteriores/temas45/2NOTAS\\_45\\_3.pdf](http://www.utm.mx/edi_anteriores/temas45/2NOTAS_45_3.pdf)

Arduino M. , Basicos Arduino, (2014) Básicos Arduino.

Recuperado de:

<http://medialab.usal.es/facartec/files/2012/10/B%C3%A1sicos-Arduino.pdf>

Gomez, (2007) Arquitectura del Computador.

Recuperado de:

<http://issuu.com/wendasii/docs/pdf.docx>



Alonso, (2006) Circuitos eléctricos Auxiliares Electromecánica de Vehículos

López, J.L (2011).GM Circuitos Eléctricos Auxiliares.España:Paraninfo.