UNIVERSIDAD DE SONORA
DIVISIÓN DE INGENIERÍA
DEPARTAMENTO DE INGENIERÍA INDUSTRIAL

INGENIERÍA EN MECATRÓNICA

Integración de las plataformas Android-LabVIEW-Arduino para el control de motores de corriente directa

TESIS

Que para obtener el Título de:

INGENIERO EN MECATRÓNICA

Presenta:

José Ángel Basurto López

Hermosillo, Sonora **Junio de 2014**

Índice

Capítulo 1

Planteamiento del problema

1.1 Antecedentes

Los sistemas ciber-físicos son aquellos que cuentan con elementos computacionales que controlan entidades físicas. Existe en la actualidad una gran variedad de sistemas ciber-físicos que se pueden encontrar en áreas tan diversas como la industria aeroespacial, automotriz, procesos químicos, energía, salud y transporte por mencionar algunas [1]. El objetivo de los sistemas ciber-físicos es la interacción de forma continua y dinámica con su entorno a través del acoplamiento de componentes computacionales y físicos, por ejemplo el control de máquinas eléctricas utilizando sistemas de Control Supervisorio y de Adquisición de Datos (SCADA).

Muchas plataformas aprovechan el uso de dispositivos móviles como sensores obtener una retroalimentación como se muestra en [2], donde la comunicación y las tareas de control se realizan mediante el uso de un dispositivo móvil. Las tareas se basan con frecuencia en datos de sensores GPS, acelerómetros, giroscopios, etcétera, sin embargo, la ejecución de tareas concurrentes en estos dispositivos introduce complejidades innecesarias si las nuevas tareas se agregan sin una revisión de diseño del sistema. En este trabajo se pretende demostrar que mediante la integración de varias plataformas, es posible realizar control concurrente de una manera sencilla aprovechando la integración de varias tecnologías.

1.2 Problema actual

La integración de un sistema operativo para dispositivos móviles, un ambiente de programación de tipo gráfico y una plataforma de desarrollo para microcontrolador es una tarea difícil si no se tienen las bases necesarias para ello o si no se conocen los procedimientos correctos para llevar a cabo dicha integración. Actualmente no se tiene documentado un proyecto de integración de tres o más plataformas para realizar una tarea específica, en este caso el control de dos motores que emulen el comportamiento de un automóvil, por lo tanto, si se desea probar que es posible comunicar el software de desarrollo en ingeniera LabVIEW con la plataforma de hardware libre Arduino UNO es necesario realizar una serie de experimentos con la finalidad de controlar de manera concurrente dos motores de corriente directa que emulen el comportamiento de un auto utilizando los sensores de un *Smartphone* para el envío de instrucciones.

La Figura 1.1 muestra el esquema general del problema a resolver, donde se identifican las tres plataformas y los actuadores finales. Como se puede apreciar en la figura, la comunicación del dispositivo móvil con la computadora, se realiza mediante el uso de tecnología bluetooth. Una vez enviada la instrucción, esta se procesa utilizando LabVIEW y los datos se envían a un sistema Arduino donde, nuevamente son procesados para enviar la orden final a los motores de corriente directa para lo cual es necesario contar con una etapa de potencia.

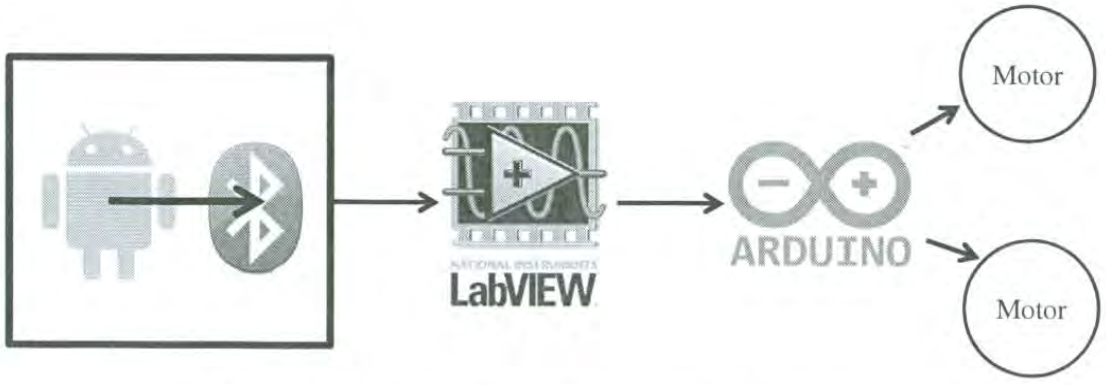


Figura 1.1 Esquema general de integración de plataformas

1.3 Justificación

Existen en la actualidad varias plataformas que se pueden integrar para la creación de un sistema ciber-físico, entre ellas, la plataforma de programación LabVIEW, la plataforma de desarrollo Arduino y el sistema operativo Android. La integración de estas tres poderosas herramientas permite realizar cálculos complejos para enviar instrucciones de manera analógica o digital hacia los efectores finales. Uno de los problemas recurrentes en la creación de sistemas ciber-fisicos es la falta de sensores o la falta de herramientas para desarrollo de estos sistemas, es por ello que el uso del sistema Android se propone en este trabajo ya que los dispositivos que utilizan este sistema son cada vez mas comunes y de fácil acceso.

Por otro lado es necesario incursionar en la integración de tecnologías de gran escala con la finalidad de reducir costos de desarrollo e implementación y con ello enfocarse en la solución de los problemas más que en las herramientas, un ejemplo claro es el diseño asistido por computadora, donde la herramienta ayuda al diseñador y los procesos internos son transparentes para este.

1.4 Objetivo general

Integrar el sistema operativo Android con el sistema de desarrollo LabVIEW y este a su vez con la plataforma Arduino y probar el funcionamiento de dicha integración mediante la construcción de un sistema ciber-físico el cual emule el comportamiento de un automóvil utilizando los sensores de un *Smartphone* para controlar de manera concurrente dos motores de Corriente Directa (CD).

Para lograr el objetivo anterior es necesario realizar una serie de experimentos con la finalidad de encontrar la manera más fácil de integrar estas plataformas, para ello se requiere:

- Enviar los datos del sensor de orientación de un *Smartphone* con sistema operativo Android a LabView.

- Analizar y procesar, en LabVIEW, los datos recibidos del *Smartphone* para encontrar los parámetros que ayuden en la velocidad y orientación del automóvil simulado.
- Enviar los datos procesados por LabVIEW a un sistema Arduino, para lo cual se debe encontrar la manera de integrar la plataforma de desarrollo LabVIEW con el sistema Arduino.
- Procesar en Arduino los datos recibidos desde LabVIEW para controlar de manera concurrente dos motores de corriente directa. El control debe ser para velocidad y dirección.

1.5 Hipótesis

Es posible controlar dos motores de corriente directa de manera concurrente utilizando para ello la plataforma Arduino como controlador y teniendo como interfaz gráfica LabVIEW, la cual recibirá instrucciones desde el sensor de posición de un *Smartphone*.

1.6 Delimitación

Emular el comportamiento de un automóvil en cuanto a velocidad y dirección utilizando para ello dos motores de CD. Las instrucciones de velocidad y dirección serán enviadas desde un Smartphone con Android hacia LabVIEW el cual a su vez enviará las instrucciones necesarias a la plataforma Arduino, la cual servirá como controlador.

1.7 Limitaciones

Este trabajo se limitará a probar el funcionamiento simultáneo de dos motores de CD, lo cual dará pie para que en trabajos futuros se emplee esta técnica en el control de autos guiados de manera autónoma, graficadores o routers, control de guías lineales, entre otros.

La etapa de potencia no estará considerada en los experimentos pues la finalidad de los mismos es la integración de los sistemas para construir un sistema ciber-físico.

Capítulo 2

2. Estado del Arte

El poder sensorial de los dispositivos Android es bastante sorprendente para muchos programadores de esta plataforma, lo que hace que los dispositivos sean herramientas sólidas que, alineadas con otras herramientas de desarrollo, pueden hacer uso de los datos generados por estos sensores para utilizarlos en distintos entornos.

Arduino es una plataforma de hardware libre, que utiliza el microcontrolador Atmel AV y un entorno de desarrollo amigable para el usuario. Esta plataforma fue diseñada para facilitar el uso de la electrónica en proyectos multidisciplinares. Mientras el hardware está dominado por lo microcontroladores Atmega, el software consiste en un entorno de desarrollo que implementa el lenguaje de programación Processing/Wiring y el cargador de arranque que es ejecutado en la plataforma. Sus distintos puertos de E/S permiten implementar algoritmos de control de una manera sencilla y económica.

LabVIEW es un lenguaje de programación gráfico que tiene la capacidad de leer la información de los sensores de los dispositivos Android para procesarla y comunicarla a la plataforma Arduino. Estas capacidades hacen que la integración de distintas plataformas y sistemas sean una realidad tangible e impulsan el desarrollo de proyectos de ingeniería de una manera rápida, enfocándose en la solución del problema más que en los problemas propios de un desarrollador que implementa soluciones a bajo nivel.

2.1 Aplicación

Hay aplicaciones realizadas para facilitar la comunicación entre el sistema operativo Android con el software de ingeniería LabVIEW, la cual permite conexión bluetooth para el envío de datos.

2.1.1 AndroView

Bluetooth AndroView fue desarrollado para facilitar la comunicación entre el dispositivo Android y un ordenador con LabView donde, simplemente, el usuario tiene acceso a los valores generados por sensores y procesadas para facilitar la manipulación algebraica. Con estos valores se hace muy fácil el desarrollo de diversas aplicaciones, desde el LabView también permite al programador desarrollar incluso la interfaz gráfica de la aplicación.

Figura 2.1 Aplicación Android para la comunicación con LabVIEW

El Bluetooth AndroView permite conexión vía Bluetooth entre LabVIEW y su dispositivo Android, el envío de los datos:

- Acelerómetro
- Sensor de campo magnético
- Sensor de luz
- Sensor de proximidad

- Sensor de presión
- Sensor de orientación
- Giroscopio
- Área jugable

2.2 Software

Existen varios entornos con los cuales es posible enlazarse eficazmente con Arduino, algunos de estos software son:

- Etoys(Squeak).
- Firefly.
- MyOpenLab.
- S4A (Scratch).

2.2.1 PHYSICAL ETOYS: Control de Arduino desde Etoys.

Figura 2.2 Pantalla principal del entorno PhysicalEtoys.

PhysicalEtoys es una herramienta de programación visual que une el mundo virtual de las computadoras con el mundo real en que vivimos. En [3] nos menciona que se puede fácilmente programar objetos del mundo real (como robots) para realizar tareas interesantes.

Es tan fácil de utilizar como agarrar elementos de la paleta (que representan algunos elementos de control de Arduino, así como otros elementos visuales para representar valores en la pantalla) y unirlos como un "*puzzle*" para que nos permita la generación de una aplicación de control y/o visualización.

Cada objeto dispone de parámetros a los que se designara unos valores de acuerdo con las preferencias y necesidades de la aplicación. PhysicalEtoys es en realidad una "extensión" de Etoys: un entorno de autor para la creación de aplicaciones multimedia y de simulación mediante la representación visual realizado por las mismas personas que crearon SmallTalk.

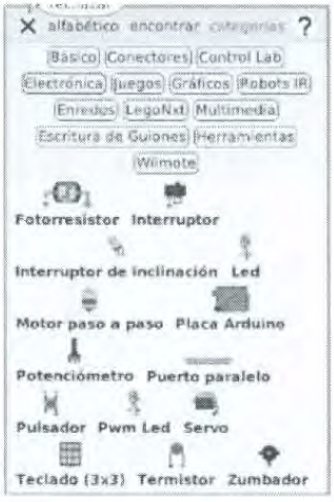


Figura 2.3 Menú principal de PhysicalEtoys.

Este entorno de programación nos permite comunicarnos con distintos dispositivos (además de Arduino) como son:

- NintendoWiimote.
- Puerto paralelo.
- RoboSapien V2.
- RoboQuad.

- I-Sobot.
- Lego MindStormNxt.

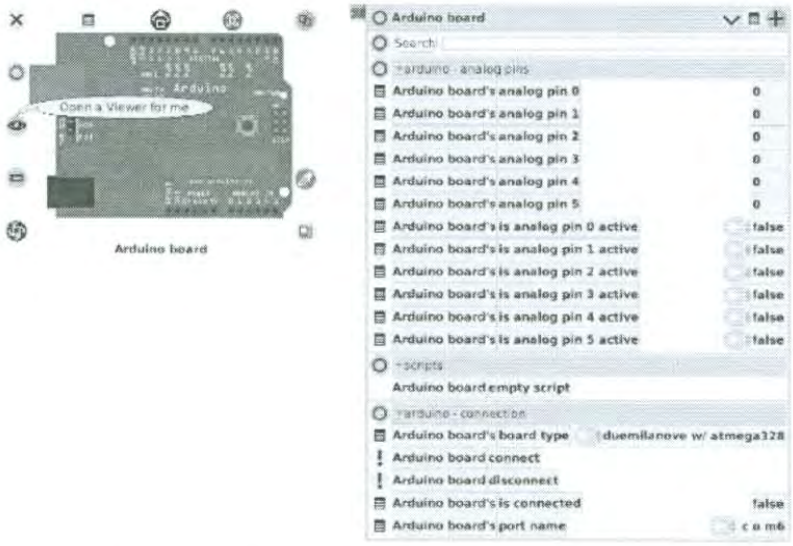


Figura 2.4 Pantalla donde es posible ver y modificar las propiedades de los objetos.

Figura 2.5 Selección del Arduino y el COM.

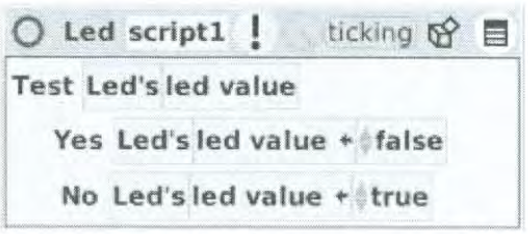


Figura 2.6 Ejemplo de una rutina de programación (led que parpadea).

2.2.2 FIREFLY

Es posible conectar Arduino al entorno gráfico Rhino a través del conocido *plugin* Grasshopper que es un entorno muy versátil y fácil de manipular, utilizado para la programación de eventos y manipulación de imágenes de Rhino.

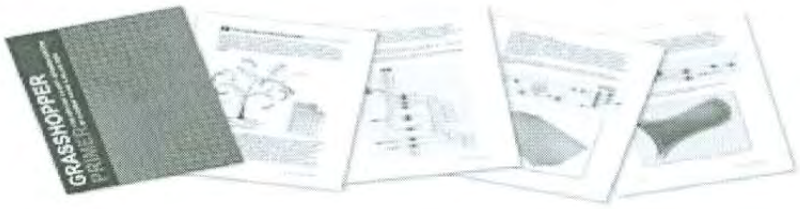
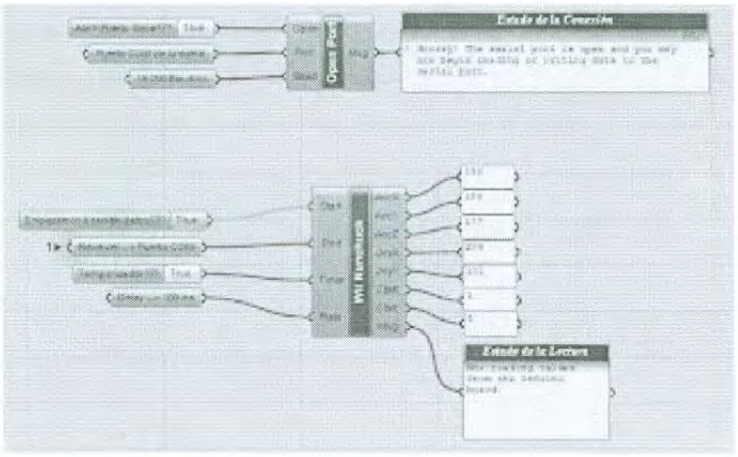


Figura 2.7 Recurso impreso para aprender acerca de Grasshopper.

La herramienta en cuestión, es la mencionada en [4], la librería Firefly, que incorpora un conjunto de bloques de función que se conectan con Arduino.



2.8 Bloques de función de la librería Firefly, utilizados en Grasshopper.

Firefly conecta el mundo del CAD/CAM y el modelado paramétrico a campos emergentes de interacción y tecnologías responsables como la robótica.

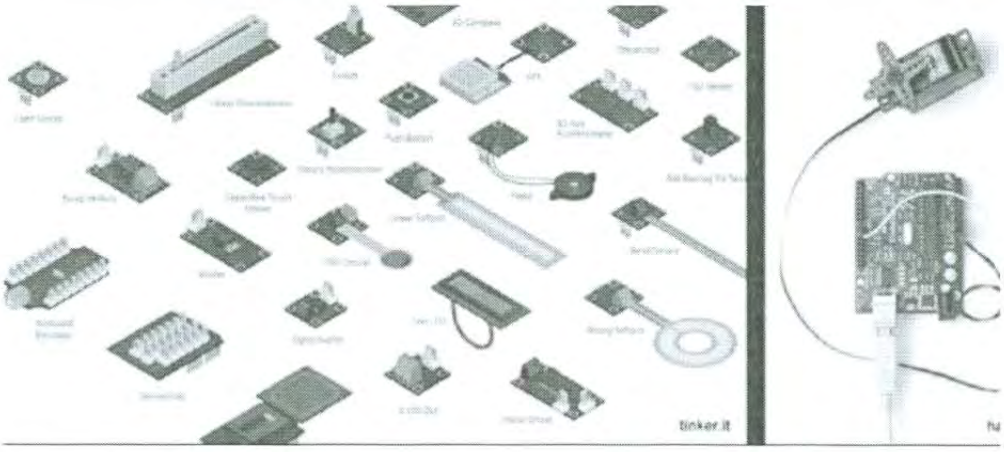


Figura 2.9 Diagrama de dispositivos disponibles para conectar con Arduino compatibles con Firefly.

Figura 2.10 Visualización y simulación de la geometría y deformación de materiales.

2.2.3 MyOpenLab

En [4] el autor nos dice que es un entorno orientado a la simulación y modelado de sistemas físicos, electrónicos y de control, con un amplio campo de aplicaciones.

Figura 2.11 Logo de MyOpenLab.

La aplicación está desarrollada en el lenguaje JAVA y por ello resulta portable a distintas plataformas. En el campo del modelado y simulación es muy interesante contar con una herramienta flexible que a partir de una amplia biblioteca, permita realizar modelos a base de conectar bloques funcionales.

La presentación de los resultados y/o el control de las simulaciones se hace mediante un potente conjunto de bloques de función de visualización y/o interacción capaz de manejar todo tipo de datos (analógicos, digitales, binarios, matrices, vectores, imágenes, sonidos), tiene bastantes aplicaciones, como son:

- Simulación de Circuitos Digitales.
- Simulación de Circuitos Analógicos.
- Simulación de Instrumentos.
- Simulación de Automatismos.
- Simulación de Fenómenos Físicos.
- Simulación de Robots.
- Control de Elementos Físicos mediante Interfaces.
- Tratamiento de Imágenes y Sonido.
- Operaciones de vectores y matrices 2D y 3D.

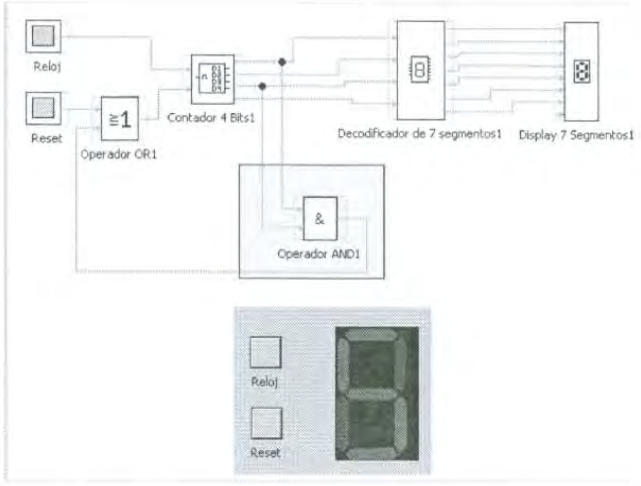


Figura 2.12 Simulación de circuitos digitales.

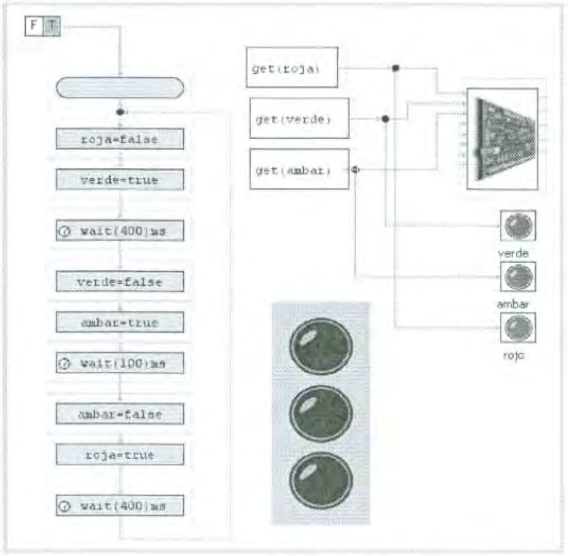


Figura 2.13 Ejemplo de programación (diagrama de flujo).

2.2.4 SA4 (Scratch)

Como nos dice [5], existe una aplicación de Citilab basada en Scratch para programar de manera gráfica Arduino. La aplicación se llama S4A (ficheros para descargar S4A y Firmware).

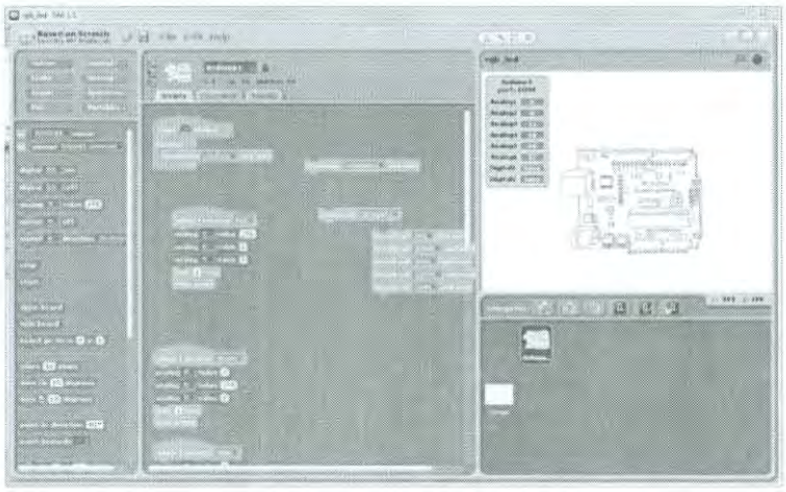


Figura 2.14 Pantalla de programación en S4A.

S4A es una modificación de Scratch que proporciona una programación sencilla de la plataforma abierta de hardware Arduino. Incluye nuevos bloques para controlar sensores y actuadores conectados a Arduino. También hay una tabla que informa del estado de los sensores similar a la PicoBoard, esta ha sido desarrollada para atraer a la gente al mundo de la programación. Su objetivo es también proporcionar una interfaz de nivel alto para programadores de Arduino con funcionalidades como la interacción de varias placas a través de eventos de usuario.

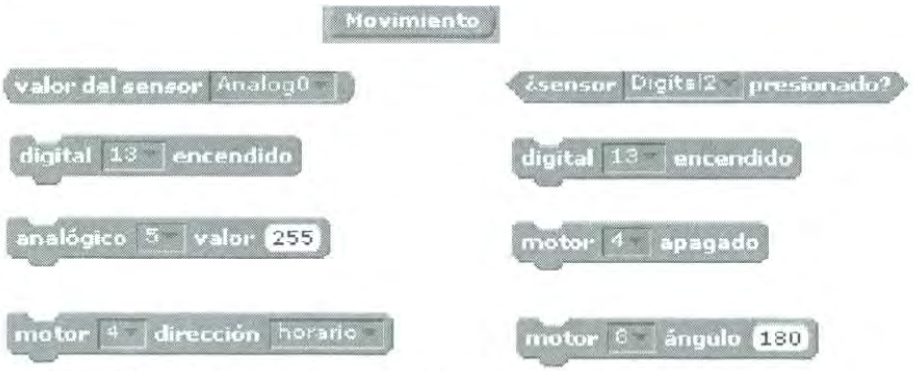


Figura 2.15 Funciones de gobierno de la Tarjeta Arduino UNO.

S4A interactúa con Arduino enviando el estado de los actuadores y recibiendo el de los sensores cada 75 ms, por lo tanto el ancho de pulso ha de ser mayor que este período. Este intercambio de información se efectúa usando el protocolo de la PicoBoard, el cual ya está implementado en un programa específico (llamado firmware) en la placa. Se pueden encontrar instrucciones de cómo cargarlo a través del entorno Arduino.

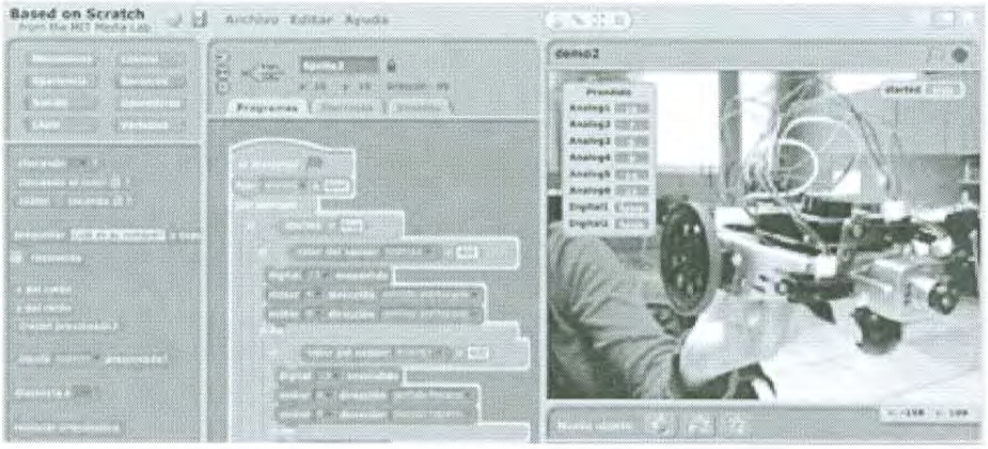


Figura 2.16 Programación y visualización de un proyecto en S4A.

2.3 Proyectos previos

Existen antecedentes de proyectos hechos con Arduino en conjunto con LabVIEW. Se trata de un diseño y desarrollo de un sistema de monitoreo y control para un invernadero de uso doméstico [6], el cual se basa en la medición, adquisición y control de parámetros como son la temperatura, humedad del suelo y luminosidad a partir de sensores, también incluye la visualización del muestreo de datos en gráficos en tiempo real.

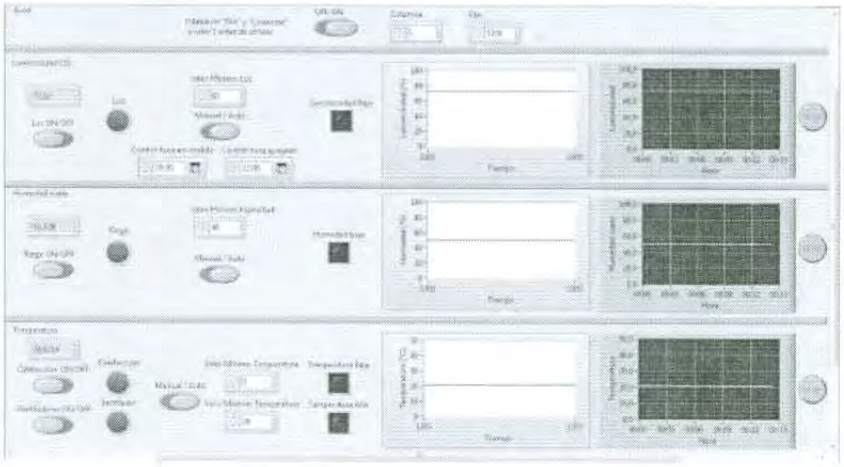


Figura 2.17 Panel frontal para interfaz LabVIEW – Arduino.

La figura 2.16 Muestra el panel frontal diseñado para este proyecto, en donde es posible modificar el valor mínimo de luminosidad, temperatura y humedad deseado, también contiene controles automáticos para el encendido y apagado de las luces y controles para modificar el rango de trabajo de la temperatura deseado.

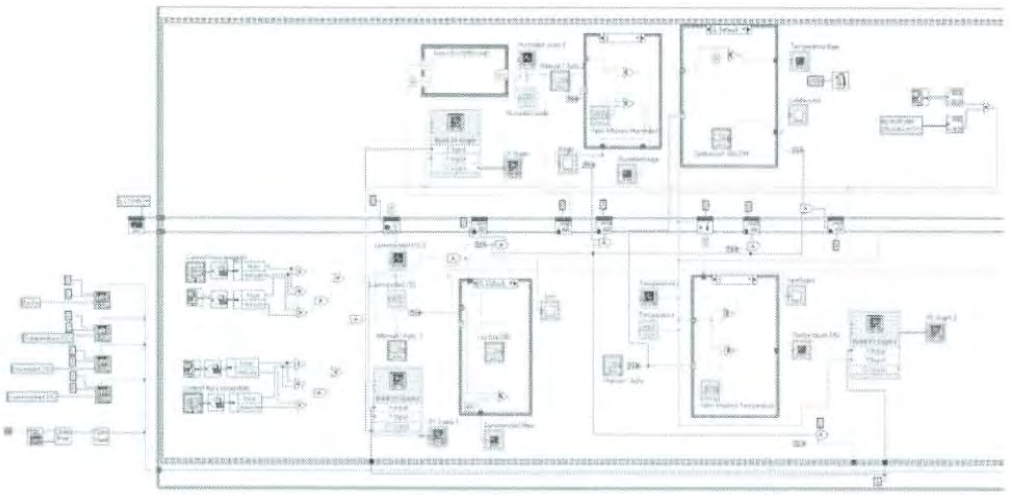


Figura 2.18 Primera etapa del programa desarrollado en LabVIEW.

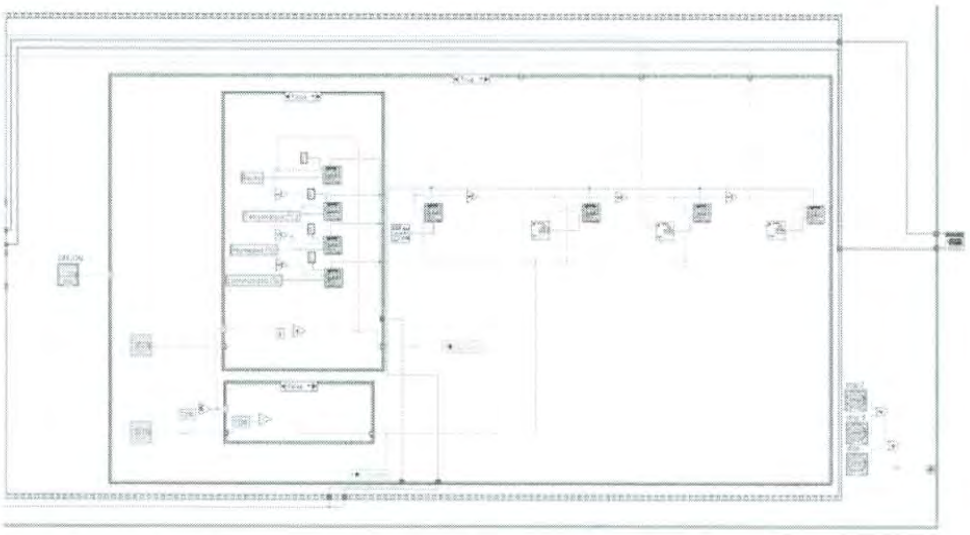


Figura 2.19 Segunda etapa del control de invernadero.

Capítulo 3

3. Marco Teórico.

3.1 Android

Android es un sistema operativo muy versátil empleado principalmente para dispositivos móviles. Basado en el Kernel de Linux, es un sistema libre, gratuito y multiplataforma, con gran capacidad de adaptación a todo tipo de dispositivos lo que le confiere un gran potencial de desarrollo [8].

Esta libertad facilita la labor a los desarrolladores ya que libera periódicamente su código y no tiene ningún costo añadido en licencias. Android tiene una serie de librerías en C/C++ y las aplicaciones se realizan principalmente en JAVA usando Dalvik, una adaptación de la máquina virtual de JAVA para dispositivos con poca memoria. El sistema operativo tiene una serie de API's para el uso de las distintas funciones del teléfono: GPS, giroscopio, etcétera.

3.2 LabVIEW

LabVIEW (abreviatura de Laboratorio Virtual Instrument Engineering Workbench) es una plataforma de sistema de diseño y entorno de desarrollo para un lenguaje de programación visual de National Instruments.

El lenguaje de programación utilizado en LabVIEW, también conocida como lenguaje G, es orientado al flujo de datos. La ejecución está determinada por la estructura de diagrama de

bloques gráfico (código fuente de LabVIEW) en el que el programador conecta diferentes nodos de funciones al dibujar cables [9]. Estos cables proponen variables y cualquier nodo puede ejecutar tan pronto como se disponga de todos los datos de entrada. Dado que se puede dar un caso de ejecución de múltiples nodos simultáneamente, el lenguaje G es intrínsecamente capaz de ejecutar funciones en paralelo. El hardware multi-procesos y multi-hilos se explota de forma automática por el planificador incorporado, que multiplexa varios hilos del sistema en los nodos listos para su ejecución.

LabVIEW vincula la creación de interfaces de usuario (llamados paneles frontales) en el ciclo de desarrollo. Los programas/subrutinas de LabVIEW se llaman instrumentos virtuales (VI's). Cada VI tiene tres componentes: diagrama de bloques, panel frontal y panel conector. El último se utiliza para representar el VI en los diagramas de bloques de otra, llamando VI's. El panel frontal está construido usando los controles e indicadores. Los controles son entradas – permiten al usuario suministrar información a la VI. Los indicadores son salidas – permiten visualizar, los resultados basados en las entradas dadas en el VI. El diagramas de bloques, contiene el código fuente gráfico. Todos los objetos situados en el panel frontal aparecerán en el diagrama de bloques como terminales. El diagrama de bloques contiene también las estructuras y funciones que realizan las operaciones sobre los controles y datos que proporcionan los indicadores. Las estructuras y funciones se encuentran en la paleta de funciones. Los controles, indicadores, estructuras y funciones, se conocen como nodos. Los nodos están conectados entre sí mediante cable.

Por lo tanto un instrumento virtual también se puede ejecutar como un programa, con el panel frontal que sirve como una interfaz de usuario, o cuando se utiliza como un nodo en el diagrama de bloques, el panel frontal define las entradas y salidas para el nodo dado a través del panel de conector. Esto implica cada VI puede ser fácilmente probado antes de ser incorporado como una subrutina en un programa más grande.

3.3 Motores de Corriente Directa

Los motores de corriente directa son máquinas eléctricas que transforman la energía eléctrica en energía mecánica. Impulsan dispositivos tales como ventiladores, bombas, prensas punzadoras y carros. Estos dispositivos pueden tener características de par o momento de torsión-velocidad muy definida (como una bomba o un ventilador) o una extremadamente variable (como un automóvil). A diferencia de los motores pasos a paso y los servomecanismos, los motores CD no pueden ser posicionados y/o enclavados en una posición específica. Estos simplemente giran a la máxima velocidad y en el sentido que la alimentación aplicada se los permite [10].

3.3.1 Partes de un motor CD

El motor de corriente continua está compuesto de dos piezas fundamentales, el rotor y el estator. El rotor constituye la parte móvil del motor, proporciona el torque para mover a la carga. Está formado por:

- Eje: Formado por una barra de acero fresada. Imparte la rotación al núcleo, devanado y al colector.
- Núcleo: Se localiza sobre el eje. Fabricado con capas laminadas de acero, su función es proporcionar un trayecto magnético entre los polos para que el flujo magnético del devanado circule.
- Las laminaciones tienen por objeto reducir las corrientes parásitas en el núcleo. El acero del núcleo debe ser capaz de mantener bajas las pérdidas por histéresis. Este núcleo laminado contiene ranuras a lo largo de su superficie para albergar al devanado de la armadura (bobinado).
- Devanado: Consta de bobinas aisladas entre sí y entre el núcleo de la armadura. Las bobinas están alojadas en las ranuras, y están conectadas eléctricamente con el colector, el cual debido a su movimiento rotatorio, proporciona un camino de conducción conmutado.
- Colector: Denominado también conmutador, está constituido de láminas de material conductor (delgas), separadas entre sí y del centro del eje por un material aislante, para evitar cortocircuito con dichos elementos. El colector se encuentra sobre uno de los extremos del eje del rotor, de modo que gira con éste y se encuentran en contacto con las

escobillas. La función del colector es recoger la tensión producida por el devanado inducido, transmitiéndola al circuito por medio de las escobillas (llamadas también cepillos).

Figura 3.1 Partes de un motor CD

Por otra parte el estator constituye la parte fija de la máquina. Su función es suministrar el flujo magnético que será usado por el bobinado del rotor para realizar su movimiento giratorio. Está formado por:

- Armazón: Denominado también yugo, tiene dos funciones primordiales: servir como soporte y proporcionar una trayectoria de retorno al flujo magnético del rotor y del imán permanente, para completar el circuito magnético.
- Imán permanente: Compuesto de material ferromagnético altamente remanente, se encuentra fijado al armazón o carcaza del estator. Su función es proporcionar un campo magnético uniforme al devanado del rotor o armadura, de modo que interactúe con el campo formado por el bobinado, y se origine el movimiento del rotor como resultado de la interacción de estos campos.
- Escobillas: Están fabricadas se carbón, y poseen una dureza menor que la del colector, para evitar que se desgaste rápidamente. Se encuentran albergadas por los portaescobillas. Ambos, escobillas y portaescobillas, se encuentran en una de las tapas del estator.

3.3.2 Tipos de motores

La característica de un par o momento de torsión-velocidad del motor deber ser adaptada al tipo de carga que tiene que impulsar, y este requerimiento ha dado lugar a tres tipos básicos de motores:

Motores en derivación (o Shunt)

Los devanados y el inductor están conectados en paralelo y alimentados por una fuente común. También se denominan máquinas shunt, y en ellas un aumento de la tensión en el inducido hace aumentar la velocidad de la máquina.

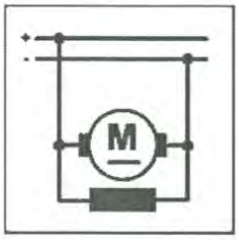


Figura 3.2 Diagrama de motor en derivación o Shunt

Motores en serie

Los devanados de inducido y el inductor están colocados en serie y alimentados por una misma fuente de tensión. En este tipo de motores existe dependencia entre el par y la velocidad; son motores en los que, al aumentar la corriente de excitación, se hace disminuir la velocidad, con un aumento del par.

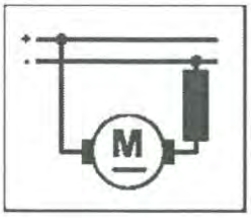


Figura 3.3 Diagrama de motor en serie

Motores compuestos

También llamados *compound*, en este caso el devanado de excitación tiene una parte de él en serie con el inducido y otra parte en paralelo. El embobinado en serie con el inducido está constituido por pocas espiras de gran sección, mientras que el otro está formado por un gran número de espiras de pequeña sección. Permite obtener por tanto un motor con las ventajas del motor serie, pero sin sus inconvenientes. Sus curvas características serán intermedias entre las que se obtienen con excitación serie y con excitación en derivación. Existen dos tipos de excitación compuesta. En la llamada compuesta adicional el sentido de la corriente que recorre los arrollamientos serie y paralelo es el mismo, por lo que sus efectos se suman, a diferencia

de la compuesta diferencial, donde el sentido de la corriente que recorre los arrollamientos tiene sentido contrario y por lo tanto los efectos de ambos devanados se restan.

Figura 3.4 Diagrama de motores compuestos

3.3.3 Aplicaciones

Aunque el precio de un motor de corriente continua es considerablemente mayor que el de un motor de inducción de igual potencia, existe una tendencia creciente a emplear motores de corriente continua en aplicaciones especiales. La gran variedad de la velocidad, junto con su fácil control y la gran flexibilidad de las características par velocidad del motor de corriente continua, han hecho que en los últimos años se emplee éste cada vez más con máquinas de velocidad variable en las que se necesite amplio margen de velocidad y control fino de las mismas. Existe un creciente número de procesos industriales que requieren una exactitud en su control o una gama de velocidades que no se puede conseguir con motores de corriente alterna. El motor de corriente continua mantiene un rendimiento alto en un amplio margen de velocidades, lo que junto con su alta capacidad de sobrecarga lo hace más apropiado que el de corriente alterna para muchas aplicaciones. Los motores de corriente continua empleados en juguetes, suelen ser del tipo de imán permanente, proporcionan potencias desde algunos vatios a cientos de volts.

Una última ventaja es la facilidad de inversión de marcha de los motores grandes con cargas de gran inercia, al mismo tiempo que devuelven energía a la línea actuando como generador, lo que ocasiona el frenado y la reducción de velocidad.

3.4 Arduino

Arduino es una plataforma de electrónica abierta para la creación de prototipos basada en software y hardware flexibles y fáciles de usar. Se creó para artistas, diseñadores, aficionados y cualquiera interesado en crear entornos u objetos interactivos. [11]

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

Las placas pueden ser hechas a mano o compradas montadas de fábrica; el software puede ser descargado de forma gratuita.

Es una plataforma de desarrollo de computación física de código abierto, basada en una placa con un sencillo microcontrolador y un entorno de desarrollo para crear software embebido.

Los proyectos de Arduino pueden ser autónomos o comunicarse con un software que se ejecute en la computadora.

3.4.1 Ventajas de Arduino respecto a otros sistemas

- Accesible - Las placas son más accesibles comparadas con otras plataformas de microcontroladores. La versión más costosa de un módulo de Arduino puede ser montada a mano, e incluso ya montada cuesta bastante menos.

- Multi-Plataforma - El software funciona en los sistemas operativos *Windows, Macintosh OSX y Linux*. La mayoría de los entornos para microcontroladores están limitados a *Windows*.

- Entorno de programación simple y directo - El entorno de programación es fácil de usar para principiantes y lo suficientemente flexible para los usuarios avanzados. Pensando en los profesores, Arduino está basado en el entorno de programación de *Procesing* con lo

que el estudiante que aprenda a programar en este entorno se sentirá familiarizado con el entorno de desarrollo Arduino.

- Software ampliable y de código abierto - El software esta publicado bajo una licencia libre y está preparado para ser ampliado por programadores experimentados. El lenguaje puede ampliarse a través de librerías de *C++*, y si se está interesado en profundizar en los detalles técnicos, se puede dar el salto a la programación en el lenguaje *AVR C* en el que está basado. De igual modo se puede añadir directamente código en *AVR C* en los programas si así se requiere.

- Hardware ampliable y de Código abierto - Arduino está basado en los microcontroladores *ATMEGA168, ATMEGA328 y ATMEGA1280.* Los diseños de los módulos están publicados bajo licencia *Creative Commons*, por lo que diseñadores de circuitos con experiencia pueden hacer su propia versión del módulo, ampliándolo u optimizándolo. Incluso usuarios relativamente inexpertos pueden construir la versión para placa de desarrollo para entender cómo funciona y ahorrar algo de dinero.

3.4.2 Arduino UNO

Arduino UNO (como se muestra en la Figura 3.5) es una placa electrónica basada en el microcontrolador *ATmega328*. Cuenta con 14 entradas digitales / pines de salida (de los cuales 6 pueden ser utilizados como salidas *PWM*), 6 entradas analógicas, un oscilador de cristal de *16 MHz*, una conexión *USB*, un conector de alimentación, una cabecera de *ICSP* (*In Circuit Serial Programming* método de programación directamente *AVR*), y un botón de reset. Contiene todos los periféricos necesarios para hacer funcionar el microcontrolador, solo tiene que conectarlo a una computadora con un cable *USB* o con un adaptador *AC-DC* o la batería para empezar.

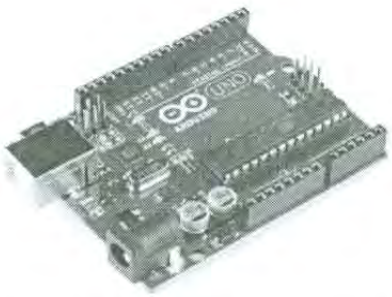


Figura 3.5 Placa de Arduino UNO

Características Técnicas de Arduino UNO

- Microcontrolador *ATmega328*
- Voltaje de Operación 5V
- Voltaje de Entrada (recomendado) 7-12V
- Voltaje de Entrada (límites) 6-20V
- Canales de E / S (de los cuales 6 proporcionar una salida *PWM*)
- Pines de entrada analógica 6
- Corriente de E / S de CC Pin 40 mA
- De corriente continua de 3.3V Pin 50 mA
- Memoria Flash de 32 MB (*ATmega328*) de los cuales 0,5 KB utilizado por gestor de arranque
- SRAM 2 KB (*ATmega328*)
- EEPROM 1 KB (*ATmega328*)
- Velocidad del reloj de *16 MHz*.

Puede ser alimentado a través de la conexión *USB* o con una fuente de alimentación externa. La fuente de alimentación se selecciona automáticamente.

- Externo (no *USB*), la fuente de alimentación puede venir de un adaptador de *CA* a *CC* o una batería. Los cables de la batería se pueden insertar en los pines *GND* y *VIN* del conector de alimentación.
- La fuente puede operar en un suministro externo de 6 a 20 voltios. Si se suministran con menos de 7V, sin embargo, el adaptador puede suministrar menos de cinco voltios y la placa puede ser inestable. Si utiliza más de 12V, el regulador de voltaje se puede sobrecalentar y dañar la placa. El rango recomendado es de 7 a 12 voltios.

Los pines de alimentación son los siguientes:

- **VIN** - El voltaje de entrada a la placa Arduino cuando se utiliza una fuente de alimentación externa (a diferencia de 5 voltios de la conexión *USB* o de otra fuente de alimentación regulada). Se puede suministrar tensión a través de este pin.
- **5V** - Este pin genera una 5V regulados por el regulador en la tarjeta. La tarjeta puede ser alimentada ya sea desde la entrada de alimentación (7 - 12 V), el conector *USB* (5V), o el pin de *VIN* de la placa (7-12V). El suministro de tensión a través de los pines de 5V o 3.3V no pasa por el regulador, y puede dañar la placa. NO ACONSEJABLE.
- **3.3V** - Un suministro voltios 3,3 generada por el regulador. El consumo de corriente máxima es de 50 mA.
- **GND -** Pin de tierra.

Memoria

El ATmega328 tiene 32 MB (con 0,5 KB utilizados para el gestor de arranque). También dispone de 2 KB de *SRAM* y 1 KB de memoria *EEPROM* (que puede ser leído y escrito con la librería *EEPROM*).

Entrada y salida

Cada uno de los 14 pines digitales en el Arduino UNO se puede utilizar como entrada o salida, usando las funciones *pinMode* (), *digitalWrite* (), y las funciones de *digitalRead* (). Ellos operan a 5 voltios. Cada pin puede proporcionar o recibir un máximo de 40 mA y tiene un arreglo interno de resistencia *pull-up* (desconectado por defecto) de 20 a 50 k□. Además, algunos pines tienen funciones especializadas:

- Serie 0 (*RX*) y 1 (*TX*) - Se utiliza para recibir (*RX*) y transmisión (*TX*) datos serie *TTL*. Estos se encuentran conectados a los pines correspondientes de la ATmega8U2 USB a chip de serie *TTL*.
- Interrupciones externas 2 y 3 - Estas patillas se pueden configurar para desencadenar una interrupción en un valor bajo, un borde ascendente o descendente, o un cambio en el valor.
- *PWM* 3, 5, 6, 9, 10 y 11 - Proporcionar 8-bit de salida *PWM* con la función *analogWrite* ().

- LED13 - Hay un *LED* conectado al pin digital 13, el cual funciona como un indicador de propósito general. Cuando el pin es de alto valor, el *LED* está encendido, cuando el pasador es bajo, es apagado.

El Arduino UNO tiene seis entradas analógicas, etiquetados A0 a A5, cada una de las cuales proporcionan 10 bits de resolución (es decir 1024 valores diferentes). Por defecto miden desde el 0 a 5 voltios, aunque es posible cambiar el extremo superior de su rango con el pin y el *AREF* y la función *analogReference* ().

Comunicación

El Arduino tiene una serie de facilidades para comunicarse con una computadora, otro Arduino, u otros microcontroladores. El *ATmega328* ofrece una *UART TTL* (5V) de comunicación en serie, que está disponible en los pines digitales 0 (*RX*) y 1 (*TX*). Un *ATmega16U2* en los canales de comunicación a través de *USB* y aparece como un puerto *COM* virtual con el software en la computadora. El *firmware 16U2* utiliza el estándar de los controladores *USB, COM*, y no es necesario ningún controlador externo. Sin embargo, en *Windows*, un archivo ".inf" es requerido. El software de Arduino incluye un monitor de datos (*Data Monitor*) que permite el monitoreo de datos que se envía desde y hacia la placa. Los *LEDs RX* y *TX* en la tarjeta parpadean cuando se están transmitiendo datos a través del puerto *USB* a serie y la conexión *USB* a la computadora (pero no para la comunicación de serie en los pines 0 y 1). La biblioteca *SoftwareSerial* permite la comunicación de serie en cualquiera de los pines digitales de Arduino UNO. El *ATmega328* también soporta comunicación *I2C* (bus de comunicaciones en serie) y *SPI*. El software incluye una librería *Wire* para simplificar el uso del bus *I2C*. Para la comunicación *SPI*, utilizar la biblioteca de *SPI*.

Programación

La placa puede ser programada bajo el entorno de programación Arduino. El *ATmega328* viene pre-quemado con un gestor de arranque que le permite cargar nuevo código a la placa sin el uso de un programador de hardware externo.

También puede pasar por alto el gestor de arranque y el programa del microcontrolador a través de la *ICSP* (programación *In-Circuit Serial*). El *ATmega16U2* (o *8U2* en los *REV1* y *REV2*) el código fuente está disponible en el *firmware*.

Reset

En lugar de requerir presionar el botón de reset antes de una carga, el Arduino está diseñado de una manera que le permite ser restaurado mediante el software que se ejecuta en una computadora conectada. Una de las líneas de control de flujo de hardware (*DTR*) de la *ATmega8U2/16U2* está conectado a la línea de reposición del *ATmega328* través de un condensador 100 *nanofaradios*. Cuando ésta toma un valor bajo, el voltaje suministrado de la línea de reset cae lo suficiente como para restablecer el chip. El software de Arduino utiliza esta capacidad que le permite cargar el código con solo pulsar el botón de carga en el entorno Arduino.

USB Protección contra sobrecorriente

Arduino UNO tiene un fusible reajustable que protege a los puertos *USB* de la computadora de pequeños cortos y de sobrecorriente. Aunque la mayoría de las computadoras ofrecen su protección interna, el fusible proporciona una capa adicional de protección. Si hay más de 500 mA aplicados al puerto *USB*, el fusible automáticamente corta la conexión hasta que el cortocircuito o una sobrecarga se han eliminado.

Dimensiones físicas de Arduino UNO

Figura 3.6 Dimensiones placa Arduino UNO.

3.4.3 Experimentos

El propósito de esta sección es familiarizarse con el entorno de hardware y software Arduino y conocer sus herramientas de programación. Se describen solo algunos de los experimentos realizados para ilustrar la metodología empleada.

Experimento 1.- Intermitente

Encender y a pagar un *LED* que se conecta en el *PIN* 13 de Arduino y se configura como salida. El tiempo de encendido y apagado es de 1 segundo.

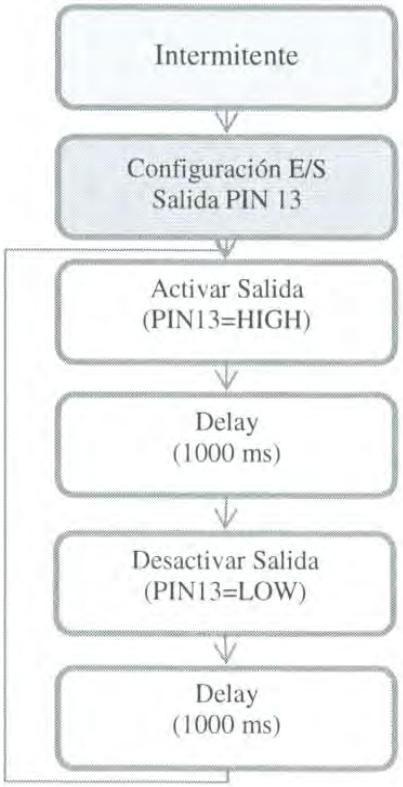


Figura 3.7 Diagrama pseudocódigo de experimento 1

Conexión física en Arduino

Solo se requiere de un led para la salida, para el Arduino UNO es opcional ya que cuenta con un *LED* incorporado en el *PIN 13*.

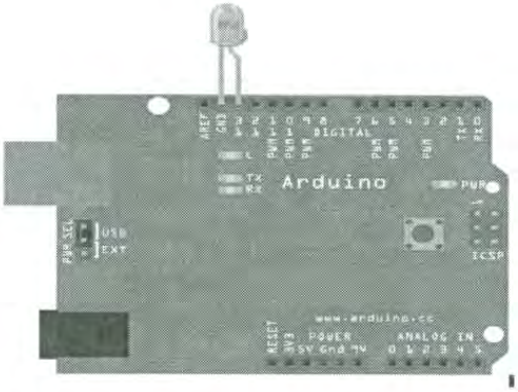


Figura 3.8 Conexión de componentes para experimento 1

Experimento 2.- Lectura de un pulsador

Encender y apagar un *LED* conectado en el *PIN* 13 de Arduino y se configura como salida, un botón pulsador en el *PIN* 10 y se configura como entrada. El nivel del *LED* será controlado mediante un botón pulsador conectado al *PIN* 10.

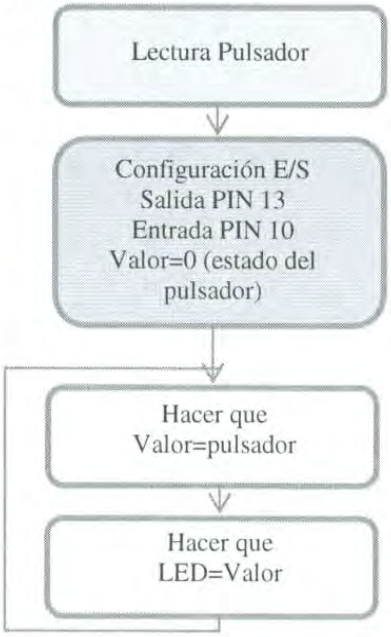


Figura 3.9 Diagrama pseudocódigo de practica 2

Conexión física de Arduino

Solo se requiere de un led para la salida, para el Arduino UNO es opcional ya que cuenta con un *LED* incorporado en el *PIN* 13. Además de un botón pulsador conectado al *PIN* 10.

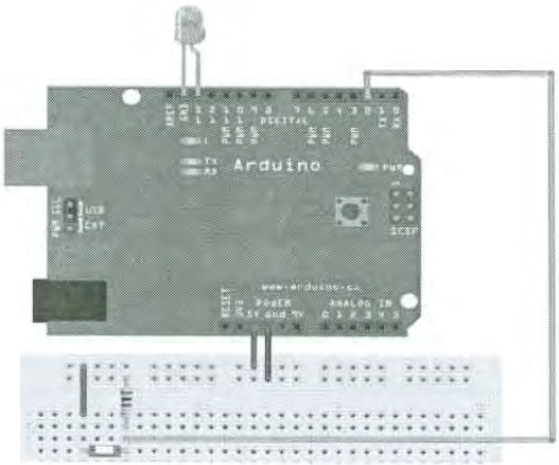


Figura 3.10 Conexión de componentes para práctica 2

3.5 Modulación por Ancho de Pulso (PWM)

Como nos dice [12], el PWM se basa en la comparación de una señal de referencia a modular y una señal portadora de forma triangular o diente de sierra, la comparación generará un tren de pulsos, de ancho específico, que puede ser utilizada como reguladora de entrega de energía. La relación entre la amplitud de la señal portadora y la señal de referencia se llama índice de modulación y se denomina con "m_a", el índice de modulación permite obtener tensión variable. Este tipo de señal puede ser generado muy fácilmente por ARDUINO, ya que algunas de sus salidas digitales pueden ser utilizadas como PWM para controlar motores de CD.

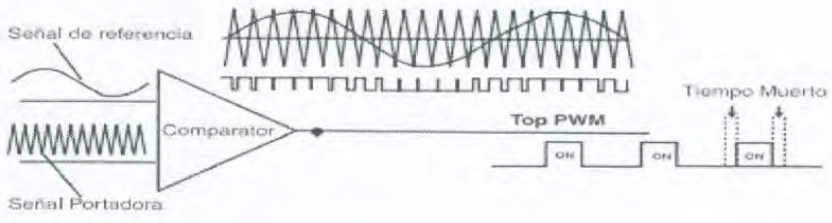


Figura 3.11 Definición grafica de una señal PWM.

Existe otro campo de la ingeniería necesario para el desarrollo del controlador, es el campo de la programación, por lo que este proyecto fue diseñado en una estructura de máquina de estados, dicha estructura es posible crearla en la plataforma de desarrollo LabVIEW.

3.6 Maquina de estados

Las máquinas de estado son una parte integral de la programación. Las máquinas de estado hacen al código más eficiente, más fácil de depurar y ayudan a organizar el flujo del programa.

Como nos dice [13], la máquina de estados es un común y muy útil diseño. Puede utilizarse para implementar cualquier algoritmo que pueda ser explícitamente descrito por un diagrama de estados o un diagrama de flujo. Una máquina de estados usualmente implementa un moderadamente complejo algoritmo toma-decisión, tal como un diagnostico rutina o un proceso de monitoreo.

Una máquina de estados, la cual es más precisamente definida como una máquina de estados finitos, consiste en un conjunto de estados y una función de transición que se relaciona al siguiente estado. Dicha maquina tiene muchas variaciones, dos de las más comunes son la máquina de Mealy y la máquina de Moore. Una máquina de Mealy desarrolla una acción por cada transición. Una máquina de Moore desarrolla una acción específica por cada estado en el diagrama de transición de estados. La plantilla de máquina de estados diseñada en LabVIEW implementa cualquier algoritmo descrito por la máquina de Moore.

La máquina de estados se usa en aplicaciones donde distinguiblemente los estados existen. Cada estado puede conducir a uno o varios estados o al fin del proceso. Una máquina de estados se basa en entradas hechas por el usuario o cálculos hechos en el estado, para determinar cuál estado es el siguiente. Muchas aplicaciones requieren un estado de inicialización, seguido de un estado *default,* donde diferentes acciones pueden ser desarrolladas. Las acciones desarrolladas pueden depender de entradas (previas y actuales) y estados.

Las máquinas de estados son comúnmente usadas para crear interfaces de usuario, donde diferentes acciones envían a la interfaz dentro de diferentes segmentos de procesamiento. Cada

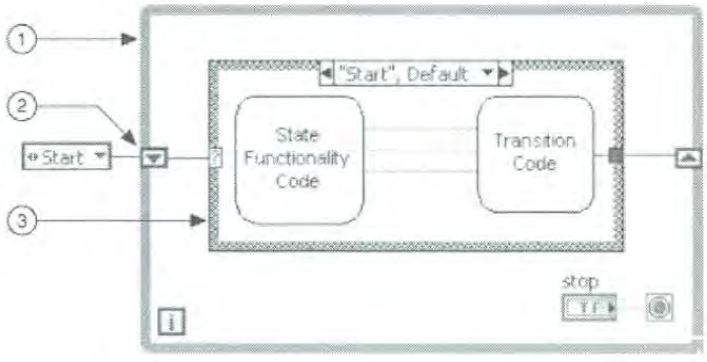
segmento de procesamiento actúa como un estado, cada segmento puede conducir a otro segmento para su posterior procesamiento o esperar otra acción del usuario. En una aplicación de interfaz de usuario, la máquina de estados constantemente monitorea al usuario para la siguiente acción a tomar.

El proceso de prueba es otra aplicación común en la máquina de estados, en donde un estado representa cada segmento del proceso. Dependiendo del resultado de cada prueba de estado, algún diferente estado puede ser llamado. Esto puede pasar continuamente, resultando en un análisis a fondo del proceso que se está probando. La ventaja de usar una máquina de estados es que después de haber creado un diagrama de estados de transición, tu diseñar en LabVIEW.

Traduciendo el diagrama de estados de transición en un diagrama de bloques en LabVIEW, requerimos la siguiente infraestructura.

- *While Loop:* Continuamente está ejecutando varios estados.
- *Case Structure:* Contiene un caso por cada estado en el código a ejecutar por cada estado.
- *Shift Register:* Contiene infotmacion de la transición de estado.
- *State Functionally Code:* Implementa la función del estado.
- *Transition Code:* Determina el siguiente estado en la secuencia.

A continuación se muestra la estructura básica de una máquina de estados en LabVIEW.



1 While Loop 2 Shift Register 3 Case Structure

Figura 3.12 Estructura básica de una máquina de estados en LabVIEW.

El mejor método para controlar la inicialización y transición de una máquina de estados es el *enumerated type control* (*enums*). Los *enums* son extensamente usados como un selector de casos.

Existen muchas maneras para controlar que caso en un *case structure* se ejecute en una máquina de estados. Debemos seleccionar el método que mejor se acomode a las funciones y complejidad de la máquina de estados. Uno de los métodos de transición más usados es el sencillo *Case structure transition code*, el cual puede ser usado para cambiar entre cualquier número de estados. Para una transición *default* no se necesita código para determinar el siguiente estado, porque hay solo un posible estado que provoque el siguiente estado.

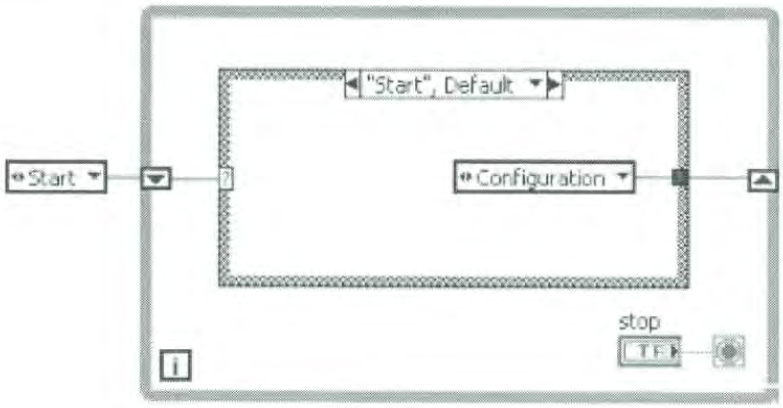


Figura 3.13 Esquema de la transición *default.*

Cuando existe la posibilidad de que un mismo estado nos mande hacia dos posibles estados siguientes, usualmente se usa una estructura como la que se muestra a continuación.

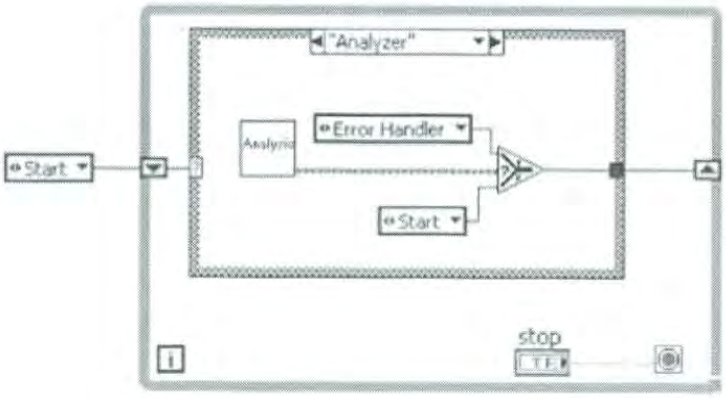


Figura 3.14 Esquema de una transición a dos posibles estados siguientes.

Es un diseño de máquina de estados se utilizan variables que nos ayudan a condicionar, controlar y conducir nuestros estados de manera que no generen conflicto entre ellos, tal herramienta utilizada para este control son las variables de estado. Estas son variables que no intervienen en desarrollo del programa, sino que son variables que existen para poder generar condicionantes en los estados, para así delimitar el orden del llamado de los estados.

Capítulo 4

Integración de las plataformas Android-LabVIEW-Arduino para el control de motores de corriente directa

Como se ha mencionado, el objetivo general es la integración de tres plataformas: el sistema operativo Android que, a través del *Smartphone* nos permite enviar los datos de posición y velocidad a través de bluetooth, los cuales serán recibidos por el software de programación LabVIEW. El trabajo de LabVIEW consiste en analizar y procesar los datos enviados por el *Smartphone* para encontrar los parámetros que ayuden en la velocidad y orientación del automóvil simulado. Al tener los datos procesados, estos serán enviados a través de un cable USB que conectara al hardware Arduino donde nuevamente serán procesados para enviar la orden final a los motores de corriente directa, para lo cual es necesario contar con una etapa de potencia, que será adecuada para recibir la señal del PWM que simulará la velocidad de los motores y dos salidas digitales que conmutarán para la orientación del giro del motor y así poder finalizar el sistema ciber-fisico.

Al obtener una comunicación exitosa entre las tres plataformas Android-LabVIEW-Arduino, se dará base para utilizar la misma técnica y/o utilizar otros componentes a simular en nuestro automóvil.

4.1 Comunicación entre Android y LabVIEW

Para poder adquirir datos de nuestro *Smartphone* es necesario tener una aplicación que esta adecuada en la lectura y envío de datos, para esto se utilizó la aplicación AndroVIEW; como bien sabemos el sistema operativo Android es a base de Linux, que este es de fuente abierta y es fácil para los programadores adquirir el código y trabajar sobre él, con el fin de obtener una mejoría y/o acceso al hardware en el que esta implementado.

En AndroVIEW se utilizó la herramienta *orientación* que este hace que el giroscopio encontrado en el *Smartphone* pueda ejecutar la posición de los ejes *x, y, z* actual leída por la aplicación y así poder ser enviada a través de bluetooth al ordenador donde se encuentra LabVIEW

4.1.1 Librería Android para LabVIEW

Las librerías que se le pueden cargar a LabVIEW son muy versátil, una de estas es la librería de comunicación mediante bluetooth con el sistema operativo Android; con este paquete, se pueden adquirir los datos generados por el Smartphone ya sea, aceleración, luz, campo magnético, orientación y temperatura.

En este trabajo se utilizaron los datos de lectura, orientación, con el fin de saber la posición del *Smartphone* actual leída.

Para realizar la comunicación y adquirir los datos generados por el móvil, se requiere los nodos: nombre y dirección bluetooth, inicializar, datos, finalizar; el nombre y dirección de bluetooth es en la cual está configurado el ordenador que se encuentra el software LabVIEW, este debe ser previamente asociado con el *Smartphone* en el cual está la aplicación AndroVIEW. La inicialización da comienzo la comunicación entre ambos dispositivos. En la adquisición de los datos se declara dentro del ciclo *while loop* para que sea una comunicación continua, aquí se deben especificar qué quieres recibir del SO Android, en el caso de este

4.1 Comunicación entre Android y LabVIEW

Para poder adquirir datos de nuestro *Smartphone* es necesario tener una aplicación que esta adecuada en la lectura y envío de datos, para esto se utilizó la aplicación AndroVIEW; como bien sabemos el sistema operativo Android es a base de Linux, que este es de fuente abierta y es fácil para los programadores adquirir el código y trabajar sobre él, con el fin de obtener una mejoría y/o acceso al hardware en el que esta implementado.

En AndroVIEW se utilizó la herramienta *orientación* que este hace que el giroscopio encontrado en el *Smartphone* pueda ejecutar la posición de los ejes *x, y, z* actual leída por la aplicación y así poder ser enviada a través de bluetooth al ordenador donde se encuentra LabVIEW

4.1.1 Librería Android para LabVIEW

Las librerías que se le pueden cargar a LabVIEW son muy versátil, una de estas es la librería de comunicación mediante bluetooth con el sistema operativo Android; con este paquete, se pueden adquirir los datos generados por el Smartphone ya sea, aceleración, luz, campo magnético, orientación y temperatura.

En este trabajo se utilizaron los datos de lectura, orientación, con el fin de saber la posición del *Smartphone* actual leída.

Para realizar la comunicación y adquirir los datos generados por el móvil, se requiere los nodos: nombre y dirección bluetooth, inicializar, datos, finalizar; el nombre y dirección de bluetooth es en la cual está configurado el ordenador que se encuentra el software LabVIEW, este debe ser previamente asociado con el *Smartphone* en el cual está la aplicación AndroVIEW. La inicialización da comienzo la comunicación entre ambos dispositivos. En la adquisición de los datos se declara dentro del ciclo *while loop* para que sea una comunicación continua, aquí se deben especificar qué quieres recibir del SO Android, en el caso de este

trabajo, se utilizó orientación y campo magnético; ambos nodos nos facilitara la posición y velocidad en los ejes *x, y, z* con respecto al campo de la tierra.

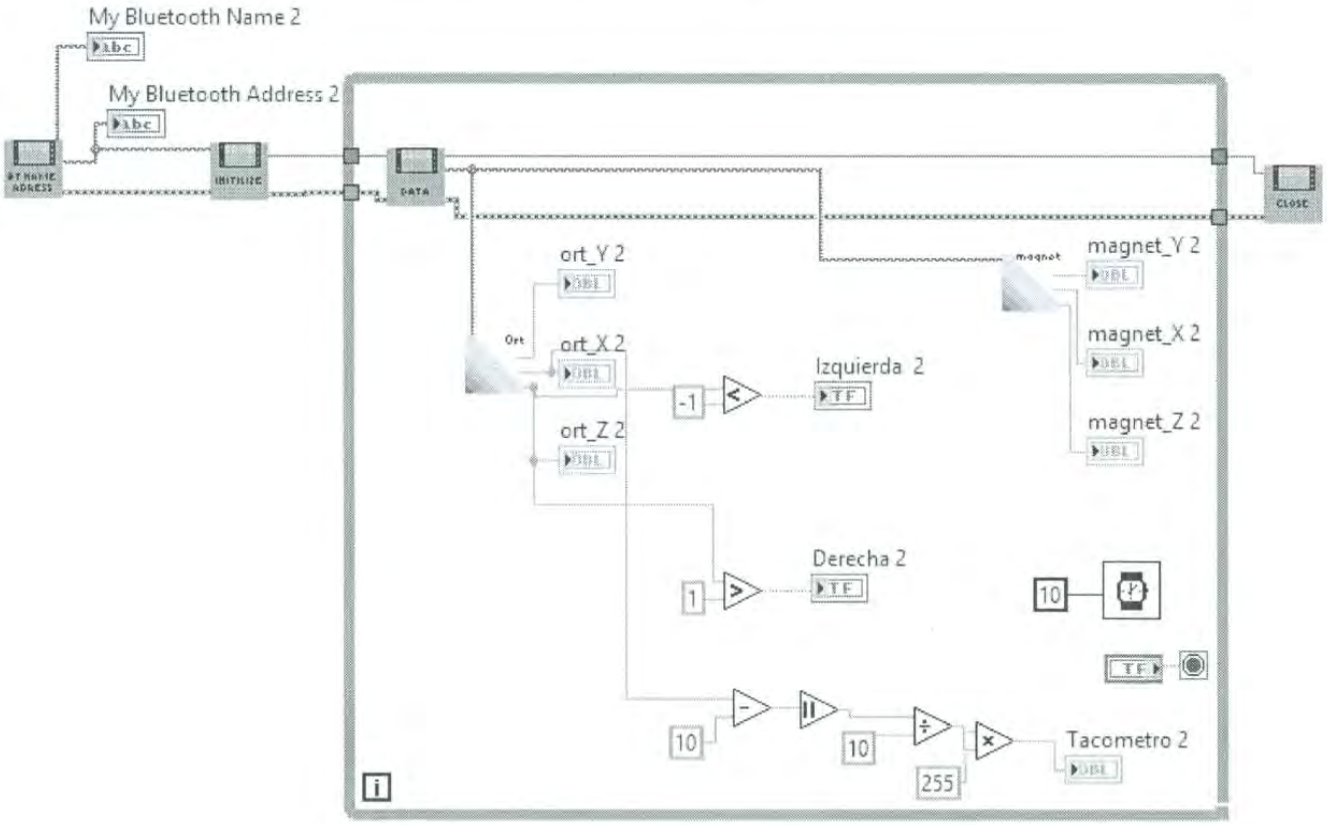


Figura 4.1 Diagrama de bloques para comunicación Android-LabVIEW

4.2 Procesamiento de la información en LabVIEW

La orientación del *Smartphone* que se utilizo fue horizontal, a partir de este punto nuestra velocidad y posición es cero; para la velocidad 90° es igual a 0 km/hr (simulado), lo cual si se van disminuyendo los grados de 90° a cero, esto indicara que la velocidad aumentara – como si fuera el pedal de aceleración.

Para obtener la velocidad el dado adquirido por Android el eje de orientación *x* fue el encargado de enviar la información con un rango de posición de 90° a 0°, donde la velocidad marcada como cero es nuestra posición del móvil en 90°; conforme estemos disminuyendo la posición del móvil, este avanzara a mayor velocidad, para ello se utilizó la ecuación (4.1).

$$\frac{|posX - 10|}{10} x100 = vel \tag{4.1}$$

La ecuación (4.1) fue utilizada en el diagrama de bloques para poder ser procesada y devolverla al panel frontal.

Para dar posición se tomó el eje orientación *z*, para mostrar la posición y sentido del motor, que este va de un rango de -10° a 10°, donde 0° ± 1° es la posición neutral (cero) del motor; para dar el sentido izquierdo se girara el *Smartphone* y el giroscopio brindara un rango de -10° a -1°, con el sentido derecho el dato mostrado fue un rango de 1° a 10°.

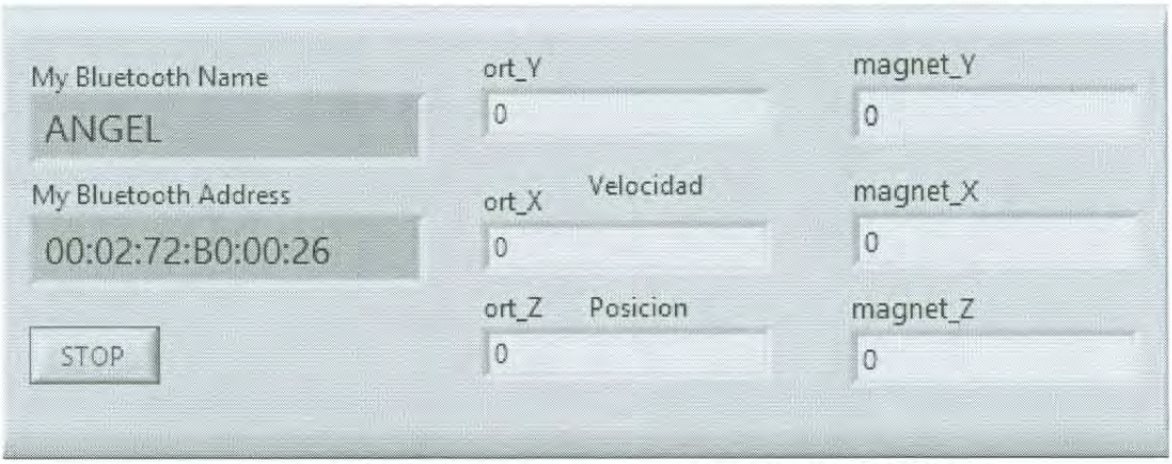


Figura 4.2 Panel Frontal de comunicación Android-LabVIEW

Los datos procesados por LabVIEW pasaran a Arduino, que este tomara el control final a los actuadores.

4.3 Comunicación entre LabVIEW y Arduino

Para realizar esta comunicación se utilizaron la versión 1.5 de Arduino y la versión 2010 de LabVIEW, también debemos tener físicamente una tarjeta de desarrollo Arduino, como puede ser Arduino UNO, Arduino Mega 2560, Arduino Duamilanove, o con controlador Atmega 328 y un cable USB/Serial.

Posteriormente necesitamos descargar e instalar el *toolkit "NI LabVIEW Interface for Arduino Toolkit"* (se puede descargar de la página oficial de *National Instrumens*).

Después de haber instalado el *toolkit* necesario, proseguimos con cargar un *Firmware* en Arduino, este *firmware* se localiza en la carpeta del directorio donde se han instalado los componentes del *toolkit* de LabVIEW, dicha carpeta se llama *"LVIFA_Base"*, la podemos encontrar en "Archivos de Programa\National Instruments\LabVIEW 2011\vi.lib\LabVIEW Interface for Arduino\Firmware", dicho *firmware* se carga a Arduino como si fuera un *sketch* cualquiera. Después se conecta el cable serial a la computadora, al conectar el cable nos indicará el puerto COM que se utilizará. Teniendo el cable conectado tanto al Arduino como a la PC y el firmware cargado a Arduino, se necesita configurar la tarjeta Arduino en LabVIEW.

4.3.1 Configuración de Arduino en LabVIEW

Después de los datos procesados en LabVIEW, este mismo se encargara de mandarlos a Arduino; para esto se utiliza el nodo de *inicialización* que este se configura el puerto COM a la que estará conectado el ordenador con el hardware Arduino, se define qué tipo de tarjeta se utilizará (Arduino UNO), los baudios que representan el número de señales por segundo en un medio de la transmisión digital (normalmente 9600) y el tipo de conexión que esta puede ser USB o Serial, en este caso se utilizó USB.

El siguiente nodo es *Configure Servo*, que aquí se le asigna un número al servomotor en el que se le enviaran los datos y a que salida de la tarjeta Arduino corresponde. Se crea un ciclo *while loop*, para mantener una comunicación continua, los nodos utilizados dentro del *while loop* fueron: *PWM Write Pin* y *Servo Write Angle;* en el nodo *PWM Write Pin* que este es utilizado para mandar la velocidad a nuestro motor, solo se requiere el *Ciclo de Trabajo* que este es la relación que existe entre el tiempo en que la señal se encuentra en estado activo y el periodo de la misma.

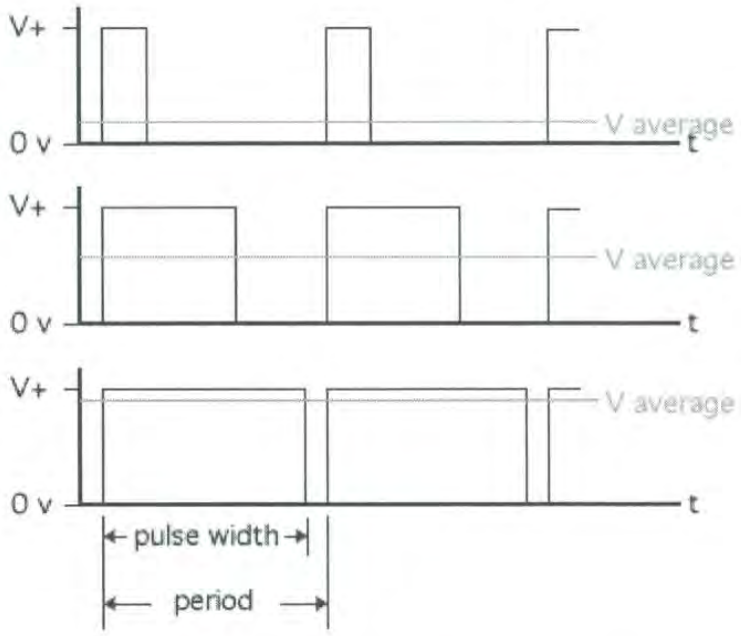


Figura 4.3 Variación del ciclo de trabajo.

$$D = \frac{T}{P} x100 \tag{4.2}$$

Donde:

D = Ciclo de trabajo

T = Ancho de pulso

P = Periodo

El *Ciclo de Trabajo* con el fin de adquirir la velocidad del motor será manipulado por el eje de orientación *x* envidada por el giroscopio encontrado en el *Smartphone*, que al hacer el movimiento de 90° a 0°, se le estará enviando el voltaje promedio de la señal activa. En este mismo nodo se declara a que pin será escrito/enviado el PWM, en este caso es el pin 3.

Para el sentido de giro del motor, se utilizaron los nodos *Servo Write Angle* que este solo declarara el ángulo de posición leído por el eje de orientación *z* que como se ha mencionado, el ángulo brindado será de -10° a + 10°, para poder simular el volante que tomara dirección al girar el motor.

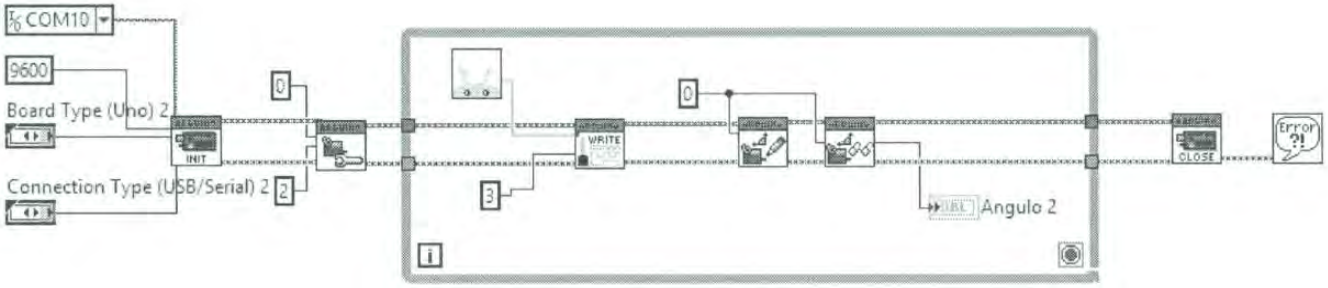


Figura 4.4 Configuracion Arduino en LabVIEW.

4.4 Código G de LabVIEW

Para poder llegar a un procesamiento de la información adecuada, ambas partes, ya sea recibir los datos, procesar y enviar, deben de trabajar dentro de un *while loop*; con esto en conjunto LabVIEW servirá como intermediario de ambas plataformas, *Android-Arduino* con el fin de recrear el sistema ciber-fisico mencionado.

En el código, para obtener visualización de que se están recibiendo los datos, se ajuntaron dos nodos booleanos que nos indicaran en qué sentido esta la posición del eje de orientación *z*, como este eje en nuestro *Smartphone* tiene un rango de ±10° siendo multiplicado por 9 para que el indicador de ángulo, pueda abarcar las direcciones de -90° a 90°, siendo 0° la dirección neutra.

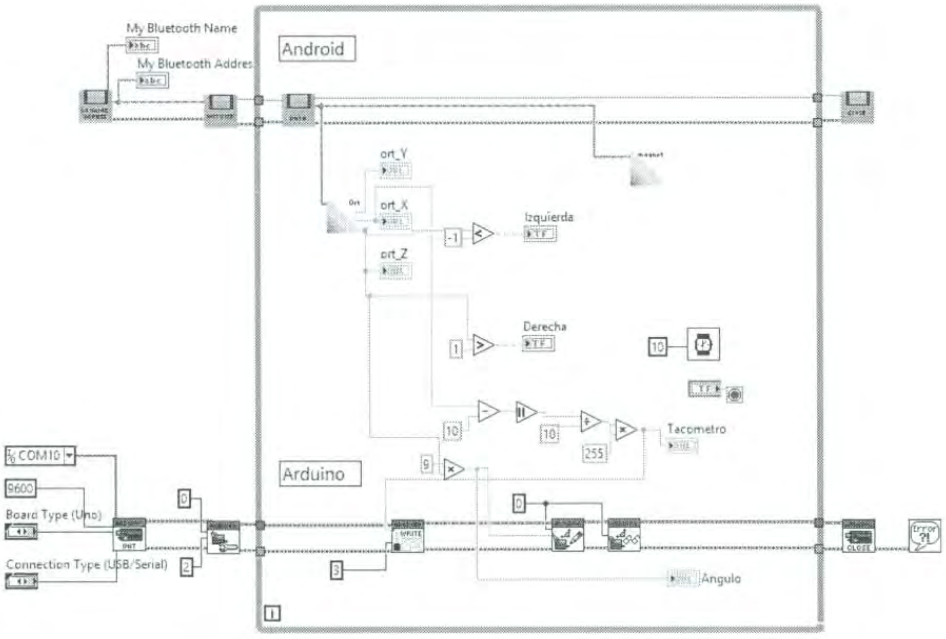


Figura 4.5 Codigo G del procesamiento de los datos.

4.5 Interfaz gráfica para lectura de sensores y control

En la siguiente imagen se muestra el panel frontal de LabVIEW, que esta mostrara los datos recibidos por Android, y lo que se está enviando a escribir en Arduino.

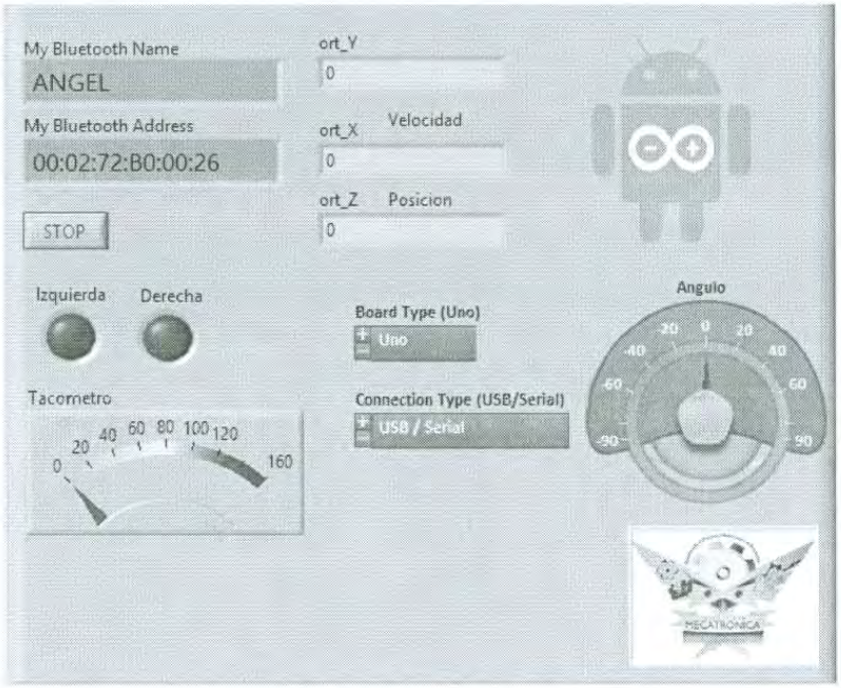


Figura 4.6 Panel frontal de LabVIEW

4.6 Actuadores finales

Arduino, como cualquier microcontrolador, es incapaz de proporcionar la potencia (el amperaje) que necesita un motor para funcionar, por lo que debemos emplear una fuente externa y una fase de potencia: la más simple posible se basa en un transistor MOSFET de nivel lógico: El esquema de la fase de potencia es:

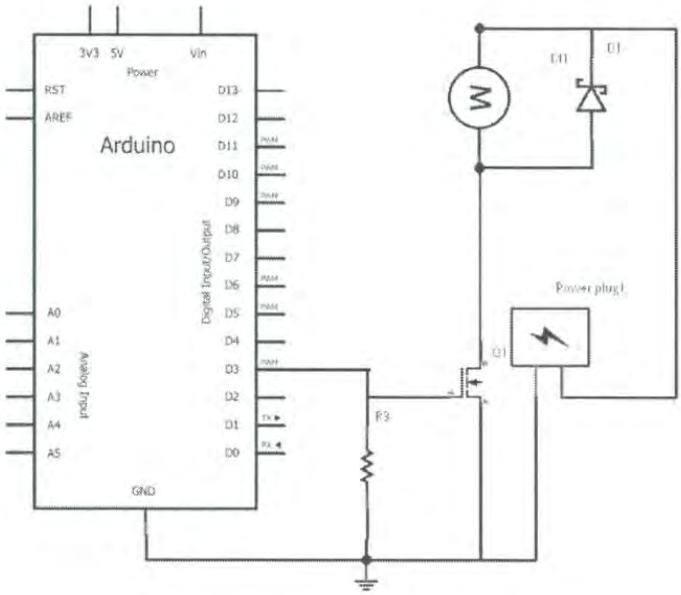


Figura 4.7 Esquema de Arduino con la fase de potencia.

Su funcionamiento es el siguiente: una señal lógica alta (5V) abre la puerta y permite el paso de la corriente, una señal de nivel bajo (0V) cierra la puerta e impide el paso de corriente. Esta señal que se debe transmitir tiene la ventaja de emplear un amperaje mínimo, del orden de mili o nanoamperios, asumible por el Arduino. El Ciclo de Trabajo del PWM será variado entre el tiempo de señal alta y tiempo de señal baja. El motor de corriente directa que es un dispositivo análogo percibirá una señal constante, cuyo voltaje será el valor medio de la función de pulsos.

Para el direccionamiento del servomotor, simplemente se utiliza el pin de salida digital 2, que este enviara el ángulo leído. El diagrama de configuración física se muestra en la Figura 4.8, donde se pueden apreciar los elementos controlados, la etapa de potencia y el controlador.

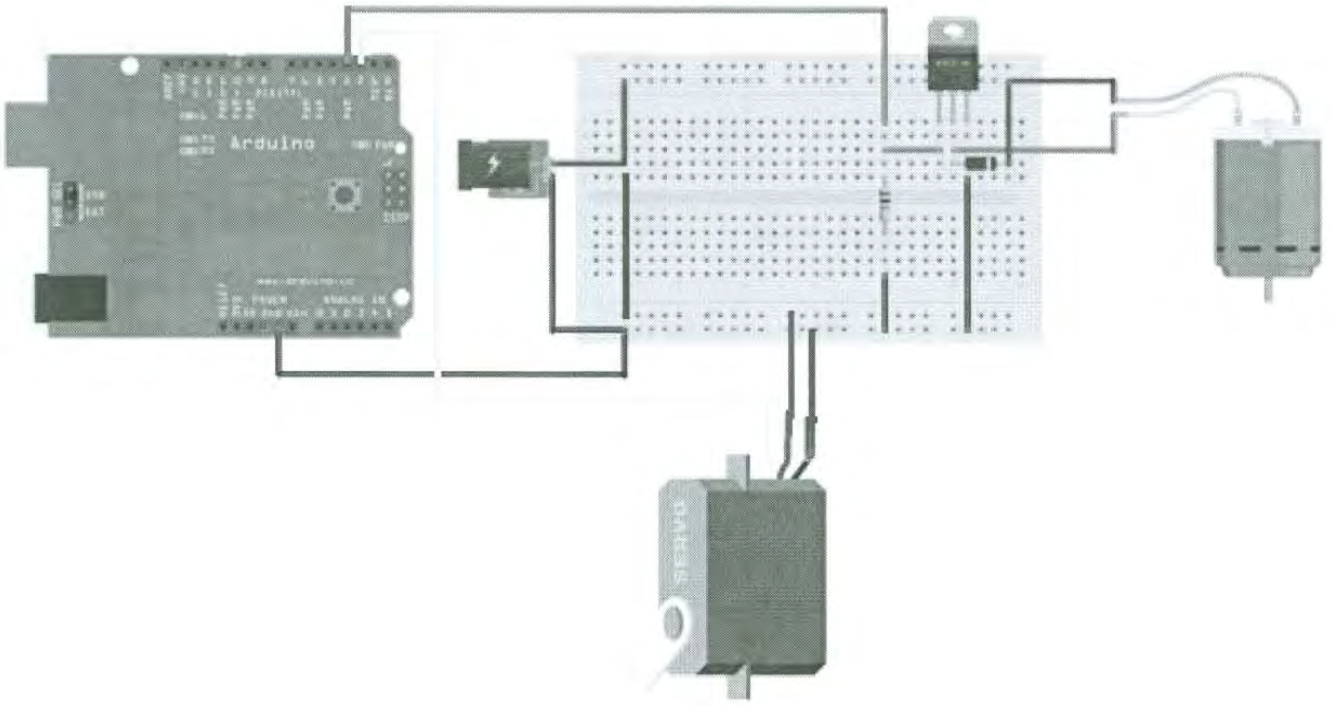


Figura 4.8 Diagrama de configuración final.

Capítulo 5

5. Conclusiones y trabajo futuro.

5.1 Conclusiones

La integración de distintos sistemas para desarrollar proyectos de ingeniería es cada vez más común, por ello varios proveedores de productos en el ramo de la automatización, comunicaciones e informática, están incluyendo en sus productos, los métodos necesarios para interactuar con otras plataformas de desarrollo para construir lo que se conoce como sistemas ciber-físicos, los cuales son diseñados para facilitar la interacción humano – sistema.

En este trabajo se utilizó la integración del sistema operativo Android, el lenguaje de programación LabVIEW y la plataforma de desarrollo Arduino. La finalidad de esta integración fue el control de un automóvil simulado por dos motores de corriente directa. Controlar los diversos parámetros en motores de corriente directa como son el sentido de giro, arranque y paro, el freno, y la velocidad son habilidades de gran importancia en los proyectos ingenieriles de la actualidad ya que al poder manipular a conveniencia dichos parámetros es posible realizar un gran número aplicaciones tanto industriales como de investigación.

Durante el desarrollo de este proyecto de tesis se demostró que la integración de distintos sistemas ayuda en gran medida a la implementación de las distintas teorías de comunicación y control que normalmente requieren gran trabajo si se utiliza solamente una plataforma de desarrollo. Un ejemplo clásico en ingeniería es el uso de MatLab, el cual es muy poderoso a la

hora de realizar cálculos pero es difícil o costoso conseguir una interacción de este con el entorno.

La selección del protocolo de comunicación es un paso muy importante en el proyecto, el hecho de poder comunicar los parámetros de Android por medio de Bluetooth con LabVIEW agiliza el desarrollo del proyecto. La comunicación de LabVIEW con Arduino es otra gran ayuda ya que permite que el procesamiento se realice en una computadora, la cual tiene grandes capacidades y como resultado del procesamiento de datos, comunica solo las instrucciones necesarias a la plataforma Arduino.

Como se mencionó en el capítulo cuatro, la tarea del dispositivo con Android es comunicar su posición, para lo cual utiliza el sensor de giroscopio interno, el cual provee el ángulo del dispositivo en los tres ejes coordenados. Estos datos son enviados a una computadora por medio de Bluetooth, un protocolo de comunicación de corto alcance pero muy utilizado en sistemas ciber-físicos ya que no añade datos innecesarios en la trama comunicada.

Los datos que envía Android son recibidos por LabVIEW quien se encarga de procesarlos y comunicarlos a la plataforma Arduino. En este sentido, LabVIEW funciona como una interfaz entre Android y Arduino. El procesamiento de LabVIEW consiste en establecer los límites y las tolerancias que se requieren para generar los parámetros del PWM que rige el comportamiento de los motores de corriente directa.

El sistema de control se desarrolló con la plataforma Arduino ya que esta fue diseñada de tal manera que cualquier persona pueda aprender a utilizarla con facilidad y también familiarizarse rápidamente con el lenguaje y el entorno de programación con el que cuenta. Arduino es muy utilizado en proyectos de esta índole dado que es un software libre con lo cual se tiene una adquisición gratuita y sencilla para todos los usuarios.

Combinar las diferentes tecnologías para el control de actuadores abre una nueva posibilidad para el desarrollo de proyectos más complejos en la facultad de Ingeniería Mecatrónica en la Universidad de Sonora.

5.2 Trabajos futuros

El proyecto de tesis cumplió con el objetivo de controlar un motor de corriente directa mediante la integración Android-LabVIEW-Arduino, este método de transmisión de información puede ser aplicado a diversos proyectos mecatrónicos futuros en la Universidad, un claro ejemplo es la aplicación que podría tener en el campo de pruebas de helióstatos de la Universidad de Sonora, al utilizar comunicación inalámbrica e integración de sistemas, se ahorraría la infraestructura de cableado para la comunicación, y tiempo de desarrollo, de igual manera la detección de errores y/o problemas de comunicación entre el dispositivo coordinador y el dispositivo final seria mas rápida puesto que ya no se tendrían que buscar los problemas en cableados, o en códigos complejos.

Para lograr controlar la cantidad total de motores que se tienen en el campo de helióstatos se requeriría de una distinta topología de red con la cual se pudiesen controlar más de un dispositivo final, lo cual se contempla en este trabajo pues el control efectuado es concurrente y con esto se tiene una mayor optimización de los recursos tecnológicos.

Bibliografía

[1] F. Pasqualetti, F. Dörfler, and F. Bullo. *Attack, Detection and Identification in Cyber-Physical Systems*. IEEE TRANSACTIONS ON AUTOMATIC CONTROL, VOL. 58, NO. 11, NOVIEMBRE 2013.

[2] K. Zhang, J. Sprinkle, *Model-Based Software Synthesis for Self-Reconfigurable Sensor Network in Water Monitoring*. 20th IEEE International Conference and Workshops on the Engineering of Computer Based Systems (ECBS), Scottsdale, Arizona, 2013.

[3] *Grupo de Investigación en Robótica Autónoma del CAETI. (2009-2012). Physical Etoys.*(1ª Edición) Argentina: WordPress.

[4] *Ruiz, J. (2008). Educación Secundaria y Universitaria: Robótica Control Simulación. (1ª Edición) España: Free CSS Templates.*

[5] Ruiz, J. (2012). *Utilización de S4A (Scratch) más la Tarjeta Arduino en un ambiente de programación grafica orientado a la educación. (1ª Edición) España: Free CSS Templates*

[6] San Miguel, C. (2012). *Diseño y desarrollo de un sistema de monitorización y control para un invernadero de uso doméstico.* Tesis Ingeniería Industrial. Universidad de Cantabria.

[8] Google (2012). *Philosophy and Goals*. Android Open Source Project. Recuperado 21 de abril del 2012.

[9] Travis, J. Kring, J. (2006). *LabVIEW for Everyone: Graphical Programming Made Easy and Fun*. Part of the National Instruments Virtual Instrumentation Series. 3rd Edition, July 27, 2006, Prentice Hall.

[10] S. J. Chapman, Máquinas Eléctricas, 4ta ed, Mc. Graw Hill.

[11] http://www.arduino.cc/es/, consultada el 10/05/2014.

[12] Zbar, P. Malvino, A. Miller, M. (2001). *Prácticas de electrónica.* (7ª Edición). México: ALFAOMEGA.

[13] Ordaz, U. (2009). *Controladores Logicos Programables.* (1ª Edición). México: Trillas.