UNIVERSIDAD CATÓLICA
DE SANTIAGO DE GUAYAQUIL

FACULTAD DE EDUCACIÓN TÉCNICA PARA EL DESARROLLO
CARRERA DE INGENIERÍA ELECTRÓNICA EN CONTROL Y AUTOMATISMO

TEMA:

Diseño y simulación de prácticas virtuales usando LabView, Proteus yArduino mediante NI VISA

AUTOR:
Cifuentes Vera, Jean Luis

Trabajo de Titulación previo a la obtención del grado de
INGENIERO ELECTRÓNICO EN CONTROL Y AUTOMATISMO

TUTOR:
Suarez Murillo, Efraín Oswaldo

Guayaquil, Ecuador

21 de Marzo del 2017

UNIVERSIDAD CATÓLICA
DE SANTIAGO DE GUAYAQUIL

FACULTAD DE EDUCACIÓN TÉCNICA PARA EL DESARROLLO

CARRERA DE INGENIERÍA ELECTRÓNICA EN CONTROL Y AUTOMATISMO

CERTIFICACIÓN

Certificamos que el presente trabajo fue realizado en su totalidad por el Sr. **Cifuentes Vera, Jean Luis** como requerimiento para la obtención del título de**INGENIERO ELECTRÓNICO EN CONTROL Y AUTOMATISMO**

TUTOR

_____

Suarez Murillo, Efraín Oswaldo

DIRECTOR DE CARRERA

_____

Heras Sánchez, Miguel Armando

Guayaquil, a los 21 del mes de Marzo del año 2017

UNIVERSIDAD CATÓLICA
DE SANTIAGO DE GUAYAQUIL

FACULTAD DE EDUCACIÓN TÉCNICA PARA EL DESARROLLO

CARRERA DE INGENIERÍA ELECTRÓNICA EN CONTROL Y AUTOMATISMO

DECLARACIÓN DE RESPONSABILIDAD

Yo, **Cifuentes Vera, Jean Luis**

DECLARÓ QUE:

El trabajo de titulación **"Diseño y simulación de prácticas virtualesusando LabView, Proteus y Arduino mediante NI VISA"** previo a laobtención del Título de **Ingeniero Electrónico en Control y Automatismo**, ha sido desarrollado respetando derechos intelectuales de terceros conforme las citas que constan en el documento, cuyas fuentes se incorporan en las referencias o bibliografías. Consecuentemente este trabajo es de mi total autoría.

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

Guayaquil, a los 21 del mes de Marzo del año 2017

EL AUTOR

_____

CIFUENTES VERA, JEAN LUIS

UNIVERSIDAD CATÓLICA
DE SANTIAGO DE GUAYAQUIL

FACULTAD DE EDUCACIÓN TÉCNICA PARA EL DESARROLLO
CARRERA DE INGENIERÍA ELECTRÓNICA EN CONTROL Y AUTOMATISMO

AUTORIZACIÓN

Yo, **Cifuentes Vera, Jean Luis**

Autorizó a la Universidad Católica de Santiago de Guayaquil, la publicación, en la biblioteca de la institución del Trabajo de Titulación: **"Diseño y simulación de prácticas virtuales usando LabView, Proteus y Arduino mediante NI VISA"**, cuyo contenido, ideas y criterios son de mi exclusivaresponsabilidad y total autoría.

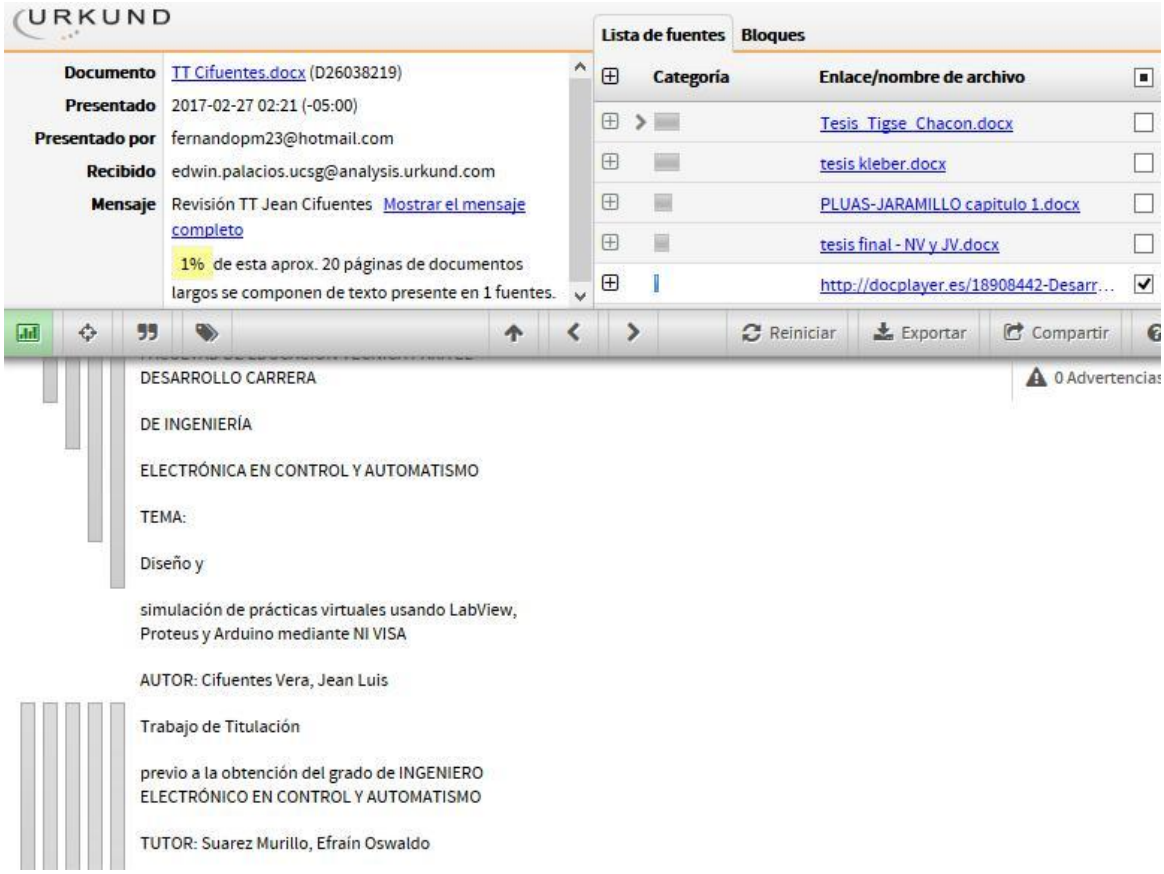
Guayaquil, a los 21 del mes de Marzo del año 2017

EL AUTOR

_____

CIFUENTES VERA, JEAN LUIS

REPORTE DE URKUND



DEDICATORIA

A mi amigo fiel, mi lugar de protección, mi más alto

escondite, mi libertador; a quien es mi escudo,

y con él me protejo, Dios.

EL AUTOR

CIFUENTES VERA, JEAN LUIS

AGRADECIMIENTO

A mi padre, Luis Cifuentes, por ser mi guía y apoyo en todo momento

A mi madre, Janeth Vera, por cada consejo a lo largo De mi vida, por estar conmigo incondicionalmente

A mi esposa,Raisha Burgos, por creer en mí, por motivarme cada día y ayudarme a crecer

A mi hermana, María Fernanda Cifuentes,por acompañarmedurante este arduo camino y estar conmigo en momentos de alegrías y fracasos.

A los profesores que tuve a lo largo de mi carrera, por prepararmepara una vida llena de retos no solo en lo profesional sino también en lo personal.

EL AUTOR

CIFUENTES VERA, JEAN LUIS

UNIVERSIDAD CATÓLICA
DE SANTIAGO DE GUAYAQUIL

FACULTAD DE EDUCACIÓN TÉCNICA PARA EL DESARROLLO

CARRERA DE INGENIERÍA EN TELECOMUNICACIONES

TRIBUNAL DE SUSTENTACIÓN

f. _____

SUAREZ MURILLO, EDWIN OSWALDO

TUTOR

f. _____

HERAS SÁNCHEZ, MIGUEL ARMANDO

DIRECTOR DE CARRERA

f. _____

CORDOVA RIVADENEIRA, LUIS SILVIO

COORDINADOR DE ÁREA ELECTRÓNICA

Índice General

Índice de Figuras

Capítulo 2

Capítulo 3

Índice de Tablas

Capítulo 2

Resumen

El propósito principal del trabajo de titulación es la integración de varias herramientas virtuales, como, LabView, Isis Proteus y programación de alto nivel de código abierto. Antes del desarrollo de las aplicaciones prácticas del trabajo de titulación, se realizó una descripción general de los fundamentos básicos de microcontroladores y de la familia de microcontroladoresAtmel, así como también de la plataforma de desarrollo integrado Arduino IDE. No ha sido necesaria la descripción general de la herramienta virtual LabView debido a que los estudiantes de la Carrera de Ingeniería Electrónica en Control y Automatismo si lo han tratado. Mientras que los microcontroladoresAtmel ni Arduino fueron discutidos en clases. Posteriormente, se desarrollaron seis aplicaciones prácticas en la que se integran diseños de instrumentación virtual, simulación virtual y programación de código abierto Arduino. Finalmente, se evaluaron las aplicaciones prácticas desarrolladas los mismos que resultaron se satisfactorios durante la ejecución de cada una de las prácticas.

Palabras claves: LABVIEW, MICROCONTROLADORES, ATMEL, ATMEGA 328P, PROTEUS, ARDUINO.

CAPÍTULO 1: Descripción del Trabajo de Titulación

1.1. Introducción de instrumentación y simulación virtual.

Un instrumento es un dispositivo diseñado para recoger datos de un entorno, o desde la unidad bajo prueba, y para mostrar información al usuario sobre la base de los datos recogidos. Tal instrumento puede emplear un transductor para detectar cambios en un parámetro físico, como la temperatura o la presión, y para convertir la información detectada en señales eléctricas, tales como las variaciones de tensión o de frecuencia.

El término instrumento también se puede definir como el software del dispositivo físico que lleva a cabo un análisis de los datos adquiridos de otro instrumento y después saca los datos procesados para mostrarse o grabarse en dispositivos extraíbles. Esta segunda categoría de instrumentos de registro puede incluir osciloscopios, analizadores de espectro, y multímetros digitales.

Los tipos de datos de origen son recogidos y analizados por los instrumentos que pueden variar ampliamente. Por lo tanto, se incluyen los parámetros físicos tales como: la temperatura, la presión, la distancia, la frecuencia y la amplitud de la luz y el sonido, y también los parámetros eléctricos, incluyendo voltaje, corriente y frecuencia.

Sin dejar de lado las plataformas de simulación virtual, tenemos al programa Isis Proteus, que permite modelar en tiempo real cualquier diseño

electrónico previamente realizado. Muy utilizado en la simulación de circuitos electrónicos digitales y de sistemas de microcontroladores. Esta herramienta interactúa de manera conjunta con la plataforma de instrumentos virtuales LabView utilizando puertos virtuales de comunicaciones. Y finalmente, se programa el microcontroladorATMega 328P que viene incorporado en la tarjeta Arduino UNO. La programación es un lenguaje de alto nivel, C lo que permite una programación versátil y el uso de librerías disponibles, ya que es una plataforma de código abierto.

1.2. Antecedentes de la investigación.

Durante la búsqueda de información de proyectos relacionados con procesos que integren instrumentos y herramientas de simulación, no se pudieron encontrar trabajos que trabajen de manera conjunta. Pero existen trabajos de investigación y de tesis en la que se desarrollan aplicaciones mediante el diseño de instrumentos virtuales a través de LabView. También se pudo encontrar trabajos en las que se desarrollan aplicaciones mediante el uso del programa Isis Proteus para simular diseños electrónicos utilizando microcontroladores PIC, y nulo con microcontroladoresATMega 328P.

Finalmente, también se encontraron proyectos donde se utiliza programación de alto nivel, que en este caso es de código abierto, como lo es, la plataforma de desarrollo Arduino IDE.

1.3. Definición del problema.

En las asignaturas de microcontroladores se realizan prácticas utilizando la familia de los microcontroladores PIC; mientras que en instrumentación virtual se desarrollan pequeñas aplicaciones de instrumentos virtuales. Por esto, surge la necesidad de desarrollar aplicaciones prácticas que integran instrumentos virtuales de LabView, simulaciones virtuales con Isis Proteus y programación en lenguaje de alto nivel con la plataforma de desarrolloArduino IDE de código abierto.

1.4. Justificación del problema.

El vertiginoso desarrollo de los microcontroladoresAtmel que trabaja en una placa o hardware de código abierto de Arduino, ha permitido que el programa Isis Proteus admita librerías de Arduino UNO que tiene incorporado el microcontroladorATMega 328P. Este es programado con un archivo ‘hex’ que se genera después de ser compilado por Arduino IDE.

Finalmente, el diseño desarrollado en Isis Proteus utiliza comunicación serial para enviar información a través de un instrumento virtual que ha sido diseñado en la plataforma LabView.

1.5. Objetivos

1.5.1. Objetivo General.

Realizar el diseño y simulación de prácticas virtuales usando LabView, Proteus y Arduino mediante NI VISA.

1.5.2. Objetivos Específicos.

- Describir la fundamentación teórica de los sistemas microcontroladores y plataforma Arduino.
- Diseñar escenarios de instrumentos y simulaciones virtuales utilizando LabView e Isis Proteus.
- Desarrollar los algoritmos de programación para cargarlos en el simulador Isis Proteus.
- Evaluar el diseño de las aplicaciones prácticas a través de LabView e Isis Proteus.

1.6. Hipótesis.

La integración de instrumentos virtuales, simulador virtual y programación de alto nivel de código abierto permitirá el desarrollo adecuado de aplicaciones prácticas que se pueden orientar como ayuda para el desarrollo de proyectos integradores o también de nuevos proyectos de trabajos de titulación.

1.7. Metodología de investigación.

Existen varios métodos investigativos que se pueden emplear, pero cada método es diferente y depende siempre del tema a investigar. En proyectos de ingeniería los métodos más utilizados son el descriptivo, explicativo y exploratorio. El presente trabajo de titulación se utiliza elmétodo descriptivo y explicativo, cuyo diseño es netamente virtual ya que se utiliza herramientas de instrumentación, simulación y programación virtual.

CAPÍTULO 2: Fundamentación Teórica

Este capítulo describe los fundamentos teóricos de los microcontroladoresAtmel y la plataforma de desarrollo Arduino.

2.1. Introducción a los microcontroladores.

Un microcontrolador es un componente electrónico que ha integrado en el mismo chip de microprocesador durante un número de periféricos que son útiles para controlar un proceso. Este dispositivo es un sistema informático completo (véase la figura 2.1) en el que se incluyen una CPU (Unidad Central de Proceso), memoria de datos y programas, un reloj de sistema, puertos I/O (entrada / salida), y otros posibles periféricos, como, por ejemplo, módulos y convertidores A/D, entre otros, integrados de temporización en el mismo componente.

Los bloques principales presentes en una arquitectura de microcontrolador son:

- Unidad Central de Procesamiento (CPU);
- Sistema de reloj para secuenciar las rutinas de la CPU;
- La memoria para obtener instrucciones de manipulación y almacenamiento de datos;
- Entradas para el tratamiento de la información del medio externo;
- Salidas para actuar en las interfaces con el entorno externo;

- Programa (firmware) para establecer una meta para el sistema

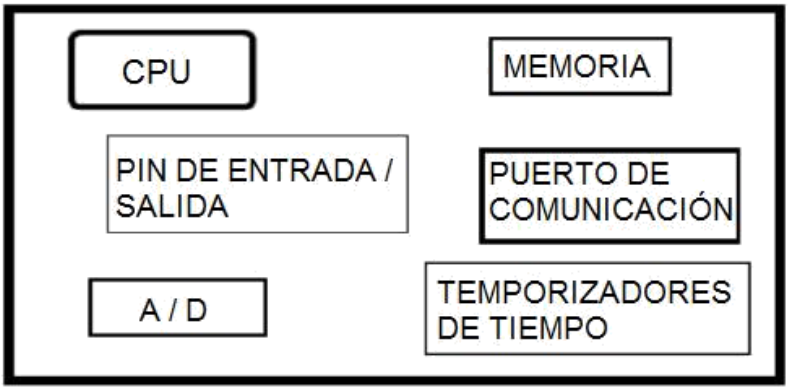


Figura 2. 1: Esquemático simplificado de un microcontrolador.
Fuente: (López P., 2014)

2.2. Descripción de los bloques interno de un microcontrolador.

Los bloques internos de un microcontrolador (véase la figura 2.2) son descritos a continuación:

- **Unidad de memoria:** la memoria es el bloque que tiene la funciónbásica de almacenamiento de datos. Para una dirección dada, hay un contenido específico;
- **Unidad central de proceso - CPU:** realiza operaciones aritméticas ylógicas y por lo tanto puede realizar operaciones de multiplicar, dividir, restar y mover su contenido desde una ubicación de memoria a otra. Estas posiciones de memoria de la CPU se llaman registros;
- **Bus:** físicamente hablando, es un grupo de 8, 16 o más líneas deinformación de tráfico. A continuación, se describe brevemente los tipos de buses:

- Bus de datos: depende de la cantidad de memoria que se quiereabordar;
- Dirección de bus: depende de la anchura de la palabra de datos.

- **Unidades de entrada/salida:** los pines o puertos de E/S se denominanasí, porque pueden ser configurados como, entrada y salidas:
  - I - entrada.
  - O - salida.
- **Puerto de comunicación serie:** unidad responsable de la realizaciónde la comunicación en serie mediante el envío y recepción de datos a través de dos vías de comunicación.
- **Unidad de sincronización:** bloque de informaciónsobre la hora, la duración y el protocolo. La unidad básica es un contador del temporizador (registro) el contenido de las cuales se incrementa de una unidad en un intervalo de tiempo fijo. Obtención de información directa de estos registros de tiempo (T1 y T2).
- **Regulador:** bloque provisto de un contador continuo a través de unreloj interno y que el programa pone a cero cada vez que se ejecuta correctamente. Cuando ocurre algún fallo de procesamiento de las instrucciones del microcontrolador, no se escribirá el cero y el contador se reinicia automáticamente el dispositivo.
- **Convertidor Analógico/Digital:** cómo las señales de los periféricosson sustancialmente diferentes de los que el microcontrolador puede comprender (señal digital cero y uno), se debe convertir en un formato que puede ser leído por el microcontrolador. Esta tarea es realizada por

medio de un bloque para la conversión analógico-digital (A/D). Este bloque realiza la conversión de un valor de información analógica en un número binario yde su camino a través del bloque de la CPU de modo que puede procesar la información.

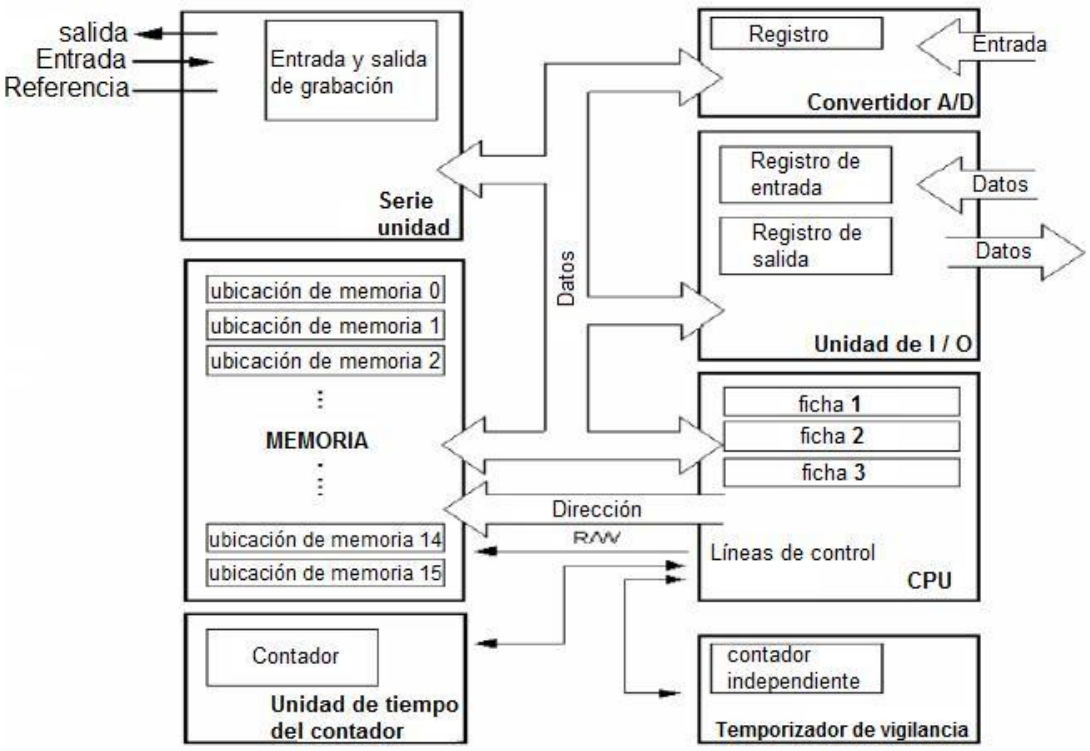


Figura 2. 2: Diagrama interno de un microcontrolador.
Fuente: (López P., 2014)

2.3. Ventajas de utilizar microcontroladores.

Usar un microcontrolador, es necesario que el periférico principal ya este incorporado en su carcasa. De este modo, se optimiza el tiempo y el espacio en la construcción y diseño de proyectos. El universo de las aplicaciones de microcontroladores está en auge, con una gran parte de las aplicaciones en sistemas integrados o embebidos. Los sistemas embebidos (del inglés EmbeddedSystem) se refiere al microcontrolador insertado en las aplicaciones (productos) y que se utiliza exclusivamente para ellos. A

medida que la complejidad de estos sistemas crece de manera espectacular,el software ha sido clave para dar respuesta a las necesidades de este mercado.

Teniendo en cuenta este creciente mercado, el software para microcontroladores representan una parte considerable del mercado mundial de software. El incremento en sistemas embebidos con el advenimiento de Microsoft (fabricante de renombre mundial de sistemas operativos y software relacionado) son responsables de la reanudación del crecimiento de la industria de software en los Estados Unidos de América.

2.4. Definición de tipos de microcontroladores.

Después de las funciones especificadas y lo que se espera de un proyecto, es el momento de determinar el mejor microcontrolador a utilizar. Esta elección debe tratar de utilizar las características del microcontrolador en su totalidad, a fin de no perder el procesamiento y los dispositivos periféricos, lo que aumentaría el costo final.

Sin embargo, no es aconsejable elegir microcontroladores con cantidades de mucha memoria cerca del límite de lo que se estima de usar. Esto puede resultar en la necesidad de cambiar el microcontrolador elegido si es necesario modificar el programa. En muchos casos, se utiliza un microcontrolador diferente durante el desarrollo del firmware final. Las diferencias son, en general, la cantidad de los procesos de memoria y de grabación.

Otros factores no relacionados con los requisitos del proyecto deben tenerse en cuenta antes de elegir el mejor microcontrolador, incluyendo:

- costo;
- conocimiento previo acerca de la arquitectura, lo que afectará el tiempo de aprendizaje para el desarrollo;
- soporte del fabricante: kits de desarrollo, librerías estándar;
- existencias de posibles reemplazos pines compatibles para otros fabricantes;
- continuidad de fabricación: el costo de un proyecto puede descarrilar su producción en el futuro si el microcontrolador en el que se basa ya no se produce y no hay sustitutos de precios competitivos en el mercado;
- facilidad de compra;
- consumo, tensión de alimentación, opciones de oscilador, frecuencias de trabajo.

2.5. MicrocontroladoresAtmel.

AVR ATmega es una familia de microprocesadores de 8 bits de Atmel. Sus características varían dependiendo del modelo, pero, sobre todo, las siguientes están presentes: (a) memoria flash entre 4 kB y 256 kB, (b) pines entre 28 y 100 cuyo encapsulamiento sean de dispositivos de montaje superficial *(Surface Mount Device, SMD)* o en línea dual (Dual

InlinePackage, DIP), (c) un temporizador de vigilancia (watchdog), y (d) velocidad de reloj de hasta 20 MHz (depende del reloj externo).

Además, la familia de microprocesadores ofrece memorias Flash, SRAM y EEPROM interna. La familia AVR ATmega también admite protocolos de comunicación, tales como:

a) bus de interfaz de periféricos serie o bus SPI, en la figura 2.3 se muestra las especificaciones del bus SPI.

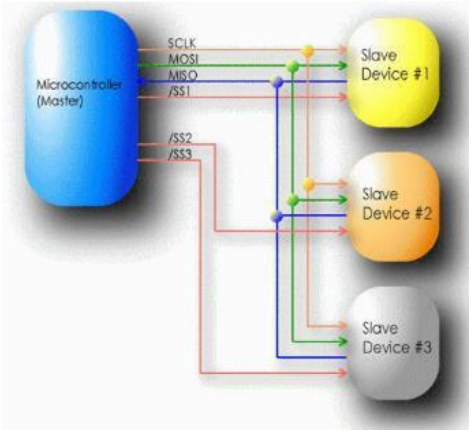


Figura 2. 3: Esquemático de las especificaciones del bus SPI.
Fuente: (López P., 2014)

b) bus de interfaz de dos cables o bus TWI (introducido por Atmel para evitar conflictos con problemas de marcas relacionadas con I^2C) compatible con I^2C, en la figura 2.4 se muestra el esquemático de TWI.

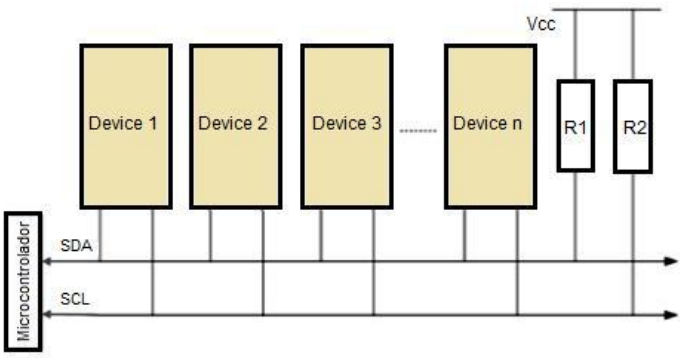


Figura 2. 4: Esquemático de comunicación del bus TWI.
Fuente: (Hossain, Shourov, Rana, &Anower, 2015)

c) bus de serie universal (USB), en la figura 2.5 se muestra la comunicación entre un microcontrolador y el puerto USB.

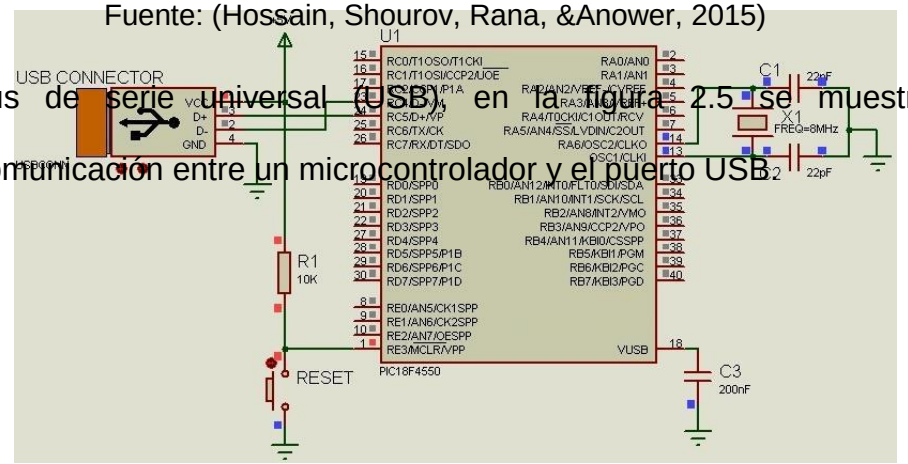


Figura 2. 5: Esquemático de comunicación entre microcontrolador y USB.
Elaborado por: Autor.

d) bus de red de área de control o bus CAN, en la figura 2.6 se muestra el esquemático de la estructura del bus CAN, para Pacheco H., (2011) CAN es un protocolo de comunicación serial orientado a control distribuido y en aplicaciones de automóviles.

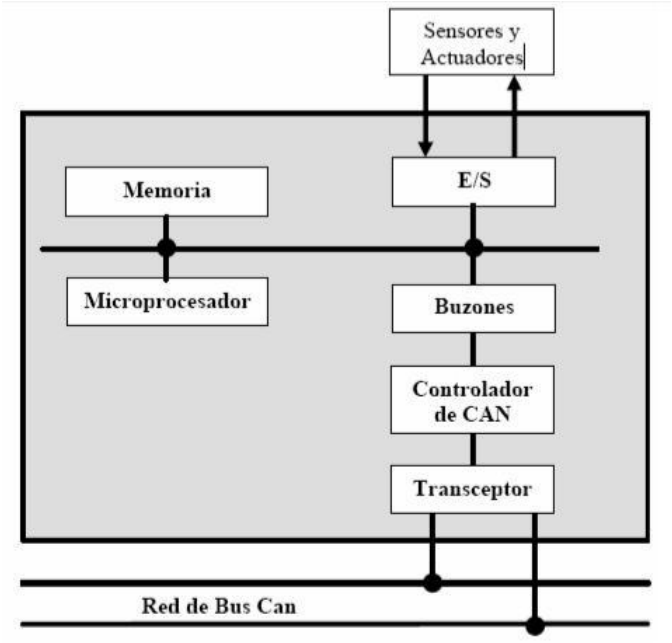


Figura 2. 6: Esquemático de la estructura de comunicación del bus CAN.
Fuente: (Alepuz S. et al., 2006)

e) bus de red local de interconexión o bus LIN.

De acuerdo con los requisitos técnicos y de acuerdo con los puntos de selección planteados anteriormente, fue seleccionado para el proyecto de la idealización del microcontrolador de la familia ATmega16 del fabricante ATMEL AVR.

2.5.1. Microcontroladores de la Familia AVR.

El fabricante Atmel, surge la familia de microcontroladores AVR como una herramienta versátil para uso en proyectos o diseños electrónicos por su agilidad y estabilidad operativa, además lleva la ventaja de una escritura de hardware relativamente simple y fácil de implementar. Tiene un sistema de apoyo a los usuarios para resolver cuestiones técnicas (función disponible sólo en inglés). En comparación con otros microcontroladores técnicamente equivalentes (de otros fabricantes), que tiene un valor en muchos casos de adquisición inferior.

Esta familia de procesadores fue diseñada y desarrollada originalmente en 1996 en el Instituto de Ciencia y Tecnología de Noruega por los estudiantes Alf-EgilBogen y VergardWollan, quien nombró entonces a esta línea de siglas dispositivos AVR, o Advanced RISC virtual. Esta línea posee la arquitectura RISC de 8 bits con concepto Harvard. A continuación, se describe el concepto de Harvard y la arquitectura RISC:

- Concepto HARVARD (véase la figura 2.7): la memoria de datos es independiente de la memoria del programa, por lo que es posible unmayor flujo de datos a través de la unidad central de proceso y una

  mayor velocidad de ejecución de la tarea por el dispositivo;

- Arquitectura RISC: proviene de ordenador con un conjunto reducido de instrucciones *(ReducedInstruction Set Computer, RISC)* para la ejecución de los comandos designados.

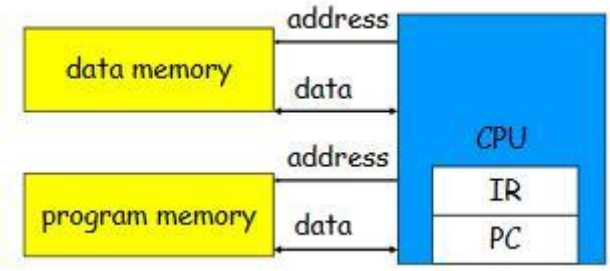


Figura 2. 7: Diagrama de bloques del concepto de HARVARD.
Fuente: (Chenyang, 2014)

2.5.2. Características del microcontroladorATMega 16.

En las figuras 2.8 y 2.9 se muestran los microcontroladores con encapsulamiento DIP y TQFP, respectivamente.

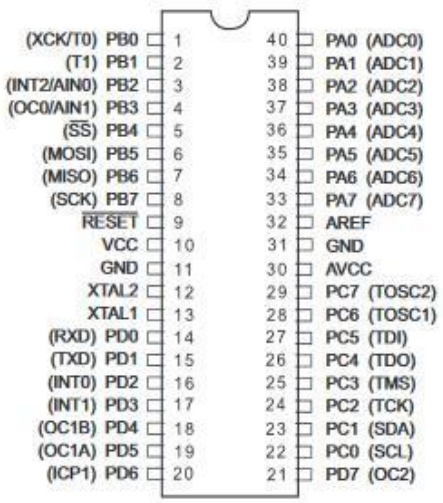


Figura 2. 8: Encapsulación DIP del microcontrolador ATmega16. Fuente: (Atmel, 2010)

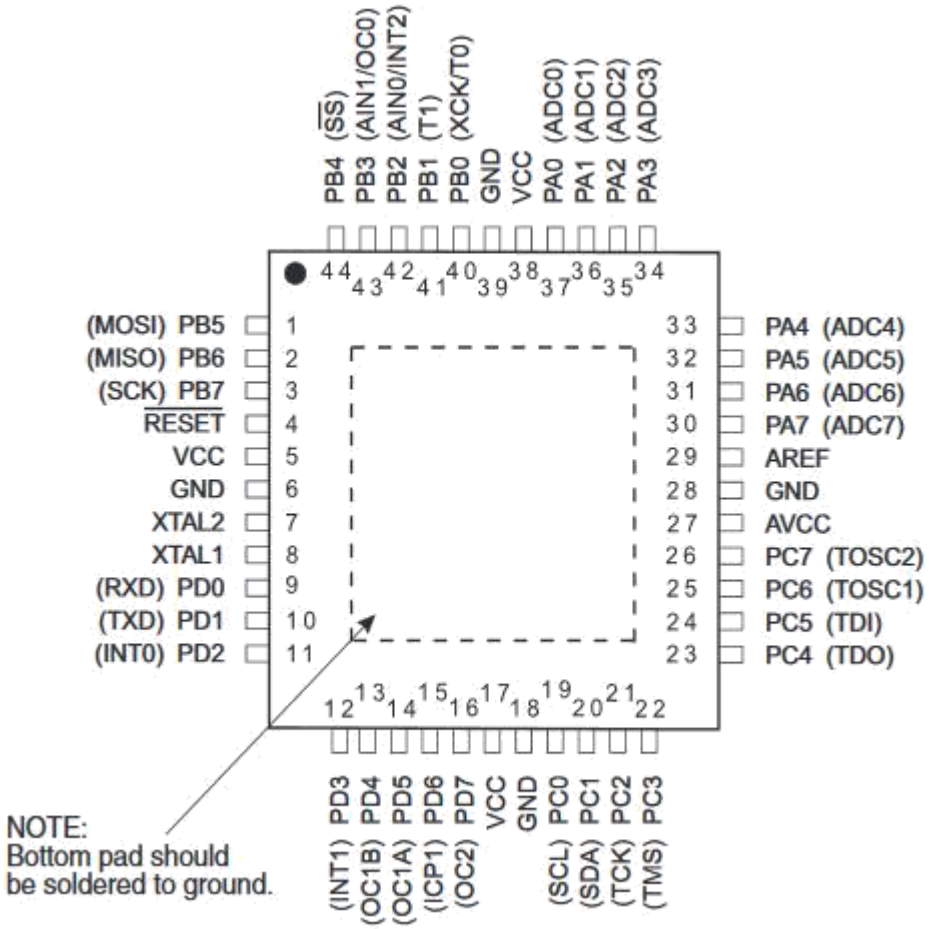


Figura 2. 9: Encapsulación TQFP del microcontrolador ATmega16.
Fuente: (Atmel, 2010)

Las principales características que presenta el microcontrolador ATmega16 (Atmel, 2010):

- Alto rendimiento y bajo consumo de energía;
- Arquitectura RISC;
- Memoria flash de 16 Kbytes;
- Memoria EEPROM de 512 bytes;
- SDRAM interna de 1 Kbyte;
- Retención de la memoria por 20 años a 85 °C o 100 años a 25 °C;
- Programación en el sistema (del inglés In-SystemProgramming, ISP);

- Protección contra escritura a través del software;
- Dos temporizadores/contadores de 8 bits;
- Un temporizador/contador de 16 bits con posibilidad de funcionamiento en modo independiente, así como un oscilador independiente;
- 4 canales de PWM;
- 8 puertos de conversión AD de 10 bits;
- 2 puertas con diferenciales de ganancia programable 1x, 10x, 200x;
- Interfaz de serie;
- USART programable de serie;
- interfaz en serie SPI maestro/esclavo;
- temporizador de vigilancia programable con oscilador independiente;
- comparador analógico integrado;
- detección de caída de tensión (Brown-out);
- Oscilador RC interno;
- Interrupciones por medios internos o externos;
- 32 líneas programables de E/S;
- voltajes de operación para: (a) ATmega16L opera entre 2.7V y 5.5V, y (b) ATmega16 opera entre 4.5V y 5.5V;
- Velocidades para: (a) ATmega16L entre 0 y 8 MHz; (b) ATmega16 entre 0 y 16 MHz;
- Consumo a 1 MHz, 3V y 25 ° C para el modelo ATmega16L: (a) Activo: 1,1 mA; (b) Modo de Espera: 12:35 mA; y (c) Modo de Apagado: <1 uA.

En las figuras 2.10 y 2.11 se muestran esquemáticamente en bloques facilidad de integración del microcontrolador periférico este modelo:

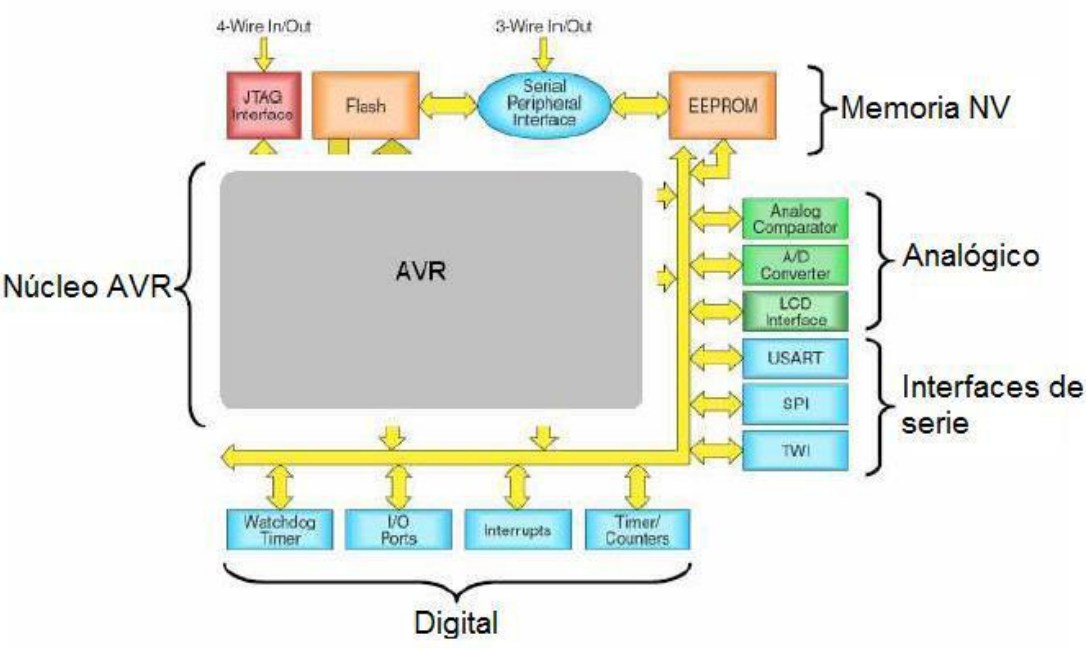


Figura 2. 10: Versatilidad de integración según el fabricante ATMEL Fuente: (Atmel, 2010)

En la tabla 2.1 se muestra una comparación de los recursos proporcionados por los microcontroladores ATmega16 en comparación con los modelos de la competencia en el mercado, según el fabricante ATMEL:

Tabla 2. 1: Comparativo entre microcontroladores contra Atmel.

Dispositivos microcontroladores	**velocidad máxima (MHz)**	**Tamaño código (bytes)**	**Ciclos**	**Tiempo de ejecución (us)**
ATMega16	16	32	227	14.2
MSP430	8	34	246	30.8
T89C51RD2	20	57	4200	210.0
PIC18F452	40	92	716	17.9
PIX16C74	20	87	2492	124.6
68HC11	12	59	1238	103.2

Fuente: (Atmel, 2010)

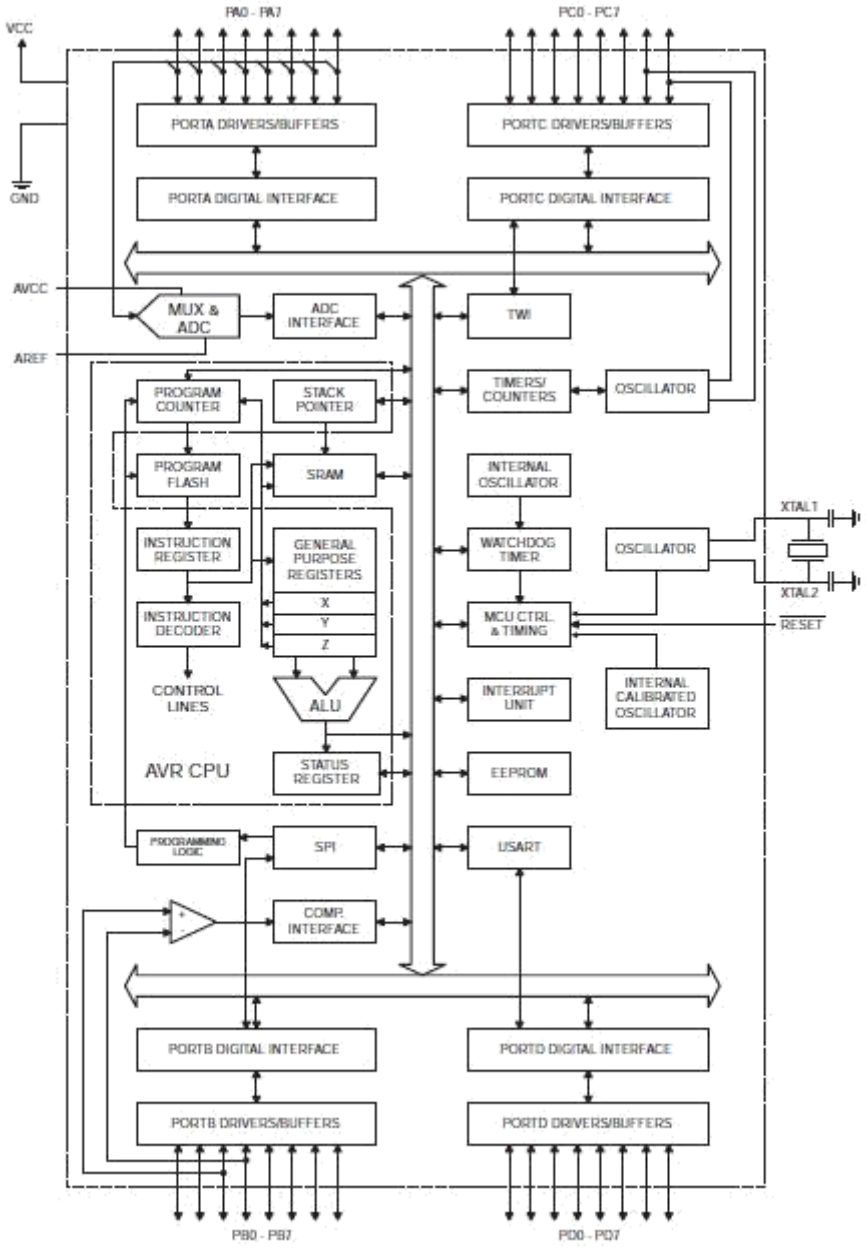


Figura 2. 11: Diagrama de bloques del microcontroladorATmega16 Fuente: (Atmel, 2010)

2.6. Arduino programa de código abierto (open source).

Toda la línea de productos Arduino se basa en el concepto visionario de hardware y software de código abierto. Es decir, los desarrollos de hardware y software se comparten abiertamente entre los usuarios

paraestimular nuevas ideas y promover el concepto de Arduino. De acuerdo con el concepto de Arduino, este equipo comparte abiertamente los esquemáticos de los diseños de las tarjetas diseñadas, por ejemplo, en la figura 2.12 se muestra el esquemático de la tarjeta Arduino UNO.

2.7. Otras plataformas de hardware basadas en Arduino.

Hay una gran variedad de plataformas basadas en Arduino. Las plataformas pueden adquirirse en SparkfunElectronics, Boulder, CO (www.sparkfun.com). La figura 2.13 proporciona una muestra representativa de la tarjeta Arduino Lily Pad equipado con el procesador ATMega 328P. Otras versiones del Lily Pad vienen equipados con el ATMega 128. Esta tarjeta de procesamiento puede ser usado y puede lavarse. Fue diseñado para ser cosido sobre tela.

En la figura 2.14 se muestra la tarjeta Arduino Mega equipado con el procesador ATMega 2560. Esta placa de procesamiento está equipada con 54 pines de E/S digitales, 14 patillas PWM, 16 entradas analógicas y cuatro canales de capacidad de comunicación en serie. En la parte superior derecha está el sello Arduino, aunque es pequeña, pero potente microprocesador que está equipada con ATMega 168.

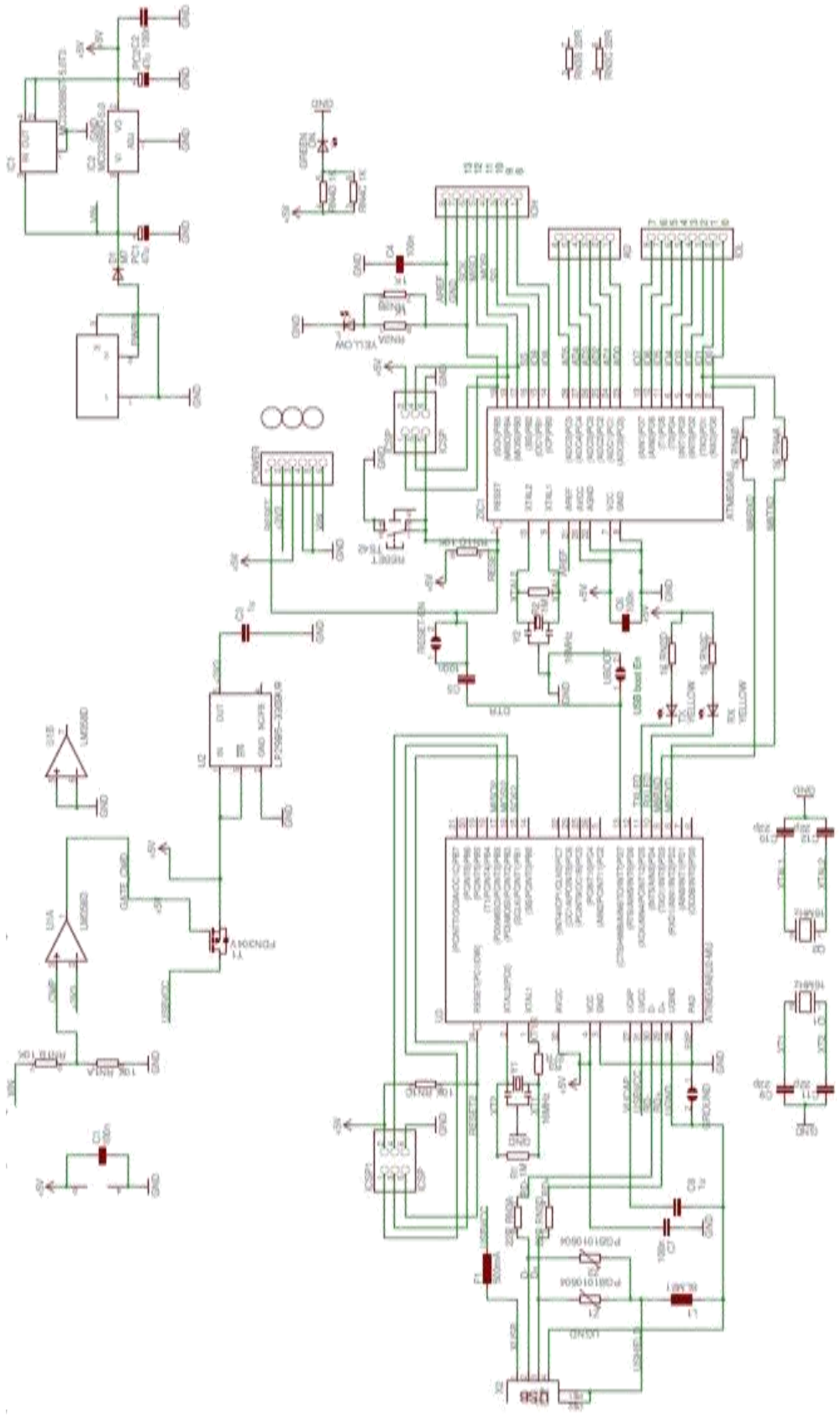


Figura 2. 12: Diseño esquemático de la tarjeta Arduino UNO de código abierto. Fuente: (Arduino, 2013)

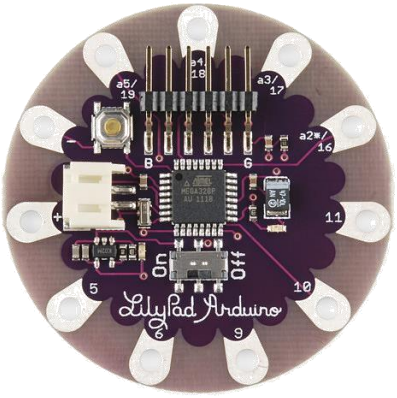


Figura 2. 13: Tarjeta Arduino Lily Pad incorporado el microcontroladorATMega 328P.Fuente: (Sparkfun, 2017)

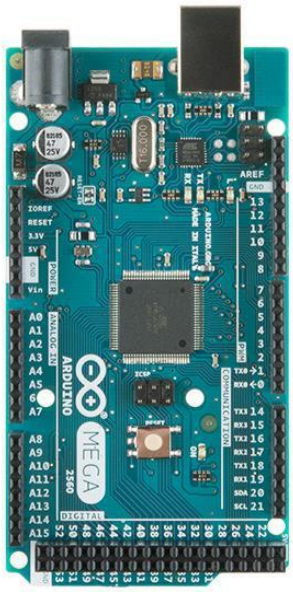


Figura 2. 14: Tarjeta Arduino Mega incorporado el microcontroladorATMega 2560. Fuente: (Sparkfun, 2017)

2.8. Ampliación de las características de Hardware de la plataforma Arduino.

Se pueden agregar funciones adicionales y hardware externo a las plataformas Arduino seleccionadas mediante el uso de un concepto de tarjeta secundaria. La tarjeta secundaria se llama ArduinoShield tal como se

muestra en la figura 2.15. El escudo o blindaje se acopla con las clavijas decabecera de la placa Arduino. El escudo proporciona una pequeña área de fabricación, un botón de reinicio del procesador y un botón de uso general y dos diodos emisores de luz (LEDs).

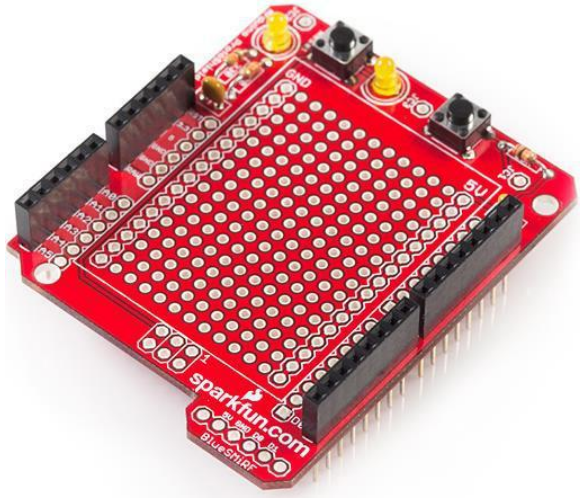


Figura 2. 15: Escudo de Arduino.
Fuente: (Sparkfun, 2017)

2.9. Plataforma de desarrollo de código abierto Arduino IDE.

En esta sección no discutiremos cómo programar la tarjeta de procesamiento Arduino UNO. En la figura 2.15 se muestra el entorno de programación de Arduino. Es esencial descargar y activar correctamente el software de la página principal de Arduino (https://www.arduino.cc/) contiene instrucciones detalladas sobre cómo descargar el software, cargar los controladores USB y obtener un programa de ejemplo que funciona en la placa de procesamiento Arduino UNO.

Debido a un espacio limitado, estas instrucciones no se duplicarán aquí. Como es una plataforma de código abierto, se recomienda al lector visitar la página web de Arduino y poner en marcha los principios básicos deprogramación. En la siguiente sección, ofrecemos una descripción más detallada de las características de hardware del procesador Arduino, el ATMega 328 de Atmel.

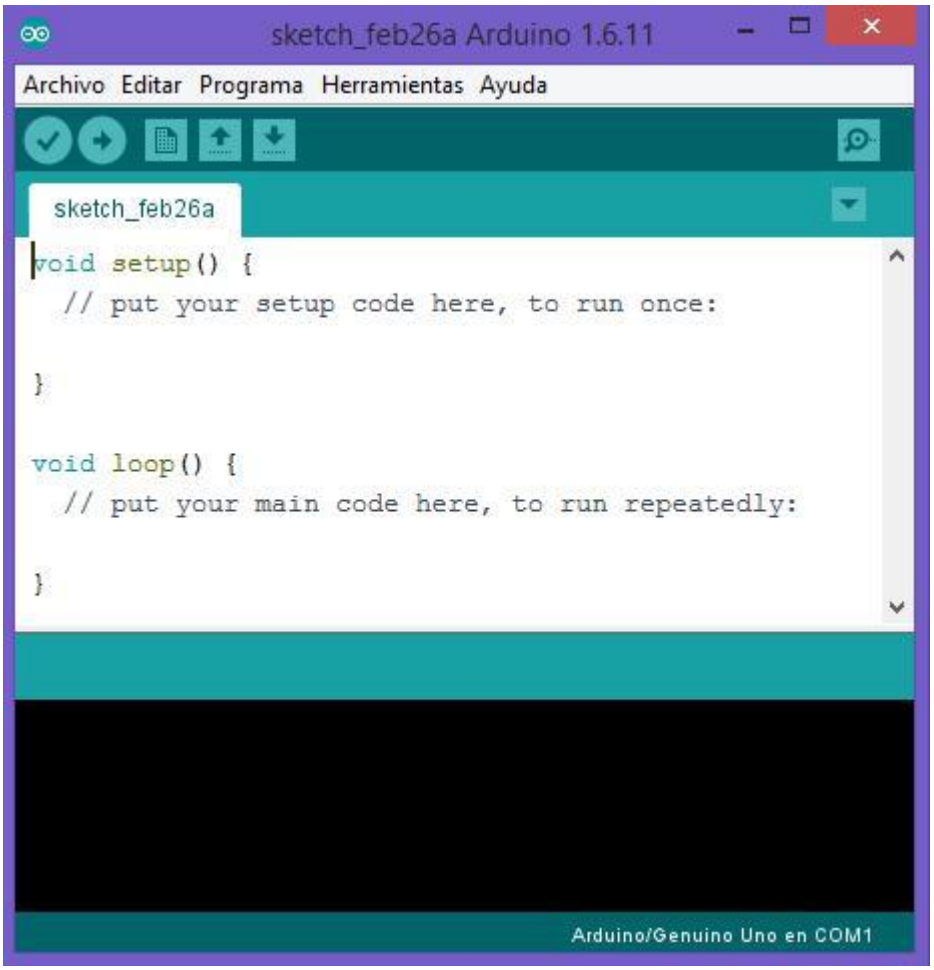


Figura 2. 16: Ventana principal del entorno de desarrollo de Arduino. Fuente: (Arduino, 2016)

2.10. Características del Hardware ArduinoATMega 328.

Como se mencionó anteriormente, la potencia de procesamiento de Arduino UNO es proporcionada por el microcontroladorATMega 328P. El diagrama

de pines de salida y el diagrama de bloques del microcontrolador en mención se muestran en las figuras 2.17 y 2.18. En esta sección,proporcionamosdetallesadicionales sobre los sistemas a bordo delmicrocontroladorATMega 328P.

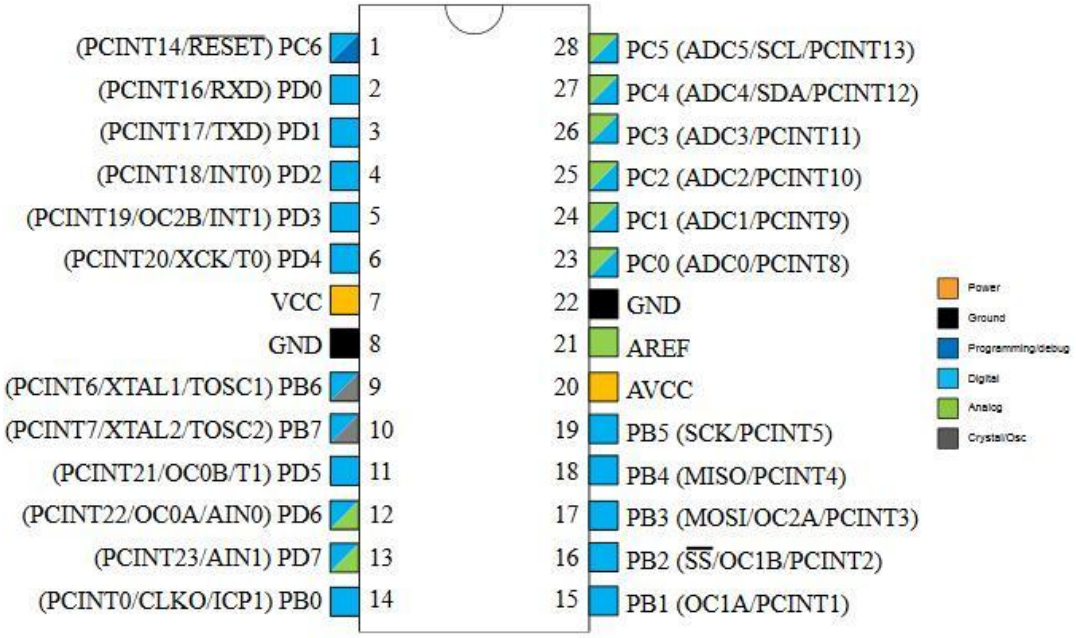


Figura 2. 17: Esquemático de los pines del microcontroladorATmega 328P
Fuente: (Atmel, 2014)

2.10.1. Memoria del microcontroladorATMega 328P.

El ATmega328 está equipado con tres secciones de memoria principales: (a) memoria de lectura programable borrable eléctricamente conocida como EEPROM, (b) memoria de acceso aleatorio estática, también conocida como SRAM, y (c) EEPROM byte direccionable para el almacenamiento de datos. A continuación, se describe brevemente cada uno de los componentes de memoria a su vez.

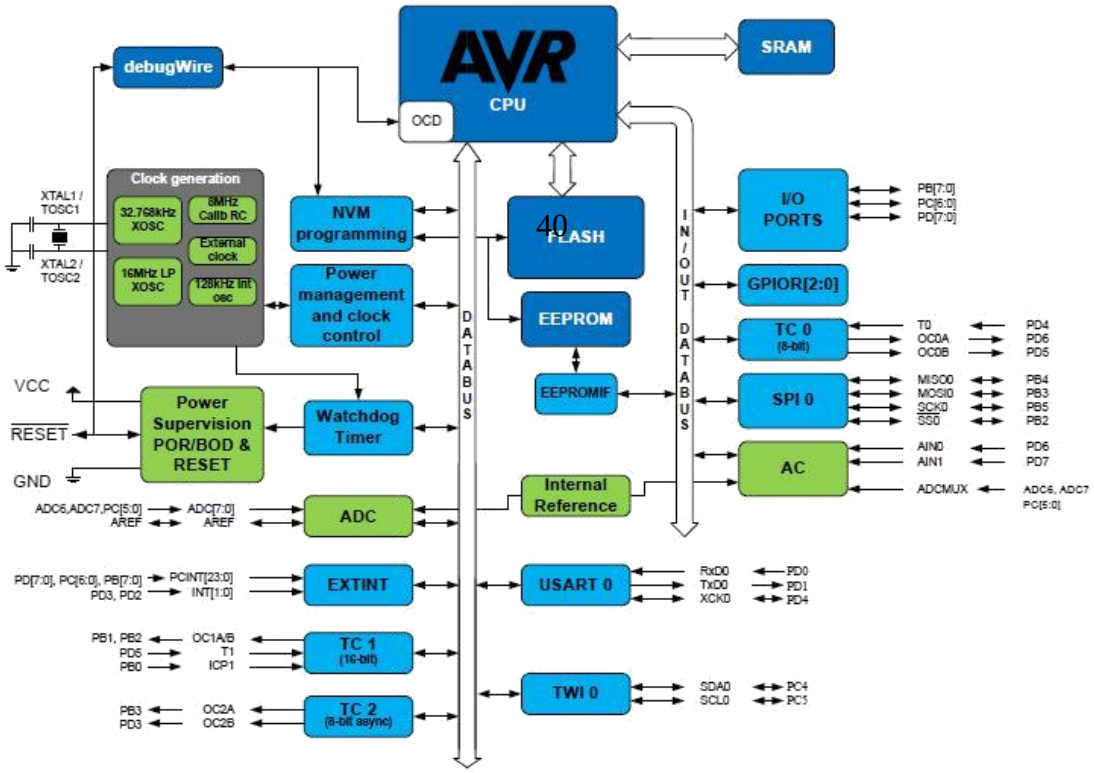


Figura 2. 18: Diagrama de bloques ATmega328.
Fuente: (Atmel, 2014)

2.10.1.1. Memoria Flash EEPROM programable en el sistema.

Este tipo de memoria se utiliza para almacenar programas. Se puede borrar y programar como una sola unidad. Además, si un programa requiere una tabla grande de constantes, puede ser incluido como una variable global dentro de un programa y programado en la memoria flash EEPROM con el resto del programa. La memoria Flash EEPROM es no volátil significa que los contenidos de la memoria se conservan cuando se pierde la potencia del microcontrolador. El microcontroladorATMega 328 está equipado con 32K bytes de memoria flash reprogramable. Este componente de memoria está

organizado en 16K ubicaciones con 16 bits en cada ubicación, tal como se muestra en la figura 2.19.

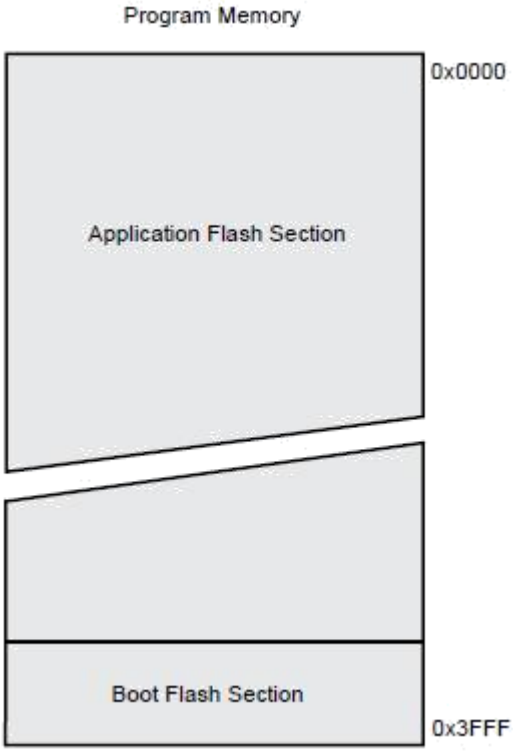


Figura 2. 19: Mapa del programa de memoria del microcontroladorATmega 328P.
Fuente: (Atmel, 2014)

2.10.1.2. Memoria EEPROM direccionable por bytes.

La memoria direccionable por bytes se utiliza para almacenar y recuperar permanentemente variables durante la ejecución del programa. También es una memoria no volátil. Es especialmente útil para registrar fallos en el sistema y fallos de datos durante la ejecución del programa. También es útil para almacenar datos que deben conservarse durante un fallo de alimentación, pero puede que tenga que cambiarse periódicamente. Los ejemplos en los que se utiliza este tipo de memoria se encuentran en aplicaciones para almacenar parámetros del sistema, combinaciones de

cerraduras electrónicas y secuencias electrónicas de desbloqueo de puertasde garaje automáticas. El ATmega328 está equipado con 1024 bytes deEEPROM.

2.10.1.3. Memoria de acceso aleatorio estática (SRAM).

La memoria RAM estática es volátil. Es decir, si el microcontrolador pierde potencia, se pierde el contenido de la memoria SRAM. Se puede escribir y leer durante la ejecución del programa. El microcontroladorATMega 328 está equipado con 2K bytes de SRAM. Una pequeña porción de la SRAM se reserva para los registros de uso general usados por el procesador y también para los subsistemas de E/S y de los periféricos a bordo del microcontrolador, tal como se puede observar en la figura 2.20.

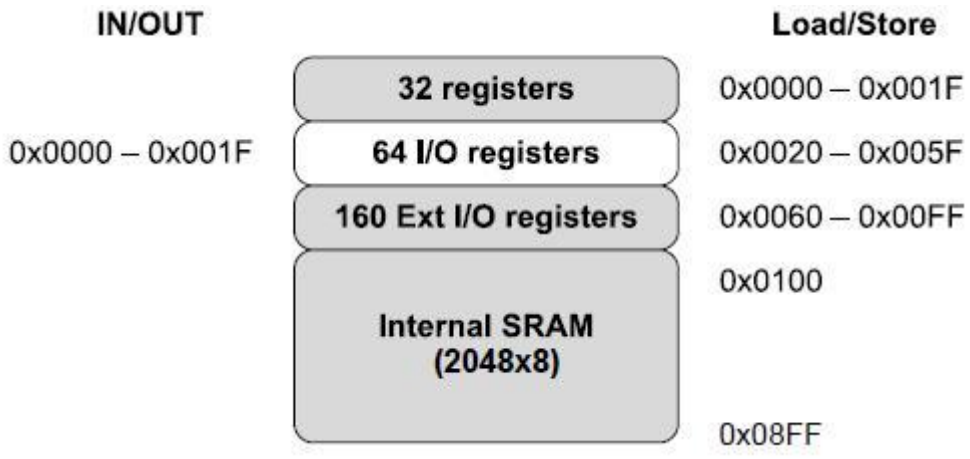


Figura 2. 20: Mapa de la memoria de datos del microcontroladorATmega 328P.
Fuente: (Atmel, 2014)

2.10.2. Sistema de puertos del microcontroladorATMega 328P.

El ATMega 328P de Atmel está equipado con cuatro puertos de 8 bits de propósito general, puertos digitales de entrada/salida (E/S) designados como PORTA, PORTB, PORTC y PORTD. Todos estos puertos también tienen funciones alternativas que se describirán más adelante. En estasección, nos concentramos en las características básicas de los puertos digitales de E/S. En la tabla 2.2 se muestran los puertos que tiene tres registros asociados con ATMega 328P.

- Puerto ‘x’ de registros de datos: utilizado para escribir datos de salida en el puerto.
- Registro de dirección de datos DDRx: se utiliza para establecer un pin de puerto específico en la salida (1) o en la entrada (0).
- Pin de entrada de direccionamiento PINx: se utiliza para leer datos de entrada desde el puerto.

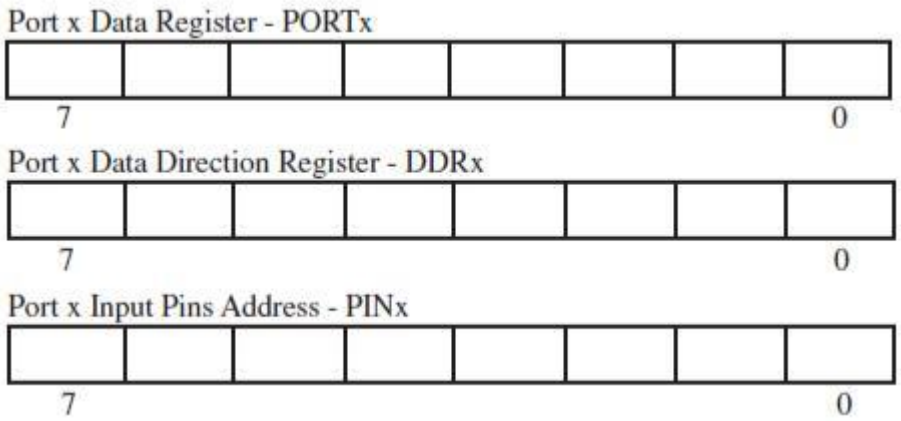


Figura 2. 21: Mapa del registro de puertos asociados de ATmega 328P. Fuente: (Atmel, 2014)

En la tabla 2.2 se describe los ajustes necesarios para configurar un pin de puerto específico a la entrada o salida. Si se selecciona para la entrada, el pin puede seleccionarse para un pin de entrada o para operar en

el modo de alta impedancia (Hi-Z). En el modo Hi-Z, la entrada aparece como alta impedancia a un pin particular. Si se selecciona para la salida, el pin puede configurarse adicionalmente para la lógica baja o la lógica alta.

Tabla 2. 2: Configuración de los pines de microcontroladorATMega 328P.

DDxn	PORTxn	PUD (in MCUCR)	I/O	Pull-up	Comment
0	0	X	Input	No	Tri-state (Hi-Z)
0	1	0	Input	Yes	Pxn will source current if ext. pulled low
0	1	1	Input	No	Tri-state (Hi-Z)
1	0	X	Output	No	Output Low (Sink)
1	1	X	Output	No	Output High (Source)

Fuente: (Atmel, 2014)

Los pines de puerto se configuran generalmente al principio de un programa para la entrada o la salida y sus valores iniciales se fijan entonces. Por lo general, los ocho pines para un puerto determinado se configuran simultáneamente.

CAPÍTULO 3: Diseño y Simulación de aplicaciones prácticas

En el presente capítulo se describen previamente las configuraciones del puerto COM virtual, obtención del archivo hexadecimal en Arduino IDE, y finalmente se realizan seis aplicaciones prácticas de programación de instrumentación virtual utilizando las plataformas de simulación, tales como: LabView, Isis Proteus y Arduino mediante la herramienta NI VISA de NationalInstrument.

3.1. Configuración del puerto COM virtual.

Usando la herramienta **Virtual Serial Port Driver** simulamos la interacción entre el software de simulación de circuitos electrónicos Proteus y el software de instrumentación Virtual Labview, ya que provee una creación, administración, y eliminación flexible de los puertos COM permitiendo el intercambio de datos mediante la interfaz de conexión virtual null modem (Virtual null-modem connection), tal como se muestra en la figura 3.1. Cabe señalar, que los datos serán enviados de forma asíncrona.

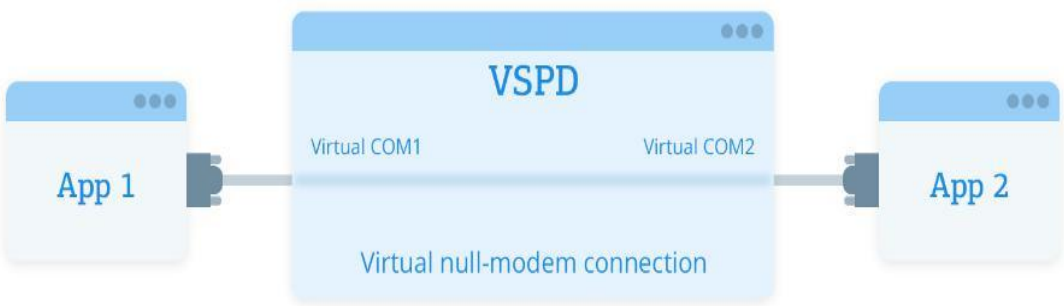


Figura 3. 1: Interfaz de conexión virtual -null modem-.
Elaborado por: Autor

3.2. Creación del puerto virtual.

En la interfaz procedemos a dar click a crear nuevos dispositivos (véase la figura 3.2), seleccionamos el tipo de conexión que se desea utilizar como el punto a punto que interconecta dos puertos entre sí, así logrando una comunicación óptima para enviar datos y recepción de los mismos.

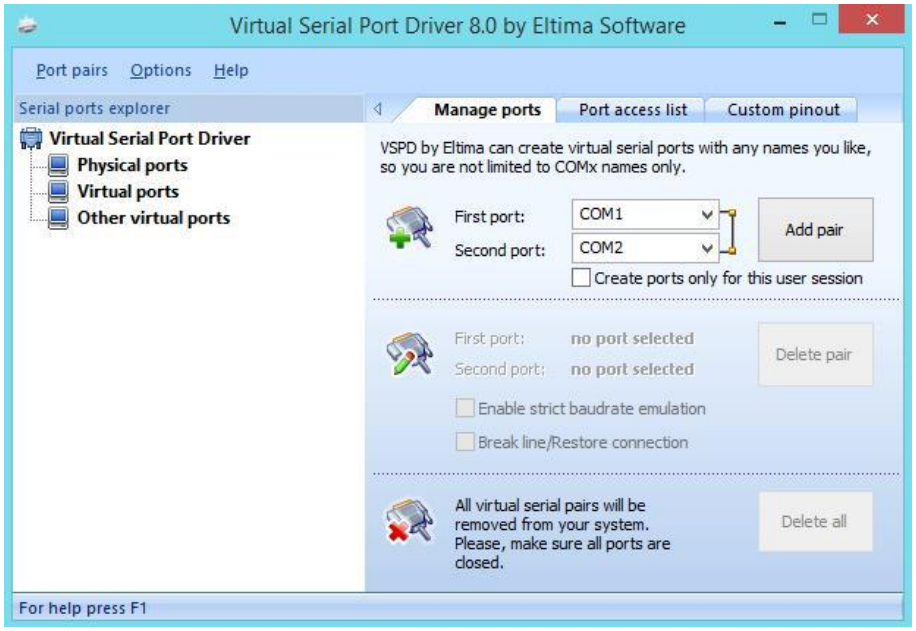


Figura 3. 2: Configuración del puerto serial virtual.
Elaborado por: Autor

3.3. Obtención del archivo Hexadecimal en ArduinoIde.

Para poder simular un proyecto en la placa Arduino UNO con Atmega 328p en Proteus hace falta el archivo con extensión ‘hex’ que se crea al compilar el programa. Al momento de compilar un archivo (programación realizada en lenguaje de alto nivel) en el ArduinoIde se crea en la carpeta un

archivo cuya extensión es ‘hex’, ubicado en la carpeta de archivos temporales del sistema y buscamos la carpeta <<build>>, en el mismo se encontrará el archivo compilado y su formato hex respectivamente.

3.4. Herramienta NI VISA de LabView.

La herramienta disponible en LabView se denomina arquitectura de software de instrumentos virtuales *(Virtual Instrument Software Architecture,VISA)*, cuyo estándar nos permite configurar, programar y solucionarproblemas de sistemas de instrumentación que comprenden interfaces GPIB, VXI, PXI, Serial, Ethernet y / o USB. VISA proporciona la interfaz de programación entre el hardware y entornos de desarrollo como Labview, LabWindows/CVI y Measurement Studio para Microsoft Visual Studio.

NI-VISA es la implementación de National Instruments del estándar VISA I/O. NI-VISA incluye bibliotecas de software, utilidades interactivas como NI I/O Trace y VISA Interactive Control y programas de configuración a través de Measurement&Automation Explorer para todas sus necesidades de desarrollo. NI-VISA es estándar en toda la línea de productos de National Instruments.

3.5. Desarrollo de aplicaciones prácticas de instrumentación virtual.

En esta sección se desarrollarán seis aplicaciones prácticas que integran plataformas de programación y simulación, como LabView, ArduinoIde e Isis Proteus, respectivamente. En la que se mostrará las

simulaciones virtuales realizadas sobre Isis Proteus utilizando la librería de Arduino Uno y finalmente lograr la comunicación serial con el programa de instrumentos virtuales de LabView.

3.5.1. Aplicación práctica 1: envió de caracteres Arduino a LabView.

El objetivo de esta práctica es conocer el funcionamiento básico del NI-VISA, así como, el desarrollo de la comunicación mediante puertos virtuales logrando la transmisión de datos desde el puerto hacia el entorno de instrumentación virtual desarrollado en LabView. En la figura 3.3 se muestra el diagrama esquemático con cada uno de los elementos que se usan para la simulación en Isis Proteus.

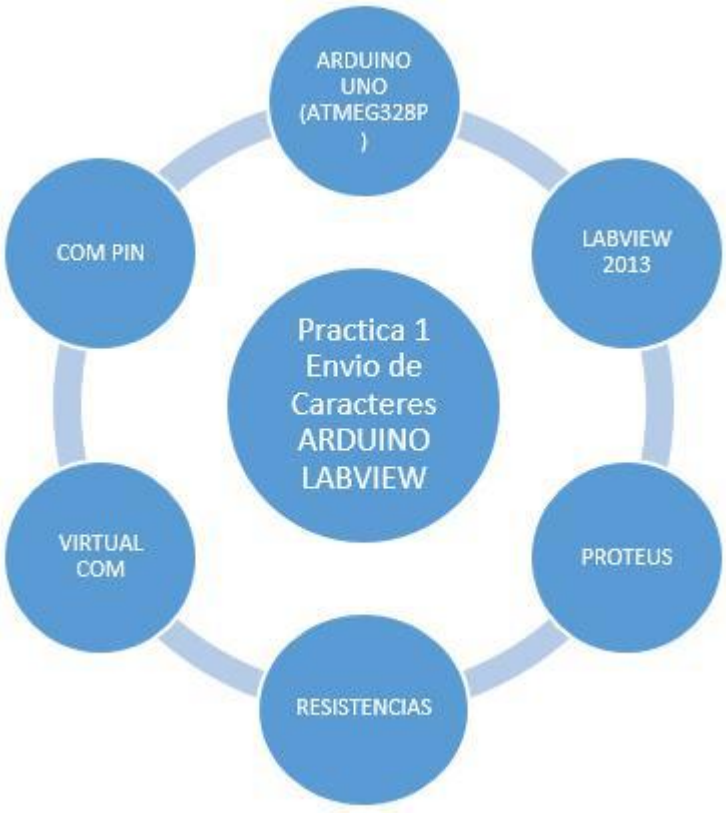


Figura 3. 3: Esquemático de elementos que intervienen en la aplicación práctica 1.
Elaborado por: Autor

Posteriormente, se realiza el diseño de simulación en Isis Proteus, cuyo diagrama esquemático consta de la siguiente arquitectura (véase la figura 3.4). El diseño realizado permite la lectura de los datos en el puerto virtual desde Isis Proteus, para posteriormente ser enviado estos datos al COM PIM (puerto de comunicación con periféricos externos y entornos de comunicación serial) que se conectará con el instrumento virtual diseñado en LabView.

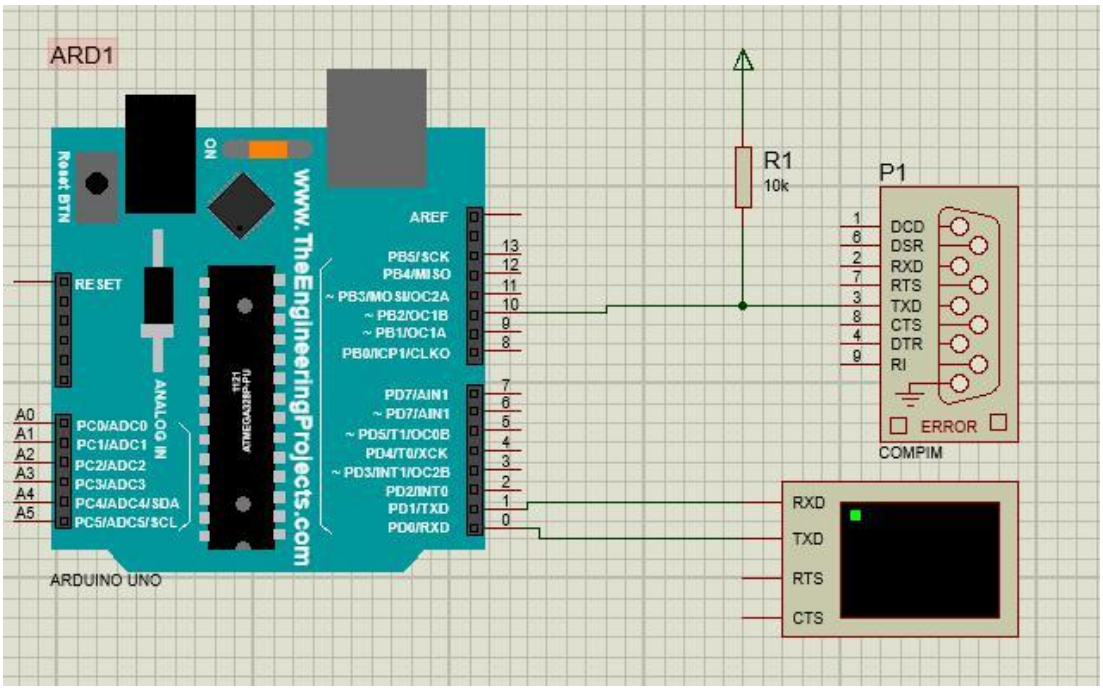


Figura 3. 4: Esquemático de simulación en Isis Proteus para aplicación práctica 1.
Elaborado por: Autor

A continuación, se realiza el algoritmo de programación en lenguaje de alto nivel C bajo la plataforma o entorno de desarrollo integrado *(IntegratedDevelopmentEnvironment, IDE)* de Arduino. El código de la aplicaciónpráctica 1 (véase la figura 3.5) permite realizar un mapeo del dato

ubicado en el buffer del puerto virtual COM y lo envía a LabView mediante un compin que se conectara al COM previamente creado.

```
#include <SoftwareSerial.h>
SoftwareSerial PuertoVirtual(11, 10);//RX,TX
void setup() {
  Serial.begin(9600);
  PuertoVirtual.begin(9600);
}
void loop() {
  while (Serial.available()) {
    PuertoVirtual.print((char)Serial.read());
  }
}
```

Figura 3. 5: Código para envio de caracteres de la aplicación práctica 1.
Elaborado por: Autor

El software de instrumentación virtual LabView diseñado y que se ilustra en la figura 3.6, el mismo que tomará los datos del puerto NI VISA y los mostrará en un label o string. En el diagrama de bloque deberá parametrizarse, es decir, cual puerto se usará como comunicación de NI VISA y que deberá hacer una lectura de la cantidad de bits que constarán en el buffer que serán enviados desde el puerto virtual.

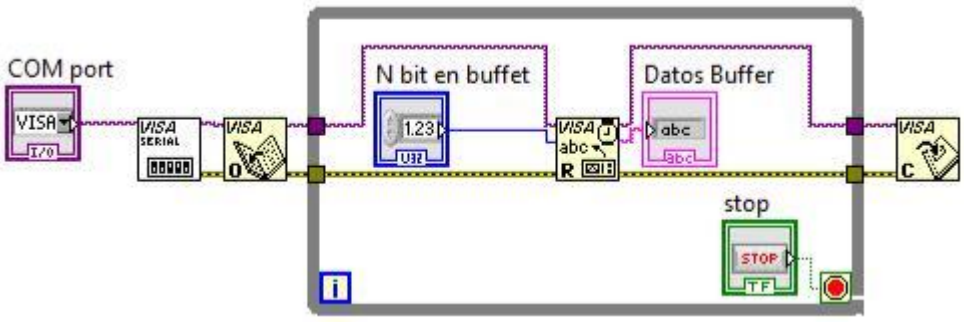


Figura 3. 6: Programación de instrumentación virtual de la aplicación práctica 1.
Elaborado por: Autor

Los resultados de la simulacion al momento de la ejecucion del programa se tendra en cuenta la configuracion de los diferentes perifericos como el compin del simulador de circuitos Isis Proteus (véase la figura3.7), el cual recibirá datos de un puerto serial enviando dicho dato al puerto virtual con LabView (véase la figura 3.8) y se haga la lectura de datos del puerto virtual en LabView.

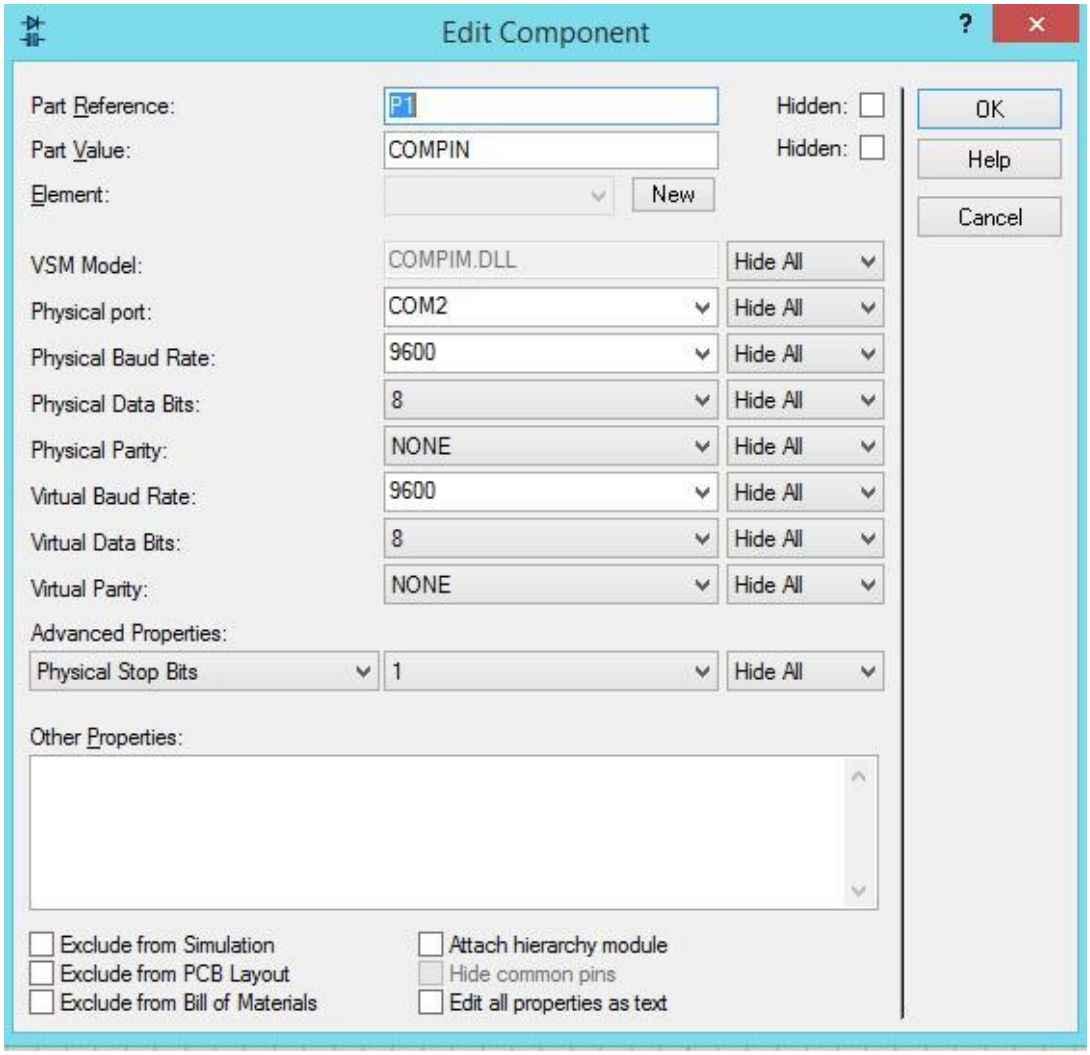
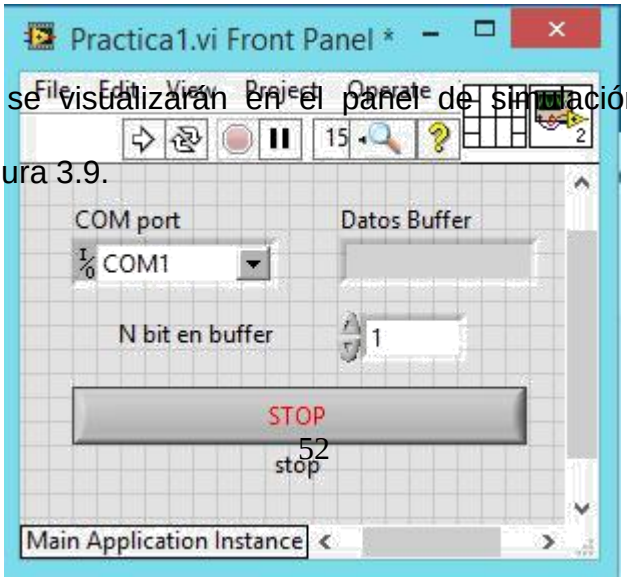


Figura 3. 7: Panel de configuracion del compin en el isisproteus práctica 1.
Elaborado por: Autor

Los datos se visualizarán en el panel de simulación tal como se muestra en la figura 3.9.



52

Figura 3. 8: Panel de simulación del instrumento virtual de la aplicación práctica.1
Elaborado por: Autor

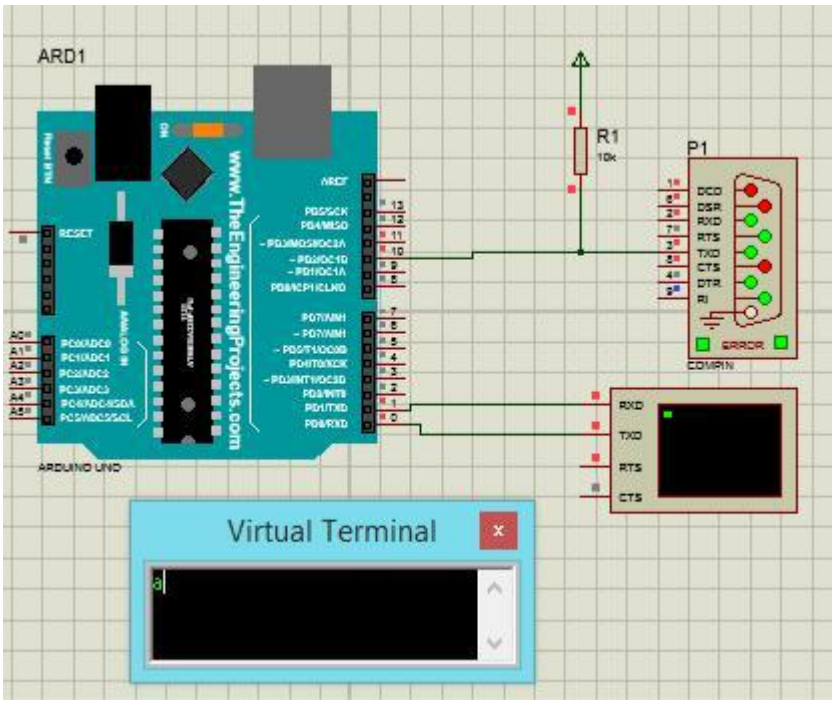


Figura 3. 9: Resultados obtenidos para la aplicación práctica 1.
Elaborado por: Autor

3.5.2. Aplicación práctica 2: control de sentido de motor de corriente continua.

El objetivo de esta práctica es la asignación de valores discretos para el control de un motor DC utilizando las herramientas, de instrumentación NI-VISA y simulación en Isis Proteus. En la figura 3.10 se muestra el esquemático de los elementos que intervienen en la implementación, pero que nos mostrara el control eficaz del mismo.

Figura 3. 10: Esquemático de elementos que intervienen en la aplicación práctica 2.
Elaborado por: Autor

El esquemático desarrollado en Isis Proteus consta de la siguiente arquitectura (véase la figura 3.11). Dicha simulación tendrá como entrada el puerto serial el cual servirá como esclavo de los datos enviados por el maestro, que en este caso sería LabView.

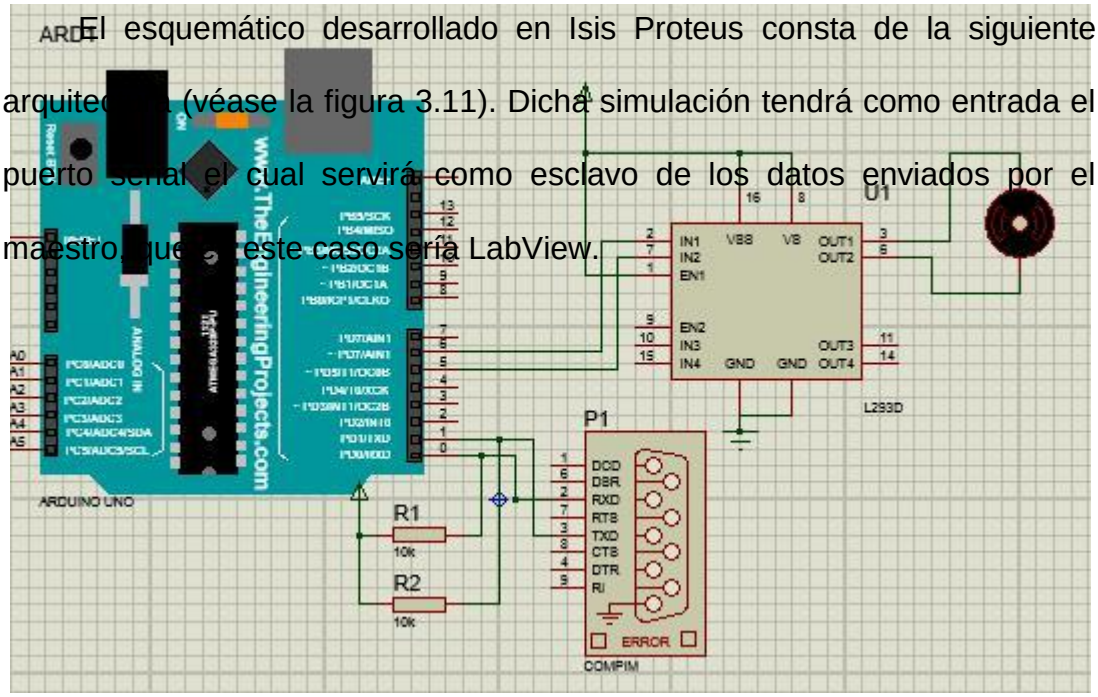


Figura 3. 11: Esquemático de simulación en Isis Proteus para aplicación práctica 2.
Elaborado por: Autor

A continuación, se realiza el planteamiento de una secuencia de funciones que se utilizara para el movimiento del motor, dichos algoritmos se plantean en lenguaje de alto nivel C bajo la plataforma o entorno de desarrollo integrado *(IntegratedDevelopmentEnvironment, IDE)* de Arduino. El código de la aplicación práctica 2 (véase la figura 3.12) permite realizar un mapeo del dato ubicado en el buffer del puerto virtual COM y lo envía a LabView mediante un compin que se conectara al COM previamente creado.

El software de instrumentación virtual LabView diseñado y que se ilustra en la figura 3.13, permitirá mapear el estado actual de los botones y enviar comandos mediante NI VISA hacia el puerto virtual.

```
char datoBuffer;
int a1 = 5;
int a2 = 6;
void setup() {
  Serial.begin(9600);//configuracion baudios
  pinMode(a1,OUTPUT);//Seteo como salida
  pinMode(a2,OUTPUT);//Seteo como salida
}
void loop() {
  while (Serial.available()) {
    datoBuffer = Serial.read();
    if (datoBuffer = 'F') {
      digitalWrite(a1, HIGH); digitalWrite(a2, LOW);
    }
    else if (datoBuffer = 'B') {
      digitalWrite(a1, LOW); digitalWrite(a2, HIGH);
    }
    else if (datoBuffer = 'S') {
      digitalWrite(a1, LOW); digitalWrite(a2, LOW);
    }
  }
}
```

55

Figura 3. 12: Código de control de motor DC para la aplicación práctica 2.
Elaborado por: Autor

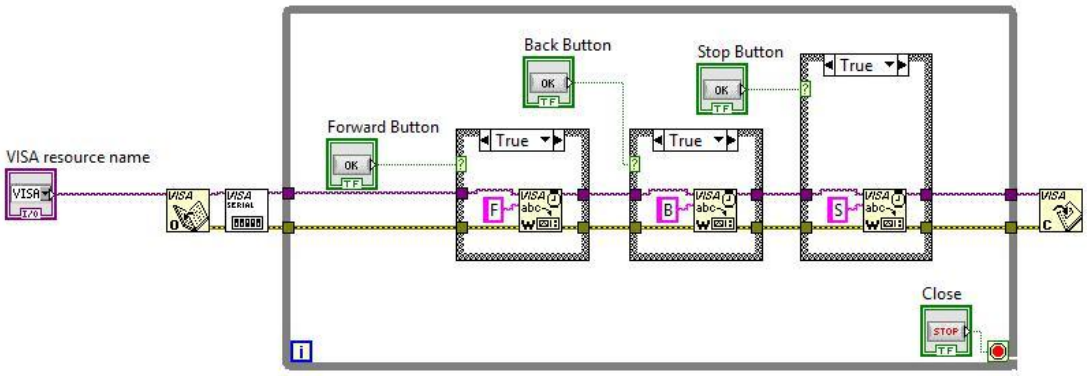


Figura 3. 13: Programación de instrumentación virtual de la aplicación práctica 2.
Elaborado por: Autor

Los resultados de la simulacion al momento de la ejecucion del programa se captará el estado de los pulsadores en el panel de frontal, tal como se ilustra en la figura 3.14, el cual enviará datos a Isis Proteus que interpreta los caracteres y se visualizará en el movimiento del motor (referencia al esquematico de la figura 3.11).

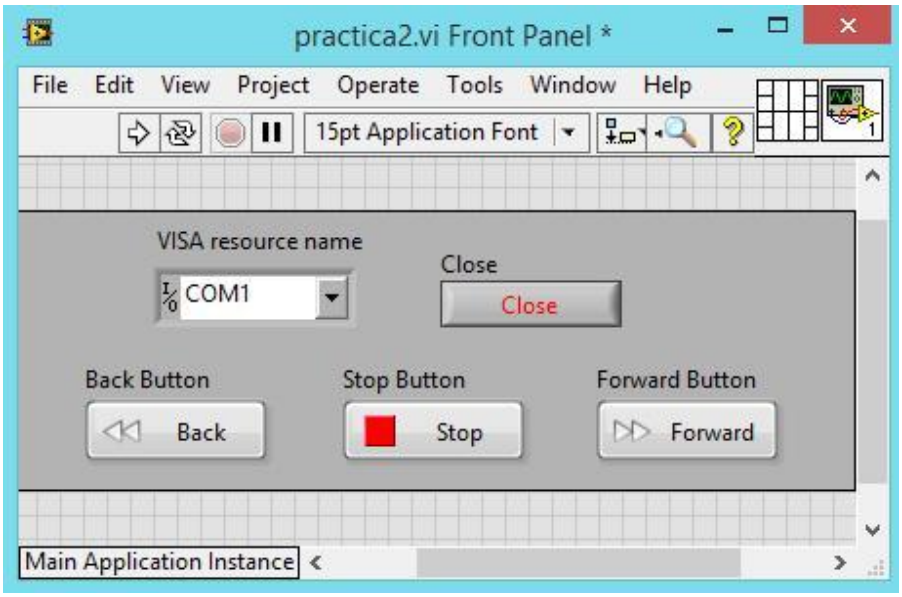
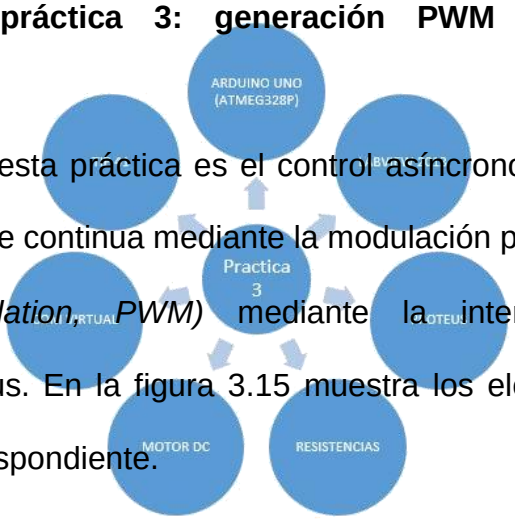


Figura 3. 14: Panel de simulación del instrumento virtual de la aplicación práctica 2.
Elaborado por: Autor

3.5.3. Aplicación práctica 3: generación PWM para control de velocidad.



El objetivo de esta práctica es el control asíncrono de la velocidad de un motor de corriente continua mediante la modulación por ancho de pulsos *(Pulse WidthModulation, PWM)* mediante la interface NI VISA y simuladorIsis Proteus. En la figura 3.15 muestra los elementos necesarios para el diseño correspondiente.

Figura 3. 15: Esquemático de elementos que intervienen en la aplicación práctica 3.
Elaborado por: Auto

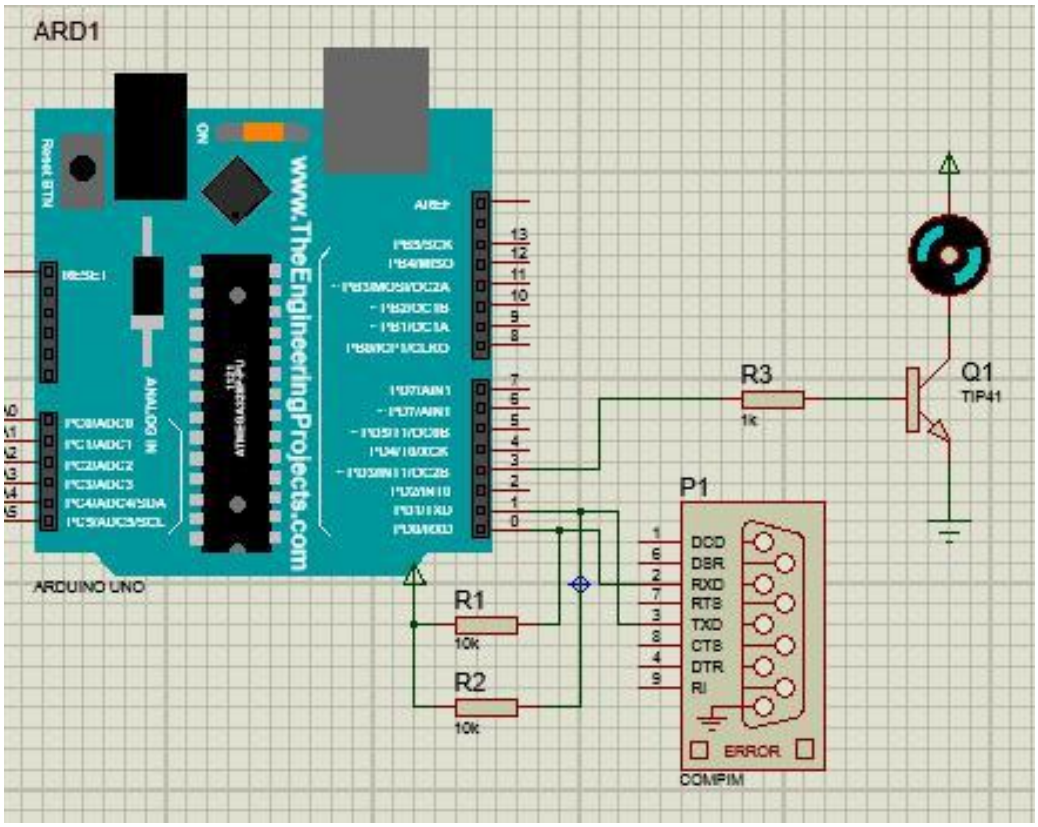


Figura 3. 16: Esquemático de simulación en Isis Proteus para aplicación práctica 3.
Elaborado por: Autor

En la figura 3.16 se muestra el diseño esquemático en Isis Proteus. Dicha simulación tendrá un puerto virtual el cual estará entrelazado con la interfaz LabView, logrando así enviar el PWM para controlar la velocidad del motor DC.

El código de programación para esta aplicación se lo realizó en lenguaje de alto nivel C como se observa en la figura 3.17. Se obtendrá el valor enviado desde LabView y realizará una escritura en el pin asignado utilizando PWM para controlar la velocidad del motor mediante anchos de pulsos respectivos a través de la comunicación virtual COM.

```
#include <SoftwareSerial.h>
char datoBuffer;
int pwmpin=3;
void setup() {
  Serial.begin(9600);//Configuracion baudios
  pinMode(pwmpin,OUTPUT);//Seteo pwm
}
void loop() {
  while (Serial.available()) {
    datoBuffer = Serial.read();
  analogWrite(pwmpin,(unsigned int)datoBuffer);
  }
}
```

Figura 3. 17: Código de control de motor DC usando PWM para la aplicación práctica 3.

Elaborado por: Autor

A continuación, en la figura 3.18 se muestra el diseño el diagrama de bloques y el panel realizado mediante el software de instrumentación virtual LabView. Esta aplicación se encargará de enviar datos de la slider barmediante la herramienta VISA y dicho dato permitirá controlar la velocidad del motor mediante PWM.

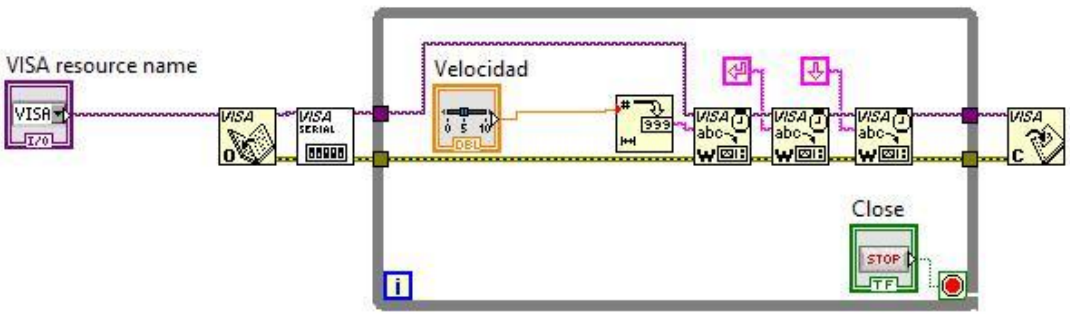


Figura 3. 18: Programación de instrumentación virtual de la aplicación práctica 3.
Elaborado por: Autor

Al momento de la ejecución del instrumento virtual diseñado (véase la figura 3.19) servirá como maestro el cual envía los datos desde slider hacia el dispositivo Arduino y este ejecuta el control de la velocidad en el esquemático respectivo que se mostró en la figura 3.16.

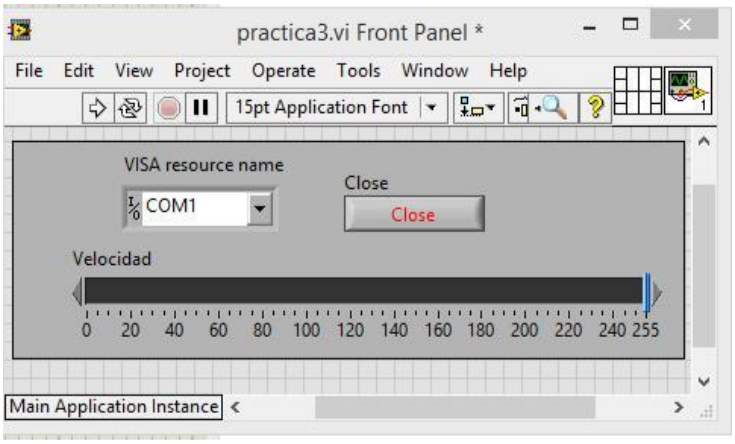


Figura 3. 19: Panel Frontal de instrumentación virtual de la aplicación práctica 3.
Elaborado por: Autor

3.5.4. Aplicación práctica 4: control discreto de salidas digitales.

El objetivo de esta práctica es el control de señales discretas para el encendido y apagado de diodos leds mediante pines digitales utilizando la herramienta NI-visa y el simulador Isis Proteus. En la figura 3.20 se muestran los elementos utilizados en la implementación virtual para el control discreto de salidas digitales.

Figura 3. 20: Esquemático de elementos que intervienen en la aplicación práctica 4.
Elaborado por: Autor

En la figura 3.21 se muestra el diseño esquemático en Isis Proteus. Dicha simulación tendrá un puerto virtual el cual estará entrelazado con la interfaz virtual diseñada en LabView, el mismo que servirá como visualizador de las asignaciones discretas en los pines digitales previamente configurados.

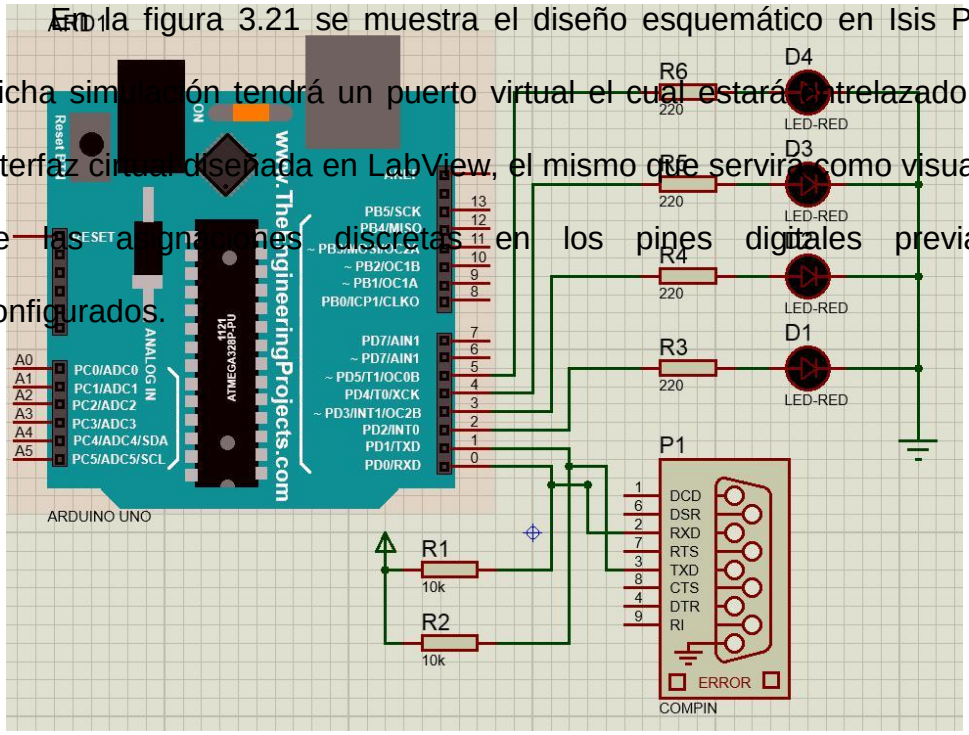


Figura 3. 21: Esquemático de simulación en Isis Proteus para aplicación práctica 4.
Elaborado por: Autor

El código de la aplicación práctica 4 obtendrá un dato char (carácter) enviado desde LabView y realiza una comparación con una base de comparación mediante una clase if y realizará la asignación respectiva de un


```
#include <SoftwareSerial.h>
char datoBuffer;
int led1 = 2; int led2 = 3;int led3 = 4; int led4 = 5;
```

alto o un bajo respectivamente. Primero se configura las variables que se usarán en el programa (véase código en la figura 3.22).

```
void setup() {
  Serial.begin(9600);//Configuración Baudios
  pinMode(led1, OUTPUT);  pinMode(led2, OUTPUT);
  pinMode(led3, OUTPUT);  pinMode(led4, OUTPUT);
  digitalWrite(led1, LOW);digitalWrite(led2, LOW);
```

Figura 3. 22: Código de programación de configuración de variables para aplicación práctica 4.
Elaborado por: Autor

Después, se asigna el inicio de la comunicación serial y configuración de pines de entrada y salida. Se inicializa las variables en 0 lógico, tal como se muestra el código de programación en la figura 2.23.

Figura 3. 23: Código de programación de comunicación serial y configuracion de pines de E/S para aplicación práctica 4.
Elaborado por: Autor

Finalmente, se realiza una comparación siempre que el puerto serial lea un dato en el buffer y compara dicho valor adquirido con una base de posibles casos para prender y apagar los leds.

```
void loop() {
  while(Serial.available()){
    datoBuffer = (char)Serial.read();

        if (datoBuffer == 'a') {digitalWrite(led4, LOW);digitalWrite(led3, LOW);digitalWrite(led2, LOW);digitalWrite(led1, LOW);}
   else if (datoBuffer == 'b') {digitalWrite(led4, LOW);digitalWrite(led3, LOW);digitalWrite(led2, LOW);digitalWrite(led1, HIGH);}
   else if (datoBuffer == 'c') {digitalWrite(led4, LOW);digitalWrite(led3, LOW);digitalWrite(led2, HIGH);digitalWrite(led1, LOW);}
   else if (datoBuffer == 'd') {digitalWrite(led4, LOW);digitalWrite(led3, LOW);digitalWrite(led2, HIGH);digitalWrite(led1, HIGH);}
   else if (datoBuffer == 'e') {digitalWrite(led4, LOW);digitalWrite(led3, HIGH);digitalWrite(led2, LOW);digitalWrite(led1, LOW);}
   else if (datoBuffer == 'f') {digitalWrite(led4, LOW);digitalWrite(led3, HIGH);digitalWrite(led2, LOW);digitalWrite(led1, HIGH);}
   else if (datoBuffer == 'g') {digitalWrite(led4, LOW);digitalWrite(led3, HIGH);digitalWrite(led2, HIGH);digitalWrite(led1, LOW);}
   else if (datoBuffer == 'h') {digitalWrite(led4, LOW);digitalWrite(led3, HIGH);digitalWrite(led2, HIGH);digitalWrite(led1, HIGH);}
   else if (datoBuffer == 'i') {digitalWrite(led4, HIGH);digitalWrite(led3, LOW);digitalWrite(led2, LOW);digitalWrite(led1, LOW);}
   else if (datoBuffer == 'j') {digitalWrite(led4, HIGH);digitalWrite(led3, LOW);digitalWrite(led2, LOW);digitalWrite(led1, HIGH);}
   else if (datoBuffer == 'k') {digitalWrite(led4, HIGH);digitalWrite(led3, LOW);digitalWrite(led2, HIGH);digitalWrite(led1, LOW);}
   else if (datoBuffer == 'l') {digitalWrite(led4, HIGH);digitalWrite(led3, LOW);digitalWrite(led2, HIGH);digitalWrite(led1, HIGH);;}
   else if (datoBuffer == 'm') {digitalWrite(led4, HIGH);digitalWrite(led3, HIGH);digitalWrite(led2, LOW);digitalWrite(led1, LOW);}
   else if (datoBuffer == 'n') {digitalWrite(led4, HIGH);digitalWrite(led3, HIGH);digitalWrite(led2, LOW);digitalWrite(led1, HIGH);}
   else if (datoBuffer == 'o') {digitalWrite(led4, HIGH);digitalWrite(led3, HIGH);digitalWrite(led2, HIGH);digitalWrite(led1, LOW);}
   else if (datoBuffer == 'p') {digitalWrite(led4, HIGH);digitalWrite(led3, HIGH);digitalWrite(led2, HIGH);digitalWrite(led1, HIGH);}

  }}
```

Figura 3. 24: Código de programación de comunicación serial y configuracion de pines de E/S para aplicación práctica 4.
Elaborado por: Autor

En la figura 3.25 se muestra el diseño del software de instrumentación virtual LabView que se encargará de enviar el dato de cada botón respectivamente, realizando un mapeo de cada estado. Al tener 4 pulsantes, se obtendrá 16 posibles combinaciones y cada una de estas combinacionesenviará un dato discreto por el buffer, el cual será enviado mediante el puerto virtual.

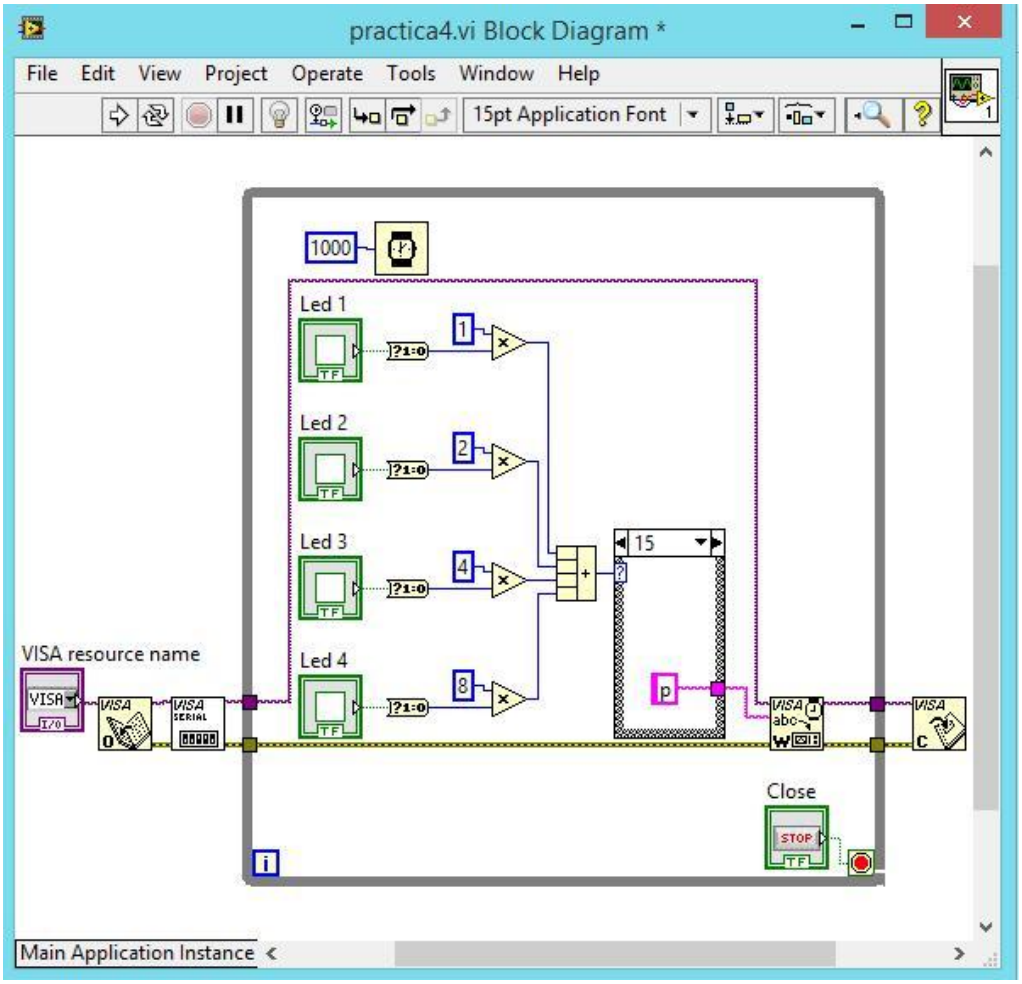


Figura 3. 25: Programación de instrumentación virtual de la aplicación práctica 4.
Elaborado por: Autor

Al momento de la ejecucion desde en panel de control diseñado (vease la figura 3.26) servirá como maestro, enviando valores de acuerdo a los estado de los botones y también envia datos de acuerdo al numero de combinaciones pertinentes configuradas previamente en el esqeumático diseñado con Arduino, tal como se mostró en la figura 3.21.

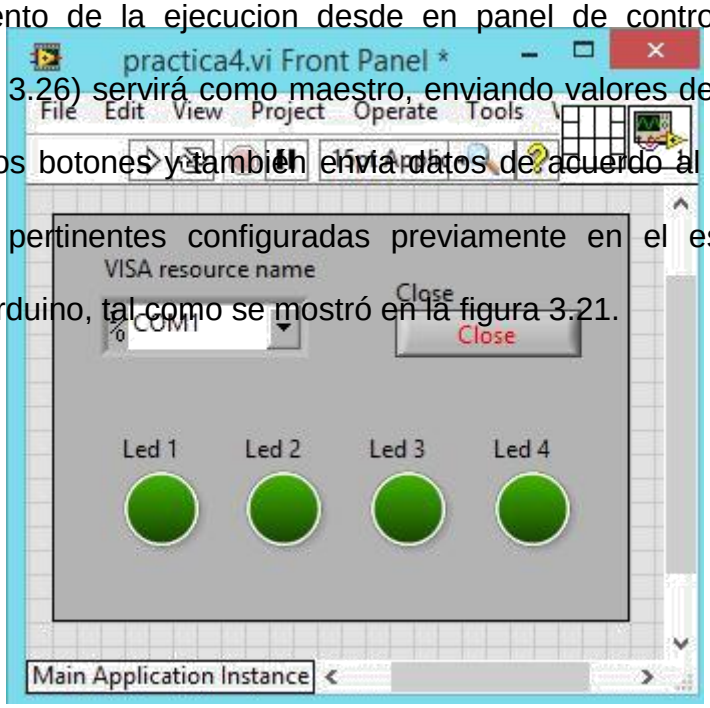


Figura 3. 26: Panel frontal de instrumentación virtual de la aplicación práctica 4.
Elaborado por: Autor

3.5.5. Aplicación práctica 5: llenado de tanque.

El objetivo de esta práctica es la asignación de valores en un tank bar propia de la librería de LabView. Como maestro se tendrá al microcontroladorAtmega 328p de Arduino que enviará el dato mediante la herramienta NI-VISA y el simulador Isis Proteus. En la figura 3.27 se muestran los elementos para la simulación virtual.

La figura 3.28 muestra el diseño esquemático en Isis Proteus. Se utiliza para esta simulación un puerto virtual el cual estará entrelazado con la interfaz LabView la cual servirá para adquirir los datos del potenciómetro y su respectiva conversión análoga a digital.

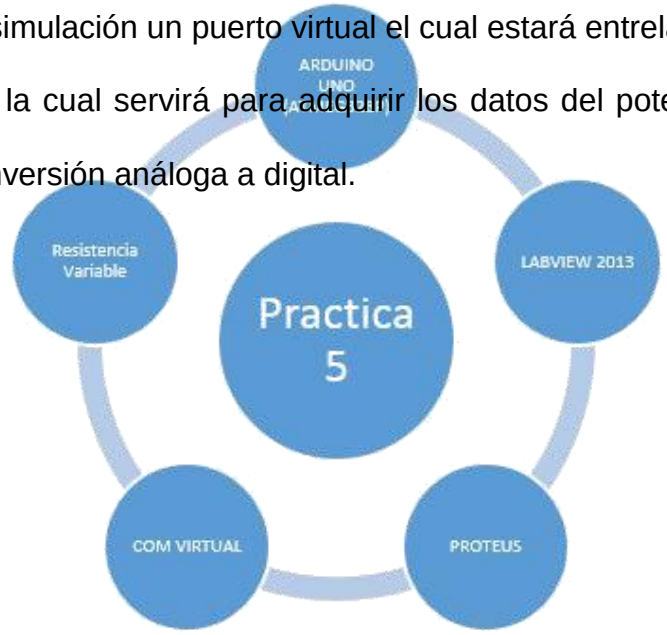


Figura 3. 27: Esquemático de elementos que intervienen en la aplicación práctica 5.
Elaborado por: Autor

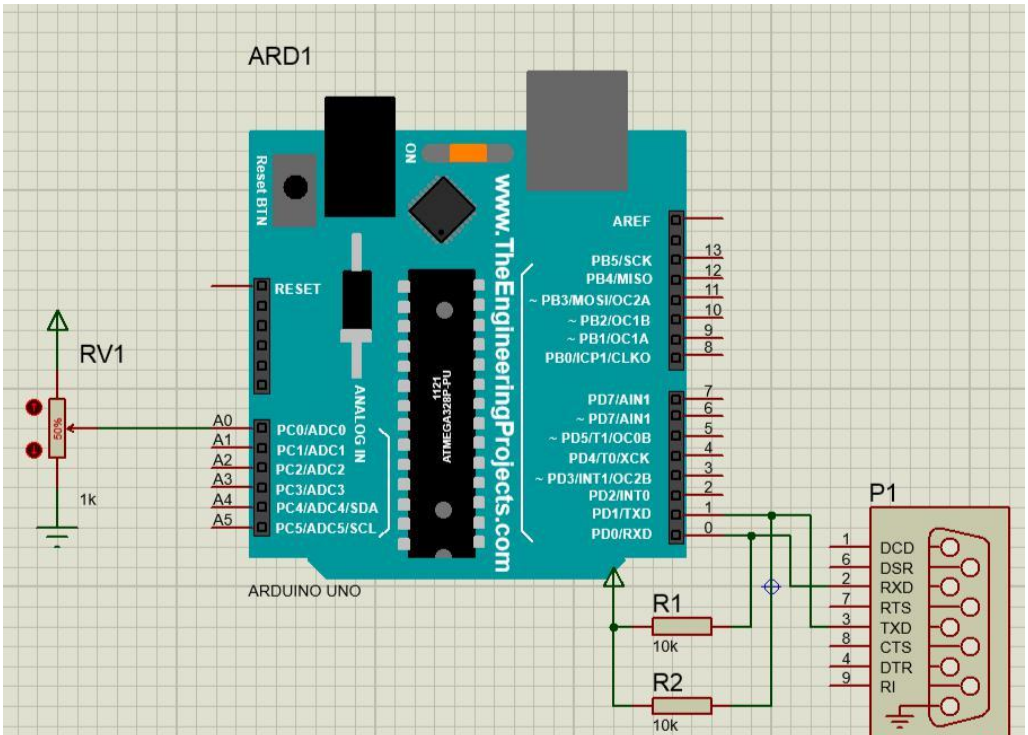


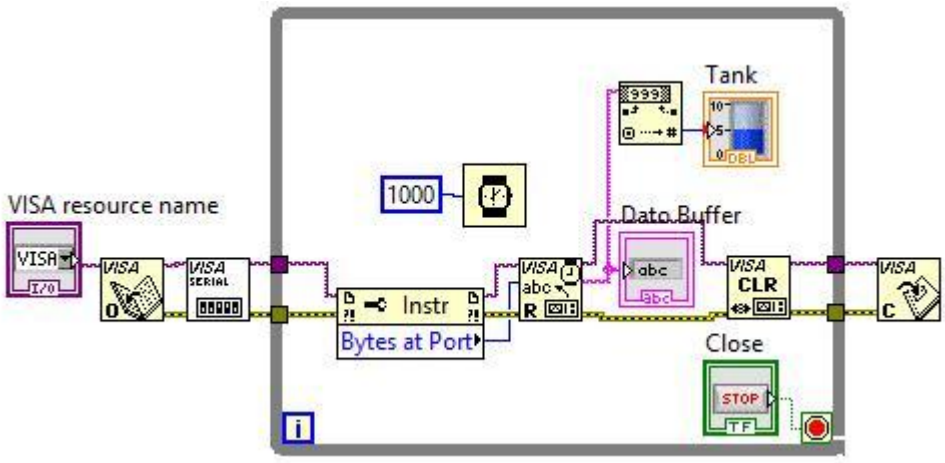
Figura 3. 28: Esquemático de simulación en Isis Proteus para aplicación práctica 5.
Elaborado por: Auto

El código de la práctica 5 (ver figura 3.29) obtendrá un dato mediante la conversión A/D, el mismo que será procesado y enviado por medio del puerto COM virtual desde Isis Proteus hacia el panel frontal de LabView.

```
const int analogInPin = A0;
int DatoAnalogico = 0;        // value read
void setup() {
  Serial.begin(9600);
}
void loop() {
  DatoAnalogico  = analogRead(analogInPin);
  Serial.println(DatoAnalogico);
  delay(2);
}
```

Figura 3. 29: Código de control para llenado de un tanque para la práctica 5.
Elaborado por: Autor

En la figura 3.30 se muestra el diseño del instrumento virtual LabView que se encargará de la lectura del dato actual en el buffer, que convertirá dicho dato string a un dato numérico para ser visualizado por un slider bar.



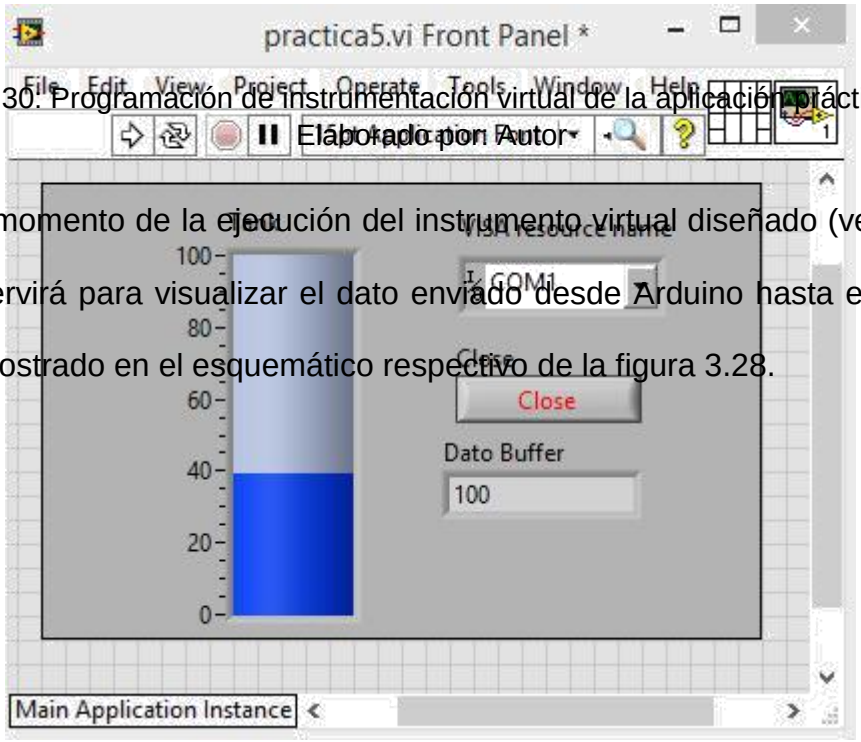


Figura 3. 30: Programación de instrumentación virtual de la aplicación práctica 5.
Elaborado por: Autor

Al momento de la ejecución del instrumento virtual diseñado (ver figura 3.31), servirá para visualizar el dato enviado desde Arduino hasta el puerto virtual mostrado en el esquemático respectivo de la figura 3.28.

Figura 3. 31: Panel frontal de instrumentación virtual de la aplicación práctica 5.
Elaborado por: Autor

3.5.6. Aplicación práctica 6: Sensor de Temperatura LM35.

El objetivo de esta práctica es la lectura y comprensión de un sensor LM35 el cual tiene una resolusion de 10 milivoltios por grado celcius captado. El nucleo procesador sera un microcontroladorAtmega 328p de Arduino que enviará el dato mediante la herramienta NI-VISA y el simulador Isis Proteus. En la figura 3.32 se muestran los elementos para la simulación virtual.

La figura 3.33 se muestra el diseño esquemático en Isis Proteus. Para esta simulación se utiliza un puerto virtual, que estará entrelazado con la interfaz LabView y que servirá para enviar datos luego del procesamiento del mismo, y finalmente captar el valor de la temperatura obtenida por el sensor.

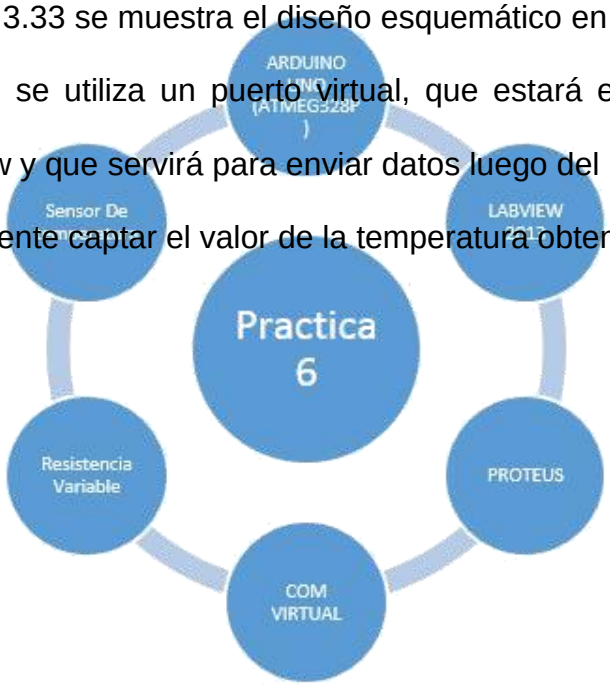


Figura 3. 32: Esquemático de elementos que intervienen en la aplicación práctica 6.
Elaborado por: Autor

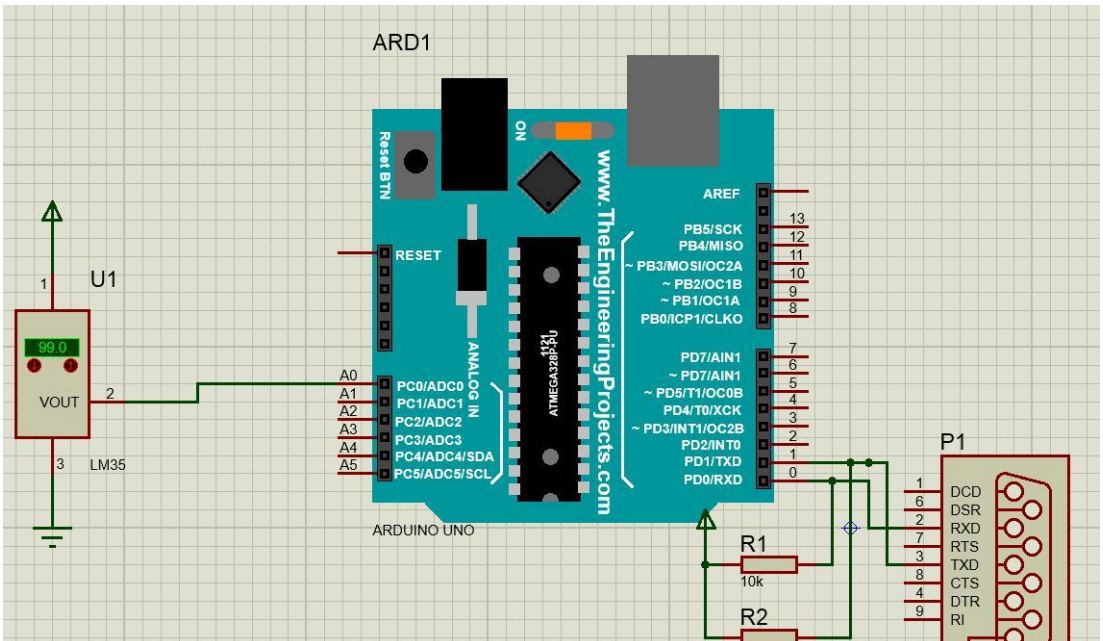


Figura 3. 33: Esquemático de simulación en Isis Proteus para aplicación práctica 6.
Elaborado por: Autor

El código de la práctica 6 primero se asigna variables como pin de entrada análoga y la variable donde se guarda dicho dato.

```
float Temperatura;
int Sensor = A0;
```

En la clase de configuración o voidsetup se designará el número de baudios de transmisión serial a trabajar, también la referencia de la conversión análoga del sensor es 10 milivoltios por grados, que se utiliza como referencia interna correspondiente al mismo microcontrolador.

```
void setup()
{
analogReference(INTERNAL);
Serial.begin(9600);
}
```

En la clase principal se obtendrá un dato mediante la conversión A/D, el mismo que será procesado y con una fórmula se divide 1.1V sobre la resolución de 10 bits que seria 1024. Cada paso en la lectura analógica es igual a aproximadamente 0.001074V = 1.0742 mV. Si 10mV es igual a 1 °C, entonces 10/1.0742=~9.31. Por lo tanto, para cada cambio de 9.31 en la lectura analógica, hay un grado de cambio de temperatura y enviado por medio del puerto COM virtual desde Isis Proteus hacia el panel frontal de LabView.

```
void loop()
{
Temperatura = analogRead(Sensor);
Temperatura = Temperatura / 9.31;
Serial.println(Temperatura);
delay(400);
}
```

En la figura 3.34 se muestra el diseño del software de instrumentación virtual LabView que se encargará de la lectura del dato. Luego de la conversión se procederá a comparar con valores constantes para visualizar una alarma siempre y cuando la temperatura sea alta o baja o estable.

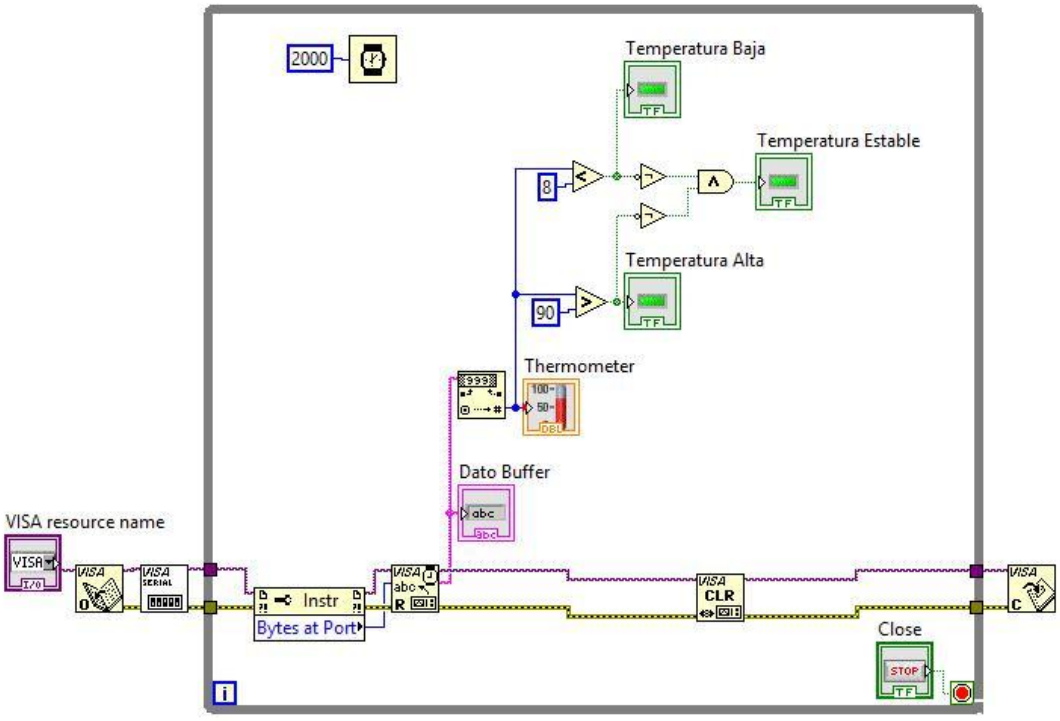


Figura 3. 34: Programación de instrumentación virtual de la aplicación práctica 6.
Elaborado por: Autor

En la figura 3.35 se muestra la ejecución del panel de control diseñado en LabView, que permite visualizar los datos y las condiciones previamente

establecidas desde el microcontroladorArduino hacia el puerto virtual (véase la figura 3.33).

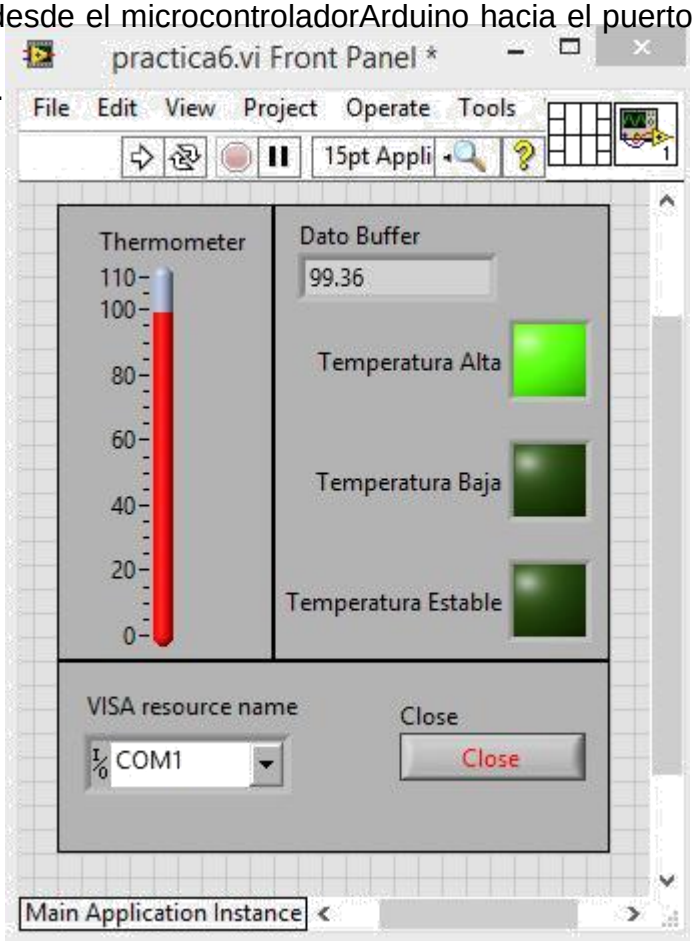


Figura 3. 35: Panel frontal de instrumentación virtual de la aplicación práctica 6.
Elaborado por: Autor

CAPÍTULO 4: Conclusiones y Recomendaciones.

4.1. Conclusiones.

- Los fundamentos básicos de microcontroladores y de la familia de ATMega del fabricante Atmel fueron de gran utilidad antes de implementar las aplicaciones prácticas mediante el uso de LabView, Isis Proteus y Arduino IDE.

- Las aplicaciones prácticas diseñadas permitieron ver la funcionalidad y operatividad de la comunicación entre la plataforma de instrumentación virtualLabView y el programa de simulación Isis Proteus, que dependieron del programa de código abierto Arduino.

- Durante las pruebas de ejecución en tiempo real de las aplicaciones prácticas virtuales se pudo constatar el correcto funcionamiento de cada aplicación desarrollada.

4.2. Recomendaciones.

- Desarrollar nuevas propuestas de trabajos de titulación que permitan utilizar sistemas de monitoreo a nivel Scada sobre la plataforma de instrumentos virtuales LabView.

- Impulsar la adquisición de licencias profesionales de LabView e Isis Proteus para la formación de profesionales en Ingeniería Electrónica en Control y Automatismo.

Bibliografía

Alepuz S., H., Giménez E., H., Jordá F., J., Ripoll G., F., Saval S., P., Serrano R., J., &Talens N., J. (2006). Bus CAN. UPV. Recuperado a partir de http://server-die.alc.upv.es/asignaturas/PAEEES/2005-06/A03-A04%20-%20Bus%20CAN.pdf

Arduino. (2013). Arduino UNO ReferencedDesign. Recuperado el 3 de enero de 2017, a partir de https://www.arduino.cc/en/uploads/Main/arduino-uno-schematic.pdf

Arduino. (2016). Arduino - ArduinoBoard Uno. Recuperado el 12 de agosto de 2016, a partir de https://www.arduino.cc/en/Main/ArduinoBoardUno

Atmel. (2010). 8-bit AVR Microcontroller with 16KBytes In-System Programmable Flash. Recuperado a partir de http://www.atmel.com/Images/2466S.pdf

Atmel. (2014). 8 bit AVR Microcontroller: ATmega328-328P. Recuperado el 16 de enero de 2017, a partir de http://www.atmel.com/Images/Atmel-42735-8-bit-AVR-Microcontroller-ATmega328-328P_Datasheet.pdf

Chenyang, L. (2014). DSP Processors Hit the Mainstream. Recuperado el 10 de enero de 2017, a partir de http://www.cse.wustl.edu/~lu/cse467s/slides/dsp.pdf

Hossain, B., Shourov, S., Rana, M., &Anower, S. (2015). Matlab Guidance Based Smart Gas Leakage Detection and Security System Using Analog to Digital Technique. *International Journal of Smart Home*, *9*(4), 13–24.

López P., E. (2014). Protocolo SPI (Serial Peripherical Interface): Teoría y Aplicaciones. Ingeniería en Microcontroladores. Recuperado a partir de http://www.i-micro.com/pdf/articulos/spi.pdf

Morles, V. (2002). Sobre la metodología como ciencia y el método científico: un espacio polémico. *Revista de Pedagogía*, *23*(66), 121–146.

Pacheco H., J. (2011, mayo 16). *Monitoreo de hábitos de manejo por mediode una red CAN automotriz*. Universidad de las Américas Puebla.Recuperado a partir de http://catarina.udlap.mx/u_dl_a/tales/documentos/lmt/pacheco_h_je/

Sparkfun. (2017). Arduino - SparkFunElectronics. Recuperado el 16 de enero de 2017, a partir de https://www.sparkfun.com/categories/103

DECLARACIÓN Y AUTORIZACIÓN

Yo, **CIFUENTES VERA, JEAN LUIS** con C.C: # 0925302994 autor del Trabajo de Titulación: **Diseño y simulación de prácticas virtuales usandoLabView, Proteus y Arduino mediante NI VISA** previo a la obtención deltítulo de **INGENIERO ELECTRÓNICO EN CONTROL Y AUTOMATISMO** en la Universidad Católica de Santiago de Guayaquil.

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

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

Guayaquil, 21 de Marzo de 2017

f. _____

Nombre: CIFUENTES VERA, JEAN LUIS

C.C: 0803321611

REPOSITORIO NACIONAL EN CIENCIA YTECNOLOGÍA

FICHA DE REGISTRO DE TESIS/TRABAJO DE TITULACIÓN

TÍTULO Y SUBTÍTULO:	DISEÑO Y SIMULACIÓN DE PRÁCTICAS VIRTUALES USANDOLABVIEW, PROTEUS Y ARDUINO MEDIANTE NI VISA.		
AUTOR(ES)	CIFUENTES VERA, JEAN LUIS		
REVISOR(ES)/TUTOR(ES)	ING. SUAREZ MURILLO, EFRAÍN OSWALDO		
INSTITUCIÓN:	Universidad Católica de Santiago de Guayaquil		
FACULTAD:	Facultad de Educación Técnica para el Desarrollo		
CARRERA:	Ingeniería Electrónica en controles y Automatismo		
TITULO OBTENIDO:	Ingeniero Electrónico en control y Automatismo		
FECHA DE PUBLICACIÓN:	**21 demarzo del 2017**	**No. DE PÁGINAS:**	59
ÁREAS TEMÁTICAS:	Microcontroladores, Instrumentación Virtual.		
PALABRAS CLAVES/ KEYWORDS:	LabView, Microcontroladores, Atmel, ATMega 328P, Proteus, Arduino		

RESUMEN/ABSTRACT (150-250 palabras):

El propósito principal del trabajo de titulación es la integración de varias herramientas virtuales, como, LabView, Isis Proteus y programación de alto nivel de código abierto. Antes del desarrollo de las aplicaciones prácticas del trabajo de titulación, se realizó una descripción general de los fundamentos básicos de microcontroladores y de la familia de microcontroladoresAtmel, así como también de la plataforma de desarrollo integrado Arduino IDE. No ha sido necesaria la descripción general de la herramienta virtual LabView debido a que los estudiantes de la Carrera de Ingeniería Electrónica en Control y Automatismo si lo han tratado. Mientras que los microcontroladoresAtmel ni Arduino fueron discutidos en clases. Posteriormente, se desarrollaron seis aplicaciones prácticas en la que se integran diseños de instrumentación virtual, simulación virtual y programación de código abierto Arduino. Finalmente, se evaluaron las aplicaciones prácticas desarrolladas los mismos que resultaron se satisfactorios durante la ejecución de cada una de las prácticas.

ADJUNTO PDF:	☒ SI	☐ NO
CONTACTO CON AUTOR/ES:	**Teléfono:**0991720202	E-mail:jinercive@hotmail.com
CONTACTO CON LA INSTITUCIÓN (C00RDINADOR DEL PROCESO UTE)::	**Nombre: Mendoza Merchan, Eduardo Vicente**	
	Teléfono: +593-9-68366762	
	E-mail:eduardo.mendoza@cu.ucsg.edu.ec	

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