ESCUELA SUPERIOR POLITÉCNICA DE CHIMBORAZO

FACULTAD DE MECÁNICA

CARRERA DE INGENIERÍA DE MANTENIMIENTO

"DESARROLLO DE UN SISTEMA DE CONTROL EN TIEMPO REAL PARA MEDIR LAS VARIABLES DE PRESIÓN Y VELOCIDAD CON SISTEMAS EMBEBIDOS"

SANDOVAL ATIAJA, ALEX PATRICIO

TRABAJO DE TITULACIÓN

TIPO: PROPUESTA TECNOLOGÍCA

Previa a la obtención del Titulo de:

INGENIERO DE MANTENIMIENTO

Riobamba–Ecuador

2018

ESPOCH

Facultad de Mecánica

CERTIFICADO DE APROBACIÓN DE TESIS

2017-12-20

Yo recomiendo que la Tesis preparada por:

ALEX PATRICIO SANDOVAL ATIAJA

Titulada: **"DESARROLLO DE UN SISTEMA DE CONTROL EN TIEMPO REAL PARA MEDIR LAS VARIABLES DE PRESIÓN Y VELOCIDAD CON SISTEMAS EMBEBIDOS."**

Sea aceptado como parcial complementación de los requerimientos para el Título de:

INGENIERO DE MANTENIMIENTO

Ing. Carlos Santillán Mariño

DECANO DE LA FAC. DE MECÁNICA

Nosotros coincidimos con esta recomendación:

Ing. Pablo Montalvo Jaramillo

DIRECTOR DE TESIS

Ing. Marco Santillán Gallegos

MIEMBRO DE TESIS

ESPOCH

Facultad de Mecánica

CERTIFICADO DE EXAMINACIÓN DE TESIS

NOMBRE DEL ESTUDIANTE: ALEX PATRICIO SANDOVAL ATIAJA

TÍTULO DE LA TESIS: "DESARROLLO DE UN SISTEMA DE CONTROL EN TIEMPO REAL PARA MEDIR LAS VARIABLES DE PRESIÓN Y VELOCIDAD CON SISTEMAS EMBEBIDOS."

Fecha de Examinación: 2018-05-21

RESULTADO DE LA EXAMINACIÓN:

COMITÉ DE EXAMINACIÓN	APRUEBA	NO APRUEBA	FIRMA
Ing. Ángel Guamán Mendoza **PRESIDENTE TRIB. DEFENSA**			
Ing. Pablo Montalvo Jaramillo **DIRECTOR DE TESIS**			
Ing. Marco Santillán Gallegos **MIEMBRO DE TESIS**			

* Más que un voto de no aprobación es razón suficiente para la falla total.

RECOMENDACIONES:

El Presidente del Tribunal certifica que las condiciones de la defensa se han cumplido.

Ing. Ángel Guamán Mendoza

PRESIDENTE DEL TRIBUNAL

DERECHOS DE AUTORÍA

DECLARACIÓN DE AUTENTICIDAD

Yo, Alex Patricio Sandoval Atiaja, declaro que el presente trabajo de titulación es de mi autoría y que los resultados del mismo son auténticos y originales. El texto constante en el documento que proviene de otra fuente está debidamente citadas y referenciados.

Como autor, asumo la responsabilidad legal y académica de los contenidos de este trabajo de titulación.

Alex Patricio Sandoval Atiaja

160057392-5

DEDICATORIA

Este trabajo dedico con mucho cariño a todas las personas que hicieron lo posible para cumplir este objetivo, en especial a mis padres Lidia Atiaja y Humberto Sandoval por ser el pilar fundamental en todo lo que soy, en toda mi educación, tanto académica, como de la vida, por su apoyo incondicional perfectamente mantenido a través del tiempo.

Mis hermanos, Diego y Edison Sandoval por apoyarme siempre y estar muy pendientes de mí, los quiero mucho.

Todo este trabajo ha sido posible gracias a ellos.

Alex Sandoval Atiaja

AGRADECIMIENTO

En primer lugar, agradezco a Dios por darme la vida y estar a mi lado, a mis padres, hermanos sobrinos y amigos por su gran apoyo y cariño incondicional.

Agradezco también a la Escuela Superior Politécnica De Chimborazo, en especial a la Escuela de Ingeniería de Mantenimiento, por abrirme las puertas y prepararme en esta hermosa profesión.

Alex Sandoval Atiaja

ÍNDICE

LISTA DE FIGURAS

Pág.

LISTA DE ANEXOS

LISTA DE ABREVIATURA

SE	Sistemas Embebidos
CPUs	Procesadores Digitales
MCU	Microcontroladores
DSP	Procesador Digital de Señales
LabVIEW	Laboratory Virtual Instrument Engineering Workbench.
DAC	Tarjetas de Adquisición de Datos
MXP	NI myRIO expansión
NI	National Instruments
MSP	NI miniSystems
FPGA	Field Programmable Gate Array

RESUMEN

Se realizó el diseño e implementación de un módulo de entrenamiento de un sistema para el control de las variables de presión y velocidad con sistemas embebidos. Para el desarrollo de este trabajo se investigó acerca de la comunicación a aplicar, topología, configuración y lenguaje de programación para lograr el envío y recepción de los datos que se obtienen de los sensores en este caso del transductor de presión y del encoder de cuadratura. El sistema de control se compone de tres aspectos sensores, hardware embebido y software embebido. Para estos dos últimos se utilizó una tarjeta Ny myRIO para la adquisición de datos y el software LabVIEW. Creada la comunicación entre la PC y cada uno de los dispositivos electrónicos que constituyen el módulo, se procede a programar un dispositivo a la vez en lenguaje FPGA, este lenguaje es exclusivo para los sistemas embebidos. El sistema está diseñado para el control, monitoreo y adquisición de datos en tiempo real. Este se implementó en el laboratorio de Control y Manipulación Automática de la Escuela de Ingeniería de Mantenimiento. Se realizaron pruebas de funcionamiento para constatar la adquisición de datos de cada una de las variables, el traductor de presión fue sometido a diferentes rangos de presión hasta verificar la activación de la alarma de seguridad configurada, se varia la frecuencia del tren de pulsos para obtener diferentes de velocidades, para conocer el estado de posición y el sentido de giro de motor. Previo a la manipulación del módulo se recomienda una correcta familiarización del uso de todos los elementos que se ha utilizado en el desarrollo de este trabajo.

PALABRAS CLAVES: <TECNOLOGÍA DE LOS PROCESOS INDUSTRIALES> <TECNOLOGÍA Y CIENCIAS DE LA INGENIERÍA> <SISTEMA EMBEBIDO>, <TARJETAS DE ADQUISICIÓN DE DATOS>, <LABVIEW>, <FIELD PROGRAMMABLE GATE ARRAY>, <CONTROL INDUSTRIAL>.

ABSTRACT

The design and implementation of a system training module for the control of pressure and velocity variables with embedded systems was carried out. For the development of this work the investigated of communication was applied, as well as topology, and configuration of programming language to achieve the sending and receiving data obtained from the sensors in this case of the pressure transducer and the quadrature encoder. The control system consists of three sensor aspects, embedded hardware and embedded software. For the latter two, a Ny myRIO card was used for data acquisition and LabVIEW software. Once the communication has been created, the PC and each one of the electronic devices make up the module come into play. One device is programmed at the same time in FPGA language; this language is exclusive for embedded systems. The system is designed for the control, monitoring and acquisition real time data. This was implemented in the Control and Automatic Manipulation Laboratory of the School of Maintenance Engineering. Performance tests were carried out to verify the acquisition of data for each of the variables, the pressure translator was subjected to different pressure ranges until the activation of the configured safety alarm was confirmed, the frequency of the pulse train was varied to obtain different speeds to know the state of position and the direction of rotation of the engine. Prior to the manipulation of the module, a correct familiarization of the use of all the elements that have been used in the development of this work is recommended.

KEYWORDS: <TECHNOLOGY OF INDUSTRIAL PROCESSES>, <TECHNOLOGY AND SCIENCE OF ENGINEERING>, <EMBEDED SYSTEM>, <DATA ACQUISITION CARDS>, <LABVIEW>, <FIELD PROGRAMMABLE GATE ARRAY>, <INDUSTRIAL CONTROL>

CAPÍTULO I

1. INTRODUCCIÓN.

1.1 Antecedentes.

En la actualidad la Carrera de Ingeniería de Mantenimiento de la Facultad de Mecánica de la ESPOCH ha tenido un crecimiento muy notable a la par con las nuevas técnicas y tecnologías de ingeniería.

La automatización es la mejor opción hoy en día para el control de procesos industriales y como es fuerte de esta escuela, día a día se desarrollan técnicas y métodos de ingeniería con la ayuda y manipulación de softwares además del uso de herramientas adicionales que permitan el desarrollo de cualquier sistema.

El Laboratorio de Mecatrónica que cuenta la Facultad de Mecánica se encuentra equipado con sistemas de control desarrollados por los mismos estudiantes durante el paso del tiempo.

Los sistemas de control han sido implementados usualmente con componentes análogos, pero a medida que estos sistemas han incrementado su la complejidad, se vuelve más difícil aplicar componentes análogos para su monitoreo. Por esta razón se ve la necesidad de implementar las nuevas tecnologías para la recolección de datos y así permitan verificar su correcto funcionamiento.

1.2 Justificación.

La tecnología en los últimos años ha tenido un crecimiento rápido, por este motivo, los sistemas de producción se han actualizado sus sistemas a la par con el desarrollo tecnológico implementando equipos y herramientas de control de última generación.

Los Sistemas de Control en tiempo real, representan la herramienta más usadas en la mayoría de las industrias desarrolladas, el objetivo de estas técnicas de control es llevar información del

funcionamiento y estado de los sistemas de producción y así de esta manera dar un diagnóstico de lo que está sucediendo en ese preciso instante.

Por tal razón se propone realizar un sistema de control en tiempo real con sistemas embebidos utilizando una tarjeta de adquisición de datos muy versátil y útil para el Laboratorio de Mecatrónica de la Facultad de Mecánica.

Esta implementación ayuda a conocer el funcionamiento y desarrollo de un sistema de control moderno y también será un gran aporte tecnológico, al ser una herramienta que permita realizar aplicaciones de supervisión y control.

El sistema de control y adquisición de datos que se realiza es moderno, didáctico, reprogramable y de fácil manipulación con el fin de practicar y expandir los conocimientos que el estudiante se proponga al sacar provecho a las ventajas de contar con este sistema.

1.3 Objetivos.

1.3.1 ***Objetivo general:*** Desarrollar un sistema de control en tiempo real para medir las variables de presión y velocidad con sistemas embebidos.

1.3.2 *Objetivos específicos:*

Determinar las características del hardware y software para la adquisición de datos.

Implementar el equipo y programar un sistema embebido.

Realizar mediciones experimentales de las variables mencionadas.

Verificar el funcionamiento del sistema desarrollado.

CAPÍTULO II

2. MARCO TEÓRICO.

2.1 Generalidades de los sistemas de control.

Un sistema de control influye sobre los elementos que tienen como función administrar, ordenar, dirigir o regular el comportamiento de otro sistema, con ello se obtiene el control del proceso.

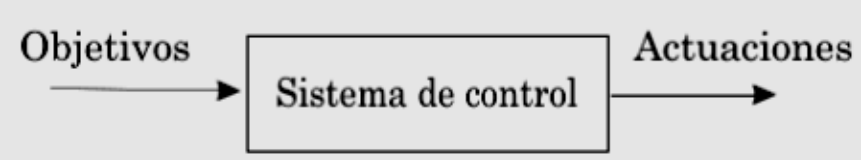

Figura 2. 1: Sistema de control.

Fuente: (GARCIA, 2004)

Hoy en día, casi todas las actividades en las que se desarrolla el ser humano aparecen involucradas de alguna u otra forma un sistema de control. (GARCÍA, 2004, p.4)

2.1.1 ***Elementos en un sistema de control:*** En todo sistema de control aparecen claramente diferenciados una serie de elementos característicos al mismo que es necesario clarificar: (GARCÍA, 2004, p.5)

- Variable a controlar. Generalmente se le conoce como señal de salida. Constituye la señal que deseamos que adquiera unos valores determinados.
- Planta o Sistema. La planta o sistema constituye el conjunto de elementos que realizan una determinada función.
- El sensor. Es el elemento que permite captar el valor de la variable a controlar en determinados instantes de tiempo.
- Señal de referencia. Es la señal consigna o valor que deseamos que adquiera la señal de salida (objetivo de control).

- Actuador. El actuador es el elemento que actúa sobre el sistema modificando de esta forma la señal de salida.
- Controlador. El controlador o regulador es el elemento que comanda al actuador en función del objetivo de control. (GARCÍA, 2004, p.5)

Todos estos elementos aparecen de alguna u otra forma en casi todo sistema de control. Identificar y estudiar cada uno de ellos de una forma correcta resulta esencial para poder diseñar un controlador que permita alcanzar el objetivo de control deseado en todo instante. (GARCÍA, 2004, p.5)

2.2 Sistemas Embebidos

Se entiende por sistemas embebidos a una combinación de hardware y software de computadora, sumado tal vez a algunas piezas mecánicas o de otro tipo, diseñado para tener una función específica. Es común el uso de estos dispositivos, pero pocos se dan cuenta que hay un procesador y un programa ejecutándose que les permite funcionar. (LLINARES, 2015, p.5)

Esta combinación de software y hardware puede ser reemplazada en muchos casos por un circuito integrado que realice la misma tarea. Pero una de las ventajas de los sistemas embebidos es su flexibilidad. Ya que a la hora de realizar alguna modificación resulta mucho más sencillo modificar una línea de código al software del sistema embebido que reemplazar todo el circuito integrado. (LLINARES, 2015, p.5)

Un uso muy común de los sistemas embebidos es en los sistemas de tiempo real, entendiéndose por sistemas en tiempo real a aquellos sistemas en los que el control del tiempo es vital para el correcto funcionamiento. (LLINARES, 2015, p.5)

2.2.1 ***Estructura de un sistema embebido.*** Las principales características de un sistema embebido son el bajo costo y consumo de potencia. Dado que muchos sistemas embebidos son concebidos para ser producidos en miles o millones de unidades, el costo por unidad es un aspecto importante a tener en cuenta en la etapa de diseño. Normalmente, los sistemas embebidos emplean procesadores muy básicos, relativamente lentos y memorias pequeñas para minimizar los costos. (MASTER, 2011, p.6)

2.2.2 ***Componentes de un sistema embebido.*** Un ES estaría formando por un microprocesador y un software que se ejecute sobre éste. Sin embargo, este software necesitará sin duda un lugar donde poder guardarse para luego ser ejecutado por el procesador. Esto podría tomar la forma de memoria RAM o ROM. Todo sistema embebido necesitará una cierta cantidad de memoria, la cual puede incluso encontrarse dentro del mismo chip del procesador. También contará con una serie de salidas y entradas necesarias para comunicarse con el mundo exterior. (MASTER, 2011, p.7)

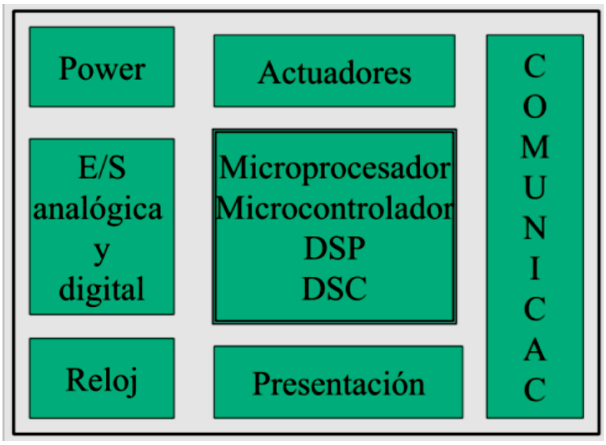

Figura 2. 2: Diagrama de bloques simplificado de un módulo típico de un SE.

Fuente: (MIÑARRO, 2009, p.7)

2.2.2.1 *Microprocesador*. Encargado de realizar las operaciones de cálculo principales del sistema. Ejecuta código para realizar una determinada tarea y dirige el funcionamiento de los demás elementos que le rodean. (MIÑARRO, 2009, p.16)

2.2.2.2 *Microcontrolador (MCU).* Es un dispositivo que alberga el sistema mínimo dentro de un único chip, esto es, incluye CPU, buses, reloj, memoria ROM, memoria RAM, E/S, otros periféricos tales como conversores A/D, temporizadores (timers), etc. (MIÑARRO, 2009, p.8)

2.2.2.3 *DSC.* Dispositivos mixtos microcontrolador/DSP. (MIÑARRO, 2009, p.8)

2.2.2.4 *Comunicaciones.* Los sistemas de comunicaciones adquieren, en el diseño de sistemas embebidos, cada vez mayor importancia. Lo normal es que el SE pueda comunicarse mediante interfaces estándar de comunicaciones por cable o inalámbricas. Así un SE normalmente incorpora puertos de comunicaciones bajo los estándares más extendidos, bien aquellos que necesitan de un cableado físico o se trate de comunicaciones inalámbricas. Podemos citar:

- RS-232
- RS485

- SPI
- CAN
- USB
- Ethernet
- Fibra óptica.
- Comunicaciones inalámbricas (WiFi, WiMax, Bluetooth, GSM, GPRS, UMTS, DSRC, RFID, etc.). (MIÑARRO, 2009, p.9)

2.2.2.5 *Presentación.* El subsistema presentación típico suele ser una pantalla gráfica, táctil, LCD alfanumérico, diodos LED, etc. Por lo general forma parte del interfaz hombre máquina del sistema. (MIÑARRO, 2009, p.9)

2.2.2.6 *Actuadores.* Denominamos actuadores a los posibles elementos encargados de llevar a cabo las acciones indicadas por la CPU. Entre éstos disponemos de Controladors de corriente, controladores de motores eléctricos, conmutadores, relés, etc. (MIÑARRO, 2009, p.10)

2.2.2.7 *Pines de E/S analógicos y digitales.* El módulo de Entrada/Salida (I/O) se encarga de hacer llegar o enviar las señales analógicas y digitales a los diferentes circuitos encargados de su generación y procesamiento. Tal es el caso de la conversión A/D para el procesamiento digital de señales analógicas procedentes de sensores, activación de actuadores mediante circuitos 'Controlador', reconocimiento del estado abierto cerrado de un conmutador o pulsador, encendido de diodos LED, etc. (MIÑARRO, 2009, p.10)

2.2.2.8 *Reloj*. El módulo de reloj es el encargado de generar las diferentes señales de reloj necesarias para la temporización de los circuitos digitales. (MIÑARRO, 2009, p.11)

2.2.2.9 *Módulo de alimentación (Power).* El módulo de energía (power) se encarga de generar las diferentes tensiones y corrientes necesarias para alimentar los componentes activos que forman el SE. Lo normal es el empleo de baterías para los dispositivos portátiles y fuentes de alimentación (conversor AC/DC) para los sistemas que disponen de acceso a la red de energía eléctrica. Los valores típicos más empleados para alimentar los sistemas embebidos son 5 Vdc, 9 Vdc, 12 Vdc y 24 Vdc. (MIÑARRO, 2009, p.12)

2.2.2.10 *Memoria RAM:* Almacena el código de los programas que el sistema puede ejecutar, así como los datos. Su característica principal es que debe tener un acceso de lectura y escritura lo más rápido posible para que el microprocesador no pierda tiempo en tareas que no son

meramente de cálculo. Al ser volátil el sistema requiere de un soporte donde se almacenen los datos incluso sin disponer de alimentación o energía. (MIÑARRO, 2009, p.16)

2.2.2.11 *Memoria Caché.* Más rápida que la principal en la que se almacenan los datos y el código accedido últimamente. Dado que el sistema realiza micro tareas, muchas veces repetitivas, la caché consigue ahorrar tiempo ya que no hará falta ir a memoria principal si el dato o la instrucción ya se encuentra en la caché. (MIÑARRO, 2009, p.16)

2.2.2.12 *Memoria No volátil.* Habitualmente conocida como 'Disco duro', en él la información no es volátil y además puede conseguir capacidades muy elevadas. A diferencia de la memoria RAM que es de estado sólido éste suele ser magnético en aplicaciones tipo PC, pero su excesivo tamaño y falta de robustez mecánica lo suele hacer inviable para PCs embebidos. (MIÑARRO, 2009, p.16)

Los avances tecnológicos, una vez más, han conseguido resolver el problema desarrollando discos de estado sólido. Existen en el mercado varias soluciones de esta clase con capacidades suficientes para la mayoría de los sistemas embebidos (hasta más de 20 GB). (MIÑARRO, 2009, p.16)

2.2.2.13 *BIOS-ROM.* Basic Input & Output System, sistema básico de entrada y salida) es código que es necesario para inicializar el ordenador y para poner en comunicación los distintos elementos de la placa madre. La ROM (Read Only Memory, memoria de sólo lectura no volátil) suele ser un chip donde se encuentra el código BIOS. (MIÑARRO, 2009, p.16)

2.2.2.14 *CMOS-RAM.* Es un chip de memoria de lectura y escritura alimentado con una pila donde se almacena el tipo y ubicación de los dispositivos conectados a la placa madre (disco duro, puertos de entrada y salida, etc.). Además, contiene un reloj en permanente funcionamiento que ofrece al sistema la fecha y la hora. (MIÑARRO, 2009, p.17)

2.2.2.15 *Chip Set:* Es un chip que se encarga de controlar las interrupciones dirigidas al Microprocesador, el acceso directo a memoria (DMA) y al bus ISA, además de ofrecer temporizadores, etc. Es frecuente encontrar la CMOS-RAM y el reloj de tiempo real en el interior del Chip Set. (MIÑARRO, 2009, p.16)

2.2.2.16 *Entradas al sistema.* Pueden existir puertos para ratón, teclado, vídeo en formato digital, comunicaciones serie o paralelo, etc. (MIÑARRO, 2009, p.17)

2.2.2.17 *Salidas del sistema:* Puertos de vídeo para monitor, pantallas de cristal líquido, altavoces, comunicaciones serie o paralelo, etc. (MIÑARRO, 2009, p.17)

2.2.2.18 *Ranuras de expansión.* Para tarjetas de tareas específicas que pueden no venir incorporadas en la placa madre. (MIÑARRO, 2009, p.16)

2.2.3 ***Tipos de interfaces de la memoria.*** A los Buses se les denomina también sistemas de líneas para la conexión interna y externa entre los dispositivos en un Sistema informático. Dependiendo de los dispositivos que se conectan, se pueden distinguir: un bus de sistema (bus principal), buses internos para la conexión con la memoria RAM principal, la conexión con la memoria Caché, buses de entrada/salida I/O, etc. (PEDRE, 2012, p.12)

2.2.3.1 *Principales tipos de buses*

- Bus de direcciones: está diseñado para enviar las direcciones, preparadas en el microprocesador, con el objetivo de elegir una celda definida de la memoria o un Puerto I/O (Entrada/salida). El bus de direcciones es de un solo sentido: las direcciones siempre son generadas por la MS
- Bus de datos: A lo largo del bus de datos de intercambio de información (instrucciones o datos) se lleva a cabo entre el microprocesador y los dispositivos periféricos – se trata de un intercambio de dos vías. Se trata de operaciones de lectura y escritura.
- Bus de control: es utilizado para el envío y la recepción de señales de control. Las señales de control aseguran la sincronización (control del tiempo) entre el MS y el resto de los componentes del Sistema. (PEDRE, 2012, p.13)

A continuación, se detalla un esquema de la conexión de los buses entre el MP, RAM y dispositivos de E/S (I/O). (PEDRE, 2012, p.13)

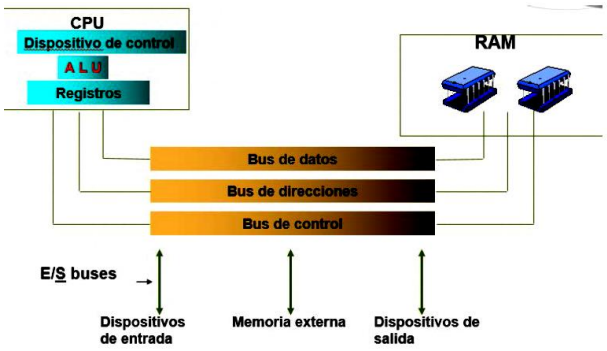

Figura 2. 3: Conexión de los buses entre el MP, RAM y dispositivos de E/S (I/O)

Fuente: (PEDRE, 2012, p.13)

2.2.4 ***Hardware embebido.*** Está compuesto por un microprocesador, microcontrolador, Procesador Digital de Señales (DSP), etc., en su parte central, es decir, es la CPU o unidad que aporta capacidad de cómputo al sistema, pudiendo incluir memoria interna, externa, o un micro con arquitectura específica según requisitos. (HERNÁNDEZ, y otros, 2010, p.37)

Dentro del hardware embebido se encuentran los actuadores y elementos electrónicos que el sistema se encarga de controlar. Puede ser un motor eléctrico, un conmutador tipo relé etc. (HERNÁNDEZ, y otros, 2010, p.37)

2.2.5 ***Software embebido.*** El software embebido conocido en inglés como firmware o embedded software, se utiliza para controlar los productos electrónicos y usualmente se ejecuta sobre un microprocesador interno, en un microcontrolador, en un DSP, en una FPGA, o en un PLC y a veces en una PC de propósitos generales adaptada para fines específicos. (HERNÁNDEZ, y otros, 2010, p.18-19)

Una solución de software para desarrollo embebido es *NI LabVIEW*, su desarrollo gráfico permite diseñar, generar prototipos e implementar aplicaciones embebidas de manera eficiente en un solo entorno. (HERNÁNDEZ, y otros, 2010, p18-19)

2.3 LabVIEW

LabVIEW (Laboratory Virtual InstrumentEngineeringWorkbench), es un entorno de programación gráfica con funciones integradas para realizar adquisición de datos, control de instrumentos, análisis de medida y presentaciones de datos, realiza cálculos complejos de señales analógicas y digitales capturadas a través de tarjetas de adquisición de datos (DAC), puertos serie y GPIB's. (HERNÁNDEZ, y otros, 2010, p.34)

2.3.1 ***Filosofía de LabVIEW.*** LabVIEW es un lenguaje completamente gráfico, y el resultado de ello es que es totalmente parecido a un instrumento, por ello a todos los módulos creados en LabVIEW se les llama Virtual Instrument (VI) o instrumento virtual. (HERNÁNDEZ, y otros, 2010, p.34)

Un *VI* es un módulo de *software* que simula el panel frontal apoyándose en elementos de *hardware* accesibles por el computador (DAC, instrumentos accesibles vía GPIB, VXI, RS-232, USB, Ethernet), que realiza una serie de medidas como si se tratase de un instrumento real. (HERNÁNDEZ, y otros, 2010, p.34)

Cuando se ejecuta un programa que funciona como *VI*, se puede observar en la pantalla del computador un panel cuya función es idéntica a la de un instrumentó físico, facilitando la visualización y el control del aparato. A partir de los datos reflejados en el panel frontal, el *VI* debe actuar recogiendo o generando señales, como lo haría su homólogo físico. (HERNÁNDEZ, y otros, 2010, p.34)

LabVIEW tiene la característica de descomposición modular ya que cualquier *VI* que se ha diseñado puede convertirse fácilmente en un módulo que puede ser usado como una sub-unidad dentro de otro *VI*. (LAJARA VIZCAÍNO, 2007)

Los programas en LabVIEW constan de dos partes principales:

2.3.1.1 *Panel Frontal (Front Panel).* El Panel Frontal es la cara que el usuario está viendo cuando se está monitorizando o controlando el sistema, o sea, el interfaz del usuario. Éste contiene una gran variedad de controles, indicadores e incluso se pueden diseñar controles e indicadores personalizados. (HERNÁNDEZ, y otros, 2010, p.35)

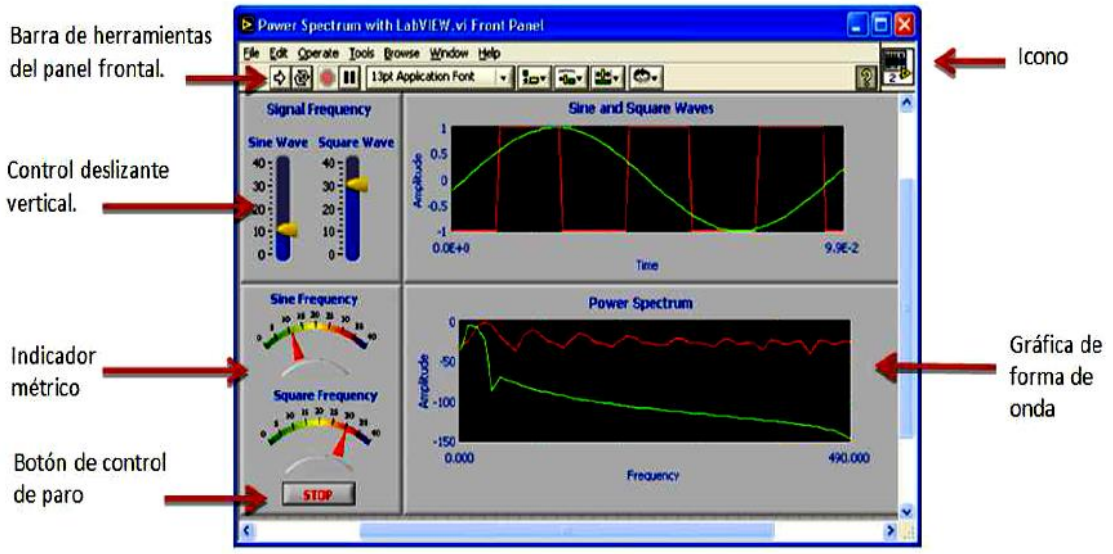

Figura 2. 4: Panel frontal

Fuente: (HERNÁNDEZ, y otros, 2010, p.35)

2.3.1.2 *Diagrama de Bloques (Block Diagram).* El diagrama de bloques del VI es la cara oculta del panel frontal, una cara que el usuario del sistema no puede ver, en ella están todos los controles e indicadores interconectados, pareciéndose mucho a un diagrama de esquema eléctrico. Esta cara es mucho menos conceptual que el panel frontal y para el usuario sería muy difícil entenderla. (HERNÁNDEZ, y otros, 2010, p.35-36)

Todos los módulos están interconectados, mediante líneas de conexión, por donde circulan los diferentes datos o valores del VI, de esta manera se logra que el VI funcione como un conjunto de elementos, módulos y submódulos. (HERNÁNDEZ, y otros, 2010, p.35-36)

2.3.2 *Características principales:*

- Facilidad de uso.
- Rapidez de programación.

Con LabVIEW pueden crearse programas de miles de VI´s (equivalente a millones de páginas de código texto) para aplicaciones complejas, programas de automatizaciones de decenas de miles de puntos de entradas/salidas(E/S), etc. (HERNÁNDEZ, y otros, 2010, p.36)

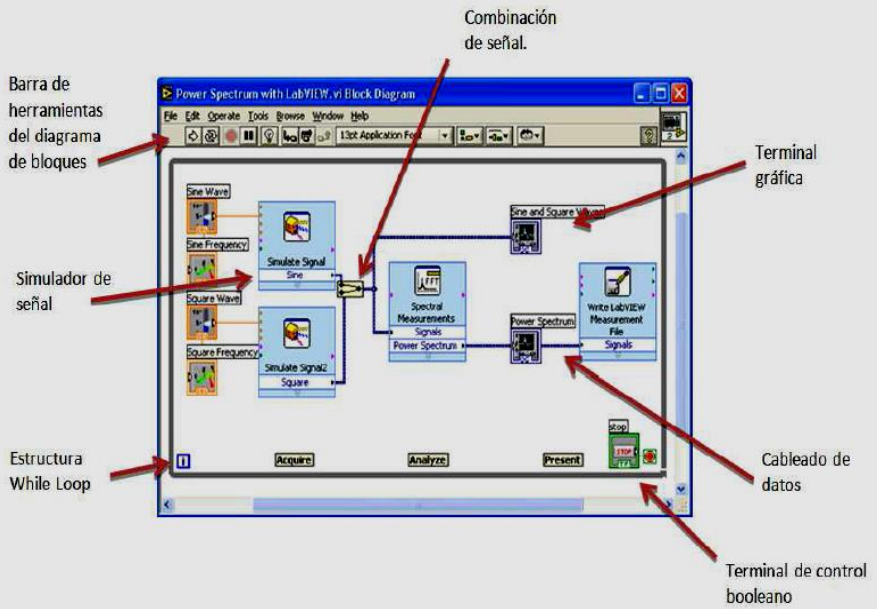

Figura 2. 5: Diagrama de bloques

Fuente: (HERNÁNDEZ, y otros, 2010, p.36)

2.3.3 ***Aplicaciones de LabVIEW.*** En tareas como:

- Adquisición de datos y proceso de señales.
- Control de instrumentos.
- Automatización industrial.
- Diseño de control.
- Diseño embebido. (HERNÁNDEZ, y otros, 2010, p.36)

2.4 Adquisición y análisis de datos

2.4.1 ***Adquisición de datos.*** La adquisición de datos implica la recopilación de señales eléctricas y físicas como voltaje, corriente, temperatura, presión o sonido de fuentes de medición y la digitalización de la señal para el almacenamiento, análisis y presentación en una PC de adquisición de datos. Se requiere de una etapa de acondicionamiento, que adecua la señal a niveles compatibles con el elemento que hace la transformación a señal digital. (HERNÁNDEZ, y otros, 2010, p.42)

La adquisición de datos basada en PC utiliza una combinación de hardware modular, software de aplicación y una PC para realizar medidas. Mientras cada sistema de adquisición de datos se define por sus requerimientos de aplicación, cada sistema comparte una meta en común de adquirir, analizar y presentar información. (HERNÁNDEZ, y otros, 2010, p.42)

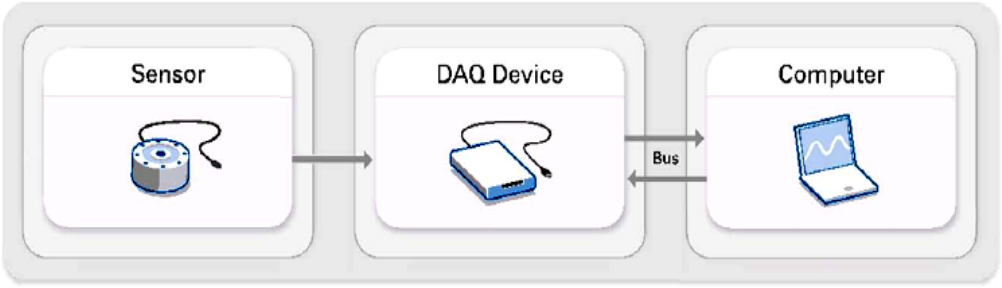

Figura 2. 6: Sistemas de adquisición de datos

Fuente: (NATIONAL INSTRUMENTS, 2011)

2.4.1.1 *Sensor.* Convierte un fenómeno físico en una señal eléctrica medible. (NATIONAL INSTRUMENTS, 2011)

2.4.1.2 *Acondicionamiento de Señales*.

- Mejorar una señal difícil de medir para el dispositivo DAQ.
- No siempre es necesario. (NATIONAL INSTRUMENTS, 2011)

2.4.1.3 *Dispositivo de adquisición de datos (Hardware).* Es la interfaz entre la señal y un PC, podría ser en forma de módulos que pueden ser conectados a la PC en los puertos (paralelo, serie, USB, etc.) o ranuras de las tarjetas conectadas a (PCI, ISA) en la placa madre. Las tarjetas DAQ a menudo contienen múltiples componentes (multiplexores, ADC, DAC, TTL-IO, temporizadores de alta velocidad, memoria RAM). Estos son accesibles a través de un bus por un microcontrolador, que puede ejecutar pequeños programas. (HERNÁNDEZ, y otros, 2010, p.45)

2.4.1.4 *Software de aplicación.* La elección del software asociado al sistema físico (tarjeta, bus de instrumentación, comunicación serie, etc.) se compone de tres niveles de decisión básicos, sistema operativo, software a nivel Controlador y software de aplicación. (HERNÁNDEZ, y otros, 2010, p.45)

NI ha desarrollado un software estructurado de tal forma, que permite la integración de una amplia variedad de instrumentos de medida y control electrónicos. Ofrece productos a varios niveles (software de aplicación, utilidades, Controladors de dispositivos, etc.), de modo que forman una arquitectura abierta, en donde se puede elegir el software que mejor se adapte a las necesidades de una determinada aplicación. (HERNÁNDEZ, y otros, 2010, p.46)

2.5 Control de Tiempo Real

"Tiempo Real" es uno de los términos más comúnmente usados en la industria, pero su definición es ambigua. La mayoría de los ingenieros están de acuerdo en que Tiempo Real significa "con retrasos aceptables". El término Tiempo Real duro comúnmente se utiliza para definir a un sistema que debe ejecutarse sin falla y cumplir con los requerimientos de tiempo real en todo momento. (NATIONAL INSTRUMENTS, 2007)

El error más común es pensar que Tiempo Real significa en realidad, rápido; cuando de hecho, muchas aplicaciones de adquisición de datos y control tienen ciclos muy lentos. Es el grado de inseguridad con cada tiempo de ciclo del lazo de control el que define los requerimientos de Tiempo Real de un sistema. (NATIONAL INSTRUMENTS, 2007)

2.5.1 ***Retos al crear un sistema de Tiempo Real.*** La tecnología Windows para desarrollar sistemas de Tiempo Real aún representa retos. En sistemas operativos de Tiempo Real, las interrupciones y eventos son jerarquizados y los eventos con la mayor prioridad se ejecutan antes que los eventos de prioridad menor. (NATIONAL INSTRUMENTS, 2007)

2.5.2 ***Desarrollo de sistemas de tiempo real con tecnologías estándar.*** National Instruments, el líder en productos para medición y automatización basadas en PC, ahora extiende su línea de hardware de adquisición de datos y software LabVIEW para simplificar el desarrollo de sistemas de Tiempo Real. Con LabVIEW RT y la Serie de hardware RT, usuarios pueden desarrollar aplicaciones de Tiempo Real usando LabVIEW para programar, computadoras con tecnología estándar como Microsoft Windows y hardware de alto rendimiento para adquisición de datos. (NATIONAL INSTRUMENTS, 2007)

2.6 MyRIO

2.6.1 ***¿Qué es NI myRIO?*** El diseño embebido del dispositivo NI myRIO fue creado para que los estudiantes realicen "ingeniería del mundo real" en un semestre. Contiene un procesador programable dual-core ARM Cortex-A9 de 667 MHz y un FPGA que los estudiantes pueden usar para empezar a desarrollar sistemas y resolver problemas de diseño complicado de manera más rápida — todo en una forma compacta. (NATIONAL INSTRUMENTS, 2016)

El dispositivo NI myRIO contiene el chip Zynq-7010, todo un sistema programable para liberar todo el poder de un sistema de LabVIEW ya sea en una aplicación de tiempo real, como en el nivel de un FPGA. En lugar de usar grandes cantidades de sintaxis, depuración de código de tiempo o el desarrollo de interfaces de usuario, los estudiantes pueden utilizar el paradigma de la programación gráfica de LabVIEW para centrarse en la construcción de sus sistemas y la solución de sus problemas de diseño y sin la presión añadida de una herramienta pesada. (NATIONAL INSTRUMENTS, 2016)

NI myRIO es una herramienta de enseñanza reconfigurable y reutilizable que ayuda a los estudiantes a aprender una gran variedad de conceptos de ingeniería, así como proyectos de diseño completos. (NATIONAL INSTRUMENTS, 2016)

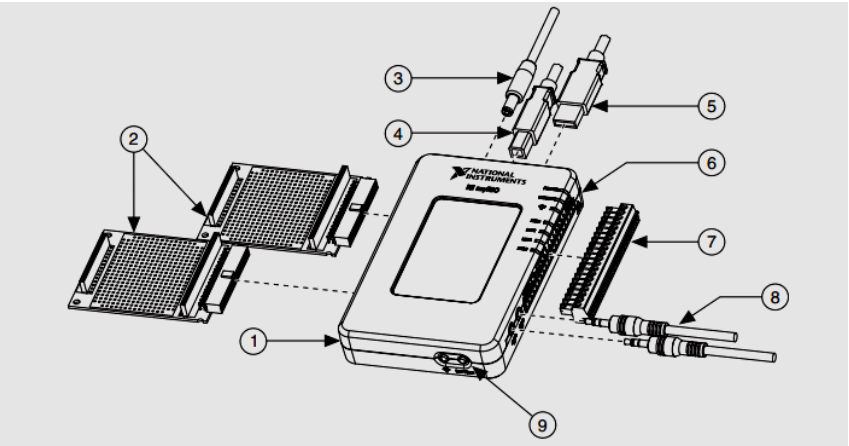

Figura 2. 7: myRIO 1900

Fuente: (NATIONAL INSTRUMENTS, 2017)

Utilizando herramientas de tiempo real, FPGA y capacidades integradas de Wi-Fi, junto con la memoria integrada; los estudiantes pueden desplegar aplicaciones de forma remota y ejecutarlos "headlessly" (sin conexión a un ordenador remoto). Tres conectores (dos puertos NI myRIO expansión [MXP] y un puerto de NI miniSystems [MSP] que es idéntico al conector NI myDAQ) envían y reciben señales desde los sensores y circuitos que los estudiantes necesitan en sus sistemas. Cuarenta líneas de E / S digitales, con el apoyo de SPI, PWM, entrada de codificador de cuadratura, UART e I2C; ocho entradas analógicas de una sola terminal; dos entradas analógicas diferenciales; cuatro salidas analógicas unipolares; y dos salidas analógicas con referencia a tierra permiten la conectividad a un sinnúmero de sensores y dispositivos y control de programación de sistemas. Toda esta funcionalidad está construida y preconfigurada en la funcionalidad FPGA por defecto. En última instancia, estas características ayudan a los estudiantes hacer la ingeniería del mundo real en este momento, desde los vehículos de radiocontrol hasta la creación de dispositivos médicos independientes. (NATIONAL INSTRUMENTS, 2016)

Una configuración para el FPGA se implementa en el dispositivo desde la fábrica, así que los principiantes pueden comenzar con una base funcional sin tener que programar una FPGA para realizar su trabajo. Sin embargo, el poder de E / S reconfigurable (RIO) se hace evidente cuando los estudiantes empezar a definir la personalidad FPGA y moldear el comportamiento del dispositivo a la aplicación. Con una escalabilidad del dispositivo, los estudiantes pueden usarlo desde cursos introductorios de sistemas embebidos hasta clases diseño de sistemas de control de último año. (NATIONAL INSTRUMENTS, 2016)

2.6.2 ***Requerimientos para realizar un sistema embebido con myRIO.***

2.6.2.1 *Software*

- NI LabVIEW 2015 o versiones superiores.
- NI LabVIEW myRIO Toolkit 2015
- NI LabVIEW Real-Time 2015
- Xilinx Compilation Tools 14.4. (NATIONAL INSTRUMENTS, 2016)

2.6.2.2 *Hardware*

- Tarjeta NI myRIO
- Fuente de alimentación para NI myRIO
- Cable USB para NI myRIO

- Kit de inicio de NI myRIO (PCB proporcionada por el instructor)
- Sistema CompactRIO
- Cable de Poder
- Cable de Ethernet. (NATIONAL INSTRUMENTS, 2016)

2.7 FIELD PROGRAMMABLE GATE ARRAY (FPGA)

2.7.1 ***¿Qué es un FPGA?*** Un FPGA (Field Programmable Gate Array) es un dispositivo semiconductor que contiene componentes lógicos programables e interconexiones programables entre ellos. Los componentes lógicos programables pueden ser programados para duplicar la funcionalidad de puertas lógicas básicas tales como AND, OR, XOR, NOT o funciones combinacionales más complejas tales como decodificadores o simples funciones matemáticas. En muchos FPGA, estos componentes lógicos programables (o bloques lógicos, según el lenguaje comúnmente usado) también incluyen elementos de memoria, los cuales pueden ser simples flip-flops o bloques de memoria más complejos. (MARÍN, 2016, p.21)

Una jerarquía de interconexiones programables permite a los bloques lógicos de un FPGA, ser interconectados según la necesidad del diseñador del sistema, algo parecido a un breadboard programable. Estos bloques lógicos e interconexiones pueden ser programados después del proceso de manufactura por el usuario/diseñador, así que el FPGA puede desempeñar cualquier función lógica necesaria. (MARÍN, 2016, p.22)

Las FPGA son generalmente más lentas que sus contrapartes, los circuitos integrados de aplicaciones específicas (ASIC por sus siglas en inglés), no pueden soportar diseños muy complejos, y consumen más energía. Sin embargo, ellas tienen muchas ventajas tales como la reducción del tiempo para la salida al mercado de productos, la habilidad para ser reprogramadas después de haber salido al mercado a fin de corregir posibles errores, y reduce los costos de ingeniería tales como investigación, diseño y prueba de un nuevo producto. (MARÍN, 2016, p.22)

2.7.2 *Los cinco beneficios principales de la tecnología FPGA*

2.7.2.1 *Rendimiento* – Aprovechando del paralelismo del hardware, los FPGAs exceden la potencia de cómputo de los procesadores digitales de señales (DSPs) rompiendo el paradigma de ejecución secuencial y logrando más en cada ciclo de reloj. (NATIONAL INSTRUMENTS, 2018)

2.7.2.2 *Tiempo en llegar al mercado* – La tecnología FPGA ofrece flexibilidad y capacidades de rápido desarrollo de prototipos para enfrentar los retos de que un producto se libere tarde al mercado. (NATIONAL INSTRUMENTS, 2018)

2.7.2.3 *Precio* – El precio de la ingeniería no recurrente de un diseño personalizado ASIC excede considerablemente al de las soluciones de hardware basadas en FPGA. (NATIONAL INSTRUMENTS, 2018)

2.7.2.4 *Fiabilidad* – Mientras que las herramientas de software ofrecen un entorno de programación, los circuitos de un FPGA son una implementación segura de la ejecución de un programa. (NATIONAL INSTRUMENTS, 2018)

2.7.2.5 *Mantenimiento a largo plazo* – Como se mencionó anteriormente, los chips FPGA son actualizables en campo y no requieren el tiempo y el precio que implica rediseñar un ASIC. Los protocolos de comunicación digital, por ejemplo, tienen especificaciones que podrían cambiar con el tiempo, y las interfaces basadas en ASICs podrían causar retos de mantenimiento y habilidad de actualización. Los chips FPGA, al ser reconfigurables, son capaces de mantenerse al tanto con modificaciones a futuro que pudieran ser necesarias. (NATIONAL INSTRUMENTS, 2018)

2.7.3 *Escoger un FPGA.* Al examinar las especificaciones de un chip FPGA, observe que generalmente están divididos en bloques de lógica configurables como segmentos o células de lógica, funciones fijas de lógica como multiplicadores, y recursos de memoria como RAM en bloque embebida. El chip FPGA tiene otros componentes, pero éstos son generalmente los más importantes cuando se seleccionan y comparan FPGAs para una aplicación en particular. (NATIONAL INSTRUMENTS, 2011)

2.7.4 *Enfoque de NI para diseño basado en FPGA.* En el pasado, la tecnología de FPGA estaba disponible solamente para ingenieros con un profundo conocimiento del diseño de hardware digital. El surgimiento de herramientas de diseño de alto nivel, como NI LabVIEW, cambia las reglas de programación de FPGAs, ofreciendo nuevas tecnologías que convierten los diagramas de bloques gráficos en circuitos de hardware digital. (NATIONAL INSTRUMENTS, 2011)

Todos los productos de hardware NI FPGA son construidos en base a una arquitectura de E/S reconfigurable (RIO), la cual tiene potentes procesadores de punto flotante, FPGAs reconfigurables y E/S modular. El hardware NI RIO, combinado con el software de diseño de sistemas LabVIEW, simplifica el desarrollo y acorta el tiempo al mercado al diseñar

aplicaciones avanzadas de control, monitoreo y pruebas. (NATIONAL INSTRUMENTS, 2011)

2.7.5 ***Definir las partes de un FPGA.*** Cada chip de FPGA está hecho de un número limitado de recursos predefinidos con interconexiones programables para implementar un circuito digital reconfigurable y bloques de E/S para permitir que los circuitos tengan acceso al mundo exterior. (NATIONAL INSTRUMENTS, 2011)

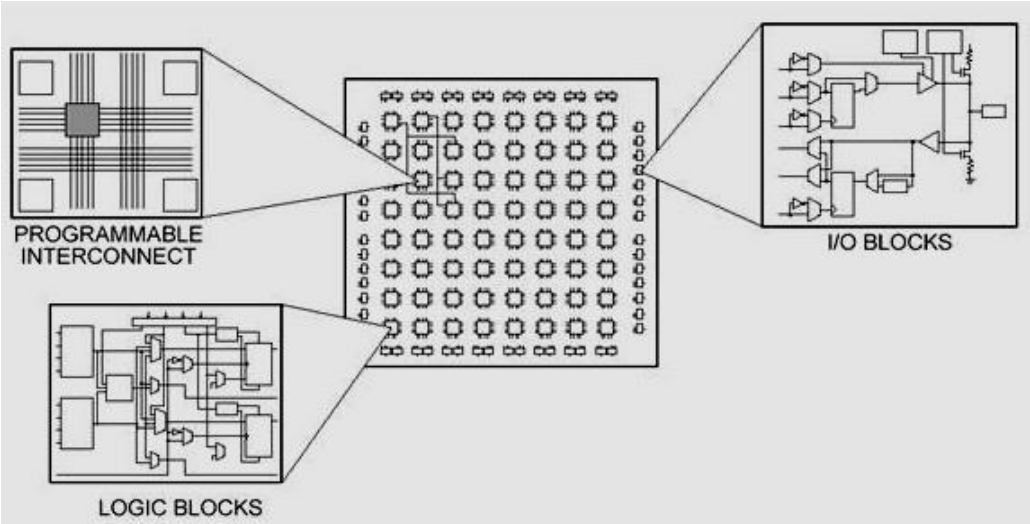

Figura 2. 8: Las Diferentes Partes de un FPGA

Fuente: (NATIONAL INSTRUMENTS, 2011)

2.7.6 ***Diseñar FPGAs en un sistema.*** Con este conocimiento de los componentes fundamentales del FPGA, usted puede ver claramente la ventaja que existe al implementar su lógica en circuitos de hardware: puede realizar mejoramientos en la velocidad de ejecución, fiabilidad y flexibilidad. Sin embargo, se enfrenta con cambios usando únicamente un FPGA para el procesamiento y la conectividad de E/S en su sistema. Los FPGAs no tienen el ecosistema controlador y base del código IP que tienen las arquitecturas de microprocesador y SOs. Además, los microprocesadores combinados con SOs ofrecen la base para las estructuras de archivos y la comunicación con periféricos usados por varias tareas, a menudo esenciales, como registrar datos en disco. (NATIONAL INSTRUMENTS, 2011)

2.7.7 ***Arquitectura de los FPGA.*** La arquitectura básica consiste en un arreglo de bloques lógicos programables (CLB) y canales de comunicación. Múltiples conectores de entrada/salida pueden caber en el tamaño largo de una fila o el ancho de una columna. Generalmente, todos los canales de comunicación tienen el mismo ancho (número de cables). (MARÍN, 2016, p.5)

Cualquier circuito de aplicación puede ser hecho dentro de la FPGA, siempre y cuando esta disponga de los recursos necesarios. Un bloque lógico típico de FPGA consiste en 4 entradas a una tabla de funciones lógicas (LookUp Table), y un flip-flop como se muestra en la siguiente gráfica. (MARÍN, 2016, p.5)

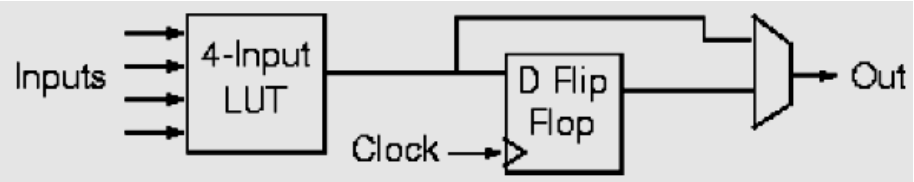

Figura 2. 9: Bloque lógico

Fuente: (MARÍN, 2016, p.5)

Hay solamente una salida, la cual puede ser ambas, la salida registrada o no registrada por el flip-flop, proveniente de la salida de la tabla de funciones lógicas. El bloque lógico tiene entonces 4 entradas para la tabla y una entrada de reloj para el flip-flop. Las señales de reloj y otras más, son manejadas por separado en FPGAs comerciales. (MARÍN, 2016, p.5)

2.7.8 ***Aplicaciones de los FPGA.*** Las aplicaciones de las FPGA incluyen a los DSP (Digital Signal Processor), radio definido por software, sistemas aeroespaciales y de defensa, prototipos de los ASIC, sistemas de imágenes para medicina, sistemas de visión para computadoras, reconocimiento de voz, bioinformática, emulación de hardware de computadora, y tienen un crecimiento de aplicaciones en otras áreas. Las FPGA encuentran aplicaciones en muchas áreas donde se requiera del paralelismo ofrecido por su arquitectura. (MARÍN, 2016, p.4)

CAPÍTULO III

3. CONSTRUCCIÓN Y PROGRAMACIÓN DEL SISTEMA EMBEBIDO PARA MEDIR LAS VARIABLES DE PRESIÓN Y VELOCIDAD.

3.1 Selección de materiales y herramientas.

3.1.1 ***MyRIO 1900.*** Esta es una tarjeta de adquisición de datos que se utiliza para el desarrollo de este trabajo.

Figura 3. 1: NI myRIO 1900.

Fuente: (Autor, 2018)

3.1.1.1 *Especificaciones Generales de la myRIO 1900.*

- Procesador + FPGA SoC (Sistema en un chip)
    - Xilinx Z-7010, 667 MHz, 2 núcleos
- Comunicaciones Wireless IEEE 802.11 b,g,n (WiFi)
- Comunicaciones USB 2.0 Alta-velocidad
- Entradas Analógicas: 10

- Tasa de muestreo agregado 500 kS/s
- Resolución12 bits
- Conectores MXP
    - 4 canales de un solo extremo por conector
    - Rango Nominal 0 V a +5 V
- Conector MSP
    - 2 canales diferenciales
    - Rango Nominal ±10 V
- Audio input
    - 1 entrada estéreo (2 Canales AC acoplado)
    - Rango Nominal ±2.5 V

- Salidas Analógicas: 6
    - AO en MXP: 345 kS/s
    - AO en MSP y audio: 345 kS/s
    - Resolución 12 bits
    - Conectores MXP
        - Configuración 2 canales de terminación única por conector
        - Rango 0 V a +5 V
    - Conector MSP
        - Configuración 2 canales de terminación única
        - Rango ±10 V
    - Salida de audio
        - Configuración una salida estéreo que consta de dos canales acoplados a CA, de extremo único

- Entradas/Salidas Digitales: 40
    - Número de líneas
        - Conectores MXP
        - 2 puertos de 16 líneas DIO (un puerto por conector)
        - una UART.RX y una línea UART.TX por conector
        - Conector MSP
        - 1 puerto de 8 líneas DIO
    - Control de dirección: cada línea DIO individualmente programable como entrada o salida
    - Nivel lógico: entrada LVTTL compatible con 5 V; Salida 3,3 V LVTTL
    - Ancho mínimo de pulso: 20 ns

- Máximas frecuencias para funciones digitales secundarias
    - SPI: 4 MHz
    - PWM: 100 kHz
    - Entrada de codificador en cuadratura: 100 kHz
    - I2C: 400 kHz

- Acelerómetro
    - Número de ejes: 3
    - Rango: ±8 g
    - Resolución: 12 bits
    - Tasa de muestreo: 800 S/s
    - Ruido: 3.9 mgrms típico a 25 ° C

- Salida de potencia
    - +5 V Salida de potencia
        - Tensión de salida 4.75 V a 5.25 V
        - Corriente máxima en cada conector 100 mA
    - +3.3 V Salida de potencia
        - Tensión de salida 3.0 V a 3.6 V
        - Corriente máxima en cada conector 150 mA
    - +15 Salida de potencia
        - Tensión de salida 15 V a +16 V
        - Corriente máxima 32 mA (16 mA durante el inicio)
    - -15 V Salida de potencia
        - Tensión de salida 15 V a -16 V
        - Corriente máxima 32 mA (16 mA durante el inicio)
        - Potencia combinada máxima desde +15 V y potencia de salida de -15 V 500 mW. (NATIONAL INSTRUMENTS, 2017)

3.1.2 *Motor de pasos NEMA 17.*

3.1.2.1 *Especificaciones Generales del motor Nema 17.*

- Fases: 2
- Pasos / Revolución: 80
- Exactitud del paso: ± 5%
- Carga del eje: 20,000 horas a 1000 RPM
- Empuje Axial: 25 N (5.6 lb) Empuje

	65 N (15 lb.) Tire
- Radial:	29 N (6.5 lb.) En el centro plano
- Grado del IP:	40
- Aprobaciones:	RoHS
- Temperatura de funcionamiento:	-20 ° C a + 40 ° C
- Clase de aislamiento:	B 130 ° C
- Resistencia de aislamiento:	100 MΩ. (DATASHEET)

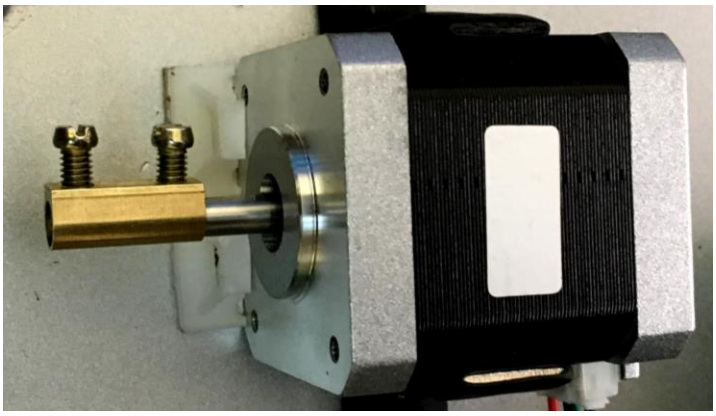

Figura 3. 2: Motor Nema 17

Fuente: (Autor, 2018)

3.1.3 ***Controlador para motor de pasos A4988.*** Opera desde 8 V a 35 V y puede entregar hasta aproximadamente 1 A por fase sin un disipador de calor (puede manejar hasta 2 A con un buen disipador). Cuenta con potenciómetro que permite ajustar la corriente máxima de salida y utilizar tensiones superiores a la tensión nominal para lograr mayores tasas de paso, además lleva un control inteligente que selecciona automáticamente el modo de reducción de la corriente correcta. (ENEKA, 2014)

3.1.3.1 *Características del controlador A4988*

- Interfaz de control de paso y dirección simple.
- Cinco diferentes resoluciones paso: paso completo, de medio paso, un cuarto de paso, el octavo paso, y decimosextos pasos.
- Control de corriente ajustable le permite ajustar la salida de corriente máxima con un potenciómetro, que le permite utilizar tensiones superiores a la tensión nominal del motor paso a paso para lograr mayores tasas de paso.
- Control Inteligente de picar que selecciona automáticamente el modo de decaimiento de corriente correcta (disminución rápida o lenta decadencia).

- El exceso de temperatura de apagado térmico, bloqueo de insuficiencia de voltaje y protección de cruce de corriente.
- Short -tierra y la protección de cortocircuito de carga. (ENEKA, 2014)

Figura 3. 3: Controlador para motor de pasos A4988

Fuente: (Autor, 2018)

3.1.3.2 *Especificaciones generales del controlador para motor de pasos A4988*

- Tensión mínima: 8 V
- Tensión máxima: 35 V
- Corriente continua por fase: 1 A2
- Máxima de fase por la corriente: 2 A3
- Tensión mínima lógica: 3 V
- Máxima tensión lógica: 5,5 V
- Resoluciones Micro paso: máximo, 1/2, 1/4, 1/8, y 1/16 (ENEKA, 2014)

3.1.4 *Fuente de alimentación 24 VDC 2.5A*

3.1.4.1 *Especificaciones generales de la fuente de alimentación 24 VDC 2.5A*

- Potencia: 60W
- Voltaje de salida: 12-24VDC-26.4VDC
- Corriente de salida: 2.5A
- Voltaje de la fuente de alimentación: 85.264VAC -124.370VAC
- Montaje: DIN
- Cantidad de salidas: 1
- Temperatura de funcionamiento: -20-60 ° C

- Protección: sobreintensidad, sobretensión, cortocircuito
- Eficiencia: 84%
- Dimensiones: 93x78x56mm
- Peso: 300g

Figura 3. 4: Fuente de alimentación 24 VDC 2.5A

Fuente: (Autor, 2018)

3.1.5 ***Encoder de cuadratura.*** Los encoder de cuadratura entregan dos señales tipo tren de pulsos A y B por cada uno de sus dos canales desfasadas 90º. Cuando el encoder gira hacia una dirección el canal A adelanta al B, y cuando gira en dirección contraria el canal B adelanta al A. (NATIONAL INSTRUMENTS, 1879)

Cuando se dispone de una tarjeta DAQ con dos contadores es sencillo medir velocidad y dirección, y así determinar posición. Sin embargo, a menudo no se dispone de tarjetas o dispositivos con dos contadores, o que no permiten ingresar directamente la señal del encoder. (NATIONAL INSTRUMENTS, 1879)

La solución es convertir la señal de cuadratura de 90º en dos señales: una de pulsos para determinar velocidad y cantidad de movimiento, y otro tipo digital On/Off para determinar la dirección. (NATIONAL INSTRUMENTS, 1879)

La desventaja de esta conversión es que en la transición pueden perderse pulsos y así contar erróneamente la cantidad de movimiento que ha habido en una dirección determinada. (NATIONAL INSTRUMENTS, 1879)

Figura 3. 5: Encoder de cuadratura.

Fuente: (Autor, 2018)

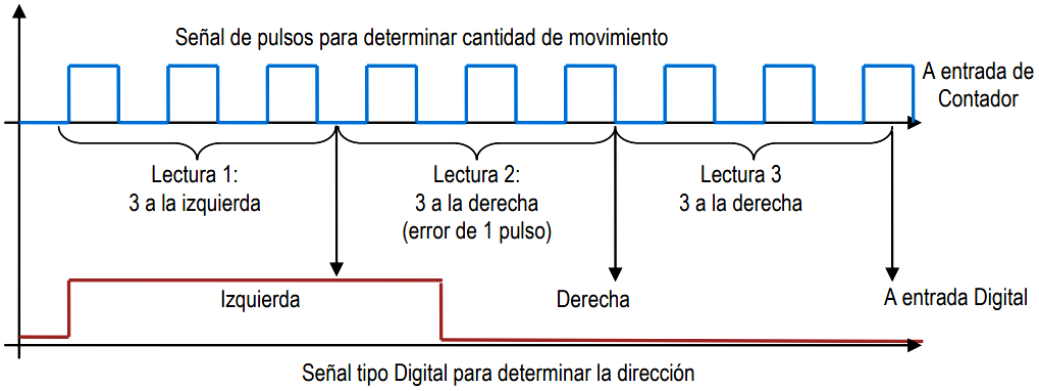

Figura 3. 6: Señales para medición de posición y velocidad.

Fuente: (NATIONAL INSTRUMENTS, 1879)

3.1.5.1 *Descripción de funcionamiento.* Se utiliza el canal A como entrada de reloj, y el canal B como entrada de Datos.

Cuando el canal A adelanta al B, la transición del reloj siempre sucede cuando D está en bajo, por lo tanto, la señal Q es baja.

Cuando el canal B adelanta al A, la transición del reloj siempre sucede cuando D está en alto, por lo tanto, la señal Q es alta. (NATIONAL INSTRUMENTS, 1879)

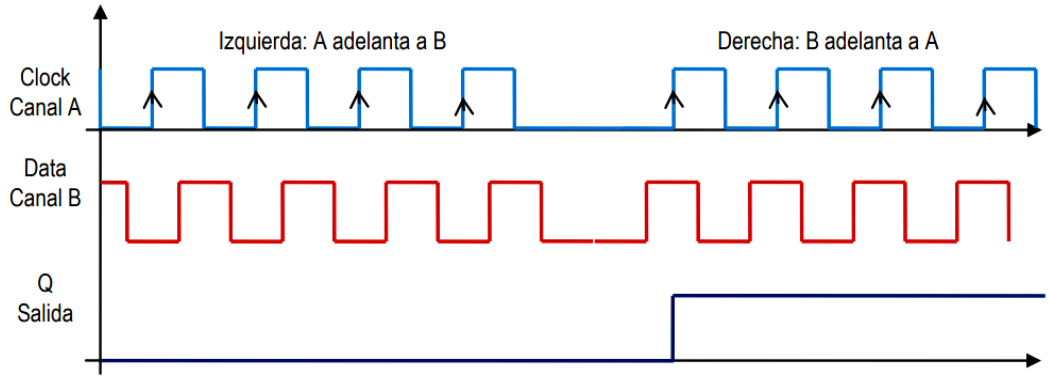

Figura 3. 7: Diagrama de tiempo de conversión de la señal.

Fuente: (NATIONAL INSTRUMENTS, 1879)

3.1.6 ***Transductor de presión.*** Sensor utilizado para medir la variable de presión, trasforma la señal de la presión del aire en una señal de corriente para ser procesada por el controlador.

Figura 3. 8: Transductor de presión.

Fuente: (Autor, 2018)

3.1.6.1 *Descripción del Transductor de presión.*

- Cuerpo en acero inoxidable.
- Alimentación: 11…30 VDC
- Salida: 4…20 mA
- Rango de medida: 0…10 BAR (0…145PSI)
- Conexión a proceso: 1/4 NPT MACHO
- Conexión eléctrica: CONECTOR DIN
- Temperatura de operación: -25-100 °C
- Marca: BAUMER (FRANCIA)

3.1.7 ***Software LabVIEW myRIO.*** Con el software NI LabVIEW, se aprovechar por completo el procesador y FPGA en NI myRIO. El software mínimo requerido para programar el procesador NI myRIO es LabVIEW.

Instale el software requerido, para una experiencia diseñada para asegurar su éxito con la herramienta NI myRIO y tener acceso a los recursos de la herramienta myRIO directamente desde la ventana LabVIEW Getting Started. (NATIONAL INSTRUMENTS, 2013)

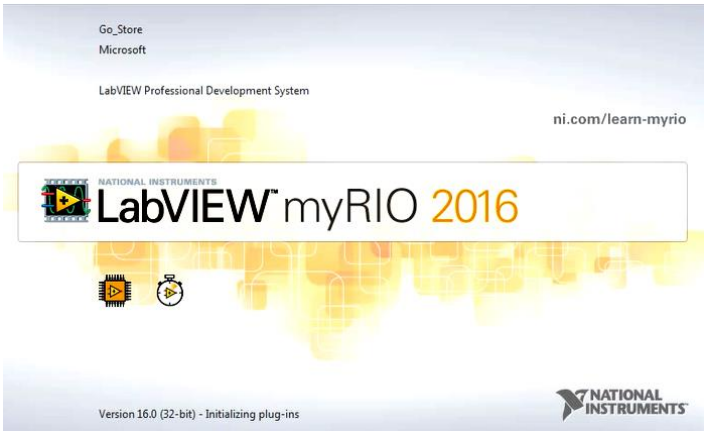

Figura 3. 9: Software LabVIEW myRIO.

Fuente: (Autor, 2018)

3.2 Pasos para el ensamblaje y conexión del sistema control.

Paso 1.- Construcción de la base con perfiles de aluminio y una plancha de alucobond para el respectivo montaje de los elementos.

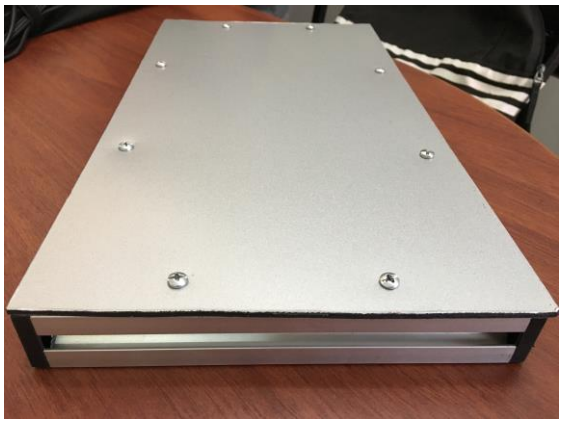

Figura 3. 10: Base.

Fuente: (Autor, 2018)

Paso 2.- Esquema de conexión de los elementos del sistema de control. En el cual consta el controlador A4988, el motor de pasos, la fuente de poder de 24 V, el encoder de cuadratura, el transductor de presión y los buses que son las entradas y salidas digitales y analógicas para la conexión de estos elementos con la tarjeta de adquisición de datos NI myRIO.

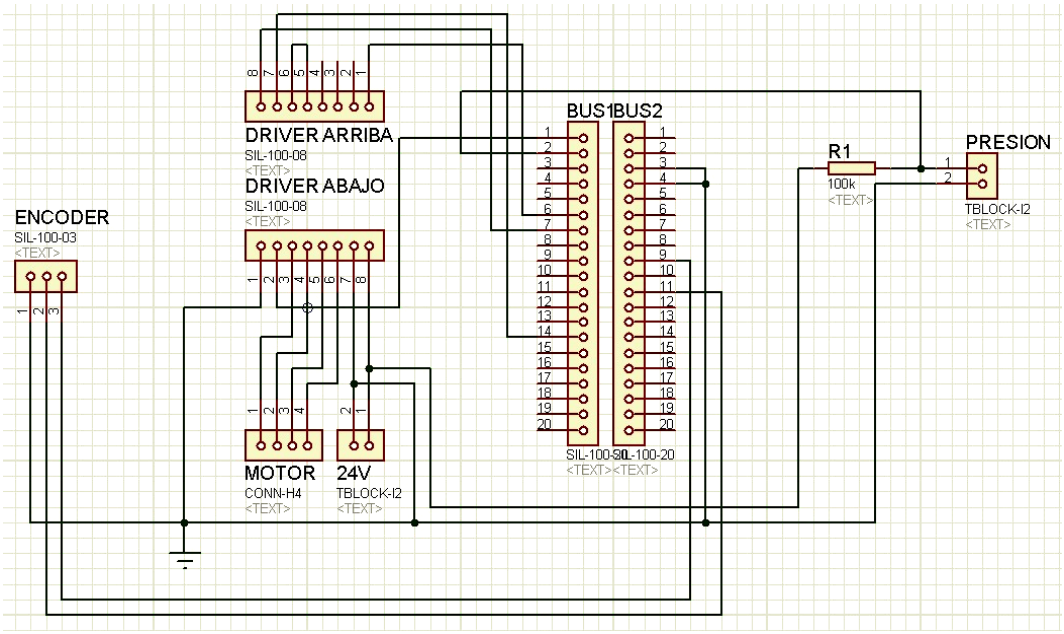

Figura 3. 11: Esquema de conexión de los elementos del sistema de control.

Fuente: (Autor, 2018)

Paso 3.- Construcción de una placa electrónica, necesaria para el funcionamiento y alimentación del motor en la cual estará montado el controlador A4988, dicha placa también sirve para la interconexión entre los sensores y la tarjeta NI myRIO.

Paso 3.1.- Diseño de las pistas de la placa electrónica según el esquema de conexión de elementos del sistema de control. (Ver Figura 3-12.)

Paso 3.2.- Impresión de las pistas de la placa electrónica. (Ver Figura 3-13.)

Paso 3.3.- Impresión de las pistas electrónicas en la baquelita. (Ver Figura 3-14.)

Paso 3.4.- Modelación en 3D de la placa electrónica previo a su construcción. (Ver Figura 3-15.)

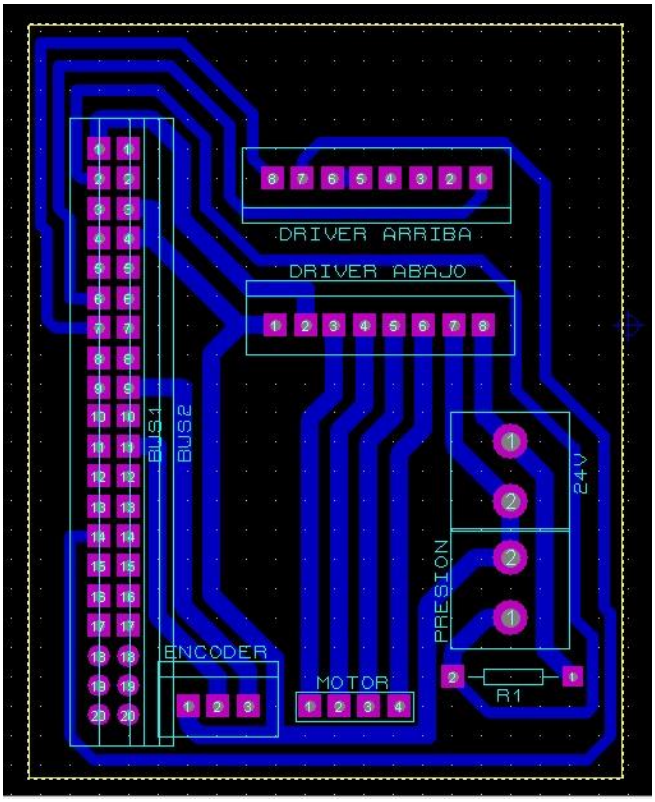

Figura 3. 12: Diseño de las pistas de la placa electrónica.

Fuente: (Autor, 2018)

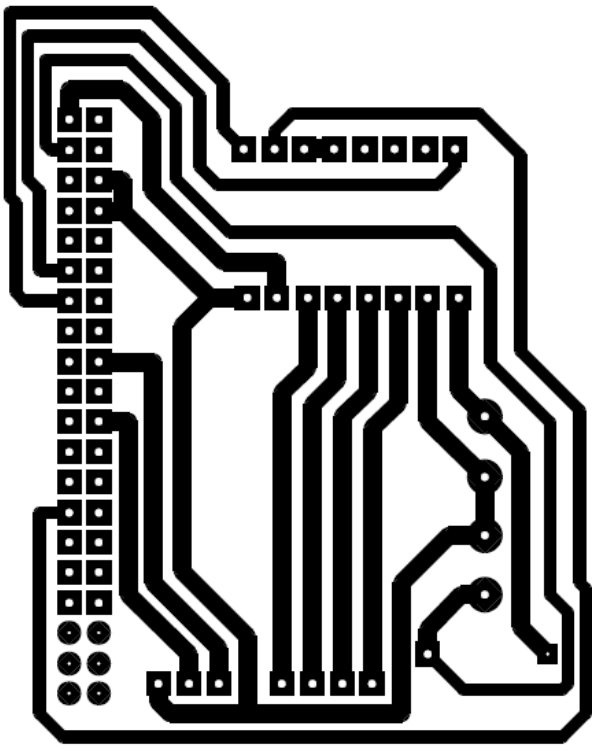

Figura 3. 13: Impresión de las pistas de la placa electrónica.

Fuente: (Autor, 2018)

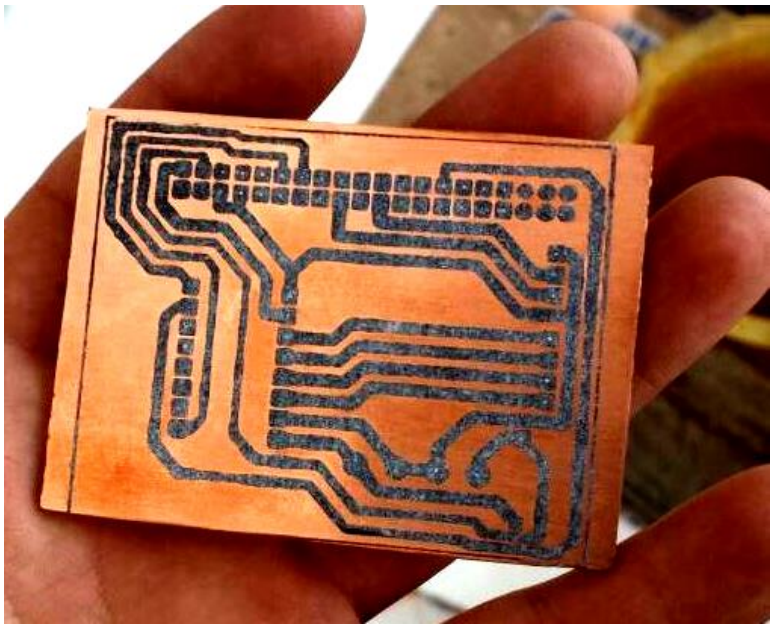

Figura 3. 14: Impresión de las pistas electrónicas en la baquelita.

Fuente: (Autor, 2018)

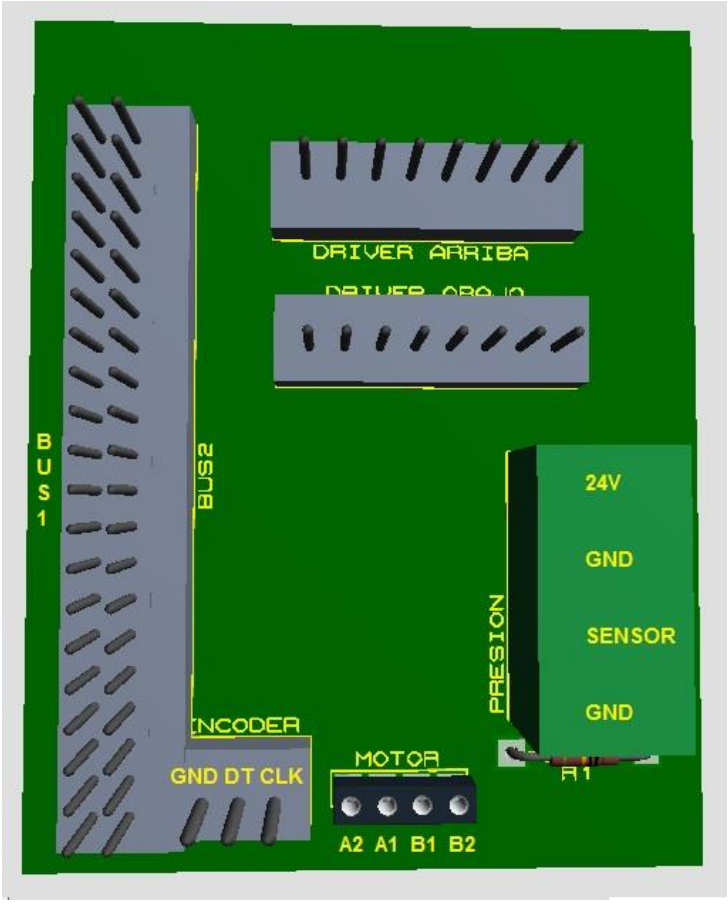

Figura 3. 15: Modelación en 3D de la placa electrónica.

Fuente: (Autor, 2018)

Paso 3.5.- Perforación y soldadura de las pistas electrónicas en la baquelita. (Ver Figura 3-16).

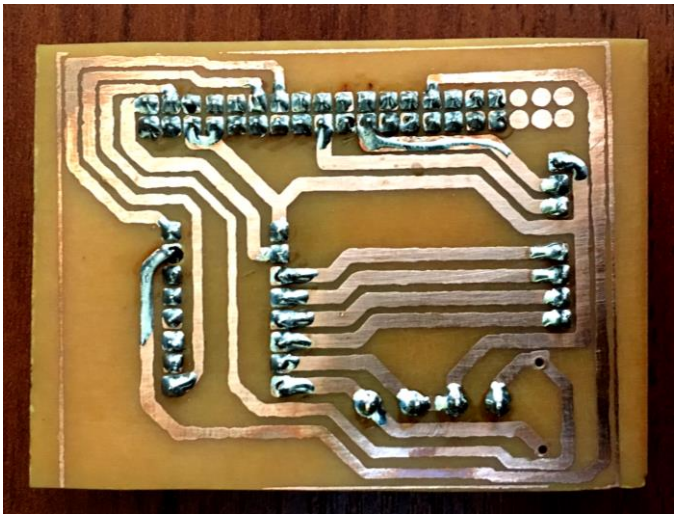

Figura 3. 16: Perforación y soldadura de las pistas electrónicas.

Fuente: (Autor, 2018)

Paso 3.5.- Terminado de la placa electrónicas. (Ver Figura 3-17).

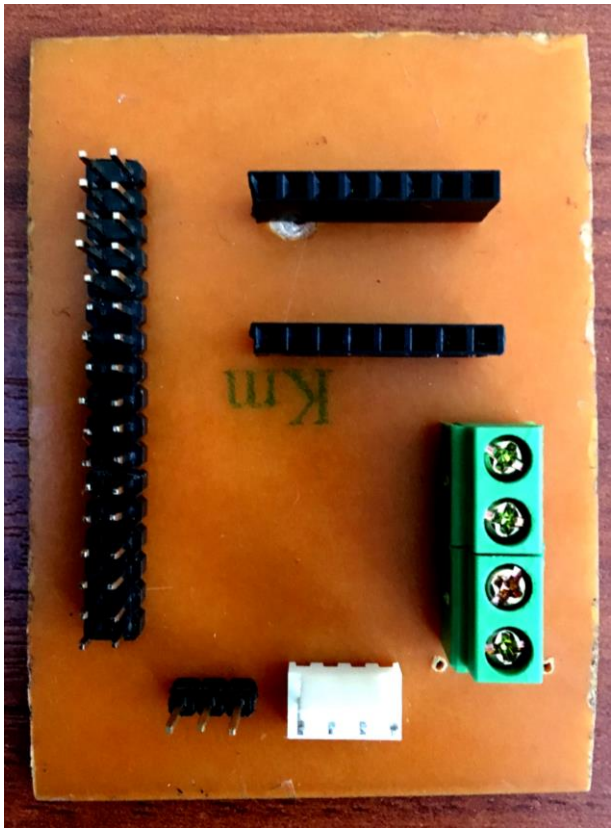

Figura 3. 17: Terminado de la placa electrónicas.

Fuente: (Autor, 2018)

Paso 4.- Montaje de los elementos del sistema de control.

Paso 4.1.- Montaje del motor de pasos

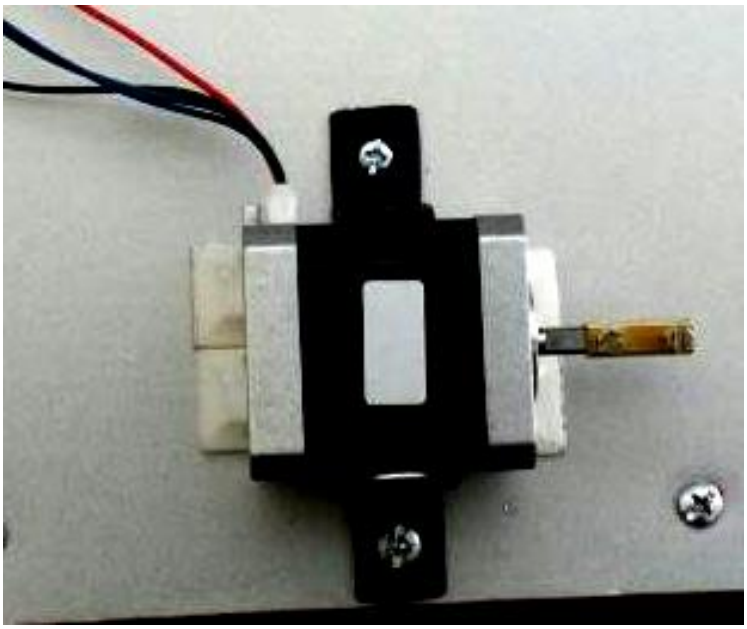

Figura 3. 18: Montaje del motor de pasos

Fuente: (Autor, 2018)

Paso 4.2.- Montaje de la fuente de alimentación de 24 VDC 2.5A.

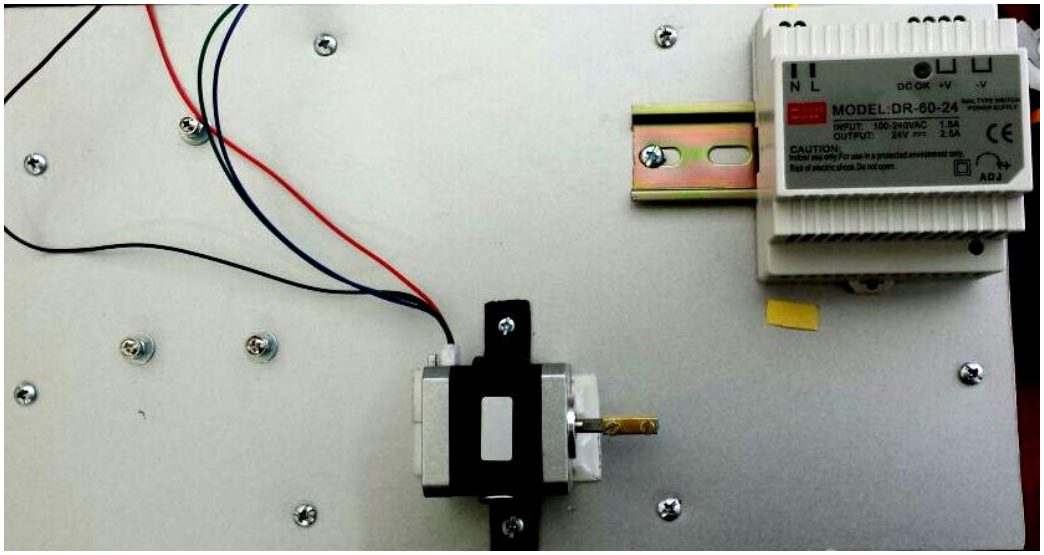

Figura 3. 19: Montaje de la fuente de alimentación de 24 VDC 2.5 A.

Fuente: (Autor, 2018)

Paso 4.3.- Montaje de la NI myRIO.

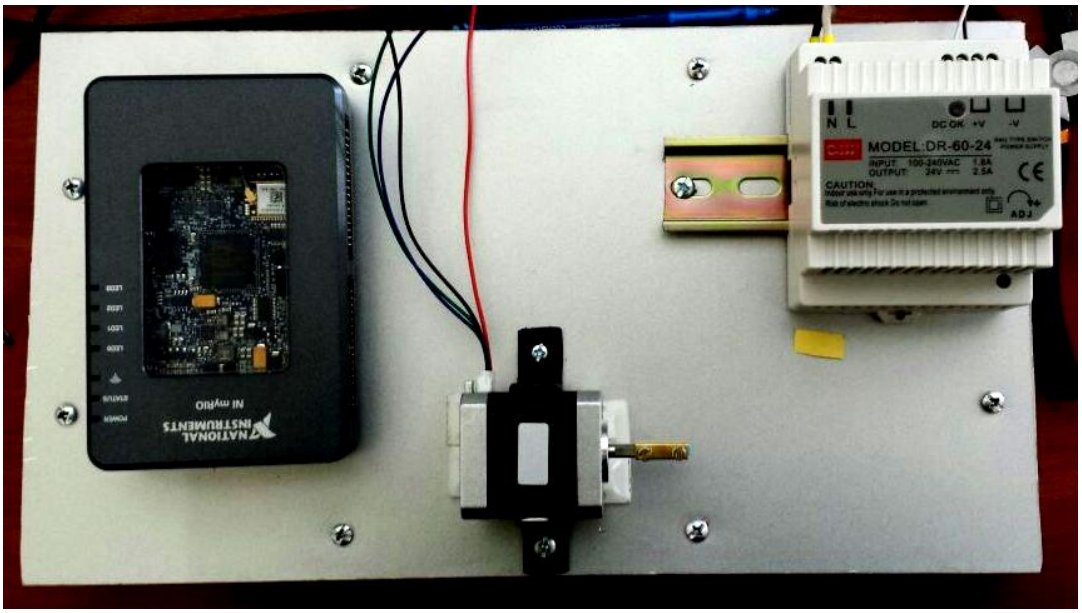

Figura 3. 20: Montaje de la NI myRIO.

Fuente: (Autor, 2018)

Paso 4.4.- Montaje del encoder de cuadratura.

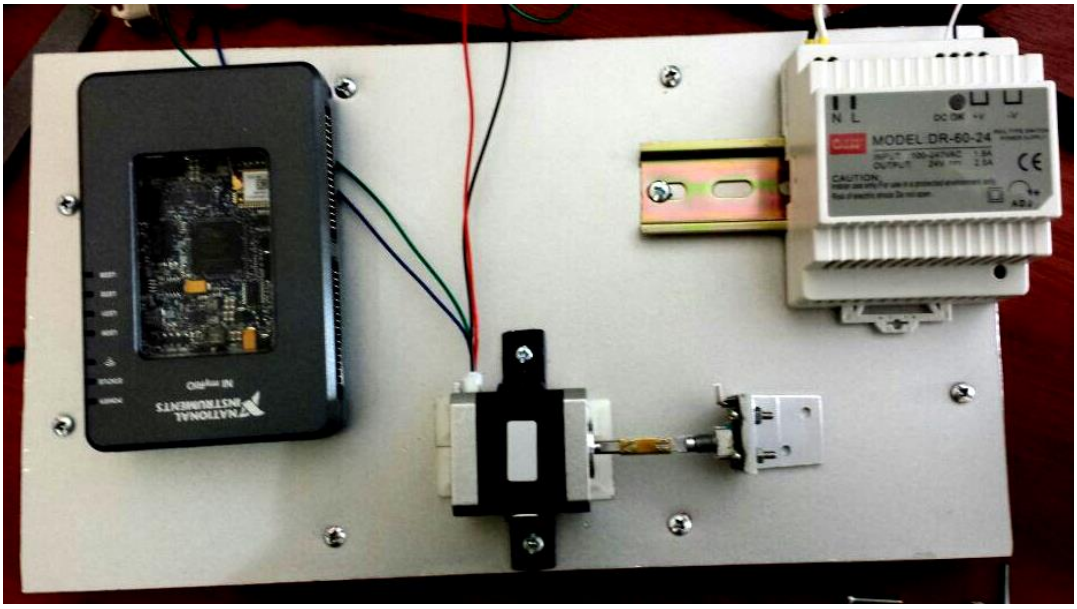

Figura 3. 21: Montaje del encoder de cuadratura.

Fuente: (Autor, 2018)

Paso 4.5.- Montaje del Controlador A4988 en la placa electrónica.

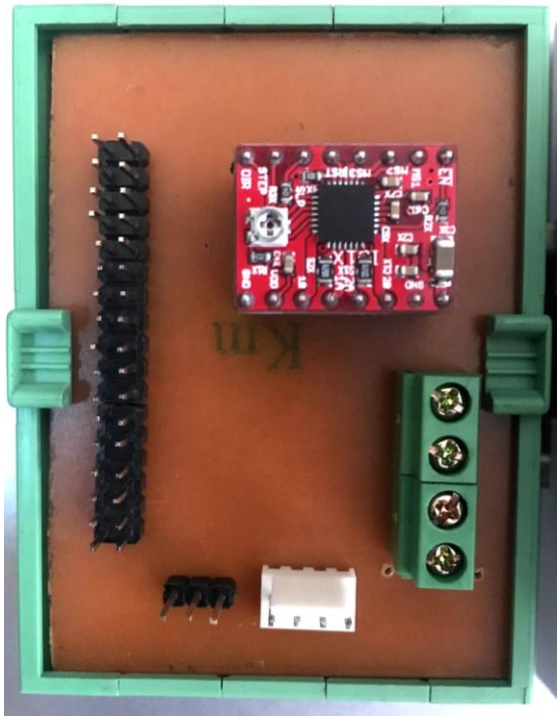

Figura 3. 22: Montaje del Controlador A4988 en la placa electrónica.

Fuente: (Autor, 2018)

Paso 4.6.- Montaje de la placa electrónica sobre el módulo.

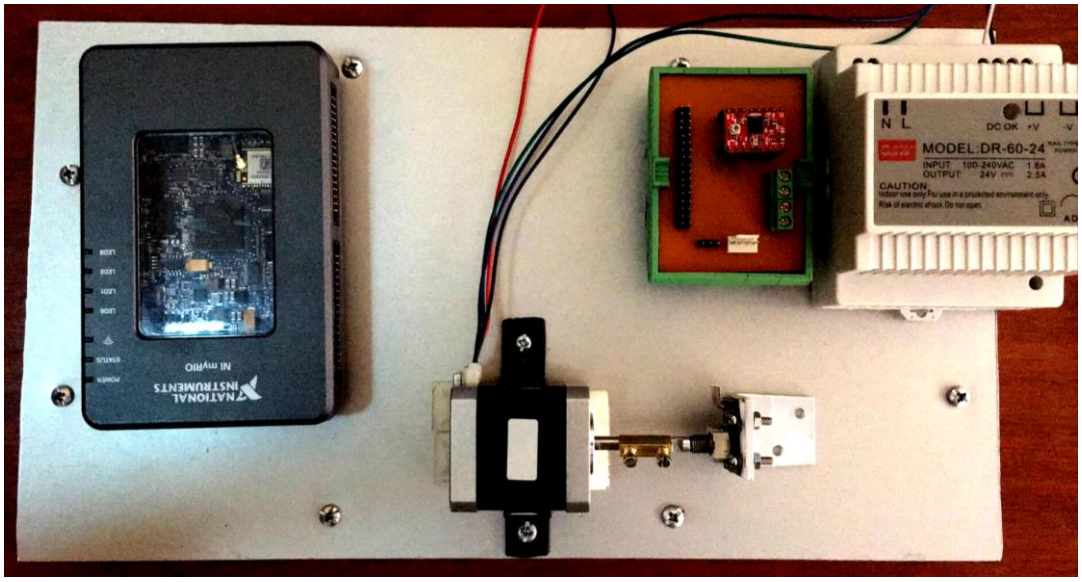

Figura 3. 23: Montaje de la placa electrónica sobre el módulo.

Fuente: (Autor, 2018)

Paso 5.- Conexión y descripción de las conexiones de los elementos del sistema de control.

Se eligió la fuente de 24 V de 2.5 A exclusivamente porque el transductor de presión trabaja a 24 V y este voltaje está en el rango permitido para la alimentación del motor. Esta fuente está conectada a los pines de alimentación del Controlador del motor de pasos y también está conectada para la alimentación del transductor de presión.

Además de esto lo que se está haciendo es medir el valor del transductor de presión a esta acción se le considera una entrada analógica así que ira conectada a la entrada A0 pin número 3 del bus B de la tarjeta NI myRIO.

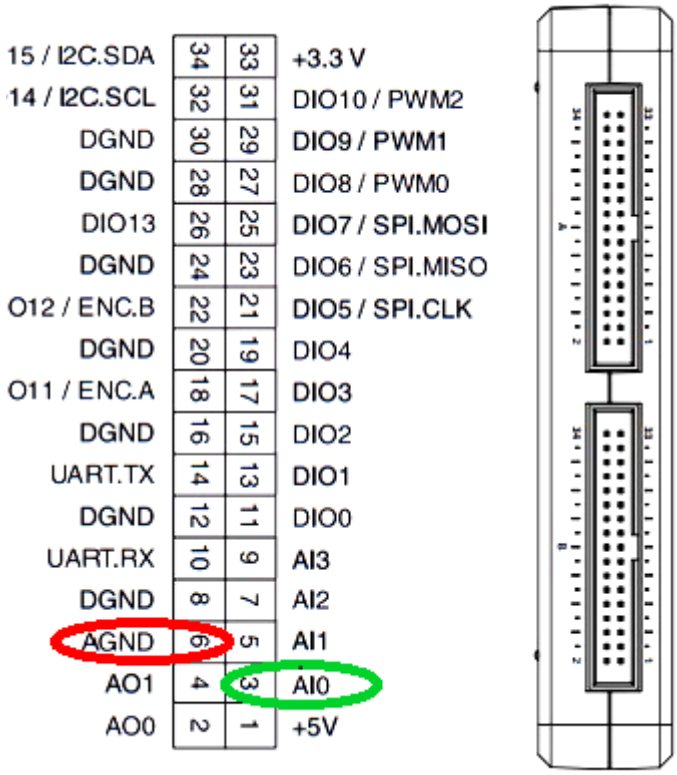

Figura 3. 24: Entrada analógica utilizada para conectar el transductor de presión.

Fuente: (Autor, 2018)

La alimentación del motor se hace en mediante el Controlador, el Controlador tiene 16 pines de los cuales estamos utilizando aproximadamente 13 pines. Cada uno de los pines tiene funciones especifica vitales para el funcionamiento de motor de pasos a continuación en la figura se visualiza cada uno de sus pines.

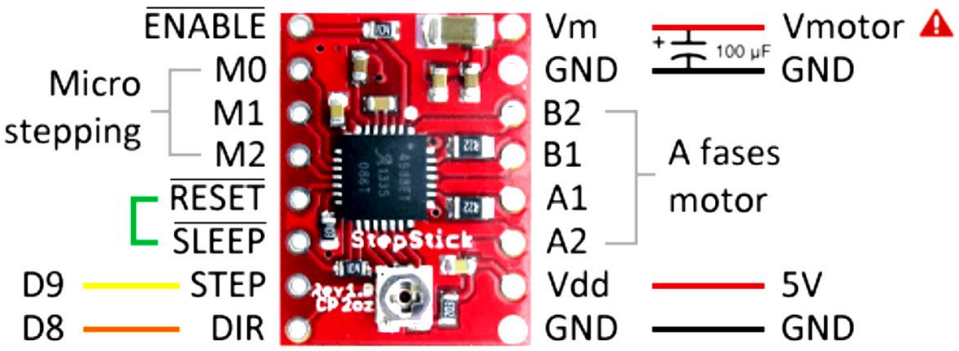

Figura 3. 25: Pines del Controlador A4988.

Fuente: (Autor, 2018)

Los pines 2B 2A / 1A y 1B se deben conectar al motor de pasos, el cual tiene cuatro pines de salida de un doble bobinado de los cuales dos pines corresponden al un bobinado y los otros dos restantes al otro bobinado, entonces los dos cables de un bobinado ir conectados al par 2B 2A o 1A 1B dependiendo de esta conexión va a ser el sentido de giro del motor por defecto.

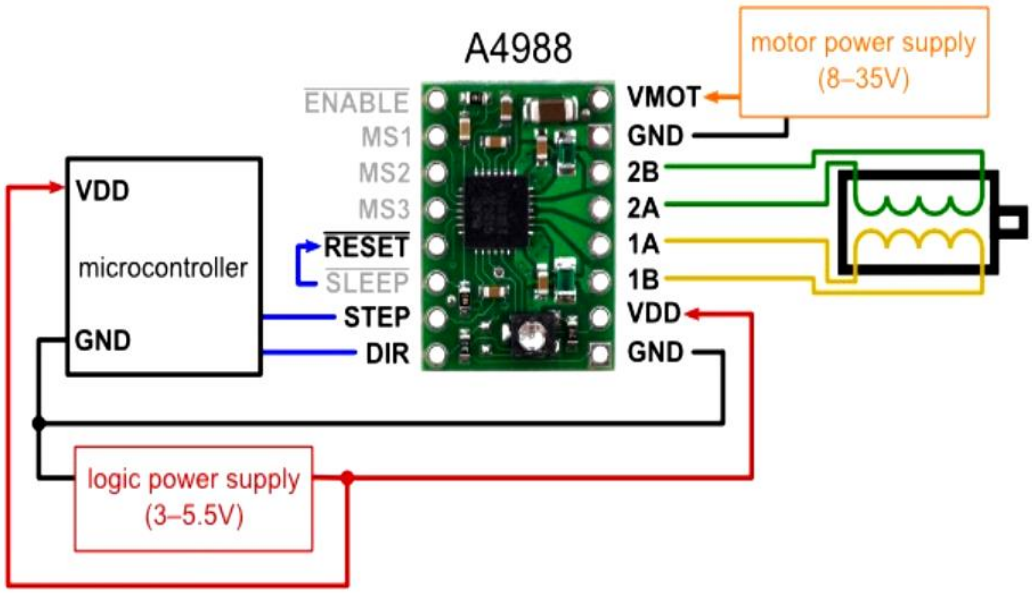

Figura 3. 26: Esquema general de conexión del controlador A4988.

Fuente: (Autor, 2018)

Los pines VDD y GND corresponden a la alimentación de este Controlador, la alimentacion que se le dá es de 5V, este voltaje es tomado de la fuente de la tarjeta myRIO.

Los pines VMOT y GND se conecta a la fuente de alimentación para el motor, puede ser un voltaje de entre 8 a 30V en DC en este caso se utiliza una fuente externa de 24V.

Las tierras como el GND de la alimentación del Controlador, el GND de la alimentación que va a ir al motor y las GND de la tarjeta myRIO se deben puentear para el funcionamiento.

Tenemos pines que son entradas digitales que permiten controlar el Controlador, por ejemplo, ENABLE si no tiene ningún pulso el motor funciona, tiene una lógica inversa, es decir si se le da un pulso el motor va a dejar de funcionar, es un pin habilitador.

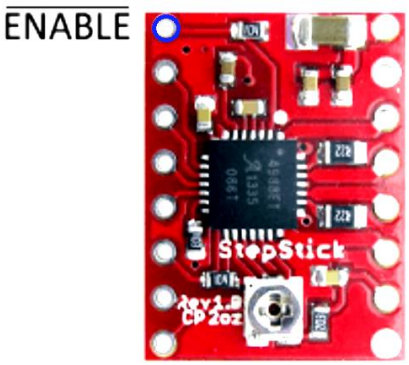

Figura 3. 27: Pin ENABLE del controlador A4988.

Fuente: (Autor, 2018)

Existen los pines MS1, MS2 y MS3, estos pines nos permiten controlar la precisión del paso a través de la combinación de los bits, es decir una combinación con números binarios con ello aparte del paso completo logramos obtener 1/16, 1/8, 1/4 y 1/2 de paso. En este caso ninguno de los tres va conectados porque el sistema no se requiere escalas de precisión sino velocidad por ello se trabaja a pasos completos.

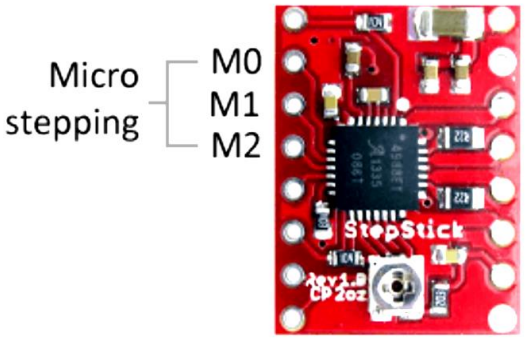

Figura 3. 28: Pines MS1, MS2 y MS3 del controlador A4988.

Fuente: (Autor, 2018)

Los pines RESET y SLEEP estos dos deben estar puenteados por características propias de configuración del Controlador.

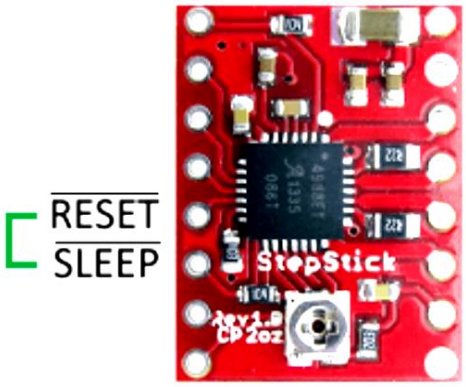

Figura 3. 29: Conexión de los pines RESET y SLEEP del controlador A4988.

Fuente: (Autor, 2018)

Los pines STEP y DIR se conecta a las entradas o salidas digitales que se encuentran en la tarjeta myRIO, con ello se puede controlar la frecuencia de los pasos y por ende la velocidad del motor. Si el pin DIR está en bajo es decir si no se le alimenta con los 5V la dirección va a estar en la que este configurada según la posición de las bobinas, pero si le damos un alto o sea una señal de 5V va a girar en la dirección contraria, el STEP cada que le damos un alto o un pulso el motor va a girar un paso, con este Controlador solo necesitamos 3 salidas digitales de la myRIO para controlar el motor.

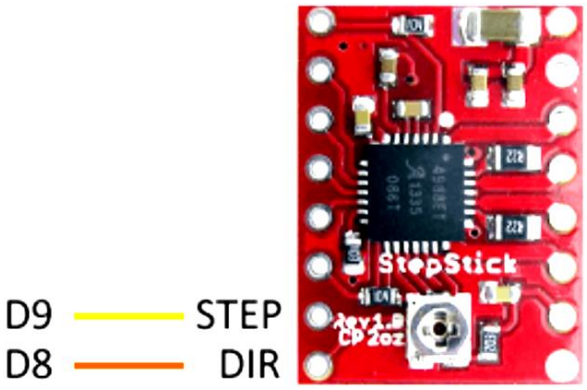

Figura 3. 30: Conexión de los pines STEP y DIR del controlador A4988.

Fuente: (Autor, 2018)

En este proyecto se está trabajando con el bus B de la tarjeta myRIO en este caso es importante que el encoder esté conectado a los pines que están configurados como entradas de encoder en la tarjeta myRIO. Se puede notar que se tiene dos entradas para encoder en cada puerto tanto en el bus A y bus B. En este caso se utilizó la entrada/salida digital DIO11 y DIO12 que corresponden al ENC.A y ENC.B conectando allí los pines del encoder CLK y DT que corresponden al ENC.A y ENC.B respectivamente.

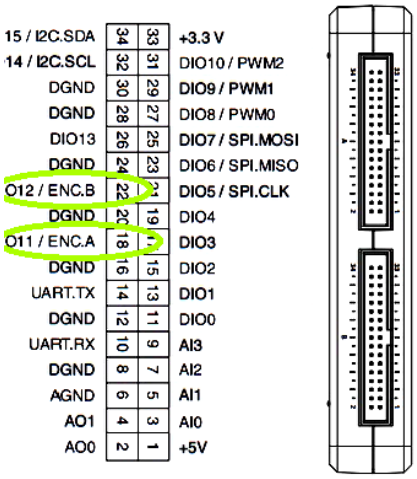

Figura 3. 31: Pines para entrada de encoder de la NI myRIO.

Fuente: (Autor, 2018)

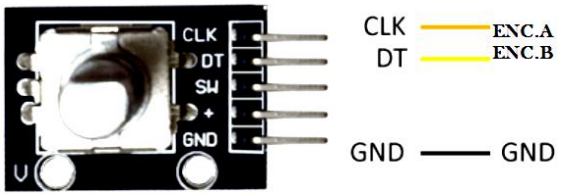

Figura 3. 32: Conexión del encoder con NI myRIO.

Fuente: (Autor, 2018)

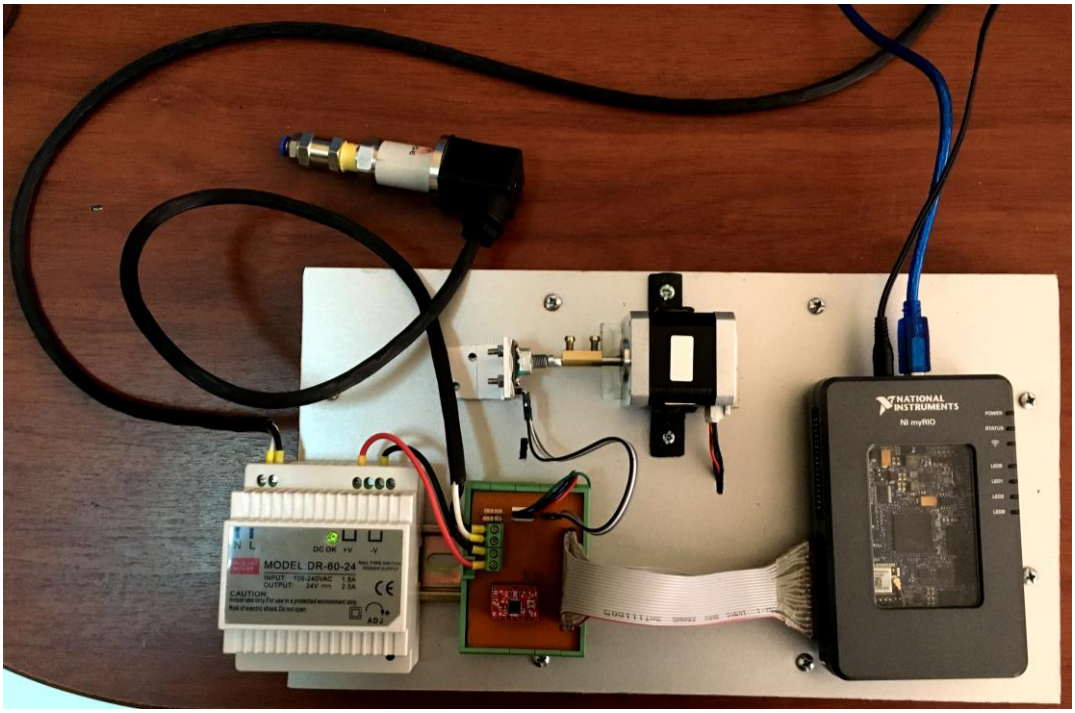

Figura 3. 33: Sistema de control conectado y terminado.

Fuente: (Autor, 2018)

3.3 Programación del sistema de control.

Energizamos la tarjeta NI myRIO y la conectamos al computador con ayuda del cable de comunicación USB 2.0 Hi-Speed. Después se abre el programa LabVIEW.

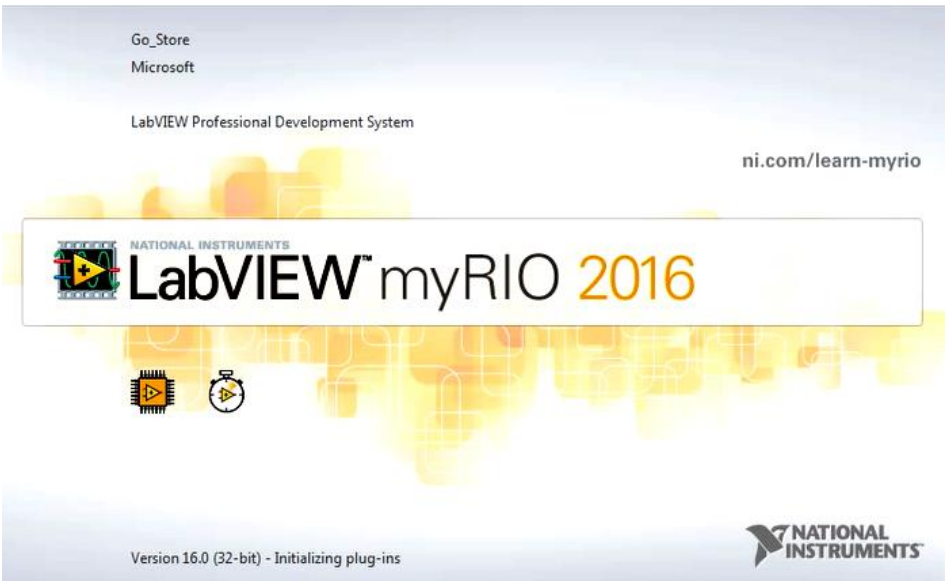

Figura 3. 34: Software LabVIEW myRIO.

Fuente: (Autor, 2018)

Una vez que el programa está abierto da opciones a escoger, en este caso se escoge la opción de crear nuevo proyecto.

Figura 3. 35: Crear nuevo proyecto en LabVIEW.

Fuente: (Autor, 2018)

Aparece un menú en el cual pide escoger un punto de partida para desarrollar el proyecto, como en el presente trabajo de titulación se planteó desarrollar un sistema de control con sistemas embebidos, entonces se pincha en la opción *myRIO*, como consecuencia de eso se despliega otro submenú del cual se escoge la opción *myRIO Custom FPGA Project* y se aceptamos en la opción *Next*.

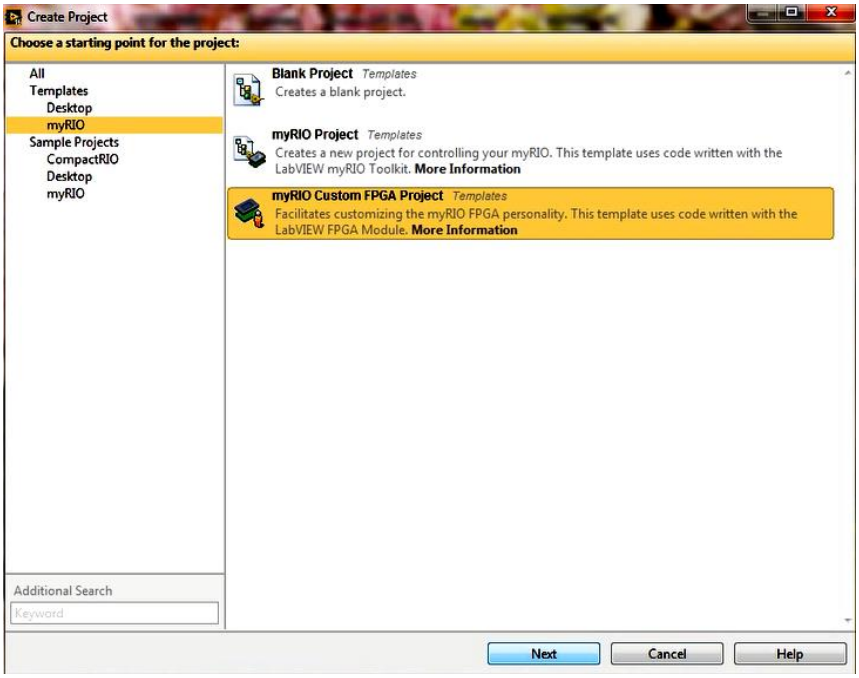

Figura 3. 36: myRIO Custom FPGA Project.

Fuente: (Autor, 2018)

Después de esto el software reconoce la tarjeta NI myRIO, esta pestaña es la *configuración del nuevo proyecto* en *myRIO custom FPGA Project*, una vez que la tarjeta ha sido reconocida en su totalidad se da clic en *Finish.*

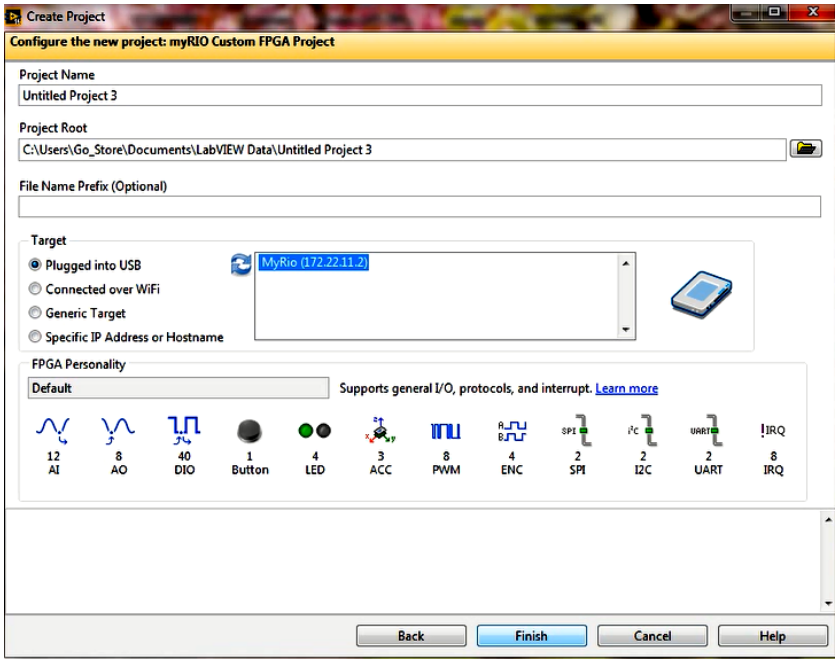

Figura 3. 37: Configuración del nuevo proyecto en myRIO.

Fuente: (Autor, 2018)

Finalmente se crea ha creado el nuevo proyecto.

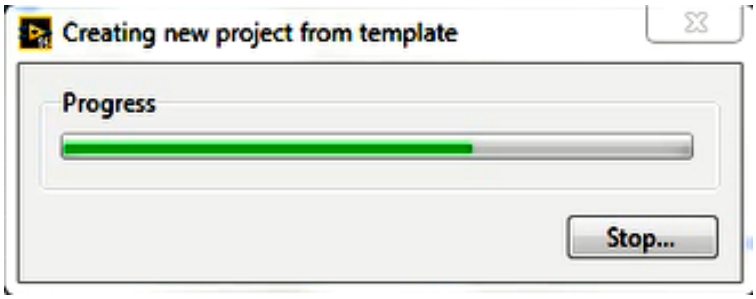

Figura 3. 38: Creación del nuevo proyecto en LabVIEW myRIO.

Fuente: (Autor, 2018)

Una vez que se ha creado el nuevo proyecto se tiene un menú en el cual se escoge la tarjeta reconocida *MyRIO (172.22.11.2)*, después clic en *Chassis (myRIO-1900)*, se despliega un submenú, se ubica el cursor en la opción *FPGA targert (RIO0, myRIO-1900)*, del cual se pincha la opción *New* y finalmente se escoge la opción *VI*.

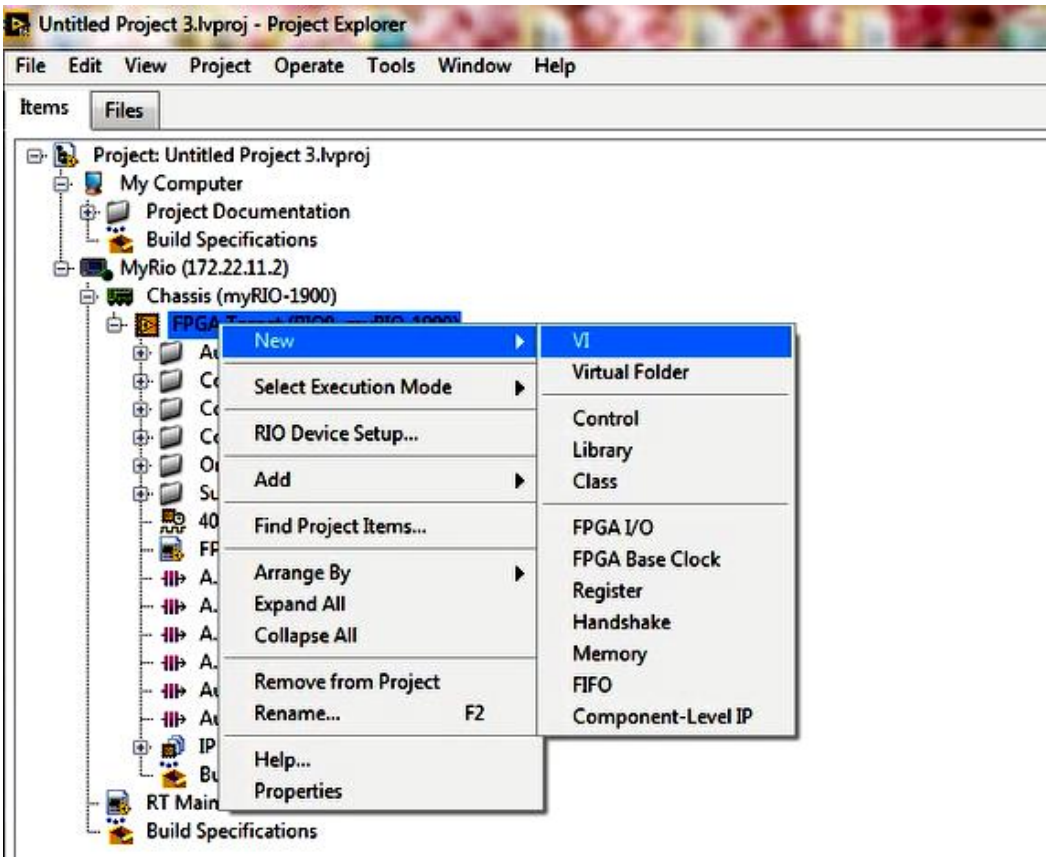

Figura 3. 39: Creación del nuevo VI en LabVIEW myRIO.

Fuente: (Autor, 2018)

Se obtiene como resultado el nuevo *VI* necesario para empezar la programación del sistema.

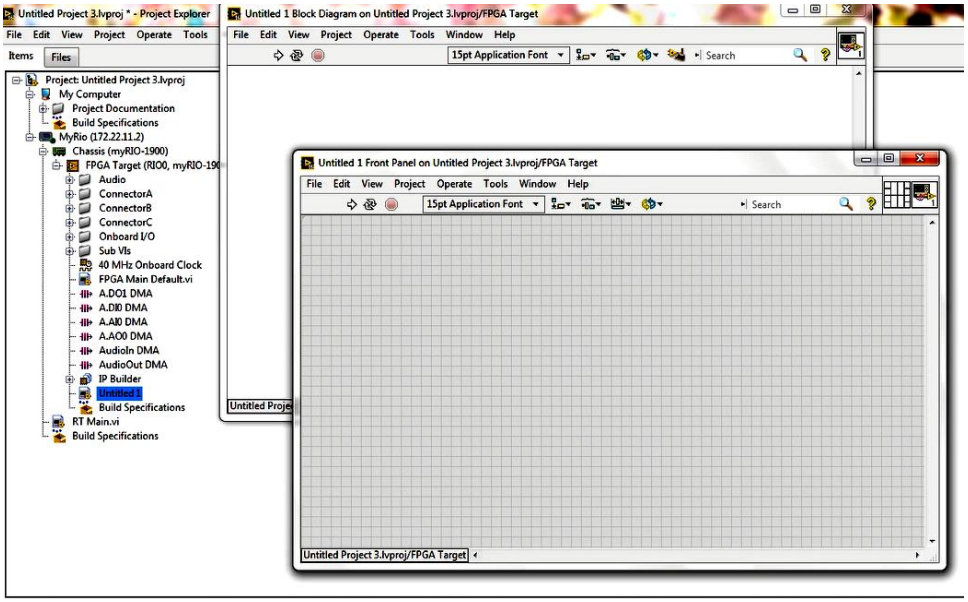

Figura 3. 40: Nuevo VI en LabVIEW myRIO.

Fuente: (Autor, 2018)

Para desarrollar el sistema embebido necesario programar en lenguaje FPGA, entonces para declarar una entrada o salida en modo FPGA en el mismo *VI* se dirige a la carpeta de *FPGA I/O* de entradas y salidas y se selecciona un nodo de entrada y salida *I/O NODE*, esto sirve tanto para las analógicas y digitales.

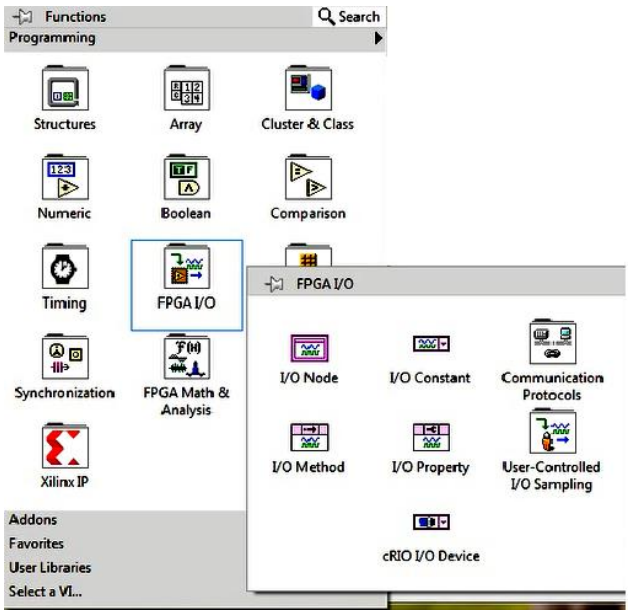

Figura 3. 41: Carpeta de FPGA I/O de entradas y salidas

Fuente: (Autor, 2018)

Una vez seleccionado el nodo declaramos la entrada o salida dependiendo de lo que se necesita se puede escoger en el menú entre *Audio, Connector A, Connector B, Connector C y Onboard I/O*. En este nodo se tiene todas las entradas y salidas que se pueden administran de la tarjeta NI myRIO, cada uno de estos elementos se subdividen.

Figura 3. 42: Menú del I/O NODE

Fuente: (Autor, 2018)

Figura 3. 43: Menú del Connector A del I/O NODE

Fuente: (Autor, 2018)

Figura 3. 44: Menú del Connector A del I/O NODE

Fuente: (Autor, 2018)

Para este caso se eligió el Connector B en cual se utiliza y programa las entradas y salidas necesarias para el funcionamiento del motor, transductor de presión, para la configuración del controlador A 4988, etc.

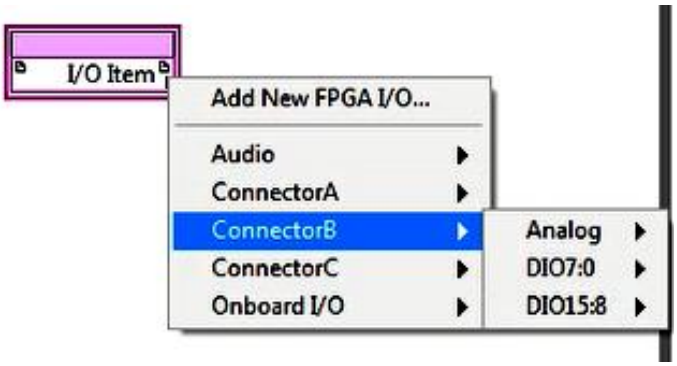

Figura 3. 45: Menú del Connector B del I/O NODE

Fuente: (Autor, 2018)

Figura 3. 46: Menú del Connector C del I/O NODE

Fuente: (Autor, 2018)

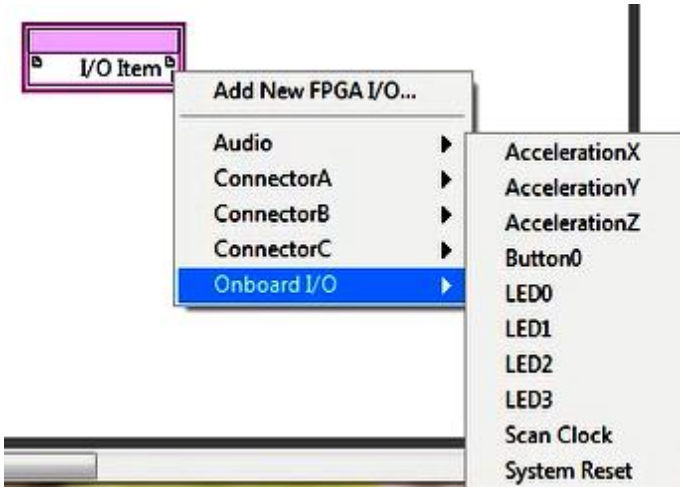

Figura 3. 47: Menú del Onboard I/O del I/O NODE

Fuente: (Autor, 2018)

Para el control de la velocidad del motor se tiene la entrada del *ENABLE,* como se dijo anteriormente, este es un pin habilitador, con esto se puede controlar el arranque del motor o su apagado, este está programado en el *Connector B* en el submenú *DIO15-8* y se elige la opción *Connector B/DIO0*.

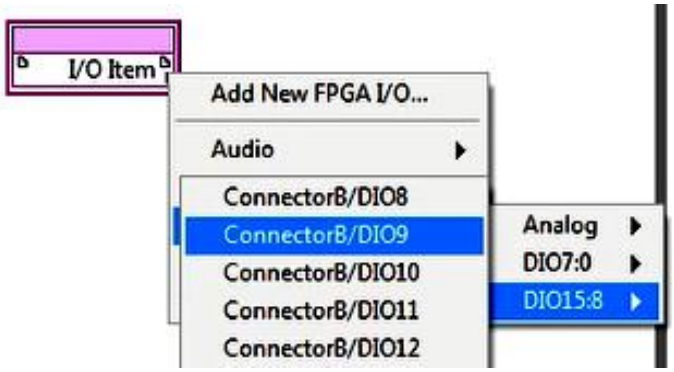

Figura 3. 48: Selección de salida digital para el pin ENABLE.

Fuente: (Autor, 2018)

El nodo da la opción para configurarlo como salida o entrada según la necesidad que se requiera. Una vez que se declara el nodo el programa lo procesa y lo carga dejándolo listo para su uso y así de esta manera se programa cada una de las entradas y salidas en LabVIEW.

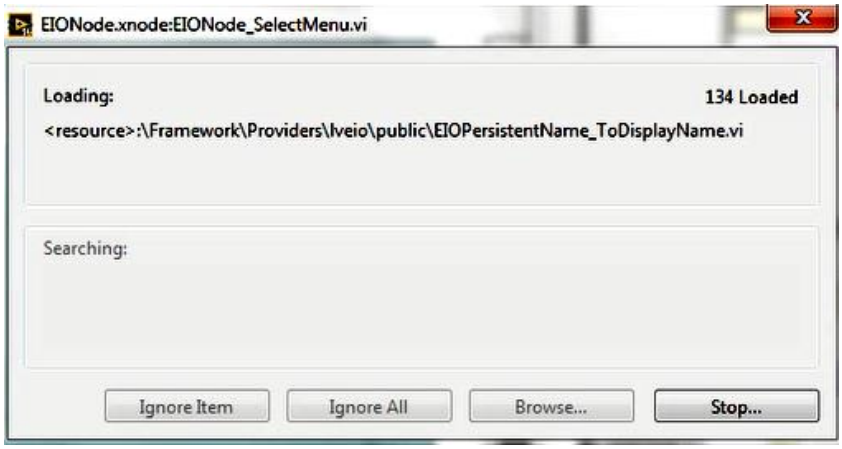

Figura 3. 49: Creando la salida digital para el pin ENABLE.

Fuente: (Autor, 2018)

Figura 3. 50: Salida digital para el pin ENABLE creada.

Fuente: (Autor, 2018)

Figura 3. 51: Salida digital declarada para el arranque y paro del motor.

Fuente: (Autor, 2018)

Con el mismo procedimiento se programó la salida en el *Connector B/DIO1* para la inversión de giro.

Figura 3. 52: Salida digital declarada para la inversión de giro del motor.

Fuente: (Autor, 2018)

Como se controla al motor de pasos en el pin *STEP*, por ejemplo, si le damos un alto va a dar un paso, si se le da otro alto dará otro paso, entonces para mantener una velocidad constante va a depender de la frecuencia que den los pasos por ende la magnitud de los periodos que estos tenga. Para esto se usó un *Dial* es nos permite enviar una señal analógica, este elemento se lo encuentra en la carpeta de *Numeric.*

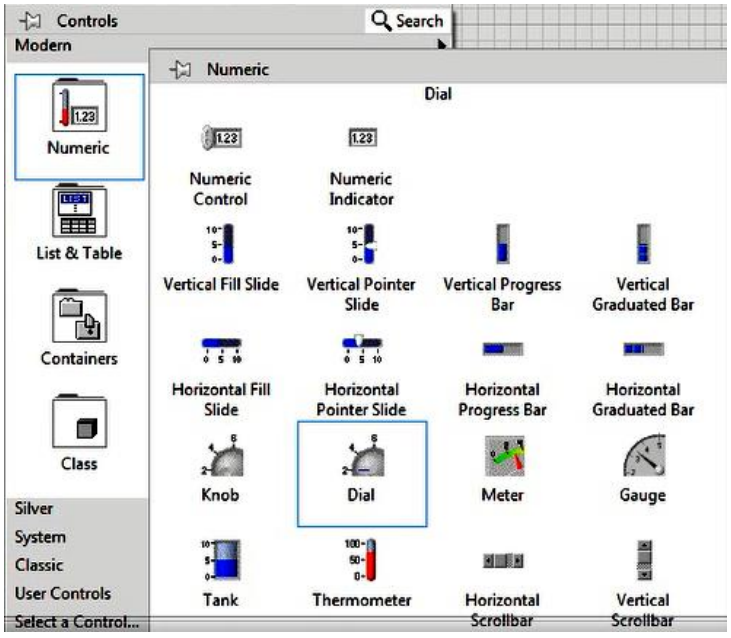

Figura 3. 53: Dial

Fuente: (Autor, 2018)

Entonces para programar al Dial el cual representa la frecuencia (f) que se dan los pasos, se tiene de dato que la frecuencia máxima es de 350, entonces se divide la frecuencia para mil con ello se transforma de milisegundos que trabajan los elementos en el programa a segundos, al dividir la unidad para esta frecuencia se obtiene el periodo total (T), el cual es necesario para definir los tiempos de encendido (Ton) y tiempo de apagado (Toff).

Entonces se dice que el *Ton* es igual al *T* por el *Duty Cycle*, donde el Duty Cycle se definió en 50% (0.5) y por ende el *Toff* es la diferencia entre *T* y *Ton.*

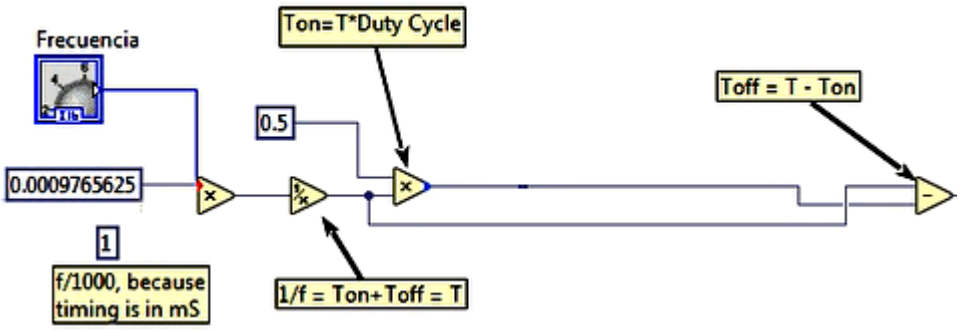

Figura 3. 54: Programación del *Dial*

Fuente: (Autor, 2018)

Con los tiempos obtenidos se procede realiza una estructura de secuencia para el control de la velocidad del motor. Entonces se declara que la secuencia empieza con un alto para el *ON* para que el programa identifique este alto se ingresa un *True [T]*, este *True* es una entrada booleana.

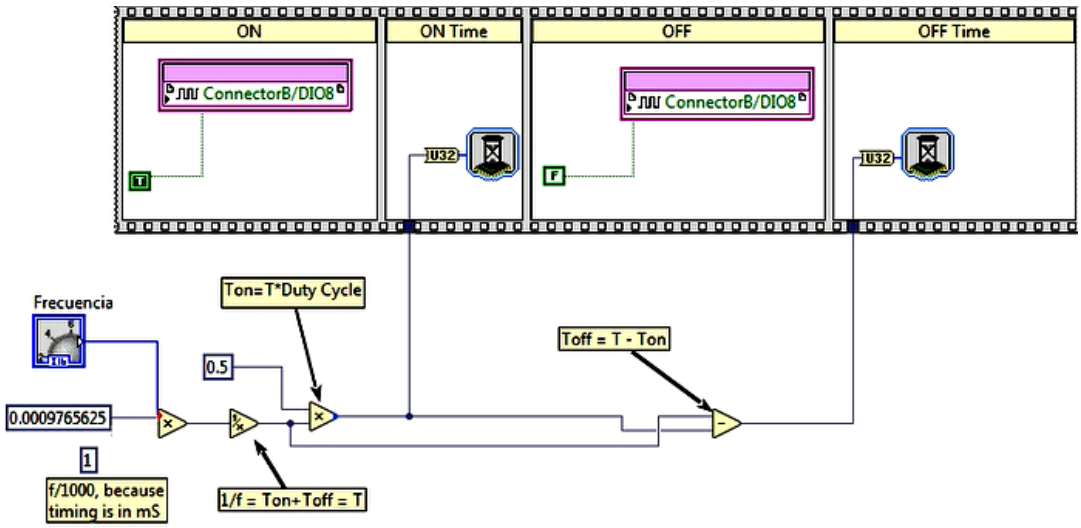

Figura 3. 55: Secuencia para el arranque y apagado del motor.

Fuente: (Autor, 2018)

El siguiente paso de la estructura de secuencia es declarar por cuanto tiempo va a estar encendido, allí va a ir la señal obtenida por el *Ton*. Después se declara el otro paso de la secuencia con un bajo para el *OFF* para que el programa identifique este bajo se ingresa un *False [F]*, este *False* es otra entrada booleana y para el último paso de la estructura de secuencia que es el tiempo de apagado, se conecta la señal del *Toff*.

Para la programación del encoder se recomienda primero conocer la estructura básica del encoder de cuadratura.

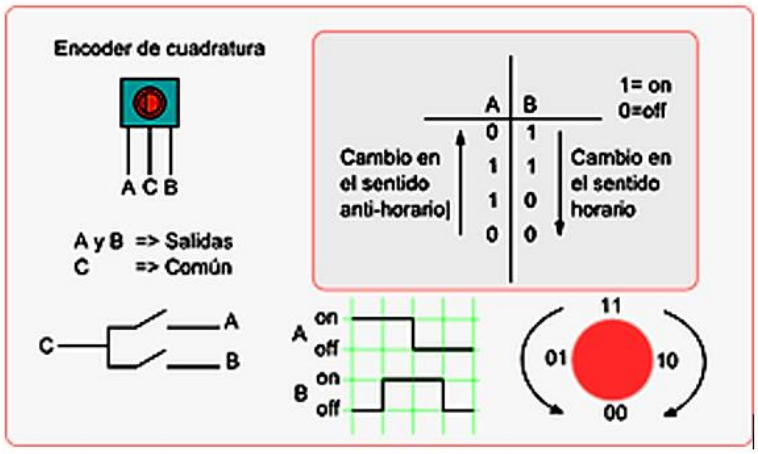

Figura 3. 56: La estructura básica del encoder de cuadratura.

Fuente: (Autor, 2018)

La imagen muestra un encoder rotativo de tipo mecánico. Estos contactos mecánicos se van activando y desactivando en una secuencia que permite saber la dirección y el número de desplazamientos que han ocurrido en el encoder. En la imagen también se puede ver cómo es la secuencia de activaciones dependiendo el sentido de giro.

Entonces, si se asume a la salida “A” como el bit menos significativo y la salida “B” como el bit más significativo, en la tabla de verdad se sabrá la dirección de giro si se compara la posición actual con la anterior.

Como ya se mencionó antes se utiliza puertos exclusivos de la myRIO para la configuración del encoder. El encoder de cuadratura tiene dos salidas digitales Canal A y Canal B que son leídas en las entradas digitales *Connector B/DIO11* y *Connector B/DIO12* del puerto.

La adquisición de datos se realiza a través de los canales A y B

Figura 3. 57: Entradas/salidas digitales para encoder.

Fuente: (Autor, 2018)

Para analizar al encoder se utiliza combinación binaria y puertas OR y XOR, se analiza los estados de los canales A y B, al determinar su estado actual y estado anterior, con ello se puede saber si se ha dado un paso y en qué sentido se ha dado. La lógica binaria ayuda a determinar si se tuvo un avance o un retroceso. Se designó que B y A sean los estados actuales de los valores del encoder con respecto a B' y A' sean los estados anteriores de los valores del encoder. Además, se añadió un botón para reseteo con el fin de regresar a la posición inicial.

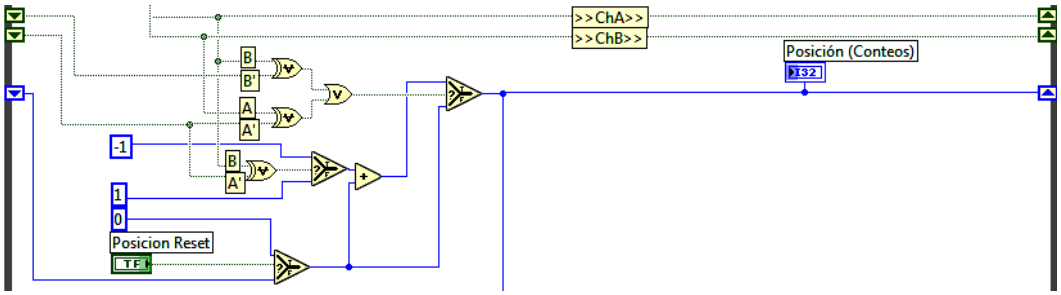

Figura 3. 58: Estados de los canales A y B.

Fuente: (Autor, 2018)

A -> Valor actual del canal A del codificador en cuadratura

A '-> Último valor del codificador en cuadratura canal A

B -> Valor del canal B del codificador de cuadratura actual

B '-> Último valor del codificador de cuadratura del canal B

Función XOR: Calcula el valor lógico exclusivo or (XOR) de las entradas. Ambas entradas deben ser valores booleanos, valores numéricos o clústeres de error. Si ambas entradas son VERDADERAS o ambas entradas son FALSAS, la función devuelve FALSO. De lo contrario, devuelve VERDADERO.

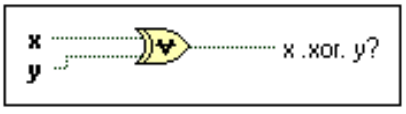

Figura 3. 59: Función XOR.

Fuente: (Autor, 2018)

Exclusive Or Details

Exclusive Or truth table

x	y	x .xor. y?
T	T	F
T	F	T
F	T	T
F	F	F

Figura 3. 60: Tabla de verdad de la función XOR.

Fuente: (Autor, 2018)

Función OR: Calcula el OR lógico de las entradas. Ambas entradas deben ser valores booleanos, valores numéricos o clústeres de error. Si ambas entradas son FALSAS, la función devuelve FALSO. De lo contrario, devuelve VERDADERO.

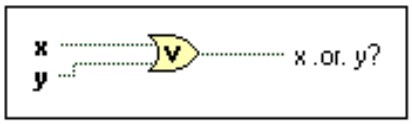

Figura 3. 61: Función OR.

Fuente: (Autor, 2018)

Or Details

Or truth table

x	y	x .or. y?
T	T	T
T	F	T
F	T	T
F	F	F

Figura 3. 62: Tabla de verdad de la función OR.

Fuente: (Autor, 2018)

Función Select: Devuelve el valor conectado a la entrada t o entrada f, dependiendo del valor de s. Si s es VERDADERO, esta función devuelve el valor conectado a t. Si s es FALSO, esta función devuelve el valor conectado a f.

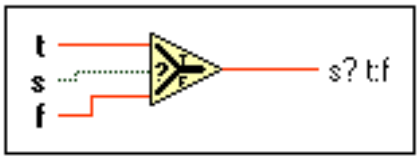

Figura 3. 63: Función Select.

Fuente: (Autor, 2018)

Mediante la función OR exclusiva se detectan los cambios de estado del valor anterior del canal A o B con respecto a su valor actual, estos resultados son las entradas de una función or la cual determina un incremento o decremento dependiendo del sentido de giro. El sentido de giro es determinado mediante una compuerta or exclusiva cuyas entradas son el estado actual del canal B y el estado anterior del canal A, en sentido horario el valor resultante será falso y en sentido antihorario el valor resultante será verdadero.

Utilizando la función select se asigna el valor de la unidad con signo negativo cuando el resultado de la función anterior es verdadero y cuando es falso se asigna el valor de la unidad con signo positivo. El valor de 1 o -1 es adicionado al valor anterior de la posición para obtener el valor de la posición actual. El botón de reset envía el valor 0 a la entrada de verdadero de la función select para asignar este valor a la posición actual.

Comparación de cambios de valores en la posición

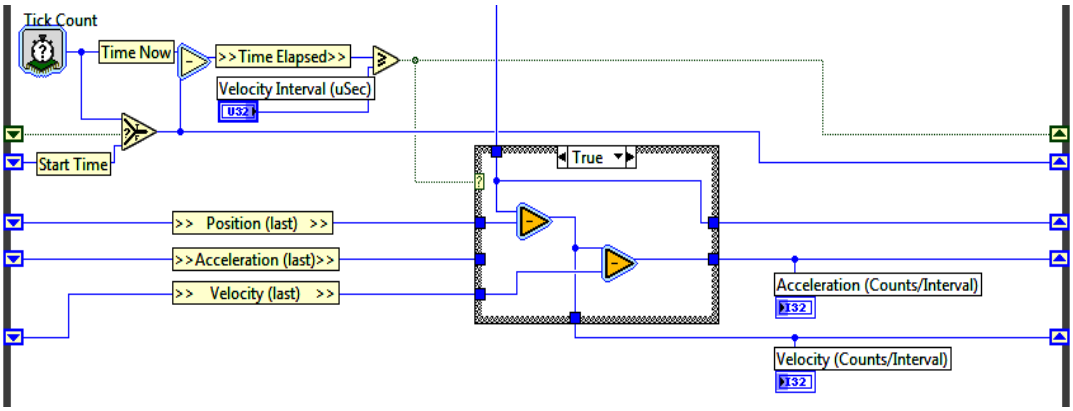

Figura 3. 64: Comparación de cambios de valores en la posición.

Fuente: (Autor, 2018)

Se utiliza un contador interno en microsegundos para determinar el intervalo de tiempo en el cual se analiza la variación de la posición, el tiempo transcurrido se compara con el valor que asignamos (1000000 us), para que cada intervalo de 1s se active el bloque de comparación if. Éste bloque aplica un tratamiento diferente a las variables de entrada en función de la entrada

booleana de control, si esta entrada es verdadera entonces se obtiene la diferencia entre el valor actual de posición y el valor leído un segundo atrás, y así se determina el valor de la velocidad en conteos/seg, de manera análoga se obtiene el valor de la aceleración.

Para la parte final de la programación se realiza una con conversión de unidades, como el resultado que se obtenía por la programación era de conteos/seg. Entonces para transformarle este valor a RPM, se dividió las constantes 60/80 y este resultado se multiplico por los conteos/seg y de esta forma se obtiene las RPM del motor.

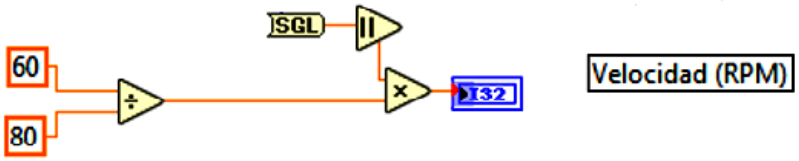

Figura 3. 65: Conversión de conteos/seg a RPM.

Fuente: (Autor, 2018)

Para el bloque que controla el transductor de presión se declaró un nuevo nodo de entradas/salidas de FPGA y se configuró en el *connector B* en *Analog* se seleccionó la entrada *Conecctor B/AI0* y se encuentra declarada de la siguiente manera.

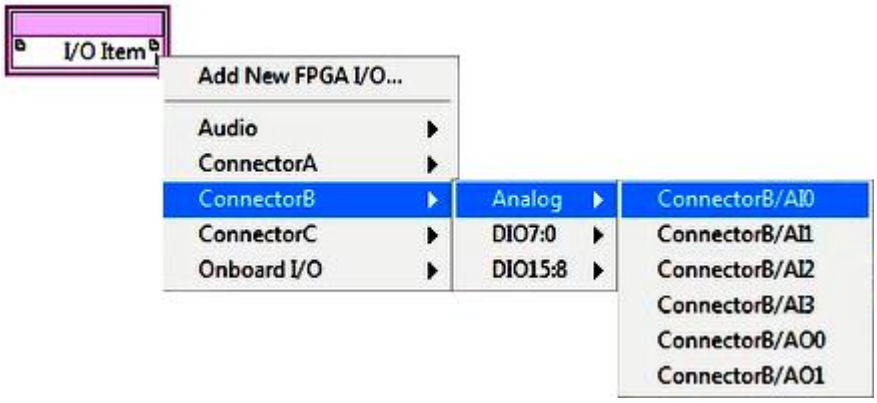

Figura 3. 66: Declaración de la entrada analógica para el transductor de presión.

Fuente: (Autor, 2018)

Para obtener una lectura clara de la señal del transductor se realiza un escalamiento multiplicando por constantes con estas relaciones se obtiene una escala de 0 a 10 Bars de presión, para confirmar estos valores se introdujo los controles y componentes gráficos de LabVIEW útiles para su visualización. Además de lo realizado se insertó una alarma de

seguridad que se activa a los 7 Bars de presión al mismo tiempo enciende unos focos tanto en el panel del principal, así como uno de los leds de la tarjeta.

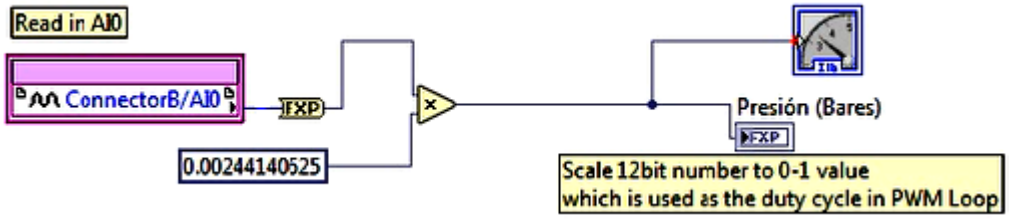

Figura 3. 67: Programación para el transductor de presión.

Fuente: (Autor, 2018)

Debido a la inestabilidad de la señal obtenida desde el transductor es necesario la utilización de un filtro pasa bajos *Butterworth* el que se encuentra en *Functions* la carpeta de *FPGA Math & Analysis*.

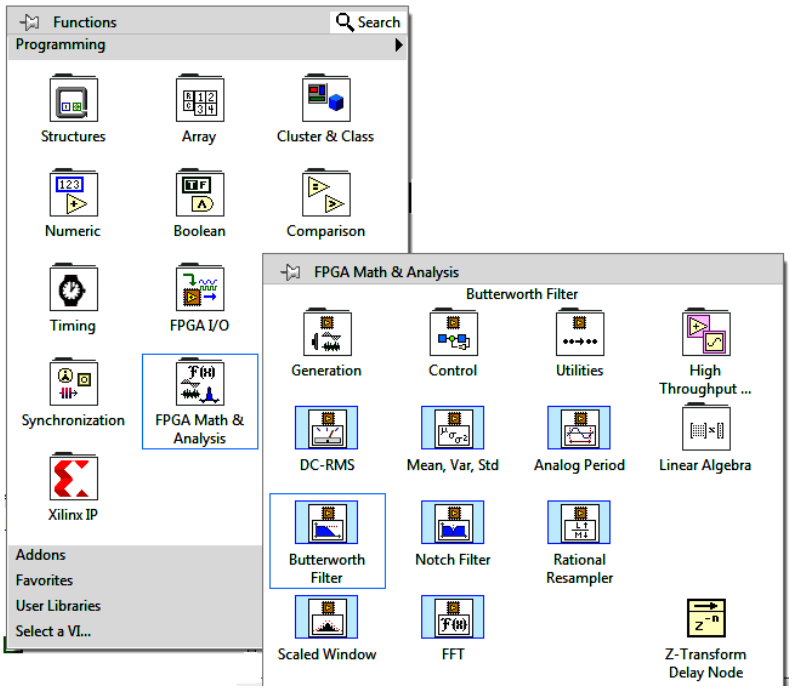

Figura 3. 68: Filtro Butterworth.

Fuente: (Autor, 2018

Este filtro está configurado con una frecuencia de corte de 50 Hz y con orden 4, esto nos ayuda a eliminar las señales de externas indeseables que se encuentran presentes en la lectura de la señal analógica del transductor de presión.

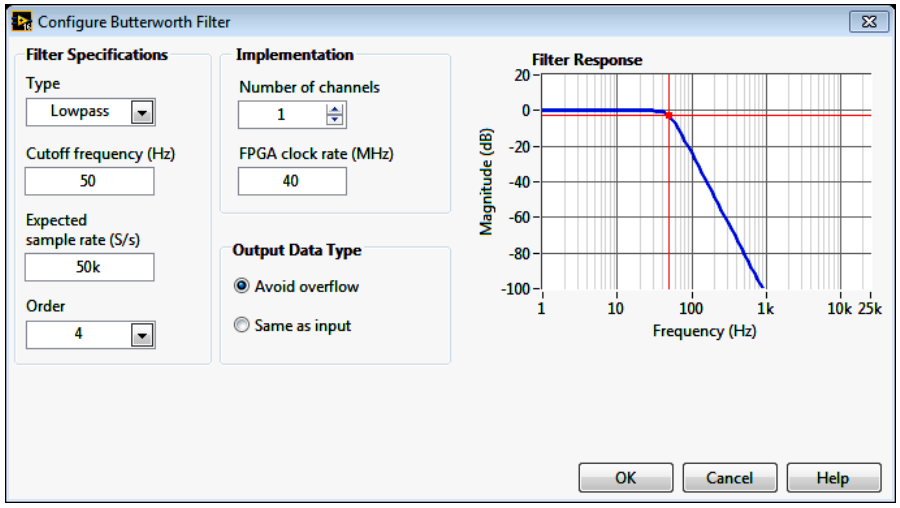

Figura 3. 69: Configuración del Filtro Butterworth.

Fuente: (Autor, 2018

Adicional a todo lo descrito se recalca que todas las programaciones se realizaron dentro de una estructura iterativa *While Loop*, esto funciona para que el programa se mantenga leyendo y escribiendo los valores a cada momento.

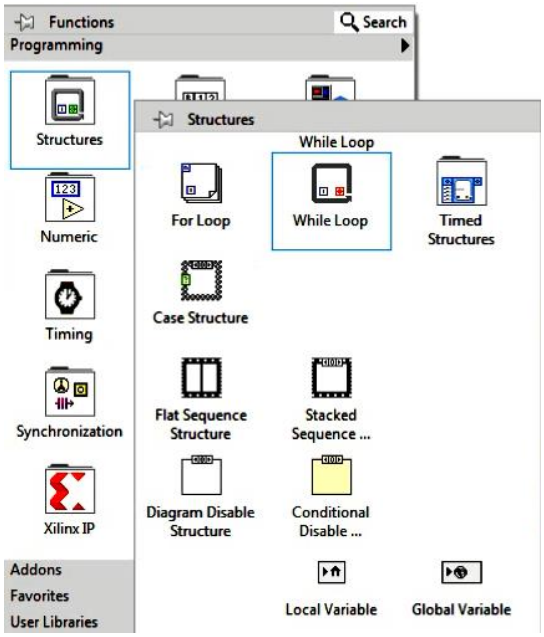

Figura 3. 70: Estructura While Loop

Fuente: (Autor, 2018)

Para la presentación y visualización de la programación se ha desarrollado una interfaz en donde podemos tener una página principal utilizando un componente llamado *Tab Control*, esto permite tener varias pestañas en un mismo *VI*, en este caso una pestaña para portada otra

para la velocidad y una última para la presión. Con la ayuda de los controles y componentes gráficos de LabVIEW se diseñó dichas pestañas.

Figura 3. 71: Portada principal.

Fuente: (Autor, 2018)

Se diseño una portada para el control de la velocidad del motor de pasos y la lectura de las señales del encoder.

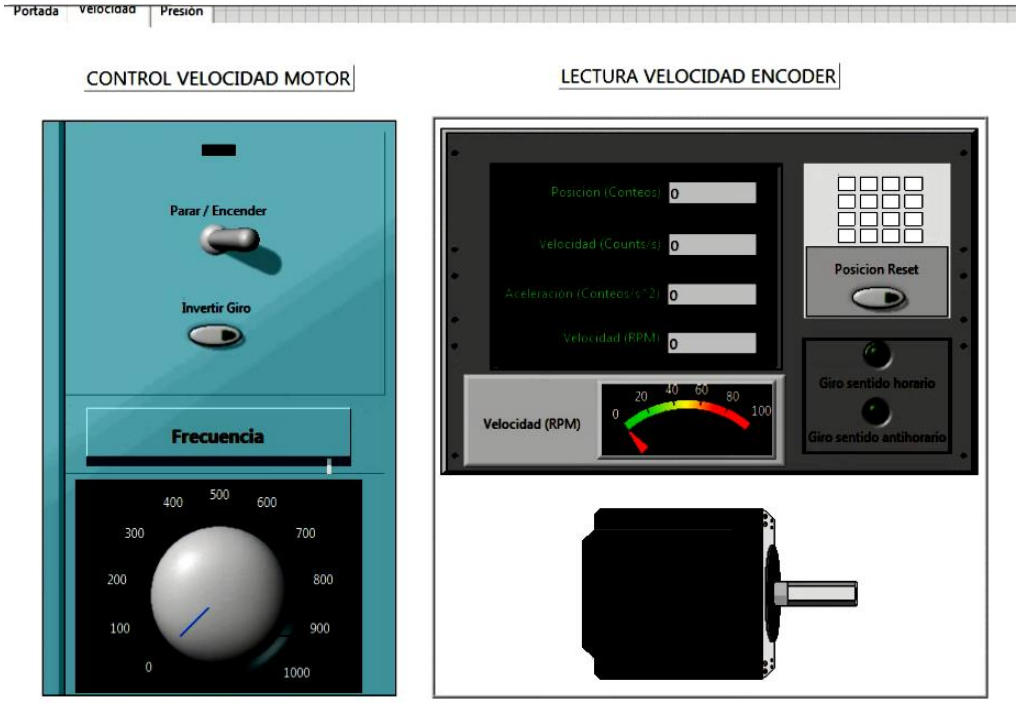

Figura 3. 72: Portada para medir la velocidad.

Fuente: (Autor, 2018)

También se diseñó una portada para el control de la presión.

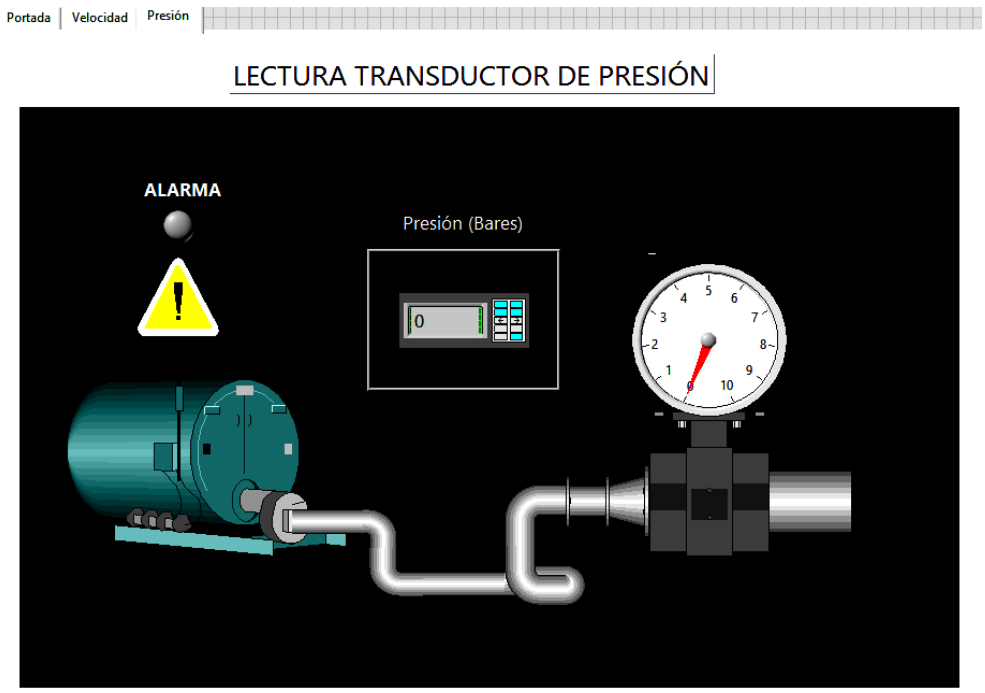

Figura 3. 73: Portada para medir la presión

Fuente: (Autor, 2018)

Una vez que se ha concluido la programación y diseño del sistema se procede a cargar el programa en la tarjeta utilizamos el icono de *run*. Este se encuentra en la esquina superior izquierda, se da clic y empezara a cargarse el programa.

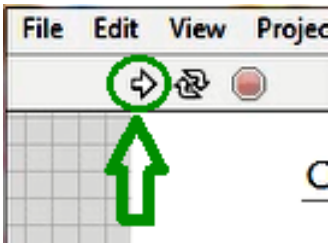

Figura 3. 74: Icono Run.

Fuente: (Autor, 2018)

Producto de ello saldrá una ventana en la cual se elige la opción *Use el servidor de compilación local* y aceptamos.

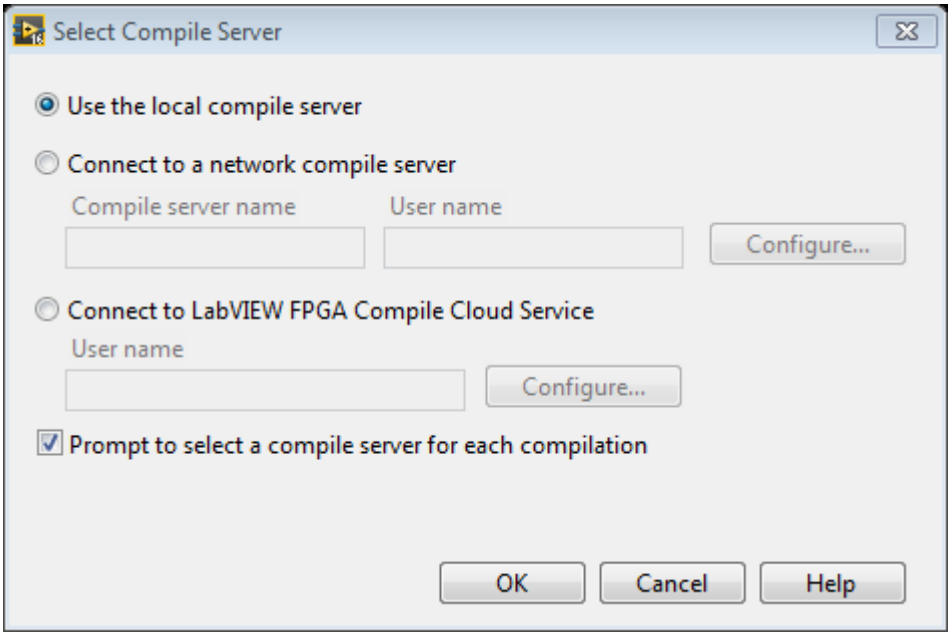

Figura 3. 75: Selección del servidor de compilación.

Fuente: (Autor, 2018)

Entonces empezara a cargarse la configuración en la myRIO, producto de esto generara muchas ventanas que son normales que aparezcan a continuación adjuntamos algunas de ellas.

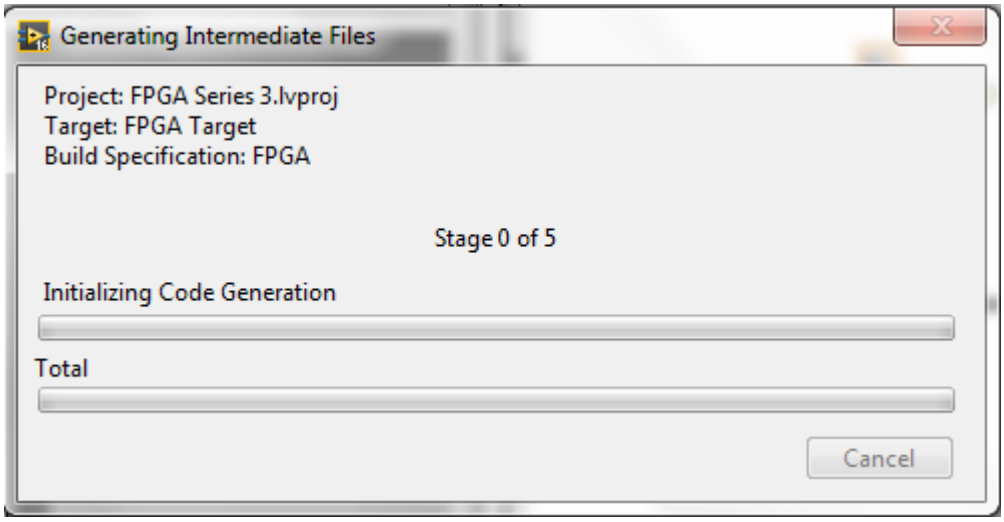

Figura 3. 76: Inicio de Generating intermediate files

Fuente: (Autor, 2018)

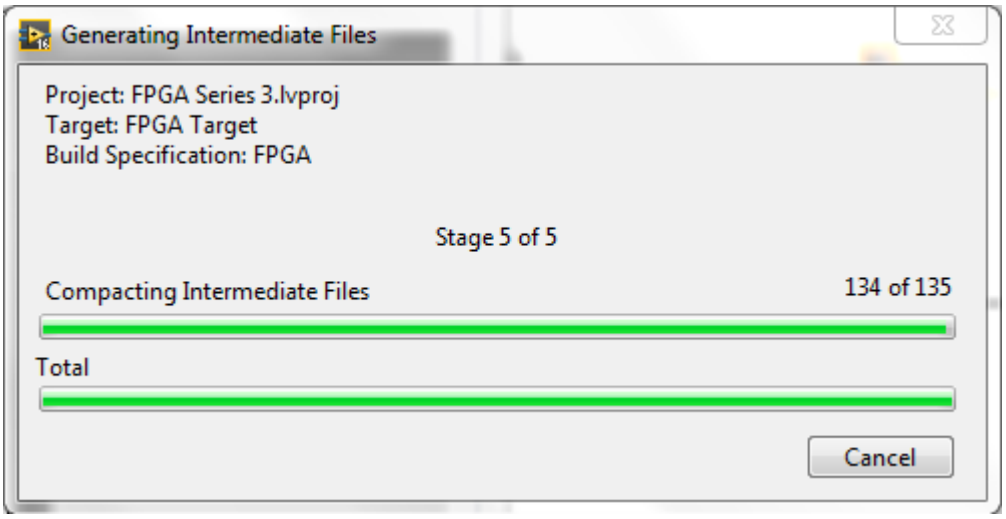

Figura 3. 77: Finalización de Generating intermediate files

Fuente: (Autor, 2018)

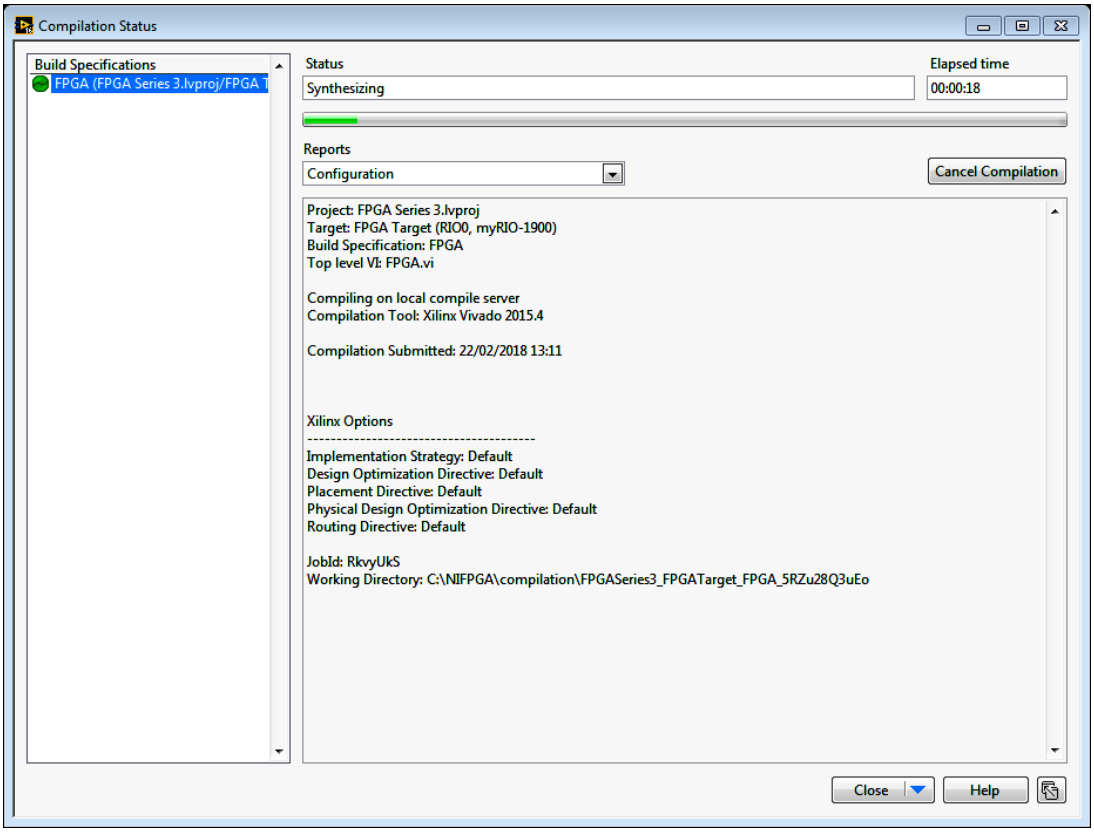

Figura 3. 78: Compilation Status: Synthesizing.

Fuente: (Autor, 2018)

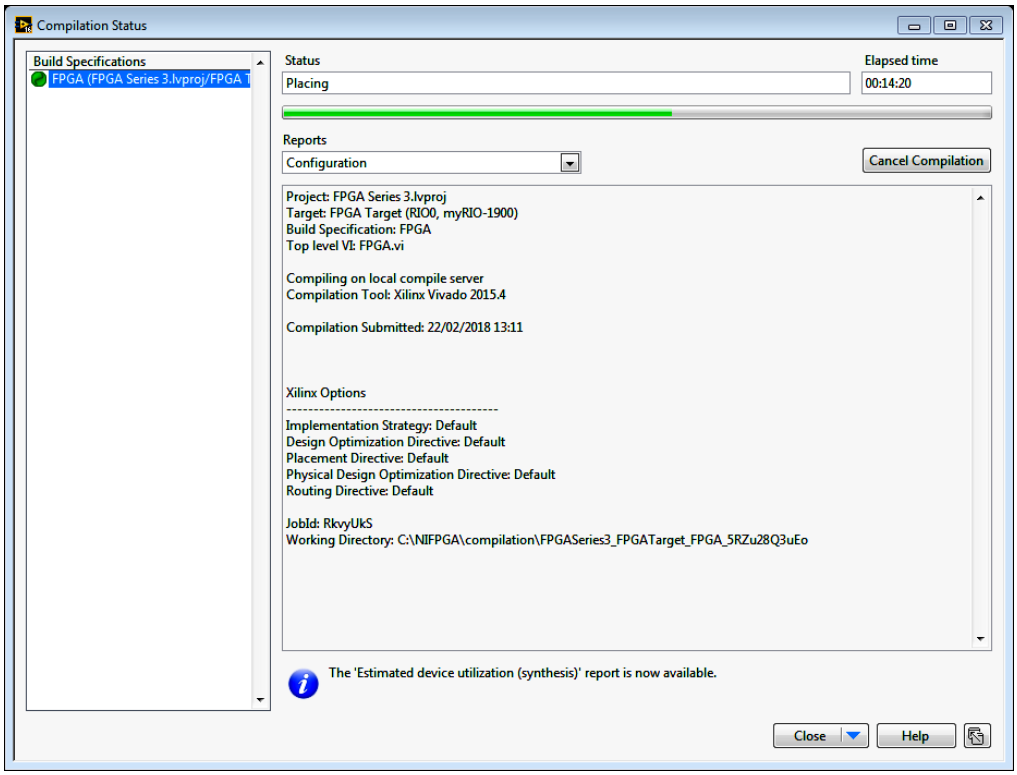

Figura 3. 79: Compilation Status: Placing.

Fuente: (Autor, 2018)

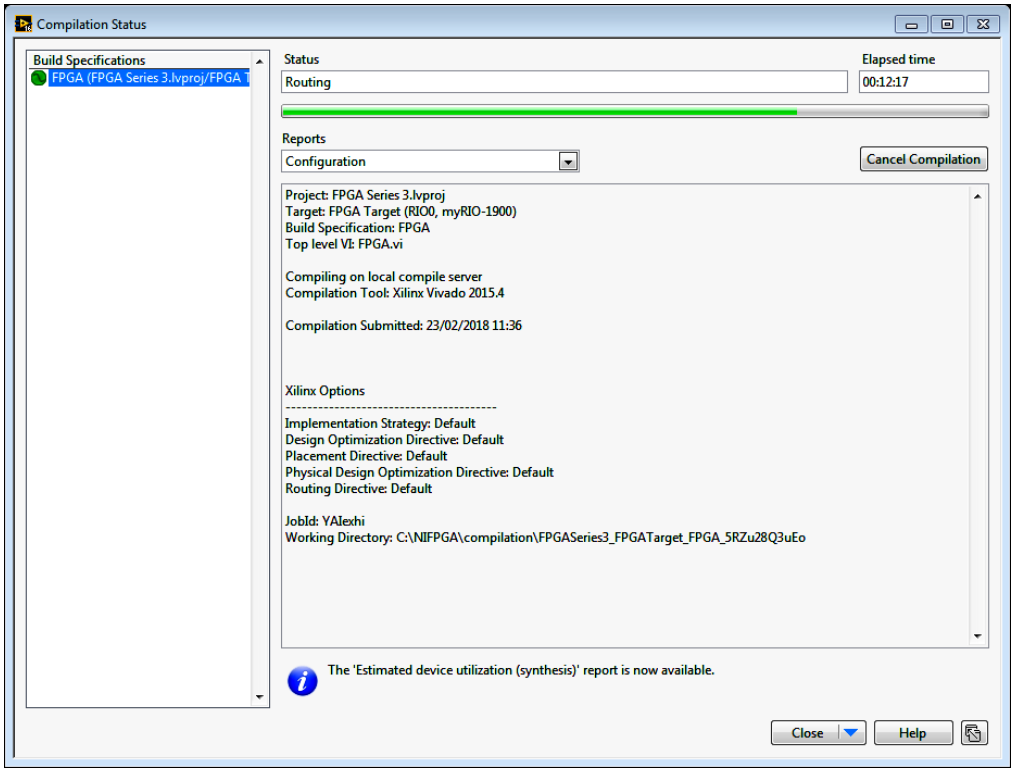

Figura 3. 80: Compilation Status: Placing.

Fuente: (Autor, 2018)

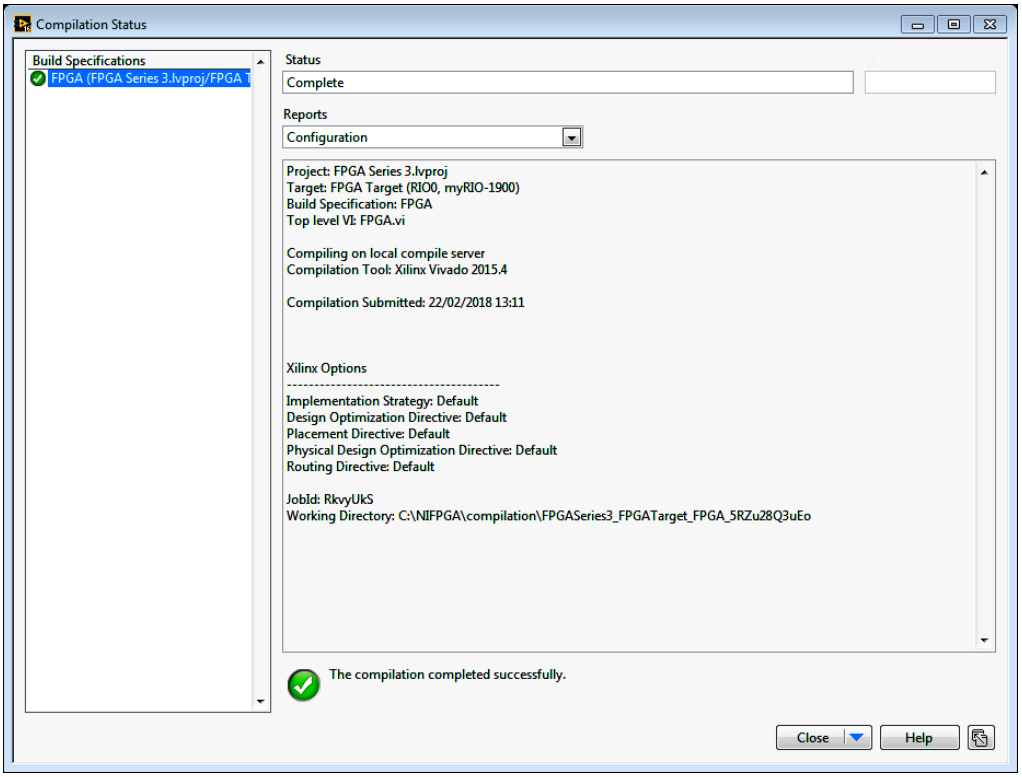

Figura 3. 81: Compilation Status: Complete.

Fuente: (Autor, 2018)

Una vez que ha terminado el proceso de carga se da clic en *Close* y el sistema está listo para las pruebas de funcionamiento y manipulación. Finalmente, si se desea parar el programa se pincha en la opción *Stop* esta opción está programada en cada pestaña de control.

Figura 3. 82: Icono Stop.

Fuente: (Autor, 2018)

CAPÍTULO IV

4. Conclusiones y recomendaciones

4.1 Conclusiones

Se comprobó que el sistema embebido almacena la configuración del proceso con el fin de que este continúe ejecutando la tarea que fue programada incluso sin estar anclada a la computadora.

Se implementó el módulo de entrenamiento con conexiones fáciles de realizar y se utilizó el software LabVIEW para visualizar los datos adquiridos en tiempo real.

Se encontró con la necesidad de utilizar las herramientas que están a disposición en el software LabVIEW con el fin de estabilizar las señales obtenida de los sensores.

Se realizaron pruebas de funcionamiento para constatar la adquisición de datos de cada una de las variables, en el caso del traductor de presión fue sometido a diferentes rangos de presión.

Se varió la frecuencia del tren de pulsos para obtener diferentes de velocidades, el estado de posición y el sentido de giro de motor.

4.2 Recomendaciones

Conocer el funcionamiento de todos los elementos que se ha utilizado para el desarrollo de este trabajo previo a la manipulación del módulo.

Respetar los rangos nominales de funcionamiento de cada uno de los elementos de módulo como es el caso del traductor de presión que soporta hasta un máximo de 10 Bars de presión, así como también los diferentes rangos de voltaje del resto de componentes.

Utilizar filtros para estabilizar la señal obtenida de los sensores de esta forma eliminar señales no deseadas producidas por ruidos exteriores, vibraciones, etc.

Dar continuidad a este tipo de trabajos ya que las prestaciones que ofrece el software LabVIEW junto con las tarjetas de adquisición de datos tienen un sin número de aplicaciones en cuanto a la automatización de procesos industriales.

BIBLIOGRAFÍA

ÑECO GARCÍA, Ramón Pedro. *Apuntes de sistemas de control*. ECU, 2004. 9788484543053. pp. 4-5.

LAJARA VIZCAÍNO, José Rafael. *LabVIEW*. l ª ed. Marcombo, 2007. pág. 385. 9788426714268.

HERNÁNDEZ CEVALLOS, María Isabel & LEDESMA MARCALLA, Denis Alejandro. Desarrollo de un sistema SCADA para la medición de voltajes con sistemas embebidos para el laboratorio de mecatrónica de la facultad de mecánica. (tesis) (ingenieria). Escuela Superior Politécnica de Chimborazo, Mecánica,Ingeniería de Mantenimiento. Riobamba, Ecuador. 2010. pp. 18-46. [Consulta: 20 de Diciembre del 2017]. Disponible en: http://dspace.espoch.edu.ec/handle/123456789/1137

LLINARES GALIANA, Antonio Nadal. *Sistemas embebidos*. [En línea] 2015. [Consulta: 28 de Diciembre de 2017.] Disponible en: https://es.scribd.com/document/108649190/A07-Sistemas-Embebidos

MARÍN, José. *Field programmable gate array (FPGA)*. [En línea] 2016. [Consulta: 28 de Diciembre de 2017.] Disponible en: http://bibing.us.es/proyectos/abreproy/11375/fichero/MEMORIA%252FFPGAs.pdf

MASTER, Degree. *Industrial systems engineering*. [En línea] 2011. [Consulta: 29 de Diciembre de 2017.] Disponible en: http://www.ieec.uned.es/investigacion/Dipseil/PAC/archivos/Informacion_de_referencia_ISE5_3_1.pdf

MIÑARRO ÚBEDA, BENITO. *Apuntes de: Sistemas embebidos*. [En línea] 2009. [Consulta: 28 de Diciembre de 2017.] Disponible en: https://www.google.com.ec/url?sa=t&rct=j&q=&esrc=s&source=web&cd=2&cad=rja&uact=8&ved=0ahUKEwjy9_Do-cbYAhXIQt8KHZTCDRQQFggwMAE&url=http%3A%2F%2Focw.um.es%2Fingenierias%2Fsistemas-embebidos%2Fmaterial-de-clase-1%2Fssee-t03.pdf&usg=AOvVaw0xjLNpjkJjnxzHVaGlt

PEDRE, SOL. 2012. *Sistemas embebidos*. [En línea] 2012. [Consulta: 29 de Diciembre de 2017.] Disponible en: http://bug.dc.uba.ar/charladeborrachos/presentaciones/charla_2012-10-12.pdf

DATASHEET. *Stepper Motor NEMA 17*. [En línea] [Consulta: 13 de Enero de 2018.] Disponible en: http://www.pbclinear.com/Download/DataSheet/Stepper-Motor-Support-Document.pdf

ENEKA. *Controladores Motores.* [En línea] 2014. [Consulta: 7 de Enero de 2018.] Disponible en: http://www.eneka.com.uy/robotica/motores/controladores-motor/7054-driver-para-motor-paso-a-paso-a4988-detail.html

NATIONAL INSTRUMENTS. *Fundamentos y tecnología de la adquisición de datos.* [En línea]. 2011. [Consulta: 23 de Diciembre de 2017.] Disponible en: ftp://ftp.ni.com/pub/branches/spain/eventos/multimedia/fundamentos_tecnologia_adquisicion_datos.pdf

NATIONAL INSTRUMENTS. *Desarrollo de Sistemas de Adquisición de Datos y Control de Tiempo real con Tecnologías Estándar.* [En línea]. 2007. [Consulta: 9 de Enero de 2018.] Disponible en: http://www.ni.com/white-paper/5913/es/#top

NATIONAL INSTRUMENTS. *Desarrollo de Sistemas con NI myRIO y CompactRIO.* [En línea]. 2016. [Consulta: 9 de Enero de 2018.] Disponible en: ftp://ftp.ni.com/pub/branches/latam/2016/educator_days/Colombia/Presentations/202-HO_RIO_Manual.pdf

NATIONAL INSTRUMENTS. *User guide and specifications NI myRIO-1900.* [En línea]. 2017. [Consulta: 9 de Enero de 2018.] Disponible en: http://www.ni.com/pdf/manuals/376047c.pdf

NATIONAL INSTRUMENTS. *Introducción a la Tecnología FPGA: Los Cinco Beneficios Principales.* [En línea]. 2018. [Consulta: 8 de Enero de 2018.] Disponible en: http://www.ni.com/white-paper/6984/es/

NATIONAL INSTRUMENTS. *Conversión de señal de encoder de cuadratura en señal de pulsos y derecha- izquierda.* [En línea]. 1879. [Consulta: 10 de Enero de 2018.] Disponible en: http://datalights.com.ec/site2/images/sensores/conversion_encoder_cuadratura.pdf

NATIONAL INSTRUMENTS. *Soporte de Software para NI myRIO.* [En línea]. 2013. [Consulta : 10 de Enero de 2018.] Disponible en: http://www.ni.com/product-documentation/14603/es/