

UNIVERSIDAD TÉCNICA FEDERICO SANTA MARÍA
SEDE VIÑA DEL MAR – JOSÉ MIGUEL CARRERA

CONTROL Y MONITOREO DE UN SISTEMA DE NIVEL CON ARDUINO A TRAVÉS DE LABVIEW

Trabajo de Titulación para optar al Título
Profesional de Ingeniero de Ejecución en
CONTROL E INSTRUMENTACIÓN INDUSTRIAL

Alumno:

Sr. Victor Alexanders Cáceres Pinto

Profesor Guía:

Mag. Guelis Montenegro Zamora

RESUMEN

KEYWORDS: Control ON-OFF, esquema de control de rango partido, LabVIEW, Factory I/O y Arduino.

El siguiente proyecto de título tiene por objetivo realizar el control ON-OFF con histéresis en un lazo cerrado de un sistema de control de nivel en Factory I/O por medio de Arduino UNO, logrando unir la parte de hardware con la parte de software, es decir, con las entradas analógicas físicas de Arduino UNO se podrá controlar un proceso simulado en Factory I/O. Para realizar el control de nivel se emplearán 2 actuadores, por lo que se hará uso de un esquema de control avanzado como el control de rango partido o “Split Range”. Este proyecto de título, además de realizar el control ON-OFF con histéresis en un lazo cerrado de un sistema de control de nivel, realizará el monitoreo de la señal de nivel del estanque, de la señal de control y de las variables manipuladas. Para ello, se hará uso del software de ingeniería “LabVIEW”, siendo éste de vital importancia para establecer la comunicación entre Arduino UNO y Factory I/O.

ÍNDICE

INTRODUCCIÓN	1
CAPÍTULO 1: ANTECEDENTES GENERALES.....	2
1. ANTECEDENTES GENERALES	3
1.1. PLANTEAMIENTO DEL PROBLEMA.....	3
1.2. SOLUCIÓN DEL PROBLEMA.....	3
1.3. LABVIEW	4
1.3.1. Programar en LabVIEW	5
1.4. ARDUINO.....	8
1.4.1. Hardware	9
1.4.2. Software	10
1.4.3. Arduino UNO	14
1.5. FACTORY I/O.....	18
1.6. OBJETIVOS DEL PROYECTO.....	21
1.6.1. Objetivo General	21
1.6.2. Objetivos Específicos	21
CAPÍTULO 2: DESARROLLO DE LA SOLUCIÓN.....	3
2. DESARROLLO DE LA SOLUCIÓN.....	23
2.1. SISTEMA DE CONTROL	23
2.1.1. Lazo Abierto.....	23
2.1.2. Lazo Cerrado.....	24
2.1.3. Controlador	27
2.1.4. Actuador	33
2.1.5. Planta.....	36
2.1.6. Sensor	41
2.2. COMUNICACIÓN	42
2.2.1. Comunicación en Factory I/O	42

2.2.2. Comunicación en LabVIEW 2020.....	54
2.3. PROGRAMACIÓN	62
2.3.1. Diagrama de bloques.....	64
2.3.2. Panel Frontal.....	74
CAPÍTULO 3: RESULTADOS OBTENIDOS	24
3. RESULTADOS OBTENIDOS	83
3.1. SISTEMA A LAZO CERRADO	83
3.1.1. Ejecutar el control	83
3.1.2. Prueba 1: Control ON-OFF sin histéresis	85
3.1.3. Prueba 2: Control ON-OFF con un valor de histéresis elevado	87
3.1.4. Prueba 3: Control ON-OFF con un valor de histéresis óptimo	89
3.2. IDENTIFICACIÓN DE SISTEMAS	90
3.2.1. Punto de operación	96
3.2.2. Función de transferencia.....	97
CONCLUSIÓN	109
BIBLIOGRAFÍA.....	110

ÍNDICE DE FIGURAS

Figura 1 - 1. Logotipo de LabVIEW	5
Figura 1 - 2. Ejemplo de un panel frontal.....	6
Figura 1 - 3. Ejemplo de un diagrama de bloques.....	7
Figura 1 - 4. Logotipo de Arduino.....	9
Figura 1 - 5. Descargando e instalando el toolkit “LINX”	11
Figura 1 - 6. Toolkit LINX.....	12
Figura 1 - 7. Paso 1 - Tipo de dispositivo	13
Figura 1 - 8. Paso 2 - Puerto COM	13
Figura 1 - 9. Paso 3 - Tipo de carga.....	14
Figura 1 - 10. Arduino UNO	15
Figura 1 - 11. Planta de nivel en Factory I/ O	19
Figura 2 - 1. Diagrama de bloques - Sistema de control de lazo cerrado	25
Figura 2 - 2. Diagrama de bloques - Sistema de control de rango partido de la planta de nivel...	26
Figura 2 - 3. Comportamiento del control de rango partido o Split Range.	27
Figura 2 - 4. Histéresis en control ON-OFF	29
Figura 2 - 5. Conexión potenciómetro - Setpoint del sistema	30
Figura 2 - 6. Conexión potenciómetro - Histéresis del sistema	30
Figura 2 - 7. Cálculo resistencia – LED verde.....	31
Figura 2 - 8. Cálculo resistencia – LED rojo	31
Figura 2 - 9. Circuito electrónico – Arduino UNO.....	32
Figura 2 - 10. Electroválvula de carga y descarga del líquido	34
Figura 2 - 11. Evolución de las electroválvulas en el tiempo	35
Figura 2 - 12. Planta de nivel a controlar	37
Figura 2 - 13. P&ID del sistema de nivel.....	39
Figura 2 - 14. Driver del sistema o escena de nivel.....	44
Figura 2 - 15. Ajustes de parámetros - Configuración en driver del sistema.....	50

Figura 2 - 16. Servidor de la escena de nivel.....	53
Figura 2 - 17. Paso 1 – Crear proyecto	54
Figura 2 - 18. Paso 2 – Crear un proyecto en blanco	55
Figura 2 - 19. Paso 3 - Ventana exploradora del proyecto.....	55
Figura 2 - 20. Paso 4 - Opción I/O Server	56
Figura 2 - 21. Paso 5 - Crear Servidor Modbus.....	56
Figura 2 - 22. Paso 6 – Configuración del Servidor Modbus	57
Figura 2 - 23. Librería de variables	58
Figura 2 - 24. Rangos de registros	59
Figura 2 - 25. Agregar rango de variable	60
Figura 2 - 26. Editor de múltiples variables.....	61
Figura 2 - 27. Elementos de datos o variables en Librería de Modbus	62
Figura 2 - 28. Nuevo VI	63
Figura 2 - 29. VI creado.....	63
Figura 2 - 30. Diagrama de bloques del proyecto – Parte 1.....	64
Figura 2 - 31. Diagrama de bloques del proyecto – Parte 2.....	65
Figura 2 - 32. Bloque “Open Serial”	66
Figura 2 - 33. Bloque “Analog Read”	66
Figura 2 - 34. Bloque “Digital Write”	67
Figura 2 - 35. Bloque “Close”	67
Figura 2 - 36. Shared Variable	68
Figura 2 - 37. Panel Frontal de LabVIEW – Primera parte.....	74
Figura 2 - 38. Panel Frontal de LabVIEW – Segunda parte.....	75
Figura 3 - 1. Puerto COM de Arduino UNO.	84
Figura 3 - 2. Botón Run para iniciar programación en Lenguaje G.	85
Figura 3 - 3. Botón iniciar/detener proceso automático.	85
Figura 3 - 4. Control ON-OFF sin histéresis.	86
Figura 3 - 5. Control ON-OFF con histéresis elevado.	88
Figura 3 - 6. Control ON-OFF con histéresis óptimo.	89

Figura 3 - 7. Sistema de control a lazo abierto.....	91
Figura 3 - 8. Sistema no lineal.	93
Figura 3 - 9. Punto de operación del sistema.....	96
Figura 3 - 10. Variables creadas en MatLab.	101
Figura 3 - 11. Datos agregados en las variables.	102
Figura 3 - 12. Cuadro de dialogo llamado “System Identification”.....	103
Figura 3 - 13. Cuadro de dialogo llamado Import Data.....	104
Figura 3 - 14. Seleccionando el rango.	105
Figura 3 - 15. Generando y validando el modelo.	106
Figura 3 - 16. Valores de ganancia, constante de tiempo o tau y retardo.....	107
Figura 3 - 17. Función de transferencia.....	108
Figura 3 - 18. Porcentaje de asertividad.	108

ÍNDICE DE TABLAS

Tabla 2 - 1. Bancos de datos o bloques de memoria.	43
Tabla 2 - 2. Colores en los tipos de datos.....	49
Tabla 3 - 1. Nivel v/s Actuación – Parte 1.....	91
Tabla 3 - 2. Nivel v/s Actuación – Parte 2.....	92
Tabla 3 - 3. Nivel v/s Actuación – Parte 3.....	93
Tabla 3 - 4. Principio de homogeneidad.....	94
Tabla 3 - 5. Principio de superposición.	95
Tabla 3 - 6. Respuesta ante un delta escalón pequeño – Parte 1.....	98
Tabla 3 - 7. Respuesta ante un delta escalón pequeño – Parte 2.....	99
Tabla 3 - 8. Datos lineales y no lineales.	100

SIGLAS Y SIMBOLOGÍA

A. SIGLAS:

AC	: Alternating Current (Corriente Alterna).
ADC	: Analog to Digital Converter (Conversor Analógico a Digital).
AREF	: Analog Reference (Referencia Analógica).
COM3	: Comunicaciones 3.
CPU	: Central Processing Unit (Unidad Central de Procesamiento).
DC	: Direct Current (Corriente Continua).
EEPROM	: Electrically Erasable Programmable Read Only Memory (Memoria de Solo Lectura Programable y Borrable Eléctricamente).
HMI	: Human Machine Interface (Interfaz Hombre Máquina).
I2C	: Inter Integrated Circuit (Circuito Inter-Integrado).
ICSP	: In Circuit Serial Programming (Programación Serie en Circuito).
IOREF	: Input/ Output Reference (Referencia de Entrada/ Salida).
IDE	: Integrated Development Environment (Entorno de Desarrollo Integrado).
IP	: Internet Protocol (Protocolo de Internet).
ISA	: International Society of Automation (Sociedad Internacional de Automatización).
LabVIEW	: Laboratory Virtual Instrumentation Engineering Workbench (Laboratorio de Ingeniería de Instrumentos Virtuales Workbench).
LTI	: Linear Time Invariant (Lineal e Invariante en el Tiempo).
LPT	: Line Print Terminal (Terminal de Impresión de Línea).
LED	: Light Emitting Diode (Diodo Emisor de Luz).
MATLAB	: Matrix Laboratory (Laboratorio de Matrices).
NI	: National Instruments (Instrumentos Nacionales).
Vis	: Virtual Instruments (Instrumentos Virtuales).
OSI	: Open Systems Interconnections (Interconexiones de Sistemas Abiertos).
PC	: Personal Computer (Computador Personal).

PCB	: Printed Circuit Board (Placa de Circuito Impreso).
PLC	: Programmable Logic Controller (Controlador Lógico Programable).
PIC	: Programmable Integrated Circuit (Circuito Integrado Programable).
P&ID	: Piping and Instrumentation Diagram (Diagrama de tuberías e instrumentación).
PWM	: Pulse Width Modulation (Modulación por Ancho de Pulsos).
RTU	: Remote Terminal Unit (Unidad Terminal Remota).
SCADA	: Supervisory Control and Data Acquisition (Supervisión, Control y Adquisición de Datos).
SPI	: Serial Peripheral Interface (Interfaz de Periféricos Serie).
SRAM	: Stactic Random Access Memory (Memoria Estática de Acceso Aleatorio).
TCP	: Transmission Control Protocol (Protocolo de Control de Transmisión).
UART	: Universal Asynchronous Receiver/ Transmitter (Transmisor/ Receptor Asíncrono Universal).
USB	: Universal Serial Bus (Bus Universal en Serie).
VDC	: Voltage of Direct Current (Voltaje de Corriente Continua).
WWW	: World Wide Web (Red Mundial).

B. SIMBOLOGÍA:

A	: Ampere. Unidad de intensidad de corriente eléctrica.
b	: Bit. Unidad mínima de información empleada en informática.
B	: Byte. Unidad de información formada por una secuencia de bits adyacentes que equivale a un conjunto ordenado de 8 bits.
Bd	: Baudios. Unidad de medida que representa el número de símbolos por segundos en un medio de transmisión digital.
cm	: Centímetros. Medida de longitud que corresponde a la centésima parte de un metro.
Hz	: Hertz. Unidad de frecuencia.
kb	: Kilobit. Unidad de medida que permite calcular la capacidad de almacenamiento de información o procesamiento de datos, la cual equivale a 1000 bit.
kB	: Kilobyte. Unidad de almacenamiento de información digital que es igual a 1000 bytes.

mA	: Miliamper. Medida de intensidad de corriente eléctrica que corresponde a la milésima parte de un Amper.
ms	: Milisegundos. Medida de tiempo que corresponde a la milésima parte de un segundo.
MB	: Megabyte. Unidad de información que es igual a 1000 kB.
MHz	: Megahercios. Unidad de medida de la frecuencia que corresponde a 1000 kHz.
Ohm	: Ohm. Unidad derivada de la resistencia eléctrica y su símbolo es Ω .
s	: Segundos. Unidad de tiempo.
V	: Volts. Unidad de la tensión eléctrica, potencial eléctrico y fuerza electromotriz.

INTRODUCCIÓN

Los procesos industriales actuales requieren que los parámetros de nivel, sean controlados y medidos para una producción continua evitando derrames de líquidos, almacenados en los estanques. La medición de estos parámetros en los líquidos dentro de un estanque parece sencilla, pero puede ser un problema, sobre todo cuando el líquido contenido en el estanque es un fluido inflamable, corrosivo, reactivo o radioactivo, donde conviene tener el recipiente o estanque sellado para evitar el contacto físico y el contacto a través de las vías de exposición (vía de ingestión, inhalación y absorción). Por lo tanto, son procesos industriales, los cuales presentan un mayor peligro o riesgo para el personal que opera en la planta.

Dicho esto, el siguiente proyecto presenta el desarrollo de un control ON-OFF con histéresis y el monitoreo de una planta de procesos ubicada en el simulador de fábrica 3D Factory I/O, a través del software de ingeniería LabVIEW, siendo la plataforma de desarrollo física Arduino UNO, la encargada de controlar dicho sistema. Por lo tanto, de manera remota el usuario podrá ingresar los valores esperados en el sistema o proceso para lograr una producción continua y estable.

El proyecto se encuentra estructurado en 3 capítulos, donde el primero da a conocer el problema, la solución y los objetivos a lograr. Mientras tanto, el capítulo 2, presenta la solución planteada en el capítulo 1, es decir, expone a gran detalle la programación llevada a cabo en LabVIEW para lograr los objetivos. Finalmente, el capítulo 3, expone la evaluación económica del proyecto.

CAPÍTULO 1: ANTECEDENTES GENERALES

1. ANTECEDENTES GENERALES

Este capítulo explica el contexto general en el cual está inmerso el proyecto, es decir, se plantea el problema, se da a conocer la solución y se definen los objetivos a alcanzar en el proyecto.

1.1. PLANTEAMIENTO DEL PROBLEMA

El problema de este proyecto se basa en controlar en un lazo cerrado un sistema de control de nivel en Factory I/O por medio de Arduino UNO. El principal desarrollo de la problemática es unir la parte de hardware de Arduino con la parte de software de Factory I/O, es decir, con las entradas analógicas físicas del Arduino UNO se podrá controlar y con las salidas digitales físicas se podrá monitorear un proceso simulado en Factory I/O.

1.2. SOLUCIÓN DEL PROBLEMA

La solución del problema planteado anteriormente, consiste en llevar a cabo la programación necesaria en Arduino UNO para lograr controlar y monitorear una escena entregada por el simulador de fábrica 3D "Factory I/O", llamada sistema de control de nivel de agua. La herramienta necesaria para controlar dicho sistema será la plataforma de desarrollo físico "Arduino UNO" y para monitorear el nivel del estanque junto con la señal de control se tiene el software de ingeniería "LabVIEW". Este último, también resulta indispensable para generar la comunicación entre Arduino y Factory I/O.

La estrategia de control a utilizar será del tipo “ON-OFF con histéresis”, también llamado todo-nada o abierto-cerrado. Este tipo de control trata de mantener la variable en un punto de consigna, valor deseado o setpoint, si la variable se encuentra por debajo de este, la salida del controlador está encendida (ON) y cuando lo sobrepasa la salida es apagada (OFF), no intervienen más lógica en el proceso más que activar o desactivar el actuador. La histéresis creará un retardo en el cambio de la señal o un error para disminuir la cantidad de conmutaciones creando un valor de umbral superior e inferior. Esto quiere decir, que las salidas del controlador no pasarán del estado ON a OFF o viceversa cuando la variable controlada pase por el punto de consigna, sino que lo hará un poco más arriba donde se encuentra el valor de umbral superior o un poco más abajo donde está el valor de umbral inferior.

Para realizar el control de nivel se emplearán 2 actuadores, por lo que se hará uso de un esquema de control avanzado como el control de rango partido o “Split Range”. Este tipo de esquema de control avanzado se explicará a detalle en el capítulo 2.

Por lo tanto, se programa en Arduino UNO para ejecutar las señales de control ON-OFF que pasarán por LabVIEW donde también se programa para monitorearla interfaz HMI, es decir, se grafica la señal de respuesta correspondiente al nivel y además la señal de control. Luego de esto, las señales de control llegan a Factory I/O para lograr que el nivel de agua en el sistema de control ascienda o descienda.

1.3. LABVIEW

LabVIEW es un programa de ingeniería de sistemas creado por NI que proporciona acceso rápido a información de datos y hardware para evaluar, medir y controlar aplicaciones. Este entorno y plataforma de desarrollo permite construir sistemas, utilizando un lenguaje de programación de gráficos visuales como el lenguaje G (G simboliza que es lenguaje gráfico), destinado a sistemas hardware y programas de pruebas, control y diseño, simulado o real y embebido. El logotipo de LabVIEW se puede conocer a través de la figura 1-1.



Fuente: www.ni.com

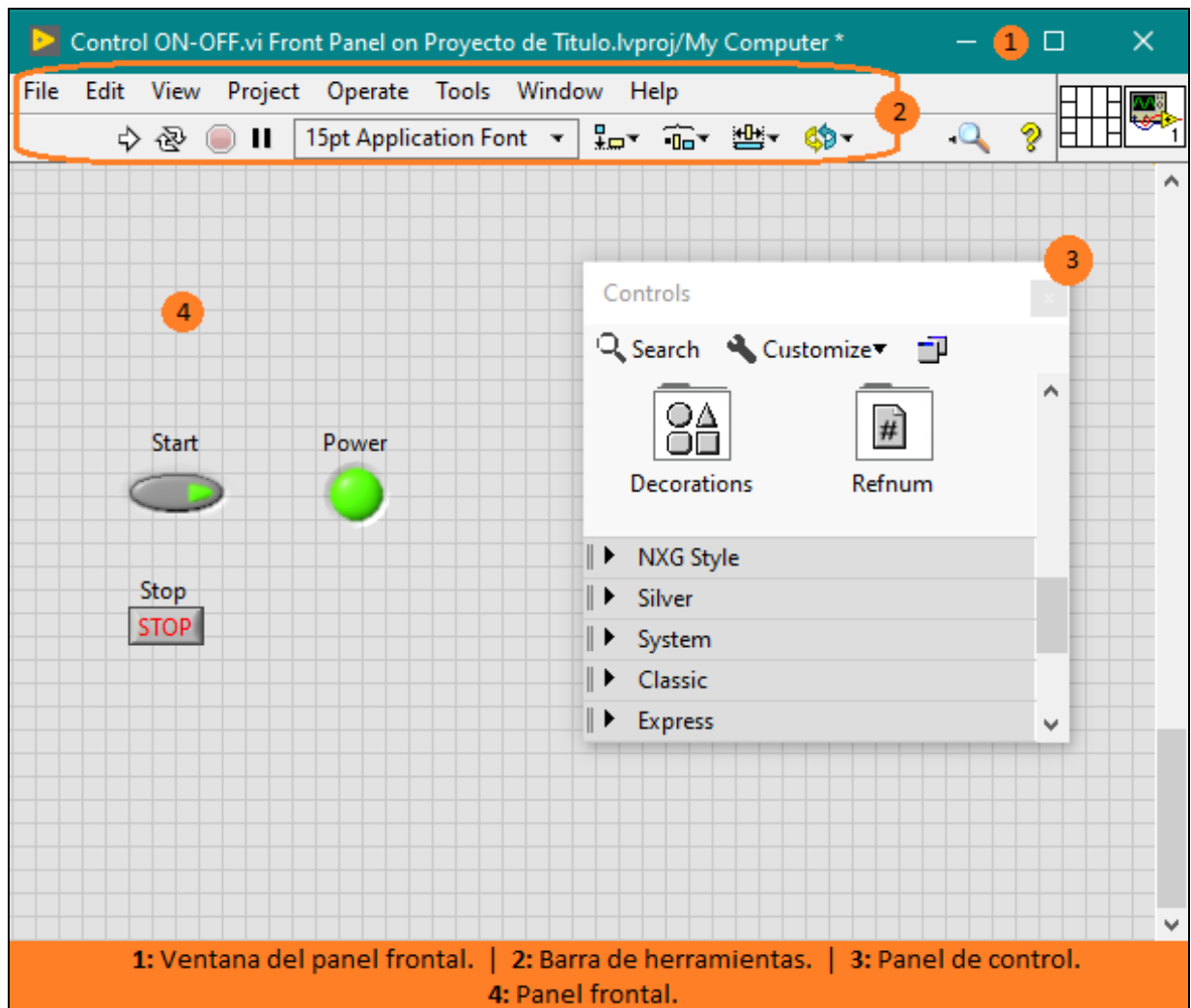
Figura 1 - 1. Logotipo de LabVIEW.

1.3.1. Programar en LabVIEW

En contraste con los lenguajes ampliamente utilizados, LabVIEW brinda a los usuarios muchas soluciones intuitivas y poderosas, gracias a su gran variedad de herramientas que permiten desarrollar aplicaciones de medición, control y automatización, las cuales se pueden programar sin escribir una sola línea de texto o código.

Los programas desarrollados en LabVIEW se llaman VIs, dado que su apariencia y funcionamiento suelen imitar instrumentos físicos, como osciloscopios y multímetros, es decir, proviene del control de instrumentos, aunque actualmente se ha expandido a no solo controlar todo tipo electrónica, sino que además a la programación embebida, comunicación, matemáticas, etc. Incluye una variedad de herramientas para adquirir, analizar y almacenar datos, así como herramientas para ayudar a solucionar problemas del código que se escribe.

Es una herramienta de programación gráfica, esto quiere decir, que el programa está dibujado, no escrito para facilitar su comprensión. Para el usuario la creación de proyectos resulta simple cuando se tiene prediseñado varios bloques, por lo cual en vez de ocupar su tiempo en la programación del dispositivo/ bloque, puede invertir más tiempo y dedicación en la interfaz gráfica y en la interacción con el usuario final. Los VIs constan de dos partes diferenciadas (Panel frontal y diagrama de bloques), las cuales se enseñan en las figuras 1-2 y 1-3.



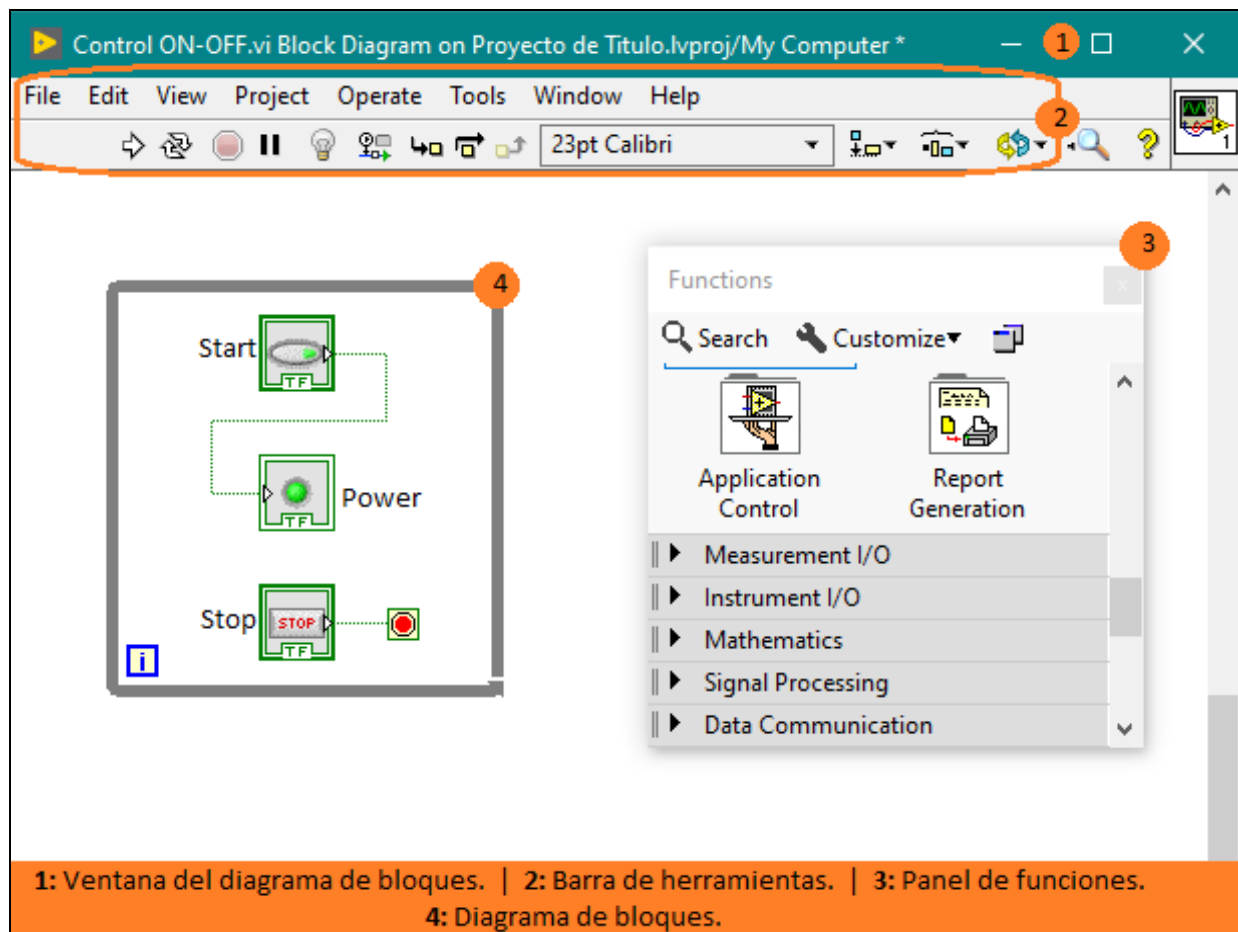
Fuente: Elaboración propia, realizada en LabVIEW 2020 y etiquetada en Paint.

Figura 1 - 2. Ejemplo de un panel frontal.

El panel frontal es la ventana que el usuario puede ver e interactuar de alguna u otra manera mientras el software se está ejecutando. Este panel de control permite a los usuarios ingresar datos, ajustar parámetros, monitorear estados, controlar variables y más. Por lo tanto, en esta ventana se podrá ver la HMI del sistema de control de nivel de agua programada en Factory I/O.

El panel frontal, tiene una variedad de controles e indicadores, donde cada elemento está asignado a un terminal en el diagrama de bloques. Esto significa que los usuarios pueden diseñar proyectos en el panel frontal, utilizando controles e indicadores, en que las entradas y salidas se comunicarán con el terminal VIs.

Para ingresar a la paleta de control de la ventana del panel frontal hay que apretar en “View>>Control Palette” en la barra de herramientas o simplemente hacer clic derecho en un espacio en blanco de la ventana. Este panel permite mostrar algunas (como se muestra en la figura 1-2) o todas las categorías para finalmente dar con las expectativas.



Fuente: Elaboración propia, realizada en LabVIEW 2020 y etiquetada en Paint.

Figura 1 - 3. Ejemplo de un diagrama de bloques.

Para empezar a enlazar el conjunto de bloques necesarios que dan funcionamiento de alguna u otra manera al proyecto de título, se debe hacer uso de la ventana llamada diagrama de bloques mostrada en la figura 1-3. En esta ventana, es donde se define su funcionalidad, van los íconos que realizan funciones específicas y se interconectan. Esto quiere decir que se pueden ver los datos de todos los controles e indicadores y como van fluyendo entre ellos cuando se está ejecutando un proceso VI.

Para crear el diagrama de bloque se debe hacer uso de la ventana denominada “Paleta de Funciones”, la cual contiene los VIs, funciones y constantes necesarias. El ingreso o acceso a esta paleta de funciones del diagrama de bloques es seleccionando “View>>Functions Palette”. Ésta paleta está dividida en numerosas categorías, donde se puede mostrar una categoría o esconderla (como se muestra en la figura 1-3) para finalmente dar con el bloque buscado.

1.4. ARDUINO

Arduino es una plataforma de desarrollo basada en una placa electrónica de hardware libre, la cual tiene un microcontrolador reprogramable y una sucesión de pines hembras, los cuales permiten conectar el microcontrolador a una variedad de sensores y actuadores de una forma muy fácil, primordialmente utilizando cables Dupont (cables de 15 cm con terminales macho-macho).

Arduino es una placa de circuito impreso o PCB, la cual está implementada con un determinado diseño de circuitería interna, permitiendo a los usuarios iniciar el desarrollo de las diversas aplicaciones electrónicas que requieran, sin tener que preocuparse por las conexiones eléctricas necesarias para que funcione el microcontrolador.

Los tipos de Arduino conocidos en el mercado son: Uno, Leonardo, Mega 2560, Mega ADK, Nano, etc. Hay PCB básicos con escasas entradas y salidas, PCB de gama alta con un mayor número de entradas y salidas, PCB con conexión de red o con conexiones inalámbricas y PCB para ropa de tamaño más reducido. Se puede apreciar el logotipo de Arduino en la figura 1-4.



Fuente: www.arduino.cl

Figura 1 - 4. Logotipo de Arduino.

1.4.1. Hardware

Un Arduino o microcontrolador tiene puertos de entradas/ salidas y puertos de comunicación, permitiendo el acceso a estos puertos a través de pines.

- Pines de Entrada Analógica: Se utilizan para la lectura de sensores analógicos como sondas de temperatura, sondas de nivel, sondas de humedad, etc, mediante un conversor analógico/ digital.
- Pines de Salida Analógica o (PWM): La mayoría de los Arduino no cuentan con un conversor digital/analógico. Por lo tanto, para tener salidas analógicas se usa la técnica PWM, sin embargo, no todos los pines digitales soportan PWM.
- Pines de Entrada/ Salida Digital: Se puede configurar como entrada para leer sensores o como salida para escribir.
- Puertos de Comunicación: USB, Serie, UART, I2C y SPI.

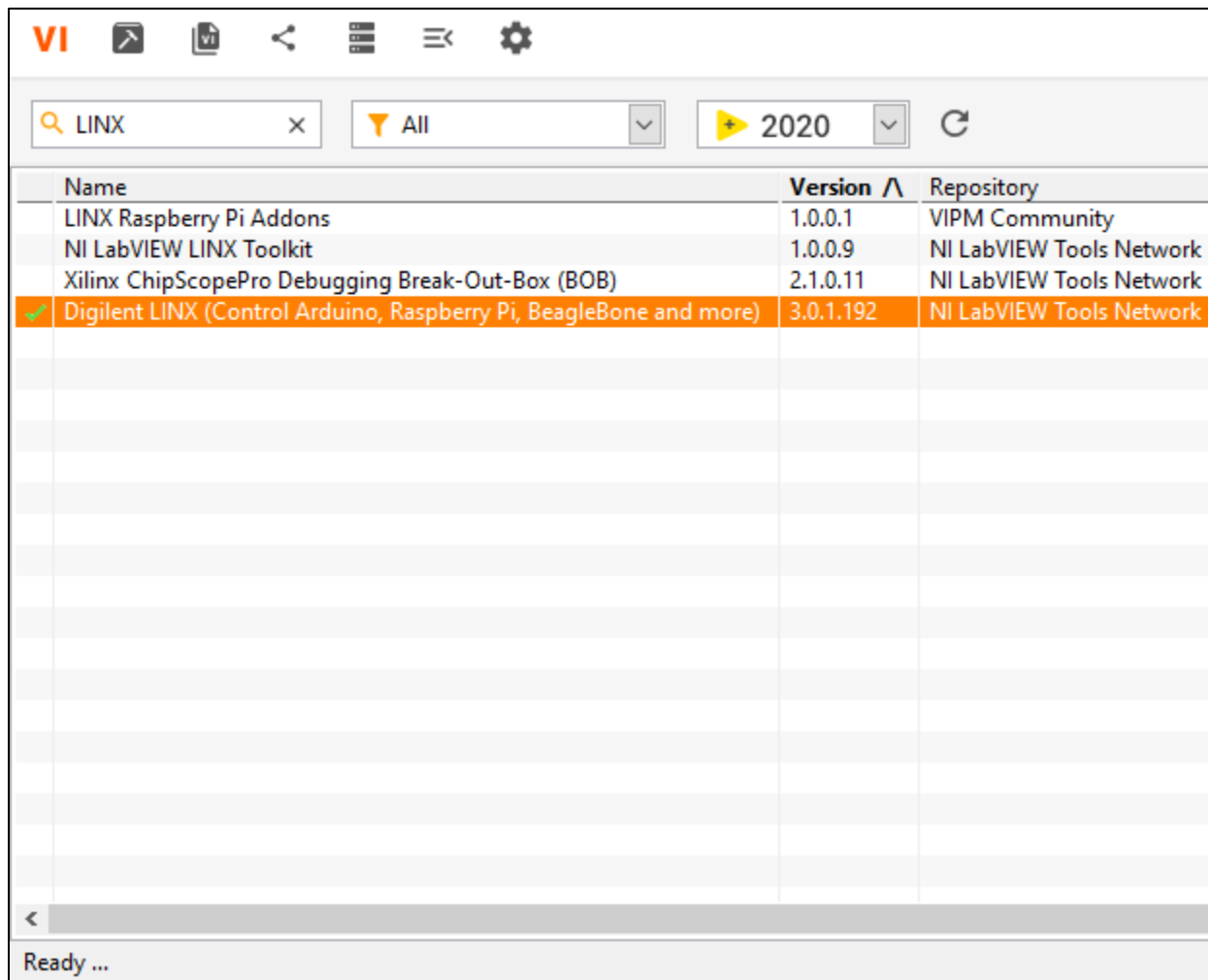
Al crear un programa o sketch, éste se almacena en la memoria de Arduino. Ahora bien, en paralelo, mientras se ejecuta el programa se generan datos, los que se guardan para lograr recuperarlos cuando se requieran. Por lo tanto, otro aspecto importante a nombrar son los 3 tipos de memoria que posee Arduino.

- Memoria EEPROM: Memoria no volátil encargada de retener datos luego de un reinicio o apagado. Existe un límite en la cantidad de veces que se puede leer/ escribir utilizando la memoria EEPROM, es decir, se debe tener en consideración a la hora de utilizarla. En la mayoría de los casos, esto se puede escribir en el software del microcontrolador como una constante del programa.
- Memoria SRAM: Arduino usa esta memoria para crear y manipular las variables en tiempo de ejecución. Éste es un recurso limitado y requiere ser monitoreado para garantizar que su uso no se gaste del todo.
- Memoria Flash: En esta memoria se guarda el sketch o programa ya compilado. Es una memoria no volátil, es decir, los datos del programa permanecen guardados incluso cuando la alimentación o el sistema se apaga. Los programas ejecutados desde esta memoria se pueden cambiar solo si fueron copiados en la SRAM.

1.4.2. Software

Para comenzar a programar en la placa Arduino, es necesario descargar un toolkit llamado “LINX” en LabVIEW, el cual permite comunicar LabVIEW con Arduino, mediante un firmware (sketch) que se debe cargar en la placa. Este toolkit es como un juego de herramientas que contiene un conjunto de elementos de control gráficos, que se utilizan para construir la interfaz gráfica. Por lo tanto, este toolkit realizará de alguna u otra manera la conversión de lenguaje G (LabVIEW) a lenguaje C++ (Arduino), sin necesidad de hacer uso del IDE (programa o aplicación que transfiere el contenido de las instrucciones a la placa de Arduino).

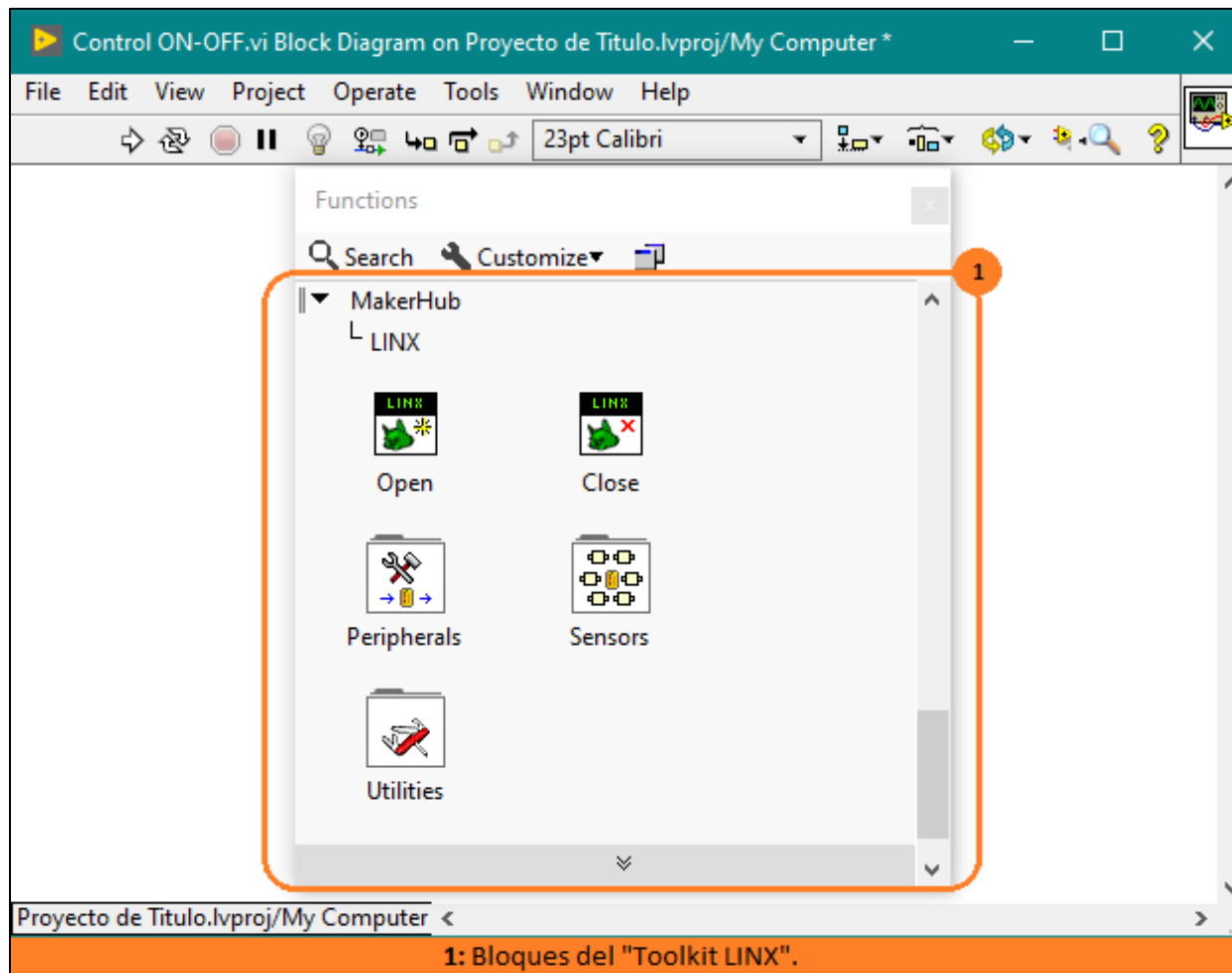
Para descargar este toolkit se debe abrir el VI Package Manager (aplicación de LabVIEW utilizada para descargar Add-ons, extensiones o plugins) y escribir en la barra de búsqueda “LINX”, para luego seleccionar y descargar el toolkit llamado “Digilent LINX (Control Arduino, Raspberry Pi, BeagleBone and more)”, que se muestra en la figura 1-5.



Fuente: Elaboración propia, realizada en VI Package Manager.

Figura 1 - 5. Descargando e instalando el toolkit “LINX”.

Una vez instalado este toolkit “LINX” en LabVIEW, es posible encontrarlo en el panel de funciones del diagrama de bloques, donde se debe presionar en MakerHub >>LINX. Aquí aparecen todos los bloques necesarios para establecer la comunicación con Arduino, es decir, se puede encontrar el bloque de inicio o apertura, el bloque de cierre, los bloques relacionados a periféricos de entrada y salida del Arduino, bloques de sensores, etc. Estos bloques se pueden observar en la figura 1-6.



Fuente: Elaboración propia, realizada en LabVIEW 2020 y etiquetada en Paint.

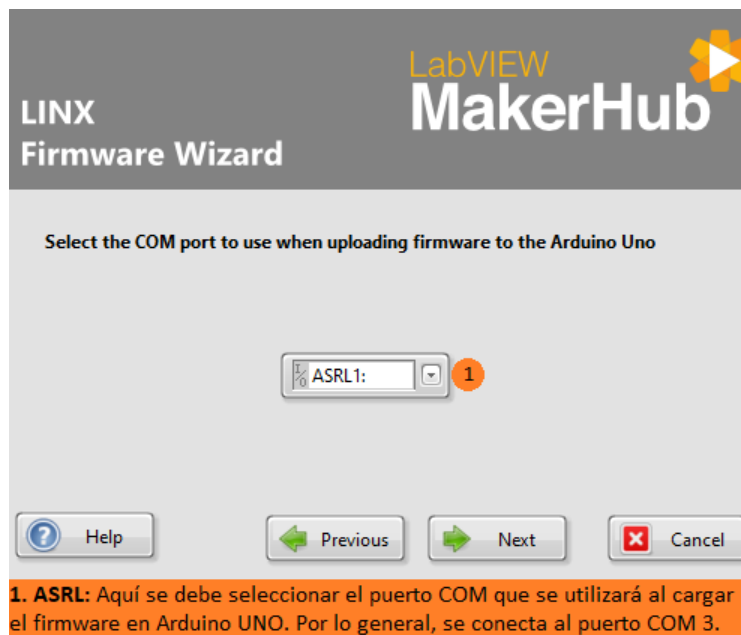
Figura 1 - 6. Toolkit LINX.

Una vez utilizados los bloques de LINX, es necesario cargar esta programación en lenguaje G de LabVIEW, en Arduino UNO. Para ello, es necesario dirigirse a la barra de herramientas del diagrama de bloques, presionar en Tool>>MarkeHub>>LINX>>LINX firmware Wizard, para abrir una ventana o un cuadro de diálogo llamado “LINX firmware Wizard”, donde se debe ir seleccionando según el tipo de dispositivo, el puerto COM que se utilizará para cargar el firmware y el tipo de carga. Estos pasos se muestran en las figuras 1-7, 1-8 y 1-9.



Fuente: Elaboración propia, realizada en LabVIEW 2020 y etiquetada en Paint.

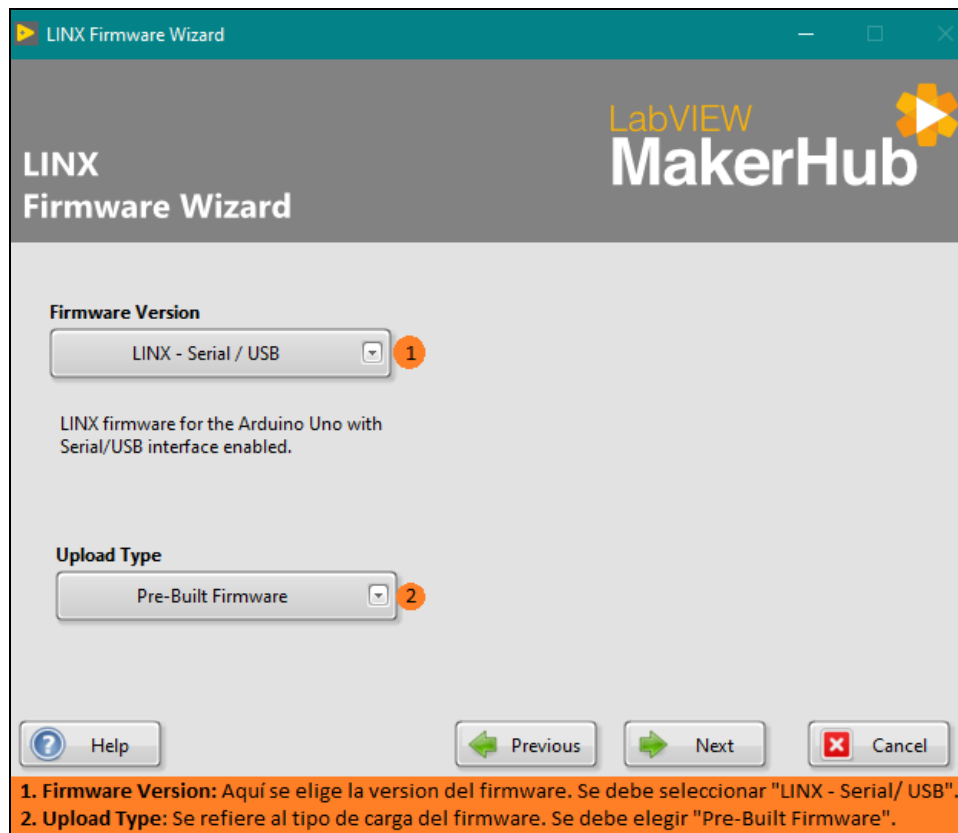
Figura 1 - 7. Paso 1 - Tipo de dispositivo.



Fuente: Elaboración propia, realizada en LabVIEW 2020 y etiquetada en Paint.

Figura 1 - 8. Paso 2 - Puerto COM.

La figura 1-9, mostrada a continuación, da a conocer mediante un cuadro de diálogo el tercer y último paso necesario para posteriormente cargar la programación en lenguaje G de LabVIEW en Arduino UNO.



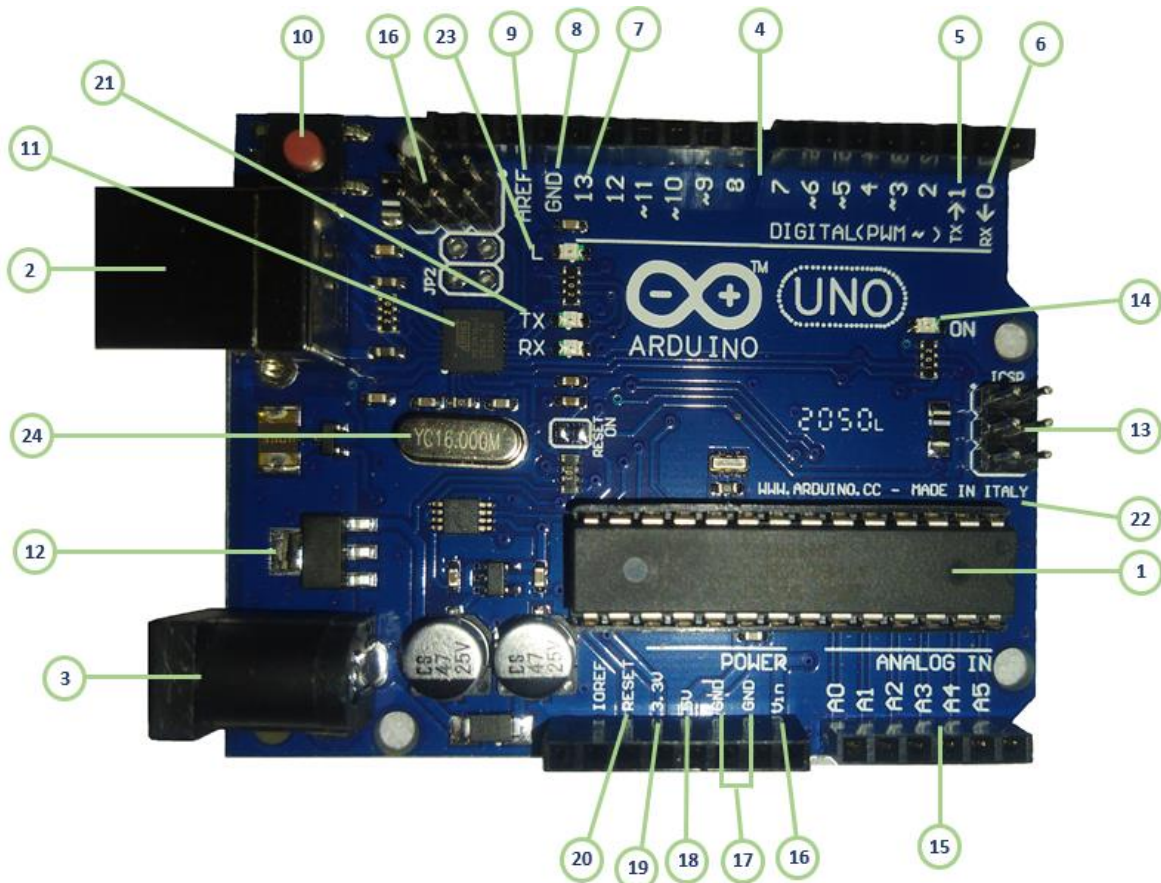
Fuente: Elaboración propia, realizada en LabVIEW 2020 y etiquetada en Paint.

Figura 1 - 9. Paso 3 - Tipo de carga.

1.4.3. Arduino UNO

Arduino UNO es la placa perfecta para comenzar con la programación y la electrónica. Es ampliamente utilizada, resultando ser la alternativa más robusta y con mayor proporción de documentación en toda la gama de Arduino.

Arduino UNO es una placa de desarrollo de código abierto, basada en el microcontrolador ATmega328P. Tiene 14 pines de entrada/salida digital (de los cuales 6 están disponible para PWM), 6 entradas analógicas, un cristal de 16MHz, una conexión USB, un conector jack de alimentación, terminales para conexión ICSP y un botón de reseteo. En su memoria flash cuenta con 32kB (0.5 kB son utilizados por el gestor de arranque), tiene 2kB de memoria SRAM y 1kB de memoria EEPROM. Además, tiene toda la electrónica necesaria para que el microcontrolador opere, simplemente hay que conectarlo a una fuente de alimentación (a través del puerto USB o por el jack de alimentación). Este Arduino UNO se puede apreciar en la figura 1-10.



Fuente: Elaboración propia, realizada en Microsoft Word.

Figura 1 - 10. Arduino UNO.

En primer lugar, se deben conocer las partes importantes que tiene la placa Arduino UNO y cómo se va a interactuar con ella. Es por eso, que en la figura 1-10 se puede ver la placa Arduino UNO con algunos componentes que reciben su respectivo “tag number” para explicar su función de manera clara y ordenada.

1. Microcontrolador ATmega328P: Es el corazón o procesador del Arduino.
2. Puerto USB: Se utiliza para la alimentación de la interfaz Arduino UNO, la carga de programas realizado en LabVIEW y para la comunicación con la placa. El regulador de tensión ubicado a su lado, convertirá la tensión que le llega a través de este puerto USB y hará que trabaje a 5V.
3. Conector o jack de alimentación: Sirve para alimentar el Arduino cuando no está conectado a un puerto USB. Este conector acepta voltajes de entrada de 7 a 12V, ya que, en serie a este se encuentra un regulador de voltaje en el que habrá una caída de tensión de 5V para que Arduino opere correctamente. Si Arduino se alimenta con una tensión inferior a 6V DC es muy probable que no encienda. Por el contrario, si se alimenta con una tensión superior a 12V DC, será fácil quemar o dañar el regulador en poco tiempo. Por lo tanto, una tensión óptima de trabajo será de 7V DC. Además, este conector de alimentación posee una protección contra inversión de polaridad.
4. Pines digitales: Son 14 pines de E/ S digital, que van desde el pin 0 al pin 13, de los cuales 6 de ellos (pin 3, 5, 6, 9, 10 y 11) proporcionan salidas PWM de 8 bits que toman valores desde 0 a 255. En estos pines irán conectados los leds correspondientes a electroválvula de llenado y electroválvula de descarga a modo de monitorear el estado de ellas.
5. Pin 1: Pin de transmisión (TX), el cual es utilizado para la comunicación serie UART. Por ejemplo, se puede conectar un módulo bluetooth u otro Arduino como esclavo.
6. Pin 0: Este pin es de recepción (RX), el cual al igual que el “pin 1” es utilizado para la comunicación serie UART.
7. Pin 13: Es un pin especial, que está conectado al led integrado ubicado cerca del “pin 13” llamado “L” en la placa.
8. GND: Pin que proporciona masa, tierra, negativo a los circuitos.

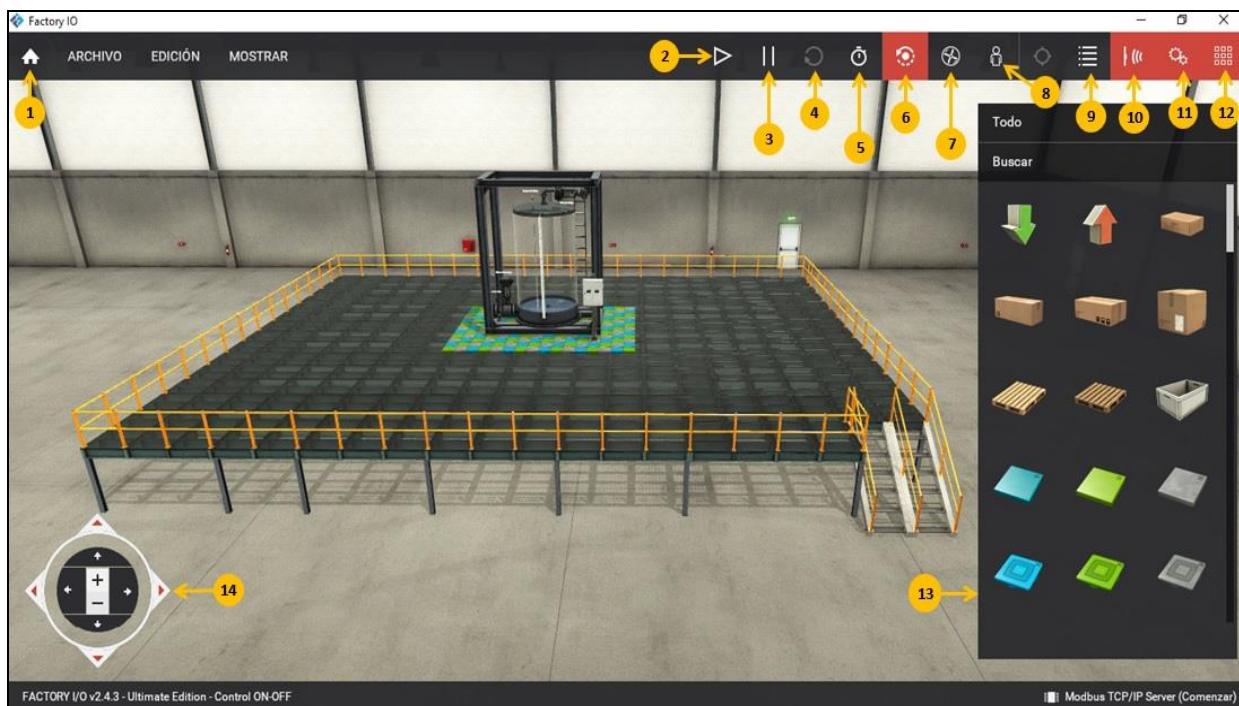
9. AREF: Es el ADC y permite conectar una tensión externa de referencia, la cual será comparada con la señal que se lea en los pines A0 al A5. El voltaje externo a aplicar debe ser solamente entre 0 y 5V.
10. Botón reset: Permite conectar temporalmente el pin 1 del microcontrolador ATmega328P a tierra para que reinicie cualquier código cargado en el Arduino.
11. Chip de comunicación serie: Este chip es necesario para lograr la comunicación entre Arduino UNO y el ordenador.
12. Regulador de tensión: Su función es convertir el voltaje de alimentación al voltaje que necesitan los elementos electrónicos, principalmente el microcontrolador. Para que la placa trabaje a 5V deberá recibir unos 6,5 a 7V, sin embargo, todo lo que esté por encima de este valor se desperdiciará, ya que, calentará la placa de Arduino. En ningún caso, se recomienda aplicar al regulador más de 12V y a los 20V se dañará.
13. ICSP: Se utiliza para grabar el bootloader en el microcontrolador o modificar el programa a través de este puerto sin necesidad de sacarlo del zócalo. El bootloader, es un pequeño programa que ha sido guardado previamente en el microcontrolador de la placa y permite cargar el código sin necesidad de hardware adicional. Por lo tanto, el bootloader ya viene grabado de fábrica en este microcontrolador.
14. LED de encendido: Indica que la placa está recibiendo alimentación.
15. Pines analógicos: Arduino cuenta con 6 pines de entrada analógicas con una resolución de 10 bits que van desde el pin A0 al pin A5, los cuales pueden tomar valores comprendidos entre 0 y 1023. En estos pines, irán conectados los potenciómetros encargados de fijar el setpoint y la histéresis del proceso.
16. Vin: Este pin se puede comportar como entrada, ya que permite alimentar al Arduino con una tensión de 6 a 12V que iría directamente al regulador de tensión. También, funciona como salida de voltaje cuando el Arduino está alimentado a través del jack de alimentación. En este caso, el voltaje presente en Vin será la resta entre el voltaje aplicado en el jack menos la caída de tensión en el diodo de protección de inversión de polaridad (alrededor de 0.7V). En este pin, no se recomienda conectar cargas mayores a 1A, ya que se puede dañar el diodo de protección.

17. GND: Proporciona masa, tierra, negativo a los circuitos.
18. 5V: Es útil cuando la placa de Arduino se alimenta por el conector USB o por el jack de alimentación, donde se tiene una tensión regulada de salida de 5VDC.
19. 3,3V: Este pin entrega una tensión de salida de 3,3VDC y una corriente máxima de 60mA.
20. Pin reset: Este pin restablece el microcontrolador ATmega328P. Es necesario llevar este pin a 0V o GND para lograr el reseteo. La diferencia entre botón reset (tag 10) y pin reset, es que el primero ya está conectado de manera interna, solo basta con presionar el botón reset para que exista el reinicio de Arduino UNO, en cambio en el segundo, se debe conectar de manera externa, es decir, se requiere de algún tipo de pulsador de por medio para que permita llevar esta salida o pin reset a tierra o GND.
21. LED TX y RX: Indican la comunicación entre la placa y el ordenador. Estos leds parpadearán rápidamente durante la carga de programas, así como durante la comunicación serie. Resultan útiles para la depuración (siendo la depuración la tarea de encontrar errores en el software).
22. Made: Es la casa del fabricante original. Aquí se puede comprobar fácilmente si la placa es original o se trata de una imitación.
23. LED "L": Sirve para realizar las primeras prácticas de encender y apagar el led, además de ser muy útil para la depuración.
24. Reloj / Cristal de 16 MHz oscilador: Se usa como reloj externo en el montaje del Arduino.

1.5. **FACTORY I/O**

Es un software de simulación 3D, creado por Real Games, el cual permite construir y controlar procesos industriales en tiempo real mediante la conexión de Factory I/O y equipos externos (PLCs, Arduino, etc). Se trata de una simulación totalmente interactiva con gráficos y sonidos de alta calidad, logrando de esta manera un entorno industrial real.

Las características principales de Factory I/O, son 21 escenarios inspirados en las apps industriales más frecuentes, usando una librería con más de 80 componentes industriales, permitiendo crear desde un escenario personalizado, hasta una fábrica virtual, la capacidad de crear estrategias para el diagnóstico de averías, ya sea por enchufe abierto o corto circuito, la seguridad y rendimiento del software, ya que, exhibe aplicaciones reales de la industria, evitando los problemas de costos, daño a personas y/o a equipos. Es posible ver este software de simulación en la figura 1-11.



Fuente: Elaboración propia, realizada en Factory I/O y etiquetada en Paint.

Figura 1 - 11. Planta de nivel en Factory I/ O.

Es posible observar de la figura 1-11, el software de simulación Factory I/O con algunas de sus funciones u opciones enumeradas, las cuales se describen a continuación.

1. Ventana de bienvenida: Muestra la ventana de inicio o bienvenida proporcionando un acceso rápido a documentación variada (manuales, documentación de PLC de diferentes marcas, tarjetas de interfaces, información de escena y componentes, etc).
2. Play y edit: Conmuta entre edición y ejecución.
3. Pausa: En un momento dado permite pausar la simulación. Por ejemplo, podría servir para fijarse en algún detalle de movimiento específico que se haya programado.
4. Reset: Resetea la simulación y en comparación con la opción “Pausa” es que este último paraliza la simulación y “Reset” hace que empiece desde el principio.
5. Escalado de tiempo: Activa una escala de tiempo que permite la acción o simulación en una velocidad mayor, menor, normal, etc.
6. Cámara orbital: Esta cámara en comparación con las demás es que permite una mayor movilidad dentro de la escena, atravesando elementos, es decir, produce que los elementos sean transparentes.
7. Cámara Mosca: Esta cámara permite recorrer dentro de la escena como si fuera una mosca o como un “dron”. Esta cámara impide atravesar los elementos dentro de la escena.
8. Cámara Peatón: Es la cámara que permite circular dentro de la escena como si el operador estuviera realmente dentro de ella, es decir, no permite atravesar objetos ni volar. Podría servir para simular los movimientos que el operador realizaría dentro del proceso o planta en primera persona.
9. Ventana de cámaras: Permite guardar o salvar una vista predeterminada en la escena.
10. Etiquetas de sensores: Al activar esta función permite mostrar etiquetas o tag de los sensores. Los nombres de estas etiquetas se pueden modificar.
11. Etiquetas de actuadores: Muestra o quita las etiquetas o tag de los actuadores dentro de la escena. Los nombres de cada etiqueta se pueden modificar.
12. Ventana de paletas: Muestra o quita la paleta de elementos o herramientas.
13. Es la paleta de elementos, la cual muestra todas las herramientas disponibles en Factory I/O.

14. Es el control de navegación, el cual permite desplazarse dentro de la escena, o sea, es posible moverse hacia adelante o atrás, izquierda o derecha, arriba o abajo, etc.

1.6. OBJETIVOS DEL PROYECTO

A continuación, se muestran tanto los objetivos generales como los específicos que cumplirá el proyecto. Estos permiten obtener una idea clara y definida de lo que se busca demostrar. Además, serán la principal guía para el desarrollo de este proyecto.

1.6.1. Objetivo General

Implementar una estrategia de control ON/OFF con histéresis programada en LabVIEW y Arduino, para regular un sistema de control de nivel en Factory I/O.

1.6.2. Objetivos Específicos

- A. Establecer la comunicación entre Factory I/O, LabVIEW y Arduino.
- B. Crear la interfaz HMI para monitorear el comportamiento del proceso en LabVIEW.
- C. Programar en LabVIEW y Arduino la estrategia de control.
- D. Obtener resultados de la implementación del proyecto.

CAPÍTULO 2: DESARROLLO DE LA SOLUCIÓN

2. DESARROLLO DE LA SOLUCIÓN

En este capítulo se dará a conocer a detalle la solución planteada en el capítulo 1, donde se explicará el sistema de control y la programación necesaria en lenguaje G, para lograr finalmente monitorear y controlar la variable de nivel del sistema entregado por Factory I/O.

2.1. SISTEMA DE CONTROL

En sistemas de ingeniería, para reducir el riesgo de falla y lograr que el rendimiento se mantenga dentro de los márgenes esperado, se utilizan los sistemas de control. El sistema de control es una interconexión de elementos que forman una configuración que permite regular el comportamiento de un proceso industrial para lograr un objetivo de control. Existen dos tipos de sistemas de control, sistemas de control de lazo abierto y de lazo cerrado.

2.1.1. Lazo Abierto

En un sistema de control de lazo abierto, la salida no tiene ningún efecto sobre el sistema de control, es decir, no hay realimentación de dicha salida hacia el controlador para que éste logre cambiar o ajustar la acción de control.

Este sistema de control permite realizar un monitoreo remoto de la planta, puede almacenar datos históricos, posibilita la generación de reportes, son económicos, etc.

Un ejemplo claro donde se aplica el sistema de control a lazo abierto, es en el control de potencia en un microondas: El controlador está formado por un temporizador, que genera un PWM, que enciende y apaga el horno para lograr mayor o menor potencia dependiendo del tiempo que dure el encendido del microondas. El actuador, es el magnetrón que produce las microondas (entre 300MHz y 30 GHz) y el sistema es el plato de comida a calentar.

Ahora bien, en este proyecto se implementará un modo automático, basado en un sistema de control de lazo cerrado, donde el operador o usuario deberá ingresar el setpoint y la histéresis del sistema.

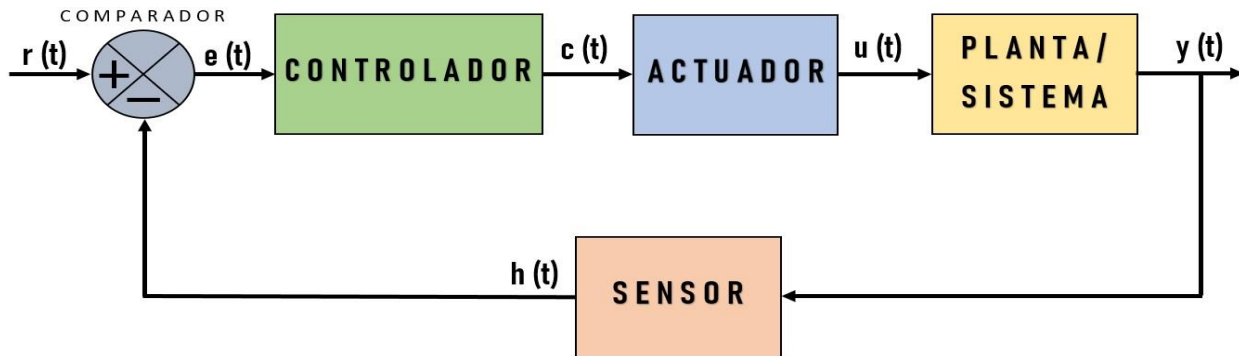
2.1.2. Lazo Cerrado

En los sistemas de control de lazo cerrado la acción de control está en función de la señal de salida, es decir, la salida que se desea controlar se realimenta para realizar la comparación con la señal de referencia (valor deseado o setpoint) y así generar la señal de error que llegará al controlador para decidir finalmente la acción a tomar sobre el proceso. La finalidad de esto, es conseguir disminuir dicho error para llevar la salida del sistema al valor deseado.

Los elementos de un sistema de control en lazo cerrado son los mismos que los de un sistema de control de lazo abierto, a excepción de dos elementos, el comparador y el sensor. El comparador, es aquel elemento que está representado a través de un círculo en el sistema de control. Su función es comparar la señal de referencia con la señal de retroalimentación y calcular el error que existe entre la respuesta deseada y el estado real del sistema. A partir de este error, se puede conseguir que el sistema llegue al estado deseado. El sensor, es aquel elemento que mide la variable a controlar en el sistema, esto permite conocer el estado del sistema y corregir las desviaciones para lograr conseguir la respuesta deseada.

Este sistema de control de lazo cerrado, aumenta la exactitud del control, reduce los efectos de las perturbaciones (perturbación es cualquier variable que ocasiona que la variable de control se desvíe del punto de control y son la causa más común para que el sistema requiera control automático), no depende de la experiencia del operador, es más robusto que el control en lazo abierto, posee una implementación más costosa, permite realizar monitoreo remoto de la planta, puede almacenar datos históricos, posibilita la generación de reportes, etc.

En resumen, un sistema de control de lazo cerrado, es un bucle en el cual todas las etapas necesarias para el control son realizadas por dispositivos; las personas solo supervisan su funcionamiento y establecen los puntos de ajuste. Dicho esto, el sistema de control de lazo cerrado se puede observar en la figura 2-1.



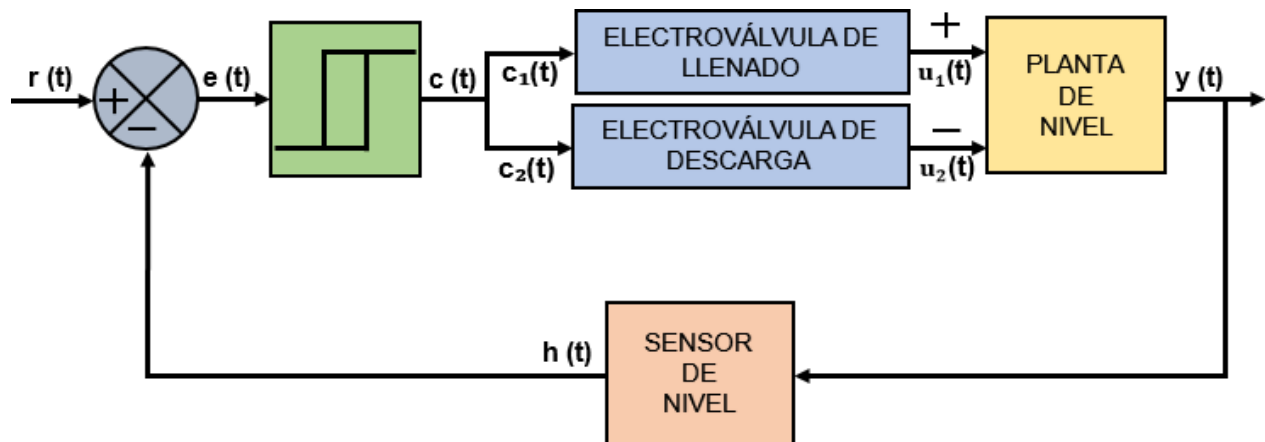
Fuente: Elaboración propia, realizada en Visio.

Figura 2 - 1. Diagrama de bloques - Sistema de control de lazo cerrado.

Las señales del sistema de control de lazo cerrado mostrado en la figura 2-1, se describen a continuación.

- Referencia ($r(t)$): Es el nivel que se desea que alcance la señal de salida ($y(t)$). Es la señal que ingresa a la entrada positiva del comparador.
- Error ($e(t)$): Es la señal de salida del comparador que corresponde a la diferencia entre el estado deseado y el estado real del sistema a controlar.
- Control ($c(t)$): Es la señal de control del sistema, la cual es generada por el controlador.
- Actuación ($u(t)$): Es la señal de salida del actuador que corresponde a la señal de acción que se ejerce sobre la planta o sistema a controlar. Es la variable manipulada.
- Respuesta ($y(t)$): Es aquella señal que corresponde al estado real en que se encuentra el sistema.
- Medición ($h(t)$): Es la medición de la señal de salida del sistema. Es la señal que ingresa a la entrada negativa del comparador.

Dicho lo anterior, a través de la figura 2-2, se darán a conocer los bloques que finalmente formarán parte del sistema de control de lazo cerrado del proyecto.

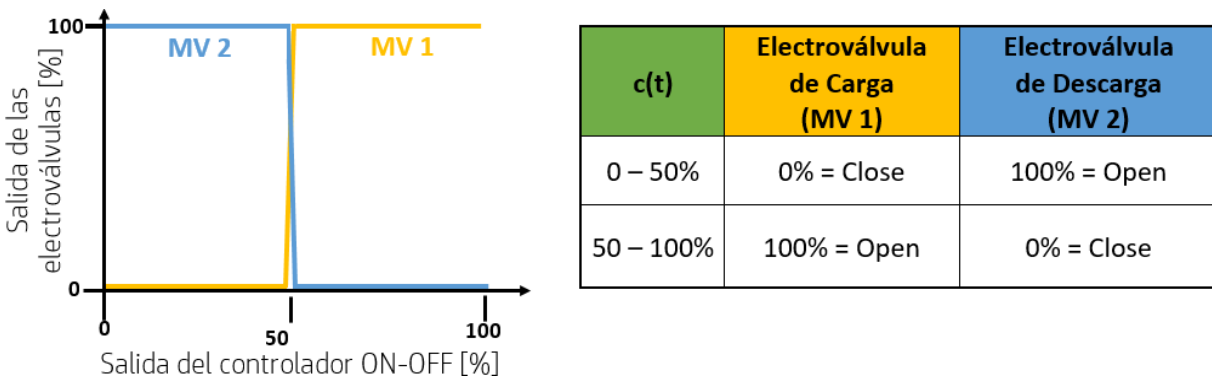


Fuente: Elaboración propia, realizada en Visio.

Figura 2 - 2. Diagrama de bloques - Sistema de control de rango partido de la planta de nivel.

La figura 2-2, muestra un esquema de control de rango partido, donde la señal de salida del controlador se divide y se envía a dos electroválvulas de control (electroválvulas de llenado y descarga). El divisor define cómo se secuencian cada electroválvula a medida que la salida del controlador cambia de 0 a 100%. En otras palabras, en este tipo de control de rango partido existe una sola variable controlada (CV, Controlled Variable), que corresponde al nivel del líquido contenido en el estanque y dos variables manipuladas (MV, Manipulated Variable) que son el flujo de llenado y el flujo de vaciado, los cuales tienen el mismo efecto sobre la variable controlada. La variable controlada, normalmente es la salida o respuesta del sistema ($y(t)$), ya que se mide y se controla la variable actual de la planta o proceso. Por otro lado, la variable manipulada, es la variable que se varía para mantener a la variable controlada en el setpoint o valor deseado.

Por lo tanto, el objetivo de un control de rango partido es controlar varios actuadores (electroválvulas, bombas, cilindros neumáticos o hidráulicos, válvulas, relés, motores de corriente continua o alterna, indicadores luminosos, calefactores, barrera de estados, ventiladores, pasadores de aire, etc), con un solo controlador. El comportamiento de este control, se podrá ver a través de la figura 2-3, mostrada a continuación.



Fuente: Elaboración propia, realizada en Microsoft Excel.

Figura 2 - 3. Comportamiento del control de rango partido o Split Range.

La figura 2-3, muestra la gráfica de “Salidas de las electroválvulas [%]” versus la “Salida del controlador ON-OFF [%]”, la cual se basa en la tabla mostrada a la derecha de esta figura. Esta tabla, indica que en el rango de 0 a 50% de la salida del controlador ON-OFF, la MV 2 o electroválvula de descarga trabaja u opera con un porcentaje de apertura del 100% y la MV 1 o electroválvula de carga se mantiene cerrada o al 0%. También indica que cuando la salida del controlador ON-OFF se encuentra en el rango de 50 a 100%, la electroválvula de descarga no opera, es decir, la salida se mantiene cerrada o al 0% y la electroválvula de carga se abre con un porcentaje de apertura del 100%.

Ahora bien, el sistema de la figura 2-2 muestra una serie de bloques, los cuales se describen de manera detallada a continuación.

2.1.3. Controlador

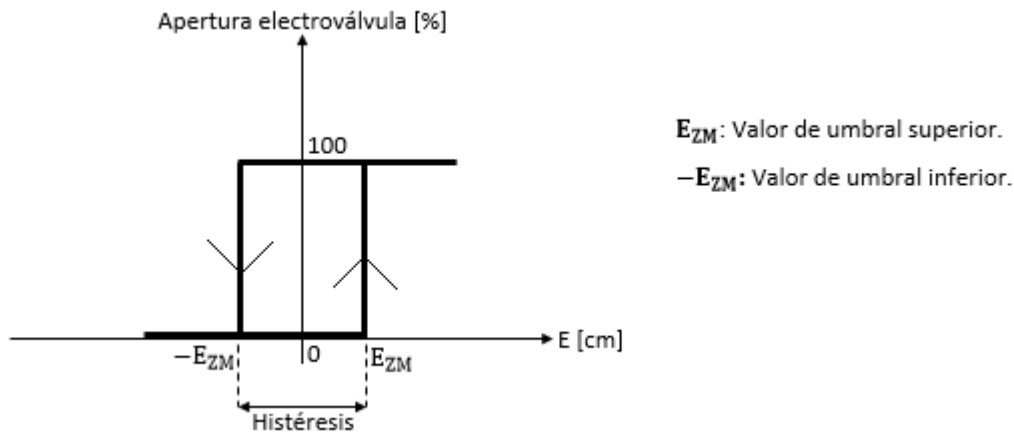
Es el bloque que determina la acción de control que debe aplicarse para modificar la variable de control en base a la salida deseada. Su acción depende del valor de entrada y de la respuesta obtenida (señal retroalimentada). El controlador recibirá todas las señales de los sensores distribuidos en la planta y producirá señales de control de acuerdo a esas entradas y a la estrategia de control que se tenga programado.

El controlador ON/ OFF, es el más común cuando no se tiene un mayor conocimiento del proceso, pero podría generar inestabilidad o imprecisiones en el control. En esta clase de controlador, el elemento final de control o actuador se mueve rápidamente entre una de las dos posiciones fijas que tiene, para un valor único de la CV. El Controlador ON-OFF, opera sobre la MV solo cuando el nivel del agua contenida en el estanque cruza por el punto de consigna. Si la señal de error es mayor a cero, significa que el setpoint está por encima del nivel del líquido contenido en el estanque. Esto ocasiona, que la electroválvula de llenado se abra completamente (ON). Si la señal de error es menor a cero, significa que el setpoint está por debajo del nivel del líquido contenido en el estanque. Esto produce, que la electroválvula de llenado se cierre (OFF).

El Control ON-OFF podría presentar muchas oscilaciones de alta frecuencia si la planta responde relativamente rápido. Las oscilaciones provocarán un envejecimiento prematuro de los elementos de control (electroválvulas, bombas, relés, etc) debido al gran desgaste que estos sufren. El envejecimiento disminuye la vida útil considerablemente. Por lo tanto, una solución sería incorporar el concepto de histéresis, en donde se exige una mínima variación en la variable de proceso (nivel) para que la salida del controlador cambie de ON a OFF o viceversa. Para ello, hay que ajustar una zona muerta con histéresis, o sea, hay que definir un valor " E_{ZM} " (donde " E ", es igual al valor de setpoint o valor deseado menos el valor de la variable del proceso ($E = SP - PV$) en centímetros y " ZM ", es igual a la zona o banda muerta), tal que:

- Si $E \geq E_{ZM}$, entonces la apertura de la electroválvula de llenado será 100% y la apertura de la electroválvula de descarga será de 0% o se mantendrá cerrada.
- Si $E \leq -E_{ZM}$, entonces la apertura de la electroválvula de llenado será 0% y la apertura de la electroválvula de descarga será del 100%.
- Si $-E_{ZM} < E < E_{ZM}$, entonces el estado de ambas electroválvulas no cambia.

El ajuste de la zona o banda muerta, se puede apreciar a través de la figura 2-4, mostrada a continuación.



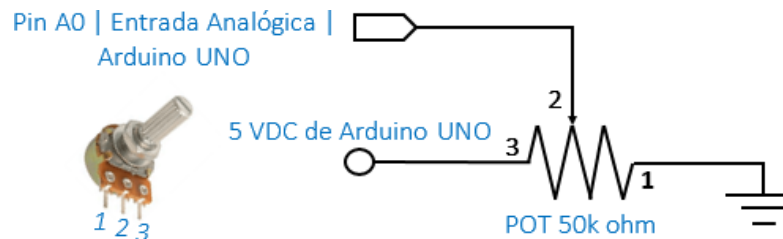
Fuente: Elaboración propia, realizada en Microsoft Word.

Figura 2 - 4. Histéresis en control ON-OFF.

Dada la figura 2-4, resulta posible definir un ancho de histéresis que permita disminuir la frecuencia de cambio en la actuación para lograr operar entorno al punto de equilibrio (setpoint) del sistema. Con lo anterior no se pone en riesgo el desgaste rápido de los elementos de control. El sistema puede oscilar a una frecuencia “controlada” por el ancho de histéresis: a mayor ancho de histéresis (cuando las desviaciones del setpoint son grandes), menor es la frecuencia de oscilación en torno al setpoint; a menor ancho de histéresis (cuando las desviaciones del setpoint son pequeña), mayor es la frecuencia de oscilación en torno al setpoint, es decir, las electroválvulas estarán constantemente abriendo y cerrando.

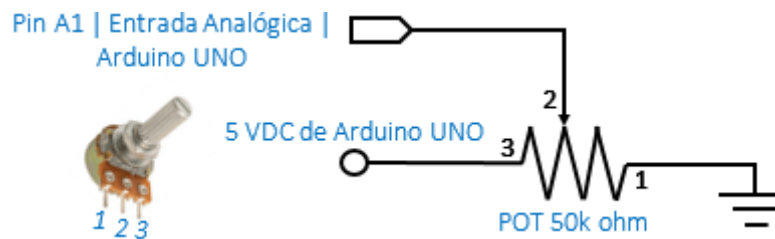
Finalmente, es Arduino UNO el encargado de actuar como controlador, el cual con la ayuda de un potenciómetro rotatorio de carbón de 50k Ω conectado a la entrada analógica correspondiente al pin “A0” y con un potenciómetro rotario de carbón de 50k Ω conectado a la entrada analógica correspondiente al pin “A1”, se podrá fijar el setpoint y además la histéresis de la planta. Para el monitoreo de estado de cada una de las electroválvulas y el funcionamiento del sistema, se utilizarán las salidas digitales correspondiente al pin 2, pin 3 y pin 4. En cada uno de estos pines digitales, irá conectada una resistencia de 150 Ω (previamente calculada) en serie a un LED de color verde para indicar las electroválvulas de llenado y descarga del estanque, y de color rojo para indicar el funcionamiento del sistema de nivel entregado por Factory I/O.

A través de las figuras 2-5 y 2-6, se podrá conocer el conexionado que llevará cada terminal de los potenciómetros encargados de fijar el SP y la histéresis del sistema.



Fuente: Elaboración propia, realizada en Visio.

Figura 2 - 5. Conexión potenciómetro - Setpoint del sistema.



Fuente: Elaboración propia, realizada en Visio.

Figura 2 - 6. Conexión potenciómetro - Histéresis del sistema.

El circuito LED, está constituido por una fuente de poder (obtenida de la salida digital de Arduino UNO, la cual entrega 5VDC), un LED (donde el cátodo generalmente es el pin más corto) y una resistencia.

Por lo tanto, para conocer los cálculos correspondientes a resistencia limitadora de corrientes del LED, se deben observar las figuras 2-7 y 2-8, mostradas a continuación.

$$\text{Resistencia Limitadora} = \frac{\text{Voltaje Salida Digital Arduino} - \text{Voltaje en LED Verde}}{\text{Corriente dentro de lo admisible del LED}}$$

La mayoría de los fabricantes recomiendan una corriente de 20 mA para LED verde.

$$\text{Resistencia Limitadora} = \frac{5V - 2.1V}{20 \text{ mA}} \rightarrow \text{Resistencia Limitadora} = \frac{2.9V}{20 \text{ mA}}$$

$$\text{Resistencia Limitadora} = 145 \text{ Ohm} \rightarrow \text{Resistencia Comercial } 150 \text{ Ohm.}$$

Fuente: Elaboración propia, realizada en Microsoft Word.

Figura 2 - 7. Cálculo resistencia – LED verde.

$$\text{Resistencia Limitadora} = \frac{\text{Voltaje Salida Digital Arduino} - \text{Voltaje en LED Rojo}}{\text{Corriente dentro de lo admisible del LED}}$$

La mayoría de los fabricantes recomiendan una corriente de 20 mA para LED rojo.

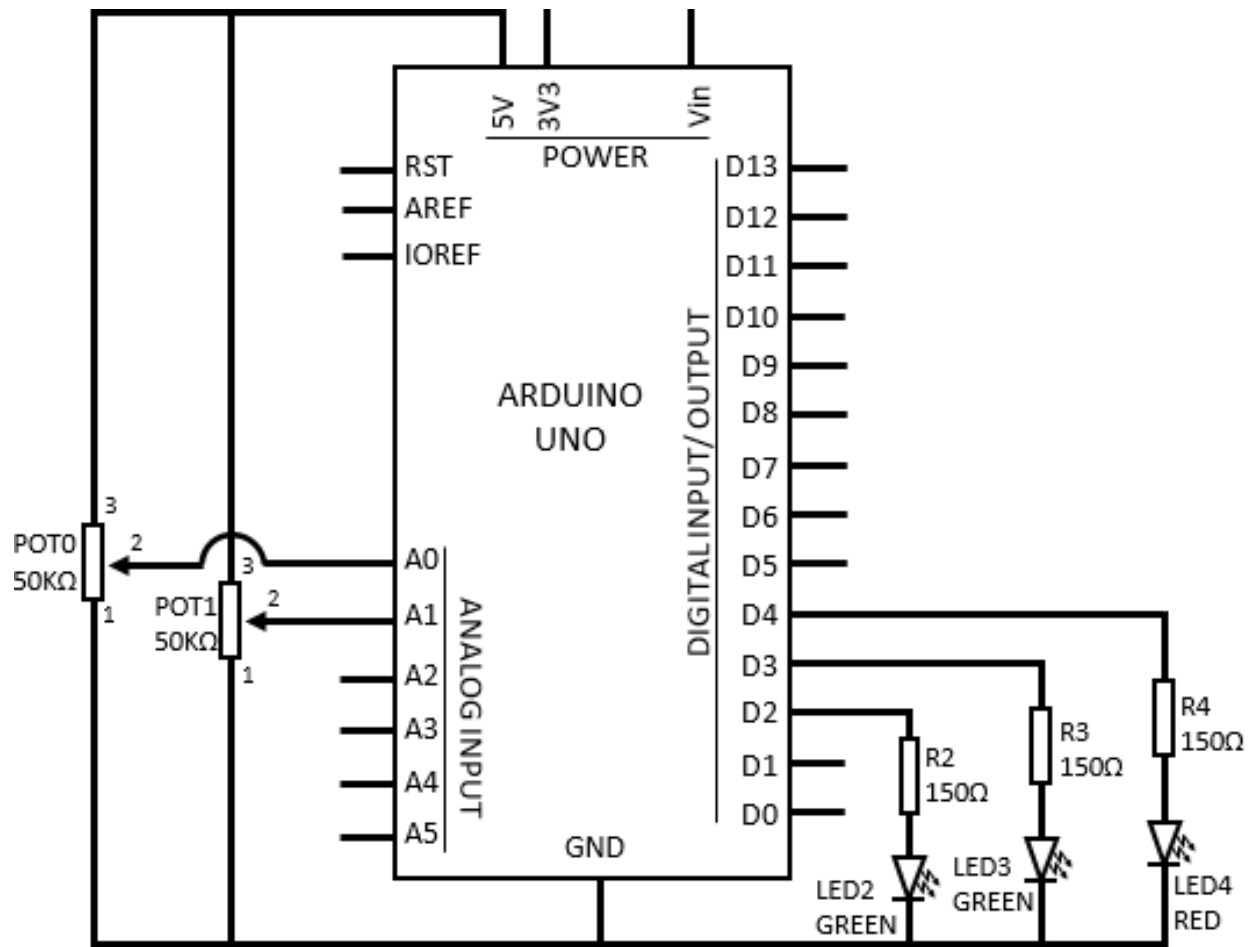
$$\text{Resistencia Limitadora} = \frac{5V - 1.7V}{20 \text{ mA}} \rightarrow \text{Resistencia Limitadora} = \frac{3.3V}{20 \text{ mA}}$$

$$\text{Resistencia Limitadora} = 165 \text{ Ohm} \rightarrow \text{Resistencia Comercial } 150 \text{ Ohm.}$$

Fuente: Elaboración propia, realizada en Microsoft Word.

Figura 2 - 8. Cálculo resistencia – LED rojo.

Finalmente, a través de la figura 2-9, se podrá conocer el circuito electrónico referido a las conexiones de cada entrada analógica y salida digital de Arduino UNO.



Fuente: Elaboración propia, realizada en Microsoft Word.

Figura 2 - 9. Circuito electrónico – Arduino UNO.

La figura 2-9, muestra algunos componentes conectados en las entradas analógicas y en las salidas digitales de Arduino UNO, los cuales se detallan a continuación.

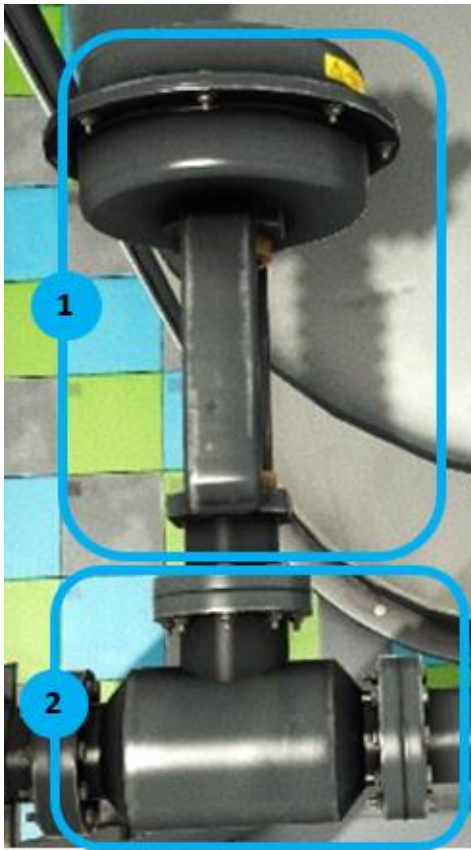
- POT0: Este potenciómetro rotatorio de carbón de 50kΩ, conectado a la entrada analógica “A0” de Arduino UNO, a través del terminal 3, tiene la función de fijar o asignar el setpoint o valor deseado del proceso. El terminal 2 de este potenciómetro, se conecta a 5VDC y el terminal 1 a GND que entrega Arduino UNO.

- POT1: Es el potenciómetro rotatorio de carbón de 50k Ω , conectado a la entrada analógica “A1” de Arduino UNO, a través del terminal 3, el cual tiene la función de fijar o asignar la histéresis del proceso.
- LED2: Este LED, enciende solamente cuando se activa la electroválvula de llenado. El ánodo va unido a un terminal de la resistencia “R2” y el cátodo va unido a tierra o GND.
- LED3: Es el LED verde, que informa mediante el encendido la activación de la electroválvula de descarga. El cátodo de este LED, se conecta a tierra, mientras que el ánodo se conecta a uno de los terminales de la resistencia “R3”.
- LED4: Se refiere al LED rojo, que enciende solamente cuando el proceso o planta de Factory I/O entra en funcionamiento. El ánodo de este LED se conecta a uno de los terminales de la resistencia “R4” y el cátodo se conecta a GND.

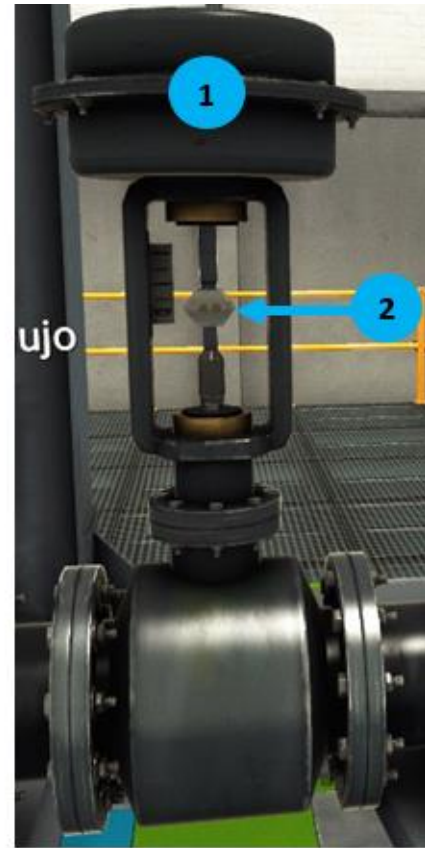
2.1.4. Actuador

En un proceso industrial los actuadores son dispositivos importantes que pueden convertir la energía eléctrica, neumática o hidráulica en la activación de un proceso para permitir que el procesamiento produzca los efectos de un proceso automatizado. El controlador genera la orden para activar o desactivar el actuador o elemento final de control (que podría ser una bomba, un motor, un relé, etc), de tal forma que el actuador modifica la variable manipulada para que la variable controlada llegue al valor deseado.

Este sistema de control de rango partido del proyecto, cuenta con 2 actuadores que son la electroválvula de carga o llenado y a la electroválvula de descarga. Estas electroválvulas de Factory I/O, juegan un rol fundamental en el bucle de regulación, ya que reciben la señal de control ($c(t)$) producida por Arduino UNO (controlador) permitiendo aumentar o disminuir de alguna manera el nivel de líquido dentro del estanque. Estas electroválvulas se pueden observar en la figura 2-10.



Electroválvula de carga.
1: Actuador. 2: Cuerpo.



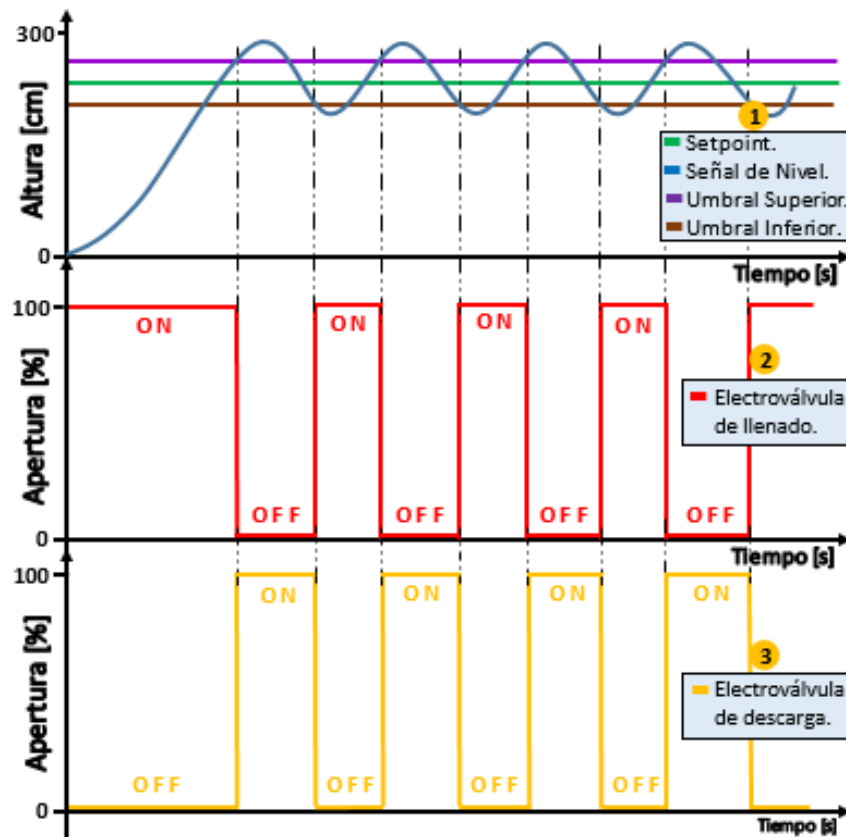
Electroválvula de descarga.
1: Diafragma. 2: Vástago.

Fuente: Elaboración propia, realizada en Paint.

Figura 2 - 10. Electroválvula de carga y descarga del líquido.

Se puede observar en la figura 2-10, que las electroválvulas están constituidas primordialmente por un actuador y un cuerpo. En el actuador, se encuentra el diafragma (puede ser activado por una señal neumática, hidráulica, eléctrica, etc), el cual se expande al activarse permitiendo el desplazamiento del vástago hacia arriba o hacia abajo. El cuerpo, proporciona la conexión de flujo y la restricción del mismo con la ayuda del obturador o tapón, el cual se encuentra acoplado o unido al vástago. Por lo tanto, la pieza que finalmente va a permitir o impedir el flujo de agua en estas electroválvulas, y, en consecuencia, llevar el nivel del estanque hacia un valor deseado o setpoint será el obturador.

Estas electroválvulas tienen un comportamiento un tanto repetitivo en el tiempo, de acuerdo al setpoint e histéresis utilizada. Dicho comportamiento, se puede estudiar por medio de la figura 2-11, mostrada a continuación.



Fuente: Elaboración propia, realizada en Microsoft Word.

Figura 2 - 11. Evolución de las electroválvulas en el tiempo.

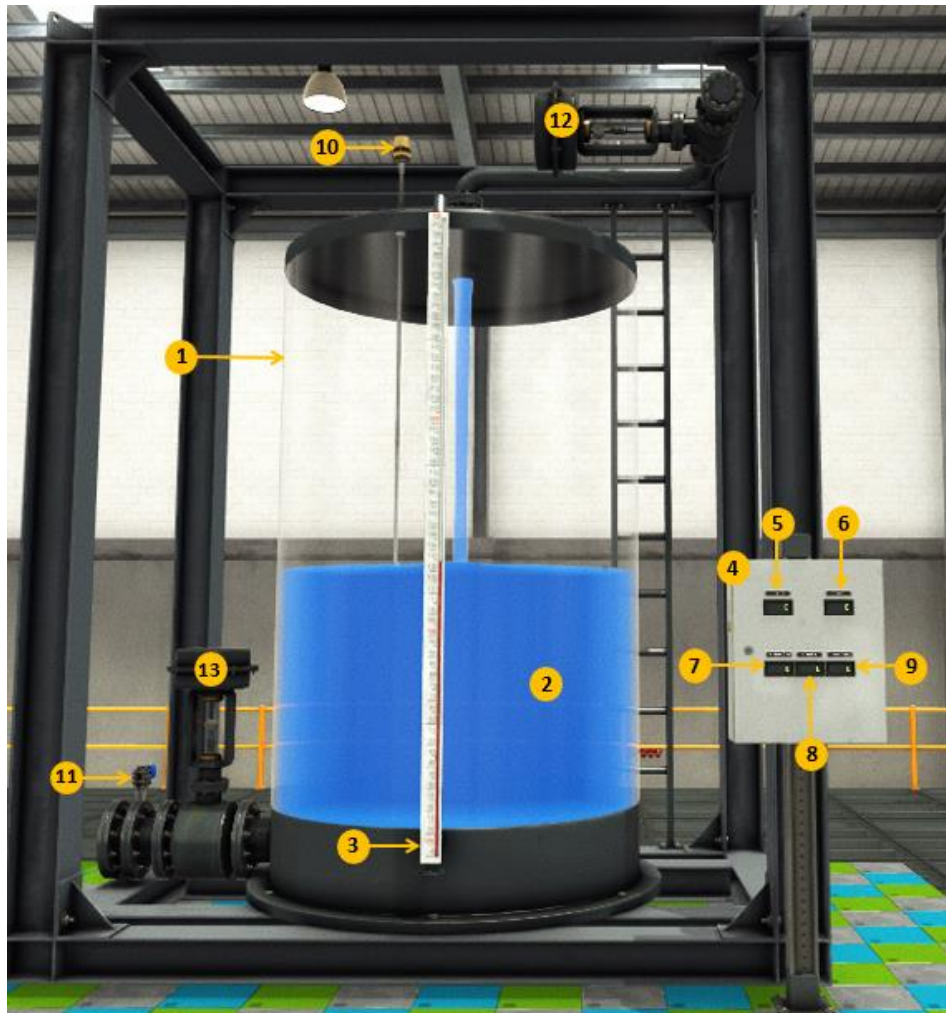
La figura 2-11, muestra el comportamiento de las variables manipuladas o MV 1 y MV 2 (que son las aperturas de las electroválvulas de carga o llenado y de descarga) cuando la respuesta del sistema o el nivel del líquido contenido al interior del estanque, llega finalmente al umbral superior ($\text{setpoint} + (0,5 \cdot \text{histéresis})$) y al umbral inferior ($\text{setpoint} - (0,5 \cdot \text{histéresis})$) fijado a través de los potenciómetros conectados a las entradas analógicas (Arduino UNO).

Estas variaciones u ondas que va formando la señal de nivel (nivel del estanque de Factory I/O en tiempo real) mostrada en la gráfica 1, se deben a que, al iniciar el sistema o proceso, éste comienza a buscar el valor del umbral superior (fijado por el usuario), donde tendrá 2 caminos a seguir según la altura que tenga en tiempo real, el nivel de líquido contenido en el estanque. Dicho esto, se debe asumir que, al momento de iniciar el sistema, el estanque se encuentra totalmente vacío, lo que hace que la electroválvula de llenado mostrada en la gráfica 2, se active y en consecuencia empiece a llenar el depósito para subir el nivel de líquido hasta el umbral superior, que es donde finalmente esta electroválvula se desactiva. Cuando esta electroválvula de llenado se desactiva, rápidamente comienza a funcionar la electroválvula de descarga mostrada en la gráfica 3, para lograr disminuir el exceso de nivel generado y para llevar el sistema al valor de umbral inferior. Al llegar el nivel o la respuesta al valor de umbral inferior, se desactiva la electroválvula de descarga y se activa nuevamente la electroválvula de llenado para subir el nivel del estanque hasta el valor de umbral superior. Esto, formará una especie de bucle o ciclo, con el fin de intentar hacer que el error sea cero.

Ahora bien, en esta programación en particular la desactivación de una electroválvula, trae consigo la activación de la otra electroválvula. Esto se debe, a que al desactivarse una de estas, no se realiza de manera rápida o de manera totalmente síncrona, sino que se produce un efecto de retardo. Por lo tanto, hace que el llenado o vaciado del estanque no se detenga exactamente en los límites correspondientes (umbral superior e inferior), sino que se detiene un poco más arriba o un poco más abajo de estos puntos o límites.

2.1.5. Planta

En este bloque llamado planta, se encuentra el sistema que finalmente se va a controlar. La planta a controlar será una escena entregada por el software de simulación Factory I/O, llamada sistema de nivel, la cual se puede ver en la figura 2-12.



Fuente: Elaboración propia, realizada en Factory I/O y etiquetada en Paint.

Figura 2 - 12. Planta de nivel a controlar.

El sistema de nivel mostrado en la figura 2-12, consta de una serie de elementos enumerados, los cuales se describen a continuación.

1. Estanque: Es el estanque o recipiente donde se depositará el líquido.
2. Líquido: Este líquido es una variable en la planta, el cual puede ser un fluido inflamable, corrosivo, reactivo o radioactivo, sin embargo, para efecto de prueba se utilizará agua.

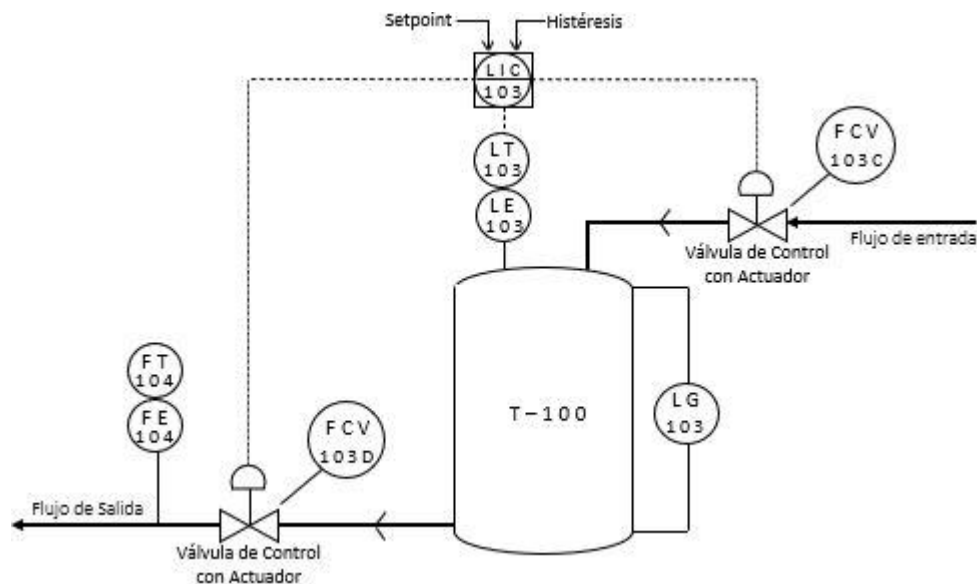
3. Indicador de nivel: Se encarga de mostrar el nivel que lleva el líquido contenido en el estanque. El nivel del agua, puede ir de 0 a 300 cm de altura.
4. Tablero de distribución: Este tablero permite mostrar algunas variables del proceso.
5. Display SP: Visualizador que permite mostrar el valor deseado o setpoint, fijado manualmente por el usuario (en centímetros). Finalmente, este display se comunicará con el programa de ingeniería LabVIEW para su posterior monitoreo.
6. Display PV: Es el visualizador ubicado en el tablero de distribución, que muestra la variable del proceso (Process Value) o la variable medida que se desea estabilizar. En pocas palabras, muestra el nivel que lleva el líquido contenido en el estanque expresado en centímetros. Al igual que el display anterior, éste será visualizado finalmente en LabVIEW.
7. Display Umbral Superior: Este visualizador, muestra el valor (en centímetros) que tomará el umbral superior. Posteriormente, se podrá visualizar o monitorear a través de LabVIEW.
8. Display Histéresis: Este visualizador ubicado en el tablero de distribución, permite mostrar el valor de la histéresis (en centímetros), fijada manualmente por el usuario. Finalmente, este display se comunicará con LabVIEW para su posterior monitoreo.
9. Display Umbral Inferior: Es un visualizador del tablero de distribución (tag 4), que muestra el valor (en centímetros) que tomará el umbral inferior. Posteriormente, se podrá visualizar en LabVIEW.
10. Sensor de nivel: Se encarga de medir el nivel de líquido que lleva el estanque. Las mediciones de este sensor se podrán apreciar posteriormente en LabVIEW.
11. Sensor de flujo: Este sensor se encarga de medir el caudal que fluye a través del terminal de salida de la electroválvula de descarga cuando está en funcionamiento.
12. Electroválvula de llenado: Tal y como su nombre lo indica esta electroválvula se encarga de llenar o aumentar el nivel de líquido contenido en el estanque, con la ayuda del obturador que se desliza hacia arriba (cuando se activa o entra en funcionamiento esta electroválvula) permitiendo el paso del líquido a través de su cuerpo.

13. Electroválvula de descarga: Al activarse permite el paso del líquido a través de su cuerpo, lo que genera la disminución del nivel de líquido contenido en el estanque.

2.1.5.1. P&ID

Un P&ID (Piping and Instrumentation Diagram) o diagrama de tuberías e instrumentación, muestra los instrumentos, los equipos instalados y el flujo del proceso en las tuberías. Exhibe una sucesión de símbolos que permiten encontrar o identificar todos los componentes que constituyen un proceso, como equipos, válvulas, controles, niveles, alarmas, drenajes, bombas, presostatos, purgas, tuberías, número de líneas de tubería y sus dimensiones, etc.

Ahora bien, la norma utilizada para definir la simbología del P&ID correspondiente al sistema de nivel, es la norma ISA S5.1(International Society of Automation). La aplicación de esta norma se podrá visualizar a continuación a través del P&ID contenido en la figura 2-13.



Fuente: Elaboración propia, realizada en la página www.lucidchart.com

Figura 2 - 13. P&ID del sistema de nivel.

La figura 2-13, da a conocer el P&ID de la planta de nivel proporcionada por Factory I/O, donde se logran apreciar algunas etiquetas, las cuales se describen a continuación.

- ✓ FCV-103 C: Válvula controladora de flujo. Esta válvula se encuentra en el lazo de control número 103. El sufijo “C”, permite hacer la diferencia con otro instrumento ubicado dentro del mismo lazo y que cumple la misma función.
- ✓ FCV-103 D: Válvula controladora de flujo. Esta válvula se encuentra en el lazo de control número 103. El sufijo “D” permite hacer la diferencia con la válvula FCV-103 C, puesto que, cumplen la misma función y se encuentran dentro del mismo lazo de control.
- ✓ FE-104: Corresponde al sensor de flujo que se encuentra ubicado en el lazo de control número 104.
- ✓ FT-104: Esta etiqueta se refiere al transmisor de flujo ubicado en el lazo de control número 104.
- ✓ LG-103: Hace referencia al indicador de nivel en vidrio ubicado en el lazo de control número 103.
- ✓ LE-103: Es el sensor de nivel en lazo de control número 103.
- ✓ LIC-103: Esta etiqueta se refiere al controlador indicador de nivel que se encuentra en el lazo de control número 103.
- ✓ LT-103: Transmisor de nivel en lazo de control número 103.
- ✓ T-100: Esta etiqueta hace referencia al estanque que se encuentra en el lazo de control número 100.
- ✓ Flujo de entrada: Se refiere a la tubería por donde va a ingresar el líquido o fluido que luego pasará por la válvula controladora de flujo FCV-103 C.
- ✓ Flujo de salida: Es la tubería donde sale o se descarga todo el líquido proveniente del estanque o recipiente.
- ✓ Setpoint: Es el valor que fija el usuario manualmente por medio de un potenciómetro conectado al pin “A0” de la entrada analógica de Arduino UNO.
- ✓ Histéresis: Es el valor que fija el usuario manualmente, a través de un potenciómetro conectado al pin “A1” de la entrada analógica de Arduino UNO.

Una vez fijado de forma manual por el operador el setpoint o valor deseado y la histéresis, el sensor LE-103 se encarga de medir el nivel de líquido contenido en el estanque. Luego, lleva esta información o medición al transmisor LT-103, quien, por medio de una señal eléctrica, envía estos datos al controlador e indicador LIC-103. El LIC-103 se ocupa de realizar la corrección necesaria para llevar el nivel de líquido al umbral superior o al umbral inferior. Cuando el nivel del estanque se encuentra por debajo del valor de umbral superior, el LIC-103 genera una señal eléctrica que llega a activar la electroválvula de llenado FCV-103C para aumentar el nivel de líquido contenido en el estanque. Por otro lado, cuando el nivel está por encima del valor de umbral inferior, el controlador e indicador de nivel LIC-103, se ocupa de generar una señal eléctrica nuevamente que llega a activar la electroválvula de descarga FCV-103D y a desactivar la electroválvula de llenado FCV-103C de forma paralela. Activada la electroválvula FCV-103D, se podrá medir el flujo de salida con la ayuda del sensor FE-104 y el transmisor FT-104. Ahora bien, mientras el sistema esté en funcionamiento, el operador en planta, podrá visualizar el nivel o la altura que lleva el líquido (en centímetros) contenido en el estanque, a través del indicador LG-103.

2.1.6. Sensor

Este bloque permite medir o conocer los valores de las variables medidas de la planta, sistema de nivel o respuesta.

Por lo tanto, mediante el sensor de nivel LE-103 se conocerán los valores necesarios para interpretar en tiempo real el punto en el que se encuentra el nivel del líquido contenido en el estanque. El sensor de nivel LE-103 entregará o arrojará datos para conocer la altura que lleva el líquido en unidad de longitud basada en centímetros. Finalmente, el usuario u operador podrá visualizar estos datos por medio del panel frontal de LabVIEW.

2.2. COMUNICACIÓN

Antes de empezar la programación necesaria en lenguaje G a través del software de ingeniería LabVIEW, se debe entender cómo se realiza la comunicación entre LabVIEW y Factory I/O.

2.2.1. Comunicación en Factory I/O

Se debe abrir la escena llamada “Level Control” en Factory I/O, que es una escena que ya viene establecida, esto quiere decir, que el usuario podrá operar la planta de manera inmediata, pero solamente de forma manual. En esta escena, se pueden agregar o quitar elementos, cambiar nombres de los elementos agregados o existentes que cumplan alguna función importante, etc. Todos los elementos de datos que cumplan alguna función dentro de la escena (como displays, sensores, actuadores, etc), irán almacenados por lo general en uno de los 4 bancos de datos o rangos de dirección del protocolo de comunicación escogido. En esta oportunidad se ha escogido como protocolo de comunicación “Modbus TCP/IP Server”, el cual tiene como bancos de datos los “Coils” (bobinas), “Discretes Input” (Entradas Discretas), “Input Registers (Registros de Entradas) y “Holding Registers” (Registros de Retención).

Modbus, es un protocolo de comunicación ubicado en los niveles 1, 2 y 7 del Modelo OSI, basado en la arquitectura maestro/ esclavo (RTU) o cliente/ Servidor (TCP/ IP). Se puede utilizar para controlar una red de dispositivos, como sistemas de medición de temperatura y humedad, y transmitir los resultados a una PC. Modbus, además, se puede usar para la conexión de un ordenador de supervisión con una unidad remota (RTU) en sistemas de supervisión y adquisición de datos (SCADA).

Modbus TCP/IP o Modbus TCP, es una variante de Modbus utilizada para comunicaciones a través de redes TCP/IP, conectándose por medio del “Puerto 502”. No es necesario calcular una suma de verificación (también llamada suma de chequeo o checksum), porque los niveles inferiores ya brindan protección de checksum.

Ahora bien, dicho esto y continuando con los bancos de datos correspondiente al protocolo de comunicación “Modbus TCP/IP Server” (Coils, Input Discretes, Input Registers y Holding Registers), resulta necesario mencionar que estos bancos de datos definen el tipo y los derechos de acceso de los datos contenidos en estos bloques, los cuales brindan la habilidad de restringir o permitir el acceso a los diferentes elementos. Estos bancos de datos, contienen un espacio de memoria de un máximo de 65536 elementos. Así mismo, Modbus define la dirección de cada elemento de dato que va desde 0 hasta 65535, por ejemplo, la bobina “4” está en el bloque de bobina en la dirección “3”; el registro de retención “6” está en la dirección “5” del bloque de memoria que el servidor ha definido como registro, etc.

Por lo tanto, es posible a través de la tabla 2-1, conocer a detalle los bancos de datos que proporciona el protocolo de comunicación “Modbus TCP/IP Server”, con sus respectivas características.

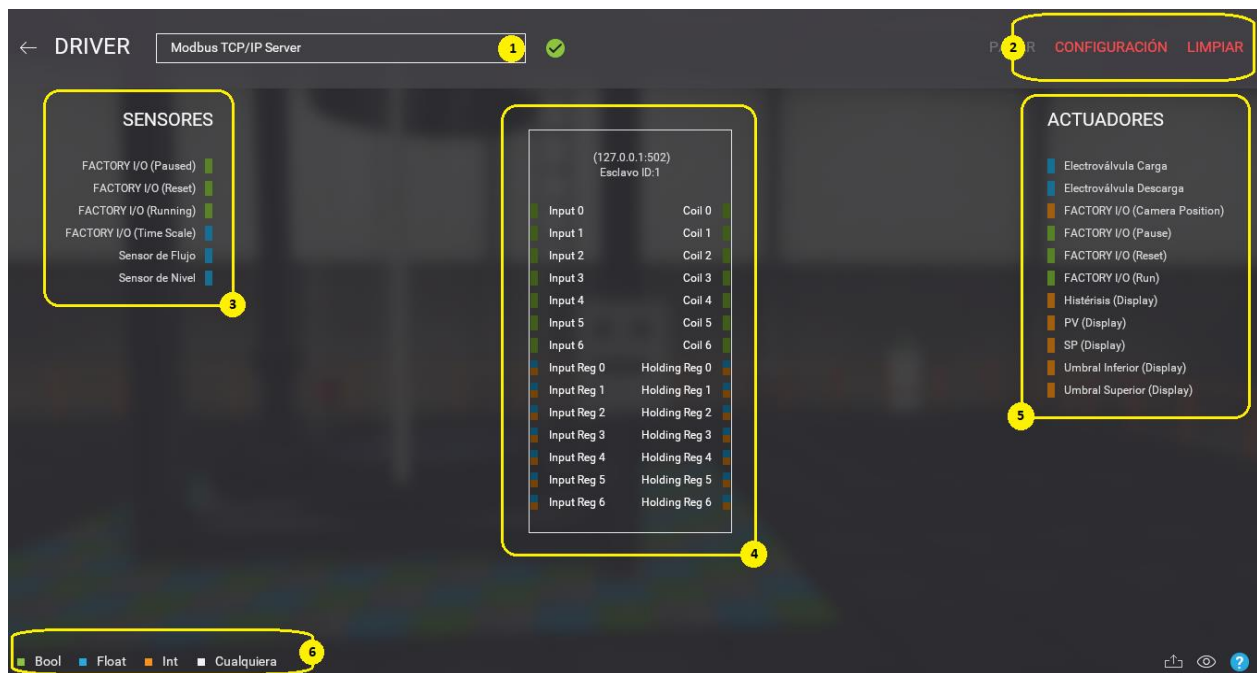
Tabla 2 - 1. Bancos de datos o bloques de memoria.

Bloque de Memoria	Tipo de Dato	Tipo de Acceso	Tipo de Registro	Rango de Registro [0 , 65535]
Coils (Bobinas)	Booleano 1 bits	Lectura/ Escritura	Salidas Digitales	[000001 , 065535]
Discrete Inputs (Entradas Discreta)	Booleano 1 bits	Lectura	Entradas Digitales	[100001 , 165535]
Input Registers (Registros de Entrada)	Word sin signo 16 bits	Lectura	Entradas Analógicas	[300001 , 365535]
Holding Registers (Registros de Retención)	Word sin signo 16 bits	Lectura/ Escritura	Salidas Analógicas	[400001 , 465535]

Fuente: Elaboración propia, realizada en Microsoft Excel.

Tal y como se puede observar en la tabla 2-1, las entradas discretas y las bobinas trabajan datos booleanos o datos de 1 bit, es decir, pueden tener solamente 2 estados que pueden ser verdadero o falso, 1 o 0, etc. En cambio, los registros de entradas y registros de retención operan datos superiores a un 1 bit, o sea, operan datos de 16 bit o 2 Bytes, que van desde el 0 a 65535. Los datos que lleguen a los bancos de entradas discretas y registros de entradas no se pueden modificar, es decir, solamente se podrá hacer lectura y no escritura de ellos. Mientras que, en los bancos restantes (bobinas y registros de retención), se puede escribir o modificar y leer o monitorear un elemento o dato.

Dicho esto, es posible observar por medio de la figura 2-14, todos los elementos y funciones disponibles del sistema de nivel que entrega Factory I/O, los cuales se podrán ocupar posteriormente con la modalidad de acceso a lectura y/o escritura según corresponda en LabVIEW.



Fuente: Elaboración propia, realizada en Factory I/O y etiquetada en Paint.

Figura 2 - 14. Driver del sistema o escena de nivel.

Para acceder a la ventana que muestra la figura 2-14, se debe presionar la tecla “F4” o hacer clic en el menú “ARCHIVO” y luego en la opción “Drivers” de Factory I/O. Ahora bien, esta figura presenta una serie de etiquetas enumeradas, las cuales se describen a continuación.

1. Driver: Aquí se debe escoger el controlador o protocolo de comunicación que permitirá el intercambio de información entre Factory I/O y LabVIEW. El protocolo de comunicación a escoger es “Modbus TCP/IP Server”.
2. Configuración muestra una ventana, donde se puede ajustar algunos parámetros de la comunicación, como definir el host o número de IP, el puerto para comunicaciones a través de redes TCP/IP, el número de bancos de memorias, la escala de los sensores y actuadores, etc. Limpiar, permite retirar o borrar todos los elementos de datos puestos en los bloques de memorias.
3. Sensores: Son todos los elementos de datos que de alguna u otra manera participan en la escena de nivel como un sensor. Estos elementos, solamente se pueden leer o monitorear, pero no se pueden modificar o escribir. Todos estos elementos de datos se pueden conocer a continuación.

- ✓ FACTORY I/O (Paused): Este elemento de dato booleano, se refiere al estado del botón “Pause” de la escena de nivel, el cual permite desde LabVIEW solamente la lectura y no la escritura, es decir, se podrá monitorear, pero no modificar el estado de dicha opción.
- ✓ FACTORY I/O (Reset): Es un elemento de dato booleano, que se refiere al estado del botón “Reset” de la escena de nivel, el cual permite desde LabVIEW solamente la lectura y no la escritura, es decir, se podrá monitorear, pero no controlar el estado de esta opción.
- ✓ FACTORY I/O (Running): Este elemento de dato booleano, se refiere al estado del botón “Play” de la escena de nivel, el cual permite desde LabVIEW solamente la lectura y no la escritura, es decir, se podrá monitorear, pero no modificar el estado de dicha opción.

- ✓ FACTORY I/O (Time Scale): Es un elemento de dato flotante, que se refiere al factor que tenga el “Selector de escalado de tiempo” en la escena de nivel, el cual permite desde LabVIEW solamente la lectura y no la escritura, es decir, se podrá monitorear o leer el factor que tenga el selector (puede ser x0.1, x0.5, x1, x2, x3 y x4), pero no cambiar dicho factor.
 - ✓ Sensor de Flujo: Este elemento de dato flotante, se refiere a los datos que entrega el sensor de flujo instalado en la tubería de descarga de la escena de nivel. Se podrá leer y no escribir en LabVIEW un valor que irá desde el 0 hasta el valor que tenga la “Escala” (asignada por el usuario) multiplicada por 10 (que es el máximo valor que pueden tomar todos los sensores y actuadores), es decir, si la escala fuera de 100, los datos de este sensor irán desde el 0 hasta 1000 ($10 * 100 = 1000$).
 - ✓ Sensor de Nivel: Es un elemento de dato flotante, que se refiere a los datos que proporciona el sensor de nivel instalado en el estanque de la escena de nivel. Se podrá leer y no escribir en LabVIEW un valor que irá desde el 0 hasta el valor que tenga la “Escala” multiplicada por 10.
4. Es el servidor o conjunto de bloques de memorias donde se colocan y se distribuyen finalmente los elementos de datos de la escena de nivel, que se ocuparán posteriormente en LabVIEW.
 5. Actuadores: Son todos los elementos de datos que participan de alguna u otra manera en la escena de nivel como un actuador. Estos elementos, se pueden leer o monitorear y además permiten la escritura de datos.
- ✓ FACTORY I/O (Camera Position): Este elemento de dato entero, se refiere al tipo de cámara que se tenga seleccionada en la escena de nivel. Se podrá desde LabVIEW leer la información del selector y escribir o elegir el tipo de cámara (cámara orbital, mosca y peatón) que se desee en la escena de Factory I/O.

- ✓ FACTORY I/O (Pause): Es un elemento de dato booleano, que se refiere nuevamente al estado del botón “Pause” de la escena de nivel, pero con la diferencia que esta vez desde LabVIEW se podrá leer la información de estado y activar o desactivar por medio de un 0 o 1 dicha opción.
- ✓ FACTORY I/O (Reset): Este elemento de dato booleano, se refiere nuevamente al estado del botón “Reset” de la escena de nivel, pero con la diferencia que esta vez desde LabVIEW, se podrá leer la información de estado y activar o desactivar por medio de un “True” o “False” dicha opción.
- ✓ FACTORY I/O (Run): Este es un elemento que trabaja con valores o datos booleanos y se refiere nuevamente al estado del botón “Play” de la escena de nivel, sin embargo, con la diferencia que esta vez desde LabVIEW, se podrá leer la información de estado y además se podrá activar o desactivar mediante un interruptor o botón (por ejemplo), el arranque de la escena de nivel o planta que entrega Factory I/O.
- ✓ Histéresis (Display): Elemento de dato entero, que se refiere al display que muestra el valor de la histéresis (en cm) en el tablero de distribución de la planta o escena de nivel de Factory I/O. Este elemento, se podrá monitorear o leer y además escribir posteriormente en LabVIEW un valor entero que luego se mostrará en el tablero de distribución ya mencionado.
- ✓ PV (Display): Este es un elemento de dato entero, que está relacionado al display instalado en el tablero de distribución de la escena de nivel, el cual indica la altura (en cm) que lleva el líquido contenido en el estanque. Se podrá monitorear o leer y además escribir desde LabVIEW un valor entero que luego se mostrará en el tablero de distribución ya mencionado.
- ✓ SP (Display): Es un elemento de dato entero, que está relacionado al display instalado en el tablero de distribución de la escena de nivel, el cual indica el valor de setpoint (en cm). Se podrá monitorear o leer y además escribir desde LabVIEW un valor entero que luego se mostrará en el tablero de distribución ya mencionado.

- ✓ Umbral Inferior (Display): Este es un elemento de dato entero, que está relacionado al display instalado en el tablero de distribución de la escena de nivel, el cual muestra el valor de umbral inferior (en cm). Se podrá monitorear o leer y además escribir desde LabVIEW un valor entero que luego se mostrará en el tablero de distribución ya mencionado.
 - ✓ Umbral Superior (Display): Es un elemento de dato entero, que se refiere al display instalado en el tablero de distribución de la escena de nivel, el cual indica el valor de umbral superior (en cm). Se podrá monitorear o leer y además escribir desde LabVIEW un valor entero que luego se mostrará en el tablero de distribución ya mencionado.
 - ✓ Electroválvula Carga: Este es un elemento de dato flotante, que está relacionado con la electroválvula de carga instalada en el estanque de la escena de nivel. Se podrá desde LabVIEW leer la información y además escribir un valor flotante que irá desde 0 hasta el valor que tenga la “Escala” multiplicada por 10 (es el máximo valor que puede tomar esta electroválvula en la escena). Por ejemplo, si la escala fuera de 20, los datos de esta electroválvula irán desde el 0 hasta 200 ($10 * 20 = 200$).
 - ✓ Electroválvula Descarga: Este es un elemento de dato flotante, que está relacionado con la electroválvula de descarga o vaciado del líquido instalada a los pies o a la base del estanque de la escena de nivel. Al igual que la electroválvula de llenado se podrá monitorear o leer y además escribir desde LabVIEW un valor flotante que irá desde 0 hasta el valor que tenga la “Escala” multiplicada por 10.
6. Muestra todos los tipos de datos o números con los que pueden llegar a operar los elementos que participan en la escena de nivel. Estos datos se pueden conocer a detalles a continuación.

- ✓ Bool (en color verde): Este se refiere al dato booleano, el cual puede tener solamente dos valores, ya que, representa valores de lógica binaria, y por lo general se pueden mostrar con un valor que sea verdadero o falso.
- ✓ Float (en color azul): Es el dato o número flotante y es un tipo de dato que representa la cantidad de 32 bit (4 bytes). Se utiliza para representar números decimales, números enteros con orden de magnitud más grande.
- ✓ Int (en color naranja): Se refiere a los datos o números enteros, o sea, a los datos que no tienen decimales, tanto positivos como negativos (además del cero)
- ✓ Cualquiera (en color blanco): Es un dato o número cualquiera, es decir, puede ser un dato booleano, flotante, entero, etc.

Estos tipos de datos junto a su respectiva representación en colores, se podrá resumir a través de la tabla 2-2, mostrada a continuación.

Tabla 2 - 2. Colores en los tipos de datos.

Tipo de Dato	Color
Booleano	Verde
Flotante	Azul
Entero	Naranja
Cualquiera	Blanco

Fuente: Elaboración propia, realizada en Microsoft Excel.

Mencionado lo anterior, a través de la figura 2-15 se podrá conocer la ventana de “Configuración”, la cual permite asignar o fijar una serie de parámetros necesarios para que exista de alguna u otra manera la comunicación entre Factory I/O y LabVIEW.

Servidor

☒ Autoarranque

Host
 1

Puerto
 2

Esclavo ID
 3

Adaptador de RED
 4

I/O Configuración

Escritura Digital
 5

Lectura Digital
 6

Escribir Registro
 7

Leer Registro
 8

Escala
 9

I/O Terminales 10

	Offset	Cuenta
Salidas Bool	0	7
Entradas Bool	0	7
Entradas Registro	0	7
Salidas Registro	0	7

Fuente: Elaboración propia, realizada en Factory I/O y etiquetada en Paint.

Figura 2 - 15. Ajustes de parámetros - Configuración en driver del sistema.

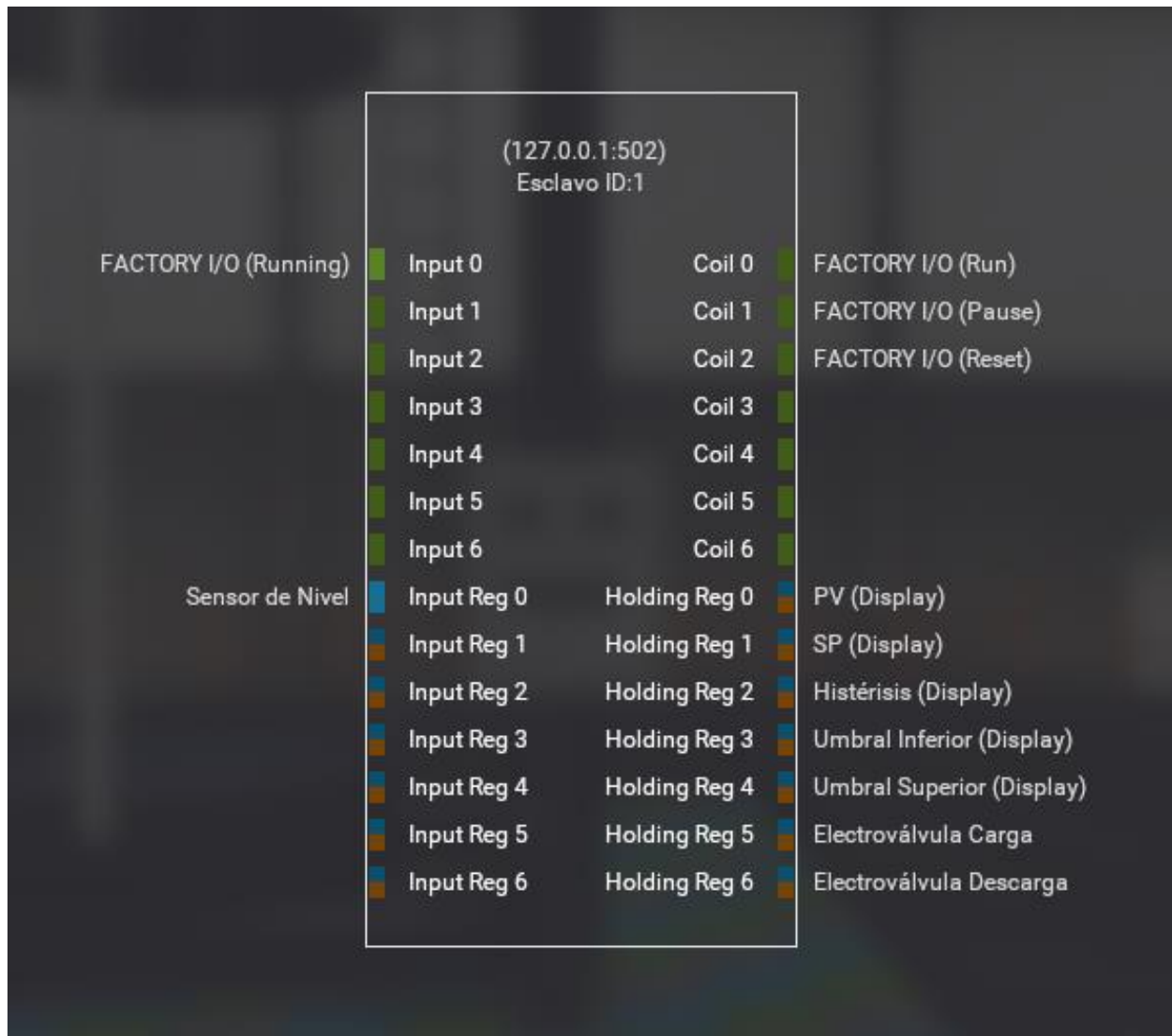
La figura 2-15, muestra la ventana de configuración con una serie de etiquetas enumeradas, las cuales se describen a continuación.

1. Host: Es la dirección IP que tomará el servidor "Modbus TCP/ IP Server". La dirección IP elegida es 127.0.0.1.
2. Puerto: Es el número de puerto local en el servidor Modbus. El puerto predeterminado es 502.
3. Esclavo ID: Este apartado se refiere a la cantidad de esclavos que hay conectados. En este caso hay solamente un esclavo o servidor, que es Factory I/O, por lo tanto, se usa la dirección 1.
4. Adaptador de RED: Se refiere al adaptador de red a utilizar. En este caso como LabVIEW y Factory I/O están instalados en el mismo ordenador se debe elegir "Software Loopback Interface 1". En caso contrario, se debe escoger el adaptador disponible en el ordenador o computador.
5. Escritura Digital: Se debe elegir el bloque de memoria que permitirá escribir o leer los datos de los sensores digitales que se tenga.
6. Lectura Digital: Aquí se debe elegir el bloque de memoria que permitirá leer los valores o datos de los actuadores digitales.
7. Escribir Registro: Se debe elegir el bloque de memoria que permitirá leer los valores o datos de los sensores analógicos que se tenga.
8. Leer Registro: Aquí se debe elegir el bloque de memoria que permitirá leer o escribir los datos de los actuadores analógicos.
9. Escala: Los valores de los sensores flotantes y actuadores se multiplican por él. Ahora, dependiendo el valor de la escala, es que se podrá convertir un número flotante en un valor entero, enviarlo a LabVIEW y volver a convertirlo en un número real dividiéndolo por el factor de la "Escala", por ejemplo, un valor flotante de 2,17 que entrega un sensor analógico (conectado al bloque de memoria "Input Registers") con una escala de 100, se envía a LabVIEW como 217.

10. Terminales I/O: Aquí se define el offset y la cuenta de cada bloque de memoria utilizado en el servidor. El offset, se refiere al desplazamiento de dirección, por ejemplo, si se elige un offset de 3 en el bloque de memoria “Coil”, este bloque comenzará desde la dirección “Coil 3” y no desde la dirección “Coil 0” (como se podrá ver en la figura 2-16). A continuación, se mostrará a detalle cómo se debe leer cada uno de estos bloques.

- ✓ Salidas Bool: Se refiere al bloque de memoria llamado “Coil”. Se coloca el desplazamiento de dirección (igual a 0) y números de salidas digitales (que son 7) a utilizar (máximo 256).
- ✓ Entradas Bool: Se refiere al bloque de memoria llamado “Discrete Input”. Se coloca el desplazamiento de dirección (igual a 0) y números de entradas digitales (igual a 7) a utilizar (máximo 256).
- ✓ Entradas Registro: Es el bloque de memoria llamado “Input Registers”. Se coloca el desplazamiento de dirección (igual a 0) y números de entradas de registro (que son 7) a utilizar (máximo 64).
- ✓ Salidas Registro: Se refiere al bloque de memoria llamado “Holding Registers”. Se coloca el desplazamiento de dirección (igual a 0) y números de salidas de registro (igual a 7) a utilizar (máximo 64).

Dicho lo anterior, por medio de la figura 2-16, se podrá conocer finalmente como queda estructurado el servidor o conjunto de bloques de memorias con sus respectivos elementos de datos a utilizaren la programación de lenguaje G en LabVIEW.



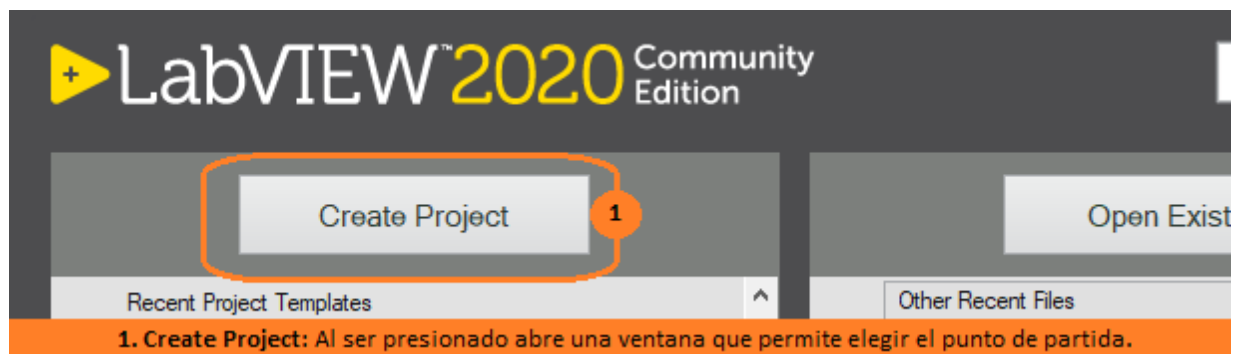
Fuente: Elaboración propia, realizada en Factory I/O.

Figura 2 - 16. Servidor de la escena de nivel.

Tal y como se puede apreciar en la figura 2-16, se usaron todos los elementos de datos de la escena de nivel (ver figura 2-14) menos el “FACTORY I/O (Paused)”, “FACTORY I/O (Reset)”, “FACTORY I/O (Camera Position), ya que se consideran innecesarios.

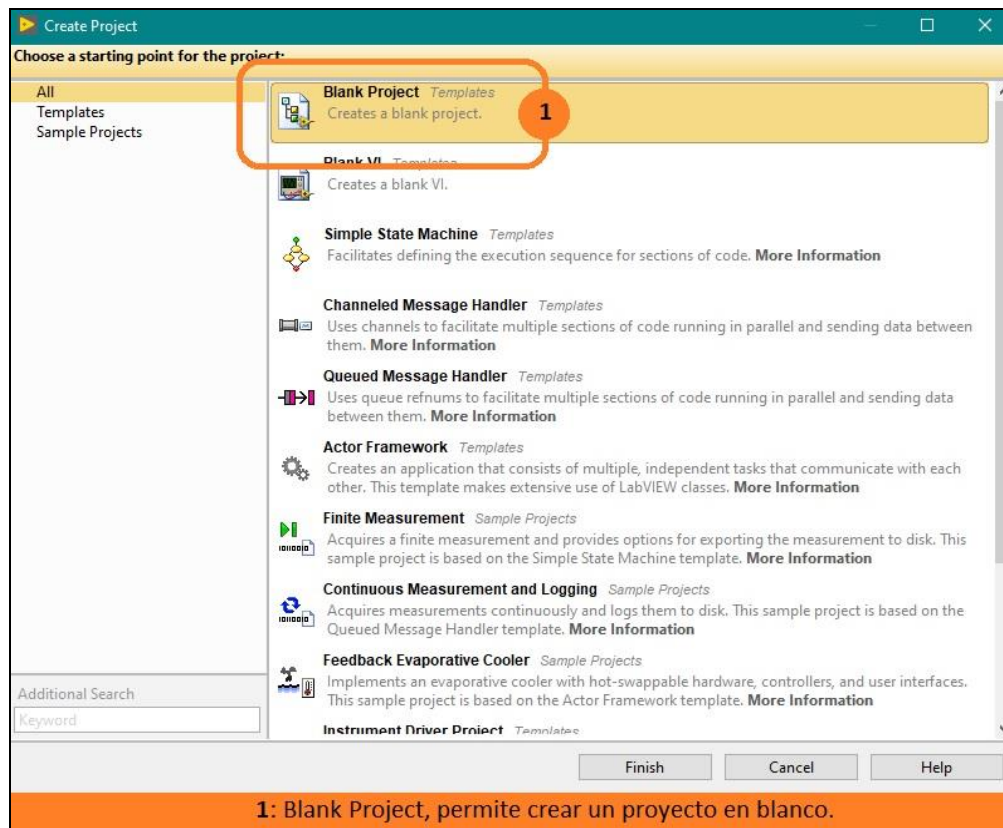
2.2.2. Comunicación en LabVIEW 2020

Para comenzar con los ajustes de parámetros necesarios para permitir la comunicación entre LabVIEW y Factory I/O, se debe previamente descargar e instalar LabVIEW 2020 con sus respectivos toolkit complementarios (LabVIEW Real-Time y LabVIEW and Supervisory Control o DSC) para que aparezca la opción “I/O Server” al momento de crear un nuevo servidor (LabVIEW). Hecho esto, es necesario abrir LabVIEW y seleccionar en la ventana introductora la opción “Create Project”, la cual muestra en una nueva ventana una serie de plantillas como puntos de partida del proyecto, donde se debe elegir la plantilla “Blank Project”, que permite crear finalmente un proyecto en blanco desplegando la ventana exploradora del proyecto. En esta ventana exploradora, se podrá manejar el proyecto, ya que, permite trabajar de una manera más organizada todos los programas o VI (debido al explorador de carpetas y archivos) que se vayan a desarrollar en LabVIEW. Ahora bien, en esta ventana exploradora se debe hacer clic derecho sobre “My Computer” para elegir en el menú “New” la opción “I/O Server”. Esta opción (I/O Server) abrirá un cuadro de diálogo, donde se podrá elegir Modbus (usado en la comunicación de Factory I/O) como el tipo de servidor a crear haciendo clic en “Continue”. Dicho esto, a continuación, por medio de las figuras 2-17, 2-18, 2-19, 2-20, 2-21 y 2-22, se mostrarán todos los pasos necesarios para crear el servidor Modbus.



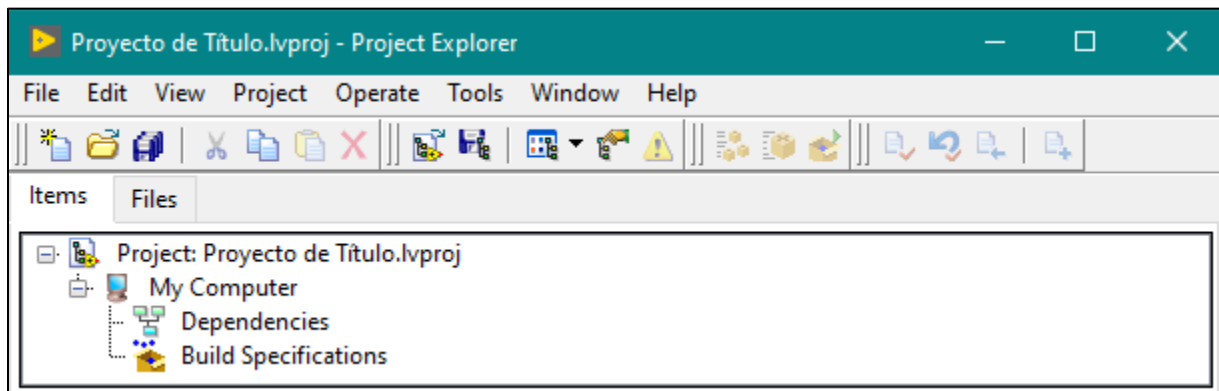
Fuente: Elaboración propia, realizada en LabVIEW 2020 y etiquetada en Paint.

Figura 2 - 17. Paso 1 – Crear proyecto.



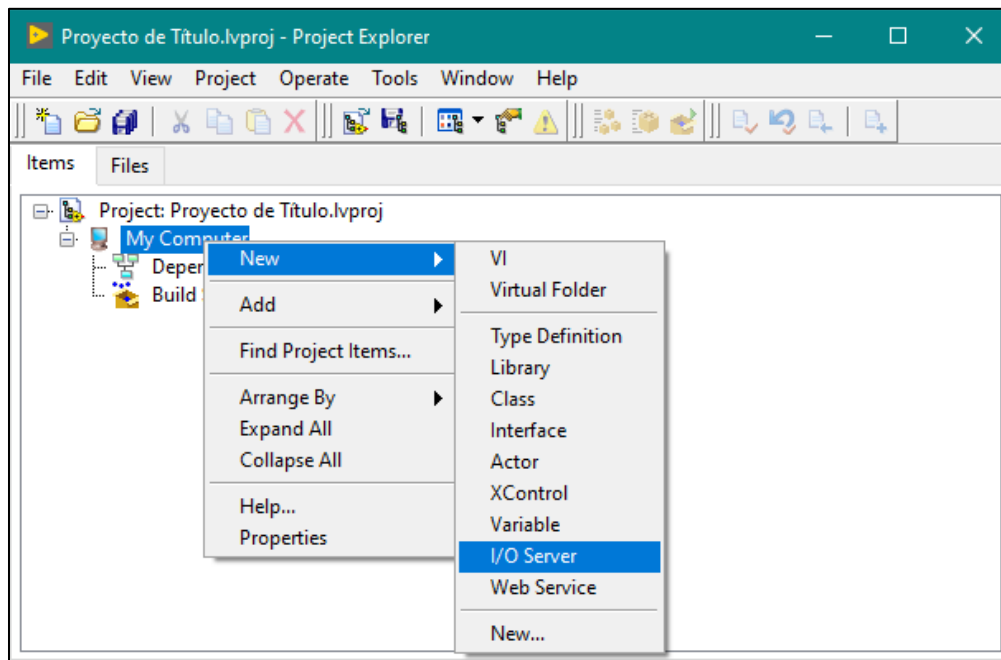
Fuente: Elaboración propia, realizada en LabVIEW 2020 y etiquetada en Paint.

Figura 2 - 18. Paso 2 – Crear un proyecto en blanco.



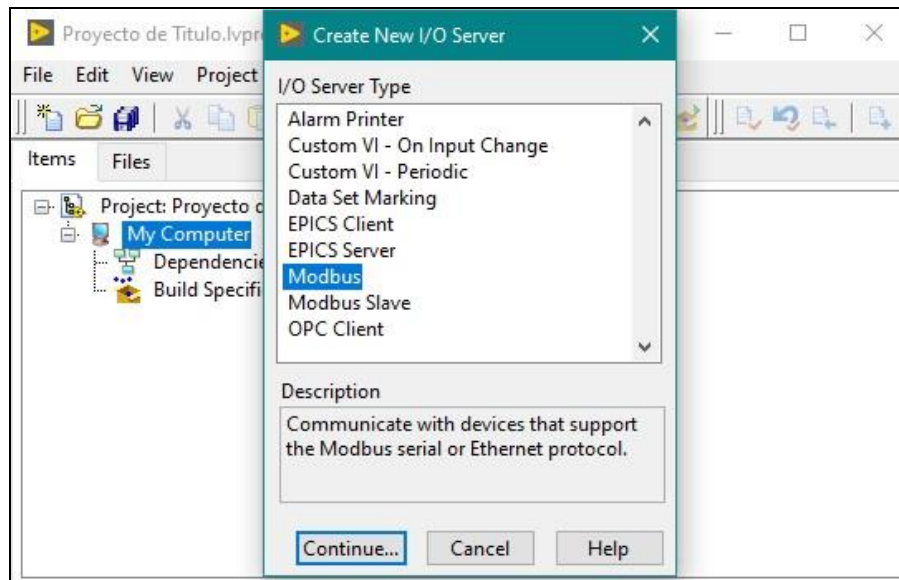
Fuente: Elaboración propia, realizada en LabVIEW 2020.

Figura 2 - 19. Paso 3 - Ventana exploradora del proyecto.



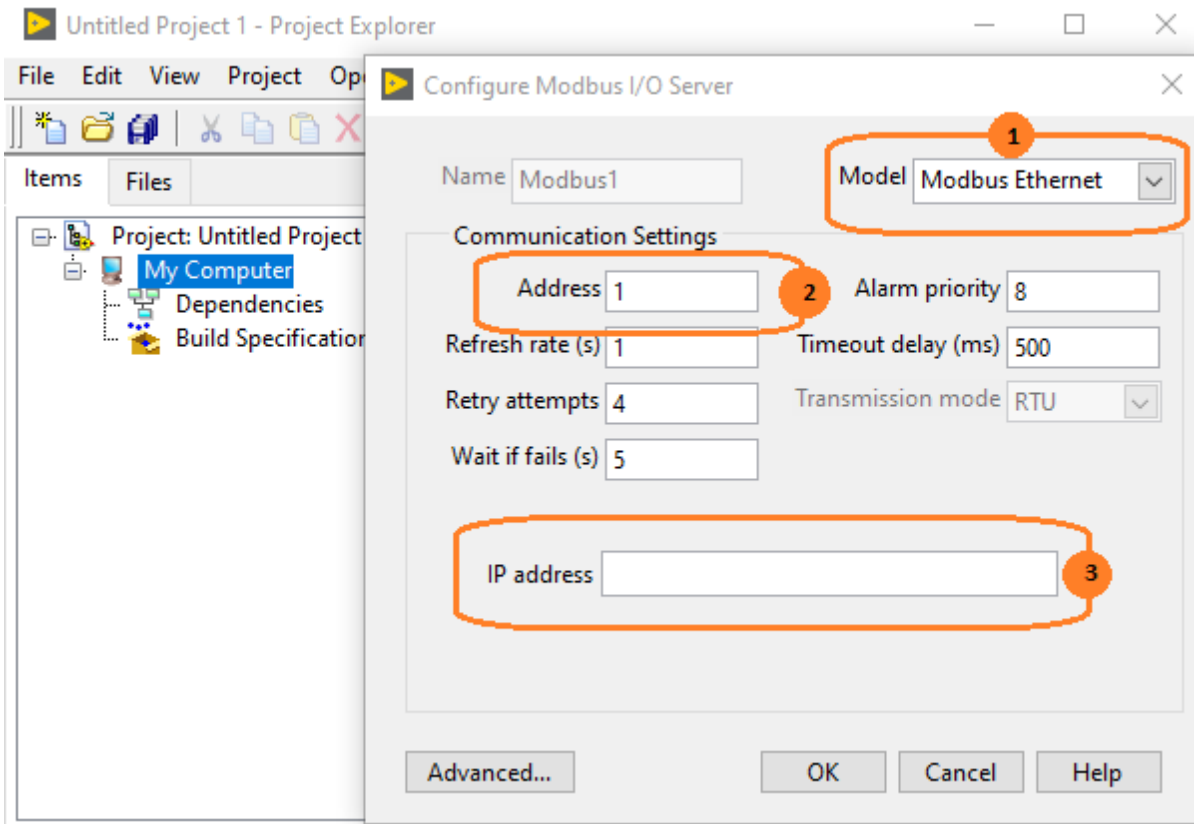
Fuente: Elaboración propia, realizada en LabVIEW 2020.

Figura 2 - 20. Paso 4 - Opción I/O Server.



Fuente: Elaboración propia, realizada en LabVIEW 2020.

Figura 2 - 21. Paso 5 - Crear Servidor Modbus.



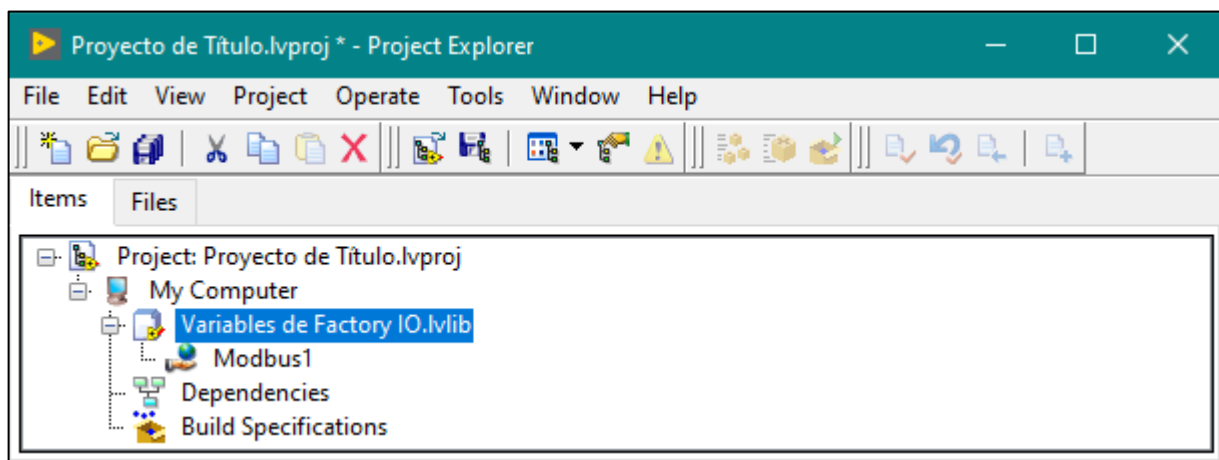
Fuente: Elaboración propia, realizada en LabVIEW 2020 y etiquetada en Paint.

Figura 2 - 22. Paso 6 – Configuración del Servidor Modbus.

Esta figura 2-22, muestra un cuadro de diálogo, donde se pueden agregar algunos parámetros necesarios para dar finalmente con la comunicación deseada. Estos parámetros, son los que se muestran por medio de etiquetas enumeradas, las cuales se describen a continuación.

1. Model: Se refiere a la versión del protocolo Modbus a utilizar, el cual corresponde a Modbus Ethernet, ya que este está relacionado con Modbus TCP/IP.
2. Address: Aquí se debe ingresar la dirección del “Esclavo ID” usado en la configuración en driver del sistema de Factory I/O (ver figura 2-15), es decir, la dirección 1.
3. IP Address: Aquí va el “Host” utilizado en Factory I/O (ver figura 2-15), o sea, 127.0.0.1.

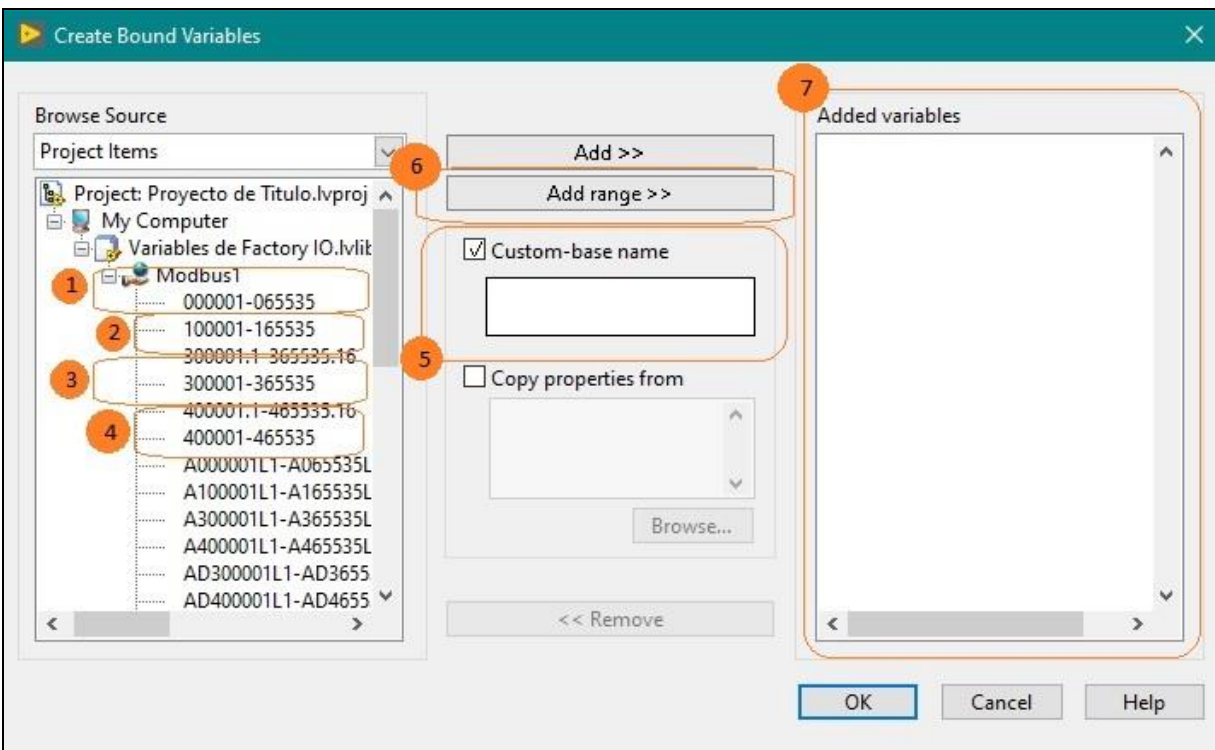
Todos aquellos parámetros que no llevan etiquetas enumeradas, se deben dejar tal cual están. Ahora bien, agregados estos parámetros en el cuadro de diálogo, solo basta con hacer clic en “OK” para que en la ventana exploradora se cree una librería. Esta librería, se apodará “Variables de Factory IO” y es donde se van a crear de alguna u otra manera todas las variables que corresponden en Factory I/O, a elementos de datos añadidos en los bloques de memoria (ver figura 2-16). Dicho esto, es posible a través de la figura 2-23 ver dicha librería creada.



Fuente: Elaboración propia, realizada en LabVIEW 2020.

Figura 2 - 23. Librería de variables.

Finalmente, para crear las variables necesarias en la programación de Lenguaje G, se debe hacer clic derecho sobre la librería creada (Variables de Factory IO) y presionar en “Create Bound Variables” para abrir una nueva ventana. En esta nueva ventana, se debe desplegar todo haciendo clic en el signo “+”, donde se podrá notar en el último despliegue un listado que corresponde básicamente a los rangos de registros que tiene cada bloque de memoria en Modbus (ver tabla 2-1). A continuación, a través de la figura 2-24, se podrá observar la columna correspondiente.



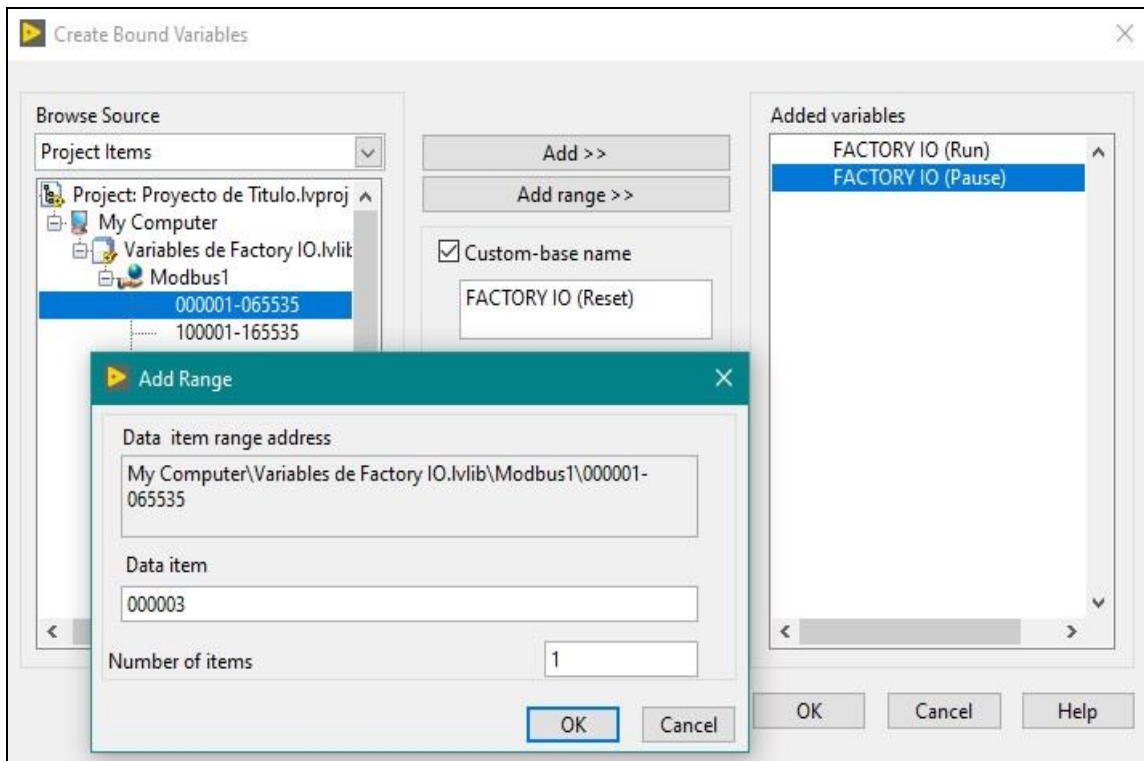
Fuente: Elaboración propia, realizada en LabVIEW 2020 y etiquetada en Paint.

Figura 2 - 24. Rangos de registros.

Las etiquetas enumeradas en esta figura 2-24, se detallan a continuación.

- ✓ 000001 – 065535: Corresponde al rango de registro del bloque de memoria del servidor Modbus llamado “Coil” o “Bobina”.
- ✓ 100001 – 165535: Corresponde al rango de registro del bloque de memoria del servidor Modbus llamado “Discrete Input” o “Entrada Discreta”.
- ✓ 300001 – 365535: Corresponde al rango de registro del bloque de memoria del servidor Modbus llamado “Input Register” o “Registro de Entrada”.
- ✓ 400001 – 465535: Corresponde al rango de registro del bloque de memoria del servidor Modbus llamado “Holding Register” o “Registro de Retención”.

- ✓ Custom base name: Aquí va el nombre que tendrá el elemento de datos de Factory I/O en LabVIEW, donde se recomienda colocar el mismo nombre que tiene en Factory I/O.
- ✓ Add range: Al presionar este recuadro, abre una ventana, la cual permite agregar el número del rango de registro que se lleva. Por ejemplo, si se desea agregar la primera "Coil", en "Data ítem" de esta nueva ventana se debe colocar la dirección "000001", si fuera la segunda "Coil" a agregar, se colocaría "000002" y así sucesivamente. En "number of ítem" se debe dejar tal cual, o sea, en 1 bit. Esto, se puede observar en la figura 2-25.

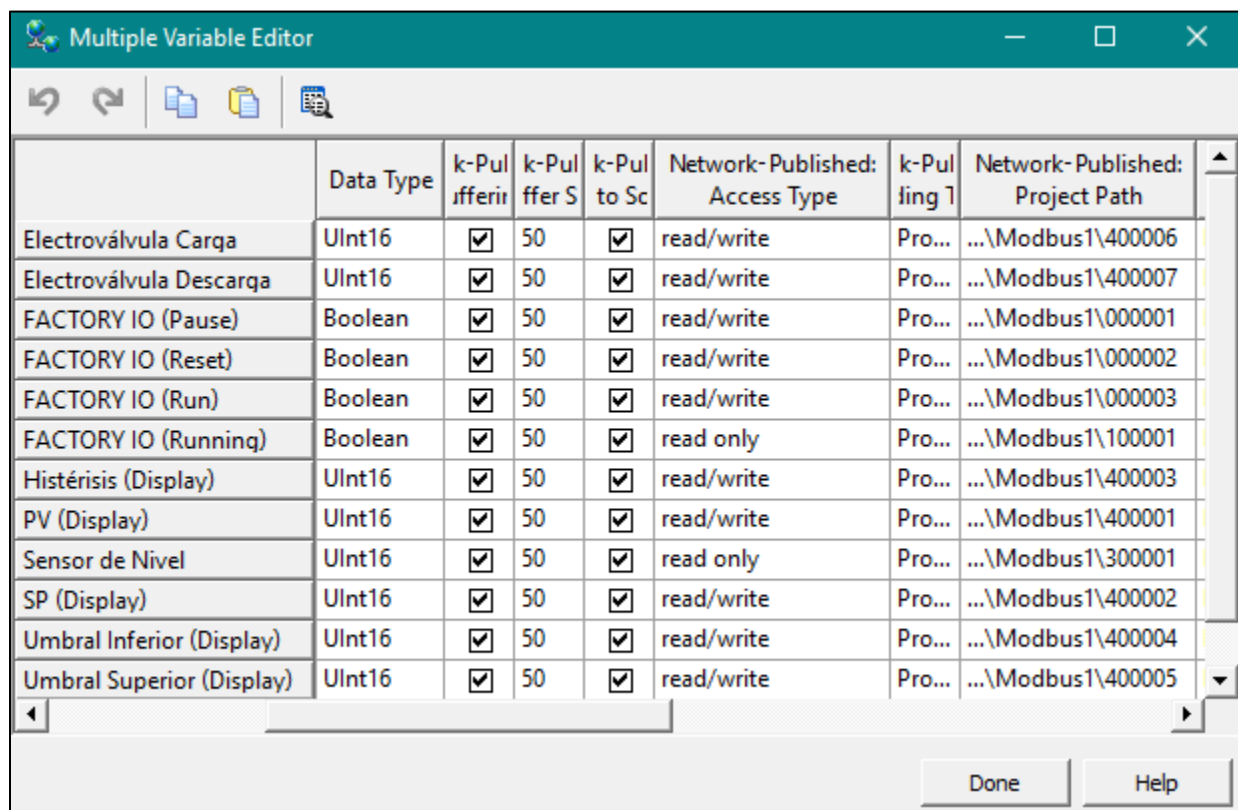


Fuente: Elaboración propia, realizada en LabVIEW 2020.

Figura 2 - 25. Agregar rango de variable.

- ✓ Added variables: Aquí van todas las variables o elementos de datos agregados.

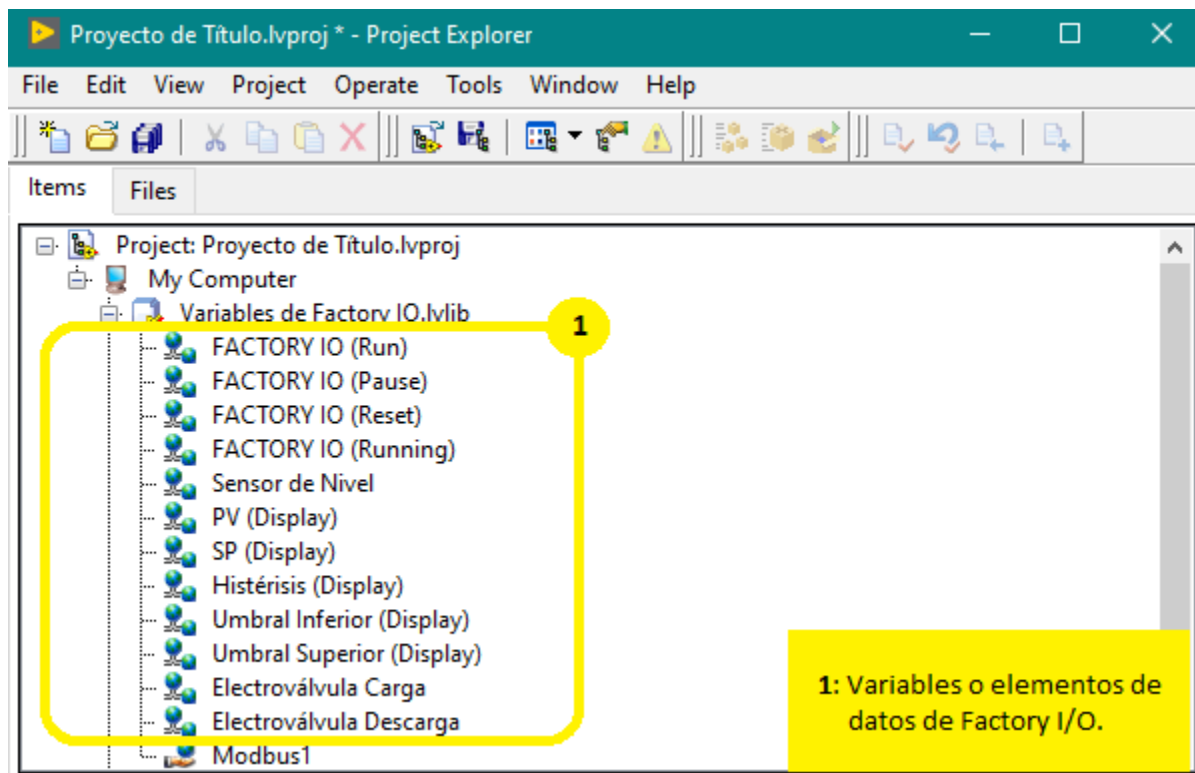
Una vez agregadas todas estas variables o elementos de datos necesarios, se debe pulsar “OK”. Pulsado “OK”, aparecerá la ventana “Editor de Variables Múltiples” con todas las variables agregadas, donde se podrá revisar, realizar la configuración y edición de parámetros de acceso y de datos de las mismas. Ahora, solo queda presionar el botón “Done” para que la configuración de la comunicación Modbus entre LabVIEW y Factory I/O esté completa. Esta ventana o editor de variables múltiples, se puede observar en la figura 2-26.



Fuente: Elaboración propia, realizada en LabVIEW 2020.

Figura 2 - 26. Editor de múltiples variables.

Ahora se podrá visualizar todas las variables o elementos de datos agregados, en la librería llamada “Variables de Factory I/O”, tal y como, se muestra en la figura 2-27.

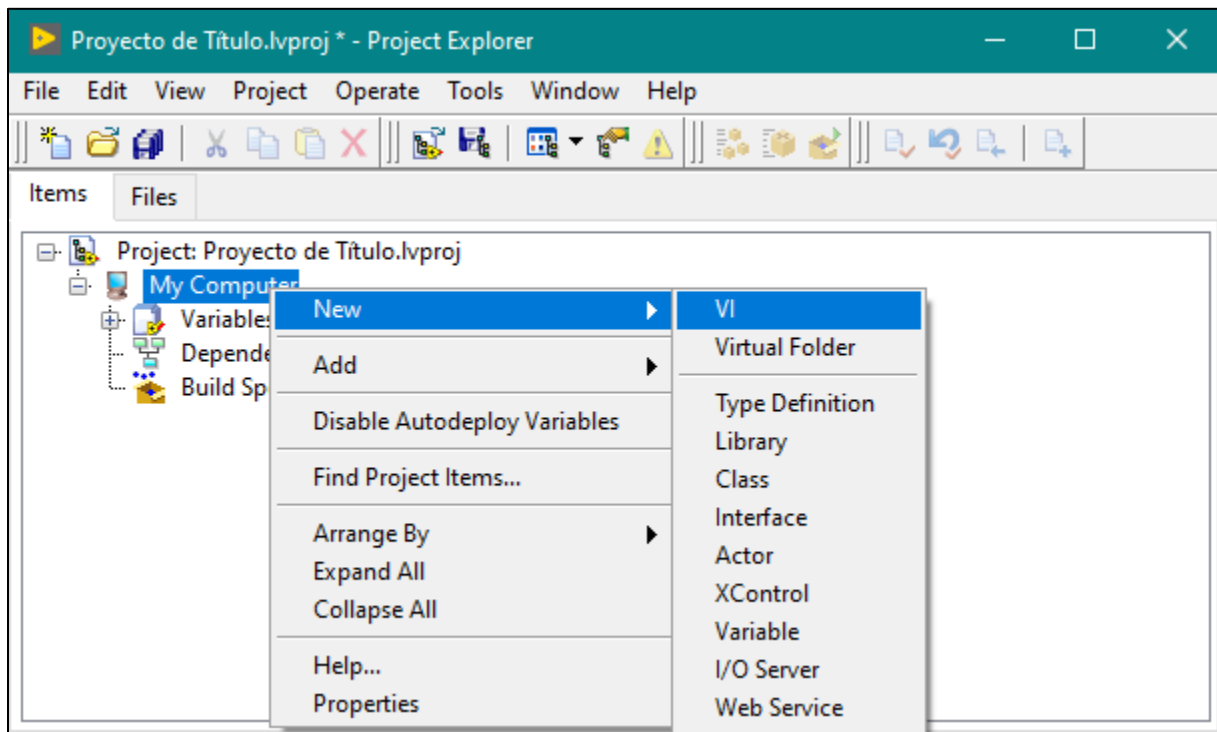


Fuente: Elaboración propia, realizada en LabVIEW 2020 y etiquetada en Paint.

Figura 2 - 27. Elementos de datos o variables en Librería de Modbus.

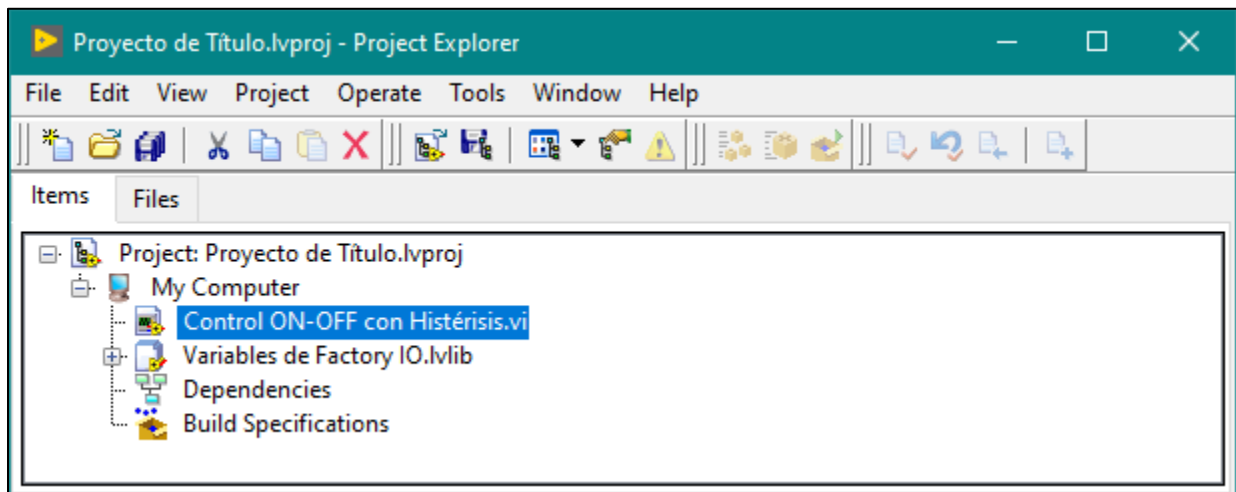
2.3. PROGRAMACIÓN

En este apartado, se verá la programación necesaria en lenguaje G para lograr controlar y monitorear la escena de nivel, a través de LabVIEW. Esta programación, se va a dividir en 2 partes (diagrama de bloques y panel frontal) para facilitar su comprensión, sin embargo, antes se debe crear un nuevo VI. Para ello, se debe hacer clic derecho en “My Computer” de la ventana exploradora y seleccionar en el menú “New” la opción “VI”, tal y como se muestra en la figura 2-28. Hecho esto, se podrá apreciar un nuevo VI en la carpeta “My Computer”, el cual se llamará “Control ON-OFF con Histéresis”. Este VI creado, se puede observar en la figura 2-29.



Fuente: Elaboración propia, realizada en LabVIEW 2020.

Figura 2 - 28. Nuevo VI.

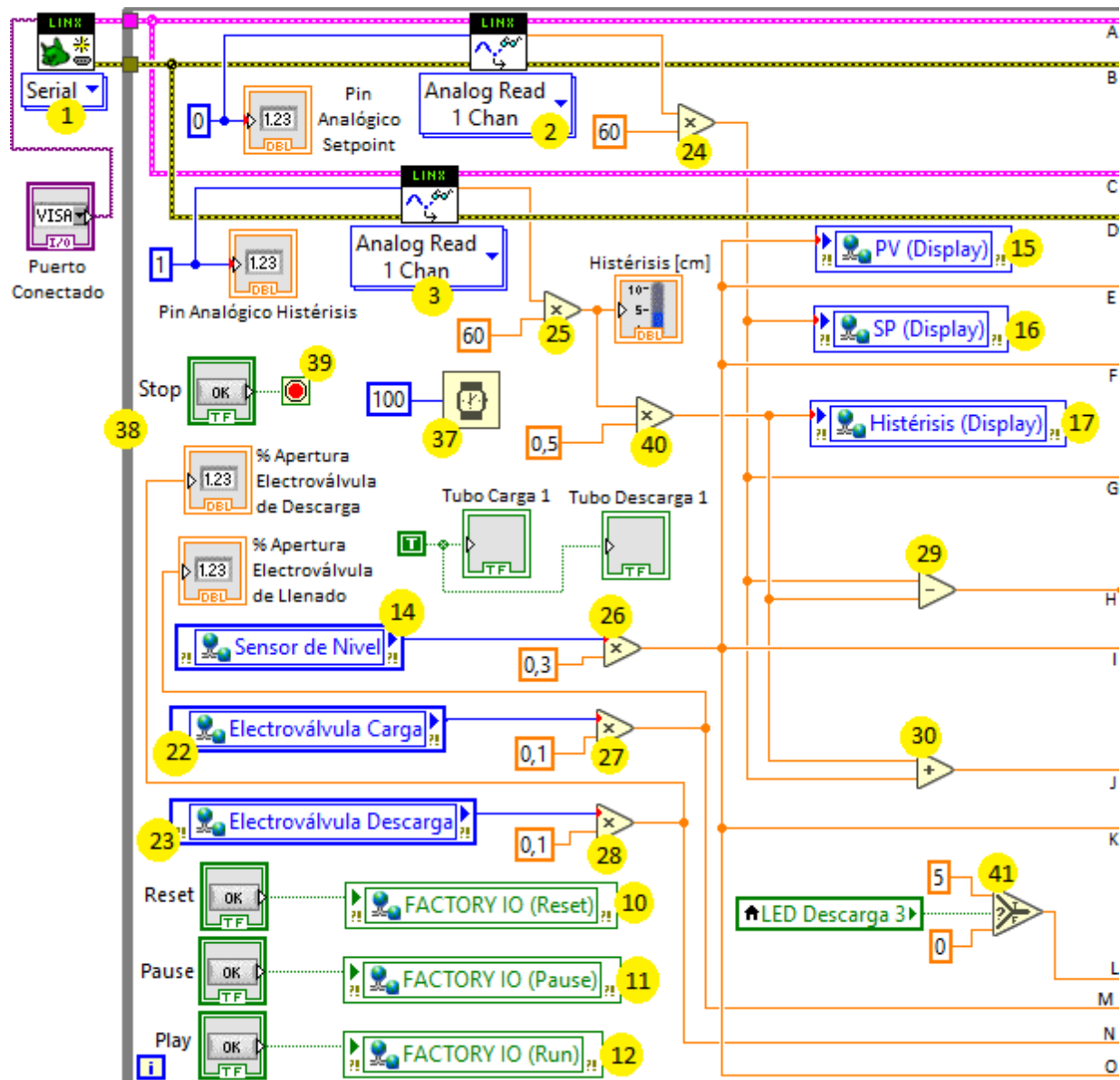


Fuente: Elaboración propia, realizada en LabVIEW 2020.

Figura 2 - 29. VI creado.

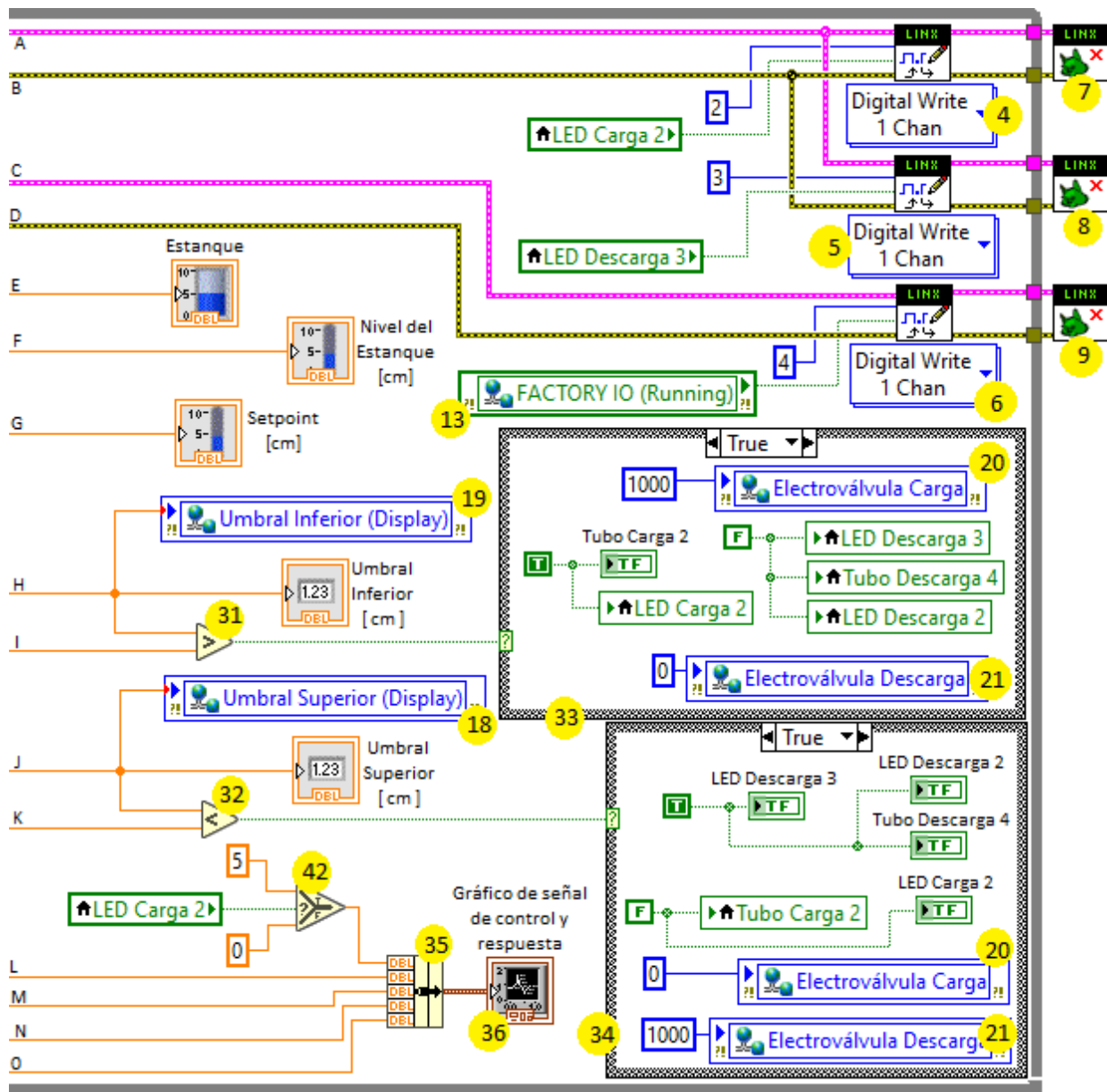
2.3.1. Diagrama de bloques

En este apartado, se verá el diagrama de bloques del proyecto dividido en dos partes, las cuales se pueden apreciar por medio de las figuras 2-30 y 2-31, mostradas a continuación.



Fuente: Elaboración propia, realizada en LabVIEW 2020 y etiquetada en Paint.

Figura 2 - 30. Diagrama de bloques del proyecto – Parte 1.

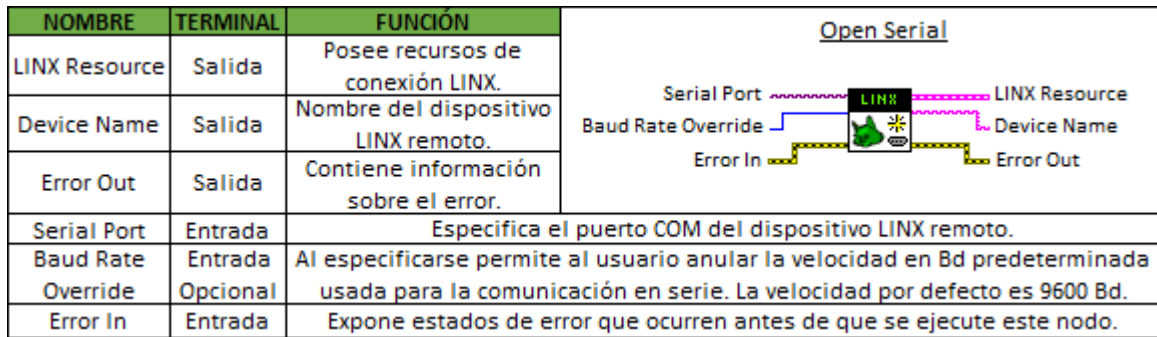


Fuente: Elaboración propia, realizada en LabVIEW 2020 y etiquetada en Paint.

Figura 2 - 31. Diagrama de bloques del proyecto – Parte 2.

Las figuras 2-30 y 2-31, muestran una gran variedad de bloques, variables, funciones, estructura, etc, los cuales, en su conjunto ayudarán a controlar y monitorear la escena de nivel que entrega Factory I/O. Estos, se darán a conocer de manera detallada a continuación.

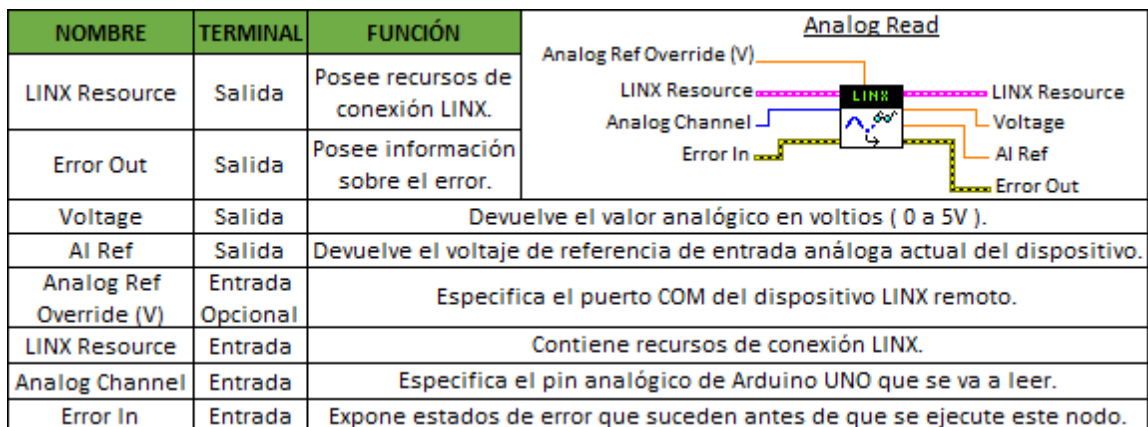
- ✓ Open Serial: Bloque que abre una conexión en serie a un dispositivo LINX remoto. Este bloque se puede conocer a través de la figura 2-32.



Fuente: Elaboración propia, realizada en Microsoft Excel.

Figura 2 - 32. Bloque “Open Serial”.

- ✓ Analog Read: Es un bloque que lee el valor del pin de entrada analógica especificado. Este bloque, se puede apreciar en la figura 2-33.



Fuente: Elaboración propia, realizada en Microsoft Excel.

Figura 2 - 33. Bloque “Analog Read”.

- ✓ Digital Write: Es un bloque que escribe un valor en el pin de salida digital especificado. Este bloque, se podrá visualizar en la figura 2-34.

NOMBRE	TERMINAL	FUNCIÓN	Digital Write
LINX Resource	Salida	Posee recursos de conexión LINX.	
Error Out	Salida	Posee información sobre el error.	
LINX Resource	Entrada	Contiene recursos de conexión LINX.	
Do Channels	Entrada	Especifica los pines de salida digital en los que se escribirán.	
Output Value	Entrada	Especifica los valores digitales a escribir.	
Error In	Entrada	Describe las condiciones de error que ocurren antes de que se ejecute este nodo.	

Fuente: Elaboración propia, realizada en Microsoft Excel.

Figura 2 - 34. Bloque “Digital Write”.

- ✓ Close: Es el bloque que cierra la conexión con el dispositivo LINX remoto. A continuación, a través de la figura 2-35, se podrá observar dicho bloque.

NOMBRE	TERMINAL	FUNCIÓN	Close LINX Resource Error In Error Out LINX 
Error Out	Salida	Posee información sobre el error.	
LINX Resource	Entrada	Posee recursos de conexión LINX.	
Error In	Entrada	Expone estados de error que ocurren antes de que se ejecute este nodo.	

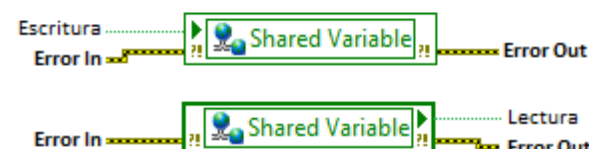
Fuente: Elaboración propia, realizada en Microsoft Excel.

Figura 2 - 35. Bloque “Close”.

- ✓ Shared Variable: Representa una variable compartida en el diagrama de bloques. Esta variable se puede observar en la figura 2-36, mostrada a continuación.

NOMBRE	TERMINAL	FUNCIÓN
Escritura	Entrada	Funciona como entrada, es decir, se puede escribir el valor que se desee en la variable.
Error In	Entrada	Describe las condiciones de error que ocurren antes de que se ejecute este nodo.
Lectura	Salida	Funciona como salida, o sea, se puede leer la información que entregue la variable.
Error Out	Salida	Contiene información de error

Shared Variable



The diagram illustrates the Shared Variable block in two configurations. In the top configuration, the 'Escritura' (Write) terminal is connected to the 'Error In' terminal, and the 'Error Out' terminal is connected to the 'Lectura' (Read) terminal. In the bottom configuration, the 'Error In' terminal is connected to the 'Error Out' terminal, and the 'Lectura' (Read) terminal is connected to the 'Escritura' (Write) terminal. Both blocks are labeled 'Shared Variable' and have a green border.

Fuente: Elaboración propia, realizada en Microsoft Excel.

Figura 2 - 36. Shared Variable.

Una vez conocido cada uno de estos bloques, variables, funciones y estructuras, es posible entender el uso o la función que tuvieron en la programación de lenguaje G del proyecto. Es por esto, que en las figuras 2-30 y 2-31, se pueden observar una gran variedad de etiquetas enumeradas, las cuales se describirán de manera detallada, a continuación.

1. Este bloque de LINX llamado "Open Serial" (Ver figura 2-32), se usó para abrir la conexión en serie con Arduino UNO.
2. Bloque de LINX llamado "Analog Read" (Ver figura 2-33), se utilizó para especificar la entrada analógica física de Arduino UNO, que ocupará el potenciómetro encargado de fijar el setpoint. La entrada analógica, se debe especificar en el terminal de entrada llamado "Analog Channel".

3. Este bloque de LINX llamado “Analog Read” (Ver figura 2-33), se usó para especificar la entrada analógica de Arduino UNO, que ocupará el potenciómetro encargado de fijar la histéresis. La entrada analógica, se debe especificar en el terminal de entrada llamado “Analog Channel”.
4. Se refiere al bloque de LINX llamado “Digital Write” (Ver figura 2-34), que se utilizó para especificar la salida digital (de Arduino UNO), que ocupará el LED verde físico, encargado de encender cuando la electroválvula de carga de la planta en Factory I/O, entra en funcionamiento. La salida digital, se debe especificar en el terminal de entrada llamado “DO Channels”.
5. Este bloque de LINX llamado “Digital Write” (Ver figura 2-34), se utilizó para especificar la salida digital de Arduino UNO, donde irá finalmente conectado el LED verde físico, encargado de encender cuando la electroválvula de descarga de la planta en Factory I/O, entra en funcionamiento. La salida digital, se debe especificar en el terminal de entrada llamado “DO Channels”.
6. Bloque de LINX llamado “Digital Write” (Ver figura 2-34), se utilizó para especificar la salida digital de Arduino UNO, que ocupará el LED rojo físico, encargado de encender cuando la planta en Factory I/O, entra en funcionamiento. La salida digital, se debe especificar en el terminal de entrada llamado “DO Channels”.
7. Es el bloque de LINX llamado “Close” (Ver figura 2-35), el cual se utilizó para cerrar la conexión en serie necesaria para que Arduino UNO active su entrada analógica “A0”, referida al setpoint (Ver tag 2) y su salida digital (Pin 2), referida a la electroválvula de carga (tag 4).
8. Bloque de LINX llamado “Close” (Ver figura 2-35), se usó para cerrar la conexión en serie necesaria para que Arduino UNO, active la salida digital correspondiente al pin 2, referida a la electroválvula de descarga (Ver tag 5).
9. Este bloque de LINX llamado “Close” (Ver figura 2-35), se usó para cerrar la conexión en serie necesaria para que Arduino UNO active su entrada analógica “A1”, referida a la histéresis (Ver tag o etiqueta número 3) y active su salida digital (pin 4), referida al funcionamiento de la planta (Ver tag 6).

10. Es la variable de Factory I/O, llamada "Factory IO (Reset)" (Ver figura 2-14 y 2-36), la cual fue utilizada en modo escritura.
11. Se refiere a la variable de Factory I/O, llamada "Factory IO (Pause)" (Ver figura 2-14 y 2-36), que se utilizó en modo escritura.
12. Es la variable de Factory I/O, llamada "Factory IO (Run)" (Ver figura 2-14 y 2-36), la cual se usó en modo escritura.
13. Se refiere a la variable de Factory I/O, llamada "FACTORY IO Running" (Ver figura 2-14 y 2-36), que se utilizó en modo lectura.
14. Es la variable de Factory I/O, llamada "Sensor de Nivel" (Ver figura 2-14 y 2-36), la cual se usó en modo lectura.
15. Es la variable de Factory I/O, llamada "PV (Display)" (Ver figura 2-14 y 2-36), que se utilizó en modo escritura.
16. Se refiere a la variable de Factory I/O, llamada "SP (Display)" (Ver figura 2-14 y 2-36), la cual se usó en modo escritura.
17. Es la variable de Factory I/O, llamada "Histéresis (Display)" (Ver figura 2-14 y 2-36), que se utilizó en modo escritura.
18. Se refiere a la variable o elemento de dato de Factory I/O, llamado "Umbral Superior (Display)" (Ver figura 2-14 y 2-36), que se utilizó en modo escritura.
19. Se refiere a la variable o elemento de dato de Factory I/O, llamada "Umbral Inferior (Display)" (Ver figura 2-14 y 2-36), la cual se usó en modo escritura.
20. Se refiere a la variable de Factory I/O, llamada "Electroválvula Carga" (Ver figura 2-14 y 2-36), la cual se usó como escritura.
21. Es la variable de Factory I/O, llamada "Electroválvula Descarga" (Ver figura 2-14 y 2-36), que se utilizó en modo escritura.
22. Se refiere a la misma variable de la etiqueta o tag número 20, pero con la diferencia que esta vez se encuentra en modo lectura.
23. Es la misma variable de la etiqueta número 21, pero con la diferencia que esta vez se encuentra en modo lectura.

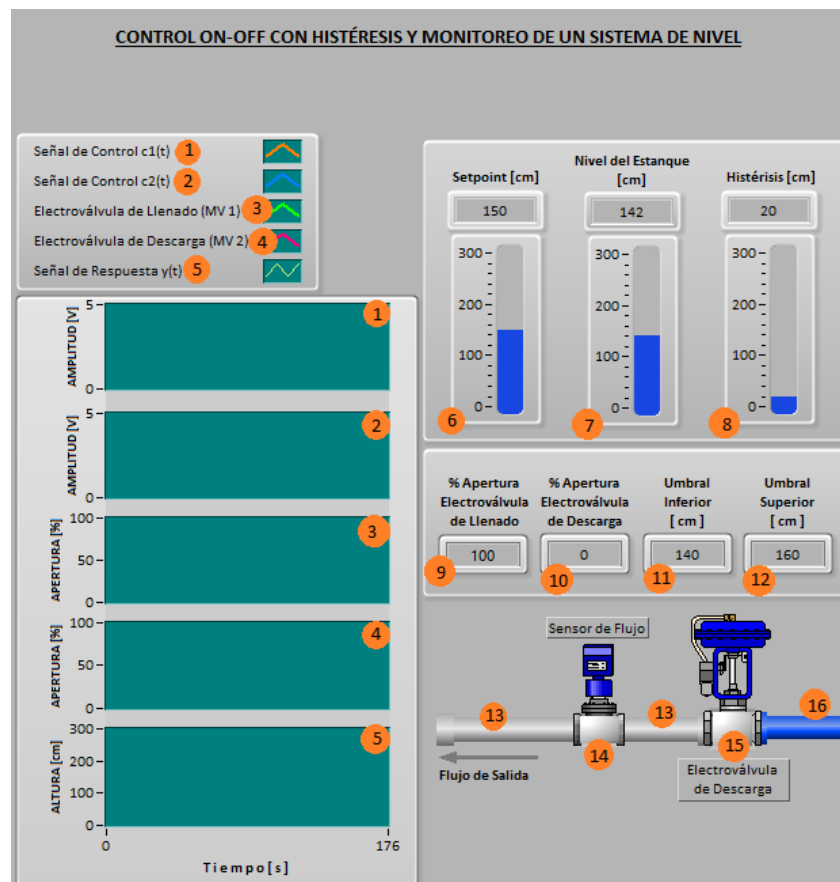
24. Esta función, llamada "Multiply", se utilizó para escalar la salida que entrega el bloque llamado "Analog Read" referido al setpoint (Ver tag 2). La salida que entrega dicho bloque va de 0 a 5, sin embargo, para utilizar el rango del setpoint correctamente debe ir de 0 a 300. Es por esto, que se multiplica por 60 ($5 * 60 = 300$).
25. Esta función llamada "Multiply", se usó para escalar la salida que entrega el bloque llamado "Analog Read", referido a la histéresis (Ver tag 3). La salida que entrega dicho bloque, va de 0 a 5, pero, para utilizar el rango de la histéresis correctamente, debe ir de 0 a 300. Es por esto, que la salida se multiplica por 60 ($5 * 60 = 300$).
26. Esta función "Multiply", se utilizó para escalar la salida que entrega la variable de Factory I/O, llamada "Sensor de Nivel" (Ver tag 14). La salida que entrega dicha variable va de 0 a 1000, sin embargo, para apreciar correctamente el comportamiento del nivel del líquido contenido en el estanque, debe ir de 0 a 300. Es por esto, que se multiplica por 0,3 ($1000 * 0,3 = 300$).
27. Se refiere a la función "Multiply", se utilizó para escalar la salida que entrega la variable de Factory I/O, llamada "Electroválvula Carga" (Ver tag 22). La salida que entrega dicha variable va de 0 a 1000, sin embargo, para apreciar correctamente el porcentaje de apertura (0 o 100%) de dicha electroválvula en la gráfica del panel frontal, se debe multiplicar por 0,1 ($1000 * 0,1 = 100$).
28. Esta función "Multiply", se utilizó para escalar la salida que entrega la variable de Factory I/O, llamada "Electroválvula Descarga" (Ver tag 23). La salida que entrega dicha variable va de 0 a 1000, sin embargo, para apreciar correctamente el porcentaje de apertura (0 o 100%) de dicha electroválvula en la gráfica del panel frontal, se debe multiplicar por 0,1 ($1000 * 0,1 = 100$).
29. Se refiere a la función llamada "Subtract", la cual al valor de setpoint escalado, le resta el valor de la histéresis escalada, para obtener el umbral inferior (Setpoint escalado – Histéresis escalada = Umbral Inferior).
30. Esta función "Add", le suma al valor de setpoint (escalado) el valor de histéresis (escalado), para obtener el valor de umbral superior (Setpoint escalado + Histéresis escalada = Umbral Superior).

31. Se refiere a la función llamada "Greater ?". Esta función, realiza la siguiente pregunta: ¿El valor de umbral inferior es mayor que el nivel del líquido contenido en el estanque?. Si fuera cierto lo anterior, entregaría un "True" a la salida y activaría el selector label "True" de la estructura "Case", referida a la electroválvula de carga (Ver tag 33). De lo contrario entrega un "False" y no activa nada
32. Es la función llamada "Lees ?". Esta función, realiza la siguiente pregunta: ¿El valor de umbral superior es menor que el nivel del líquido contenido en el estanque?. Si fuera cierto lo anterior, entregaría un "True" a la salida y activaría el selector label "True" de la estructura "Case", referida a la electroválvula de descarga (Ver tag 34). De lo contrario entrega un "False" y no activa nada.
33. Se refiere a la estructura llamada "Case" con el "selector label" en "True". Esta estructura, es activada con la ayuda de la función "Greater ?". En esta estructura, se podrá encontrar todo lo necesario para activar la electroválvula de carga y desactivar la electroválvula de descarga, es decir, se realiza la apertura por completo de la electroválvula de carga (ubicada en la planta de Factory I/O), mediante una constante igual a 1000 asignada a la entrada de la variable "Electroválvula Carga" y una constante igual a 0 asignada a la entrada de la variable llamada "Electroválvula Descarga.
34. Es la estructura "Case" con "selector label" en "True", la cual es activada por la función "Lees ?". Esta estructura, contiene todo lo necesario para activar la electroválvula de descarga y desactivar la electroválvula de carga, es decir, realiza la apertura por completo de la electroválvula de descarga (ubicada en la planta de Factory I/O), mediante una constante igual a 1000, asignada a la entrada de la variable "Electroválvula Descarga" y una constante igual a 0, asignada a la entrada de la variable llamada "Electroválvula Carga.
35. Es la función llamada "Bundle" y agrupa los datos de la actuación $c1(t)$, la actuación $c2(t)$, de la variable "Electroválvula de Carga" escalada (para representar MV 1), de la variable llamada "Electroválvula de Descarga" escalada (para representar MV 2) y de la variable llamada "Sensor de Nivel" escalada (para mostrar $y(t)$). La finalidad de agrupar estos datos, es mostrarlos luego en el gráfico llamado "Charts" (Ver tag 36).

36. Charts, es un gráfico de forma de onda, que muestra uno o más gráficos de datos adquiridos normalmente a una velocidad constante. Posteriormente, en el panel frontal de LabVIEW, se podrá apreciar este gráfico mostrando todas las gráficas mencionadas en la etiqueta número 35.
37. Es la función llamada "Waits", la cual permitirá que el subdiagrama de la estructura "While Loop" (Ver tag 38 de la figura 2-30), espere 100 milisegundo para que vuelva a ejecutarse.
38. Se refiere a la estructura llamada "While Loop", la cual permitirá ejecutar el código de programación en lenguaje G del proyecto repetidamente hasta que se cumpla la "Conditional Terminal" (Ver tag 39).
39. Es la "Conditional Terminal" de la estructura "While Loop" (Ver tag 38 de la figura 2-30), la cual mediante un "True" proveniente del botón "Stop" permitirá detener el ciclo o bucle While del proyecto.
40. Se refiere a la función llamada "Multiply", la cual se utilizó para dividir el valor de histéresis en 2. Por ejemplo, si el valor de histéresis ya escalada (ver tag 25) fuera de 20, esta función lo que haría es bajar este valor a la mitad para luego con la ayuda de las funciones llamadas "Subtract" y "Add" asignar el valor de umbral superior e inferior, ya que "setpoint + (0,5*histéresis)" es igual a umbral superior y "setpoint - (0,5*histéresis)" es igual a umbral inferior.
41. Es la función llamada "Select", referida a la señal de control $c2(t)$ (electroválvula de descarga) que luego se mostrará en la gráfica (ver tag 36). Si a la entrada "s" de esta función, hay un "True", muestra a la salida un 5 (para referirse a los 5V que podría entregar Arduino UNO). Por otro lado, si a la entrada "s" hay un "False", mostrará a la salida un 0 (para referirse a los 0V que podría entregar Arduino UNO).
42. Es la función llamada "Select", referida a la señal de control $c1(t)$ (electroválvula de carga o llenado) que luego se mostrará en la gráfica (ver tag 36). Si a la entrada "s" de esta función, hay un "True", muestra a la salida un 5 (para referirse a los 5V que podría entregar Arduino UNO). Por otro lado, si a la entrada "s" hay un "False", mostrará a la salida un 0 (para referirse a los 0V que podría entregar Arduino UNO).

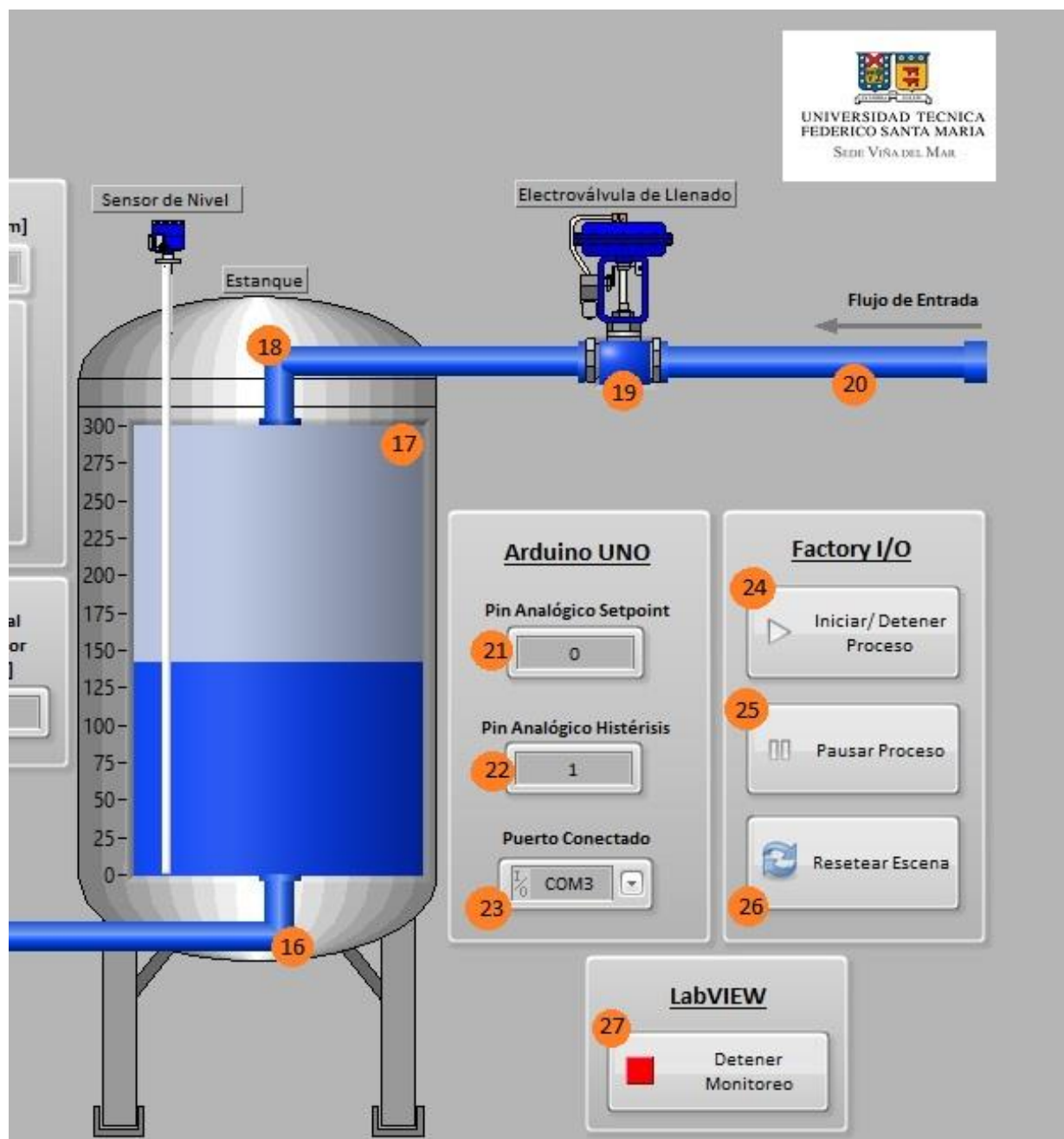
2.3.2. Panel Frontal

Finalmente, será ésta la pantalla que verá el operador de la planta o escena de nivel que entrega Factory I/O, la cual permitirá monitorear el comportamiento de las señales de control ($c_1(t)$ y $c_2(t)$), de la señal de respuesta ($y(t)$) y de las variables manipuladas (MV 1 y MV 2). El operador de la planta, también podrá visualizar el estado de ambas electroválvulas, el valor del setpoint e histéresis mediante indicadores, el valor de umbral superior e inferior, el estado del sensor de flujo y el nivel por medio de indicadores. Este panel frontal del proyecto, se puede observar de manera detallada, a través de las figuras 2-37y 2-38, mostradas a continuación.



Fuente: Elaboración propia, realizada en LabVIEW 2020 y etiquetada en Paint.

Figura 2 - 37. Panel Frontal de LabVIEW – Primera parte.



Fuente: Elaboración propia realizada en LabVIEW 2020 y etiquetada en Paint.

Figura 2 - 38. Panel Frontal de LabVIEW – Segunda parte.

En las figuras 2-37 y 2-38, es posible encontrar algunas etiquetas enumeradas, las cuales se describen de manera detallada a continuación.

1. Gráfico 1: Este gráfico muestra la señal de control $c1(t)$, la cual se genera a la salida del controlador (Arduino UNO) para poder activar la electroválvula de carga. En el eje de las ordenadas de este gráfico, irá la amplitud expresada en términos de voltaje y en el eje de las abscisas el tiempo, expresado en segundos. El máximo valor que puede tomar la amplitud son 5V y el mínimo son 0V. Para lograr que este gráfico se posicione en la primera posición, es necesario que la salida de la función llamada "Select" (Ver tag 42 de la figura 2-31), se conecte al elemento 0 de la función "Bundle".
2. Gráfico 2: Este gráfico muestra la señal de control $c2(t)$, la cual es generada por el controlador (Arduino UNO) para lograr activar la electroválvula de descarga. En el eje de las ordenadas de este gráfico, irá la amplitud expresada en términos de voltaje y en el eje de las abscisas el tiempo, expresado en segundos. El máximo valor que puede tomar la amplitud son 5V y el mínimo son 0V. Para lograr que este gráfico se posicione en la segunda posición, es necesario que la salida de la función llamada "Select" (Ver tag 41 de la figura 2-30), se conecte al elemento 1 de la función "Bundle".
3. Gráfico 3: Aquí se grafica el comportamiento que va teniendo la variable manipulada (MV 1) en el tiempo, es decir, se podrá ver el porcentaje de apertura, la activación y desactivación de la electroválvula de carga. Este proyecto, se basa en un control "ON/OFF" o "Todo/Nada", lo que significa que esta electroválvula permitirá todo el flujo o nada a través de su cuerpo. Por lo tanto, para representar lo anterior en el eje de las ordenadas de este gráfico, irá la apertura expresada en porcentaje y en el eje de las abscisas el tiempo, expresado en segundos. El máximo valor que puede tomar la apertura de esta electroválvula es de 100 % y el mínimo es 0 %. Para lograr que este gráfico se posicione en la tercera posición, es necesario que la variable llamada "Electroválvula Carga" ya escalada en el diagrama de bloque (Ver tag 27 de la figura 2-30), se conecte al elemento 2 de la función "Bundle".

4. Gráfico 4: Se verá el comportamiento que va teniendo la variable manipulada (MV 2) en el tiempo, es decir, se podrá ver el porcentaje de apertura, la activación y desactivación de la electroválvula de descarga. En este gráfico, las variables y los rangos que se fijarán en el eje de las abscisas y en el eje de las ordenadas son los mismo que en gráfico 3. Para lograr que este gráfico se posicione en la cuarta posición, es necesario que la variable llamada “Electroválvula Descarga” ya escalada en el diagrama de bloque (Ver tag 28 de la figura 2-30), se conecte al elemento 3 de la función “Bundle”.
5. Gráfico 5: Este gráfico, muestra la señal de respuesta del sistema ($y(t)$), es decir, muestra el nivel del líquido contenido en el estanque en tiempo real. En el eje de las abscisas, se encuentra el tiempo expresado en segundos y en el eje de las ordenadas, está la altura expresada en centímetros. El máximo valor que puede alcanzar en altura son 300 cm y el mínimo valor son 0 cm. Para lograr que este gráfico se posicione en la quinta posición, es necesario que la variable llamada “Sensor de Nivel” ya escalada en el diagrama de bloque (Ver tag 26 de la figura 2-30), se conecte al elemento 4 de la función “Bundle”.
6. Setpoint [cm]: Es un indicador o visualizador expresado en forma numérica, el cual se encarga de mostrar el setpoint (en centímetros) fijado por el usuario. Para que este indicador cumpla con su función, debe ir conectado en el diagrama de bloque a la salida del escalamiento que corresponde al setpoint (Ver tag 24 de la figura 2-30) para que su máximo valor sea 300.
7. Nivel del Estanque [cm]: Es un indicador o visualizador expresado en forma numérica, el cual se encarga de mostrar el nivel del estanque (en centímetros). Para que este indicador cumpla con su función, debe ir conectado en el diagrama de bloque a la variable llamada “Sensor de Nivel” ya escalada (Ver tag 26 en la figura 2-30) para que su máximo valor sea de 300.
8. Histéresis [cm]: Es un indicador, que muestra de forma numérica el valor de la histéresis (en centímetros) fijada por el usuario. Para que este indicador cumpla con su función, debe ir conectado en el diagrama de bloque a la salida del escalamiento que corresponde a la histéresis (Ver tag 25 de la figura 2-30) para que su máximo valor sea 300.

9. Porcentaje de apertura de la electroválvula de llenado: Es un visualizador que permite mostrar en porcentaje la apertura de la electroválvula de carga, es decir, es lo mismo que se mostrará en el gráfico 3, con la diferencia que esta vez, estará expresado en forma numérica. Para que este indicador cumpla con su función, debe ir conectado en el diagrama de bloque a la salida del escalamiento que corresponde a la variable llamada “Electroválvula Carga” (Ver tag 27 de la figura 2-30) para que su máximo valor sea 100.
10. Porcentaje de apertura de la electroválvula de descarga: Es un visualizador que permite mostrar en porcentaje la apertura de la electroválvula de descarga, es decir, es lo mismo que se mostrará en el gráfico 4, con la diferencia que esta vez, estará expresado en forma numérica. Para que este indicador cumpla con su función, debe ir conectado en el diagrama de bloque a la salida del escalamiento que corresponde a la variable llamada “Electroválvula Descarga” (Ver tag 28 de la figura 2-30) para que su máximo valor sea 100.
11. Umbral Inferior [cm]: Es un visualizador o indicador que muestra en centímetros el valor que tomará el umbral inferior luego de fijar el setpoint y la histéresis de forma manual. Para que este indicador cumpla con su función, debe ir conectado en el diagrama de bloque a la salida de la función llamada “Subtract” (Ver tag 29 de la figura 2-30).
12. Umbral Superior [cm]: Es un indicador o visualizador que muestra en centímetros el valor que tomará el umbral superior luego de fijar el setpoint y la histéresis de forma manual. Para que este indicador cumpla con su función, debe ir conectado en el diagrama de bloque a la salida de la función llamada “Add” (Ver tag 30 de la figura 2-30).
13. Tubo Descarga 4: Es la tubería de descarga, es decir, se activa o se enciende solamente cuando la electroválvula de descarga entra en funcionamiento. Para lograr que esta tubería cumpla con su función, debe ir conectada a un “True” dentro de la estructura “Case” (Ver tag 34 de la figura 2-31) con el selector label en “True” para que se active y a la vez con una variable local conectada a un “False” dentro de la otra estructura “Case” (Ver tag 33 de la figura 2-31) con el selector label en “True” para que se desactive.

14. LED Descarga 2: Es un led o indicador que se enciende solamente cuando el sensor de flujo entra en funcionamiento. Para lograr que este indicador cumpla con su función, debe ir conectada a un "True" dentro de la estructura "Case" (Ver tag 34 de la figura 2-31) con el selector label en "True" para que se active y a la vez con una variable local conectada a un "False" dentro de la otra estructura "Case" (Ver tag 33 de la figura 2-31) con el selector label en "True" para que se desactive.
15. LED Descarga 3: Es un led o indicador que se enciende solamente cuando la electroválvula de descarga entra en funcionamiento. Para lograr que este indicador cumpla con su función, debe ir conectada a un "True" dentro de la estructura "Case" (Ver tag 34 de la figura 2-31) con el selector label en "True" para que se active y a la vez con una variable local conectada a un "False" dentro de la otra estructura "Case" (Ver tag 33 de la figura 2-31) con el selector label en "True" para que se desactive.
16. Tubo Descarga 1: Es un led o indicador que está siempre encendido, ya que es parte de la base del estanque. Para lograr que este indicador cumpla con su función, debe ir conectado a un "True" en cualquier parte dentro de la estructura "While Loop" (ver tag 38 de la figura 2-30).
17. Estanque: Es un estanque que muestra el nivel del líquido contenido en el estanque de la planta que entrega Factory I/O. Para que este simulador cumpla con su función, debe ir conectado en el diagrama de bloque a la salida del escalamiento, referido a la variable llamada "Sensor de Nivel" (Ver tag 26 de la figura 2-30) para que tome un máximo valor de 300.
18. Tubo Carga 2: Esta tubería de carga, se enciende solamente cuando la electroválvula de carga entra en funcionamiento. Para que esta tubería cumpla con su función, debe ir conectada a un "True" dentro de la estructura "Case" (Ver tag 33 de la figura 2-31) con el selector label en "True" para que se active y a la vez con una variable local conectada a un "False" dentro de la otra estructura "Case" (Ver tag 34 de la figura 2-31) con el selector label en "True" para que se desactive.

19. Led Carga 2: Es un led o indicador que se enciende solamente cuando la electroválvula de carga o llenado entra en funcionamiento. Para lograr que este indicador cumpla con su función, debe ir conectado a un "False" dentro de la estructura "Case" (Ver tag 34 de la figura 2-31) con el selector label en "True" para que se desactive y a la vez con una variable local conectada a un "True" dentro de la otra estructura "Case" (Ver tag 33 de la figura 2-31) con el selector label en "True" para que se active.
20. Tubo Carga 1: Es un led o indicador que está siempre activado o encendido. Para lograr que este indicador cumpla con su función, debe ir conectado a un "True" en cualquier parte dentro de la estructura "While Loop" (Ver tag 38 de la figura 2-30).
21. Pin Analógico Setpoint: Es un visualizador o indicador que informa el pin analógico físico de Arduino UNO, usado para conectar el potenciómetro encargado de fijar el setpoint. Para que este indicador cumpla con su función, en el diagrama de bloques debe ir conectado a una constante igual a 0 (para referirse al pin "A0" de Arduino UNO) en terminal de entrada llamado "Analog Channel" del bloque LINX "Analog Read", referido al setpoint (Ver figura 2-33 y tag 2 de la figura 2-30).
22. Pin Analógico Histéresis: Es un visualizador o indicador que informa el pin analógico físico de Arduino UNO, usado para conectar el potenciómetro encargado de fijar la histéresis. Para que este indicado cumpla con su función, en el diagrama de bloques debe ir conectado a una constante igual a 1 (para referirse al pin "A1" de Arduino UNO) en terminal de entrada llamado "Analog Channel" del bloque LINX "Analog Read", referido a la histéresis (Ver figura 2-33 y tag 3 de la figura 2-30).
23. Puerto Conectado: Este interactivo o cuadro de diálogo, permite seleccionar el puerto COM utilizado para la comunicación entre Arduino UNO y el ordenador. Para que este interactivo cumpla con su función, en el diagrama de bloque debe ir conectado al terminal de entrada llamado "Serial Port" del bloque "Open Serial" (Ver figura 2-32 y tag 1 de la figura 2-30).

24. Play: En el panel frontal es llamado “Iniciar/ Detener Proceso” y en el diagrama de bloques “Play”. Es un interruptor que se enciende solamente cuando es presionado por primera vez. Al ser presionado por primera vez, permitirá iniciar el proceso o planta que entrega Factory I/O. La segunda vez que se presione, hará detener el proceso en Factory I/O. Para que este interruptor cumpla con su función, debe ir conectado en el diagrama de bloque a la variable llamada “FACTORY IO (Run)” (Ver tag 12 de la figura 2-30).
25. Pause: En panel frontal es llamado “Pausar Proceso” y en diagrama de bloques “Pause”. Este interruptor, se enciende solamente cuando es accionado por 1ra vez. Al ser accionado por 1ra vez, permitirá pausar el proceso o planta en Factory I/O. La 2da vez que se presione, hará que el proceso continúe desde el momento que se presionó el interruptor. Para que este interruptor cumpla con su función, debe ir conectado en el diagrama de bloque a la variable llamada “FACTORY IO (Pause)” (Ver tag 11 de la figura 2-30).
26. Reset: En panel frontal es llamado “Resetear Escena” y en diagrama de bloque “Reset”. Es un pulsador normalmente abierto que se enciende al ser presionado. Al ser presionado, permite resetear la escena o proceso en Factory I/O. Para que este pulsador cumpla con su función, debe ir conectado en el diagrama de bloque a la variable llamada “FACTORY IO (Reset)” (Ver tag 10 de la figura 2-30).
27. Stop: En panel frontal es llamado “Detener Monitoreo” y en diagrama de bloques “Stop”. Es un pulsador normalmente abierto, que se enciende al ser presionado. Este pulsador al ser accionado, permite detener el bucle o ciclo While, es decir, permite detener el monitoreo en LabVIEW. Para que este interruptor cumpla con su función, debe ir conectado en el diagrama de bloque al ícono llamado “Conditional Terminal” (Ver tag 39 de la figura 2-30) de la estructura “While Loop”.

CAPÍTULO 3: RESULTADOS OBTENIDOS

3. RESULTADOS OBTENIDOS

En este capítulo, se mostrarán las pruebas de control de nivel de la planta que entrega Factory I/O. La finalidad de estas pruebas, es visualizar mediante la interfaz HMI o panel frontal de LabVIEW, el comportamiento de la variable de nivel. Esta prueba, se llevará a cabo a través de un sistema de control a lazo cerrado. Además, se realizará la identificación del sistema.

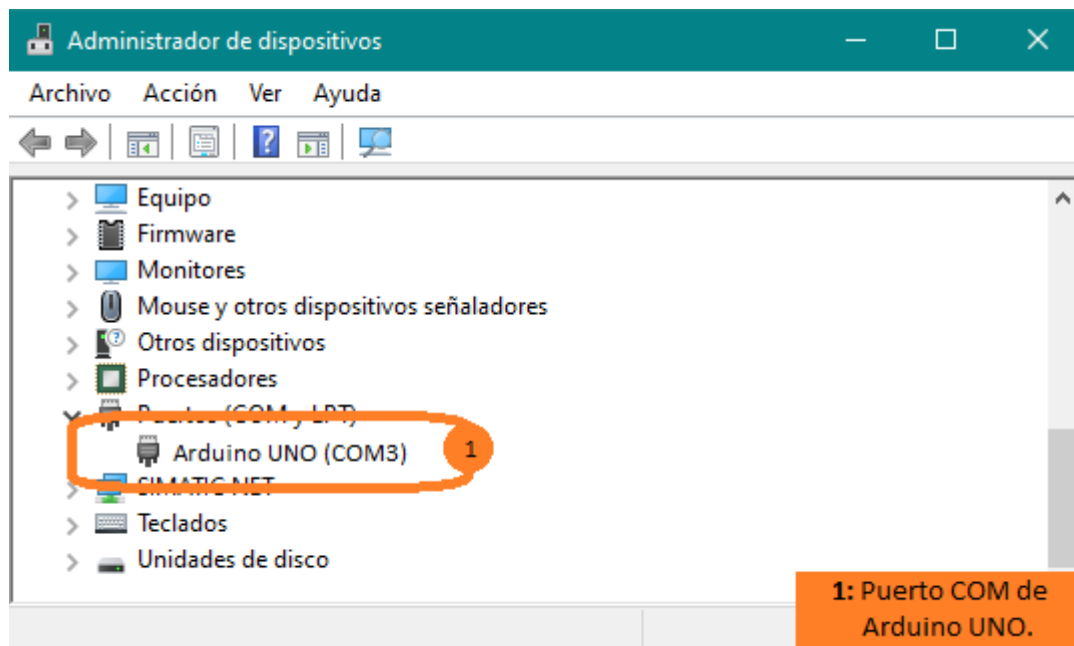
3.1. SISTEMA A LAZO CERRADO

Este sistema de control a lazo cerrado, ha sido descrito con anterioridad en el Capítulo 2 de este informe. Por lo tanto, se procede directamente a detallar el comportamiento de la escena de nivel que entrega Factory I/O, haciendo uso de este método de control.

3.1.1. Ejecutar el control

Para ejecutar el control de nivel y ver el monitoreo del estanque que entrega Factory I/O, se debe seguir una serie de pasos, los cuales se describen a continuación.

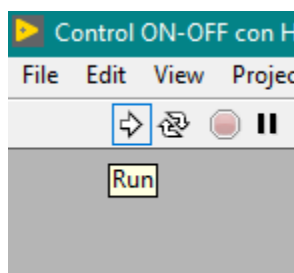
- ✓ Paso 1: Conectar la placa Arduino UNO al PC o computador, a través de los puertos USB.
- ✓ Paso 2: Presionar en el computador la combinación de teclas “Win + x” o escribir en el menú de inicio “Administrador de Dispositivo”. Se abrirá una ventana con una gran cantidad de secciones. Se debe buscar la sección llamada “Puertos COM y LPT”, donde se va a encontrar los USB, impresoras y otros dispositivos encontrados. Es en esta sección, donde se podrá conocer el puerto COM referido a Arduino UNO. Esta sección se puede apreciar en la figura 3-1, (mostrada a continuación), donde se podrá conocer que Arduino UNO, se encuentra conectado al ordenador a través del puerto COM3.



Fuente: Elaboración propia.

Figura 3 - 1. Puerto COM de Arduino UNO.

- ✓ Paso 3: Abrir la programación de lenguaje G del proyecto en LabVIEW y también abrir el archivo correspondiente a la planta de nivel creada en Factory I/O.
- ✓ Paso 4: En el panel frontal de LabVIEW en el recuadro llamado “Arduino UNO” donde dice “Puerto COM”, se debe seleccionar el puerto de comunicación visto anteriormente a través de la figura 3-1, es decir, COM3.
- ✓ Paso 5: Se debe cargar toda la programación en lenguaje G de LabVIEW a Arduino UNO. Para ello, se debe seguir una serie de pasos que se mostraron anteriormente por medio de las figuras 1-7, 1-8 y 1-9 (Capítulo 1).
- ✓ Paso 6: Presionar en el botón “Run”, tal y como se muestra en la figura 3-2.
- ✓ Paso 7: Fijar los valores de setpoint e histéresis del sistema mediante los potenciómetros conectados a la entrada analógica (A0 y A1) de Arduino UNO.
- ✓ Paso 8: Presionar en el botón llamado “Iniciar/ Detener Proceso” (tal y como se muestra en la figura 3-3) para que empiece el control sobre los actuadores de la planta.



Fuente: Elaboración Propia, realizada en LabVIEW 2020.

Figura 3 - 2. Botón Run para iniciar programación en Lenguaje G.



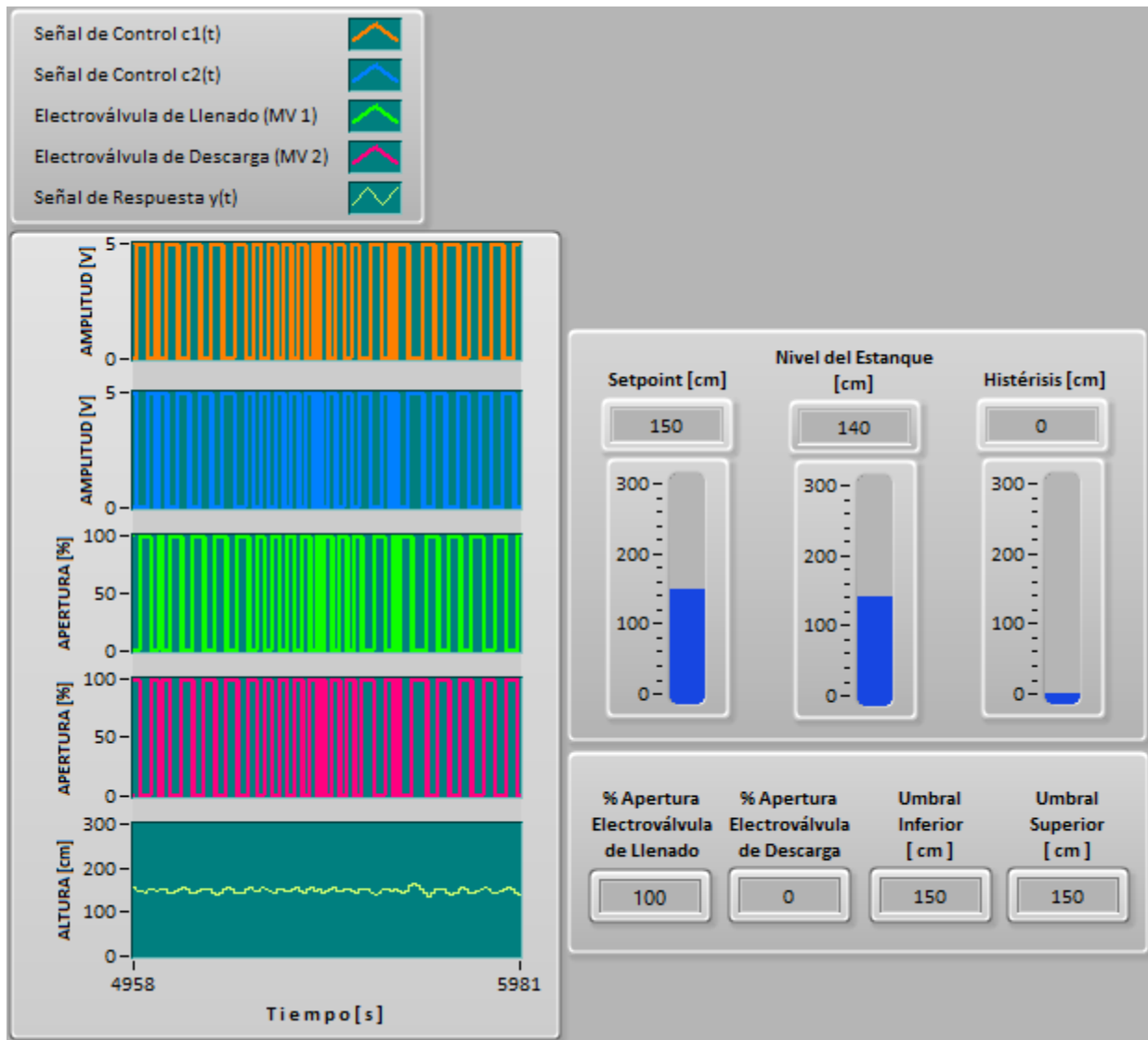
Fuente: Elaboración Propia, realizada en LabVIEW 2020.

Figura 3 - 3. Botón iniciar/detener proceso automático.

Ahora que se conoce el procedimiento necesario para ejecutar o iniciar el proceso automatizado del sistema de nivel que entrega Factory I/O, se procede directamente a realizar las pruebas correspondientes.

3.1.2. Prueba 1: Control ON-OFF sin histéresis

La primera prueba aplicada a la planta de nivel, se basa en un control ON-OFF sin histéresis para ver el comportamiento que se tiene a la salida del sistema y en ambas electroválvulas (variables manipuladas, MV 1 y MV 2). Por lo tanto, a través de la figura 3-4, mostrada a continuación, se puede apreciar dicho comportamiento.



Fuente: Elaboración propia, realizada en LabVIEW 2020.

Figura 3 - 4. Control ON-OFF sin histéresis.

En los gráficos de la figura 3-4, se pueden observar algunas líneas de colores, las cuales se describen a continuación.

- ✓ Línea naranja: Muestra el comportamiento de la señal de control $c_1(t)$ (señal que llega a la entrada de la electroválvula de carga o llenado) en el tiempo.

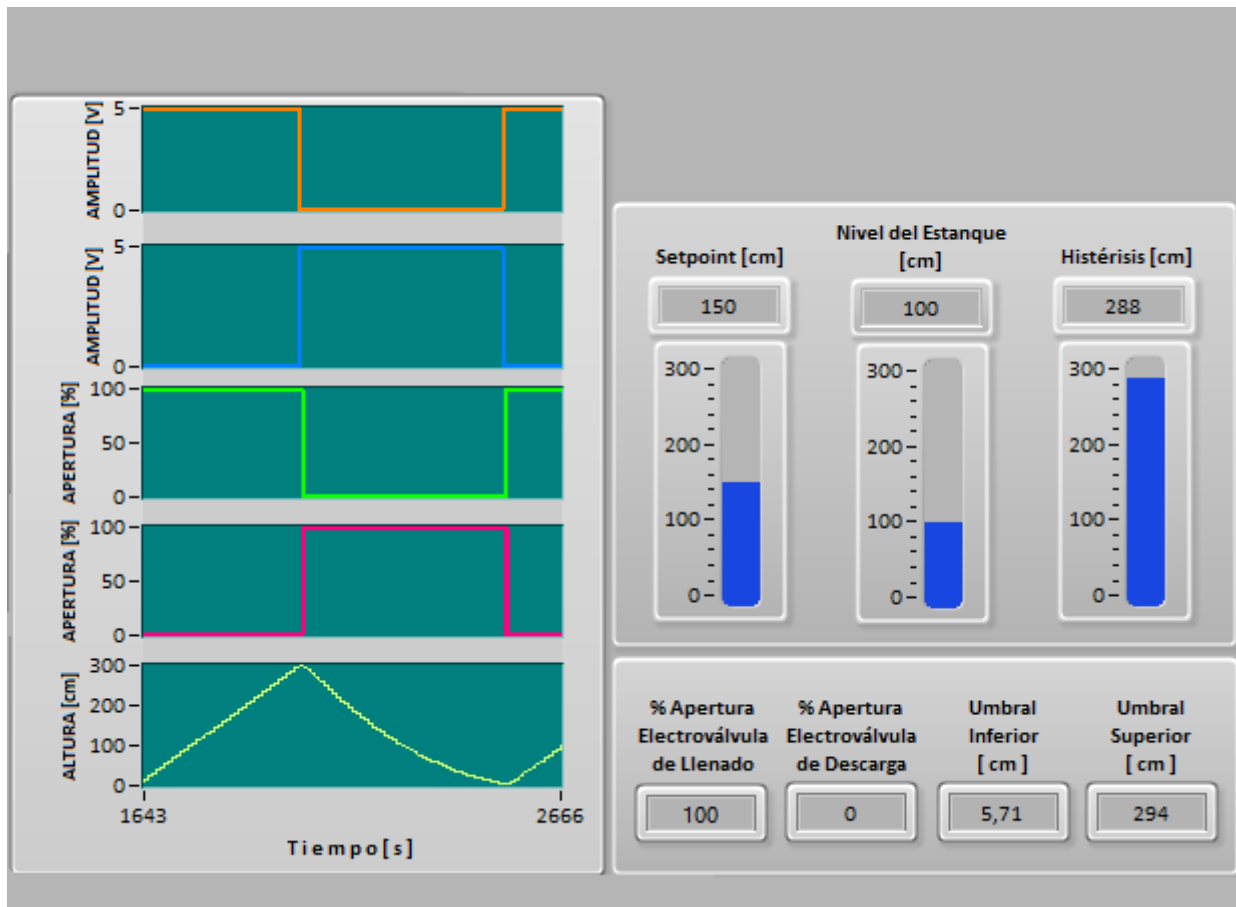
- ✓ Línea azul: Muestra el comportamiento en el tiempo de la otra señal de control, es decir, la $c_2(t)$ (señal que llega a la entrada de la electroválvula de descarga).
- ✓ Línea verde: Muestra el comportamiento de la variable manipulada MV 1 (variable que se obtiene a la salida de la electroválvula de carga o llenado) en el tiempo.
- ✓ Línea rosada: Muestra el comportamiento en el tiempo de la otra variable manipulada MV 2 (variable que se obtiene a la salida de la electroválvula de descarga).
- ✓ Línea roja: Muestra el comportamiento de la señal de respuesta $y(t)$ (señal que se obtiene a la salida de la planta y es la que indica el nivel del líquido) en el tiempo.

Además, en esta figura 3-4, se puede apreciar a un costado derecho unos indicadores informativos referidos al nivel del estanque, la histéresis, los porcentajes de apertura de ambas electroválvulas y los valores de umbral superior e inferior.

Ahora bien, como se puede ver en la figura 3-4, se fijó un valor de setpoint igual a 150 cm (equivalente a 2,49V medido desde el pin 2 del potenciómetro conectado a la entrada A0 de Arduino UNO, a tierra) y un valor de histéresis igual a 0 cm (equivalente a 0V medido desde el pin 2 del potenciómetro conectado a la entrada A1 de Arduino UNO, a tierra). La ventaja de no contar con una histéresis, se ve reflejada en la respuesta, ya que, permite tener al sistema más cerca al punto de control, pero ambas electroválvulas conmutarían muy seguido, lo que es un hecho no favorable, porque, se reduce la vida útil y esto a la larga se traduce en más gastos para la empresa.

3.1.3. Prueba 2: Control ON-OFF con un valor de histéresis elevado

La segunda prueba aplicada a la planta de nivel, se basa en un control ON-OFF con un valor de histéresis elevado para ver el comportamiento que se tiene a la salida del sistema y en ambas electroválvulas (variables manipuladas, MV 1 y MV 2). Por lo tanto, a través de la figura 3-5, mostrada a continuación, se puede apreciar dicho comportamiento.



Fuente: Elaboración propia, realizada en LabVIEW 2020.

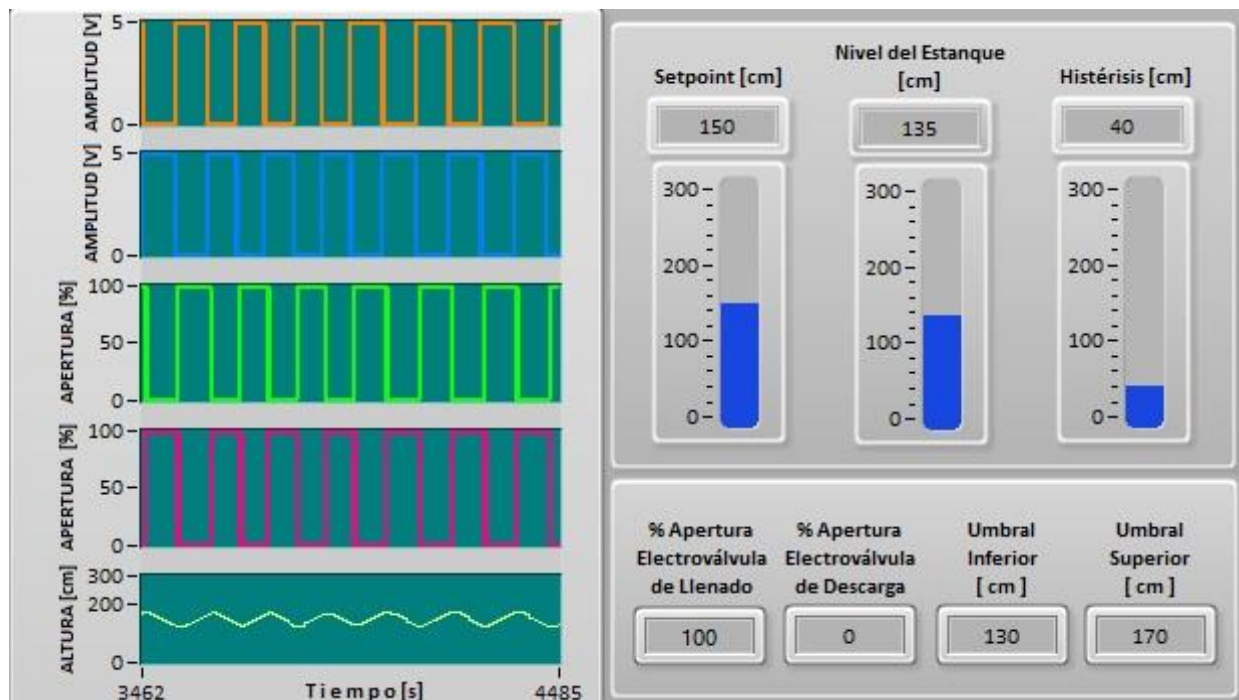
Figura 3 - 5. Control ON-OFF con histéresis elevado.

Como se puede apreciar en la figura 3-5, se fijó un valor de setpoint igual a 150 cm (equivalente a 2,49V medido desde el pin 2 del potenciómetro conectado a la entrada A0 de Arduino UNO, a tierra) y un valor de histéresis elevado igual a 288 cm (equivalente a 4,80V medido desde el pin 2 del potenciómetro conectado a la entrada A1 de Arduino UNO, a tierra). La ventaja de contar con una histéresis elevado, se ve reflejado en la duración de ambas electroválvulas, es decir, alargaría la vida útil de estas lo que es un hecho favorable en cuanto gastos para la empresa, sin embargo, la respuesta se aleja demasiado del punto de control, lo cual no es óptimo ni tampoco adecuado para el sistema.

Cabe mencionar, que el valor de histéresis se divide en dos, ya que se necesita sumar la mitad al valor de setpoint para obtener el valor de umbral superior y se necesita la otra mitad para restárselo al valor de setpoint para obtener el valor de umbral inferior.

3.1.4. Prueba 3: Control ON-OFF con un valor de histéresis óptimo

La tercera prueba aplicada a la planta de nivel, se basa en un control ON-OFF con un valor de histéresis óptimo para ver el comportamiento que se tiene a la salida del sistema y en ambas electroválvulas (variables manipuladas, MV 1 y MV 2). Por lo tanto, a través de la figura 3-6, mostrada a continuación, se puede apreciar dicho comportamiento.



Fuente: Elaboración propia, realizada en LabVIEW 2020.

Figura 3 - 6. Control ON-OFF con histéresis óptimo.

Como se puede apreciar en la figura 3-6, se fijó un valor de setpoint igual a 150 cm (equivalente a 2,49V medido desde el pin 2 del potenciómetro conectado a la entrada A0 de Arduino UNO, a tierra) y un valor de histéresis óptimo igual a 40 cm (equivalente a 0,65V medido desde el pin 2 del potenciómetro conectado a la entrada A1 de Arduino UNO, a tierra).

El valor de histéresis escogido, resulta óptimo, ya que la respuesta se mantiene más cerca del valor de setpoint lo que es un hecho favorable y también las electroválvulas no conmutarán tan seguido, sino que lo justo para mantener a la respuesta en el valor de setpoint.

El valor de histéresis al igual que en la prueba 2 se divide en dos, ya que se necesita sumar la mitad al valor de setpoint para obtener el valor de umbral superior y se necesita la otra mitad para restárselo al valor de setpoint para obtener el valor de umbral inferior.

3.2. IDENTIFICACIÓN DE SISTEMAS

Para identificar el sistema utilizado en el proyecto, se hará uso del sistema de control a lazo abierto (descrito anteriormente en el capítulo 2), por lo que se procede directamente a detallar el comportamiento de la escena de nivel que entrega Factory I/O, haciendo uso de este método de control.

Existen 2 tipos de sistemas, que son sistemas lineales y sistemas no lineales. En un sistema lineal, existe una relación proporcional entre la variable de entrada y de salida, es decir, en este tipo de sistema se cumple con los principios de superposición y homogeneidad. Este tipo de sistema lineal, resulta útil para el modelado y síntesis de controladores, sin embargo, en la mayoría de sistemas encontrados en la práctica son prácticamente no lineales, siendo muy poquitos los que son lineales. Los sistemas no lineales, son todos aquellos sistemas que no cumplen con los principios de superposición y homogeneidad, es decir, en estos sistemas la relación entre la variable de entrada y salida no puede ser representada con una ecuación diferencial lineal, sin embargo, se pueden considerar lineales en ciertos rangos de trabajo u operación.

Todos los sistemas no lineales, se pueden linealizar alrededor de un punto de operación conocido. Para ello, se debe realizar la siguiente prueba, que consiste en evaluar la salida del sistema ante distintos valores a la entrada. En esta oportunidad, la entrada será una actuación manual de 0 a 5[V] (para controlar el porcentaje de apertura de la electroválvula de carga o llenado, mientras se mantiene totalmente abierta la electroválvula de descarga) y la salida será el nivel o la altura del líquido contenido en el estanque (en cm). El sistema de control a lazo abierto se puede apreciar en la figura 3-7 y la prueba a través de las tablas 3-1, 3-2 y 3-3, mostradas a continuación.



Fuente: Elaboración propia, realizada en Microsoft Word.

Figura 3 - 7. Sistema de control a lazo abierto.

Tabla 3 - 1. Nivel v/s Actuación – Parte 1.

Actuación [V]	Nivel [cm]
0	0
0,1	0
0,2	0
0,3	0,3
0,4	0,9
0,5	1,2
0,6	2,1
0,7	2,7
0,8	3,6
0,9	4,8
1	5,7

Fuente: Elaboración propia, realizada en Microsoft Excel.

Tabla 3 - 2. Nivel v/s Actuación – Parte 2.

Actuación [V]	Nivel [cm]
1,1	7,2
1,2	8,4
1,3	9,9
1,4	11,7
1,5	13,2
1,6	15
1,7	17,1
1,8	19,2
1,9	21,3
2	23,7
2,1	26,1
2,2	28,8
2,3	31,5
2,2	28,8
2,3	31,5
2,4	34,2
2,5	37,2
2,6	40,2
2,7	43,5
2,8	46,8
2,9	50,1
3	53,7
3,1	57,3
3,2	60,9
3,3	64,8
3,4	69
3,5	72,9
3,6	77,4
3,7	81,6
3,8	86,1
3,9	90,6
4	95,4
4,1	100,2
4,2	105,3
4,3	110,4
4,4	115,5

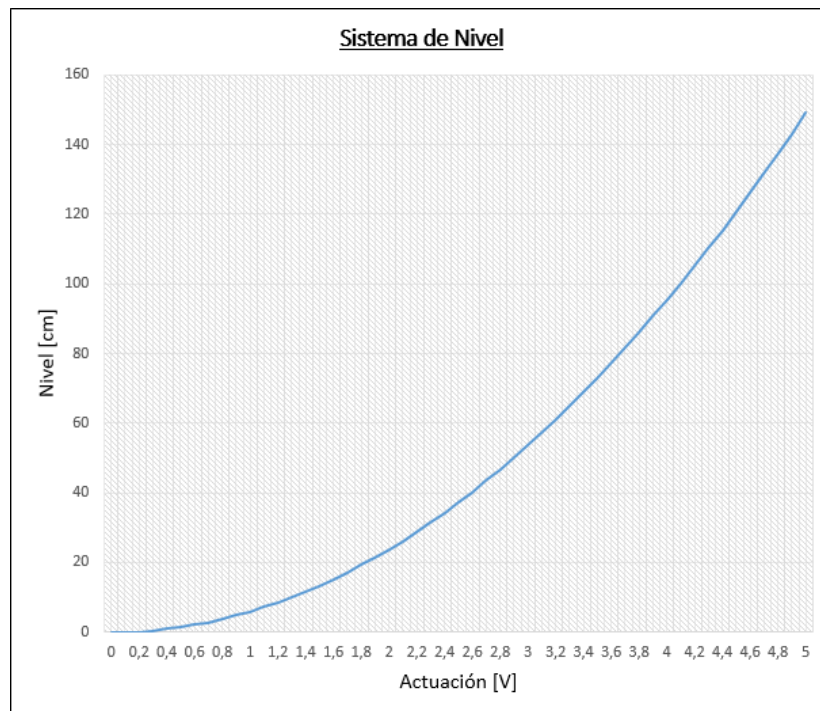
Fuente: Elaboración propia, realizada en Microsoft Excel.

Tabla 3 - 3. Nivel v/s Actuación – Parte 3.

Actuación [V]	Nivel [cm]
4,5	120,9
4,6	126,3
4,7	131,7
4,8	137,4
4,9	143,1
5	149,1

Fuente: Elaboración propia, realizada en Microsoft Excel.

Tal y como se muestra en la tabla 3-1, 3-2 y 3-3, se realizó la evaluación de la salida del sistema ante variaciones de voltaje a la entrada. Dicho esto, se procede a elaborar un gráfico que muestre o represente dicha evaluación. Para ello, se debe observar o estudiar la figura 3-8.



Fuente: Elaboración propia, realizada en Microsoft Excel.

Figura 3 - 8. Sistema no lineal.

La figura 3-8, muestra la relación que existe entre la actuación y el nivel. Entonces, se puede desprender a simple vista que el sistema no es lineal, ya que, no presenta una línea recta en la gráfica. Este sistema es no lineal, por lo que se puede trabajar en los más diversos puntos de operación, es decir, en cualquier punto de la gráfica. Ahora bien, para reafirmar esta no linealidad en términos matemáticos, se debe realizar las pruebas siguiendo el principio de homogeneidad y superposición.

El principio de homogeneidad, dice que si se tiene un sistema con una entrada $u(t)$ multiplicada por una constante o una ganancia " α ", entonces eventualmente la salida $y(t)$ debe ser igualmente afectada por esta constante. Por lo tanto, la aplicación de este principio se puede observar a través de la tabla 3-4, mostrada a continuación.

Tabla 3 - 4. Principio de homogeneidad.

	Actuación [V]	Nivel [cm]	α
Prueba	1,5	13,2	2
Prueba (teórica)	$1,5 * 2 = 3$	$13,2 * 2 = 26,4$	
Prueba (práctica)	3	53,7	

Fuente: Elaboración propia, realizada en Microsoft Excel.

De acuerdo a los datos entregados a través de la tabla 3-4, se realizaron 2 pruebas prácticas. En la primera prueba, se aplica un valor de actuación de 1,5V, donde se obtuvo un nivel a la salida de 13,2cm. En la prueba (teórica), se multiplicó el valor de actuación por la constante α (que es igual a 2), donde se obtuvo teóricamente un valor de actuación de 3V y un nivel a la salida del sistema de 26,4cm. En la prueba (práctica), se esperaba que al aplicar un valor de actuación de 3V, se obtendría a la salida del sistema un nivel de 26,4cm, sin embargo, esto en la práctica no fue así, sino que resultó ser 53,7cm. Por lo tanto, se puede volver a reafirmar que el sistema es no lineal.

El principio de superposición, consiste en que si se tiene un sistema con una entrada $u_1(t)$, entonces a la salida se debe tener un $y_1(t)$. Ahora, si se toma el mismo sistema, pero con una entrada diferente $u_2(t)$, entonces a la salida se tendrá una salida diferente $y_2(t)$. Dicho esto, un sistema cumpliría con el principio de superposición si la suma de las entradas $u_1(t) + u_2(t)$ ocasiona una salida $y_1(t) + y_2(t)$. Por lo tanto, se aplica este principio al sistema, tal y como se muestra en la tabla 3-5, mostrada a continuación.

Tabla 3 - 5. Principio de superposición.

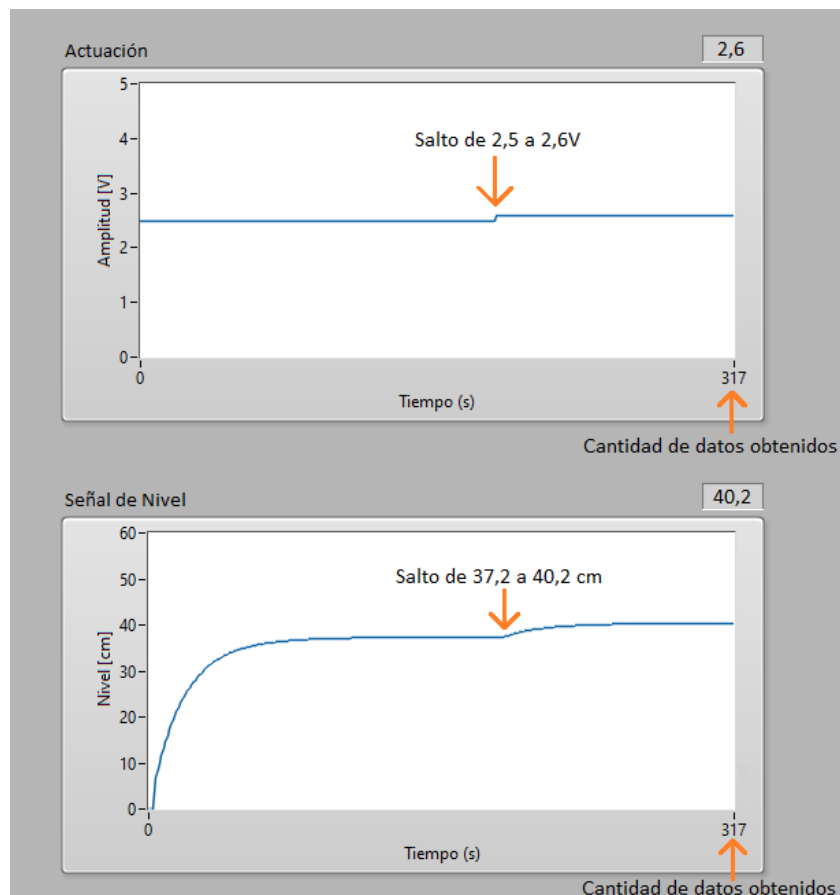
	Actuación [V]	Nivel [cm]
Prueba 1	1,5	13,2
Prueba 2	3	53,7
Prueba 3 (teórica)	$1,5 + 3 = 4,5$	$13,2 + 53,7 = 66,9$
Prueba 3 (práctica)	4,5	120,9

Fuente: Elaboración propia, realizada en Microsoft Excel.

De acuerdo a los datos entregados a través de la tabla 3-5, se realizaron 3 pruebas prácticas. En la prueba 1, se aplicó un valor de actuación de 1,5V, donde el nivel resultó ser 13,2cm. En la prueba 2, se realizó algo similar, pero con otros valores, es decir, se aplicó 3V como valor de actuación, donde se obtuvo un nivel a la salida de 53,7cm. En La prueba 3 (teórica), se suman los valores de actuación y nivel de las pruebas 1 y 2, donde resulta un valor de actuación de 4,5V y un nivel de 66,9cm. En la prueba 3 (práctica), se comprueba lo realizado en la prueba 3 (teórica), donde se esperaba que, al ingresar un valor de actuación de 4,5V a la entrada del sistema, la salida sería 66,9cm, sin embargo, en la práctica no fue así, es decir, resultó ser 120,9cm. Por lo tanto, se puede volver a reafirmar que el sistema es no lineal.

3.2.1. Punto de operación

Dado que el sistema es no lineal, puede ser trabajado en varios puntos de operación. Se decide en cuál se desea trabajar y en base a este punto, se debe aplicar un delta escalón muy pequeño. La idea de esto, es asimilar la curva (que es no lineal) en base a puras líneas rectas (que es lo lineal). Por lo tanto, de las tablas mostradas anteriormente se escoge el punto de operación (2,5V; 37,2cm) y se aplica un salto pequeño de 0,1V, es decir, el valor de la actuación saltará de 2,5V a 2,6V y de manera respectiva saltará el nivel de 37,2cm a 40,2cm. Este salto en el tiempo, se puede ver en la figura 3-9, mostrada a continuación.



Fuente: Elaboración propia, realizada en LabVIEW 2020.

Figura 3 - 9. Punto de operación del sistema.

En el gráfico superior de la figura 3-9, se puede apreciar el salto (pequeño) de 0,1V (para modelar la curva como un sistema lineal), es decir, la actuación paso de 2,5V a 2,6V, como también se puede observar, la cantidad de datos obtenidos, que son 317. Mientras tanto, en el gráfico inferior de esta figura, se puede apreciar el salto del nivel o respuesta del sistema, es decir, el nivel subió de 37,2cm a 40,2cm y además se puede observar que la cantidad de datos obtenidos son los mismo que en la gráfica superior. Entonces, $\Delta u=0,1$ y $\Delta y=3$.

3.2.2. Función de transferencia

La función de transferencia, es un modelo matemático que, por medio de un cociente, relaciona la respuesta de un sistema (modelada) con una señal de entrada o excitación (también modelada). En la teoría de control, las funciones de transferencia se utilizan a menudo para describir las relaciones de entrada y salida de componentes o sistemas descritos por ecuaciones diferenciales lineales e invariante en el tiempo (sistemas que cumplen con propiedades de linealidad e invarianza en el tiempo).

Ahora bien, la función de transferencia de un sistema LTI, se define como la transformada de Laplace de la respuesta y la transformada de Laplace de la entrada, asumiendo que todas las condiciones iniciales son iguales a cero. Esto quiere decir lo siguiente.

$$G(s) = \frac{L(\text{Salida})}{L(\text{Entrada})} = \frac{B(s)}{A(s)}$$

Si no hay factor común en el polinomio del numerador $B(s)$ y el polinomio en el denominador $A(s)$ de la función de transferencia, entonces las raíces de $B(s)$, serán los ceros de $G(s)$ y las raíces de $B(s)$, serán los polos de $G(s)$.

Ahora que se ha estudiado el concepto de la función, resulta viable utilizar la ecuación algebraica de “s” para representar la dinámica del sistema. Si la potencia o exponente más alto de las “s” en el denominador de la función de transferencia (polinomio $A(s)$), es igual a uno se denomina entonces que el sistema es de primer orden, lo que corresponde a lo siguiente.

$$G(s) = \frac{K}{\tau s + 1}$$

Para conseguir la función de transferencia de forma correcta, se lleva a cabo una prueba en lazo abierto. Esta prueba, consiste en generar un voltaje a la entrada de la electroválvula de carga, de 2,5V para luego añadir un salto o escalón de 2,6V, es decir, se aplicarán 2 entradas de tipo escalón, una para generar el modelo y la segunda para validar el modelo de la planta.

Por lo tanto, de la figura 3-9 mostrada anteriormente, se deben extraer todos los puntos de ambas gráficas (actuación y nivel). Para ello, se debe colocar el cursor del “mouse” en cualquier parte dentro de la gráfica, hacer clic con el botón derecho, para reposar el cursor en el menú “Export”, donde se desplegará y deberá presionar en “Export Data To Excel”. Esta acción, exportará de manera automática todos los valores de dicha gráfica a través de un libro Excel, donde se mostrarán 2 columnas, una que muestra el tiempo por dato obtenido y la segunda que muestra el valor en centímetros (para el caso de la gráfica de nivel). Siguiendo los mismos pasos, se obtiene una tercera columna del otro gráfico (gráfico de actuación), correspondiente al escalón aplicado en los intervalos de tiempos. La cantidad obtenida, es de 317 datos con un período de muestreo de 1 segundo para cada dato. Por lo tanto, el tiempo total del desarrollo de dicha prueba fue de aproximadamente de 5 minutos y 30 segundos. Ahora, de los 317 datos obtenidos de la gráfica en LabVIEW, resulta práctico mostrar solamente 20 cada 16 segundos, como se observa en las tablas 3-6 y 3-7, mostradas a continuación.

Tabla 3 - 6. Respuesta ante un delta escalón pequeño – Parte 1.

Tiempo [s]	Amplitud [V]	Nivel [cm]
0	2,5	0
16	2,5	22,2
32	2,5	30,9
48	2,5	34,5
64	2,5	36

Fuente: Elaboración propia realizada en Microsoft Excel.

Tabla 3 - 7. Respuesta ante un delta escalón pequeño – Parte 2.

Tiempo [s]	Amplitud [V]	Nivel [cm]
80	2,5	36,6
96	2,5	36,9
112	2,5	37,2
128	2,5	37,2
144	2,5	37,2
160	2,5	37,2
176	2,5	37,2
192	2,6	37,2
208	2,6	39
224	2,6	39,6
240	2,6	39,9
256	2,6	40,2
272	2,6	40,2
288	2,6	40,2
304	2,6	40,2

Fuente: Elaboración propia realizada en Microsoft Excel.

Ahora solo queda obtener la identificación de sistemas desde MatLab R2019a, a través de la función “ident”, o sea, de esta manera se podrá obtener o conseguir los valores para la función de primer orden y en consecuencia los de la función de transferencia en el punto de operación elegido anteriormente (2,5V; 37,2cm).

Ahora bien, como se estaba trabajando con un sistema que es no lineal (ver tabla 3-6 y 3-7), se debe trabajar con un sistema lineal para ingresarlos a MatLab, lo más normal es que se tenga que trabajar en el punto de operación “0, 0”, o sea, el salto de la actuación es de 2,5V a 2,6V para un sistema que es no lineal y de 0V a 0,1V para un sistema que es lineal. Para ello, se debe restar a todos los valores de la actuación 2,5V y para el caso de los valores de nivel, se debe restar el valor que resulta en la respuesta al aplicar el escalón de 0,1V en la actuación, o sea, 37,2cm. De los 317 datos que fueron modificados para que sean lineales, resulta práctico mostrar solamente 20 cada 16 segundos, como se observa en la tabla 3-8, mostrada a continuación, donde el período de muestreo se mantuvo, es decir, de 1 segundo para cada dato.

Tabla 3 - 8. Datos lineales y no lineales.

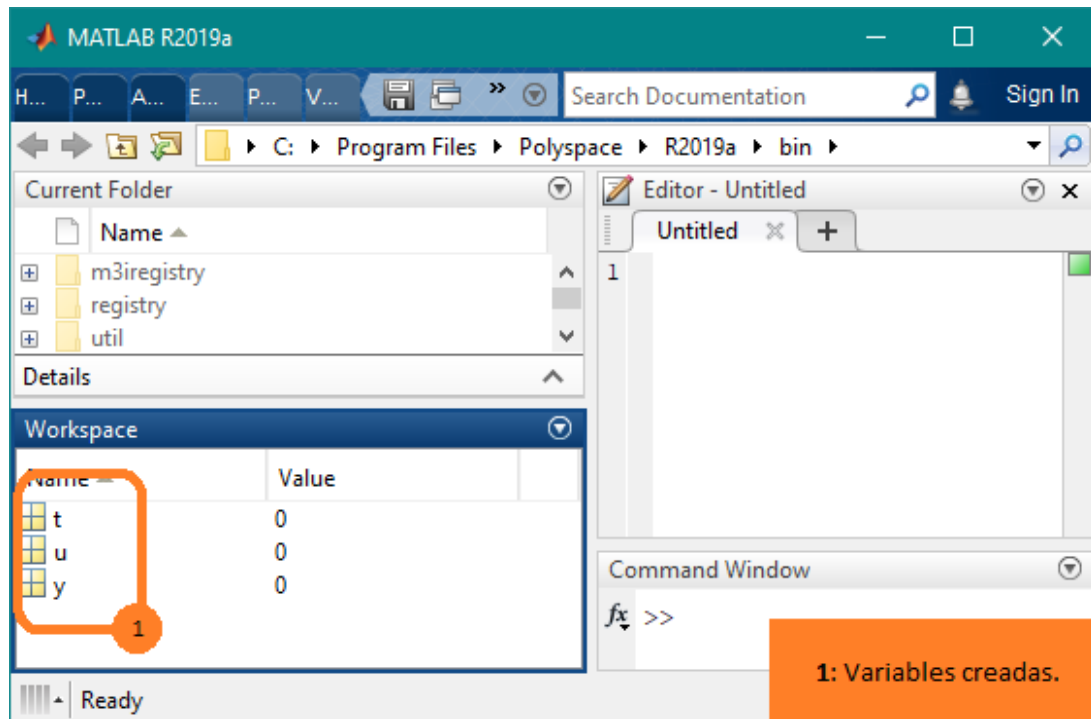
Tiempo [s]	Datos originales		Datos modificados	
	Amplitud [V]	Nivel [cm]	Amplitud [V]	Nivel [cm]
0	2,5	0	0	-37,2
16	2,5	22,2	0	-15
32	2,5	30,9	0	-6,3
48	2,5	34,5	0	-2,7
64	2,5	36	0	-1,2
80	2,5	36,6	0	-0,6
96	2,5	36,9	0	-0,3
112	2,5	37,2	0	0
128	2,5	37,2	0	0
144	2,5	37,2	0	0
160	2,5	37,2	0	0
176	2,5	37,2	0	0
192	2,6	37,2	0,1	0
208	2,6	39	0,1	1,8
224	2,6	39,6	0,1	2,4
240	2,6	39,9	0,1	2,7
256	2,6	40,2	0,1	3
272	2,6	40,2	0,1	3
288	2,6	40,2	0,1	3
304	2,6	40,2	0,1	3

Fuente: Elaboración propia, realizada en Microsoft Excel.

La tabla 3-8, muestra algunos de los datos obtenidos para un sistema lineal. Por lo tanto, estos son los datos en correcto que se deben enviar finalmente a MatLab. Para ello, se debe seguir una serie de pasos, los cuales se explican a continuación.

- ✓ Paso 1: Abrir MatLab y ubicarse con el cursor del mouse en el recuadro llamado "Workspace".
- ✓ Paso 2: Hacer clic derecho y presionar en "New" para crear una nueva variable. Este paso, se debe repetir 3 veces.

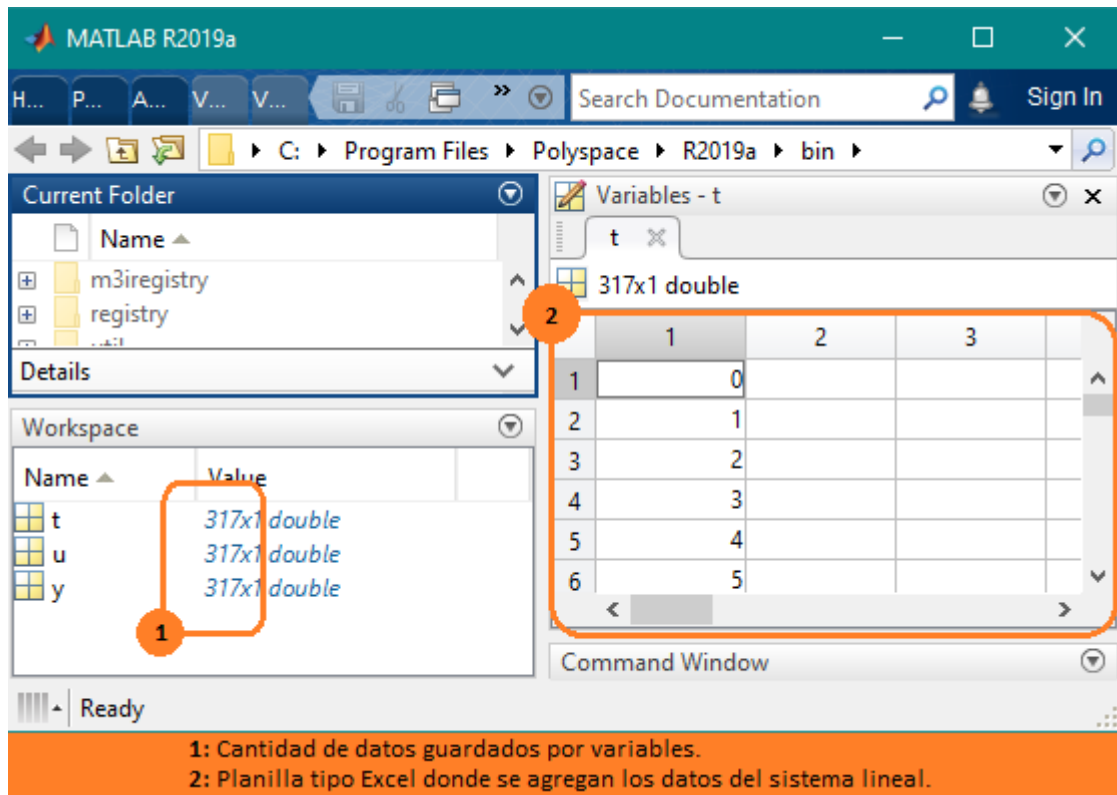
- ✓ Paso 3: Asignar nombres a cada variable creada. Los nombres serán “y”, “u” y “t”. Estas variables creadas, se pueden apreciar a través de la figura 3-10, mostrada a continuación.



Fuente: Elaboración propia, realizada en MatLab R2019a y etiquetada en Paint.

Figura 3 - 10. Variables creadas en MatLab.

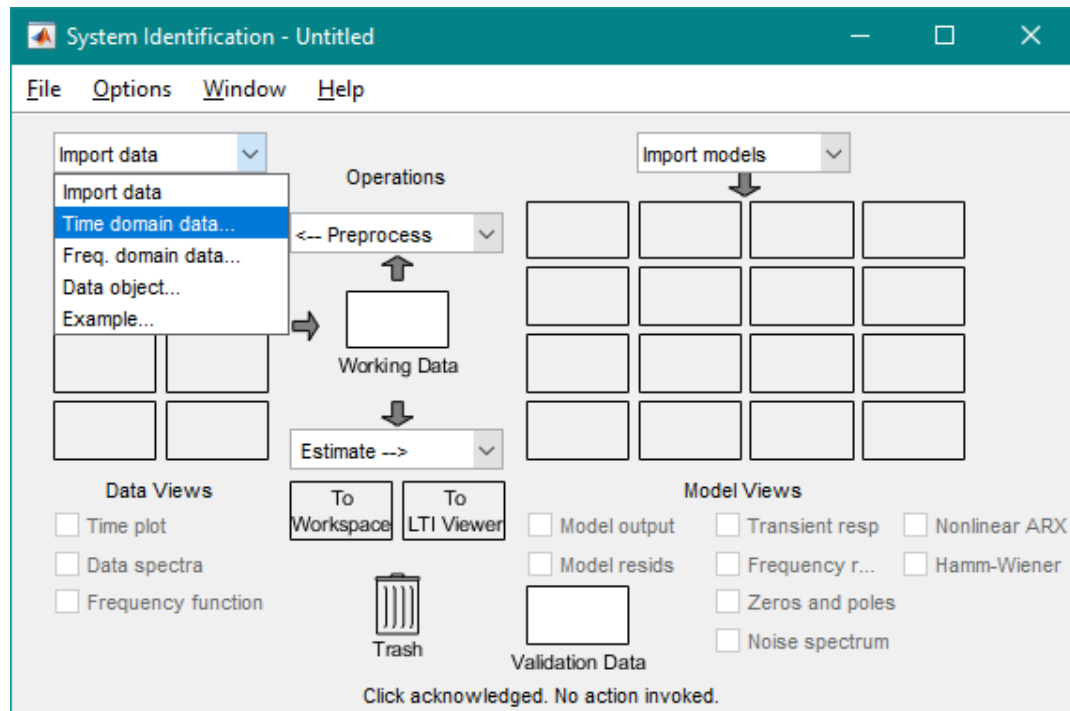
- ✓ Paso 5: Abrir la variable “t”, haciendo con el mouse doble clic izquierdo. Esta acción, mostrará una planilla (como de Excel), donde se debe pegar todos los valores copiados de la columna llamada “Tiempo [s]” de Excel.
- ✓ Paso 6: Realizar el mismo procedimiento del paso anterior (paso 5) para agregar los valores de las columnas llamadas “Actuación [V]” y “Nivel [cm]” en las variables llamada “u” e “y”. La cantidad de datos guardados por medio de planilla en MatLab, se podrá ver a través de la figura 3-11, mostrada a continuación.



Fuente: Elaboración propia, realizada en MatLab R2019a y etiquetada en Paint.

Figura 3 - 11. Datos agregados en las variables.

- ✓ Paso 7: Escribir en el recuadro llamado "Command Window" de MatLab "ident" (sin comillas). Al darle "enter" luego de escribir este comando, se abrirá un cuadro de dialogo llamado "System Identification", donde se debe agregar algunas variables. Para agregar las variables, se debe presionar en el despliegue que aparece en "Import Data" y luego presionar en "Time Domain Data...". El cuadro de dialogo llamado "System Identification" junto con el despliegue llamado "Import Data", se puede apreciar por medio de la figura 3-12, mostrada a continuación.



Fuente: Elaboración propia, realizada en MatLab R2019a.

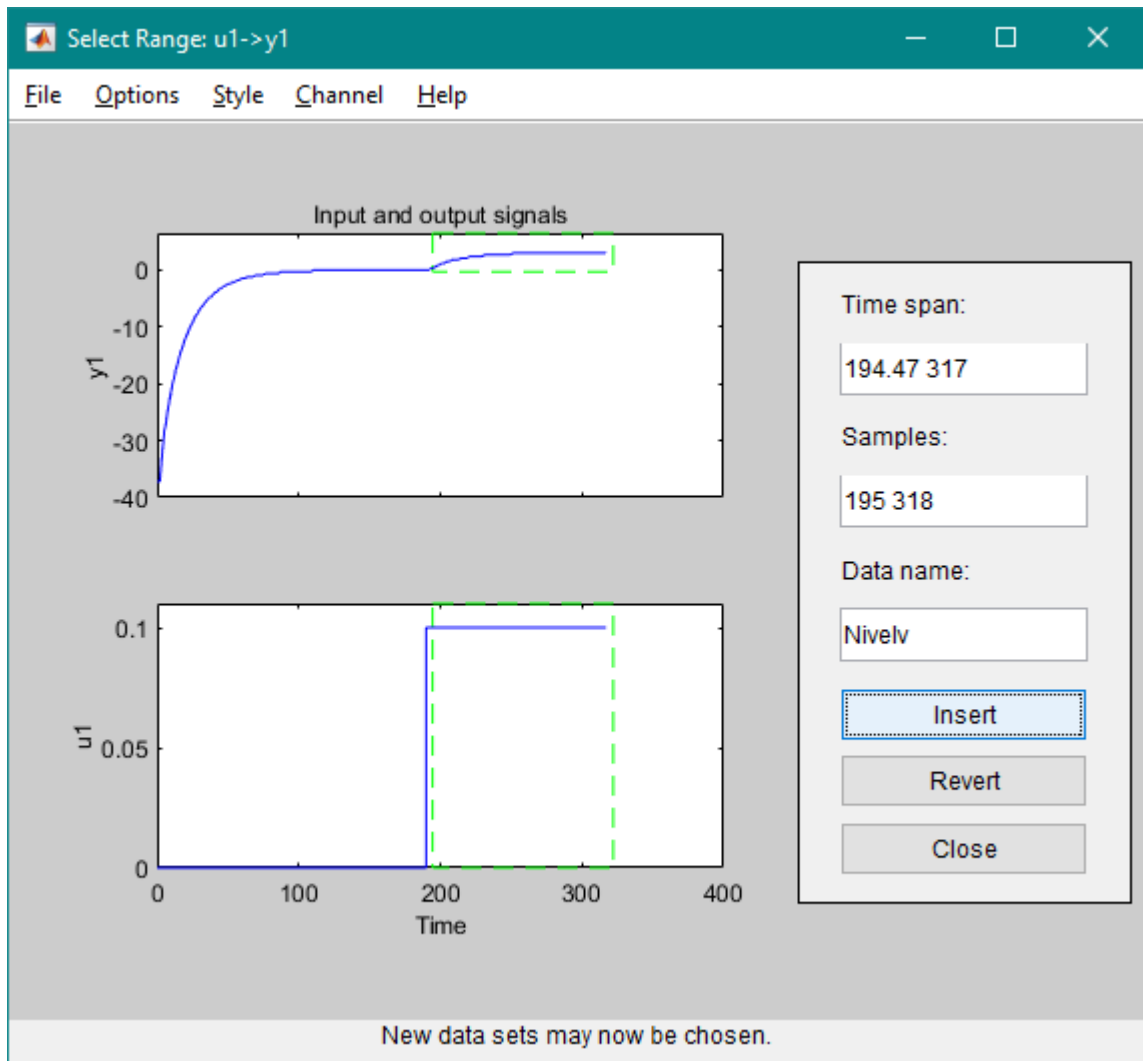
Figura 3 - 12. Cuadro de dialogo llamado “System Identification”.

- ✓ Paso 8: Al presionar en “Time Domain Data”, aparecerá un cuadro de dialogo llamado “Import Data”, donde se debe agregar las variables “u” e “y”. Para agregar los datos de la variable “u”, se debe escribir donde dice “Input” en la sección llamada “Workspace Variable” la palabra “u”. Por otro lado, para agregar la variable “y”, se debe escribir en la misma sección, pero donde dice “Output” la palabra “y”. También, se debe agregar el nombre de la gráfica, el tiempo de inicio y el tiempo de muestreo. Para agregar el nombre “Nivel”, se debe escribir donde dice “Data name” en la sección “Data Information”. En la misma sección, se podrá agregar el tiempo de inicio que es 0 en “starting time” y también el tiempo de muestreo que es 1 en “Sample time”. Luego, solo debe hacer clic en “Import” y ya se puede cerrar presionando en “Close”. Estos datos agregados en el cuadro de dialogo llamado “Import Data”, se puede apreciar a través de la figura 3-13, mostrada a continuación.

Fuente: Elaboración propia, realizada en MatLab R2019a.

Figura 3 - 13. Cuadro de dialogo llamado Import Data.

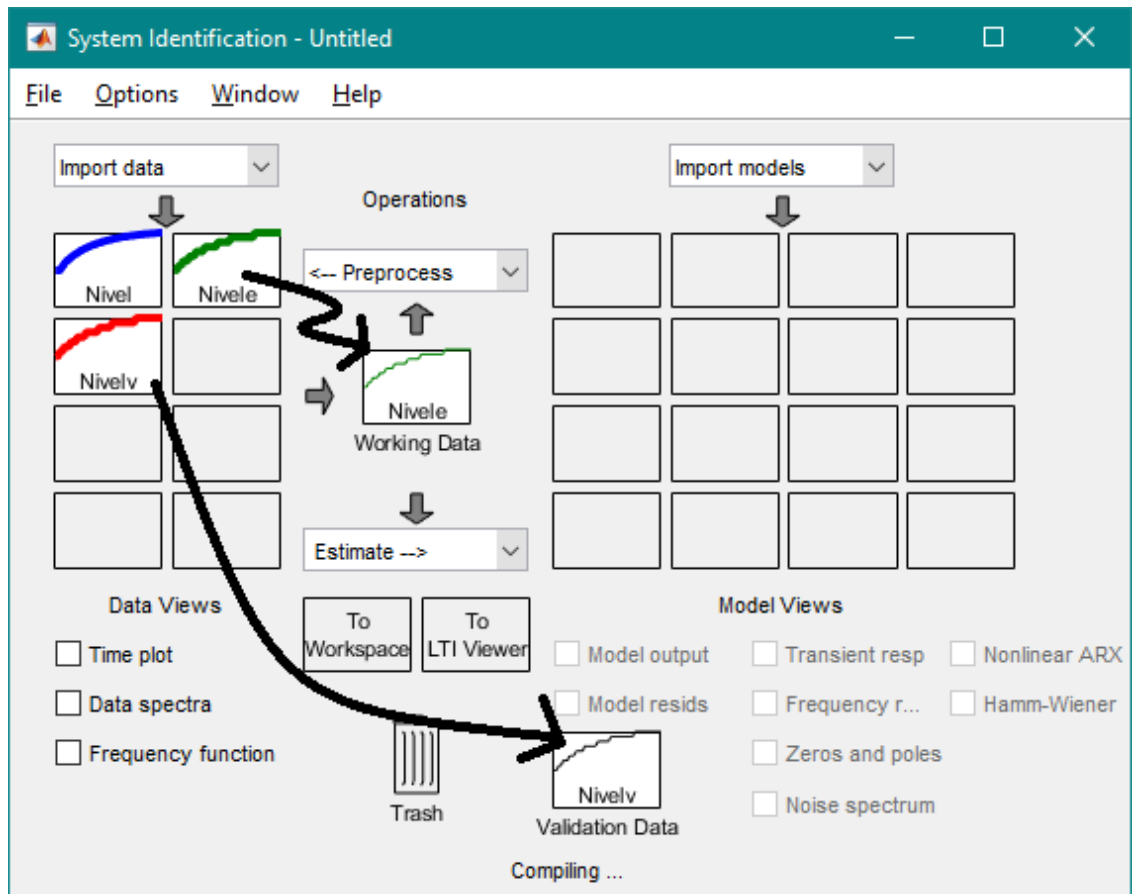
- ✓ Paso 9: Ahora se debe seleccionar el rango, haciendo clic izquierdo con el mouse en el despliegue que aparece en “Preprocess” y presionar en “Select Range”. Al realizar esta acción, se abrirá un cuadro de dialogo llamado “Select Range: u1->y1”, el cual posee las gráficas de entrada u1 (que va de 0 a 0,1 en el eje y) y salida y1 (que va de -37 a 3 en el eje y), como también cuenta con un menú de opciones. En la gráfica de salida y1 se debe seleccionar un rango, manteniendo presionado el clic izquierdo del mouse y soltarlo cuando ya se tenga el rango. Luego se debe presionar en “Insert” dos veces, la primera para generar el modelo y la segunda para validarlo. Luego, se debe cerrar la ventana presionando en “Close”. Para apreciar el rango elegido se debe observar la figura 3-14, mostrada a continuación.



Fuente: Elaboración propia, realizada en MatLab R2019a.

Figura 3 - 14. Seleccionando el rango.

- ✓ Paso 10: Se debe tomar el "Nivel e", que es la planta para generar el modelo y arrastrarlo hasta donde dice "Working Data". También, se debe tomar "Nivel v", que es la planta de validación y arrastrarlo hasta donde dice "Validation Data". Este procedimiento, se puede observar a través de la figura 3-15, mostrada a continuación.



Fuente: Elaboración propia, realizada en MatLab y etiquetada en Paint.

Figura 3 - 15. Generando y validando el modelo.

- ✓ Paso 11: Luego, se debe estimar el modelo, haciendo clic izquierdo del mouse en el desplegable llamado "Estimate" para presionar en "Process Models". Al realizar esta acción, se abre un cuadro de dialogo llamado "Process Models", donde se debe dejar activo en la sección llamada "Transfer Function", la opción "Delay" (para conocer el valor que se necesita tener) y luego hacer clic en "Estimate" para obtener finalmente los valores de la función de transferencia, es decir, la constante de tiempo o tau (τ), la ganancia (K) y el tiempo de retardo (T_d). Por lo tanto, para conocer estos valores en el cuadro de dialogo llamado "Process Models", se debe observar la figura 3-16, mostrada a continuación.

Process Models

Transfer Function

$$\frac{K \exp(-T_d s)}{(1 + T_{p1} s)}$$

Poles

1 ▾ All real ▾

☐ Zero
☒ Delay
☐ Integrator

Par	Known	Value	Initial Guess	Bounds
K	☐	30.3467	Auto	[-Inf Inf]
Tp1	☐	20.5525	Auto	[0 20921.25]
Tp2	☐	0	0	[0 Inf]
Tp3	☐	0	0	[0 Inf]
Tz	☐	0	0	[-Inf Inf]
Td	☐	0	Auto	[0 30]

Initial Guess

☒ Auto-selected
☐ From existing model:
☐ User-defined

Disturbance Model: None ▾ Initial condition: Auto ▾

Focus: Simulation ▾ Covariance: Estimate ▾

☐ Display progress

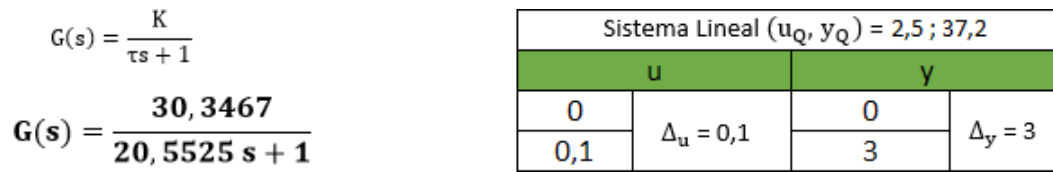
Name: PID

1: Valor de la ganancia. | 2: Valor de la constante de tiempo o tau en segundos. | 3: Tiempo de retardo en segundos.

Fuente: Elaboración propia, realizada en MatLab y etiquetada en Paint.

Figura 3 - 16. Valores de ganancia, constante de tiempo o tau y retardo.

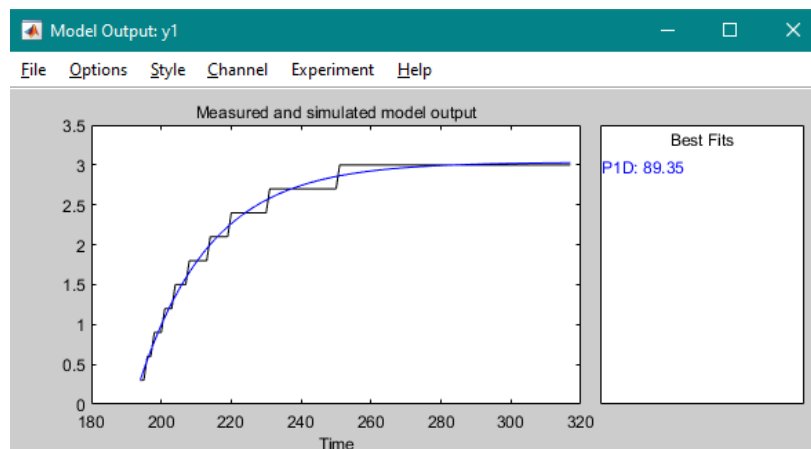
Tal y como se puede observar de la figura 3-16, el tiempo de retardo o T_d es cero, por lo que se puede denominar como un sistema de primer orden normal, donde el valor de la ganancia en continua (K) es 30,3467, la constante de tiempo (τ) o T_{p1} , es de 20,5525 segundos y el tiempo de retardo (T_d) es de 0 segundos. Por lo tanto, teniendo estos datos es posible armar la función de transferencia quedando como se muestra a través de la figura 3-17, mostrada a continuación.



Fuente: Elaboración propia.

Figura 3 - 17. Función de transferencia.

Resulta posible obtener también el porcentaje de asertividad. Para ello, en el recuadro llamado “System Identification” solo basta con activar la opción llamada “Model Output” para que se abra una ventana llamada “Model Output: y1” que muestra un gráfico y la información de asertividad, tal y como se muestra en la figura 3-18, mostrada a continuación.



Fuente: Elaboración propia, realizada en MatLab R2019a.

Figura 3 - 18. Porcentaje de asertividad.

El porcentaje de asertividad en el cálculo de la función de transferencia que muestra la figura 3-18 es de 89,35%, lo que es óptimo para el diseño del controlador para el sistema de nivel.

CONCLUSIÓN

En conclusión, los resultados del proyecto son muy satisfactorios. Los objetivos propuestos, se cumplieron satisfactoriamente, ya que se logró finalmente implementar la estrategia de control ON/OFF con histéresis programada en LabVIEW y Arduino UNO, para regular el sistema de control de nivel en Factory I/O. Los resultados de la implementación del proyecto, fue óptima, ya que dejan abierta la posibilidad de diseñar a futuro el controlador más adecuado para el sistema de nivel, por ejemplo, un controlador PID.

Para la comunicación entre la plataforma de desarrollo físico Arduino UNO y el software de ingeniería LabVIEW, no fue necesario hacer uso del software “Arduino IDE” como se acordó en un comienzo, puesto que, el “toolkit LINX” de LabVIEW llamado “MakerHub” demostró grandes ventajas en comparación a otros tipos de comunicación, como lo son “LIFA” y “NI VISA”.

El proyecto fue ventajoso, porque permitió un mayor acercamiento hacia la programación en lenguaje G de LabVIEW, la cual se basa simplemente en bloques funcionales, pudiendo realizar grandes proyectos que se pueden aplicar en la industria, ya que, hoy en día ha ido en aumento el uso de la automatización en procesos. Por lo tanto, es muy importante familiarizarse con este tipo de programación para hallar soluciones creativas y efectivas a los diversos problemas que se presentan en la industria.

La interfaz HMI de información proporcionó al usuario u operador una visualización, manipulación y medición del nivel de una manera sencilla, permitiendo observar los comportamientos de las señales de control, variables manipuladas y de la señal de respuesta o nivel.

Durante este proyecto se logró validar, integrar y perfeccionar conocimientos adquiridos en las asignaturas de microprocesadores y microcontroladores, instrumentación y automatización, control de procesos y análisis de sistemas lineales aprendidos a lo largo de todo el proceso académico.

BIBLIOGRAFÍA

FACTORY I/O. Servidor Modbus TCP / IP. [En línea]
<<https://docs.factoryio.com/manual/drivers/modbus-server/>>[Consulta: 16 de abril de 2021].

Wikipedia, la enciclopedia libre. Arduino Uno. [En línea]
<https://es.wikipedia.org/wiki/Arduino_Uno>[Consulta: 15 de abril de 2021].

Wikipedia, la enciclopedia libre. Arduino. [En línea]
<<https://es.wikipedia.org/wiki/Arduino>>[Consulta: 15 de abril de 2021].

Wikipedia, la enciclopedia libre. LabVIEW. [En línea]
<<https://es.wikipedia.org/wiki/LabVIEW>>[Consulta: 14 de abril de 2021].

YouTube. Guelis Montenegro Zamora, Identificación de Sistemas con Matlab. [En línea]
<<https://www.youtube.com/watch?v=HvCcFcMEL7E>>[Consulta: 29 de julio de 2021].

YouTube. Guelis Montenegro Zamora, Parte 2: Sistema de control de nivel de un estanque: Función de transferencia en punto de operación. [En línea]
<<https://www.youtube.com/watch?v=qIQgY07CZHA>>[Consulta: 29 de julio de 2021].

YouTube. Guelis Montenegro Zamora, Parte 1: Sistema de control de nivel de un estanque: ¿Sistema lineal o no lineal?. [En línea]
<<https://www.youtube.com/watch?v=4V6BK6JCekw&t=218s>>[Consulta: 29 de julio de 2021].

YouTube. LabVIEW Para Todos, Instalación de MAKERHUB-LINUX en LabVIEW. [En línea]
<https://www.youtube.com/watch?v=g0VtOOcxy_k>[Consulta: 15 de abril de 2021].