INSTITUTO TECNOLÓGICO DE TUXTLA GUTIÉRREZ

DEPARTAMENTO DE ELECTRICA Y ELECTRONICA

INGENIERIA ELECTRONICA

RESIDENCIA PROFESIONAL

INGENIERÍA EN ELECTRÓNICA

"DISEÑO DE UN SISTEMA DE SUPERVISIÓN INDUSTRIAL PARA UNA RED HIDRÁULICA PILOTO"

LABORATORIO DE DISEÑO ELECTROMECÁNICO DE LA MAESTRÍA EN CIENCIAS EN INGENIERÍA MECATRÓNICA

PRESENTA	N° DE CONTROL
RIOS SANTIAGO OMAR ALONSO	13270885
ROSALES POSADA ELVIS ALBERTO	13270886

ASESOR

DR. FRANCISCO RONAY LOPEZ ESTRADA

TUXTLA GUITIÉRREZ, CHIAPAS.

A 6 DE JUNIO DE 2018

Contenido

CAPITULO 1: ASPECTOS GENERALES 4
Introducción 4
Planteamiento del problema 6
Justificación 6
Antecedentes 6
Objetivo general 7
Objetivos específicos 7
Alcance y limitaciones 7
Alcance: 7
Limitaciones: 7
Caracterización del área en el que se participo 7
Antecedentes de la empresa 7
Misión, visión y valores 9
Descripción del área dónde se realizó el proyecto 9
Impacto 10
Caso de estudio 10
CAPITULO 2: FUNTAMENTOS TEORICOS 12
Marco teórico 12
Adquisición de datos (DAQ) 12
Sensor de Caudal 15
Sensor de Presión 16
Electroválvulas 18
Módulo de relevadores 20
Transformador 21
Software necesario 22
CAPITULO 3: DESARROLLO 25
Cronograma de actividades propuesto 25
Funcionamiento del sistema de fugas de serpentín. 25
Materiales que utilizaron para las primeras pruebas 29
..... 30
Desarrollo de la instrumentación y programación del sistema 30

.....32
..... 32
Conclusión 52
Referencias..... 52
Anexo a programación de MatLab 55

CAPITULO 1: ASPECTOS GENERALES

Introducción

Una red hidráulica es un conjunto de tuberías y accesorios que sirven para transportar los diferentes tipos de fluidos, los cuales deben cumplir con un mínimo de presión para su transportación de un lugar a otro. Existen diferentes tipos de redes hidráulicas en las cuales se transportan diferentes tipos de fluidos, principalmente el agua, que es un líquido vital para la sobrevivencia del ser humano, como también la transportación de los fluidos de PEMEX tales como son los gases, el diésel, la gasolina entre otros. Para poder transportar estos fluidos de una manera segura existen normas que exigen utilizar el material adecuado de acuerdo al tipo de fluido que se transportara. Estos materiales son tubería de PVC, tubería de cobre, tubería galvanizado, tubería de fibrocemento, tubería de hierro fundido, tubería de acero, entre otros [Comisión Nacional del Agua]. Los fluidos mencionados anteriormente, son de suma importancia transportarlos de una manera segura ya que ocasionan daños al medio ambiente y a la sociedad por la fuga de los líquidos y gases, por ejemplo, la fuga de agua potable en una ciudad (Figura 1) ocasiona daño a las vías públicas, ya que el agua que se fuga provoca deterioro en las calles, además de generar focos de infecciones debido a los encharcamientos que se forman, provocando grandes pérdidas económicas y litros de agua desperdiciados que se escapan sin control. Por lo consiguiente, PEMEX ocasiona daños al medio ambiente ya que las fugas provocan explosiones, lo cual contaminan a la flora y fauna que está cerca de las fugas, además de las hectáreas de cultivos de los campesinos que tienen terrenos por donde pasan los ductos, por consiguiente esto provoca pérdidas millonarias, así como poner en riesgo la vida de muchas personas [Greenpeace México].

Figura 1.1 Calle de Tuxtla Gutiérrez, Chiapas.

De acuerdo a lo anterior, las fugas hidráulicas provocan grandes pérdidas millonarias y daños al medio ambiente, como es el caso del agua potable en México, donde la OCDE (Organización para la Cooperación y el Desarrollo Económico) realizó una encuesta a nivel mundial en la que mide el porcentaje que cada ciudad pierde de su agua potable, donde México es el país que más agua pierde a nivel mundial encabezando la lista con 9 ciudades, Tuxtla Gutiérrez es la ciudad que más agua pierde con alrededor del 70%; le sigue San Luis Potosí con el 50%; la Ciudad de México con más del 40%, y Chihuahua, Toluca, Querétaro, Culiacán, Acapulco y Hermosillo son las ciudades que menos pierden [El País]. Por otra parte PEMEX es el que más sufre con las pérdidas millonarias por causa de las fugas en sus ductos, además de la contaminación que ocasiona al medio ambiente, en los últimos 6 años PEMEX ha perdido 100 mil mdp [Animal Político] por robo de combustible y fugas en sus ductos, en volumen, lo perdido asciende a 14 mil 652 millones de litros, que implica hasta 250 pipas de combustible cada día, lo cual indica que las fugas son demasiadas, además de las perforaciones para el robo de combustible, por consiguiente el daño al medio ambiente es de manera impactante e irreparable, ya que la flora y la fauna que se encuentra cerca de estas fugas las cuales se convierten en explosiones sufren daños que provocan la desaparición de algunas especies en el área, además de poner en riesgo la vida de los seres humanos que viven cerca de los ductos, ocasionando daños a la salud [Greenpeace México].

Existen trabajos que han intentado solucionar el problema de las fugas, por ejemplo; en el [2015] proponen determinar la posición de múltiples fugas a partir de una onda de presión transitoria provocada por un cambio controlado en la dinámica del fluido, la principal ventaja de esta prueba es que solo se requiere la medición de la presión.

De acuerdo a los problemas que generan las fugas en las redes hidráulicas, es de alta importancia desarrollar técnicas que ayuden a disminuir la problemática de las fugas de fluidos en las redes hidráulicas, es necesario la construcción y la instrumentación automatizada de una planta piloto que se acerque a las características reales de una red hidráulica industrial, para poder realizar los experimentos necesarios que ayuden a resolver estos problemas, además de obtener resultados más precisos y confiables. De este modo, se podrá probar la precisión de los algoritmos que se esperan desarrollar a través de diferentes métodos y técnicas para detectar las fugas en las redes hidráulicas, también ayudar a disminuir las fugas de agua en la red de distribución de agua potable de Tuxtla Gutiérrez y reducir el porcentaje de pérdida a nivel nacional.

Por consiguiente, el desarrollo de esta tesis propone automatizar una red hidráulica de dos niveles en el Instituto Tecnológico de Tuxtla Gutiérrez de aproximadamente 200 m de longitud (100 m por cada nivel) con una configuración de tipo serpentín horizontal, donde contara con 4 sensores, dos de entrada, uno de presión y uno de flujo y dos de salida, uno de presión y uno de flujo, los cuales medirán la presión de entrada y la presión de salida, del mismo modo medirán el flujo de entrada y el flujo

de salida, se contara con una tarjeta de adquisición de datos la cual obtendrá la lectura de los sensores, tendrá 4 electroválvulas que simularan las fugas reales de una red, del mismo modo habrá una DAQ NI USB 6002, que se programara a través de Matlab para el control de un módulo de relevadores los cuales activaran unos transformadores de 24 VCA que estos a su vez activaran a las electroválvulas de nuestra planta piloto.

Planteamiento del problema

Las fugas de agua son el mayor problema que existe en una ciudad ya que provoca afectaciones a la población por desabastecimiento de este vital liquido que es necesario para la vida del ser humano, y en muchas ciudades se fugan miles de litros de agua por día, a causa de las fugas que existen en los tubos que distribuyen el agua por la ciudad. Por lo tanto uno de los mayores problemas de estas fugas es que no se pueden reparar de una manera inmediata ya que la mayoría de las veces se desconoce la ubicación exacta de dicha fuga, por lo tanto los encargados de dar mantenimiento y reparación de las fugas se les complica detectar la ubicación exacta de la fuga para así poder repararlos y excavar una sola vez en la posición correcta de la fuga, de lo contrario se excavaría varias veces hasta encontrar la fuga.

Justificación

Este proyecto está diseñado para detectar fugas en las redes de agua, con el objetivo de ayudar a las ciudades de tener un mejor abastecimiento de agua a su población, ya que al existir fugas en las redes de distribución ocasiona que no todas las familias les llegue el servicio del agua, es un proyecto que ayudaría a disminuir el porcentaje de perdida de agua en la ciudad.

Además este proyecto puede ser utilizado para detectar fugas en los ductos de Pemex, porque está diseñado para detectar diferentes tipos de fugas de los diferentes fluidos que existen.

Antecedentes

En los últimos años se han desarrollado técnicas que intentan resolver el problema de las fugas. Como se ha mencionado anteriormente los problemas que causan estos fenómenos en las redes hidráulicas son prácticamente devastadoras en una región o en una industria, ya que la pérdida de producto ocasiona todo tipo de pérdidas que afectan el desarrollo de una empresa o una ciudad. Por esto, el impacto que se genera ha motivado a un enfoque profundo acerca de esta temática, por ejemplo, (Sangolqui, abril del 2015) por medio de una investigación se fundamentó resolver un histórico problema en una inadecuada gestión del sistema de agua potable, por el cual los usuarios llevaban años sufriendo por el abastecimiento de este vital liquido debido a la baja presión del sistema ocasionada por las fugas, el sistema cuenta con tres partes, la primea se divide en tres zonas, zona crítica, zona no crítica y zona media; la segunda parte es el plan de medición y recuperación de pérdidas y la tercera parte es el plan de renovación integral del

sistema dañado. En (2017, jornadas de automática) propusieron un modelo basado en el principio de conservación de la masa en el que se incluyen las fugas como señales aditivas, donde se plantean observadores tipo PI para la estimación de las fugas. (Quito, abril 2018) la realización de un software para la simulación en detección de fugas en un ducto de características similares a los instalados en el sector petrolero ecuatoriano, el cual cuenta con los parámetros necesarios para establecer un escenario de transporte y como resultado muestra en tiempo real el comportamiento de las variables tanto en la entrada como en la salida del ducto.

Objetivo general

Instrumentar con sensores de caudal y flujo una red hidráulica piloto que se construirá en el laboratorio de hidráulica del doctorado en ciencias de la ingeniería

Objetivos específicos

- Apoyar con la construcción de la red hidráulica, la cual constará con al menos 4 nodos para tomas laterales o retroalimentación
- Realizar un estudio estructural de la red para determinar las ubicaciones óptimas de los sensores. Estas posiciones dependen de ubicaciones de los nodos, distancia de codos, y de la bomba hidráulica.
- Instalar sensores de presión y flujo industriales en los puntos seleccionados y conectar estos sensores aun sistema SCADA.
- Diseñar un sistema de monitoreo en tiempo real de los diferentes sensores de caudal y presión a lo largo de la red hidráulica.
- Instalar válvulas electrónicas en puntos aleatorios de la red para simular fugas.

Alcance y limitaciones

Alcance:

- Interfaz en MATLAB de los sensores y actuadores.
- Se obtendrá mediciones de presión y de flujo utilizando sensores, como también activación de las electroválvulas del sistema.

Limitaciones:

- Sensores industriales de corriente.
- Terminación de la residencia.

Caracterización del área en el que se participo

Antecedentes de la empresa

El Instituto Tecnológico de Tuxtla Gutiérrez, ubicado en carretera Panamericana km.1080, brinda servicios educativos de calidad certificada, con la misión de formar de manera integral a profesionistas de excelencia en el campo de la ciencia y la

tecnología con actitud emprendedora, respeto al ambiente y apego a los valores éticos.

Además, es una institución de excelencia en la educación superior tecnológica del sureste, comprometida con el desarrollo sustentable de la región, el cual es uno de los ejes transversales de su oferta educativa, tanto a nivel licenciatura como posgrado.

Así mismo, trabaja con sistemas de gestión ambiental certificados, en donde los estudiantes aplican sus conocimientos en función del cuidado del ambiente: aire, agua, suelo, flora, fauna y seres humanos.

Entre las principales líneas estratégicas de educar con responsabilidad ambiental se encuentran los Sistemas de Gestión de Calidad, Sistema de Gestión Ambiental, Programas Académicos Acreditados por COPAES y por CONACYT; se trata de transformar conciencias, evolucionar culturas, sensibilizar a la población y mostrar responsabilidad ante las acciones que realiza en pro de la conservación de la biodiversidad del planeta.

Como parte de la oferta educativa de esta institución, se encuentra el departamento de Ingeniería Eléctrica y Electrónica el cual se encarga de coordinar distintas actividades académicas que permiten a los alumnos poner en práctica los conocimientos adquiridos dentro de las aulas, además de ayudarlos a desarrollar habilidades y aptitudes que les permitan resolver problemas de manera eficiente.

De igual manera, es importante mencionar que este departamento cuenta con el apoyo de distintos laboratorios dentro de la institución, uno de ellos es el Laboratorio de Ingeniería Electrónica en el cual los alumnos refuerzan los conocimientos teóricos adquiridos en clases mediante prácticas que les permiten desarrollar capacidades y habilidades para su desenvolvimiento futuro como profesionistas.

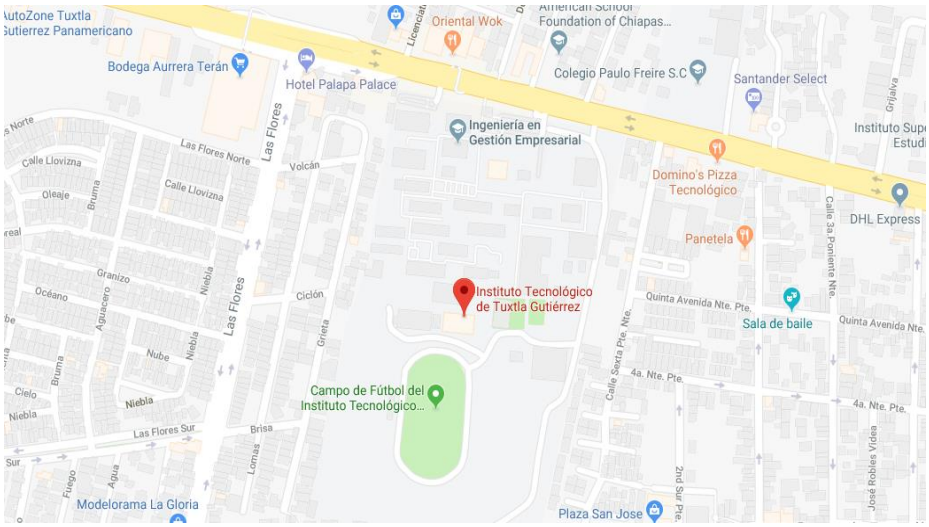


Figura 1.2 Localización donde se realizó el proyecto

Misión, visión y valores

- **Misión**

Formar de manera integral profesionistas de excelencia en el campo de la ciencia y la tecnología con actitud emprendedora, respeto al medio ambiente y apego a los valores éticos.

- **Visión**

Ser una Institución de Excelencia en la Educación Superior Tecnológica del Sureste, comprometida con el desarrollo socioeconómico sustentable de la región.

- **Valores**

El ser humano, el espíritu de servicio, el liderazgo, el trabajo en equipo, la calidad, el alto desempeño y respeto al medio ambiente.

Descripción del área dónde se realizó el proyecto

En el Instituto Tecnológico de Tuxtla Gutiérrez, como parte del apoyo académico que se brinda, en el laboratorio de detección de fugas, hidráulicas, de la maestría en ciencias en ingeniería mecatronica, el laboratorio cuenta con los materiales e instrumentos necesarios para llevar a cabo prácticas e investigación como también hacer pruebas de captura de datos de este sistema de fugas.

Figura 1.3 Laboratorio de hidráulica del edificio I del Instituto Tecnológico de Tuxtla Gutiérrez.

Impacto

El proyecto tiene como fin la obtención de datos de fugas de forma más precisa utilizando electroválvulas y utilizando sensores industriales que nos ayudaran con mejor la obtención de datos con el menor margen de error posible.

Los beneficios de la construcción de esta red hidráulica son muchas, una de ellas es detectar fugas de agua en la redes de tubería, ya que por lo general Tuxtla tiene el primer lugar de desperdicio de agua, debido a este problema se generan grandes pérdidas económicas a los organismos operadores del sistema de distribución. También este proyecto se basa en el beneficio de detectar fugas en las redes de distribución de Pemex, ya que las fugas de este tipo sufren más perdidas económicas, se pueden producir explosiones por una fuga como también la contaminación del suelo y aire.

Caso de estudio

Este trabajo está caracterizado en la construcción de una red de fugas, haciendo una interfaz gráfica de los sensores que se le intentaran implementar. Actualmente hay tres laboratorios de sistemas de fugas en todo el país, el primero se encuentra en la UNAM en el estado de México, el segundo se encuentra en el Cinvestav que se encuentra en Guadalajara y el tercero que se encuentra en nuestras instalaciones del Instituto Tecnológico de Tuxtla Gutiérrez del estado de Chiapas.

El sistema de fugas cuanta de un serpentín que contiene dos sensores de presión y dos sensores de caudal, en el cual un sensor de presión y un sensor de flujo se encuentran al inicio de la tubería, mientras que los dos sensores se encuentran al final. Contiene 2 bombas de agua una de 5 HP controlado por un variador de frecuencia y otro de ½ HP controlado independientemente, también contiene 4

válvulas mecánicas distribuidas aleatoriamente en la tubería, una fuente de alimentación y una PC.

CAPITULO 2: FUNTAMENTOS TEORICOS

Marco teórico

Adquisición de datos (DAQ)

La adquisición de datos (DAQ) es el proceso de medir con una PC un fenómeno eléctrico o físico como voltaje, corriente, temperatura, presión o sonido. Un sistema DAQ consiste de sensores, hardware de medidas DAQ y una PC con software programable. Comparados con los sistemas de medidas tradicionales, los sistemas DAQ basados en PC aprovechan la potencia del procesamiento, la productividad, la visualización y las habilidades de conectividad de las PCs estándares en la industria proporcionando una solución de medidas más potente, flexible y rentable.

Partes de un sistema DAQ

Figura 2.1 Sistema DAQ

En el sistema de red de fugas se implementaron dos tarjetas de adquisición de datos que son NI 9203 y NI USB-6002.

NI 9203

- 8 canales, entrada de corriente de 200 kS / s
- Rangos de entrada programables de ± 20 mA, 0 mA a 20 mA; 16 bits resolución
- Calibración trazable según NIST
- Conectividad con terminal de tornillo o terminal de resorte
- 250 Vrms, aislamiento de banco CAT II
- Rango de funcionamiento de -40 ° C a 70 ° C, 5 g de vibración, choque de 50 g

El NI 9203 es un módulo DAQ de la Serie C con 8 canales de entrada de corriente analógica para aplicaciones de monitoreo y control de alto rendimiento. Cuenta con rangos de entrada programables de ± 20 mA o 0 mA a 20 mA, resolución de 16 bits y una velocidad máxima de muestreo de 200 kS / s. Para protegerse contra transitorios de señal, el NI 9203 incluye una barrera de doble aislamiento de tierra de canal a tierra (aislamiento de 250 Vrms) para la inmunidad de seguridad y ruido

Figura 2.2 NI 9203

Circuito de entrada

Las señales de entrada son almacenadas en búfer, acondicionadas y muestreadas por un único ADC de 16 bits. El módulo protege cada canal contra sobretensiones. Consulte la sección Especificaciones para obtener información sobre la protección contra sobretensiones.

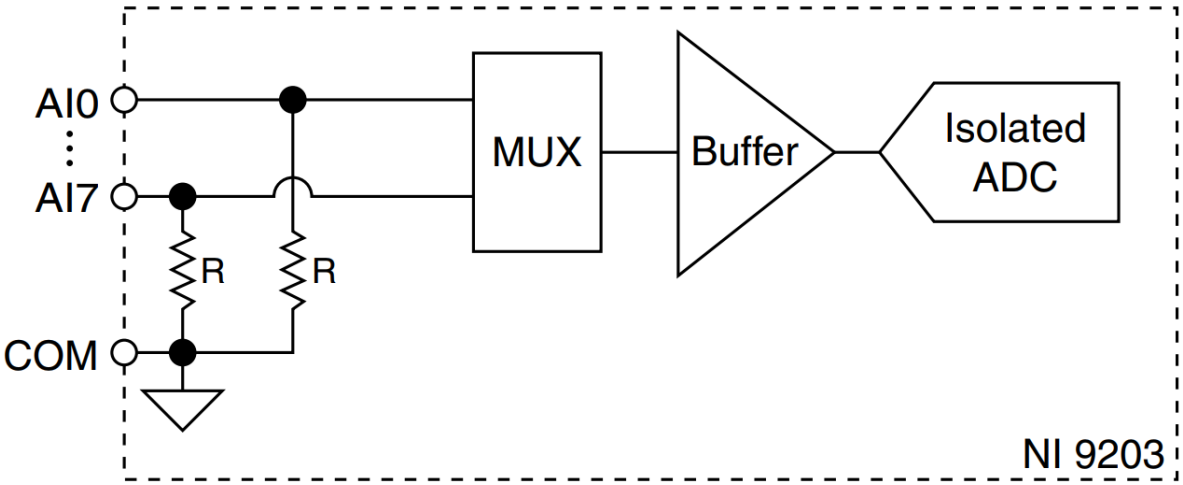


Figura 2.3 Circuito de entrada del NI 9203

NI USB-6002

El NI USB-6002 es un dispositivo USB de velocidad completa que proporciona ocho canales de entrada analógica (AI) de terminación única, que también pueden configurarse como cuatro canales diferenciales. También incluye dos canales de

salida analógica (AO), 13 canales de entrada / salida digital (DIO) y un contador de 32 bits.

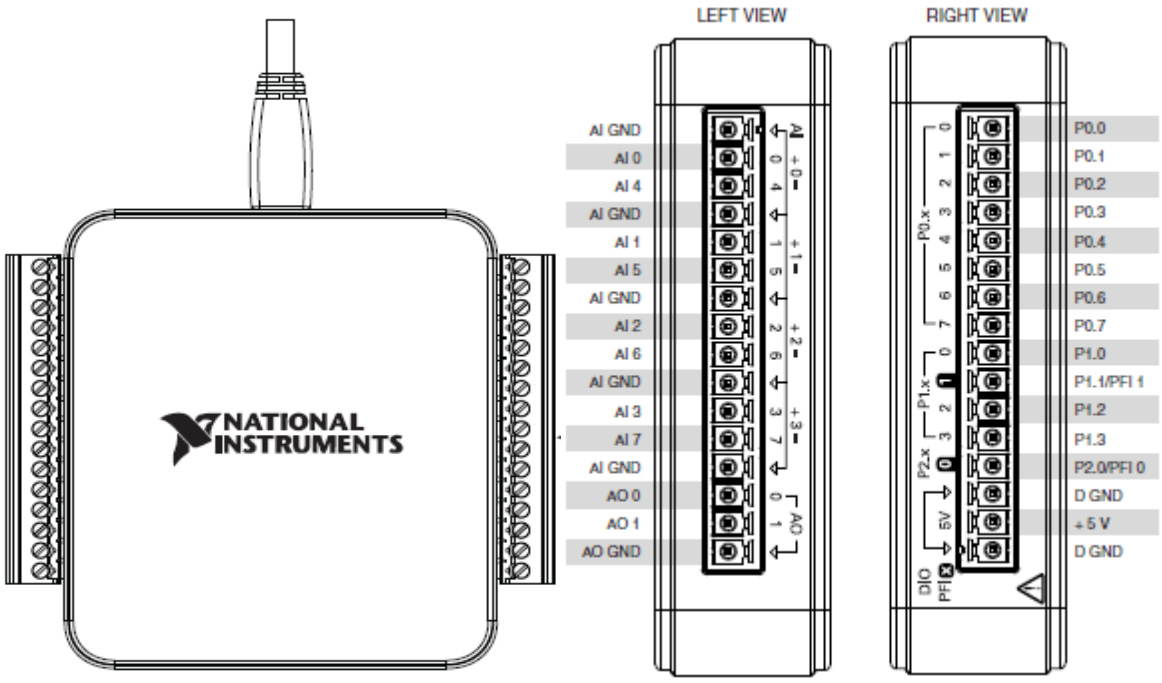


Figura 2.4 NI USB-6002

Descripciones de señales

Señal	Referencia	Descripción
AI GND	—	Tierra de entrada analógica: el punto de referencia para las mediciones de entrada analógica de terminación única.
AI <0..7>	Varía	Canales de entrada analógica: para mediciones de una sola terminación, cada señal corresponde a un canal de voltaje de entrada analógica. Para mediciones diferenciales, AI 0 y AI 4 son las entradas positiva y negativa del canal de entrada analógica diferencial 0. Los siguientes pares de señal también forman canales de entrada diferenciales: AI <1,5>, AI <2,6> y AI <3, 7>.
A la GND	—	Salida analógica a tierra: el punto de referencia para la salida analógica.
AO <0,1>	A la GND	Canales de salida analógica: estos terminales suministran salida de voltaje.
P0. <0..7>	D GND	Canales de E / S digitales del puerto 0: puede configurar cada señal individualmente como entrada o salida.
P1. <0..3>	D GND	Canales de E / S digitales del puerto 1: puede configurar cada señal individualmente como entrada o salida.

P2.0	D GND	Puerto 2 Canal de E / S digital: puede configurar esta señal como entrada o salida.
PFI <0,1>	D GND	Interfaz de función programable o canales de E / S digitales: una entrada de contador de borde o entrada de disparo digital.
D GND	—	Digital Ground: el punto de referencia para señales digitales.
+5 V	D GND	5 V Fuente de energía: proporciona 5 V de potencia de hasta 150 mA.

Sensor de Caudal

Uno pensaría que la medición del caudal de un fluido no debiera ser algo complicado, pero lo es. Diferentes presiones, temperaturas, viscosidades, conductividad, abrasividad, composición, estado de agregación, etc. hace que para cada problema haya solo un número limitado (o uno... o ninguno) de soluciones.

Constructivamente los caudalímetros electromagnéticos se pueden dividir en dos partes, el sensor o elemento primario y el computador de caudales o elemento secundario, en lo que sigue describiremos el sensor primario.

Debido a que carece de partes móviles o mecánicas, es un equipo que no necesita mantenimiento y el hecho de que no ponga obstáculos al fluido lo hace ideal para sistemas en los que la perturbación es un factor limitante. Es totalmente inmune a vibraciones. Posee un recubrimiento interno aislante (ya que la conductividad de la cañería afectaría la medición) que puede ser de Neoprene, Teflon, PTFE, u otros similares, lo que permite que el fluido a medir sea realmente hostil (PH muy alto/bajo, temperaturas extremas -50°C a 200°C, abrasividad mecánica) sin que afecte la durabilidad del equipo.

La versatilidad del principio de funcionamiento de este caudalimetro le permite poder diferenciar el sentido de circulación del fluido, siendo una de las pocas tecnologías de medición no solo que le es indiferente a la hora de medir caudal, sino que las puede diferenciar en el caso que le sea de interés al usuario.

Un aspecto a destacar en este tipo de caudalímetros, es el intervalo de medición ya que, para un modelo determinado, el caudal máximo llega a ser 100 veces superior al mínimo.
El error de medición de este tipo de caudalímetros ronda el 0.5% y a pedido del cliente puede ser de 0.2%.

Los sensores de flujo, también conocidos como "detector de flujo" o "interruptor de caudal" son equipos para monitoreo de fluidos en tuberías y funcionan con el desplazamiento de un pistón magnético que indica el aumento o disminución del flujo de líquido, accionando el contacto de un interruptor de láminas (reed switch). El pistón es controlado por un resorte y regresa a la posición inicial cuando no hay fluido, incluso si hay presión en la tubería.

El Sensor de caudal o flujo que se instrumentó en el sistema fue el GF Signet 2537 Paddlewheel sus características son las siguientes:

- Salida digital (S3L) o de 4 a 20 mA, o interruptor de flujo o salida de pulso (multifunción)
- Permite hasta seis sensores para el Controlador Signet 8900
- Capacidades de flujo bajo hasta 0.1 m / s (0.3 pies / s)
- Materiales mojados de polipropileno o PVDF
- Interfaz de usuario incorporada para la configuración en el sitio
- Baja potencia y alta resolución

Un caudalímetro de rueda de paletas ofrece la capacidad de medir el flujo de una gran variedad de compuestos. Los caudalímetros de rueda de paletas también se conocen como medidor de flujo de inserción o medidor de flujo en línea, el medidor de paleta tiene forma de hélice y es compacto. Con sensores computarizados, cada medidor de rueda de paletas se prueba y se muestra su precisión dentro del uno por ciento de una lectura de rango de escala completa. Estos parámetros se imprimen y se empaquetan y con el medidor de rueda de paletas comprado. El medidor se puede instalar vertical u horizontalmente y no afectará el rendimiento del dispositivo. Además, la herramienta puede ser de batería o accionada por CA, lo que la hace altamente flexible. Un medidor de rueda de paletas es ideal para aplicaciones en las que se necesita un alto grado de precisión, pero el costo es un factor importante.

Figura 2.5 Sensor de flujo GF Signet 2537 Paddlewheel

Sensor de Presión

Un transductor de presión, a veces llamado transmisor de presión, es un transductor que convierte presión en una señal eléctrica analógica. Aunque hay varios tipos de transductores de presión, uno de los más comunes es el transductor de base de calibrador de tensión. La conversión de presión en una señal eléctrica se logra

mediante la deformación física de medidores de tensión que están unidos al diafragma del transductor de presión y cableados a una configuración de puente de Wheatstone. LA presión APLICADA AL transductor de presión produce una deflexión del diafragma que introduce tensión en los calibradores. La tensión producirá un cambio en la resistencia eléctrica proporcional a la presión.

La salida eléctrica de los transductores de presión.

Los Transductores de presión en general están disponible con tres tipos de salida eléctrica: milivoltios, voltaje amplificado y 4-20mA. A continuación está un resumen de las salidas y cuándo es su mejor uso.

Transductores de presión con salida en milivoltios.
Un transductor con salida en milivoltios es normalmente el transductor de presión más económico. La salida del transductor de milivoltios es nominalmente alrededor de 30mV. La salida real es directamente proporcional a la energía de entrada o excitación del transductor de presión. Si la excitación fluctúa la salida también cambiará. Debido a esta dependencia del nivel de excitación, se sugieren fuentes de energía reguladas para usar con transductores de milivoltios. Debido a que la señal de salida es tan baja, el transductor no se deberá ubicar en un entorno eléctricamente ruidoso. Se deberán mantener relativamente cortas las distancias entre el transductor y el instrumento.

Transductores de presión de salida de voltaje.
Los transductores de presión de salida de voltaje incluyen un acondicionador de señales incorporado que proporciona una salida mucho más alta que un transductor de milivoltios. La salida normalmente es 0-5 Vcc o 0-10 Vcc. Aunque es específica para un modelo, la salida del transductor no es normalmente una función directa de la excitación. Esto significa que con frecuencia las fuentes de energía no reguladas son suficientes siempre y cuando caigan dentro de un rango de energía especificado. Debido a que tienen una salida de nivel más alto, estos transductores no son tan susceptibles al ruido eléctrico como los transductores de milivoltios y por lo tanto se pueden usar en muchos más entornos industriales.

Transductores de presión de salida de 4-20 mA.
Estos tipos de transductores también se n como transmisores de transmisor. Puesto que una señal de 4-20mA es menos afectada por el ruido eléctrico y la resistencia en los alambres de señal, estos transductores se usan cuando la señal debe transmitirse por distancias grandes. Con frecuencia estos transductores se usan en aplicaciones en la que el alambre debe tenderse 1000 pies o más.

Los sensores de presión que se usaron en el sistema son el Transmisor de presión de manómetro en línea con el modelo de EJA530E de la marca Yokogawa.

El transmisor de presión absoluta y de alto rendimiento EJA510E y EJA530E tiene un sensor resonante de silicio monocristalino y es adecuado para medir la presión del líquido, el gas o el vapor. EJA510E y EJA530E emiten una señal de CC de 4 a 20 mA correspondiente a la presión medida. También cuenta con respuesta rápida, configuración remota y monitoreo a través de comunicaciones BRAIN o HART y autodiagnóstico.

- Características del EJA530E:
- 0.055% de Precisión (0.04% opción)
- 0.1% de Estabilidad por 10 años
- 90 milisegundos de Tiempo de Respuesta
- Certificación Exida y TUV SIL2 / SIL3
- Ajuste de Parámetro Local (LPS)

Figura 2.6 EJA530E Transmisor de presión de manómetro en línea

Electroválvulas

Las electroválvulas o válvulas solenoides son dispositivos diseñados para controlar el flujo (ON-OFF) de un fluido. Están diseñadas para poder utilizarse con agua, gas, aire, gas combustible, vapor entre otros. Estas válvulas pueden ser de dos hasta cinco vías. Pueden estar fabricadas en latón, acero inoxidable o pvc. Dependiendo del fluido en el que se vayan a utilizar es el material de la válvula.

En las válvulas de 2 vías, normalmente se utilizan las que funcionan con tres modalidades diferentes, dependiendo del uso que están destinadas a operar; pueden ser de acción directa, acción indirecta y acción mixta o combinada, además cada una de estas categorías puede ser Normalmente Cerrada (N.C.) o Normalmente Abierta (N.A.) , esto dependiendo de la función que va a realizar ya sea que esté cerrada y cuando reciba la señal a la solenoide abra durante unos segundos, o que esté abierta y cuando reciba la señal la solenoide corte el flujo.

Acción directa

El comando eléctrico acciona directamente la apertura o cierre de la válvula, por medio de un embolo.

La diferencia entre la válvula N.C. a la N.A. de acción directa es que, cuando la válvula N.C. no está energizada el embolo permanece en una posición que bloquea el orificio de tal manera que impide el flujo del fluido, y cuando se energiza la bobina el embolo es magnetizado de tal manera que se desbloquea el orificio y de esta manera fluye el fluido. La N.A. cuando la bobina no está energizada mediante la acción de un resorte el embolo se mantiene en tal posición que siempre está abierta y cuando se energiza la bobina la acción es hacia abajo empujando el resorte haciendo que cierre el orificio e impida que fluya el fluido.

Acción Indirecta

La característica principal de la válvula del tipo acción indirecta es que cuando recibe el comando eléctrico se acciona el embolo el cual permite a su vez como segunda acción, o acción indirecta, que el diafragma principal se abra o se cierre, en una acción indirecta. Esta serie de válvulas necesita una presión mínima para poder funcionar correctamente. También en esta serie de comando indirecto tenemos válvulas normalmente cerradas y válvulas normalmente abiertas.

Acción Mixta

En las válvulas de Acción Mixta o Combinada una característica es que no requieren una presión mínima como las de acción indirecta. Estas válvulas al igual que las de acción indirecta el comando de abertura se hace en 2 tiempos, primero se vacía la presión superior del diafragma grande y después, segunda acción, la presión de abajo del diafragma lo empuja para que se abra. Además el embolo está sujetado por medio de un resorte al diafragma grande y este resorte acelera la acción de la presión de abajo hacia arriba para abrir el mismo diafragma, esta es la segunda etapa de apertura.

Estas válvulas de acción mixta pueden ser ya sea normalmente abiertas o normalmente cerradas.

En las válvulas de acción mixta y de acción indirecta, los diafragmas que se utilizan dependen del material que vaya a fluir a través de ellas. Los diafragmas pueden ser de BUNA, VITON o TEFLON debido a que cada uno de estos diafragmas tiene ciertas características. Por ejemplo el diafragma de BUNA soporta temperaturas de (-10+90 C) y es recomendado para agua, aire, gas inerte. En el caso del VITON soporta temperaturas más altas (-10+140 C) y se recomienda para aceite ligero, gasolina, diésel. En el caso del TEFLON soporta temperaturas (-10+180 C) este se recomienda para vapor debido a la temperatura que puede soportar y que el teflón es más resistente.

La electroválvula que se instrumentó en el sistema fue un Rain Bird modelo 100–HV con las siguientes especificaciones:

Presión: 1,0 a 10,3 bares.
Caudal: 0,05 a 6,82 m3/h; para caudales inferiores a 0,68 m3/h; o para cualquier aplicación en riego localizado, use el filtro RBY-100-200MX instalado en la parte anterior.
Temperatura: temperatura máxima del agua de 43°C; temperatura ambiente máxima de 52°C.

ESPECIFICACIONES ELÉCTRICAS
Electroválvula de 24 V de CA a 50/60 HZ.
Corriente de entrada máxima: 0,250 Amperios a 60 Hz.
Corriente de retención: 0,143 Amperios a 60 Hz.
Resistencia de la bobina: 52 a 55 Ohmios.

Figura 2.7 Electroválvula Rain Bird Modelo 100- HV (Hembra x Hembra)

Módulo de relevadores

Este módulo de relevadores (relés) para conmutación de cargas de potencia. Los contactos de los relevadores están diseñados para conmutar cargas de hasta 10A y 250VAC (30VDC), aunque se recomienda usar niveles de tensión por debajo de estos límites. Las entradas de control se encuentran aisladas con optoacopladores para minimizar el ruido percibido por el circuito de control mientras se realiza la conmutación de la carga. La señal de control puede provenir de cualquier circuito de control TTL o CMOS como puede ser un microcontrolador. Este módulo es ideal para conmutar cargas de corriente alterna conectadas a la red eléctrica. Soporta todos los microcontroladores, aplicaciones en zonas industriales, control del PLC, entre otros. Este módulo es capaz de controlar varios equipamientos de alta corriente durante un tiempo prolongado. Puede ser controlado por muchos microcontroladores como Arduino, 8051, AVR, PIC, DSP, ARM, MSP430, TTL.

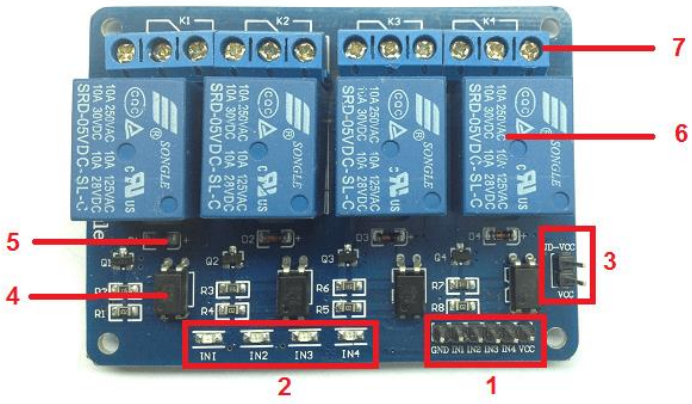


Figura 2.8 Modulo de relés

1. Conector de entradas (IN1 a IN4) y alimentación (GND es masa o negativo y Vcc es el positivo).
2. Cuatro leds que indican el estado de las entradas.
3. Un jumper selector para la alimentación de los relés.
4. Cuatro optoacopladores del tipo FL817C.
5. Cuatro diodos de protección.
6. Cuatro relés con bobinas de 5V y contactos capaces de controlar hasta 10 Amperes en una tensión de 250V.
7. Cuatro borneras, con tres contactos cada una (Común, Normal abierto y Normal cerrado), para las salidas de los relés.

Transformador

Los transformadores son dispositivos electromagnéticos estáticos que permiten partiendo de una tensión alterna conectada a su entrada, obtener otra tensión alterna mayor o menor que la anterior en la salida del transformador.

Permiten así proporcionar una tensión adecuada a las características de los receptores. También son fundamentales para el transporte de energía eléctrica a largas distancias a tensiones altas, con mínimas perdidas y conductores de secciones moderadas.

Principio de funcionamiento

Uno de los devanados, denominado primario (ω1), se conecta a la fuente de corriente alterna cuyo voltaje se necesita variar. La corriente del devanado primario crea en el núcleo un flujo magnético alterno Φ, que se expresa en Weber (Wb). El núcleo del transformador se fabrica formando un circuito cerrado de manera que el flujo en todo su recorrido cruce por dentro del mismo y no se disperse. El flujo magnético variable Φ induce en el devanado secundario ω2 una fuerza electromotriz (FEM) variable, cuyo valor depende del número de vueltas de este devanado y de la velocidad de variación del flujo magnético, según establecen las leyes de la inducción electromagnética.

Aplicaciones de los Transformadores:

Entre las mucha aplicaciones de los transformadores se encuentra utilizarlos como: Soldadores eléctricos, relevadores o relés; calentadores; formando parte de eliminadores de baterías y su aplicación original, elevadores de tensión para transmitir energía eléctrica a grandes distancias a costo bajo.

Como soldadores se pueden utilizar transformadores de subida o de reducción, en los dos casos las corrientes intensas producidas al cerrar el secundario del transformador, directa o indirectamente, llegan a fundir un metal con otro.

Por lo que respecta al uso como calentadores de agua, un transformador reductor es capaz de aumentar la temperatura de un fluido, si éste pasa por el secundario del transformador o se deposita de alguna forma, ya que la corriente en este devanado es muy grande.

Con el advenimiento de la electrónica, el uso de los transformadores se ha incrementado debido a que los circuitos electrónicos usan bajas tensiones para su alimentación y consumen grandes cantidades de corriente para sus funciones, siendo esto propio para el uso de transformadores de bajada.

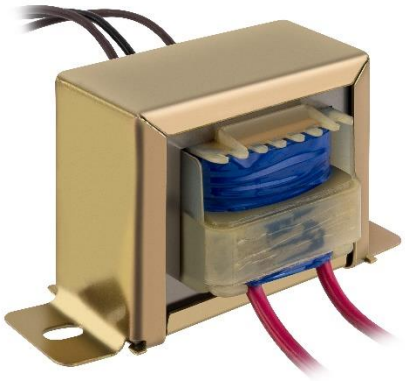


Figura 2.9 Transformador de 24 V

Software necesario

NI-DAQmx

El software de NI-DAQmx va más allá de un controlador básico de adquisición de datos y le brinda mayor productividad y rendimiento en el desarrollo de aplicaciones DAQ y control. Controla cada aspecto de su sistema DAQ (incluyendo dispositivos de acondicionamiento de señales de NI), desde la configuración a la programación en LabVIEW, hasta el control a nivel del sistema operativo y del dispositivo. Obtenga rápidamente datos del mundo real con canales virtuales listos para medir y el DAQ Assistant. Construya sus aplicaciones con VIs específicos para medida, funciones, tipos de datos e integraciones de análisis. Obtenga sus medidas más rápido y de manear más fiable con transferencia de datos por DMA optimizada y E/S de un solo

punto. NI-DAQmx funciona con LabVIEW, SignalExpress, LabWindows/CVI, C/C++, Visual Basic, Visual Basic .NET y C#. Junto con LabVIEW, NI-DAQmx es una de las principales razones por las que National Instruments es líder en la instrumentación virtual y DAQ basada en PC.

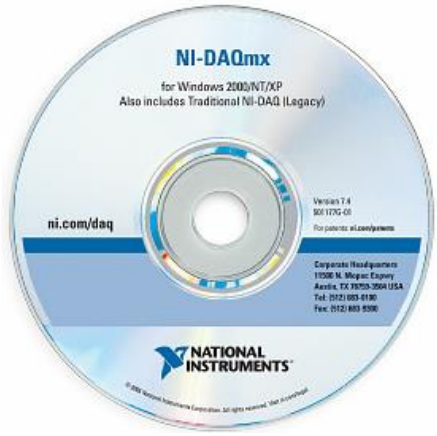


Figura 2.10 Disco de instalación del NI-DAQmx

MATLAB

MATLAB (Matrix Laboratory, “laboratorios de matrices”). Software matemático con entorno de desarrollo integrado (IDE) que tiene un lenguaje de programación propio (Lenguaje M) y es multiplataforma (Unix, Windows y Apple Mac Os X). Software de un gran uso en Centros de Investigación y Desarrollo así como en universidades.

Funciones

- Dentro de sus principales funciones se encuentran:
- Manipulación de Matrices.
- La representación de datos y funciones.
- Implementación de algoritmos.
- Creación de interfaces de usuario (GUI).
- Comunicación con programas en otros lenguajes y con otros dispositivos Hardware.

Herramientas Adicionales

- Simulink (plataforma de simulación multidominio).
- GUIDE (editor de interfaces de usuario - GUI).

Y también se pueden ampliar sus capacidades con las cajas de herramientas de MATLAB, y con los paquetes de bloques de Simulink.

Figura 2.11 Logo del software de MATLAB

CAPITULO 3: DESARROLLO

Cronograma de actividades propuesto

Actividad	Semana															
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
Estudio exhaustivo del estado del arte sobre detección de fugas en redes hidráulicas																
Construcción de la red hidráulica																
Instrumentación con sensores de presión y caudal																
Diseño del sistema SCADA																
Simulación de fugas mediante válvulas electrónicas																
Realizarán pruebas a campo abierto																

Funcionamiento del sistema de fugas de serpentín.

En el laboratorio de hidráulica se tiene un serpentín con lo que cuenta con sensores (presión caudal) y válvulas con activación manual, en la siguiente imagen contiene una válvula manual a la que se pretende cambiar que muestra cuando esta activada y desactivada.

Figura 3.1 Válvula manual desactivada.

Figura 3.2 Válvula manual activada.

El serpentín cuenta una simulación de 4 fugas que se encuentran puestas aleatoriamente definas como la variable Z1, Z2, Z3, Z4.

El sistema funciona con una bomba de 5HP que está a 220 volts con un variador de frecuencia, se configura ese variador y se enciende la bomba. Al encender la bomba se empieza a extraer el agua de un tinaco de 2500 litros, el fluido pasa por la tubería y se vuelve a deposita en el mismo tinaco, este sistema contiene 4 sensores, dos sensores (caudal y presión) al inicio y los otros dos (caudal y presión) al final.

Figura 3.3 Variador de frecuencia configurada

Cada vez que se abra una válvula se simula una fuga el agua cae en un contenedor como se puede apreciar en la imagen anterior. Cada fuga simulada contiene unos contenedores conectados entre sí por medio de tubos. Todo el fluido que cae de la fuga se va directo a un tinaco de menor capacidad. El tinaco de menor capacidad contiene una bomba de 1/2HP su función es mandar el agua al tinaco de 2500 litros. En la siguiente imagen se puede apreciar el diagrama del sistema.

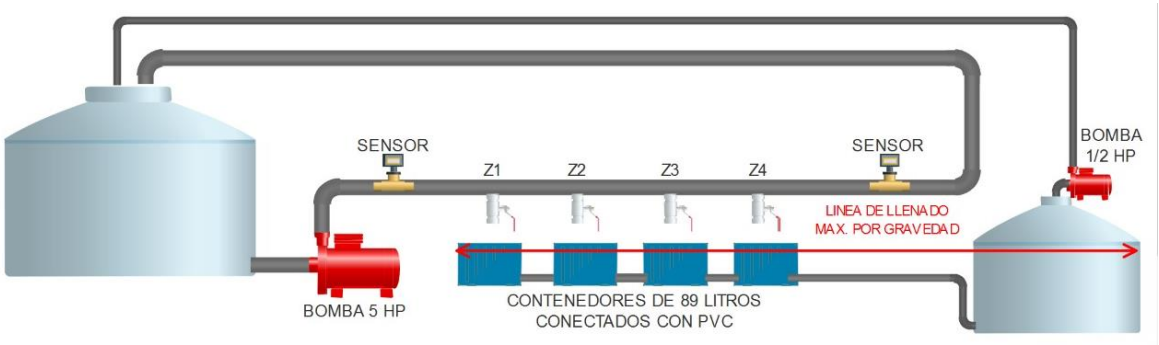


Figura 3.4 Diagrama del sistema de fugas de serpentín

Los sensores (caudal y presión) son controlados por una DAQ NI 9203 programado especialmente para obtener datos en un cierto tiempo deseado, por lo general las válvulas se abren cada 3 minutos (180 segundos) y se toman tiempos de 15, 12, 9

minutos. En el primer programa se muestra la obtención de datos de este sistema mientras que el segundo muestra la conversión de corriente a litros por minuto para caudales y la conversión de corriente a kilo pascales para presiones.

captura.m | mapeo.m

```
function [t,u,y] = captura(duracion,TEMP,archivo)
if nargin < 1
    duracion = 900;
    TEMP = 27;
    archivo = 'prueba';
end
s = daq.createSession('ni');
addAnalogInputChannel(s,'cDAQ1Mod1',0:3,'Current');
s.Rate = 1000;
s.DurationInSeconds = duracion;
disp('Pulsa una tecla para iniciar...')
pause
disp('Capturando...')
[datos,t] = s.startForeground;
disp('Finalizado')
u = datos(:,1:2); % presiones
y = datos(:,3:4); % caudales
[t,u,y] = mapeo(t,u,y,TEMP);
subplot(211); plot(t,u)
subplot(212); plot(t,y)
save(archivo,'t','u','y','TEMP')
end
```

Figura 3.5 Programa para la obtención de datos y gráficas del sistema de fugas

captura.m | mapeo.m

```
function [t,u,y] = mapeo(t,u,y,T)
if nargin < 4
    T = 27;
end
datos = [0,0.99987;
4.0,1.00000;
4.4,0.99999;
10,0.99975;
15.6,0.99907;
21,0.99802;
26.7,0.99669;
32.2,0.99510;
37.8,0.99318;
48.9,0.98870;
60,0.98338;
71.1,0.97729;
82.2,0.97056;
93.3,0.96333;
100,0.95865];
rho = 1000*interp1(datos(:,1),datos(:,2),T);
g = 9.7827577922898;
c1 = polyfit([4e-3,20e-3],[0,1e5]/rho/g,1);
u(:,1) = polyval(c1,u(:,1));
c2 = polyfit([4e-3,20e-3],[0,1e5]/rho/g,1);
u(:,2) = polyval(c2,u(:,2));
c3 = polyfit([4e-3,20e-3],[0,500],1);
y(:,1) = polyval(c3,y(:,1));
c4 = polyfit([4e-3,20e-3],[0,500],1);
y(:,2) = polyval(c4,y(:,2));
y = y/60000;
```

Figura 3.6 Programa para la conversión de corriente a litros por minuto (Caudal) y a kilo pascales (presión)

Al término del tiempo en el que se ha indicado al programa se obtienen los siguientes resultados plasmados en una gráfica, y esos datos se guardan en un archivo .m, a continuación se muestra una gráfica de una prueba que se obtuvo.

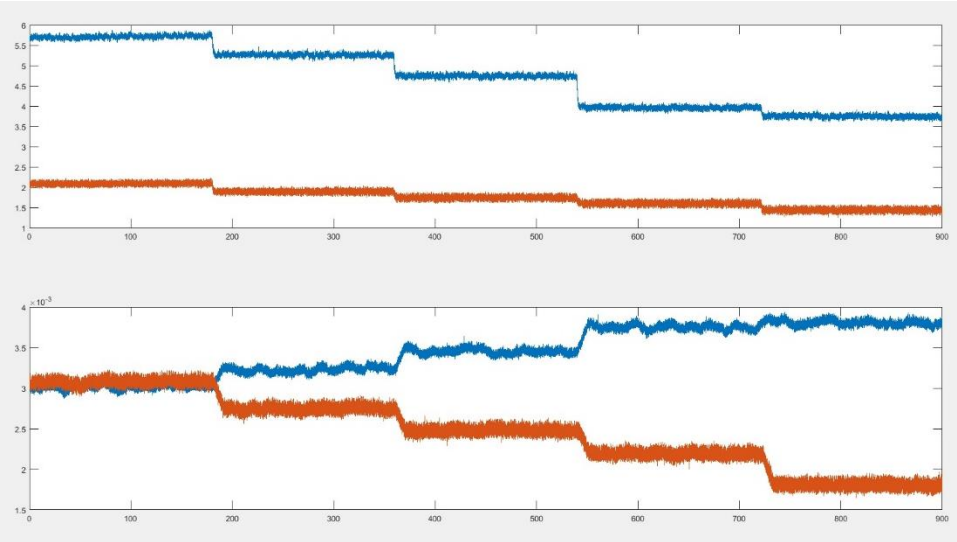


Figura 3.7 Grafica de captura de datos utilizando las 4 válvulas

La grafica mostrada con anterioridad es de una simulación de 4 fugas por lo que se llevó un tiempo de 15 minutos. Los datos que se obtienen son para hacer cálculos para poder así detectar a que distancia se encuentra la fuga y los litros por minuto que se pierden.

Materiales que utilizaron para las primeras pruebas

En las primeras pruebas que se hicieron fueron probar las electroválvulas y los transformadores que estuvieran en buen estado, estas electroválvulas serán sustituidas por las válvulas mecánicas que se encuentran en el serpentín. Los materiales que usaron para probar las electroválvulas son:

- Cable calibre 10
- Transformador de 24V
- Un multímetro
- Una clavija
- Bushing
- Niple

Los Niples y los Bushing su función iba ser que la electroválvula se pueda acoplar muy bien al orificio donde se encontraba las válvulas mecánicas.

Lo que se implemento fue conectar la clavija con el cable y del cable al transformador para que después se valla a la electroválvula y así a la hora de conectar el cable se iba a accionar la electroválvula. Se necesita utilizar el transformador para las electroválvulas ya que la electroválvula no trabaja con corriente directa si no trabaja con corriente alterna lo que se obliga a utilizar el transformador.

Como resultado se obtuvieron un buen funcionamiento de las 5 electroválvulas y 5 transformadores.

Figura 3.8 Conexionado para probar el funcionamiento de la electroválvula

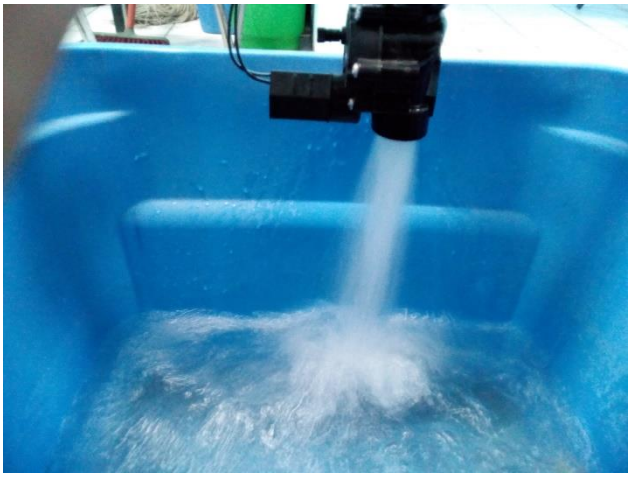


Figura 3.9 Electroválvula funcionando correctamente

Desarrollo de la instrumentación y programación del sistema

Se empezó a desarrollar haciendo programas para poder controlar las electroválvulas, una de ellas fue utilizando el software de LabVIEW, este software su tipo de programación está basado por bloques cosa que era un poco más sencillo de utilizar. La tarjeta de adquisición de datos que se uso fue la NI USB 6002 ya que esta contenía salidas digitales que lo diferencia a la otra DAQ NI 9203 que contiene puras entradas analógicas. La siguiente imagen muestra el sistema de como quedo el control de las electroválvulas.

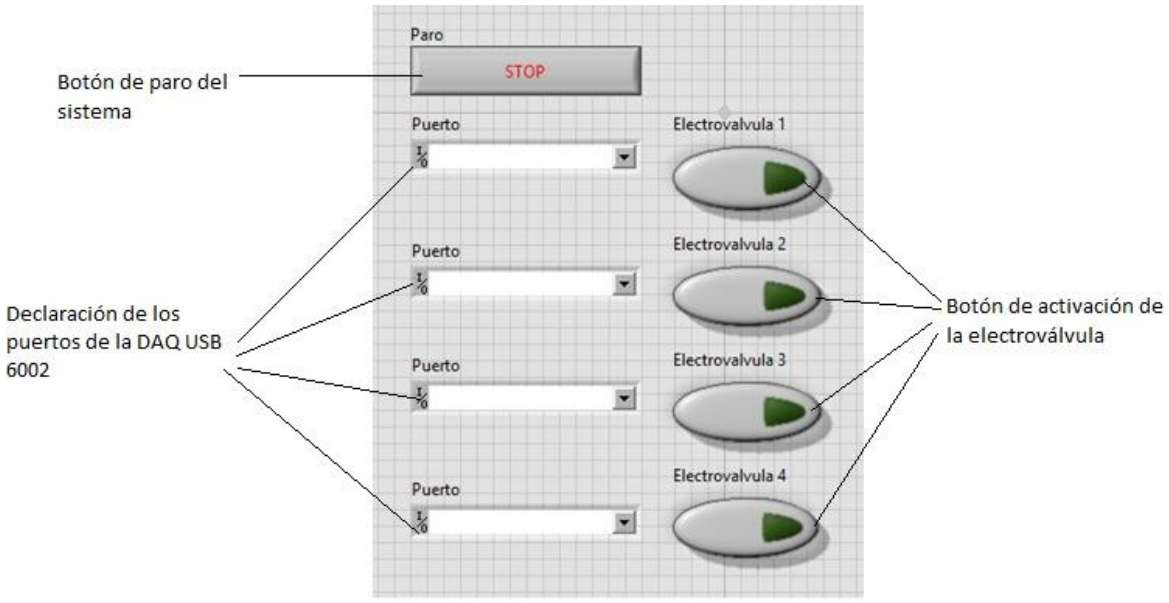


Figura 3.10 Sistema de control de electroválvulas

A continuación se mostrara el programa en LabVIEW del sistema de control de electroválvulas.

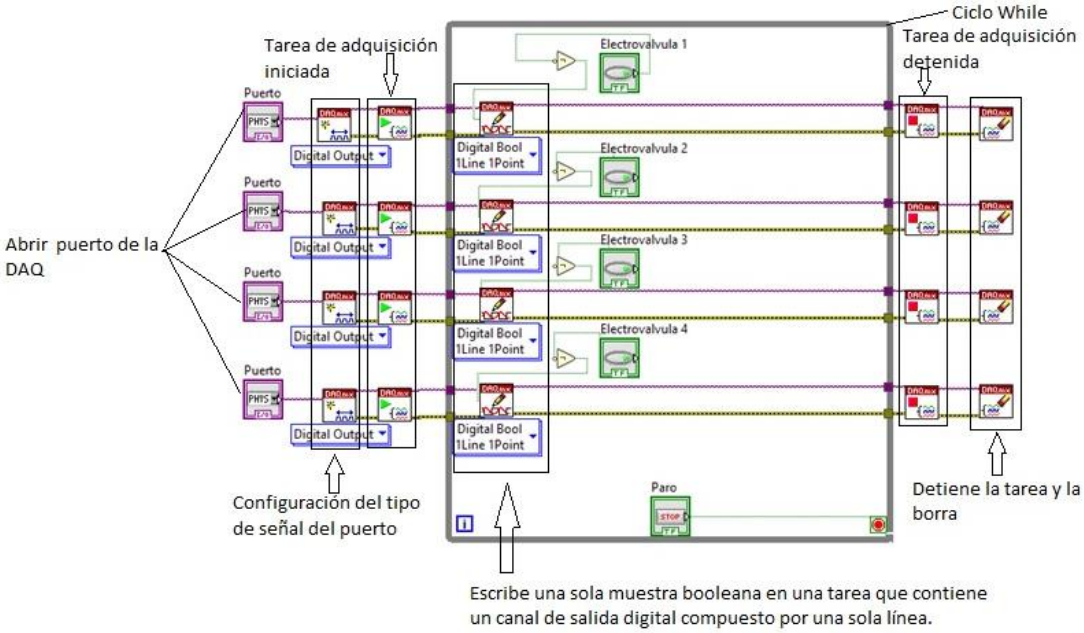


Figura 3.11 Programa de control de electroválvulas

El funcionamiento del control de las electroválvulas fue bastante bueno ya que el programa funciono correctamente. Prácticamente se probó con el módulo de relevadores ya que el módulo contenía leds indicando que la señal que se había enviado era la correcta. Se fue probando cada una de los leds que tenía el modulo como se muestra en la siguiente figura.

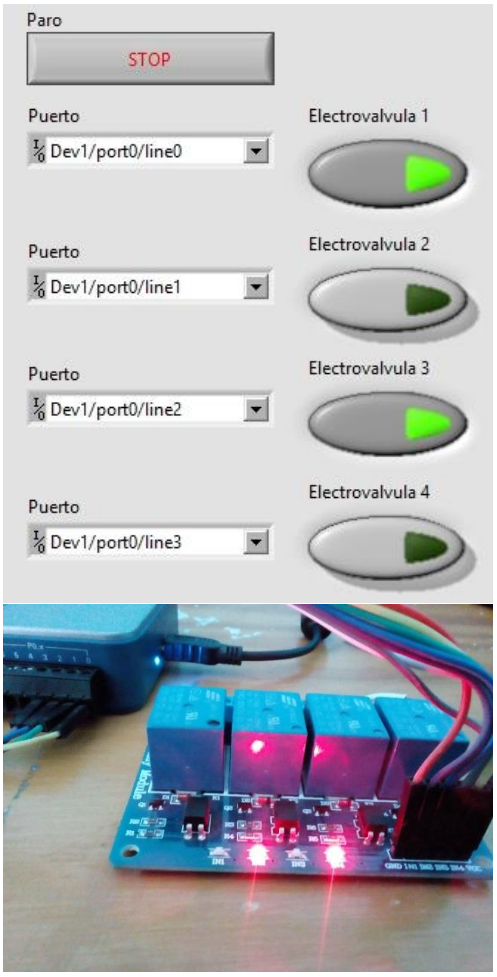


Figura 3.12 Activación de dos electroválvulas al mismo tiempo

Figura 3.13 Activación de las cuatro electroválvulas al mismo tiempo

Una vez que el programa funciono correctamente se empezaron a cambiar las válvulas mecánicas por las electroválvulas.

Figura 3.14 Sustitución de las válvulas mecánicas por electroválvulas

Después de ensamblar las electroválvulas se empezó hacer toda la conexión, en el siguiente esquema se muestra el primer diagrama de conexionado del sistema.

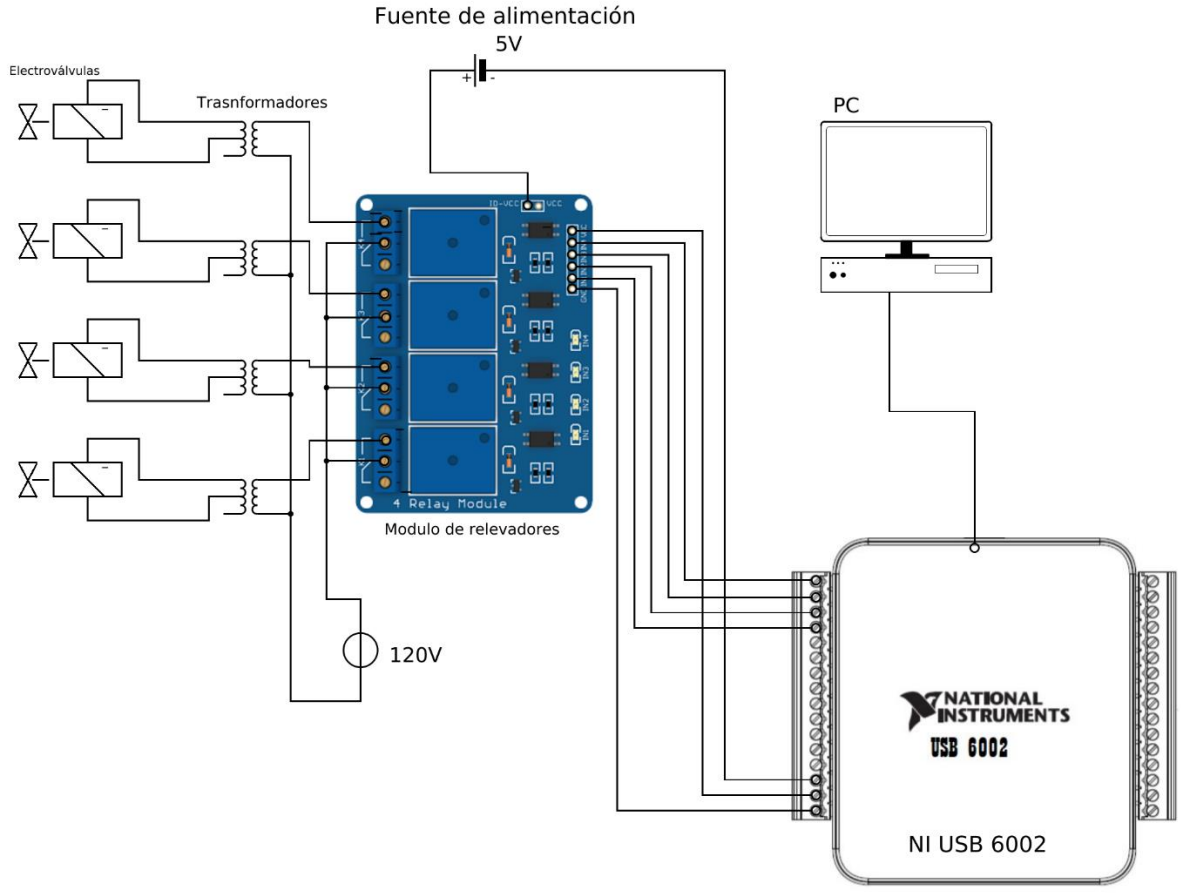


Figura 3.15 Diagrama 1 de conexión de las electroválvulas al módulo de relevadores controlado por la DAQ NI USB 6002

Figura 3.16 Proceso del cableado

Una vez terminado el proceso de cambio de válvulas mecánicas por las electroválvulas y el proceso de cableado tal como lo indicaba el diagrama 1 de conexión se probaron las electroválvulas y como resultado se obtuvo un buen funcionamiento, hubo algunos detalles al activar al activar las 4 electroválvulas al mismo tiempo, ya que en la cuarta electroválvula prácticamente el flujo que expulsaba era menor a las demás electroválvulas, esto se debió a que la cuarta electroválvula que simulaba la fuga se encontraba casi al final del serpentín, y la presión que se manejaba donde se simulaba la cuarta fuga era muy baja a diferencia de la primera fuga, por lo que se llegó a entender que se necesita un poco más de presión de agua para que el flujo de la fuga se un poco mayor.

Figura 3.17 Resultado del buen funcionamiento de las electroválvulas

El sistema de conexionado se le agregó las siguientes conexiones de la DAQ NI 9203, en el siguiente esquema se muestra el conexionado final de todo el sistema, con la utilización de las dos tarjetas de adquisición de datos.

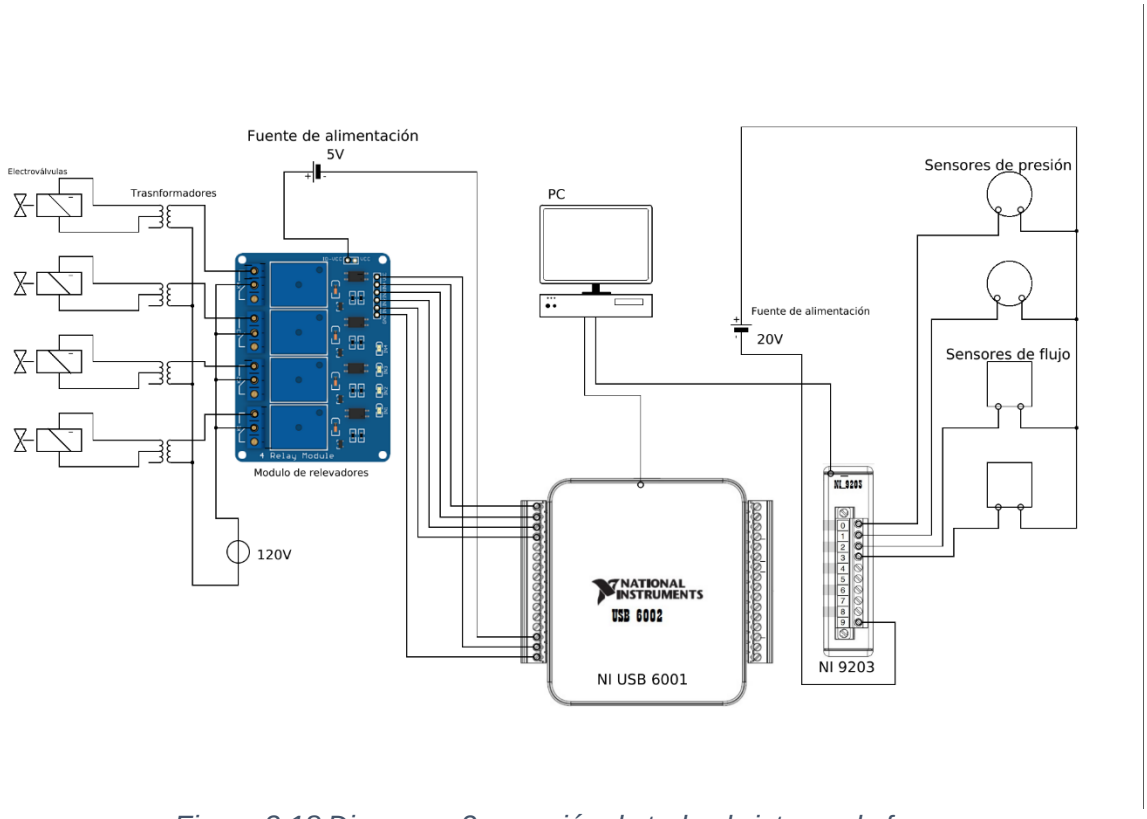


Figura 3.18 Diagrama 2 conexión de todo el sistema de fugas.

Una vez probado el sistema y obteniendo resultados buenos, se hizo una prueba de captura de datos del programa de MATLAB, para poder obtener una nueva gráfica en función con las electroválvulas ingresadas y poder compararla con la gráfica anterior que eran válvulas mecánicas. La siguiente grafica muestra los resultados nuevos con las electroválvulas en función.

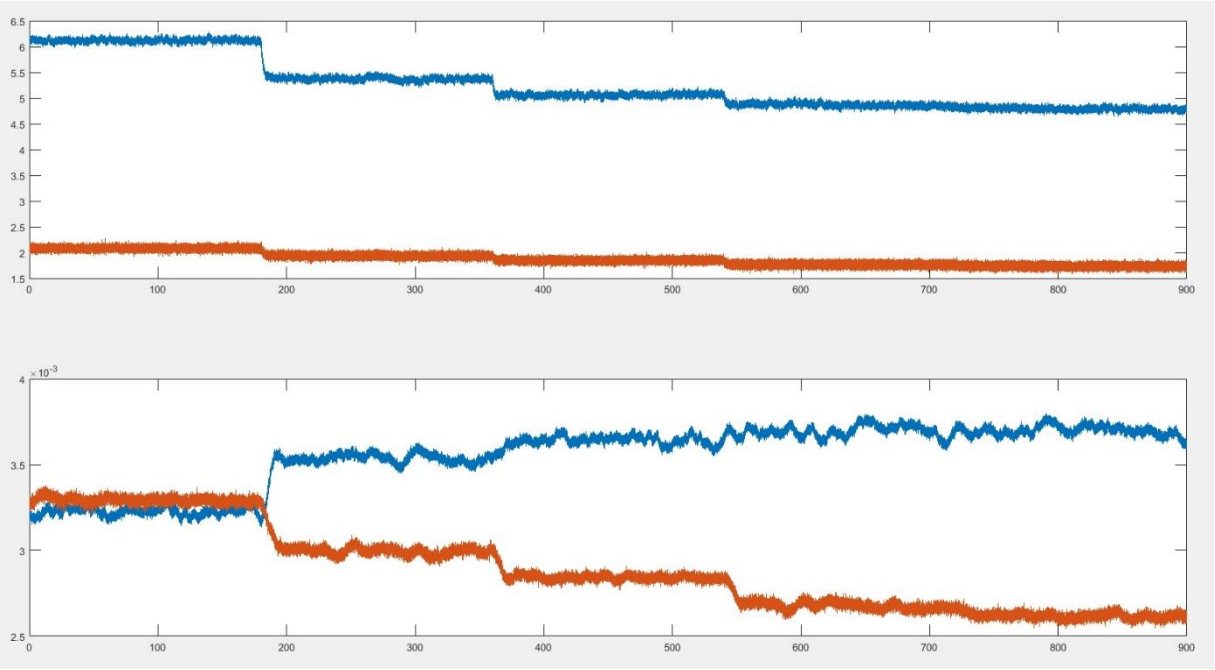


Figura 3.19 Grafica de captura de datos utilizando las 4 electroválvulas

Podemos observar en esta nueva grafica que es muy diferente a la anterior, en la nueva gráfica se puede observar que al activar la cuarta fuga, podemos apreciar en la gráfica que la caída de presión no se notó mucho, esto fue debido a lo que se explicó anteriormente del problema que fue al poco flujo y presión de las electroválvulas.

Después de saber que se obtuvieron buenos resultados con el control de las electroválvulas, se dio a la tarea de instrumentar el control de las electroválvulas, graficar la cantidad de flujo y presión que está en la tubería, así como también introducirle cuatro manómetros para cada uno de los sensores, que podrán indicar ya sea el flujo o la presión de los sensores, todo esto será en tiempo real, utilizando el soporte de App Designer de MATLAB.

Se empezó a instrumentar la parte de control de electroválvulas, este sistema era algo parecido a lo que se hiso en LabVIEW. Se le inserto cuatro switches con sus respectivos indicadores, en el siguiente esquema se puede observar lo siguiente.

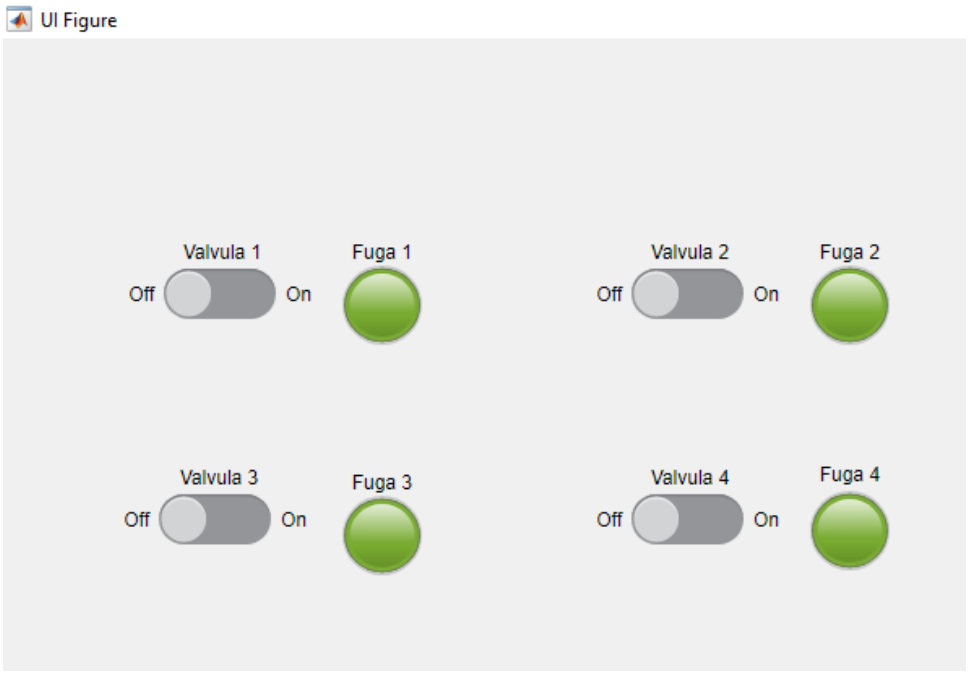


Figura 3.20 Sistema de control de electroválvulas por MatLab con el soporte de App Designer

Una vez terminado de insertar los switches y los indicadores se empezó a programar cada uno de los switch e indicadores, señalando que si el switch pasa de off a on el indicador de fuga cambiaria de color y la electroválvula se iba a encender, caso contrario de que el switch pasa de on a off el indicador cambiara a su estado actual y la electroválvula se tendría que apagar.

Comenzamos la programación definiendo las propiedades que serán accesibles para todo tipo de funciones y devoluciones de llamada.

```
47      properties (Access = private)
48 -        s; % Crea la sesion de la primera Daq
49 -        estado = [1 1 1 1]; % La matriz indica la ativacion de las valvulas
```

Después de definir las propiedades damos a conocer las condiciones que de cómo se va a empezar a ejecutar el programa, diciendo que el programa se empiece a ejecutar con los switches estando en “off”, como también declaramos que los indicadores se encuentren todos del mismo color, también crear la sesión de la NI USB 6002 para poder abrir número de puertos digitales de salida que se utilizaran para cada switch.

```
91          % Code that executes after component creation
92          function startupFcn(app)
93              app.Valvula1Switch.Value = 'Off';
94              app.Valvula2Switch.Value = 'Off';
95              app.Valvula3Switch.Value = 'Off';
96              app.Valvula4Switch.Value = 'Off';
97              app.Fuga1Lamp.Color = [.47 .67 .19];
98              app.Fuga2Lamp.Color = [.47 .67 .19];
99              app.Fuga3Lamp.Color = [.47 .67 .19];
100             app.Fuga4Lamp.Color = [.47 .67 .19];
101             delete(instrfind)
102             app.s = daq.createSession('ni');
103             addDigitalChannel(app.s,'Dev1','Port0/Line0:3','OutputOnly');
104             outputSingleScan(app.s,[1,1,1,1]);
105             delete(instrfind)
```

Una vez hecho lo anterior procedemos a programar cada switch con su respectivo indicador declarando que cada vez que haya un cambio posición en el switch el indicador automáticamente cambiara de color, dependiendo de lo que indique el switch si es “off” la electroválvula permanecerá apagado, en caso contrario si el switch indica “on” la electroválvula encenderá.

```
130         % Value changed function: Valvula1Switch
131         function Valvula1SwitchValueChanged(app, event)
132             value = app.Valvula1Switch.Value;
133             app.estado = xor(app.estado,[0,0,0,1]);
134             outputSingleScan(app.s,app.estado);
135             actual = app.estado;
136             if actual(4) == 1
137                 app.Fuga1Lamp.Color = [0.47,0.67,0.19];
138             else
139                 app.Fuga1Lamp.Color = [0,1,0];
140             end
141         end
142
143         % Value changed function: Valvula2Switch
144         function Valvula2SwitchValueChanged(app, event)
145             value = app.Valvula2Switch.Value;
146             app.estado = xor(app.estado,[0,0,1,0]);
147             outputSingleScan(app.s,app.estado);
148             actual = app.estado;
149             if actual(3) == 1
150                 app.Fuga2Lamp.Color = [0.47,0.67,0.19];
151             else
152                 app.Fuga2Lamp.Color = [0,1,0];
153             end
154         end
```

```
156         % Value changed function: Valvula3Switch
157         function Valvula3SwitchValueChanged(app, event)
158             value = app.Valvula3Switch.Value;
159             app.estado = xor(app.estado,[0,1,0,0]);
160             outputSingleScan(app.s,app.estado);
161             actual = app.estado;
162             if actual(2) == 1
163                 app.Fuga3Lamp.Color = [0.47,0.67,0.19];
164             else
165                 app.Fuga3Lamp.Color = [0,1,0];
166             end
167         end
168
169         % Value changed function: Valvula4Switch
170         function Valvula4SwitchValueChanged(app, event)
171             value = app.Valvula4Switch.Value;
172             app.estado = xor(app.estado,[1,0,0,0]);
173             outputSingleScan(app.s,app.estado);
174             actual = app.estado;
175             if actual(1) == 1
176                 app.Fuga4Lamp.Color = [0.47,0.67,0.19];
177             else
178                 app.Fuga4Lamp.Color = [0,1,0];
179             end
180         end
```

Los resultados fueron satisfactorios ya que el sistema de control de las electroválvulas funciono correctamente tal y como se esperó.

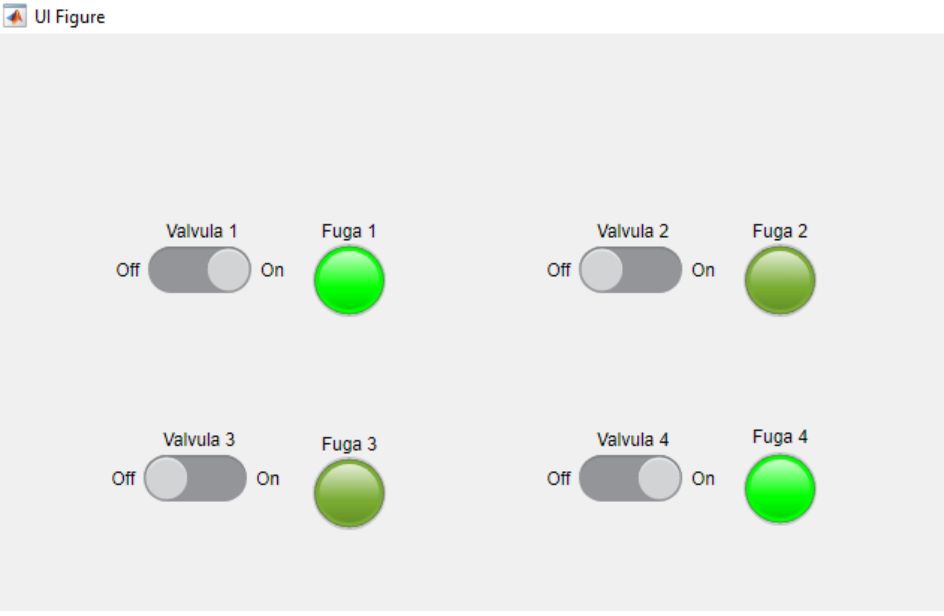


Figura 3.21 Sistema de control de electroválvulas funcionando con dos electroválvulas activados

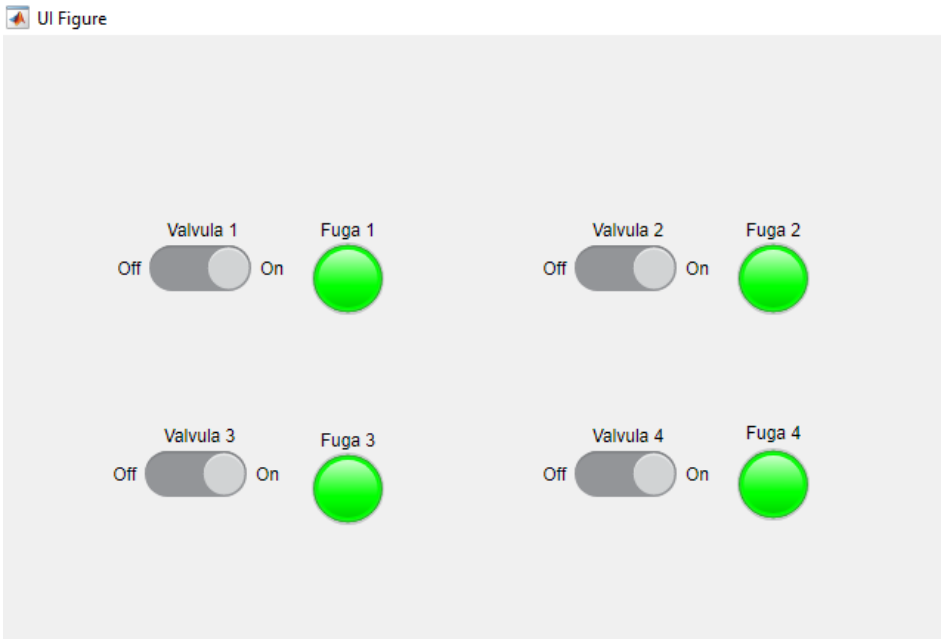


Figura 3.22 Sistema de control de electroválvulas funcionando con las cuatro electroválvulas activados

Teniendo un buen resultado del sistema de control de las electroválvulas pasamos a instrumentar los sensores de presión y de caudal adicionándole al sistema cuatro manómetros que son para cada uno de los sensores indicando así el valor que indica cada sensor.

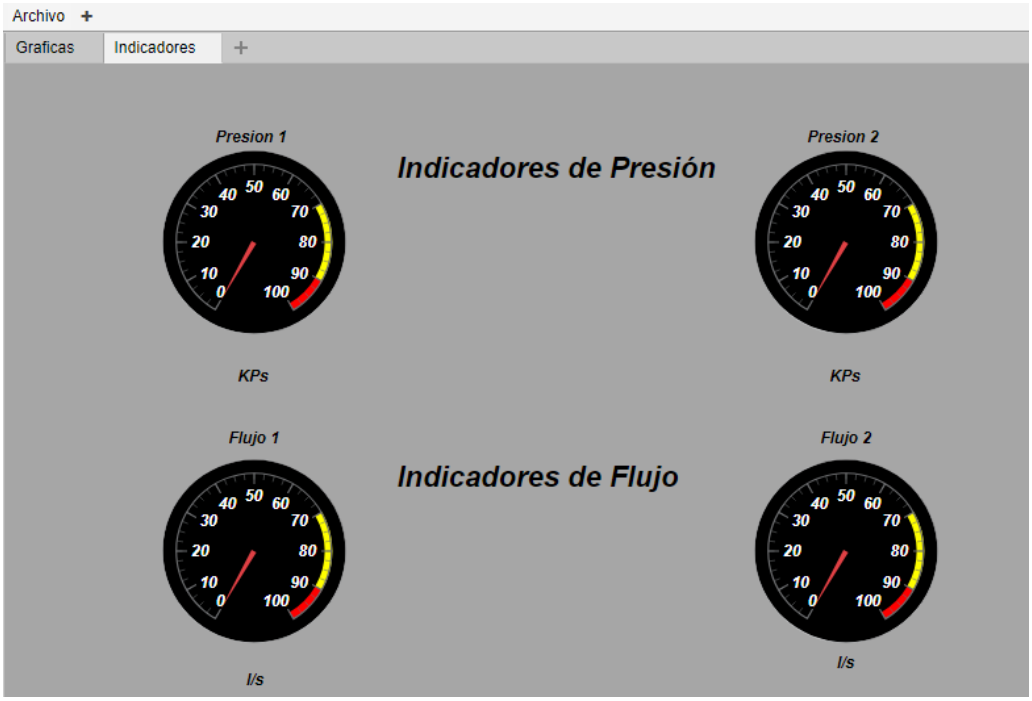


Figura 3.23 Manómetros de sensores de presión y de caudal.

Una vez introducidos los manómetros pasamos a la programación, donde indicara el valor de presión y de flujo.

```
91          % Code that executes after component creation
92          function startupFcn(app)
```

```
107 -           app.se = daq.createSession('ni');
108 -           app.ch = addAnalogInputChannel(app.se,'cDAQ1Mod1', 0:3, 'Current');
109 -           app.se.Rate = 10;
110 -           app.se.DurationInSeconds = 600;
111 -           app.lh = addlistener(app.se,'DataAvailable', @app.Grafique);
112 -           app.se.NotifyWhenDataAvailableExceeds = 10;
113 -           app.se.startBackground();
```

En la misma función que dice “function startupFcn(app)” agregamos la segunda sesión de la NI 9203 abriendo los cuatro puertos analógicos de entrada que serán para los cuatro sensores, agregamos una tasa de operaciones de escaneos por segundo, usamos un addlistener para agregar una función anónima a la sesión. Esta función se invoca cada vez que se produce el evento DataAvailable y traza los datos adquiridos contra el tiempo.

```
58          function Grafique(app,objeto,evento)
59 -            dato = evento.Data;
60 -            u = dato(:,1:2);
61 -            y = dato(:,3:4);
62 -            t = evento.TimeStamps;
63 -            TEMP = 27;
64 -            [t,u,y] = mapeo(t,u,y,TEMP);
65 -            P1 = 10*mean(u(:,1));
66 -            P2 = 10*mean(u(:,2));
67 -            F1 = 10*mean(y(:,1));
68 -            F2 = 10*mean(y(:,2));
69
70 -            app.Presion1Gauge.Value = P1;
71 -            app.KPsLabel.Text = sprintf('%4.1f KPa',P1);
72 -            app.Presion2Gauge.Value = P2;
73 -            app.KPsLabel2.Text = sprintf('%4.1f KPa',P2);
```

Agregamos una “function” donde mandamos a llamar la función app.Grafique, donde los datos “u” son de presión y los datos “y” son de flujo, pasamos agregándole el mapeo que esto nos ayudara a convertir los valores de corriente a valores de KPa y L/s, y todos esos datos lo mandamos a los manómetros, se puso unas etiquetas para cada manómetro para que indique el valor exacto.

Una vez terminado el programa de los manómetros de los sensores pasamos a probar el sistema.

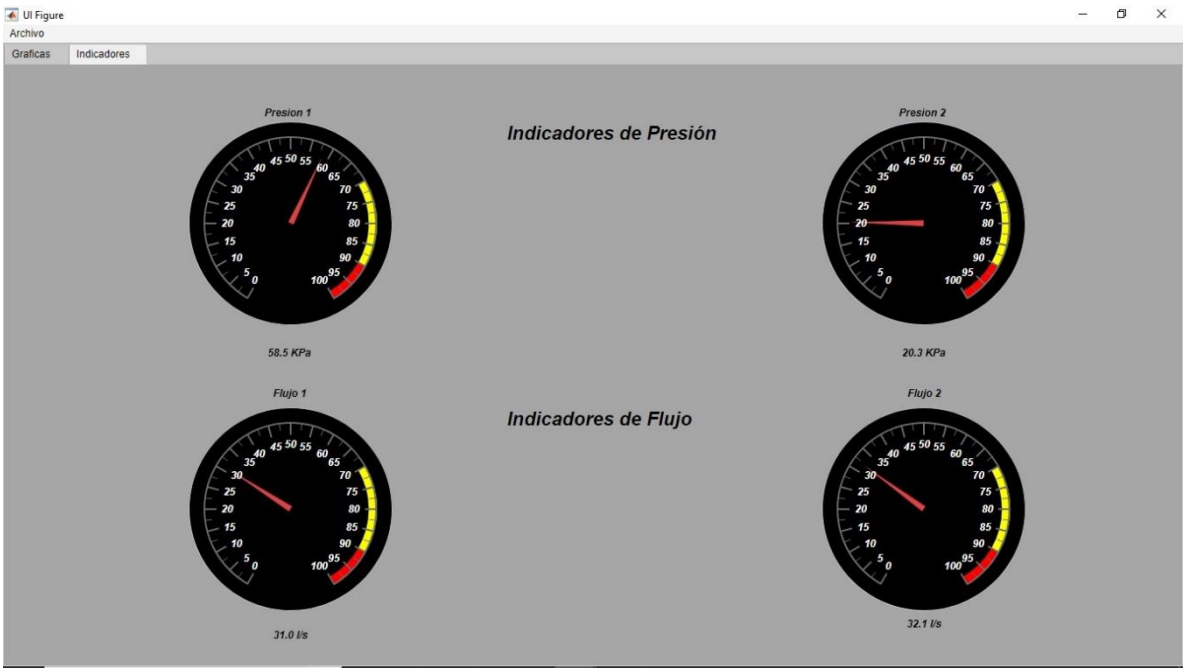


Figura 3.24 Resultados del buen funcionamiento de los manómetros de sensores de presión y de caudal.

A pesar de hacer muchas pruebas y correcciones al programa para que se tuviera un buen funcionamiento del sistema se obtuvo un buen funcionamiento de los manómetros. Los datos que se estaban obtenido se reflejaban en los manómetros y en las etiquetas.

Por ultimo pasamos agregándole las dos graficas donde una gráfica marcara las presiones y la otra grafica marcara los flujos.

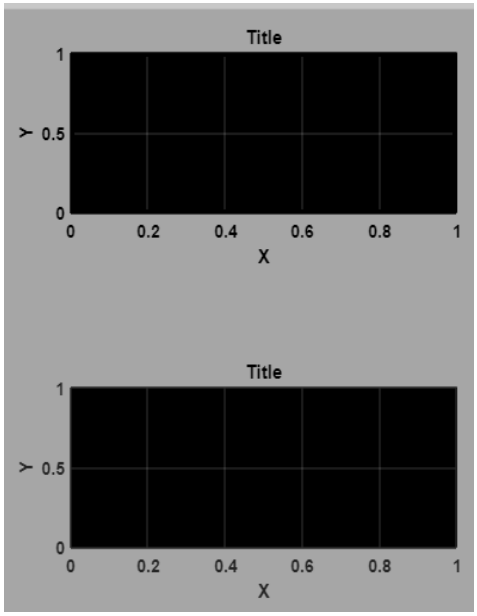


Figura 3.25 Graficas de presión y de flujo

Una vez insertadas las dos graficas pasamos a programar el sistema.

```
91          % Code that executes after component creation
92          function startupFcn(app)
```

```
115 -           plot(app.UIAxes,0,[0,0]);
116 -           app.UIAxes.YLim = [0,10];
117 -           title(app.UIAxes,'Grafica de Presión');
118 -           xlabel(app.UIAxes,'Tiempo (s)')
119 -           ylabel(app.UIAxes,'Presión (KPa)')
120 -           grid(app.UIAxes,'on')
121 -           plot(app.UIAxes2,0,[0,0]);
122 -           app.UIAxes2.YLim = [0,6];
123 -           title(app.UIAxes2,'Grafica de Flujo');
124 -           xlabel(app.UIAxes2,'Tiempo (s)')
125 -           ylabel(app.UIAxes2,'Caudal (l/s)')
126 -           grid(app.UIAxes2,'on')
127
128 -       end
```

En la misma función que dice “function startupFcn(app)” mandamos a llamar a las dos graficas donde se agregaran los títulos, agregar loas etiquetas de la función en “y” y en “x”. Una vez terminada las etiquetas que se agregaron a las gráficas, pasamos a plasmar los datos que se obtienen de los sensores a las gráficas en tiempo real.

```
58          function Grafique(app,objeto,evento)
59 -            dato = evento.Data;
60 -            u = dato(:,1:2);
61 -            y = dato(:,3:4);
62 -            t = evento.TimeStamps;
63 -            TEMP = 27;
64 -            [t,u,y] = mapeo(t,u,y,TEMP);
65 -            P1 = 10*mean(u(:,1));
66 -            P2 = 10*mean(u(:,2));
67 -            F1 = 10*mean(y(:,1));
68 -            F2 = 10*mean(y(:,2));
```

```
74 -             plot(app.UIAxes,u);
75 -             legend(app.UIAxes,{'Presión 1','Presión 2'},'Location','eastoutside','TextColor','w')
76 -             app.Flujo1Gauge.Value = F1;
77 -             app.lsLabel.Text = sprintf('%4.1f l/s',F1);
78 -             app.Flujo2Gauge.Value = F2;
79 -             app.lsLabel2.Text = sprintf('%4.1f l/s',F2);
80 -             plot(app.UIAxes2,y);
81 -             legend(app.UIAxes2,{'Flujo 1','Flujo 2'},'Location','eastoutside','TextColor','w')
82
83 -         end
84
85      end
```

En la “function Grafique (app,objeto,evento)” agregamos las dos gráficas, indicando que los datos que se obtenga de los sensores se plasmen en la gráfica, algo relacionado con los manómetros. Terminado el programa, corremos nuevamente el programa.

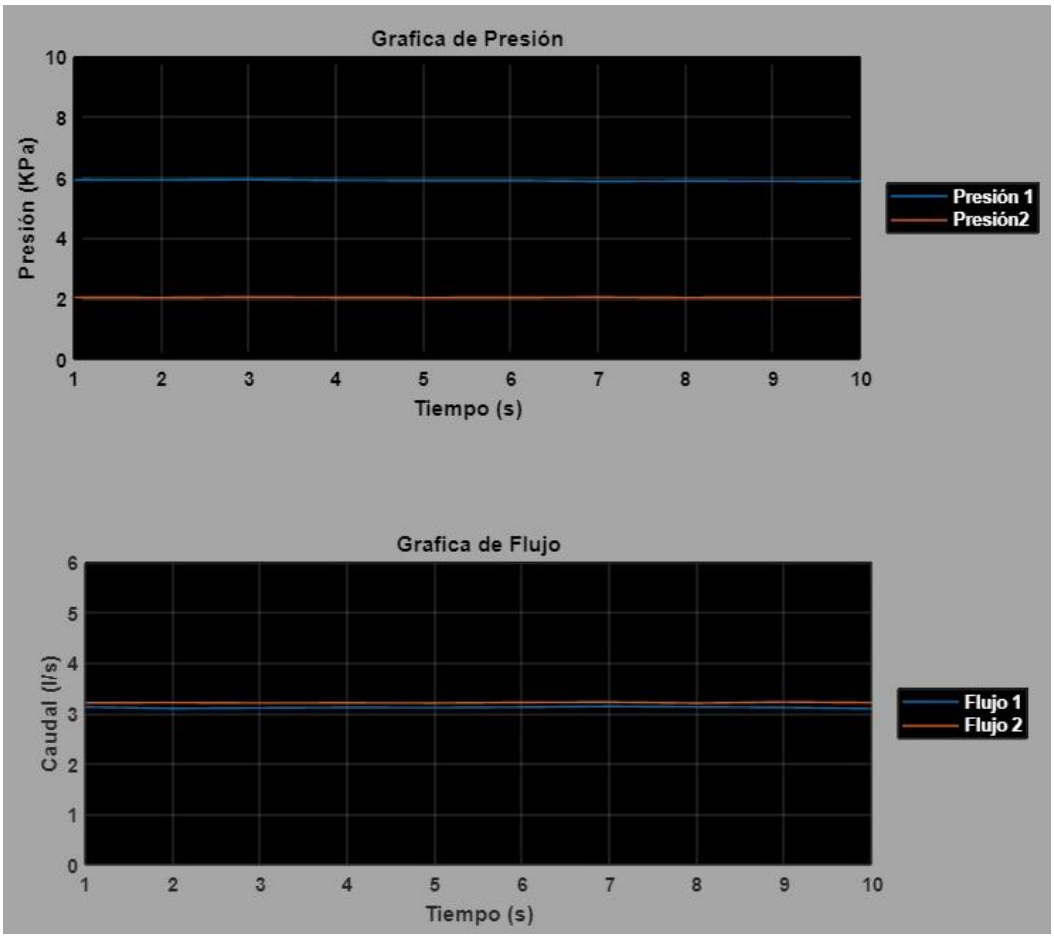


Figura 3.26 Graficas de presión y de flujo, funcionando correctamente.

Se obtuvieron buenos resultados de las dos gráficas, al bajar la velocidad de la bomba de la tubería con el variador de frecuencia las presiones y los caudales las gráficas y los manómetros bajaban, caso contrario cuando se incrementaba la frecuencia aumentaban los manómetros y las gráficas.

Para terminar el programa solo se agregaron algunos detalles a la programación y se acomodaron lo que es bien los switches e indicadores de las electroválvulas, con las gráficas, se agregaron algunos paneles al programa como también se colocaron barra de menús, grupo de tabulación y un panel para los switches. El sistema final quedo de la siguiente manera.

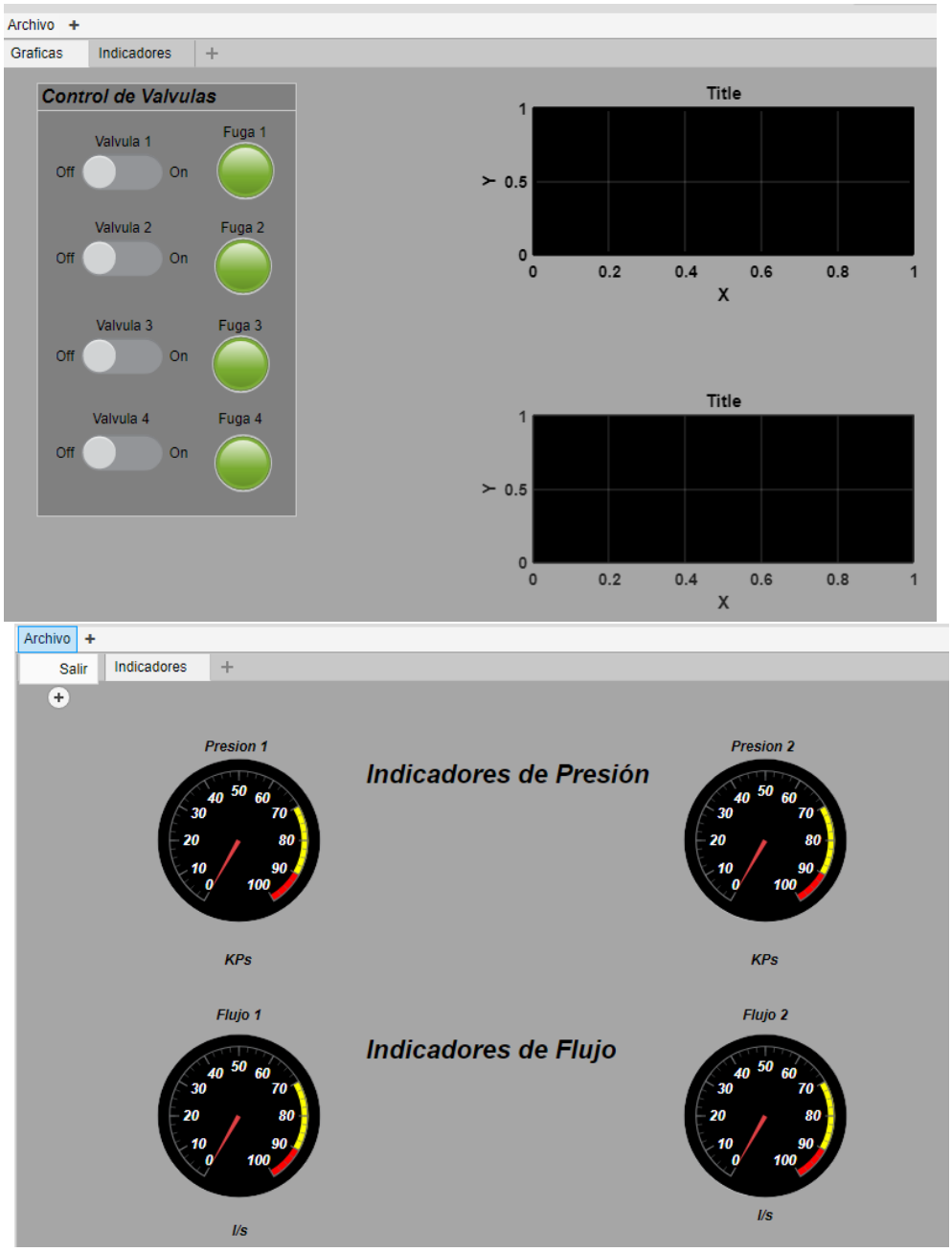


Figura 3.27 Sistema terminando

Una vez terminado se corrió todo el sistema para el funcionamiento completo de las electroválvulas, los manómetros y las gráficas. El resultado de todo este sistema fue bueno, ya que todo el sistema funciono perfectamente bien, al activar cualquier electroválvula los manómetros y las gráficas de presión donde recibían datos de los sensores disminuían a diferencia de los manómetros y gráficas de flujo, un sensor disminuía mientras que el otro sensor aumentaba el flujo, esto pasaba cada vez que se activaba una electroválvula, entre más se activaba el

número de electroválvulas mayor iba a ser el cambio en los manómetros y en las gráficas.

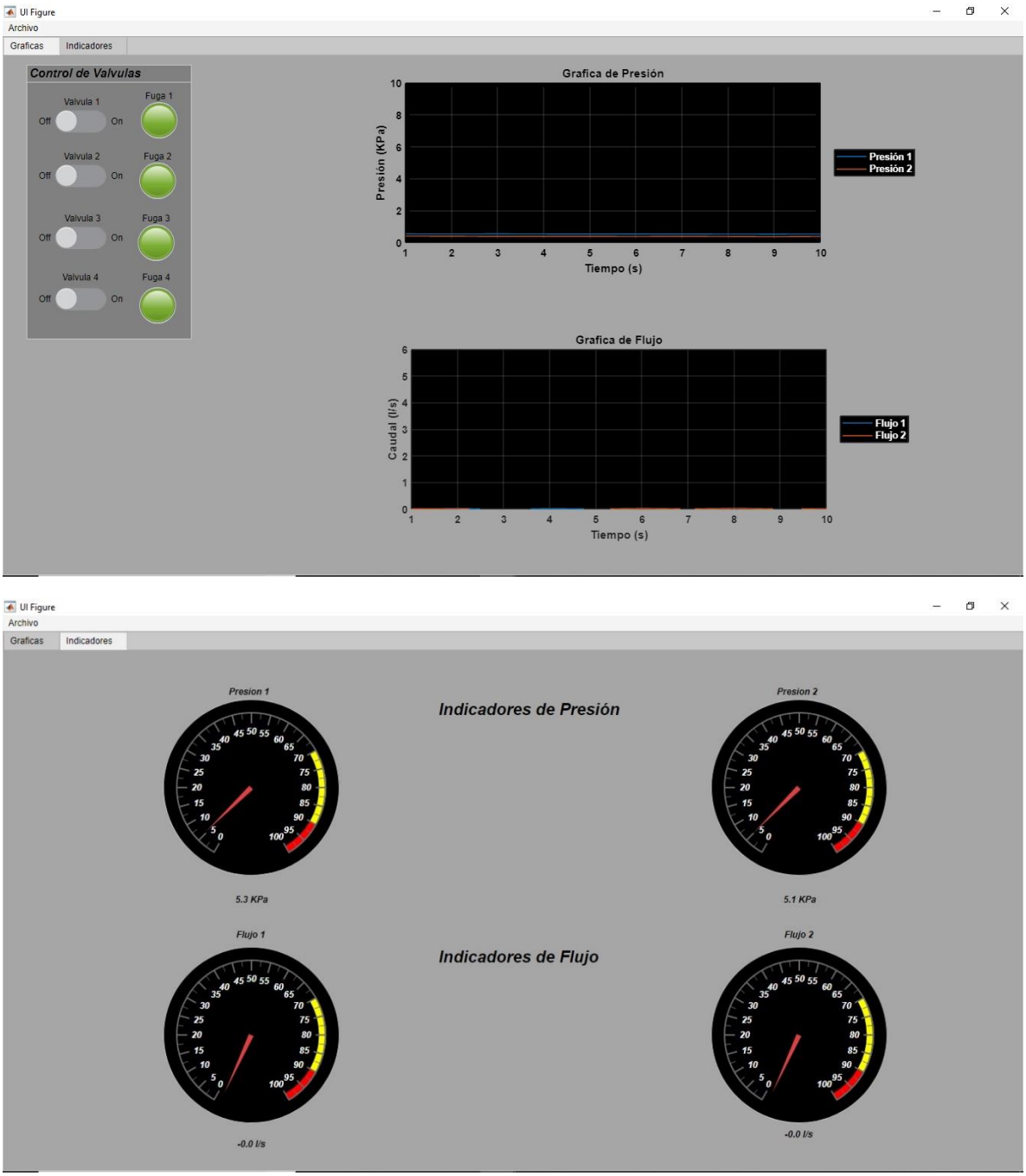
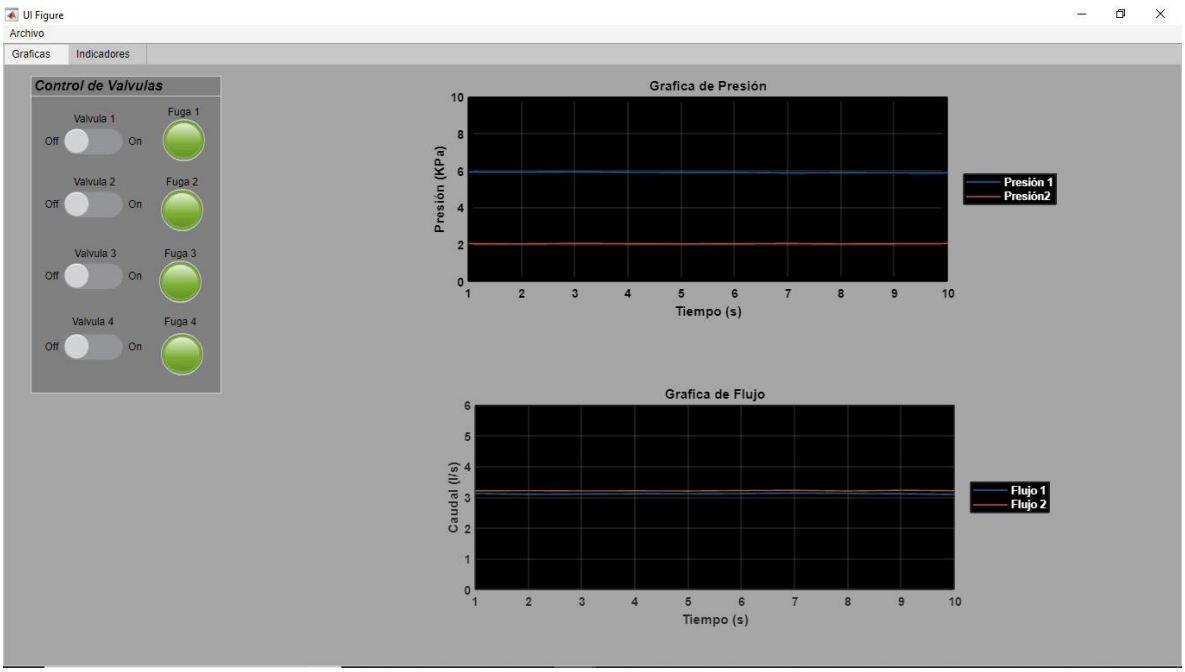


Figura 3.28 Sistema funcionando con la bomba apagada

La imagen anterior se puede apreciar la función de todo el programa, como la bomba está apagada podemos observar que hay una presión mínima de 5.3 a 5.1 KPa y obviamente en los sensores de flujo no marcara nada porque la bomba está apagada y no hay circulación de agua sobre el serpentín.



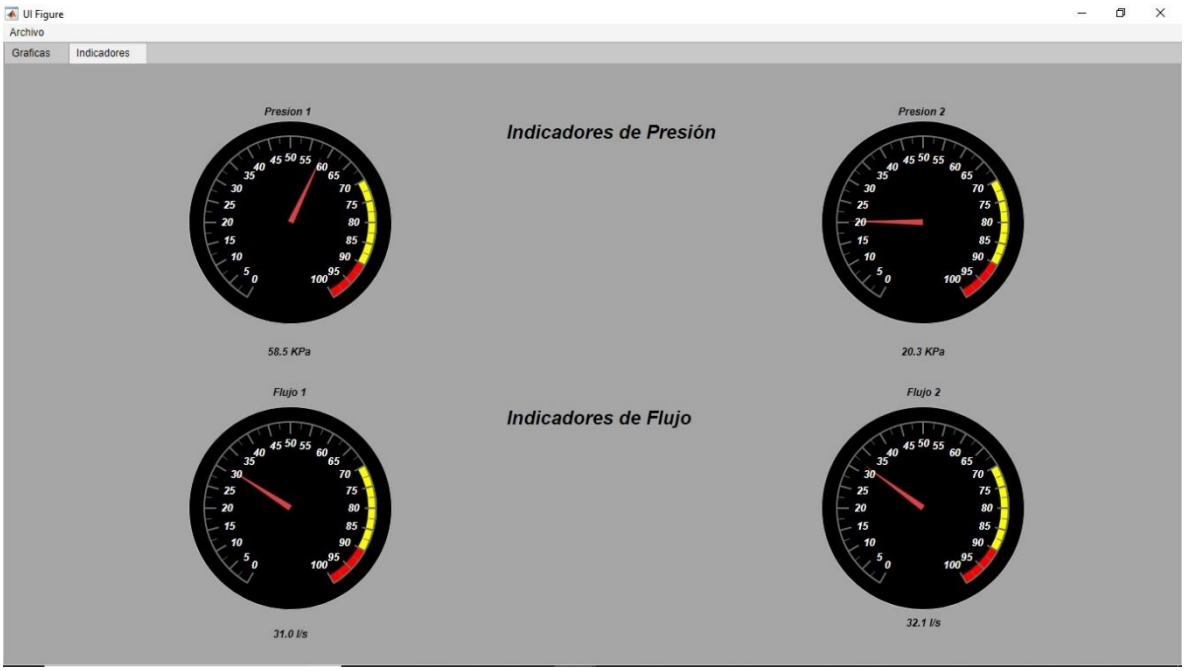


Figura 3.29 Sistema funcionando con la bomba encendida

Una vez encendida la bomba los sensores de caudal empiezan a trabajar. Por el momento se tienen las cuatro electroválvulas apagadas.

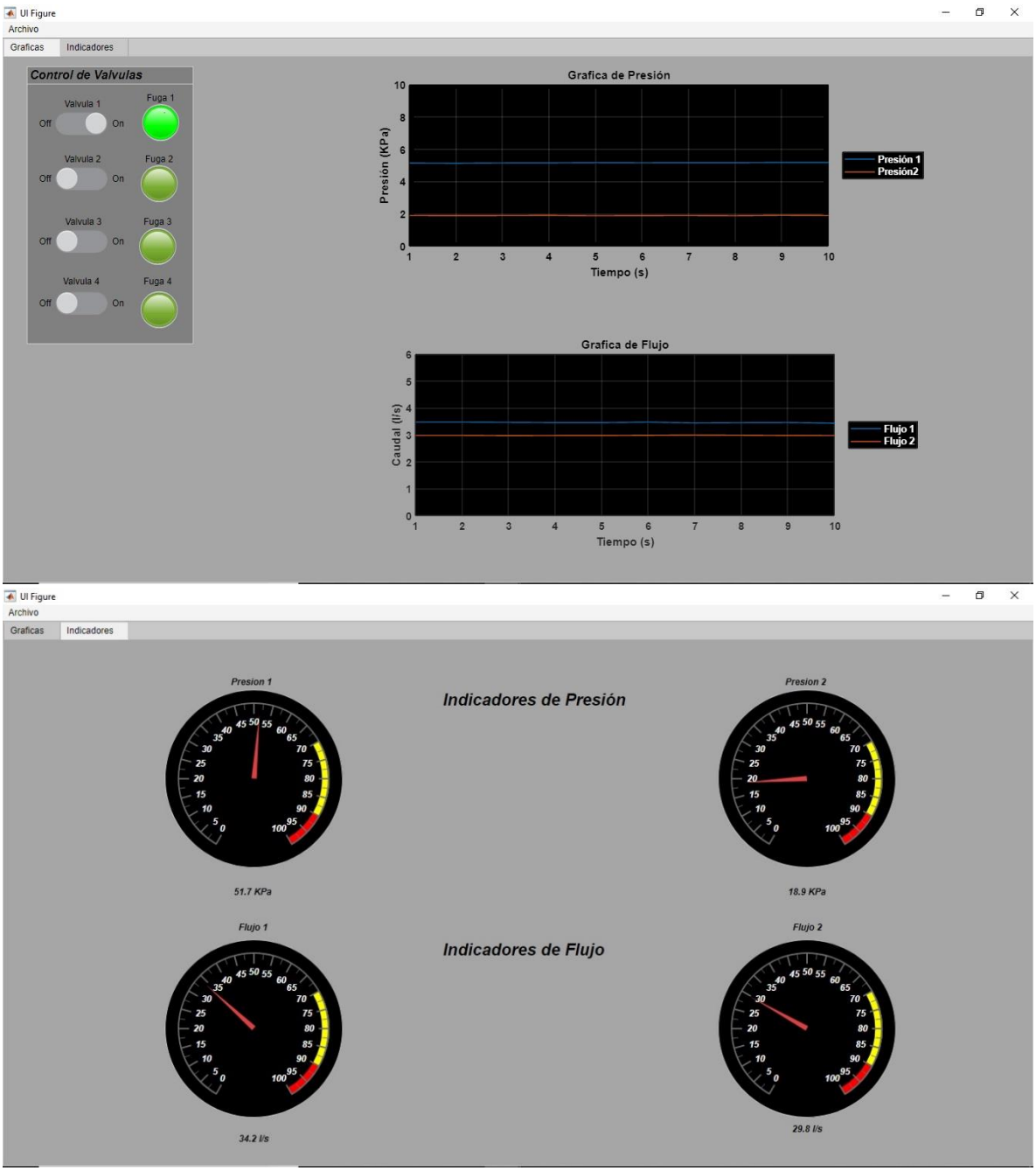


Figura 3.30 Sistema con una electroválvula encendida

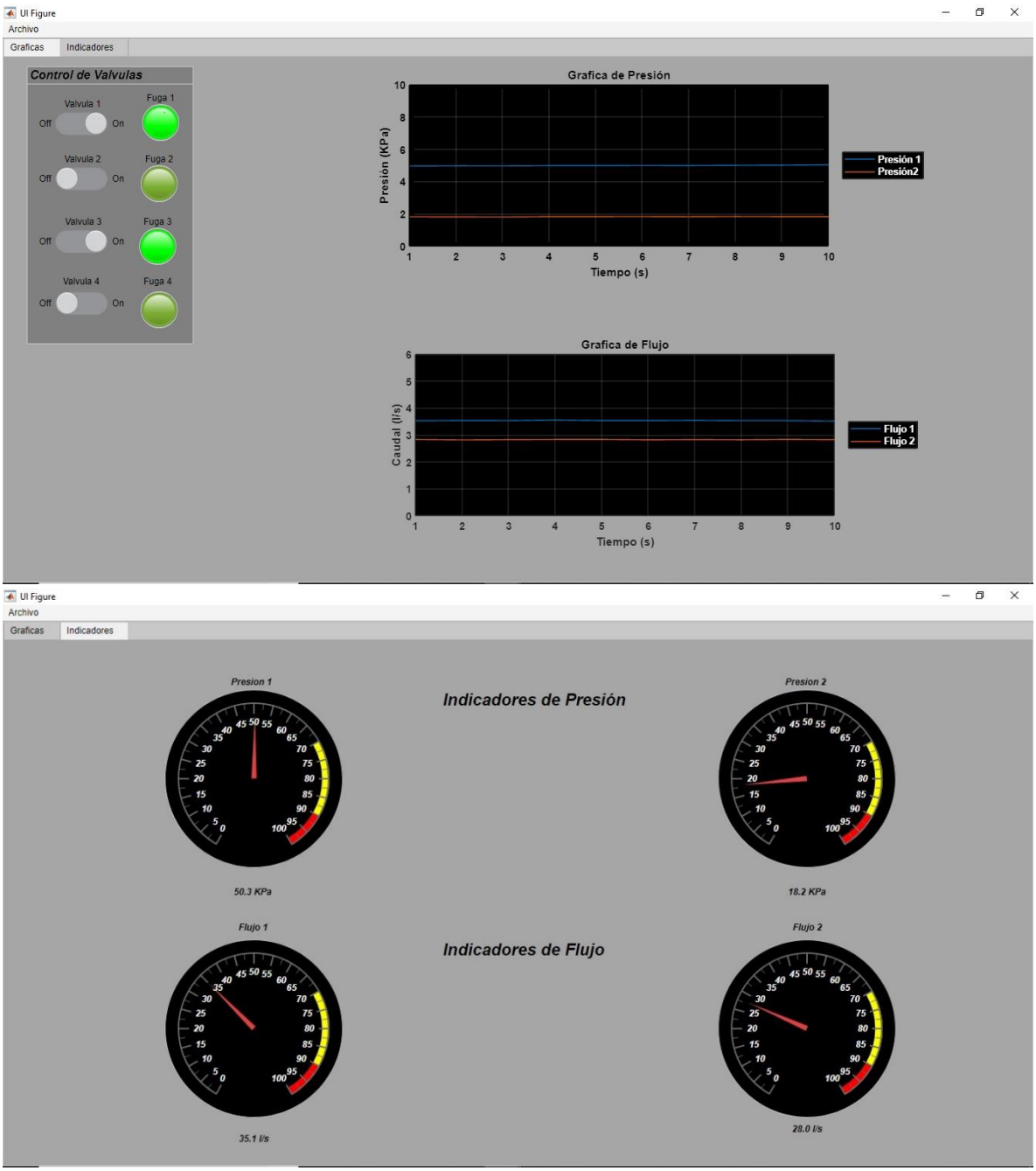


Figura 3.31 Sistema con dos electroválvulas encendidas

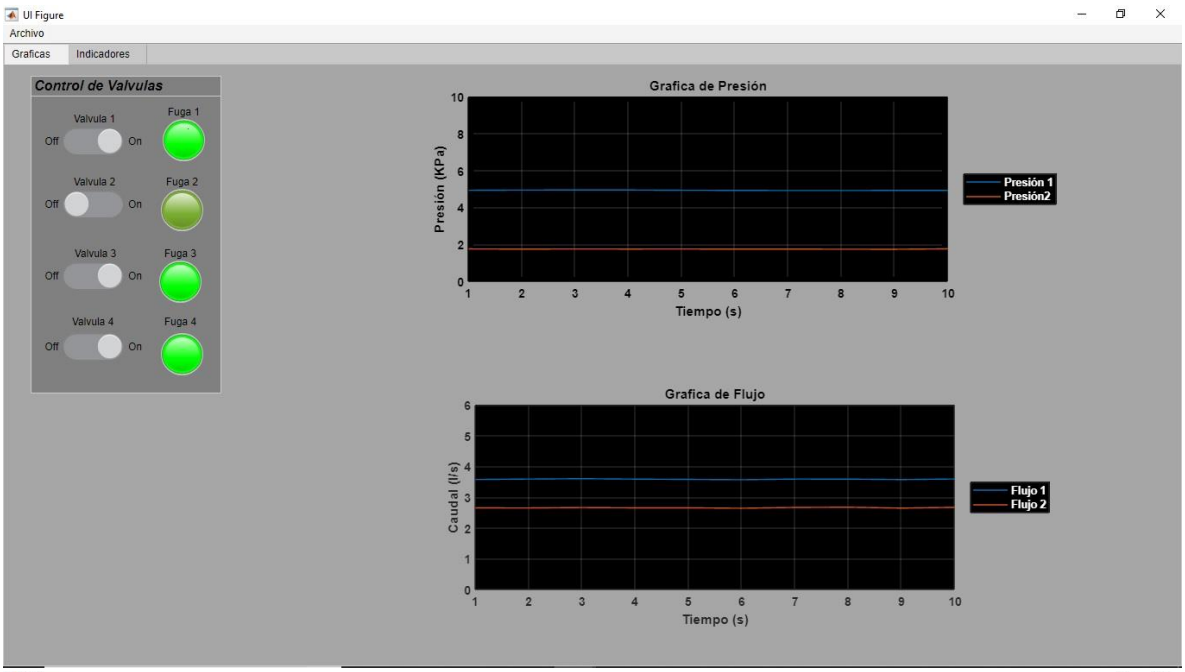
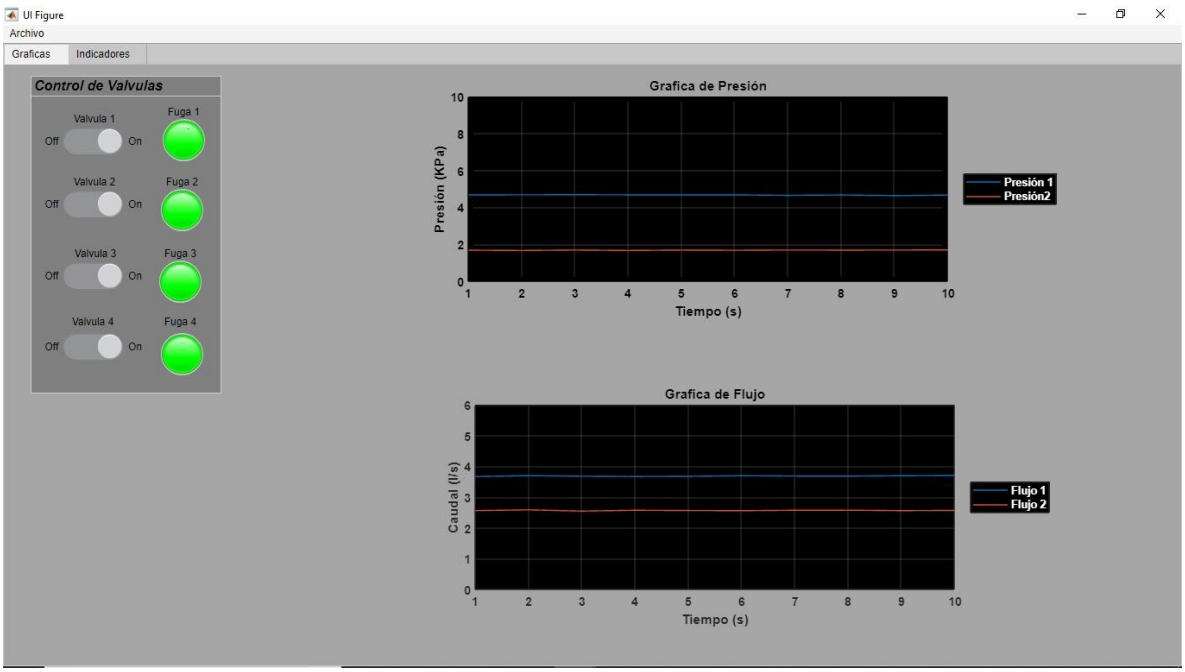


Figura 3.32 Sistema con tres electroválvulas encendidas



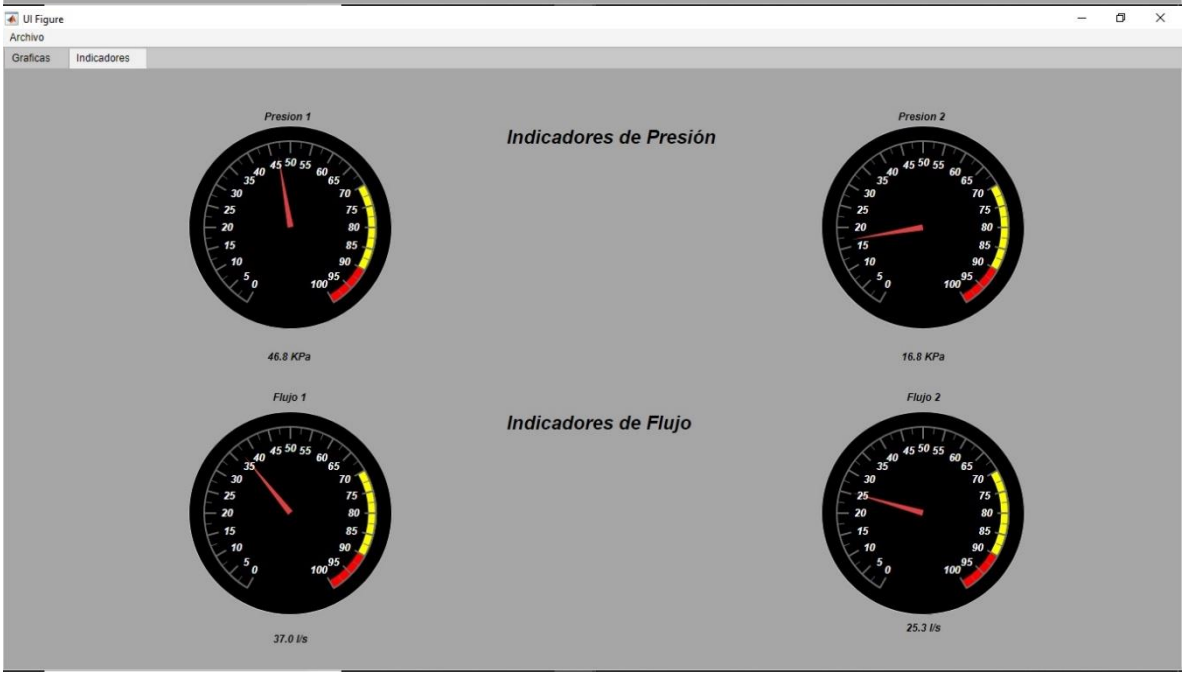


Figura 3.3 Sistema con cuatro electroválvulas encendidas

Como se dijo antes cada vez que se active una electroválvula los valores en los manómetros y en la gráfica cambiaran y volverán a su estado normal apagando todas las electroválvulas.

Conclusión

Las ventajas de usar estas tarjetas de adquisición de datos es de que se puesta en la PC puede tratar a señales analógicas sin problemas, con mucha velocidad de medición y toda la memoria de la PC para almacenamiento de datos y toda la potencia de las procesadoras modernas para su tratamiento. Las complicaciones que se tuvieron en este proyecto se debieron a la completa inexperiencia en este campo.

Mas sin embargo los resultados obtenidos en este proyecto son sin duda satisfactorios ya que se logró alguno de los objetivos que fue diseñar un sistema de monitoreo en tiempo real de los diferentes sensores de caudal y presión a lo largo del serpentín como también el sistema de control de las electroválvulas.

Sin duda el sistema implementado se puede mejorar en trabajos futuros para hacerlo mucho más eficiente, por ejemplo puede introducirle sensores industriales de mejor calidad para mayor precisión de obtención de datos, cambiar las electroválvulas por unas que sean de baja presión al activarse y prácticamente la aplicación que se hiso se puede mejorar agregándole que capture los datos como también programar las electroválvulas para que se activen en determinado tiempo.

Referencias

[1] Billmann, L., & Isermann, R. (1987). Leak Detection Methods for Pipelines. *Automatica,* 23(3), 381-385.

[2] Jowitt, P. (1995). Análisis del impacto de las roturas de tubería en redes de distribución. Mejora de rendimiento y fiabilidad en sistemas de distribución de agua. Curso UIMP.

[3] Sánchez B. y Fuentes M. Método para detectar fugas mayores en una red de agua potable. Revista Ingeniería del Agua, Universidad Politécnica de Valencia, España, 3(1), marzo 1996, ISNN 1134–2196.

[4] Arreguín F., Ochoa L. y Fernández A. Evaluación de pérdidas en redes de distribución de agua. TLALOC–AMH, Órgano informativo de la Asociación Mexicana de Hidráulica (AMH), No.10. 1997.

[5] Mpesha, W., Chaudry, M. N., & Gassman, S. (2001). Leak Detection in Pipes by Frequency Response Method. *Journal of Hydraulic Engineering, 127,* 137-147.

[6] Verde, C. (2001). Multi-leak detection and isolation in fluid pipelines. Control Engineering Practice, 9:673—682.

[7] Verde, C. (2003). Accomodation of multi-leak position in a pipeline. 5th Safeprocess, IFAC, June 9-11.

[8] Verde, C., Bornard, G., and Gentil, S. (2003). Isolability of multi-leaks in a pipeline. 4th MATHMOD Proceedings, February 5-7, ARGESIM report 24, ISBN-3-901608-24-9.

[9] Fuentes M.O.A., Palma N.A., Rivera T.F.G. y Rodríguez V.K. Localización de fugas y determinación de sus gastos en una red de tuberías de agua potable usando un algoritmo genético. XVIII Congreso Nacional de Hidráulica, San Luis Potosí, S. L. P., México, 2004. Tema 11: Agua potable y alcantarillado. P. 1557.

[10] Fuentes M.O.A., Rodríguez V.K., Jímenez M.M.R.,De Luna C.F. y Vega S.B.E. Método para la detección de fugas en redes de distribución de agua potable. Memorias del 3er Seminario Hispano–Brasileño sobre Planificación, Proyecto y Operación de redes de abastecimiento de agua, Universidad Politécnica de Valencia, España. 2004.

[11] Wang, X. J., Lambert, M., Simpson, A., & Vtkovsky, J. (2005). Leak Detection in Pipelines Using the Damping of Fluid Transients. *Journal of Hydraulic Engineering,* 128(7), 697-711.

[12] Rodríguez V.K., Jiménez M.M. R. y Fuentes M.O.A. (2004). Los algoritmos genéticos en la Ingeniería de los sistemas de abastecimiento ¿la modelación del

futuro de las redes de distribución? San Luis Potosí, S. L. P, México: XVIII Congreso Nacional de Hidráulica.

[13] Torres, A. M. (AMCA 2015). Localizacion de multiples fugas usando la onda de presio´n. *Cenidet*, 263-268.

[14] Soledispa, I. V. (Abril del 2015). *Diagnostico, Analisis y Propuesta de un Sistema Optimo de Gestion del Manejo del Agua Potable en la Ciudad de Guayaquil.* Sangolqui.

[15] Ester Sales-Seti´en, D. T. (Gijón · Palacio de Congresos · 6, 7 y 8 de Septiembre de 2017). Estimacion de fugas en un sistema industrial real mediante modelado por señales aditivas. *XXXVIII Jornadas de Automática*, 160-166.

[16] Palomeque, M. P. (Abril 2018). Ingenieria de detalle y simulacion de un sistema de deteccion de fugas en una linea de flujo multifasico. Quito.

[16] Cleve Moler. (1970). Matlab. 2018, de The MathWorks, Inc. Sitio web: https://la.mathworks.com/products/matlab/app-designer.html

Anexo a programación de MatLab

```
classdef Sistema_de_fugas_6 < matlab.apps.AppBase
    % Properties that correspond to app components
    properties (Access = public)
        UIFigure                    matlab.ui.Figure
        ArchivoMenu                 matlab.ui.container.Menu
        SalirMenu                   matlab.ui.container.Menu
        TabGroup                    matlab.ui.container.TabGroup
        GraficasTab                 matlab.ui.container.Tab
        UIAxes                      matlab.ui.control.UIAxes
        UIAxes2                     matlab.ui.control.UIAxes
        ControldeValvulasPanel      matlab.ui.container.Panel
        Fuga1LampLabel              matlab.ui.control.Label
        Fuga1Lamp                   matlab.ui.control.Lamp
        Valvula1SwitchLabel         matlab.ui.control.Label
        Valvula1Switch              matlab.ui.control.Switch
        Valvula3SwitchLabel         matlab.ui.control.Label
        Valvula3Switch              matlab.ui.control.Switch
        Valvula2SwitchLabel         matlab.ui.control.Label
        Valvula2Switch              matlab.ui.control.Switch
        Valvula4SwitchLabel         matlab.ui.control.Label
        Valvula4Switch              matlab.ui.control.Switch
        Fuga2LampLabel              matlab.ui.control.Label
        Fuga2Lamp                   matlab.ui.control.Lamp
        Fuga3LampLabel              matlab.ui.control.Label
        Fuga3Lamp                   matlab.ui.control.Lamp
        Fuga4Lamp                   matlab.ui.control.Lamp
        Fuga4LampLabel              matlab.ui.control.Label
        IndicadoresTab              matlab.ui.container.Tab
        KPsLabel                    matlab.ui.control.Label
        KPsLabel2                   matlab.ui.control.Label
        lsLabel                     matlab.ui.control.Label
        lsLabel2                    matlab.ui.control.Label
        Flujo1GaugeLabel            matlab.ui.control.Label
        Flujo1Gauge                 matlab.ui.control.Gauge
        Flujo2GaugeLabel            matlab.ui.control.Label
        Flujo2Gauge                 matlab.ui.control.Gauge
        Presion1GaugeLabel          matlab.ui.control.Label
        Presion1Gauge               matlab.ui.control.Gauge
        Presion2GaugeLabel          matlab.ui.control.Label
        Presion2Gauge               matlab.ui.control.Gauge
        IndicadoresdePresinLabel    matlab.ui.control.Label
        IndicadoresdeFlujoLabel     matlab.ui.control.Label
    end
    properties (Access = private)
        s; % Crea la sesion de la primera Daq
        estado = [1 1 1 1]; % La matriz indica la ativacion de las valvulas
        se; % Crea la sesion de la segunda Daq
        ch; % Abrir los canales analogicos de la Daq
```

```
        lh;

    end
    methods (Access = private)

        function Grafique(app,objeto,evento)
            dato = evento.Data;
            u = dato(:,1:2);
            y = dato(:,3:4);
            t = evento.TimeStamps;
            TEMP = 27;
            [t,u,y] = mapeo(t,u,y,TEMP);
            P1 = 10*mean(u(:,1));
            P2 = 10*mean(u(:,2));
            F1 = 10*mean(y(:,1));
            F2 = 10*mean(y(:,2));
            app.Presion1Gauge.Value = P1;
            app.KPsLabel.Text = sprintf('%4.1f KPa',P1);
            app.Presion2Gauge.Value = P2;
            app.KPsLabel2.Text = sprintf('%4.1f KPa',P2);
            plot(app.UIAxes,u);
            legend(app.UIAxes,{'Presión 1','Presión
2'},'Location','eastoutside','TextColor','w')
            app.Flujo1Gauge.Value = F1;
            app.lsLabel.Text = sprintf('%4.1f l/s',F1);
            app.Flujo2Gauge.Value = F2;
            app.lsLabel2.Text = sprintf('%4.1f l/s',F2);
            plot(app.UIAxes2,y);
            legend(app.UIAxes2,{'Flujo 1','Flujo
2'},'Location','eastoutside','TextColor','w')

        end

    end


    methods (Access = private)
        % Code that executes after component creation
        function startupFcn(app)
            app.Valvula1Switch.Value = 'Off';
            app.Valvula2Switch.Value = 'Off';
            app.Valvula3Switch.Value = 'Off';
            app.Valvula4Switch.Value = 'Off';
            app.Fuga1Lamp.Color = [.47 .67 .19];
            app.Fuga2Lamp.Color = [.47 .67 .19];
            app.Fuga3Lamp.Color = [.47 .67 .19];
            app.Fuga4Lamp.Color = [.47 .67 .19];
            delete(instrfind)
            app.s = daq.createSession('ni');
            addDigitalChannel(app.s,'Dev1','Port0/Line0:3','OutputOnly');
```

```
            outputSingleScan(app.s,[1,1,1,1]);
            delete(instrfind)

            app.se = daq.createSession('ni');
            app.ch = addAnalogInputChannel(app.se,'cDAQ1Mod1', 0:3,
'Current');
            app.se.Rate = 10;
            app.se.DurationInSeconds = 600;
            app.lh = addlistener(app.se,'DataAvailable', @app.Grafique);
            app.se.NotifyWhenDataAvailableExceeds = 10;
            app.se.startBackground();

            plot(app.UIAxes,0,[0,0]);
            app.UIAxes.YLim = [0,10];
            title(app.UIAxes,'Grafica de Presión');
            xlabel(app.UIAxes,'Tiempo (s)')
            ylabel(app.UIAxes,'Presión (KPa)')
            grid(app.UIAxes,'on')
            plot(app.UIAxes2,0,[0,0]);
            app.UIAxes2.YLim = [0,6];
            title(app.UIAxes2,'Grafica de Flujo');
            xlabel(app.UIAxes2,'Tiempo (s)')
            ylabel(app.UIAxes2,'Caudal (l/s)')
            grid(app.UIAxes2,'on')

        end
        % Value changed function: Valvula1Switch
        function Valvula1SwitchValueChanged(app, event)
            value = app.Valvula1Switch.Value;
            app.estado = xor(app.estado,[0,0,0,1]);
            outputSingleScan(app.s,app.estado);
            actual = app.estado;
            if actual(4) == 1
                app.Fuga1Lamp.Color = [0.47,0.67,0.19];
            else
                app.Fuga1Lamp.Color = [0,1,0];
            end
        end
        % Value changed function: Valvula2Switch
        function Valvula2SwitchValueChanged(app, event)
            value = app.Valvula2Switch.Value;
            app.estado = xor(app.estado,[0,0,1,0]);
            outputSingleScan(app.s,app.estado);
            actual = app.estado;
            if actual(3) == 1
                app.Fuga2Lamp.Color = [0.47,0.67,0.19];
            else
                app.Fuga2Lamp.Color = [0,1,0];
            end
        end
```

```
        % Value changed function: Valvula3Switch
        function Valvula3SwitchValueChanged(app, event)
            value = app.Valvula3Switch.Value;
            app.estado = xor(app.estado,[0,1,0,0]);
            outputSingleScan(app.s,app.estado);
            actual = app.estado;
            if actual(2) == 1
                app.Fuga3Lamp.Color = [0.47,0.67,0.19];
            else
                app.Fuga3Lamp.Color = [0,1,0];
            end
        end
        % Value changed function: Valvula4Switch
        function Valvula4SwitchValueChanged(app, event)
            value = app.Valvula4Switch.Value;
            app.estado = xor(app.estado,[1,0,0,0]);
            outputSingleScan(app.s,app.estado);
            actual = app.estado;
            if actual(1) == 1
                app.Fuga4Lamp.Color = [0.47,0.67,0.19];
            else
                app.Fuga4Lamp.Color = [0,1,0];
            end
        end
        % Menu selected function: SalirMenu
        function UIFigureCloseRequest(app, event)
            delete(app)
        end
    end
    % App initialization and construction
    methods (Access = private)
        % Create UIFigure and components
        function createComponents(app)
            % Create UIFigure
            app.UIFigure = uifigure;
            app.UIFigure.Color = [0.651 0.651 0.651];
            app.UIFigure.Position = [100 100 799 531];
            app.UIFigure.Name = 'UI Figure';
            % Create ArchivoMenu
            app.ArchivoMenu = uimenu(app.UIFigure);
            app.ArchivoMenu.Text = 'Archivo';
            % Create SalirMenu
            app.SalirMenu = uimenu(app.ArchivoMenu);
            app.SalirMenu.MenuSelectedFcn = createCallbackFcn(app,
@UIFigureCloseRequest, true);
            app.SalirMenu.Text = 'Salir';
            % Create TabGroup
            app.TabGroup = uitabgroup(app.UIFigure);
            app.TabGroup.Position = [1 3 799 529];
            % Create GraficasTab
```

```
            app.GraficasTab = uitab(app.TabGroup);
            app.GraficasTab.Title = 'Graficas ';
            app.GraficasTab.BackgroundColor = [0.651 0.651 0.651];
            % Create UIAxes
            app.UIAxes = uiaxes(app.GraficasTab);
            title(app.UIAxes, 'Title')
            xlabel(app.UIAxes, 'X')
            ylabel(app.UIAxes, 'Y')
            app.UIAxes.PlotBoxAspectRatio = [1 0.385542168674699
0.385542168674699];
            app.UIAxes.FontSize = 14;
            app.UIAxes.FontWeight = 'bold';
            app.UIAxes.GridColor = [1 1 1];
            app.UIAxes.MinorGridColor = [1 1 1];
            app.UIAxes.Box = 'on';
            app.UIAxes.XColor = [0 0 0];
            app.UIAxes.YColor = [0 0 0];
            app.UIAxes.ZColor = [0.15 0.15 0.15];
            app.UIAxes.Color = [0 0 0];
            app.UIAxes.XGrid = 'on';
            app.UIAxes.YGrid = 'on';
            app.UIAxes.BackgroundColor = [0.651 0.651 0.651];
            app.UIAxes.Position = [406 307 382 185];
            % Create UIAxes2
            app.UIAxes2 = uiaxes(app.GraficasTab);
            title(app.UIAxes2, 'Title')
            xlabel(app.UIAxes2, 'X')
            ylabel(app.UIAxes2, 'Y')
            app.UIAxes2.PlotBoxAspectRatio = [1 0.385542168674699
0.385542168674699];
            app.UIAxes2.FontSize = 14;
            app.UIAxes2.FontWeight = 'bold';
            app.UIAxes2.GridColor = [0.9412 0.9412 0.9412];
            app.UIAxes2.Box = 'on';
            app.UIAxes2.Color = [0 0 0];
            app.UIAxes2.XGrid = 'on';
            app.UIAxes2.YGrid = 'on';
            app.UIAxes2.BackgroundColor = [0.651 0.651 0.651];
            app.UIAxes2.Position = [406 46 382 185];
            % Create ControldeValvulasPanel
            app.ControldeValvulasPanel = uipanel(app.GraficasTab);
            app.ControldeValvulasPanel.Title = 'Control de Valvulas';
            app.ControldeValvulasPanel.BackgroundColor = [0.502 0.502
0.502];
            app.ControldeValvulasPanel.FontAngle = 'italic';
            app.ControldeValvulasPanel.FontWeight = 'bold';
            app.ControldeValvulasPanel.FontSize = 16;
            app.ControldeValvulasPanel.Position = [32 124 221 368];
            % Create Fuga1LampLabel
            app.Fuga1LampLabel = uilabel(app.ControldeValvulasPanel);
```

```
            app.Fuga1LampLabel.HorizontalAlignment = 'right';
            app.Fuga1LampLabel.Position = [153 315 43 22];
            app.Fuga1LampLabel.Text = 'Fuga 1';
            % Create Fuga1Lamp
            app.Fuga1Lamp = uilamp(app.ControldeValvulasPanel);
            app.Fuga1Lamp.Position = [153 269 49 49];
            app.Fuga1Lamp.Color = [0.4706 0.6706 0.1882];
            % Create Valvula1SwitchLabel
            app.Valvula1SwitchLabel = uilabel(app.ControldeValvulasPanel);
            app.Valvula1SwitchLabel.HorizontalAlignment = 'center';
            app.Valvula1SwitchLabel.Position = [47 307 54 22];
            app.Valvula1SwitchLabel.Text = 'Valvula 1';
            % Create Valvula1Switch
            app.Valvula1Switch = uiswitch(app.ControldeValvulasPanel,
'slider');
            app.Valvula1Switch.ValueChangedFcn = createCallbackFcn(app,
@Valvula1SwitchValueChanged, true);
            app.Valvula1Switch.Position = [38 277 69 31];
            % Create Valvula3SwitchLabel
            app.Valvula3SwitchLabel = uilabel(app.ControldeValvulasPanel);
            app.Valvula3SwitchLabel.HorizontalAlignment = 'center';
            app.Valvula3SwitchLabel.Position = [48 151 54 22];
            app.Valvula3SwitchLabel.Text = 'Valvula 3';
            % Create Valvula3Switch
            app.Valvula3Switch = uiswitch(app.ControldeValvulasPanel,
'slider');
            app.Valvula3Switch.ValueChangedFcn = createCallbackFcn(app,
@Valvula3SwitchValueChanged, true);
            app.Valvula3Switch.Position = [38 121 69 31];
            % Create Valvula2SwitchLabel
            app.Valvula2SwitchLabel = uilabel(app.ControldeValvulasPanel);
            app.Valvula2SwitchLabel.HorizontalAlignment = 'center';
            app.Valvula2SwitchLabel.Position = [47 234 54 22];
            app.Valvula2SwitchLabel.Text = 'Valvula 2';
            % Create Valvula2Switch
            app.Valvula2Switch = uiswitch(app.ControldeValvulasPanel,
'slider');
            app.Valvula2Switch.ValueChangedFcn = createCallbackFcn(app,
@Valvula2SwitchValueChanged, true);
            app.Valvula2Switch.Position = [38 204 69 31];
            % Create Valvula4SwitchLabel
            app.Valvula4SwitchLabel = uilabel(app.ControldeValvulasPanel);
            app.Valvula4SwitchLabel.HorizontalAlignment = 'center';
            app.Valvula4SwitchLabel.Position = [45 72 54 22];
            app.Valvula4SwitchLabel.Text = 'Valvula 4';
            % Create Valvula4Switch
            app.Valvula4Switch = uiswitch(app.ControldeValvulasPanel,
'slider');
            app.Valvula4Switch.ValueChangedFcn = createCallbackFcn(app,
@Valvula4SwitchValueChanged, true);
```

```
app.Valvula4Switch.Position = [38 39 69 31];
% Create Fuga2LampLabel
app.Fuga2LampLabel = uilabel(app.ControldeValvulasPanel);
app.Fuga2LampLabel.HorizontalAlignment = 'right';
app.Fuga2LampLabel.Position = [151 234 43 22];
app.Fuga2LampLabel.Text = 'Fuga 2';
% Create Fuga2Lamp
app.Fuga2Lamp = uilamp(app.ControldeValvulasPanel);
app.Fuga2Lamp.Position = [151 188 49 49];
app.Fuga2Lamp.Color = [0.4706 0.6706 0.1882];
% Create Fuga3LampLabel
app.Fuga3LampLabel = uilabel(app.ControldeValvulasPanel);
app.Fuga3LampLabel.HorizontalAlignment = 'right';
app.Fuga3LampLabel.Position = [149 151 43 22];
app.Fuga3LampLabel.Text = 'Fuga 3';
% Create Fuga3Lamp
app.Fuga3Lamp = uilamp(app.ControldeValvulasPanel);
app.Fuga3Lamp.Position = [149 105 49 49];
app.Fuga3Lamp.Color = [0.4706 0.6706 0.1882];
% Create Fuga4Lamp
app.Fuga4Lamp = uilamp(app.ControldeValvulasPanel);
app.Fuga4Lamp.Position = [151 21 49 49];
app.Fuga4Lamp.Color = [0.4706 0.6706 0.1882];
% Create Fuga4LampLabel
app.Fuga4LampLabel = uilabel(app.ControldeValvulasPanel);
app.Fuga4LampLabel.HorizontalAlignment = 'right';
app.Fuga4LampLabel.Position = [149 72 43 22];
app.Fuga4LampLabel.Text = 'Fuga 4';
% Create IndicadoresTab
app.IndicadoresTab = uitab(app.TabGroup);
app.IndicadoresTab.Title = 'Indicadores';
app.IndicadoresTab.BackgroundColor = [0.651 0.651 0.651];
% Create KPsLabel
app.KPsLabel = uilabel(app.IndicadoresTab);
app.KPsLabel.HorizontalAlignment = 'center';
app.KPsLabel.FontWeight = 'bold';
app.KPsLabel.FontAngle = 'italic';
app.KPsLabel.Position = [162 256 57 22];
app.KPsLabel.Text = 'KPs';
% Create KPsLabel2
app.KPsLabel2 = uilabel(app.IndicadoresTab);
app.KPsLabel2.HorizontalAlignment = 'center';
app.KPsLabel2.FontWeight = 'bold';
app.KPsLabel2.FontAngle = 'italic';
app.KPsLabel2.Position = [614 253 53 28];
app.KPsLabel2.Text = 'KPs';
% Create lsLabel
app.lsLabel = uilabel(app.IndicadoresTab);
app.lsLabel.HorizontalAlignment = 'center';
app.lsLabel.FontWeight = 'bold';
```

```
app.lsLabel.FontAngle = 'italic';
app.lsLabel.Position = [162 25 60 22];
app.lsLabel.Text = 'l/s';
% Create lsLabel2
app.lsLabel2 = uilabel(app.IndicadoresTab);
app.lsLabel2.HorizontalAlignment = 'center';
app.lsLabel2.FontWeight = 'bold';
app.lsLabel2.FontAngle = 'italic';
app.lsLabel2.Position = [611 38 60 22];
app.lsLabel2.Text = 'l/s';
% Create Flujo1GaugeLabel
app.Flujo1GaugeLabel = uilabel(app.IndicadoresTab);
app.Flujo1GaugeLabel.BackgroundColor = [0.651 0.651 0.651];
app.Flujo1GaugeLabel.HorizontalAlignment = 'center';
app.Flujo1GaugeLabel.FontWeight = 'bold';
app.Flujo1GaugeLabel.FontAngle = 'italic';
app.Flujo1GaugeLabel.Position = [169 209 44 22];
app.Flujo1GaugeLabel.Text = 'Flujo 1';
% Create Flujo1Gauge
app.Flujo1Gauge = uigauge(app.IndicadoresTab, 'circular');
app.Flujo1Gauge.ScaleColors = [1 0 0;1 1 0];
app.Flujo1Gauge.ScaleColorLimits = [90 100;70 90];
app.Flujo1Gauge.BackgroundColor = [0 0 0];
app.Flujo1Gauge.FontWeight = 'bold';
app.Flujo1Gauge.FontAngle = 'italic';
app.Flujo1Gauge.FontColor = [1 1 1];
app.Flujo1Gauge.Position = [121 64 140 140];
% Create Flujo2GaugeLabel
app.Flujo2GaugeLabel = uilabel(app.IndicadoresTab);
app.Flujo2GaugeLabel.BackgroundColor = [0.651 0.651 0.651];
app.Flujo2GaugeLabel.HorizontalAlignment = 'center';
app.Flujo2GaugeLabel.FontWeight = 'bold';
app.Flujo2GaugeLabel.FontAngle = 'italic';
app.Flujo2GaugeLabel.Position = [619 209 44 22];
app.Flujo2GaugeLabel.Text = 'Flujo 2';
% Create Flujo2Gauge
app.Flujo2Gauge = uigauge(app.IndicadoresTab, 'circular');
app.Flujo2Gauge.ScaleColors = [1 0 0;1 1 0];
app.Flujo2Gauge.ScaleColorLimits = [90 100;70 90];
app.Flujo2Gauge.BackgroundColor = [0 0 0];
app.Flujo2Gauge.FontWeight = 'bold';
app.Flujo2Gauge.FontAngle = 'italic';
app.Flujo2Gauge.FontColor = [1 1 1];
app.Flujo2Gauge.Position = [571 64 140 140];
% Create Presion1GaugeLabel
app.Presion1GaugeLabel = uilabel(app.IndicadoresTab);
app.Presion1GaugeLabel.BackgroundColor = [0.651 0.651 0.651];
app.Presion1GaugeLabel.HorizontalAlignment = 'center';
app.Presion1GaugeLabel.FontWeight = 'bold';
app.Presion1GaugeLabel.FontAngle = 'italic';
```

```
            app.Presion1GaugeLabel.Position = [159 438 63 22];
            app.Presion1GaugeLabel.Text = 'Presion 1 ';
            % Create Presion1Gauge
            app.Presion1Gauge = uigauge(app.IndicadoresTab, 'circular');
            app.Presion1Gauge.ScaleColors = [1 0 0;1 1 0];
            app.Presion1Gauge.ScaleColorLimits = [90 100;70 90];
            app.Presion1Gauge.BackgroundColor = [0 0 0];
            app.Presion1Gauge.FontWeight = 'bold';
            app.Presion1Gauge.FontAngle = 'italic';
            app.Presion1Gauge.FontColor = [1 1 1];
            app.Presion1Gauge.Position = [121 299 140 140];
            % Create Presion2GaugeLabel
            app.Presion2GaugeLabel = uilabel(app.IndicadoresTab);
            app.Presion2GaugeLabel.BackgroundColor = [0.651 0.651 0.651];
            app.Presion2GaugeLabel.HorizontalAlignment = 'center';
            app.Presion2GaugeLabel.FontWeight = 'bold';
            app.Presion2GaugeLabel.FontAngle = 'italic';
            app.Presion2GaugeLabel.Position = [609 438 63 22];
            app.Presion2GaugeLabel.Text = 'Presion 2 ';
            % Create Presion2Gauge
            app.Presion2Gauge = uigauge(app.IndicadoresTab, 'circular');
            app.Presion2Gauge.ScaleColors = [1 0 0;1 1 0];
            app.Presion2Gauge.ScaleColorLimits = [90 100;70 90];
            app.Presion2Gauge.BackgroundColor = [0 0 0];
            app.Presion2Gauge.FontWeight = 'bold';
            app.Presion2Gauge.FontAngle = 'italic';
            app.Presion2Gauge.FontColor = [1 1 1];
            app.Presion2Gauge.Position = [571 299 140 140];
            % Create IndicadoresdePresinLabel
            app.IndicadoresdePresinLabel = uilabel(app.IndicadoresTab);
            app.IndicadoresdePresinLabel.FontSize = 22;
            app.IndicadoresdePresinLabel.FontWeight = 'bold';
            app.IndicadoresdePresinLabel.FontAngle = 'italic';
            app.IndicadoresdePresinLabel.Position = [300 412 250 27];
            app.IndicadoresdePresinLabel.Text = 'Indicadores de Presión';
            % Create IndicadoresdeFlujoLabel
            app.IndicadoresdeFlujoLabel = uilabel(app.IndicadoresTab);
            app.IndicadoresdeFlujoLabel.FontSize = 22;
            app.IndicadoresdeFlujoLabel.FontWeight = 'bold';
            app.IndicadoresdeFlujoLabel.FontAngle = 'italic';
            app.IndicadoresdeFlujoLabel.Position = [300 177 220 27];
            app.IndicadoresdeFlujoLabel.Text = 'Indicadores de Flujo';
        end
    end
    methods (Access = public)
        % Construct app
        function app = Sistema_de_fugas_6
            % Create and configure components
            createComponents(app)
            % Register the app with App Designer
```

```
            registerApp(app, app.UIFigure)
            % Execute the startup function
            runStartupFcn(app, @startupFcn)
            if nargout == 0
                clear app
            end
        end
        % Code that executes before app deletion
        function delete(app)
            % Delete UIFigure when app is deleted
            delete(app.UIFigure)
        end
    end
end
```