FACULTAD
**DE INGENIERIA**
Universidad de Buenos Aires

CARRERA DE ESPECIALIZACIÓN EN
SISTEMAS EMBEBIDOS

MEMORIA DEL TRABAJO FINAL

# Automatización y control de redes de distribución de agua


**Autor:**
**Ing. Luis Mariano Campos**

Director:
Mg. Ing. Eric Pernia (UNQ/FIUBA)

Jurados:
Esp. Ing. Franco Bucafusco (FIUBA)
Mg. Ing. Leonardo Carducci (FIUBA)
Ing. Marcelo Romeo (UNSAM)


*Este trabajo fue realizado en la ciudad de Tucumán,*
*entre marzo de 2019 y Abril de 2021.*

# *Resumen*

El presente documento trata el diseño y construcción de un prototipo a escala para el control automático del caudal de agua a canal abierto. El sistema desarrollado tiene como finalidad principal realizar una gestión eficiente de una red de canales. Esto permite resolver problemas de disponibilidad en regiones con cierto grado de sequía, así como mejorar el suministro a cada usuario y optimizar la gestión por parte del proveedor.

Se aplicó un algoritmo PI para la función de control. Se diseñó una válvula de control servoalimentada implementada con un motor paso a paso y se la utilizó como elemento actuador. La medición de la variable controlada, se realizó mediante un caudalímetro que fue diseñado e implementado mediante una placa de aforo triangular cuyas características dimensionales se determinaron en forma empírica. El componente principal de la etapa de control es un microcontrolador con arquitectura ARM Cortex-M4, en el que se implementó una aplicación que se ejecuta empleando un sistema operativo de tiempo real. Además se desarrollaron interfaces de programación de aplicaciones (API) necesarias para la modularización del software.

# Índice general

# Índice de figuras

# Índice de tablas

***Dedicado a mis padres Pedro y Nora por su apoyo incondicional.***

# Capítulo 1

# Introducción general

Este capítulo introduce al lector sobre la conformación de una red de canales de agua a cielo abierto, las dificultades que presentan actualmente, y la necesidad de desarrollar, como solución, un sistema tecnológico que permita gestionar el recurso hídrico de manera eficiente.

## 1.1. Descripción del estado actual sobre el manejo de agua en canales abiertos

Una red de canales de riego es una importante zona geográfica que puede definirse como un conjunto de canales de riego [1]. Estos se componen por una o más fuentes comunes de abastecimiento. Su función principal es conducir el agua desde la captación hasta los campos agrícolas o parcelas donde se aplicarán a los cultivos. Por decreto del poder ejecutivo se otorgan títulos de concesión a usuarios que se encuentran agrupados en asociaciones civiles para el uso, administración, operación y conservación tanto del agua como de la infraestructura hidrológica. Puede concluirse que una red de canales de riego es mucho más que una colección de agua, infraestructura y superficie, ya que implica además aspectos legales, administrativos, socioeconómicos y productivos muy importantes e interdependientes entre sí. Para tener una idea más general sobre cómo está constituido generalmente un conjunto de red de canales de riego, la figura 1.1, brinda mayores detalles.

Este sistema de distribución, que se puede apreciar en la figura 1.1, cuenta con 128 kilómetros de canales de tierra, 64 kilómetros de canales revestidos con hormigón y 400 kilómetros de acequias de tierra en general para caudales inferiores a los 150 l/seg. Estos diferentes puntos, son visitados a diario por los operarios que recorren desde pocos hasta varios kilómetros, algunos dos o más veces por día, para verificar, operar (aumentar o disminuir caudal), medir y/o aforar (medir nivel de agua) de forma manual. Luego, según los valores obtenidos en mediciones se deben llevar a cabo cálculos precisos para fijar un determinado caudal. Algunos son de mayor o menor importancia, ya que corresponden a canales principales, secundarios, terciarios, cuaternarios o acequias que entregan agua a un grupo reducido de usuarios. Es interesante destacar que la operación de mover el agua hasta un usuario específico, en general, depende de varios movimientos de compuertas en cadena.


FIGURA 1.1. Sistema de distribución de agua de la provincia de Jujuy.

Tradicionalmente, se ha denominado como "operación de canales de riego" o simplemente "operación", al conjunto de actividades y aspectos realizados directamente en la infraestructura hidráulica con objeto de planear, programar, distribuir y entregar el agua de riego a los productores en forma eficiente y oportuna. Estas condiciones de eficiencia y oportunidad son fundamentales en la tarea de operación.

## 1.2. Problemáticas actuales en la gestión del recurso hídrico y posible solución tecnológica

La problemática del agua es central en el sostenimiento de los sistemas socio-productivos de distintas regiones de nuestro país. Las fuentes del vital recurso para la vida son escasas y en muchas circunstancias son aprovechadas de forma deficiente o bien el acceso a ellas se ve limitado por diversas causas. En este contexto, existen organismos tanto públicos como privados que dan diagnósticos y resaltan dicha problemática de escasez, baja calidad y formas precarias de aprovechamiento de agua, muchas veces sin poder resolverla por cuestiones referidas a la disponibilidad de tecnologías apropiadas, a la organización de la demanda y la gestión.

De acuerdo a lo manifestado, se entiende que la gestión de canales y redes de distribución de agua en la mayoría de las regiones del país se basa en operaciones de control manual en la que los operarios supervisan el estado de cada elemento y actúan en función de protocolos establecidos.

Actualmente, estos procedimientos presentan una capacidad de reacción lenta ante la demanda variable de cada uno de los usuarios de agua, y esto, a su vez, representa un costo sumamente elevado al momento de mantener este servicio.

En estas circunstancias, es posible reconocer algunas situaciones problemáticas y sus posibles soluciones. Por ejemplo, cuando el recurso es en abundancia se puede entregar mayores caudales a los solicitados. De esta forma, se garantiza el suministro a todos los usuarios, lo que permite identificar dos situaciones posibles, grandes disminuciones por derrames y baja eficiencia en la gestión del agua.

Es importante clasificar las pérdidas de agua en canales abiertos en intrínsecas y operacionales. Las primeras se refieren a las pérdidas por evaporación, infiltración y fugas. Es decir, aquellas que se deben a las condiciones climáticas, textura de suelos, longitud y estado de conservación de la red de canales, que son relativamente constantes para cada canal. Las segundas se generan por causas relacionadas, fundamentalmente, con el manejo del agua y que, por lo general, son función del caudal manejado, experiencia del personal y supervisión realizada.

Las pérdidas por infiltración se pueden disminuir con revestimientos, lo que significa grandes inversiones, o en menor medida, manejando los niveles de diseño, ya que estos corresponden a la máxima eficiencia y mínima filtración. En lo que respecta a las pérdidas por operación, se reconoce que dichas pérdidas son aceptables si son menores de 5 %. Se considera que un canal de agua de riego está bien operado si las pérdidas operacionales se mantienen entre 5 % y 10 %. Este tipo de pérdidas es mayor en prácticamente todos los casos a nivel país.

Con el propósito esencial de disminuir las pérdidas operacionales, en la figura 1.2 se muestra un esquema que realizaría una mejor gestión. Además, se puede apreciar que involucra la utilización de diferentes herramientas tecnológicas. El desarrollo comprende una segunda instancia al presente trabajo.

El esquema de la figura 1.2, tiene por objetivo principal resolver los problemas de disponibilidad en regiones que presentan, entre otros factores, cierto grado de sequía. Además, no sólo busca maximizar la eficiencia en el abastecimiento a cada uno de los usuarios en tiempo y forma, sino también minimizar las pérdidas en las redes de canales, y así, poder brindar mayor seguridad en el suministro de agua para el riego a cada productor en sus predios. Con base en lo descrito, se puede también determinar que el principal beneficiario al optimizar la gestión del vital recurso hídrico es el proveedor de dicho recurso.

El esquema presentado, está compuesto principalmente por la automatización de cada una de las compuertas que constituyen la red y un centro de monitoreo donde reside el sistema principal. La comunicación de las compuertas entre sí y el sistema principal se llevará a cabo mediante el empleo de la tecnología LoraWan. Esta red de baja potencia y de área amplia ofrece la posibilidad de comunicar puestos de trabajo o dispositivos cuyas distancias pueden circunscribirse, desde unos pocos, hasta varios kilómetros de separación. Algunos puntos se encuentran situados en zonas donde la cobertura telefónica, internet y energía eléctrica no tienen alcance. Para resolver la carencia de energía eléctrica, se utilizaría energía solar fotovoltaica. Además, se desarrollará una aplicación móvil para que el operario realice gestiones de forma remota.

En relación a la automatización de las compuertas es prácticamente idéntica en cada una de ellas. En la figura 1.2, podemos apreciar que es necesario la utilización de un motor, un microcontrolador como componente principal, un caudalímetro para monitorear el flujo de agua en un determinado instante, un panel solar y un dispositivo perteneciente a la tecnología LoraWan que admite la comunicación entre los diversos puestos o puntos de trabajo.

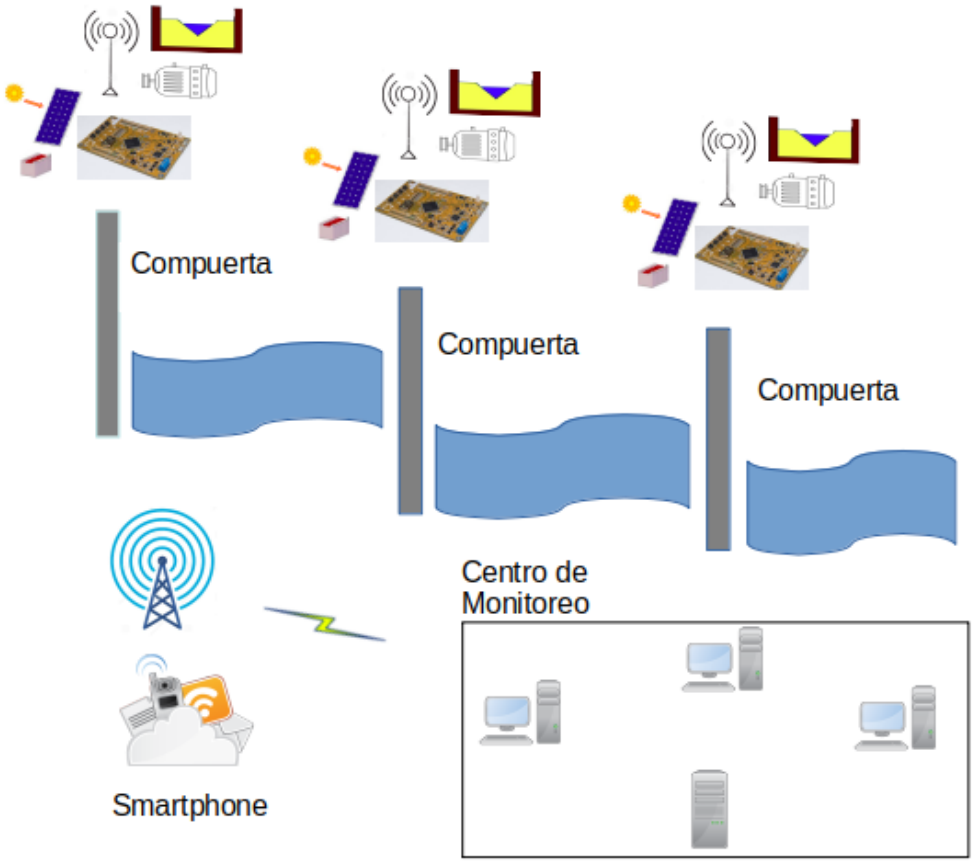


FIGURA 1.2. Diagrama general.

El funcionamiento, a nivel sistema, consistiría en realizar movimientos necesarios de diversas compuertas para suministrar la cantidad de agua en tiempo y forma. Esta maniobra se llevaría a cabo por el operario desde el centro de monitoreo, o bien, se podría definir una determinada operación preprogramada. De modo que el sistema trabajaría de forma autónoma por medio de comunicaciones entre compuertas.

Se denominó celda primaria a una fracción pequeña del sistema global, que en general, se reitera en cada una de las compuertas que integran una red de canal. Esta celda primaria, en términos generales, está constituida principalmente por un firmware que controla el movimiento de una compuerta por medio de un motor eléctrico y, utilizando un caudalímetro, se regula el caudal de agua a un valor especificado por el usuario. A modo de inicio de este proyecto global, este documento describe de forma detallada la construcción de un prototipo a escala que permite ejecutar esta celda primaria perteneciente al sistema completo como se puede apreciar en la figura 1.3.

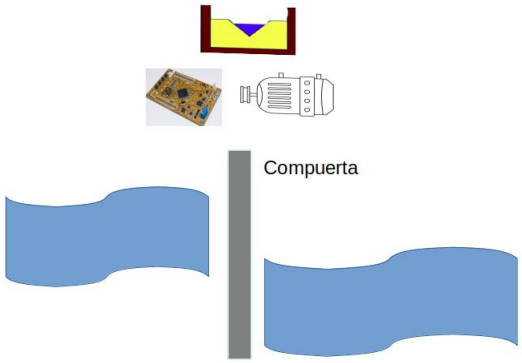


FIGURA 1.3. Celda primaria.

## 1.3. Motivación

Actualmente, en algunas provincias del país la situación hídrica, en las redes de canales, carece de mantenimiento y de control en el manejo de agua. Esto pone en evidencia las necesidades concretas que existen en diversas asociaciones que administran el recurso. En la provincia de Jujuy, más precisamente el Consorcio de Riego del Valle de Perico, si bien, en general, no presentan problemas por la falta de agua, carecen de un control preciso en la cantidad de agua que se debe suministrar a cada uno de los usuarios empadronados. A lo largo del desarrollo de este proyecto los administradores brindaron información al respecto y también mencionaron las dificultades que resultan al realizar movimientos de compuertas que se encuentran alejadas, a varios kilómetros de distancia, en diferentes puntos de la provincia y que, en muchas circunstancias, se deben visitar dos o más veces al día elevando los costos de traslados.

Por otro lado, en la provincia de La Rioja, el principal inconveniente que presentan los productores pequeños que subsisten del agua superficial, es la falta de agua. Hay una gran cantidad de superficie que ocupan estos productores con diversos cultivos. Esto ocasiona que el agua que se les suministra a cada uno de los turnados no les alcance para cubrir las necesidades hídricas del cultivo. Esta necesidad se intensifica principalmente en los meses de octubre, noviembre y diciembre. Las principales causas de la falta de disponibilidad se deben a la ineficiencia de los sistemas de recolección de agua que no aseguran una óptima captación y a la deficiencia en la supervisión de apertura, cierre de compuertas y conducción. Además, se suma la falta de verificación de caudal, debido a la inexistencia de un instrumento de medición que registre variaciones. Y, por la sencilla manipulación de las compuertas, existen casos de consumo ilegal por parte de usuarios con la intención de conducir el agua hacia sus cultivos. Estas causas son las que motivaron llevar a cabo este proyecto.

En este punto es importante destacar que no existen soluciones nacionales, por

lo tanto, el desafío es lograr ser competitivo en precio y prestaciones con respecto a productos importados que se encuentran en el mercado. Esto trae aparejado, desde el punto de vista del cliente el beneficio de contar con soporte local y la posibilidad de modificar algún aspecto del sistema según las necesidades del cliente.

## 1.4. Objetivos y alcances

El objetivo general de este trabajo es aportar el diseño y elaboración de un sistema que mediante el empleo de un algoritmo de control PID permita regular el caudal de agua en canal abierto y con el uso de las tecnologías más actuales pueda ser empleado tanto en red de canales complejos hasta las redes más simples. A continuación se detallan los subobjetivos específicos que se tuvieron en cuenta :

- Diseño y construcción de un sistema de control que permita al usuario establecer un caudal de agua por medio de una aplicación externa.
- Diseño y fabricación de un circuito de control de señales cuya finalidad es cumplir la función de interfaz entre la etapa de control y la de potencia correspondiente al driver del servomotor paso a paso.
- Incluir como parte del firmware un sistema de tiempo real.
- Para regular el caudal de agua, aplicar un algoritmo de control PID.
- Empleo de un sensor de presión para detectar el nivel de la superficie del agua.
- La apertura, control y cierre de las compuertas, se realizará mediante un sistema de servomotor, paso a paso, cuyo ángulo de actuación estará medido y controlado a través de un sensor potenciométrico resistivo.
- Proporcionar una gestión de agua de riego de forma eficiente.
- Desarrollar un firmware que sea escalable que permita la comunicación con otras celdas primarias dispuestas en otras compuertas.

# Capítulo 2

# Introducción específica

En este capítulo se abordarán temas referentes al análisis de la estructura del sistema que se desarrolló, gestión, planificación del trabajo y técnicas relacionadas al sensor empleado.

## 2.1. Estructura general del sistema

En un plano para la construcción del prototipo a escala, se pudo reconocer un inconveniente. Al emplear una compuerta que se desplaza verticalmente y ubicada en forma perpendicular al canal, se generarían filtraciones de agua en sus guías de desplazamiento. Como se trata de una maqueta a escala, estas filtraciones provocarían errores considerables al momento de realizar las mediciones relacionadas con la altura de la superficie de agua, y cálculos pertinentes al aplicar el método de aforo por compuerta para determinar el caudal [2]. Por lo tanto, en el sistema no se obtendrían los resultados esperados. En este contexto, el caudal que circularía por debajo de la compuerta sería menor que el que se registraría en el caudalímetro. De esta forma, se obtendrían dos valores de caudales diferentes debido, entre otros factores, a las filtraciones. Esto motivo a reemplazar, en el prototipo, la compuerta por una válvula y así eliminar los errores que introducen dichas filtraciones. Para el control de la válvula se elaboró un servomotor con el empleo de un motor paso a paso y su correspondiente controlador. En la figura 2.1, se puede apreciar un vertedero triangular en un canal abierto y una regla que mide el nivel de la superficie de agua.

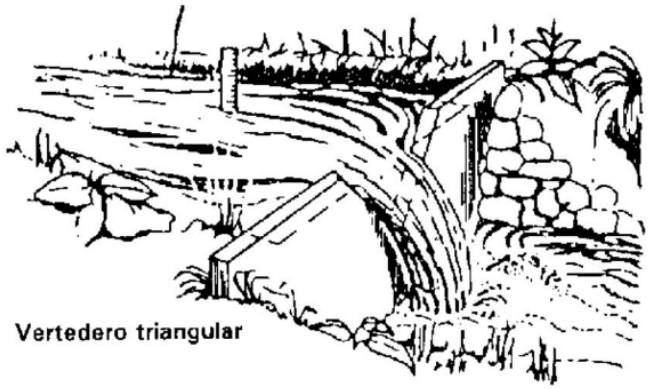


FIGURA 2.1. Ilustración de un vertedero triangular en el canal[1].

[1]Imagen tomada de http://www.fao.org/3/t0848s/t0848s06.htm

Para medir el caudal, se fabricó un caudalímetro por placa de aforo triangular. Estos tipos de instrumentos, también denominados vertederos [3], representan un dique o pared que intercepta en un determinado punto una corriente de líquido, en este caso agua, con superficie libre, como se puede apreciar en la figura 2.1 . En general, se utilizan para mantener un nivel de superficie de aguas arriba que no exceda un valor límite, o bien para medir el caudal de agua circulante por un canal. Este instrumento resulta un medidor de caudal sencillo pero efectivo en canales abiertos.

El principio de funcionamiento de la celda primaria consiste en establecer el caudal de agua a un valor establecido por el usuario. Para llevar a cabo esto, fue necesario el desarrollo de un firmware para controlar el eje de un servomotor. Este se encuentra adherido a una válvula, lo que permite maniobrar su rango de apertura para regular el flujo del fluido según las necesidades.

El diagrama de control correspondiente es el que se muestra en la figura 2.2 donde se aprecia que la salida del servomotor actúa directamente sobre la válvula. Esta puede ser una llave esférica, como es este el caso, o un sistema de cremallera. Este tipo de sistemas se aplican a compuertas verticales, como las que podemos encontrar frecuentemente en los canales de agua a cielo abierto. La salida de la válvula es el flujo controlado de caudal al canal. En este punto se coloca un medidor de caudal. Luego su resultado se compara con el set point, valor de caudal fijado por el usuario. En el sumador, a partir de la diferencia que existe entre el set point y el valor de caudal medido se obtiene una señal de error, que seguidamente se procesa por un algoritmo de control PID para que sea igual a cero.

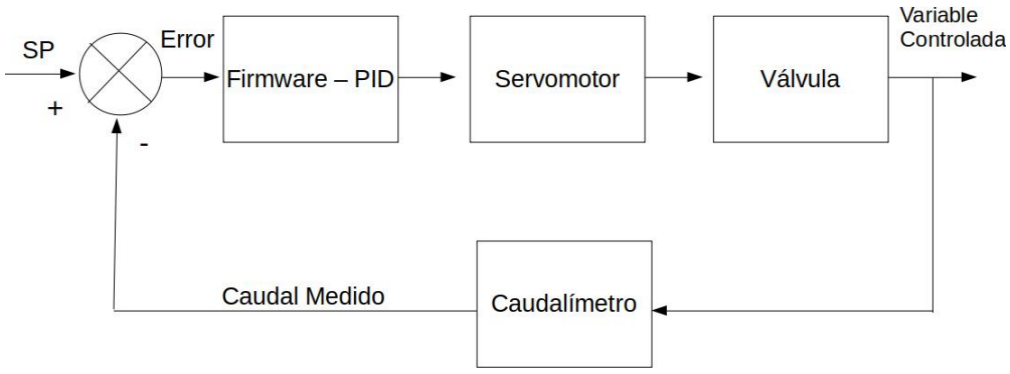


FIGURA 2.2. Diagrama en bloque de control - celda primaria.

## 2.2. Compuerta Miller

La compuerta Miller es el mecanismo más utilizado en los canales de agua de riego para controlar la entrada de agua de los canales ramales y tomas directas a las parcelas. Están construidas de fierro fundido (vaciado o colado) por su resistencia a la oxidación. Sin embargo, son frágiles y poco maleables. La toma Miller, en esencia, es una compuerta circular que obtura la entrada a la tubería de salida, y es izada por un mecanismo elevador compuesto de un vástago cilíndrico con cuerda tipo tornillo (roscas), generalmente de 2" de diámetro y longitud variable, en función de la altura a colocar la toma y un volante. Las tomas-granja Miller se

clasifican por el diámetro de la tubería a obturar, y son de 18" y 24" las más comunes para tomas-granjas, mientras que las de 30" y 36" son usadas para abastecer ramales y subramales. Las ventajas de este tipo de tomas es que son relativamente baratas, la mayoría de las fundidoras las pueden construir y son de fácil colocación. Como desventajas, en primer lugar destacan que pueden tener filtraciones considerables, puesto que el material de la tubería y de la compuerta (comal) dificultan un cierre totalmente hermético, especialmente si, durante su fundición, no hay cuidado de tener acabados completamente a nivel. En segundo lugar, la calibración de esta estructura es difícil, ya que al abrirse parcialmente la compuerta sobre la tubería ambas de figura circular, se forman secciones tipo "media luna" con área hidráulica variable, sin seguir un patrón de fácil cálculo. En las figuras 2.3, 2.4 y 2.5 se puede apreciar como es la estructura de la compuerta, el conducto para la toma y su colocación.

FIGURA 2.3. Compuerta tipo Miller para toma-granja.

FIGURA 2.4. Conducto preparado para colocar compuerta tipo Miller.

FIGURA 2.5. Toma-granja tipo Miller de compuerta circular de 18" de diámetro.

Este mecanismo que se mostró y que se emplea en muchos países para el control de entrada de agua a los canales principales o bien a las parcelas, impulsó a utilizar, en el prototipo a escala, una válvula esférica simulando ser una compuerta Miller.

## 2.3. Requerimiento a nivel de software y hardware

En esta sección se presentan los requerimientos específicos de software y hardware del sistema. Estos se tuvieron en cuenta al momento de inicio del desarrollo de este proyecto, cuyo fin es, mediante el control de movimiento de una válvula, regular el caudal de agua según las necesidades del usuario.

### 2.3.1. Requisitos específicos del software

Los requisitos específicos de software fueron:

1. El firmware deberá generar como señal, trenes de pulsos para controlar el movimiento del eje perteneciente al servomotor.
2. El firmware deberá determinar, mediante el sensor de presión en conjunto con la placa de aforo triangular, el caudal de agua que fluye a través de este instrumento de medición.
3. El firmware deberá incluir un algoritmo de control PID, que por medio de un lazo de retroalimentación permitiese regular la variable a controlar.
4. El firmware deberá ser capaz, a través de un pin configurado como salida, establecer el sentido de giro del eje del servomotor, horario - antihorario.
5. El firmware deberá ser capaz, a través de un pin configurado como salida, habilitar - deshabilitar el servomotor.

6. El firmware deberá interactuar, mediante el empleo del puerto serial, con otra aplicación.
7. Con un protocolo de comunicación definido entre la aplicación externa y el firmware, este último deberá ser capaz de recibir y enviar datos.
8. El firmware deberá enviar una notificación de correcta recepción de cualquier comando mediante el envío de un comando específico hacia la aplicación externa.
9. El firmware deberá reportar o notificar a la aplicación externa la recepción de un comando inválido mediante un comando específico.
10. En caso de recepción de comandos válidos, el firmware deberá informar internamente que hay datos a procesar.

### 2.3.2. Requisitos específicos del hardware

Los requisitos específicos principales de hardware fueron:

1. Se hará uso de la placa EDU-CIAA-NXP [4] como computadora principal para el prototipo a escala.
2. Se construirá un mecanizado de válvula de control estándar mediante un servomotor energizado paso a paso y un elemento medidor de ángulo tipo potenciométrico resistivo.
3. Se construirá un medidor de caudal por canal de aforo utilizando para la medición de la altura de nivel de agua un sensor MPX5010DP.
4. Se diseñará y fabricará un circuito como interfaz que controlará las señales enviadas desde la placa EDU-CIAA-NXP al controlador del servomotor paso a paso.

## 2.4. Características de hardware propias del equipo

### 2.4.1. Motor paso a paso y controlador

Para la construcción de la válvula de control fue preciso emplear un motor que tenga la capacidad de mover una llave esférica. Para este proyecto, por cuestión de disponibilidad, se utilizó un motor paso a paso y un controlador cuyo modelos son 86HS85 y MA860H respectivamente. El motor paso a paso de 8 hilos posee un torque nominal de 8,5 Nm, suficiente como para realizar movimientos de apertura parcial o total y cierre total de la carrera de una válvula, según las necesidades requeridas de caudal. El MA860H [5] es un controlador para motores paso a paso compatibles con motores 86HS85 [6] con las siguientes características principales:

1. Permite ajustar la corriente que se dirige hacia el motor.
2. Posibilita controlar al motor hasta en 200 micropasos.
3. Frecuencia de entrada de tren de pulso hasta 300 kHz.
4. Corriente de salida hasta 7,2 A.

5. Entrada TTL compatible y ópticamente aislada.
6. Soporta modos PUL / DIR y CW / CCW.

El MA860H tiene dos conectores, el conector P1 para conexiones de señales de control y el conector P2 para conexiones de potencia y motor. Para el control de la válvula fue necesario un motor que pueda producir un momento de 1 Nm o más.

**Configuraciones del conector P1**

- Señal de pulso: esta entrada representa la señal de pulso, activa en cada flanco ascendente o descendente (para este trabajo se encuentra configurado en flanco ascendente). Entre 4 V y 5 V equivale a un pulso alto, entre 0 V y 5 V a un pulso bajo. Para una respuesta confiable, el ancho de pulso debe ser superior a 1,5 $\mu$s.
- Señal de dirección: esta señal tiene niveles de voltaje bajo y alto, que representan dos direcciones de rotación del motor. Para una respuesta de movimiento confiable, la señal de dirección debe estar por delante de la señal de pulso por lo menos 5 $\mu$s. Entre 4 V y 5 V equivale a un pulso alto, entre 0 V y 5 V a un pulso bajo.
- Señal de habilitación: esta señal se utiliza para habilitar/deshabilitar el controlador. Nivel alto para habilitar y nivel bajo para inhabilitar al controlador.

### 2.4.2. Sensor de presión

Para monitorizar el nivel de la columna de agua, necesario para determinar el caudal en un determinado instante, se optó por su alta precisión un sensor de presión cuyo modelo es MPX5010DP [7]. Este transductor piezo-resistivo, es un sensor de presión de silicio monolítico que puede ser utilizado en aplicaciones que disponen de un microcontrolador o microprocesador con entradas ADC. El MPX5010DP entrega a su salida un rango de voltaje entre 0 V y 5 V, mediante esta información y una fórmula que se expone en el siguiente párrafo se puede estimar la presión hasta un valor de 10 kPa, esto es, una columna de agua de hasta 100 cm de altura. El dispositivo proporciona una salida lineal como se puede observar en la figura 2.6, extraída de su hoja de datos.

Como se pudo observar la función de la figura 2.6, la ecuación para obtener la presión del sensor viene dada por la ecuación 2.1:

$$P = \frac{Vout - 0{,}04 * Vs \pm Tol}{0{,}09 * Vs} \tag{2.1}$$

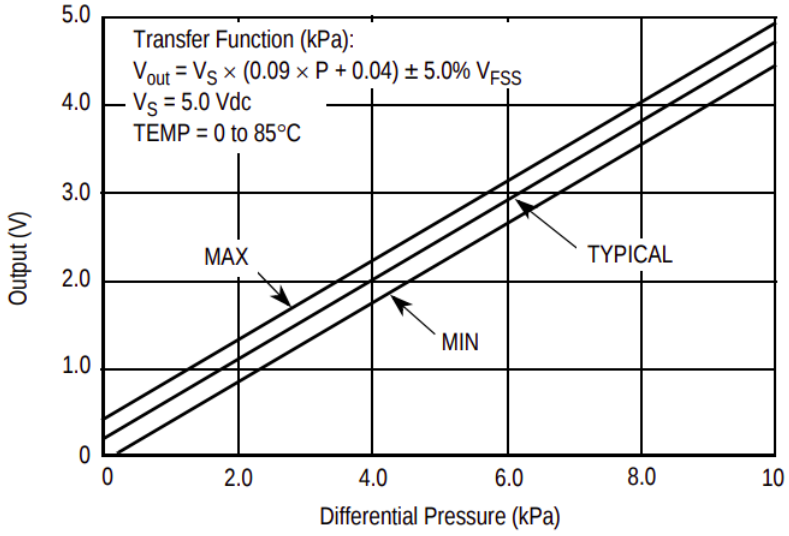


FIGURA 2.6. Función de salida del sensor de presión MPX5010DP[2].

Donde:

- Vs: es el valor de voltaje de alimentación, Vs = 5 V.
- Vout: es el valor de voltaje que entrega el sensor a su salida.
- Tol: es la tolerancia, un ajuste que se debe aplicar al sensor para calibrar a medida.

Tomando como base la ecuación de presión diferencial que relaciona la altura se resuelve que (recordando que a la presión atmosférica normal es cero):

$$P = PH - PL = P = \rho gh = \frac{Vout - 0{,}04 * Vs \pm Tol}{0{,}09 * Vs} \tag{2.2}$$

Donde:

- P: es la presión.
- $\rho$: es la densidad del agua.
- g: es la gravedad.
- h: es el nivel de la columna líquida.

Si:

- la densidad de agua $\rho$ = 1000 $\frac{kg}{m^3}$
- la aceleración de la gravedad g = 10 $ms^2$

la ecuación resultante para obtener h es:

[2]Imagen tomada de https://www.nxp.com/docs/en/data-sheet/MPX5010.pdf

$$h = \frac{P}{g * \rho} \tag{2.3}$$

La disposición de los pines del dispositivo lo podemos advertir en la figura 2.7:

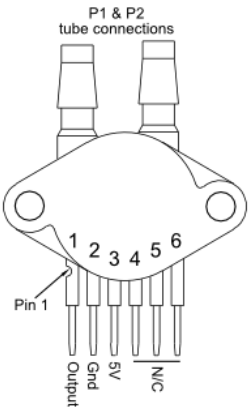


FIGURA 2.7. Disposición de pines del sensor de presión MPX5010DP[3].

## 2.5. Planificación

La planificación original presentada al iniciar el proyecto contemplaba un trabajo de 600 horas. Los atrasos relacionados con la problemática identificada y expuesta en la sección 2.1, significó repensar un nuevo prototipo con una válvula de control mecanizada. Esto implicó recurrir a una empresa dedicada a la mecanización de piezas para adaptar el servomotor y una válvula esférica. Como consecuencia de esto, no se pudo contar en tiempo y forma con la pieza total, por lo que se tuvo que modificar la planificación inicial. En la tabla 2.1 se detalla el desglose de tareas pertinente a la planificación original.

[3]Imagen tomada de https://controlautomaticoeducacion.com/arduino/medidor-de-nivel-de-agua-por-presion-con-arduino/

TABLA 2.1. Desglose de tareas

| WBS | Tareas | Tiempo[hs] |
|---|---|---|
| 1. | Documentación y Análisis Preliminar | 19 |
| 1.1 | Planificación del proyecto | 24 |
| 1.2 | Análisis de bibliografía especifica | 20 |
| 1.3 | Definición de casos de pruebas | 15 |
| 2. | Investigación, Diseño e Implementación | 127 |
| 2.1 | Inicialización del entorno de trabajo | 34 |
| 2.2 | Investigación y diseño del circuito controlador del motor | 12 |
| 2.3 | Investigación y diseño del circuito para el sensor lineal | 31 |
| 2.4 | Definición de la arquitectura del firmware | 54 |
| 2.5 | Definición de las interfaces y API's | 32 |
| 2.6 | Implementación de módulo controlador de motor | 15 |
| 2.7 | Implementación de módulos de adquisición de datos | 12 |
| 2.8 | Implementación de módulo de comunicación | 16 |
| 2.9 | Implementación de herramientas de testing | 18 |
| 2.10 | Integración de módulos | 21 |
| 2.11 | Investigación del sensor de presión diferencial | 15 |
| 2.12 | Calibración del sensor de presión diferencial y diseño del circuito | 13 |
| 3. | Verificación y Validación | 48 |
| 3.1 | Pruebas unitarias en submódulos | 24 |
| 3.2 | Pruebas de integración | 19 |
| 3.3 | Pruebas de Sistema | 23 |
| 4. | Cierre | 36 |
| 4.1 | Elaboración del Informe del proyecto | 24 |
| 4.2 | Elaboración de presentación final del proyecto | 12 |

# Capítulo 3

# Diseño e implementación

En este capítulo se exponen las tomas decisiones referidas al diseño e implementación relacionada al hardware y firmware a lo largo del desarrollo del trabajo.

## 3.1. Hardware

### 3.1.1. Construcción de la válvula de control

En la elaboración del prototipo se creó una caja desmultiplicadora de fuerza con la finalidad de dirigir el recorrido de apertura y cierre de una válvula. Para la generación de estos movimientos se utilizó un servomotor paso a paso.

La válvula es un ensamblaje compuesto de un cuerpo con conexión a una tubería y de un obturador operado por accionamiento. Su función principal es variar el caudal del fluido que circula a través de ella, comportándose como un orificio cuya área está continuamente variando. Generalmente el obturador, conforme se va desplazando, produce un área de pasaje que posee una determinada relación característica entre la fracción de carrera y el caudal correspondiente que se escurre a través de ella. Esta relación recibe el nombre de característica inherente de caudal de válvula.

Particularmente, para este trabajo se utilizó una válvula cuya característica inherente es tipo de apertura rápida. Se trata de una propiedad que produce una variación grande de caudal con una carrera pequeña. Esto posibilita el pasaje de casi la totalidad del caudal nominal con apenas una abertura de 25 % de la carrera total. De esta forma, genera una ganancia muy alta a bajas aperturas de carrera y una ganancia muy baja en aperturas por encima de 60 % de carrera total. La figura 3.1, muestra la curva típica de una válvula de apertura rápida.

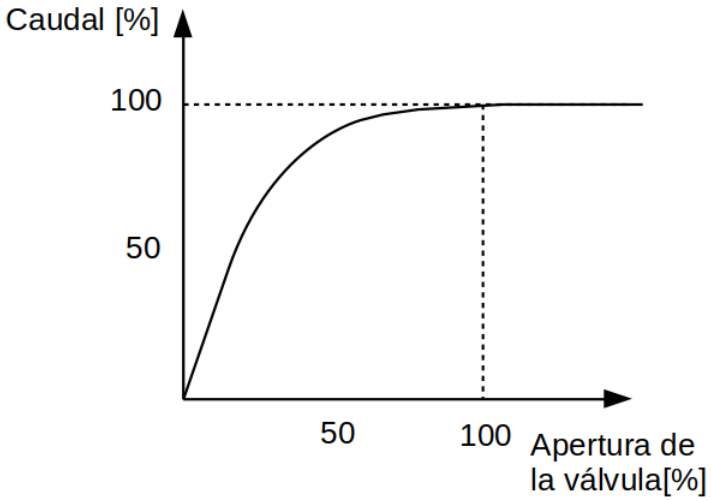


FIGURA 3.1. Gráfica del caudal en función de la apertura de la válvula.

### 3.1.2. Servomotor

Los componentes principales para la construcción del servomotor fueron un motor paso a paso, un controlador, un sensor resistivo de ángulo y una entrada ADC del microcontrolador.

#### Circuito interfaz del conector de señal de control P1

En cuanto a la conexión entre el microcontrolador y el controlador fue necesario la fabricación de un circuito de adaptación de niveles de señales. El MA860H puede aceptar entradas diferenciales y de un solo extremo (incluida la salida de colector abierto y PNP). Tiene 3 entradas lógicas aisladas ópticamente que están ubicadas en el conector P1 para aceptar señales de control del microcontrolador. Estas están aisladas para minimizar o eliminar los ruidos eléctricos acoplados a las señales de control del variador. La figura 3.2, ilustra las conexiones a colector abierto.

Siguiendo las recomendaciones del fabricante relacionadas a la construcción del circuito interfaz entre el driver y el microcontrolador se diseñó y fabricó el circuito eléctrico que se muestra en la figura 3.3.

En el esquemático se observa dos etapas, la primera es un circuito negador y la segunda corresponde a un circuito de colector abierto NPN. Por lo tanto, se puede decir que la etapa de potencia está formada por el circuito interfaz, el controlador y el motor paso a paso.

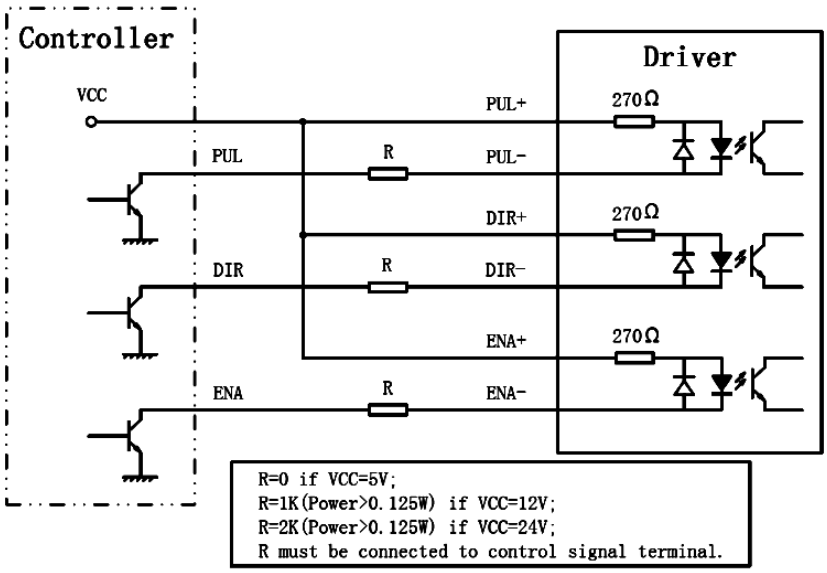


FIGURA 3.2. Circuito interfaz - conexiones de señales a colector abierto.

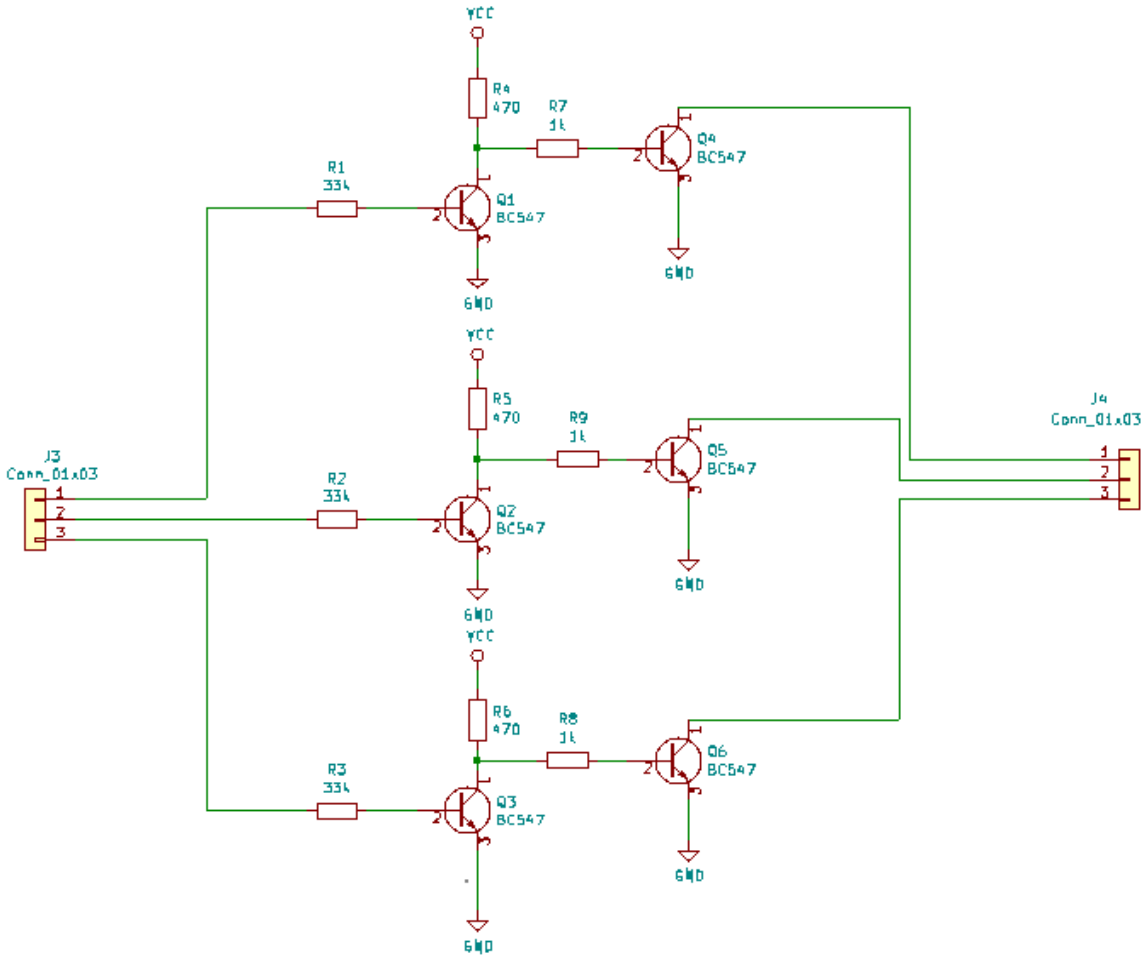


FIGURA 3.3. Esquemático de circuito interfaz entre el microcontrolador y driver.

**Funcionamiento del servomotor**

El servomotor está compuesto también por una fracción del firmware que comprende la parte inteligente del dispositivo fabricado. Asimismo, es el encargado de establecer al obturador de la válvula en una determinada posición y así proporcionar a su salida el caudal de agua requerido por el usuario. Para esto fue indispensable obtener una estimación del porcentaje de apertura de la válvula. Por ello, se empleó un sensor resistivo de ángulo cuya señal se inyecta de forma retroalimentada a una de las entradas ADC del microcontrolador para su procesamiento. Este dispositivo adosado al eje de la válvula es un potenciómetro de una vuelta de 50 kΩ cuyo modelo es 91A-503, Bourns Cermet.

En la figura 3.4, se puede apreciar un diagrama en bloque general que indica entre otras partes del trabajo, cómo está constituido el servomotor. Cada componente se encuentra delimitado en líneas punteadas de forma tal, que se puede tener una idea sobre qué partes pertenecen al servomotor y al microcontrolador.

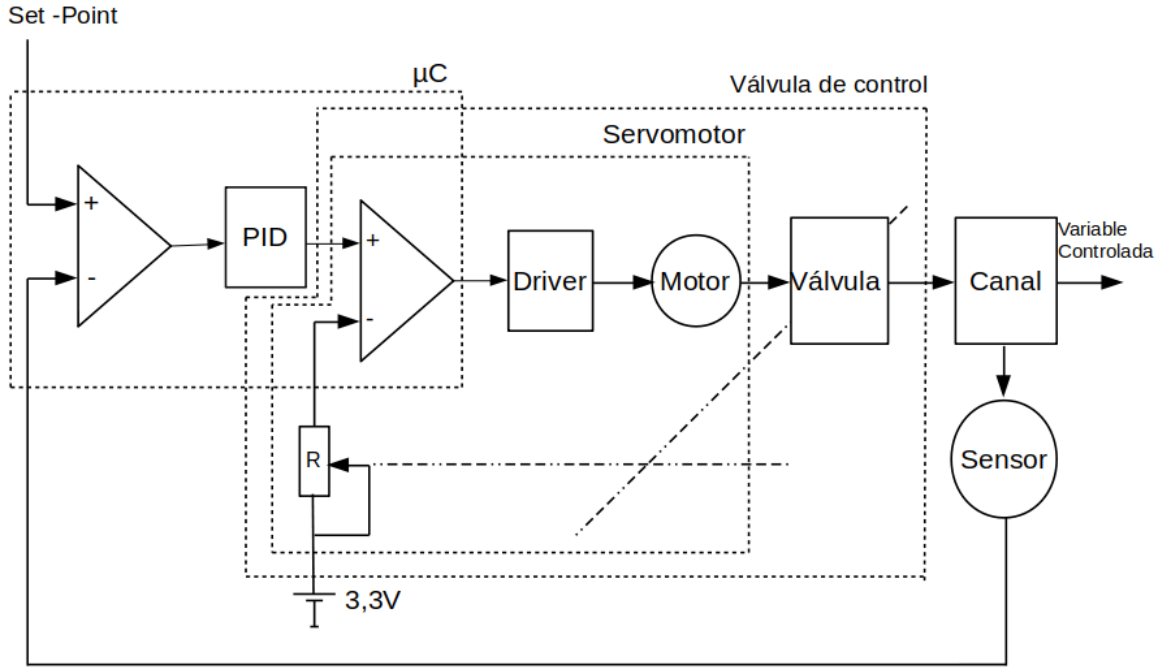


FIGURA 3.4. Diagrama en bloque de la celda primaria.

### 3.1.3. Medidor de caudal

La medición del agua, también llamada aforo, es la cuantificación del caudal de agua que pasa por la sección transversal de un río, canal o tubería. En la mayoría de los casos, la medición del agua resulta de la necesidad de brindar mayor control sobre su uso y distribución. Dicha medición se realiza a través de medidores de caudal. Estos son dispositivos que utilizan diferentes principios mecánicos o físicos para permitir que un flujo de agua pueda ser cuantificado. La herramienta de medición utilizada en este trabajo para verificar el caudal de agua proporcionado, recibe el nombre de medidor de caudal de canal abierto tipo aforo.

Existen varias formas de aforo en canales abiertos, dentro de las principales se encuentran:

1. Método volumétrico.
2. Vertederos.
3. Canal Parshall.
4. Método hidráulico.

Para efectuar el presente trabajo se utilizó el tipo de aforo con vertedero triangular con escotadura en V. La medición del caudal de las corrientes naturales nunca puede ser exacta debido a que el canal suele ser irregular y por lo tanto es irregular la relación entre nivel y caudal. Los canales de corrientes naturales están también sometidos a cambios debidos a erosión o depósitos. Se pueden obtener cálculos más confiables cuando el caudal pasa a través de una sección donde esos problemas se han limitado. Los vertederos pueden ser definidos como simples aberturas, sobre las cuales un líquido fluye. El término se aplica también a obstáculos en el paso de la corriente y a las excedencias de los embalses. Los vertederos son orificios sin el borde superior y ofrecen las siguientes ventajas en la medición del agua:

- Se logra con ellos precisión en los aforos
- La construcción de la estructura es sencilla
- No son obstruidos por materiales que flotan en el agua
- La duración del dispositivo es relativamente larga

Los vertederos son utilizados, intensiva y satisfactoriamente en la medición del caudal de pequeños cursos de agua y conductos libres, así como en el control del flujo en galerías y canales. Para la construcción del vertedero triangular empleado, se precisó de una chapa de hierro plana de 1 mm de espesor, a la que se le realizó una hendidura de sección triangular, cuyo valor de ángulo es de 18 grados. Esta chapa supera los límites del ancho del canal de forma que, ubicándola verticalmente y en forma transversal al canal, opera como un embalse que por su hendidura sale un flujo de agua. De esta forma se puede medir en distintos lugares del recorrido del canal, cuál es el caudal real que está pasando por ese punto en particular. La figura 3.5, muestra la vista de frente de una estructura de un vertedero con escotadura en V.

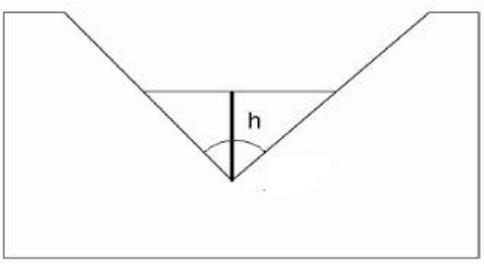


FIGURA 3.5. Placa de aforo: vista de frente.

La figura 3.6, ilustra la vista lateral del vertedero y cómo fluye el líquido por su abertura.

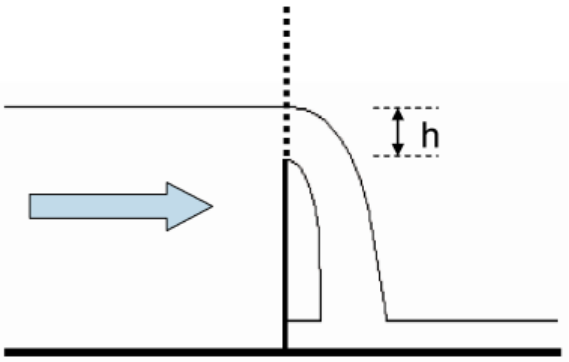


FIGURA 3.6. Placa de aforo: vista lateral.

Entonces, para calcular el caudal se debe tener en cuenta la siguiente fórmula:

$$Q = \frac{8}{15}\sqrt{2g}c\tan\alpha H^{\frac{2}{5}} \tag{3.1}$$

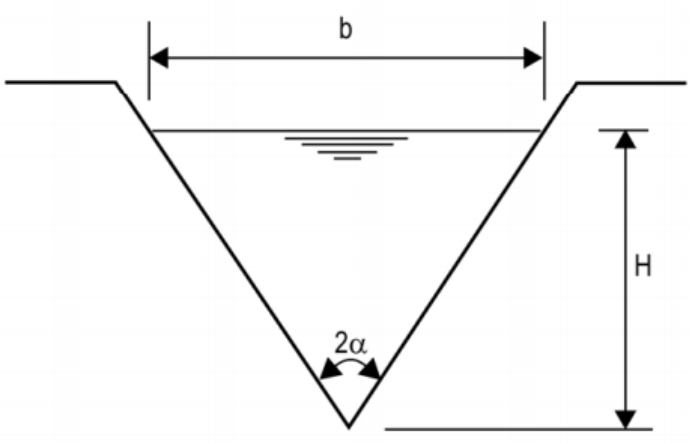


FIGURA 3.7. Placa de aforo: dimensiones.

Teniendo en cuenta la ecuación 3.1

- Q: es caudal en $\frac{m^3}{s}$
- g: coeficiente de gravedad
- c: coeficiente de descarga
- $\alpha$: ángulo del vertedero
- H: nivel de agua en metros

Los medidores de caudal con aforo triangular como el utilizado en el presente trabajo, responden a una ecuación del modelo que se expone arriba obtenida empíricamente.El medidor utilizado en este proyecto se usó por única vez, por lo tanto la tabla de calibración del caudalímetro fue obtenida experimentalmente. En la figura 3.8, se detalla la vista de frente del canal con el vertedero triangular y los diversos niveles de altura que se consideraron al momento de realizar las mediciones pertinentes.

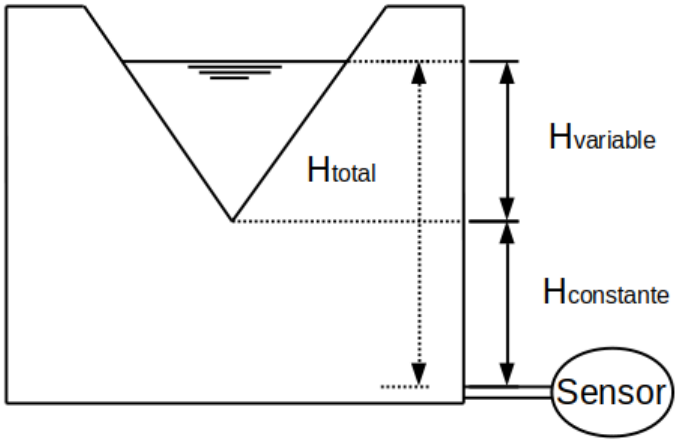


FIGURA 3.8. Placa de aforo: dimensiones.

A partir de la figura 3.8 se simboliza una manguera, conectada por un extremo a la base del canal y, en el otro, a una de las dos espigas del sensor. Además, se puede advertir los niveles de altura que se deben tener en cuenta. $H_{total}$ corresponde a la altura entre el nivel de agua en un instante dado y la manguera conectada al sensor, $H_{constante}$ se encuentra asociada al nivel de altura que está comprendido entre dicha manguera y el vértice del triángulo del vertedero, y finalmente $H_{variable}$ es la altura que está contenida entre el nivel de agua y el vértice del triángulo de dicho vertedero. Por lo tanto, este caudalímetro de canal abierto por sistema de aforo utiliza como medidor de nivel de columna de agua un sensor de presión cuyo modelo es MPX5010DP.

Esta señal proveniente del sensor en conjunto con un algoritmo de control PID, permite regular el caudal de agua en un determinado valor preestablecido.

Una vez construido el prototipo, se realizó una calibración de caudal obteniéndose una tabla de Caudal vs $H_{total}$ (altura). Esta es utilizada en el firmware para obtener el valor de caudal en lugar de la fórmula presentada.

## 3.2. Software

### 3.2.1. Arquitectura de software

Durante la definición de la arquitectura de software se seleccionaron dos y se las integraron para confeccionar una híbrida, de modo tal que se adapte a las necesidades concretas de este proyecto. Por un lado, se seleccionó una arquitectura que está conformada por tres capas claramente definidas. En la capa HAL se encuentran los drivers encargados de abstraer los detalles de acceso al hardware. De esta forma, facilitará la migración a otro microcontrolador en caso de que en el futuro hiciera falta. Adicionalmente, permite una mayor claridad en la implementación, al separar los drivers de la lógica de la aplicación. Debido a que en la producción se utilizará como hardware a la CIAA-NXP y durante el desarrollo

a la EDU-CIAA-NXP, se empleó el firmware de la CIAA versión 3.0 como capa de abstracción de hardware. En la capa de aplicación se encuentran todos los componentes que encapsulan la lógica de aplicación, y finalmente, la tercera corresponde al sistema operativo de tiempo real FreeRTOS [8]. El sistema incluye sensores que proveen información relacionada al ámbito a controlar y un actuador para operar sobre el sistema. Por lo tanto, en respuesta a las alteraciones identificadas por los sensores, se envían señales de control a los actuadores. Entonces, al identificar que el sistema debe poseer este tipo de comportamiento se definió emplear una arquitectura de tiempo real inherente al "Control Ambiental" [9] , ya que este patrón de arquitectura de software brinda la posibilidad de recopilar datos del entorno por medio de sensores como así también el estado en el que se encuentran los actuadores conectados al sistema. Con base en datos reunidos de sensores y actuador, se envían señales de control hacia el actuador para producir cambios en el entorno controlado. En la figura 3.9, se puede ver el diagrama del patrón arquitectónico que es la base del diseño del sistema de control, y además utilizado en la capa aplicación.

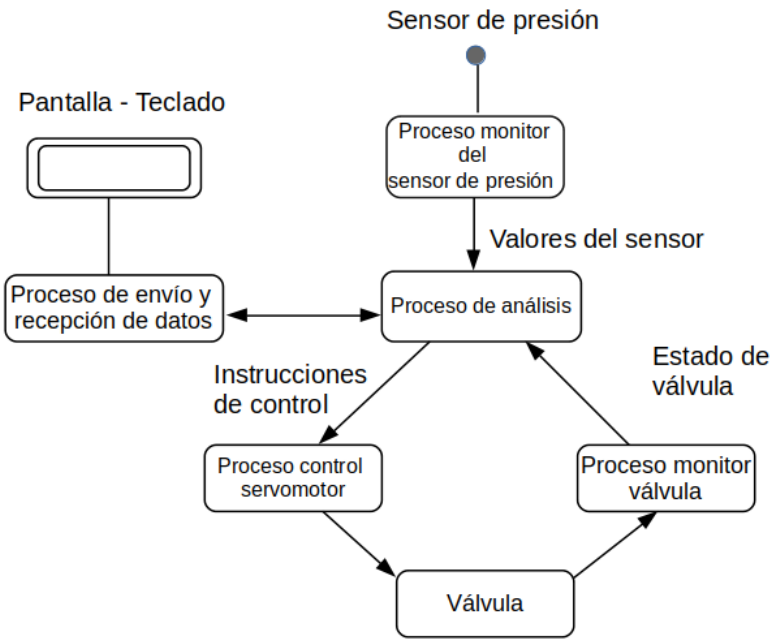


FIGURA 3.9. Arquitectura de sistema de control de caudal de agua en canal a cielo abierto.

Al aplicar ambos patrones se constituyó un patrón híbrido de tres capas. La capa inferior es la capa de abstracción de hardware. La capa superior, pertenece a la aplicación que incluye los componentes de un patrón control ambiental.

### 3.2.2. Componentes de software

Cada capa de software es considerada un componente de software. Con lo cual se poseen los siguientes componentes:

- HAL
- Sistema Operativo

- Aplicación

A su vez, la capa de aplicación está compuesta por los siguientes componentes de software:

- Monitor sensor de presión.
- Control servomotor.
- Monitor válvula.
- Control de recepción y envío de datos.

En la figura 3.10, se puede apreciar el nivel de jerarquía de cada una de las capas donde la capa HAL la más baja:

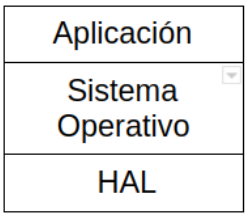


FIGURA 3.10. Jerarquía de componentes de software.

### Diseño detallado

En esta subsección se incluye el diseño detallado de cada uno de los componentes de software presentados en la figura 3.9, todos ellos con sus respectivos diagramas de actividades [10].

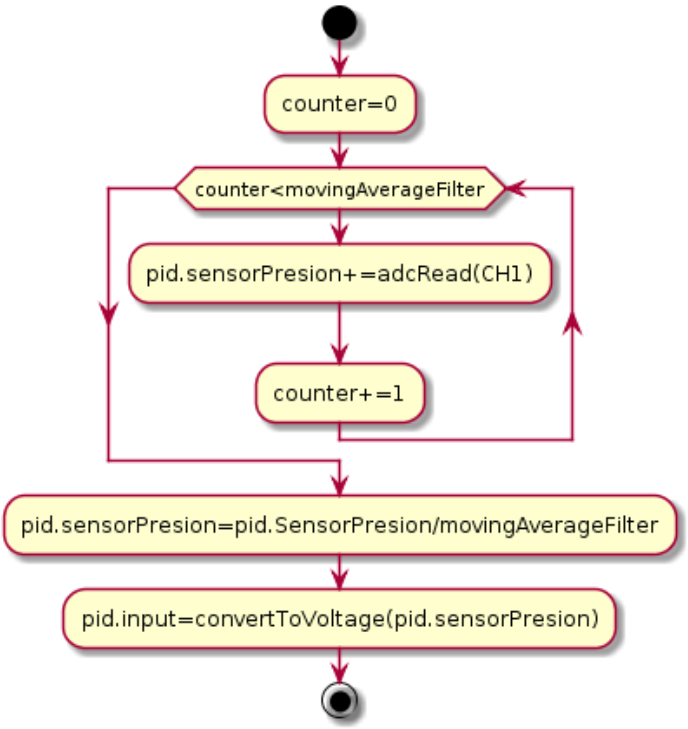


FIGURA 3.11. Proceso monitor del sensor de presión.

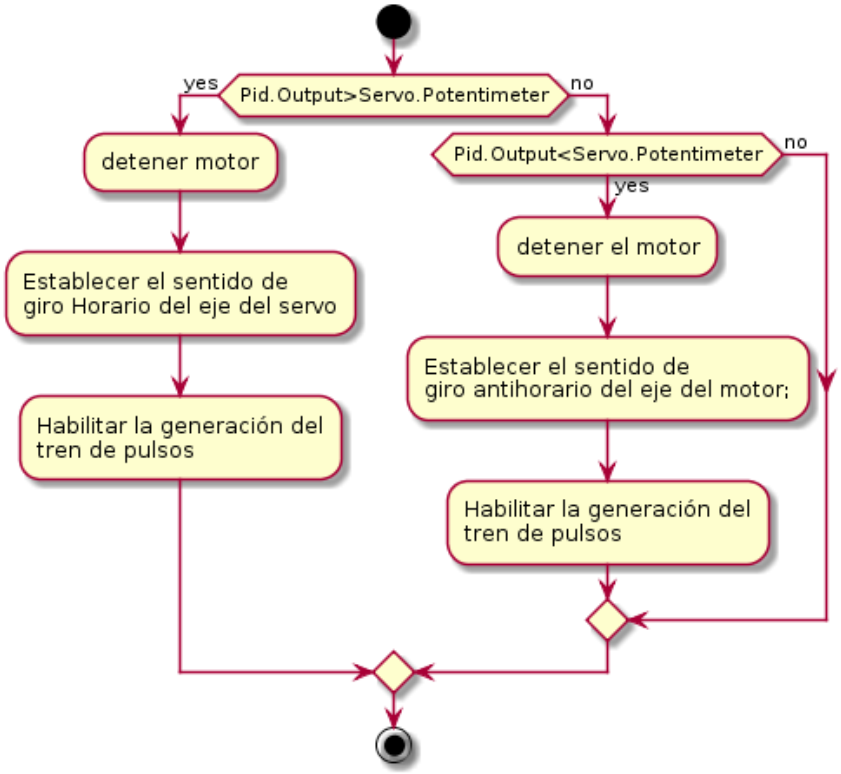


FIGURA 3.12. Proceso de control del servomotor.

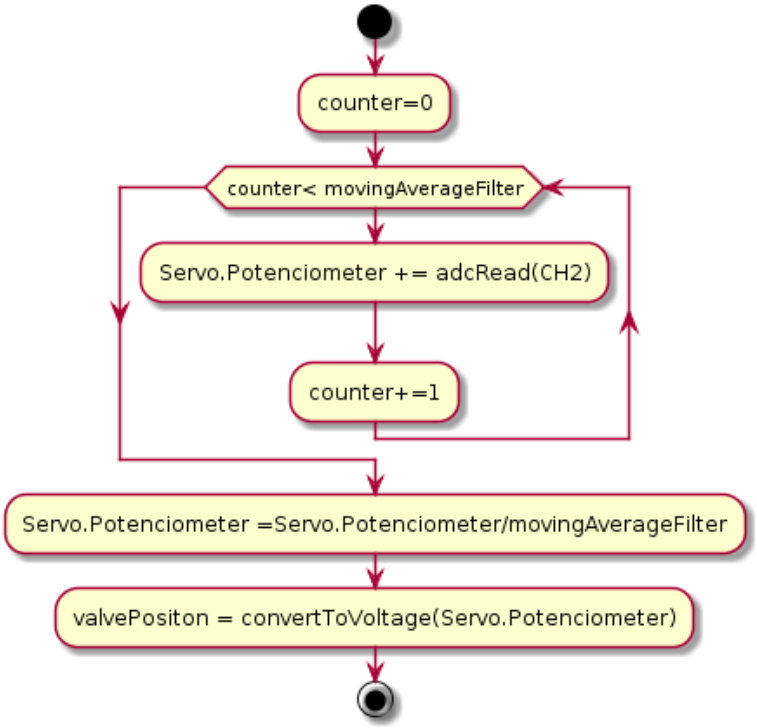


FIGURA 3.13. Proceso monitor válvula.

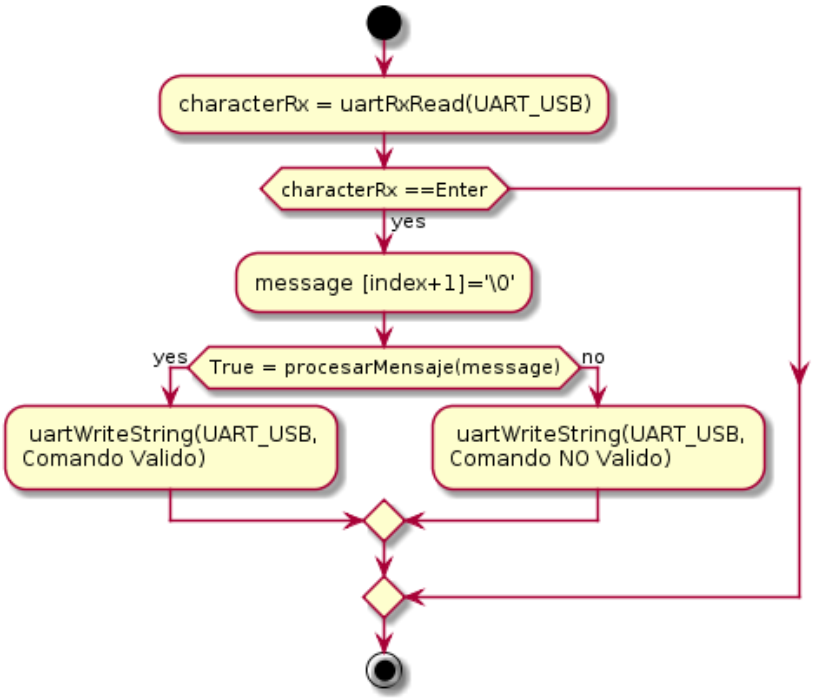


FIGURA 3.14. Proceso de envío y recepción de datos.

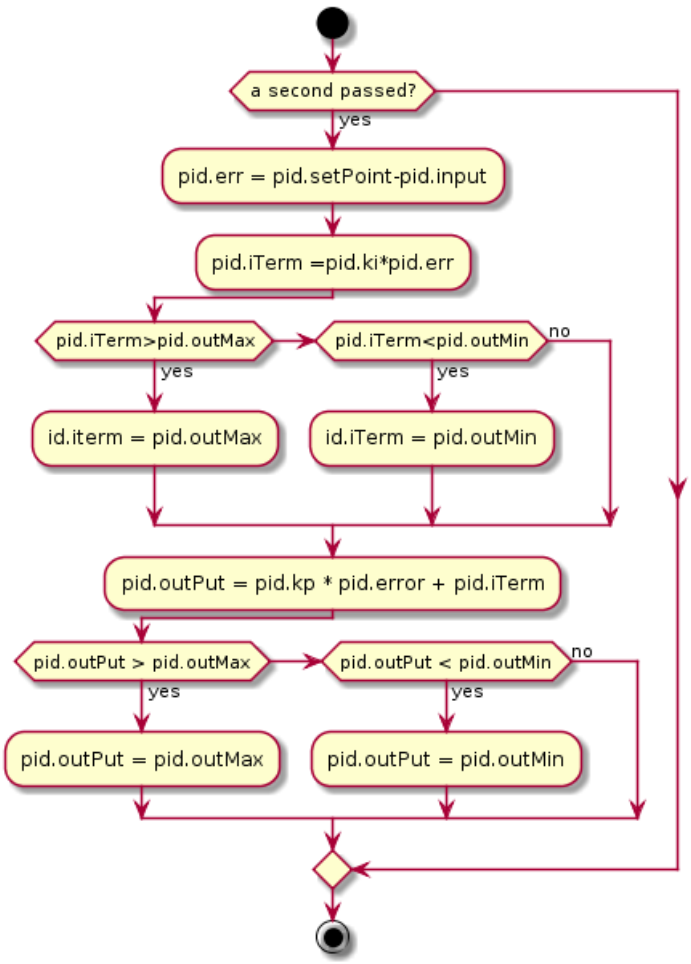


FIGURA 3.15. Proceso de control PID.

### 3.2.3. Casos de uso

En la etapa de captura de requerimientos funcionales que el sistema debía satisfacer, se emplearon casos de uso correspondientes al Lenguaje Unificado Modelado (UML) [11]. Esta técnica es de mucha utilidad y permite especificar, visualizar, construir y documentar los requisitos del sistema especialmente en sistemas con un alto grado de interacción hombre/máquina [12]. En las tablas 3.1 y 3.2 se presentan dos casos de uso que se documentaron como parte integral de la especificación de requisitos del software del equipo.

TABLA 3.1. Caso de uso: detectar el nivel de agua del canal.

| | |
|---|---|
| Nombre | Detectar el nivel de agua del canal |
| 1.1 Breve descripción | El firmware debe detectar por medio del sensor de presión el nivel de agua existente en el canal. |
| 1.2 Actor Principal | La aplicación móvil. |
| 1.3 Disparadores | La recepción de un comando. |
| Flujo de Eventos | |
| 2.1 Flujo básico | El firmware recibe el comando desde un smartphone o desde una aplicación de comunicación serial desde una pc. |
| 2.1.2 Flujo básico | El firmware deberá desencapsular el comando recibido y extraer los diferentes campos de información. |
| 2.1.3 Flujo básico | El firmware deberá identificar el tipo de operación. |
| 2.1.4 Flujo básico | El firmware deberá leer un pin configurado como adc asociado al sensor de presión un valor digital. |
| 2.1.5 Flujo básico | Se repite el paso 2.1.4 hasta 5 veces, de forma que el firmware deberá realizar una acumulación de los valores obtenidos. |
| 2.1.6 Flujo básico | El firmware con base al acumulador obtenido en el punto anterior deberá, realizar un filtro de promedio móvil. |
| 2.1.7 Flujo básico | El firmware deberá convertir el valor digital promediado a voltaje. |
| 2.1.8 Flujo básico | El firmware deberá obtener el nivel de agua con el dato obtenido en el punto anterior empleando una tabla que se obtendrá experimentalmente. |
| 2.1.9 Flujo básico | El firmware deberá controlar si el resultado se encuentra dentro de las dimensiones establecidas. |
| 2.1.10 Flujo básico | El firmware deberá enviar por el puerto serial el valor de nivel de agua en el orden de los centímetros. |
| 2.2 Flujo alternativo | |
| 2.2.1 Flujo alternativo | 2.1.9 Si la distancia obtenida se encuentra fuera de rango se enviará por la UART “Distancia Fuera de Rango - Verificar el correcto Funcionamiento del sensor”. |
| Requisitos Especiales | |
| Pre - condiciones | |
| 4.1 Pre - condiciones | El firmware debe estar en estado de correcto funcionamiento. |
| 4.2 Pre - condiciones | La comunicación serial se debe encontrar en condiciones óptimas para su funcionamiento correcto. |
| Post- Condiciones | El firmware deberá quedar operativo y en correcto funcionamiento y en condiciones para la recepción de futuros comandos. |

TABLA 3.2. Caso de uso: establecer un determinado valor de caudal.

| | |
|---|---|
| Nombre | Establecer un determinado de valor de caudal. |
| 1.1 Breve descripción | El firmware debe establecer un caudal mediante la manipulación de la posición de la válvula de control. |
| 1.2 Actor Principal | La aplicación móvil. |
| 1.3 Disparadores | La recepción de un comando |
| Flujo de Eventos | |
| 2.1 Flujo básico | El firmware recibe el comando desde un smartphone o desde una aplicación de comunicación serial desde una pc. |
| 2.1.2 Flujo básico | El firmware deberá desencapsular el comando recibido y extraer los diferentes campos de información. |
| 2.1.3 Flujo básico | El firmware deberá interpretar el tipo de operación. |
| 2.1.4 Flujo básico | El firmware deberá leer un pin configurado como adc asociado al sensor de presión un valor digital. |
| 2.1.5 Flujo básico | Se repite el paso 2.1.4 hasta 5 veces, de forma que el firmware deberá realizar una acumulación de los valores obtenidos. |
| 2.1.6 Flujo básico | El firmware con base al acumulador obtenido en el punto anterior deberá, realizar un filtro de promedio móvil. |
| 2.1.7 Flujo básico | El firmware deberá convertir el valor digital promediado a voltaje. |
| 2.1.8 Flujo básico | El firmware deberá obtener el caudal con el dato obtenido en el punto anterior empleando una tabla que se obtendrá experimentalmente. |
| 2.1.9 Flujo básico | El firmware deberá comparar este resultado de caudal obtenido con el set point. |
| 2.1.10 Flujo básico | En caso de ser valores diferentes, el firmware deberá generar una señal tren de pulsos en el sentido correspondiente hasta posicionar la válvula de control. |
| 2.1.11 Flujo básico | Se repite desde 2.1.4 hasta que el caudal obtenido a través del sensor y el set point sean iguales. |
| 2.2 Flujo alternativo | |
| 2.2.1 Flujo alternativo | 2.1.10 Si los valores son iguales el firmware deberá detener la generación del tren de pulsos. |
| Requisitos Especiales | |
| Pre - condiciones | |
| 4.1 Pre - condiciones | El firmware debe estar en estado de correcto funcionamiento. |
| 4.2 Pre - condiciones | La comunicación serial se debe encontrar en condiciones óptimas para su funcionamiento correcto. |
| Post- Condiciones | El firmware deberá quedar operativo y en correcto funcionamiento y en condiciones para la recepción de futuros comandos. |

### 3.2.4. Diagrama de clases del firmware

Previo al desarrollo del firmware, se realizó un diagrama de clases [13], que se puede apreciar en la figura 3.16.

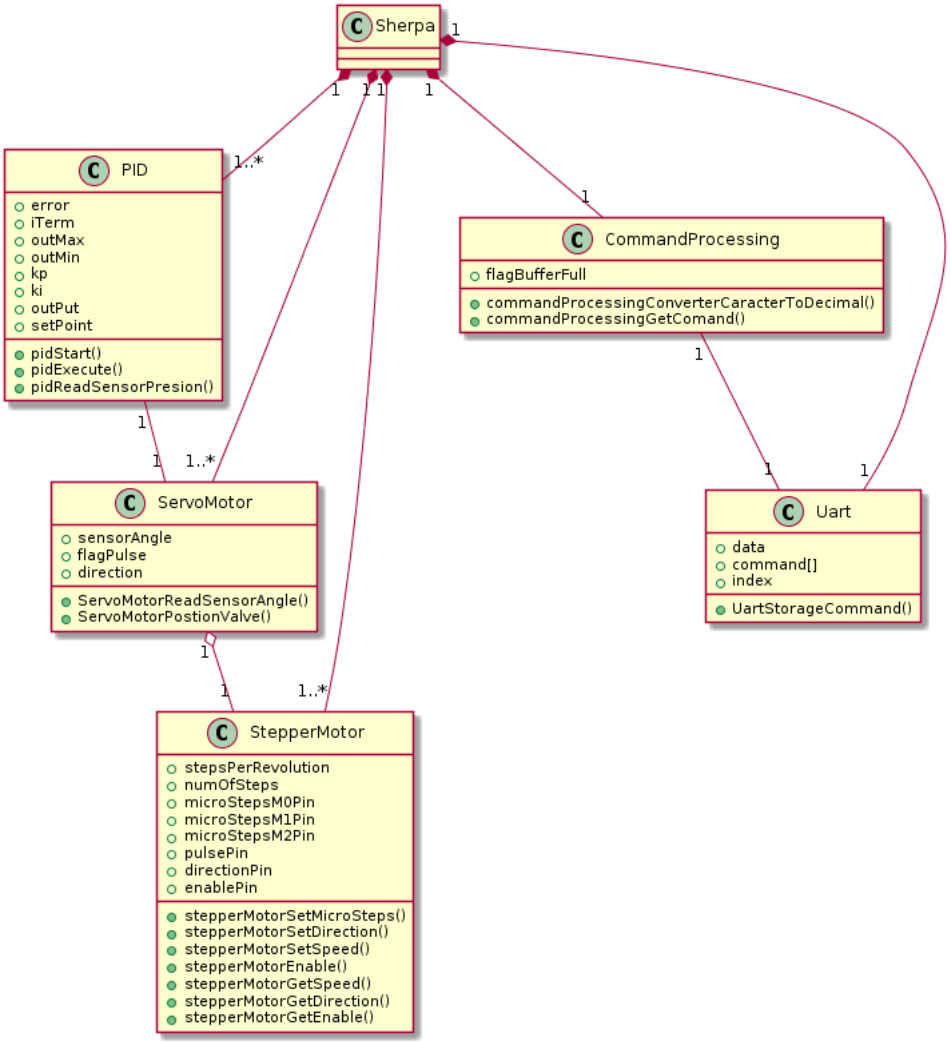


FIGURA 3.16. Diagrama de clases.

- Uart: este módulo recibe y envía los comandos desde y hacia la aplicación de comunicación serial que se ejecuta en la PC.
- CommandProcessing: acumula y convierte de carácter a decimal los comandos recibidos y los devuelve cuando es solicitado.
- StepperMotor: este módulo posee todas las funciones inherentes al control del motor paso a paso. Interactúa con el módulo Servomotor para integrar las características de un servomotor convencional.
- ServoMotor: este módulo tiene la misión de establecer en cierta posición al obturador a través de la generación de una señal de tren de pulsos. Esto se realiza mediante una comparación entre la salida del algoritmo de control PID y la señal del sensor resistivo de ángulo. Cuando la comparación resulta igual a cero implica que alcanzó situar el obturador en la posición deseada.

- PID: este módulo implementa un algoritmo de control PID que se ejecuta cada un segundo para cuantificar el error o desviación que existe entre el valor de caudal medido a través del sensor de presión y el valor deseado.
- Sherpa: es la clase central, que al momento de la ejecución esta compuesta por al menos una o más instancias de las clases *PID* y *StepperMotor* y, además por solo una instancia de *commandProseccing*.

### 3.2.5. Protocolo de comunicación

Se definió un paquete de datos para que una aplicación externa y el firmware puedan interaccionar de modo tal que permita verificar el correcto funcionamiento del sistema. A continuación se presentan las principales funciones que debe ejecutar el motor paso a paso:

1. Habilitar o deshabilitar el motor.
2. Establecer los micro pasos (MicroStep).
3. Establecer el sentido de giro (Horario/Antihorario).
4. Cantidad de pasos a realizar.
5. El ángulo a girar.
6. Cantidad de vueltas a girar.

Luego que se reconoció el funcionamiento que debe desempeñar el servomotor, se procedió a especificar los comandos admitidos para la comunicación entre el firmware y la aplicación externa. En la tabla 3.3 se presenta cada uno de los comandos con su respectiva explicación.

TABLA 3.3. Descripción de comandos.

| Comandos | Descripción |
|---|---|
| ME<br>MD | Deshabilita el motor.<br>Habilita el motor. |
| MMSF<br>MMSH<br>MMS08<br>MMS16<br>MMS32 | Establece los micropasos. |
| MTA<br>MTH | Establece el sentido de giro del eje del motor horario y antihorario. |
| MS1201 | Establece la cantidad de pasos. Cuatro dígitos para especificar el número de pasos, 1201. |
| MA0360 | Establece el ángulo de giro del eje del motor. Cuatro dígitos para especificar el ángulo en grados, 360. |
| MFT0160 | Establece la cantidad de vueltas completas. Cuatro dígitos para especificar el número de vueltas, 160. |
| SP100 | Establece el valor del set point en porcentajes. Tres dígitos para especificar el número de vueltas, 160. |
| NA | Devuelve el valor de altura del nivel del agua. |

Al final de cada una de las tramas, se incorpora el carácter de salto de línea que indica el límite final del comando. Una vez recibido, el sistema lo procesa, de forma que si presenta un error, el firmware envía por el mismo medio una cadena de notificación "comando no válido", caso contrario "comando válido".

### 3.2.6. Algoritmo PID

Durante el diseño del proyecto se identificó la posibilidad de implementar un control de proceso con realimentación. La tarea particular del sistema de control, es la de determinar y actualizar la posición de la válvula a medida que cambian las condiciones de carga. De esta manera, la variable controlada (caudal) alcanzará el valor deseado y permanecerá en un entorno de este, aún produciéndose perturbaciones externas. Con base en las necesidades expuestas se definió incorporar al firmware un algoritmo de control PID [14] como una herramienta tecnológica que permite cuantificar el error o desviación que existe entre un valor medido y un valor deseado de caudal.

$$Salida = \frac{100}{BP}[e_0 + \frac{1}{I_t}\int e_0 dt - D_t\frac{dm}{dt}] \tag{3.2}$$

Donde:

- $\frac{100}{BP}$: ganancia del sistema ( %)
- $e_0$: error o diferencia entre el set point y el valor medido
- $\frac{1}{I_t}$: tiempo de integración
- $D_t$: tiempo de derivación

El mismo está basado en el algoritmo publicado por Brett Beauregard [15] y se modificó para satisfacer las necesidades concretas del presente proyecto. Es relevante destacar que se seleccionó un modo de control que consiste en una combinación de la acción integral con la acción proporcional, y eliminar la acción derivativa. Por lo que la ecuación 3.2 queda de la siguiente forma:

$$Salida = \frac{100}{BP}[e_0 + \frac{1}{I_t}\int e_0 dt] \tag{3.3}$$

Cuando el objeto control es una válvula la acción derivativa no se utiliza, dado que, su propiedad es la de actuar en porcentajes elevados de salida frente a variaciones muy pequeñas o rápidas de error. Es la parte del algoritmo que en el desarrollo teórico, se añadió al final de todo, para acelerar la salida. Por la naturaleza matemática del algoritmo, la acción de control de la parte derivativa es muy alta en porcentaje, cuando la velocidad del error es elevada, aunque en magnitud sea pequeña. Esto hace que ante cualquier variación rápida del error, la válvula reaccione fuertemente, abriéndose o cerrándose. Estos movimientos repetitivos y violentos, lógicamente pueden provocar dañosa esta pieza. Es por esto que la acción derivativa no se utiliza cuando el actuador del sistema de control es una válvula. Entonces, ante una rápida variación (por ejemplo ruido electromagnético) se establece un porcentaje de salida muy grande.

### 3.2.7. Descripción general del firmware

El firmware emplea un algoritmo de control PI con el objeto de corregir el error que existe entre el set point y el valor de caudal medido. El resultado de procesar la señal realimentada del sensor de presión contra el set point preestablecido por el usuario, es la salida o acción de control. Luego, este valor de salida obtenido en voltaje, se compara con la señal del sensor resistivo. De esta forma, se habilita y deshabilita la señal de tren de pulsos que son enviadas al controlador con el fin de girar el eje del motor paso a paso en un sentido y otro. Cuando ambas señales se anulan ante un segundo comparador significa que se estableció la posición del obturador para suministrar el caudal deseado. En esta única ocurrencia se inhabilita la señal de tren de pulsos generada por el microcontrolador. En caso que la salida del algoritmo PI sea mayor que la señal del sensor resistivo, se habilita el giro del eje en sentido horario. En caso contrario se habilita el giro en sentido antihorario. Por lo tanto, la señal proveniente del sensor resistivo de ángulo que también es retroalimentada, como se puede apreciar en la figura 3.4, brinda la posición actual del obturador a medida que este varíe. Entonces, los tres fundamentales casos a tener en cuenta son:

- Si la salida del algoritmo de control PI es igual a la señal del sensor resistivo, se inhabilita la señal de tren de pulsos.
- Si la salida del algoritmo de control PI es mayor que la señal del sensor resistivo, se habilita la señal de tren de pulsos y el giro del eje del servomotor en sentido de horario.
- Si la salida del algoritmo de control PI es menor que la señal del sensor resistivo, se habilita la señal de tren de pulsos y el giro del eje del servomotor en sentido antihorario.

En cuanto al modelado del firmware se utilizó una metodología orientada a objetos. El inconveniente que se presenta al aplicar este paradigma de programación, es que el lenguaje C, el más utilizado en el desarrollo de sistemas embebidos, no es un lenguaje orientado a objetos. Como solución se utilizó una técnica que permite explotar el uso de punteros a estructuras, y emular muy fácilmente una programación orientada a objetos, utilizando el lenguaje C estándar[16]. Esta técnica considera un archivo de cabeceras (.h) como la interfaz pública de la clase y el archivo de código fuente (.c) como la implementación privada. La ventaja que ofrece implementar esta técnica es que permite generar un código que puede escalar muy fácilmente. En las siguientes porciones de código se puede ver de forma simplificada esta técnica de programación descrita.

```
typedef struct{
    uint32_t stepsPerRevolution;
    uint16_t lastNumberOfSteps;
    float LastAnglePosition;

    gpioMap_t pulsePin;
    gpioMap_t directionPin;
    gpioMap_t enablePin;

    gpioMap_t microStepsM0Pin;
    gpioMap_t microStepsM1Pin;
    gpioMap_t microStepsM2Pin;

    stepperMotorDirection_t direction;

    stepperMotorEnable_t isEnable;
    stepperMotorMove_t isMoveAxis;

    bool_t flagPulse;

    float rpm;
    float stepAngle;


}stepperMotor_t;
stepperMotor_t  stepper;

// esta funcion inicializa la estructura del motor paso a paso
void stepperMotorInit(stepperMotor_t *stepper, uint32_t
    stepsPerRevolution,
    gpioMap_t pulsePin, gpioMap_t directionPin, gpioMap_t enablePin,
    gpioMap_t microStepsM0Pin, gpioMap_t microStepsM1Pin,
    gpioMap_t microStepsM2Pin, float stepAngle);

//esta funcion  establece la velocidad del eje del motro PaP
void stepperMotorSetSpeed(stepperMotor_t * stepper, float rpm);

```

```
37 //esta funcion permite leer la velocidad del eje del motor PaP
38 float stepperMotorGetSpeed(stepperMotor_t * stepper);
39
40 //esta funcion establece la resolucion de los microspasos
41 void stepperMotorSetMicroSteps( bool_t m0MicroStep,
42     bool_t m1MicroStep, bool_t m2MicroStep);
43
44 //esta funcion permite leer el microsteps que tiene configurado el motor
     .
45 stepperMotorMicroSteps_t stepperMotorGetMicroSteps(stepperMotor_t *
     stepper);
46
47 // esta funcion permite establecer el sentido de giro del motor PaP
48 void stepperMotorSetDirection(stepperMotor_t * stepper,
49     stepperMotorDirection_t dir);
50
51 //esta funcion permite leer el sentido de giro del eje del motor PaP
52 stepperMotorDirection_t stepperMotorGetDirection(stepperMotor_t *
     stepper);
53
54 //esta funcion permite Habilitar el motor PaP
55 void stepperMotorEnable(stepperMotor_t * stepper, stepperMotorEnable_t
     enable);
56
57 //esta funcion permite leer si el motor PaP esta habilitado
58 stepperMotorEnable_t stepperMotorGetEnable(stepperMotor_t * stepper);
59
60 //esta funcion permite mover el eje del motor una cierta cantidad de
     pasos
61 void stepperMotorMoveSteps(stepperMotor_t * stepper, uint32_t
     numberOfSteps);
62
63 //esta funcion permite mover el eje del motor PaP una cierta cantidad
     vueltas completas
64 void stepperMotorMoveTurns(stepperMotor_t * stepper, uint32_t
     numberOfTurns);
65
66 //Esta funcion permite girar el eje del motor un determinado angulo
67 void stepperMotorMoveAngle(stepperMotor_t * stepper, float angle);
68
69 //Esta funcion detiene el movimiento del eje del motor
70 void stepperMotorStopMoveSteps(stepperMotor_t * stepper);
71
72 // Esta funcion permite establecer el angulo o la ultima cantidad de
     pulsos que se proporciono al Motor PaP
73 void stepperMotorSetAngle(stepperMotor_t* stepper,
     stepperMotorDirection_t dir );
74
75 // Esta funcion retorna el angulo del eje del motor paso a paso
76 float stepperMotorGetAngle(stepperMotor_t* stepper);
```

CÓDIGO 3.1. Porción de código del archivo de cabecera con la interfaz pública de la clase StepperMotor.

```
1 void stepperMotorInit(stepperMotor_t *stepper, uint32_t
     stepsPerRevolution,
2 /*     GPIO2        GPIO1    ,            GPIO0    ,            GPIO3 */
3 gpioMap_t pulsePin, gpioMap_t directionPin, gpioMap_t enablePin,
4     gpioMap_t microStepsM0Pin,
5     /*                  GPIO4          ,     GPIO5        */
6     gpioMap_t microStepsM1Pin, gpioMap_t microStepsM2Pin, float
     stepAngle) {
7
```

```
8   stepper->stepsPerRevolution = stepsPerRevolution;
9   stepper->pulsePin = pulsePin;
10  stepper->directionPin = directionPin;
11  stepper->enablePin = enablePin;
12  stepper->microStepsM0Pin = microStepsM0Pin;
13  stepper->microStepsM1Pin = microStepsM1Pin;
14  stepper->microStepsM2Pin = microStepsM2Pin;
15  stepper->stepAngle = stepAngle;
16
17  // configuro el sentidos de los GPIO'S
18  gpioConfig(stepper->pulsePin, GPIO_OUTPUT);
19  gpioConfig(stepper->directionPin, GPIO_OUTPUT);
20  gpioConfig(stepper->enablePin, GPIO_OUTPUT);
21  gpioConfig(stepper->microStepsM0Pin, GPIO_OUTPUT);
22  gpioConfig(stepper->microStepsM1Pin, GPIO_OUTPUT);
23  gpioConfig(stepper->microStepsM2Pin, GPIO_OUTPUT);
24 }
25
26 void stepperMotorSetSpeed(stepperMotor_t * stepper, float rpm) {
27  stepper->rpm = rpm;
28 }
29
30 float stepperMotorGetSpeed(stepperMotor_t * stepper) {
31  return stepper->rpm;
32 }
33
34 void stepperMotorSetDirection(stepperMotor_t * stepper,
      stepperMotorDirection_t dir) {
35
36  switch (dir) {
37  case STEPPER_RIGHT_OPEN:
38    gpioWrite(stepper->directionPin, TRUE);
39    stepper->direction = dir;
40    break;
41  case STEPPER_LEFT_CLOSE:
42    gpioWrite(stepper->directionPin, FALSE);
43    stepper->direction = dir;
44    break;
45  }
46 }
47
48 stepperMotorDirection_t stepperMotorGetDirection(stepperMotor_t *stepper
      ) {
49  return stepper->direction;
50 }
51
52 void stepperMotorSetMicroSteps(stepperMotor_t *stepper, bool_t
      m0MicroStep, bool_t m1MicroStep, bool_t m2MicroStep) {
53
54  if (stepCount == 0) {
55    gpioWrite(stepper->microStepsM0Pin, m0MicroStep);
56    gpioWrite(stepper->microStepsM1Pin, m1MicroStep);
57    gpioWrite(stepper->microStepsM2Pin, m2MicroStep);
58  }
59 }
60
61 void stepperMotorEnable(stepperMotor_t *stepper, stepperMotorEnable_t
      enable) {
62  switch (enable) {
63  case STEPPER_ENABLE:
64    gpioWrite(stepper->enablePin, TRUE); //activo en bajo, en el
      hardware implemente una compuerta NOT
65    stepper->isEnable = enable;
```

```
66      break;
67    case STEPPER_DISABLE:
68      gpioWrite(stepper->enablePin, FALSE);
69      stepper->isEnable = enable;
70    }
71  }
72
73  stepperMotorEnable_t stepperMotorGetEnable(stepperMotor_t *stepper) {
74    return stepper->isEnable;
75  }
76
77  void stepperMotorMoveSteps(stepperMotor_t *stepper, uint32_t
        numberOfSteps) {
78      if (stepper->isEnable == STEPPER_ENABLE) {
79
80      stepper->lastNumberOfSteps = numberOfSteps;
81
82      printf("cantidad de pasos recibidos:%d\n", stepper->
        lastNumberOfSteps);
83
84    } else {
85      if (stepper->isEnable == STEPPER_DISABLE)
86        printf("Motor Inhabilitado. \n");
87    }
88  }
```

CÓDIGO 3.2. Porción simplificada del archivo de código fuente con algunas funciones de la implementación de la clase StepperMotor.

# Capítulo 4

# Ensayos y resultados

En este capítulo se detallan las pruebas realizadas durante el desarrollo del trabajo y finalmente se analizan los resultados obtenidos.

## 4.1. Ensayos de funcionamiento y calibración del sensor de presión

Se llevó a cabo pruebas acerca del correcto funcionamiento y calibración del sensor de presión MPX5010DP. Para esto, como primera instancia se utilizó un osciloscopio como instrumento de medición de la señal derivada del sensor, una fuente de alimentación de 5 V para la energización del dispositivo y finalmente una manguera flexible de cristal cuyo diámetro externo es de 5 mm, 3 mm de diámetro interno y 300 mm de largo. Un extremo se conecta a una de las espigas del sensor y el otro se utiliza para introducir y evacuar agua mediante el uso de una jeringa. En la figura 4.1 se puede apreciar la conexión del dispositivo trasnductor con la manguera flexible y con la herramienta de medición.

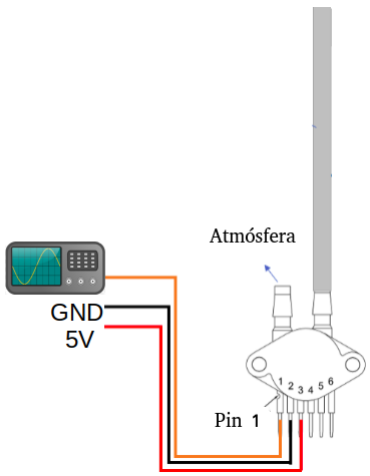


FIGURA 4.1. Herramientas utilizadas para la verificación del funcionamiento del sensor y obtención de una función patrón.

La finalidad de este ensayo fue la adquisición datos para reconocer que el sensor presente un comportamiento lineal, Voltaje vs. Presión kPa, como lo especifica su respectiva hoja de datos. Para eso, a medida que se variaba la altura de la columna de agua de la manguera se obtenía a la salida del sensor su correspondiente valor

de voltaje. De esta forma, se logró generar una tabla con valores de Voltaje vs. Altura, que podremos corresponder con la presión, mediante la ecuación 2.2.

En la tabla 4.1, se puede apreciar que para cada valor de altura de columna de agua tiene asociado un valor de voltaje. A partir de los resultados conseguidos,

TABLA 4.1. Adquisición de datos para la calibración del sensor.

| Número | Altura[cm] | Voltios[V] |
|---|---|---|
| 1 | 29,1 | 1.52 V |
| 2 | 28,1 | 1,48 V |
| 3 | 25,25 | 1,4 V |
| 4 | 23,6 | 1,34 V |
| 5 | 21,9 | 1,26 V |
| 6 | 20,1 | 1,18 V |
| 7 | 17,82 | 1,06 V |
| 8 | 16,5 | 1,03 V |
| 9 | 15,4 | 0,96 V |
| 10 | 14,6 | 0,9 V |
| 11 | 12,7 | 0,84 V |
| 12 | 10,7 | 0,72 V |
| 13 | 8,3 | 0,66 V |
| 14 | 5,55 | 0,51 V |
| 15 | 3,5 | 0,46 V |
| 16 | 1,1 | 0,36 V |
| 17 | 0,7 | 0,3 V |
| 18 | 0,65 | 0,26 V |
| 19 | 0 cm | 0,25 V |

como se puede ver en la tabla 4.1, se pudo proyectar una gráfica que responde a una función lineal como se puede apreciar en la figura 4.2. Es importante notar en la gráfica de la figura 4.2, se encuentra presente un desplazamiento hacia arriba de la función de 0.250V respecto del eje de las abscisas, tal como lo especifica en la hoja de datos del sensor. Entonces para una altura de columna de agua de 0 cm el sensor está entregando a su salida 0.250V. Por lo tanto, luego de esta comprobación se concluyó que el sensor presentó un correcto desempeño.

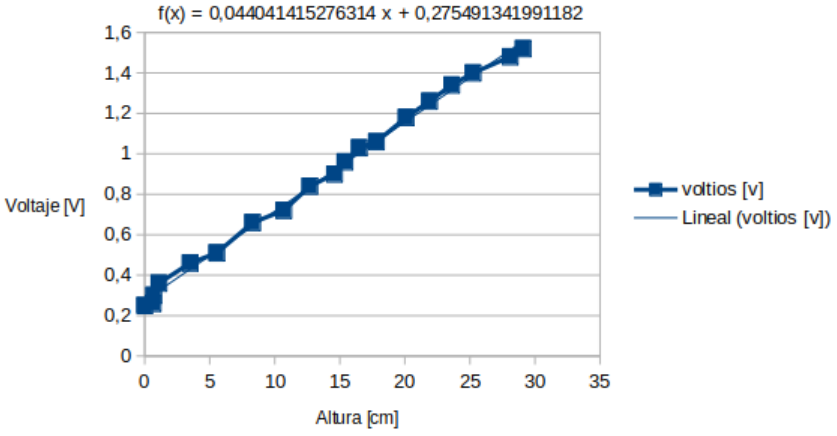


FIGURA 4.2. Función patrón del sensor de presión calibrado: voltaje vs. altura.

## 4.2. Calibración del sensor en el prototipo

Una vez concluida la prueba de correcto funcionamiento del sensor y construido el prototipo se realizó un ensayo que consiste en la adquisición de datos del sistema completo mediante el empleo del firmware. Los datos proporcionados por dicho firmware fueron valores de presión, voltaje y altura del nivel de agua. Además, con el uso de una regla estándar como instrumento de medición, se obtuvo el nivel de agua real en el prototipo. Así, se logró realizar un ajuste, para la necesidad del trabajo en altura correspondiente a la columna de agua, utilizando una regresión lineal. La tabla 4.2, muestra valores obtenidos en el ensayo realizado. Los mismos corresponden a un rango de altura comprendido entre el suelo del canal y el vértice del caudalímetro.

TABLA 4.2. Adquisición de datos del sistema I.

| Número | Presión[Pa] | Voltios[V] | Altura-Sistema[cm] | Altura-Real[cm] |
|---|---|---|---|---|
| 1 | 0,065 | 0,229 | 0,66 | 1 |
| 2 | 0,1 | 0,245 | 1,03 | 1,5 |
| 3 | 0,136 | 0,261 | 1,39 | 2 |
| 4 | 0,179 | 0,281 | 1,83 | 2,6 |
| 5 | 0,229 | 0,303 | 2,35 | 3 |
| 6 | 0,272 | 0,323 | 2,79 | 3,5 |
| 7 | 0,315 | 0,342 | 3,23 | 4,1 |
| 8 | 0,351 | 0,358 | 3,6 | 4,5 |
| 9 | 0,401 | 0,381 | 4,11 | 5,1 |
| 10 | 0,444 | 0,4 | 4,55 | 5,5 |
| 11 | 0,487 | 0,419 | 4,99 | 6,1 |
| 12 | 0,523 | 0,435 | 5,36 | 6,5 |
| 13 | 0,566 | 0,455 | 5,8 | 7 |
| 14 | 0,616 | 0,477 | 6,31 | 7,5 |
| 15 | 0,667 | 0,5 | 6,82 | 8 |
| 16 | 0,703 | 0,516 | 7,19 | 8,5 |
| 17 | 0,753 | 0,539 | 7,7 | 9,05 |
| 18 | 0,81 | 0,565 | 8,29 | 9,55 |
| 19 | 0,846 | 0,581 | 8,66 | 10 |
| 20 | 0,896 | 0,603 | 9,17 | 10,5 |
| 21 | 0,939 | 0,623 | 9,61 | 11 |
| 22 | 0,996 | 0,648 | 10,2 | 11,5 |
| 23 | 1,039 | 0,668 | 10,64 | 12 |
| 24 | 1,09 | 0,69 | 11,15 | 12,5 |

De acuerdo con los valores expuestos en la tabla 4.2, se puede reconocer que la altura entre el suelo del canal y el vértice del caudalímetro es de 12,5 cm. Entonces, si el nivel de altura de agua es menor o igual que dicho valor el caudal de agua es equivalente a cero, ya que no supera la altura del vértice de dicho caudalímetro. En la tabla 4.3, se aprecia que para una altura superior a los 12,5 cm, hay presencia de agua que fluye por la hendidura de la placa de aforo, por lo que se puede obtener valores de caudal de agua, litros/minuto vs altura h de columna de agua.

TABLA 4.3. Adquisición de datos del sistema II.

| Número | Presión[Pa] | Vol.[V] | Alt.-Sistema[cm] | Alt.-Real[cm] | Caud.[l/m] |
|---|---|---|---|---|---|
| 25 | 1 | 0,706 | 11,52 | 13 | 0,1 |
| 26 | 1,19 | 0,735 | 12,18 | 13,6 | 0,3 |
| 27 | 1,269 | 0,771 | 12,91 | 14,4 | 1 |
| 28 | 1,348 | 0,806 | 13,79 | 15,4 | 2,25 |
| 29 | 1,419 | 0,839 | 14,53 | 16,1 | 3,74 |
| 30 | 1,455 | 0,855 | 14,89 | 16,5 | 4,25 |
| 31 | 1,47 | 0,861 | 15,04 | 16,6 | 4,75 |
| 32 | 1,491 | 0,871 | 15,26 | 17,1 | 5,375 |

De la tabla 4.3, se puede deducir que el valor máximo y mínimo de caudal obtenido es de 5,375 y 0,1 l/m respectivamente. De la información obtenida a través de las mediciones que componen la tabla 4.3, se extrajeron los valores correspondientes a las variables caudal y voltaje para construir la tabla 4.4.

TABLA 4.4. Datos obtenidos para construir función polinómica de segundo grado.

| Número | Voltios[V] | Caudal[l/m] |
|---|---|---|
| 1 | 0,706 | 0,1 |
| 2 | 0,771 | 1 |
| 3 | 0,806 | 2,25 |
| 4 | 0,839 | 3,74 |
| 5 | 0,855 | 4,25 |
| 6 | 0,861 | 4,75 |
| 7 | 0,871 | 5,375 |

Estos datos se utilizaron para esbozar una gráfica y comprobar que responde a una función polinómica como la ecuación del modelo 3.1. Seguidamente, en la figura 4.3, se expone la gráfica obtenida y en la que se puede apreciar que manifiesta comportamientos de una función polinómica de segundo grado que involucra la variable caudal en función del voltaje.

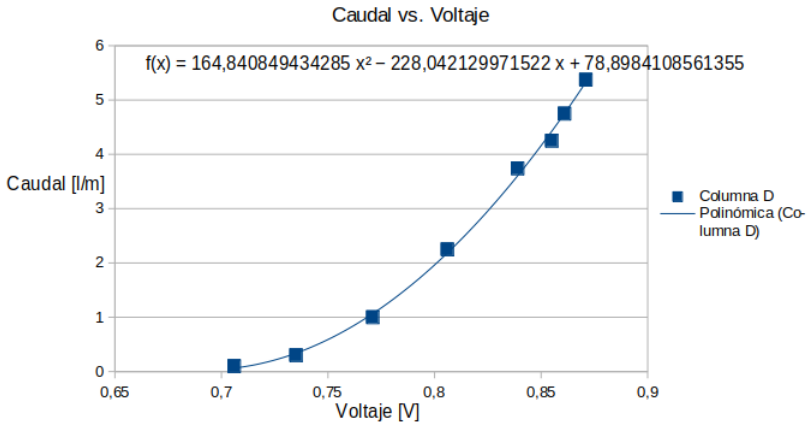


FIGURA 4.3. Función polinómica de segundo grado: caudal vs voltios.

El diagrama correspondiente al prototipo se puede apreciar en la figura 4.4. En la imagen se observa que está compuesto por un depósito que suministra agua al banco de prueba. A su salida se encuentra una llave que durante los ensayos siempre está en el estado de apertura completa. Luego, se ubica la válvula de control con sus propios componentes que la conforman, el motor paso a paso, driver y fracción del firmware. Además se puede notar la presencia del algoritmo de control PI. Posterior a la válvula de control se puede dar con el canal que está construido por placas de madera impermeabilizadas para evitar filtraciones de agua. Finalmente se puede advertir el medidor de caudal constituido por el vertedero con hendidura en V y el sensor de presión.

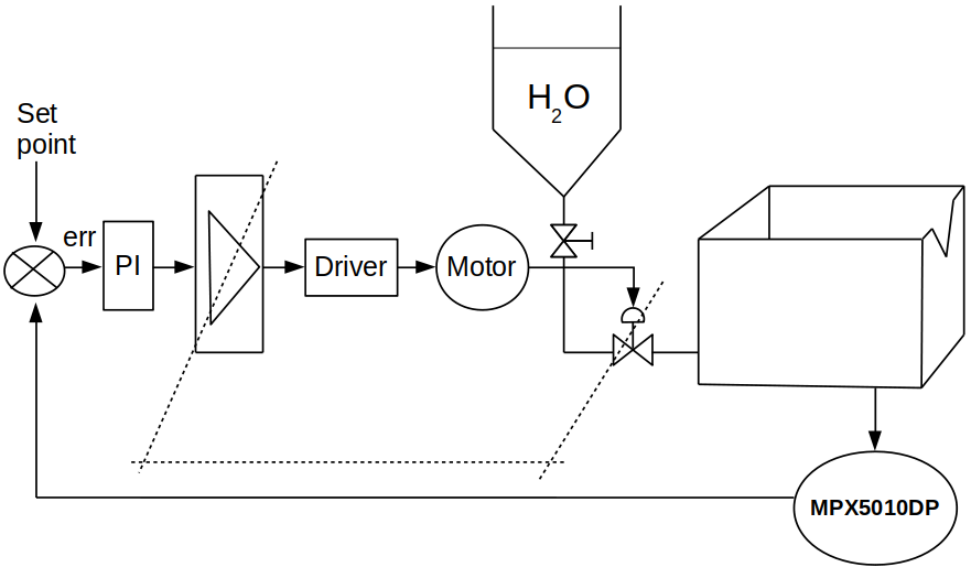


FIGURA 4.4. Diagrama en bloque del prototipo.

## 4.3. Resultados de las pruebas

Al ser un entorno en el que se opera con agua, el tiempo mínimo para lograr el máximo caudal es de unos 120 segundos. Esto surgió de un ensayo que consistió en abrir la válvula de 0 % al 100 % de manera manual e instantánea. Por esta razón, se determinó que el sistema posee un comportamiento lento por naturaleza. En la figura 4.5, se puede apreciar que el sistema con lazo cerrado alcanza el estado de salida deseado.

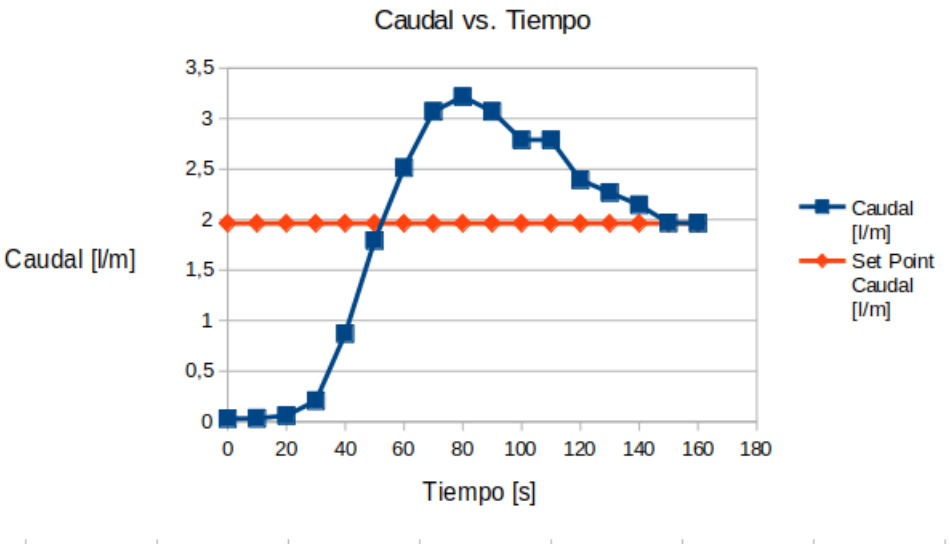


FIGURA 4.5. Función de salida del sistema.

En este ensayo, se estableció un valor de consigna o set point del 35 % del valor máximo de caudal. En la función de salida, se puede notar que en el momento que existe un error y, más aún si es grande, el sistema tiende a minimizarlo. De esta forma, se puede observar que para llevar el caudal de 0 % al 35 % ocupa aproximadamente unos 150 segundos. Sin embargo, el sistema mostró un buen comportamiento durante las pruebas realizadas al establecer el caudal en 0 %, 25 %, 50 %,75 % y 100 % con idénticas respuestas esperadas de salida.

# Capítulo 5

# Conclusiones

En este último capítulo se presentan las conclusiones generales acerca de la respuesta proporcionada por el sistema completo. Además se indica las posibles mejoras y los trabajos pendientes a realizar.

## 5.1. Resultados obtenidos

En el trabajo realizado se logró implementar de forma exitosa un prototipo correspondiente a un sistema de control de caudal en un canal a cielo abierto. A través de las pruebas realizadas se pudo validar y demostrar su correcto funcionamiento. Además, se obtuvieron importantes conclusiones:

1. Se consiguió construir un medidor de caudal de forma tal que el firmware pueda conocer esta variable con mayor precisión.
2. Se construyó una electroválvula cuyo funcionamiento establece el obturador de forma aceptable.
3. Se aplicó un algoritmo de control PI que permite regular el caudal a un valor preestablecido, lo que proporciona una gestión eficiente del fluido.
4. El diseño y la codificación empleados en el desarrollo del firmware, optimiza la escalabilidad de manera sencilla sin pérdida de calidad del sistema.

De esta manera, se pudo completar con la construcción del prototipo que permitirá obtener un mayor control sobre la entrega del recurso en canales abiertos. Debido a los inconvenientes descritos en la sección 2.1, que se generarían en el prototipo original, se tuvo que repensar un modelo alternativo, lo que ocasionó que el objetivo de finalizar en tiempo y forma no pudiera alcanzarse.

## 5.2. Próximos pasos

A continuación se detallan los trabajos futuros que se deben tener en cuenta para desarrollar y lograr un sistema global que pueda controlar el caudal en gran cantidad de compuertas que conforman una red de canal, de modo tal que se pueda desempeñar de forma autónoma y eficiente. Como se trata de un prototipo, es imprescindible, como primera medida, adaptar este sistema a una compuerta de escala real perteneciente a una red de canal de agua y luego someter a este equipo a pruebas de campo para validar su correcto funcionamiento.

El trabajo a efectuar, posterior a las pruebas de campo, es desarrollar un sistema de software principal con características de escalabilidad, de modo tal que permita añadir o integrar una gran cantidad de "celdas primarias", ya que la mayoría de las redes de canales de distribución de agua están conformadas por un gran número de compuertas.

Adicionalmente, este sistema deberá poseer la capacidad de gestionar la comunicación entre diferentes celdas primarias, instaladas en cada una de las compuertas de una red, mediante el empleo de la tecnología Lorawan para que el sistema mismo trabaje de forma independiente. También será necesario desarrollar una aplicación móvil que permita gestionar a través del sistema principal la conducción de flujo de agua hacia un determinado canal para abastecerse de dicho recurso y así cumplir con la dotación de riego de canal establecido previamente.

# Bibliografía


[1] Wikipedia. *Canal de Riego*. https://es.wikipedia.org/wiki/Canal_de_riego. (Visitado 02-02-2020).

[2] Ing. en Rec. Hídr. (M.Sc.) Mario Basán Nickisch. *Aforadores de Corrientes de Agua*. https://inta.gob.ar/sites/default/files/script-tmp-inta-_curso_aforadores_de_agua.pdf. (Visitado 02-04-2019).

[3] Wikipedia. *Vertedero de Pared Delgada*. https://es.wikipedia.org/wiki/Vertedero_de_pared_delgada. (Visitado 02-02-2020).

[4] Proyecto CIAA. *Computadora Industrial Abierta Argentina*. Visitado el 2016-06-25. 2014. URL: http://proyecto-ciaa.com.ar/devwiki/doku.php?id=start.

[5] Leadshine Technology Co. Ltd. *User's Manual For MA860H*. https://www.leadshine.com.ua/pdf/ma860h.pdf. (Visitado 02-04-2019).

[6] Leadshine Technology Co. Ltd. *86HS Series Hybrid Stepping Motors*. http://www.elindar.com.ar/img/leadshine/motor-pap/86HSxxdm.pdf. (Visitado 02-04-2019).

[7] NXP Semiconductors. *MPX5010DP*. URL: https://www.nxp.com/docs/en/data-sheet/MPX5010.pdf.

[8] *FreeRTOS$^{TM}$, Real-time operating system for microcontrollers*. https://www.freertos.org/. (Visitado 26-06-2019).

[9] Ian Smmorville. *Ingeniería de Software*. 2011.

[10] Wikipedia. *Diagrama de flujo*. https://es.wikipedia.org/wiki/Diagrama_de_flujo. (Visitado 02-04-2019).

[11] James Rumba, Ivar Jacobson y Grady Booch. *EL Lenguaje Unificado de Modelado. Manual de Referencia*. 2000.

[12] *Diagramando UML con PlantUML*. 2019.

[13] Wikipedia. *Diagrama de clases*. https://es.wikipedia.org/wiki/Diagrama_de_clases. (Visitado 02-04-2019).

[14] *CONCEPTOS BÁSICOS, TERMINOLOGÍA Y TÉCNICAS PARA EL CONTROL DE PROCESOS*. http://infoplc.net/files/documentacion/control_procesos/infoPLC_net_ControlProcesos.pdf. (Visitado 15-05-2020).

[15] Brett Beauregard. *Improving the Beginner's PID – Introduction*. http://brettbeauregard.com/blog/2011/04/improving-the-beginners-pid-introduction/. Mar. de 2011. (Visitado 08-08-2020).

[16] Bruce Powel Douglass. *Design Patterns For Embedded System in C*. 2011.