UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO

Facultad de Estudios Superiores Aragón

“Sistema para automatizar la distribución de agua en los depósitos del hogar incorporando el paradigma de internet de las cosas”.

T E S I S

Para obtener el título de:

INGENIERO ELÉCTRICO ELECTRÓNICO

Presenta:

López Cruz Ángel Iván

Director de Tesis:

Dr. Ismael Díaz Rangel

FES Aragón

Nezahualcóyotl, Estado de México, 2023

Agradecimientos

En el punto culminante de esta etapa de mi vida académica, quiero dedicar un espacio especial a las personas que han sido mi fuente inagotable de apoyo, amor y aliento a lo largo de esta travesía:

A mis queridos padres: Martina Cruz Núñez y Angel Jesús López Santiago, desde el primer día de mi educación hasta este logro que hoy celebro, ustedes han sido mi guía constante y mi inspiración. Sus palabras alentadoras, su inquebrantable fe en mí y su sacrificio incansable han sido los cimientos sobre los que he construido mi camino hacia el éxito.

A mis queridas hermanas: Lizbeth López Cruz y Melanie Jatzirie López Cruz, han sido más que hermanas para mí, han sido mis amigas y compañeras incansables.
Desde nuestros primeros pasos en este camino hasta este logro que celebro hoy, ustedes han estado a mi lado, brindándome su apoyo incondicional y compartiendo cada paso de esta travesía. Sus palabras de ánimo, su interminable paciencia y su inquebrantable confianza en mí han sido mi guía constante y mi fuente de fortaleza.

Al DR. Ismael Díaz Rangel. Gracias por su dedicación para ayudarme a alcanzar mis metas educativas. Sus conocimientos y consejos han enriquecido mi aprendizaje y han sido una fuente constante de inspiración. Su compromiso con mi éxito académico ha sido excepcional, y estoy verdaderamente agradecido por su paciencia y guía a lo largo de este proceso.

En resumen, su influencia en mi educación y desarrollo no tiene precio, y estoy profundamente agradecido por su compromiso con mi crecimiento personal y profesional.

A mi querida novia Hanaid Andrea Flores Anastacio.

Hoy, al finalizar este importante capítulo de mi vida con la culminación de mi tesis, quiero dedicarte unas palabras de agradecimiento muy especiales. Han pasado cinco años desde que comenzamos nuestro viaje junto como pareja, y durante ese tiempo, has sido mi fuente constante de amor, apoyo y motivación. Tu presencia en mi vida ha hecho que cada desafío sea más llevadero y cada logro sea más significativo.

En esta tesis, veo reflejada una parte de ti, ya que has sido mi compañera en este recorrido académico. Tu paciencia infinita en las noches de estudio, tus palabras de aliento cuando la inspiración escaseaba y tu comprensión incondicional en los momentos de estrés ha sido un regalo inigualable.

Estos cinco años de noviazgo y este logro académico están entrelazados en mi historia, y quiero agradecerte por ser mi compañera, mi mejor amiga y mi amor inquebrantable. Tu amor y apoyo me han sostenido en los momentos difíciles y han multiplicado la alegría de los momentos felices.

Mirando hacia el futuro, estoy emocionado por todos los nuevos capítulos que aún nos esperan. Gracias por ser parte integral de mi vida y por ser la razón detrás de mi éxito.

Con todo mi amor y gratitud.

ÍNDICE

Capítulo 1. Introducción

El agua es una fuente de vida en el planeta, alrededor del 0.007% de agua es apta para consumo humano en forma de lagos, ríos, represas y fuentes subterráneas poco profundas, reabastecidos por los procesos de evaporación y precipitación (UNIVERSIDAD TEC Virtual DEL SISTEMA TECNOLÓGICO DE MONTERREY, 2022).

La disponibilidad de agua es un desafío actual y complejo que involucra una serie de factores más allá del aumento de la población que demanda cada vez más este recurso para el consumo humano y actividades económicas. El crecimiento urbano-industrial, la sobreexplotación y la contaminación de los recursos hídricos han provocado conflictos y escasez de agua, impactando de manera significativa a ciudades y localidades (Duran Juárez & Torres Rodríguez, 2006), como en el caso del Municipio de Ecatepec de Morelos.

En este contexto, el Municipio de Ecatepec enfrenta una serie de dificultades para satisfacer la demanda de agua potable. El crecimiento de la población, junto con el desarrollo urbano e industrial, ha sobrepasado la capacidad de la infraestructura hidráulica existente, lo que se traduce en un suministro insuficiente para abastecer las necesidades de sus habitantes. Esta situación se agrava por la distribución desigual del agua, generando inequidades donde algunos disponen de suministro ininterrumpido, mientras que otros enfrentan situaciones de escasez.

Existen casos, donde comunidades solo cuentan con el abasto en algunos días y horarios al azar, con flujos intermitentes y que para sus habitantes es un desafío obtener y racionalizar el agua disponible. Para quienes cuentan con tinaco, bajo el contexto mencionado, deben estar revisando continuamente su nivel para utilizar el agua de manera adecuada para cubrir sus necesidades, pero debido a la ubicación de estos depósitos, puede ser difícil su revisión e incluso representar riesgos a la integridad física.

Para abordar estos problemas, se propone el desarrollo de un sistema para realizar el monitoreo de nivel de agua en un tinaco, el cual se presenta en una aplicación móvil/web, para que los usuarios puedan conocer el porcentaje de agua presente en el depósito (cero a cien por ciento). También se cuenta al nivel de piso, un módulo que muestre con tres LED, que indican estimaciones del nivel de agua.

Adicionalmente, el sistema monitorea continuamente el nivel de agua de la cisterna o deposito primario, y cuando existe la presencia de agua proveniente de la red pública, en caso de ser necesario, activa una bomba presurizadora, para acelerar el llenado del depósito. Cabe aclarar que hay regiones donde el uso de estas bombas está prohibido, y que de ser el caso, solo debe omitirse en la implementación su incorporación, lo cual no requiere modificar algo adicional, tanto en componentes como en la programación; es decir, todo podrá funcionar de acuerdo a la esperado.

También se propone, que a través de la aplicación se configuren los niveles para la activación de la bomba que surte al tinaco, así como del establecimiento de horarios de encendido de dicha bomba, ya que el ser considerablemente ruidoso, podría perturbar el sueño de los habitantes.

Objetivo general

Desarrollar un sistema para automatizar el bombeo de agua cisterna-tinaco en el hogar incorporando el paradigma de internet de las cosas.

Objetivos particulares

- Activar una bomba presurizadora para reducir el tiempo de llenado de la cisterna o depósito primario.
- Incorporar opciones para activación-desactivación del sistema en horarios definidos por el usuario.
- Configurar niveles para encendido-apagado de bomba de agua a criterio del usuario.
- Monitorear los niveles de agua en depósitos (cisterna y tinaco).
- Desarrollar la etapa de control para su operación con independencia a la comunicación con la aplicación web.
- Utilizar un microcontrolador con capacidades de wifi para la gestión lógica y de comunicación del sistema.
- Desarrollar una interfaz de usuario que permita el monitoreo y configuraciones de manera remota, mediante una aplicación web y/o móvil.

Actividades

- Investigar bombas de agua.
- Conocer sensores de flujo de agua.
- Revisar sensores de nivel de agua.
- Analizar relojes de tiempo real.
- Examinar etapas de potencia.
- Distinguir plataformas IoT.

Justificación

La propuesta busca proporcionar varios beneficios importantes para los residentes, los cuales pueden ser denotados de la siguiente manera:

Comodidad: un sistema automático releva a las personas de realizar esta actividad, y lo puede hacer de manera más eficiente, ya que se mantiene todo el tiempo realizando las acciones de gestión de bombeo de agua, de acuerdo con las indicaciones configuradas por parte del usuario. Además, al tener la opción de restringir por horarios la activación de la bomba destinada al tinaco, contribuye a no perturbar el descanso de los usuarios.

Seguridad: cuando se tienen problemas de abastecimiento de agua, es muy importante conocer cuánta agua hay disponible en los depósitos, para el caso de los tinacos, suelen estar ubicados en lugar que son de difícil acceso, e incluso imposibles para persona con alguna

discapacidad motriz, y para adultos mayores. Incluso para quienes no tienen las condiciones mencionadas, se corre el riesgo de caídas. Gestionar el agua es importante, porque al tener poca disponible, se deben establecer prioridades, como el uso en los baños, limpieza y preparación de alimentos, así como lavado de manos; que de no realizarse, podrían conducir a condiciones de salud como enfermedades gastrointestinales, entro muchos problemas más.

Ahorro: un punto es que para los usuarios, ya no necesitan invertir tiempo en estar al pendiente del llenado de sus depósitos y bombeo de agua, lo cual les permite invertir su tiempo en otras actividades, también la automatización impide el desbordamiento de agua, lo cual es un desperdicio de tan preciado recurso, y podría representar también un ahorro energético, ya que suele darse el caso de encender un bomba de agua y por descuido dejarla trabajando de más o incluso sin agua, conduciendo al riesgo de dañarla.

Descripción del capitulado

Capítulo 1 Introducción. Se detallan los objetivos generales del proyecto, así como los objetivos particulares, la justificación del trabajo donde se dice porque se realizó este trabajo y con qué fin, los antecedentes que dan un panorama de prototipos parecidos que fueron hechos.

Capítulo 2 Marco Teórico. Se muestra un panorama de conocimientos general para el entendimiento del trabajo, las definiciones de los dispositivos a usar, software utilizado en el trabajo, sensores, amplificadores, entre otros conocimientos de electrónica.

Capítulo 3 Desarrollo Experimental. En este capítulo se muestra la metodología que se siguió para desarrollar los diferentes dispositivos, detallando el procedimiento de creación como: imágenes de los modelos 3D, modelos impresos, programación, esquemas de bloques, diagramas de flujo, y desarrollo de circuitos.

Capítulo 4 Pruebas y Resultados. En este capítulo se muestran los prototipos de los dispositivos, sus errores, modelos que no fueron funcionales, adaptaciones y cambios que se hicieron para luego probarlos.

Se continúa proporcionando las conclusiones y las mejoras a realizar como trabajo a futuro; por último, se indican las referencias tomadas y los anexos del trabajo.

Capítulo 2. Marco teórico

En este apartado se proporciona información relativa a la distribución de agua, dando particular énfasis a los hogares, así como la información teórica y técnica referente a los diversos dispositivos considerados para el desarrollo experimental de los diversos módulos creados para satisfacer los objetivos de este trabajo.

2.1 Abastecimiento de agua en ciudades y sus problemáticas

Una red de abastecimiento de agua potable es aquella que facilita que el agua avance desde el punto de captación hasta el punto de consumo en condiciones aptas para su consumo. Por aptas no solo se entiende en cuanto a condiciones sanitarias de calidad, sino también de cantidad (AM GROUP, 2023).

2.1.1 Clasificación de las obras de captación

Se denomina "obras de captación" a las obras civiles y electromecánicas que permiten disponer del agua superficial o subterránea de la fuente de abastecimiento. (Siapa, 2014).

Las superficiales se refieren a fuentes visibles, como son ríos, arroyos, lagos y lagunas, mientras las subterráneas, a fuentes que se encuentran confinadas en el subsuelo, como pozos, Manantiales y galerías filtrantes (Rodriguez Gonzalez & Cortez Guardiola, 2020).

2.1.2 Líneas de conducción

La línea de conducción es la parte del sistema que transporta el agua desde el sitio de la captación ya sea por medio de bombeo y/o rebombeo, o a gravedad, hasta un tanque de regulación, Planta potabilizadora o un crucero predeterminado de la red. También se considera como parte de la línea de conducción al conjunto de conductos, estructuras de operación y especiales y cruceros.

Su capacidad se calculará con el gasto máximo diario (QMD), o con el que se considere conveniente según el sitio de procedencia, según lo autorice el SIAPA. De los accesorios que se tienen que instalar junto con las líneas de conducción tanto a gravedad como por bombeo, se deberán tomar en cuenta las válvulas de seccionamiento, expulsoras de aire, combinadas, de flotador, altitud, Check, de alivio de presión (en bombeos), desfogues, juntas de dilatación, etc., cuya ubicación y cantidad variará de acuerdo con el proyecto en cada caso.

También se denomina línea de conducción a la línea de interconexión entre pozos y que conduce uno o varios caudales acumulados (Siapa, 2014).

2.1.3 Obras de regulación y almacenamiento.

La regulación es la parte del sistema de abastecimiento de agua potable que tiene por objeto lograr la transformación de un régimen de aportaciones (de la conducción) que normalmente es constante, en un régimen de consumos o demandas (de la red de distribución) que siempre es variable. Cuando además de ser regulador, el tanque tiene la capacidad de almacenar suficiente agua para dos días de reserva a Gasto Medio diario (Qmd), entonces se considera como tanque de Almacenamiento. En ambos casos, los tanques siempre deben proporcionar

un servicio continuo y eficiente, bajo normas y condiciones estrictas de higiene y seguridad. La construcción del tanque deberá ser considerada en un predio cuya función será la de alojar la estructura de regulación y los servicios de la propia infraestructura cuya ubicación deberá ser revisada y validada por el área técnica y el área operativa, dicho terreno podrá ser ubicado dentro del desarrollo o fuera del mismo siendo responsabilidad del desarrollador su obtención y entrega al sistema operador (Siapa, 2014).

2.1.4 Tratamiento

Aquí se procede a purificar las aguas. Este tratamiento cambiará dependiendo la calidad del 'agua bruta'. Consta de las siguientes partes:

- *Reja*. Impide el paso de 'material grueso' y lo retira una vez en la superficie. Este material puede ser superficial y flotante, o de arrastre de fondo.
- *Desarenador*. Impide el paso de materiales en suspensión.
- *Floculadores*. Añaden productos químicos para decantar materiales finos y sustancias en suspensión coloidal.
- *Decantadores y filtros*. Los decantadores, llamados también sedimentadores, apartan una parte de material fino. Los filtros son útiles para retirar el material en suspensión.
- *Filtros*. Para retirar totalmente el material en suspensión.
- Dispositivo de desinfección.

(AM GROUP, 2023).

2.1.5 Redes de distribución

El sistema de distribución consiste en una red de tuberías subterráneas que tiene por objeto entregar el agua hasta la entrada de los predios de los usuarios. Este sistema se forma con dos partes principales:

A. Instalaciones del servicio público: De acuerdo con la magnitud de sus diámetros, las tuberías se clasifican en: líneas de alimentación, redes primarias, redes secundarias o de relleno y tomas domiciliarias.

B. Instalaciones particulares: Instalación hidráulica de toda la edificación, que a partir del límite de propiedad, es responsabilidad de los usuarios, pero deben cumplir con el Reglamento de Instalaciones Hidráulicas, Sanitarias y Pluviales en vigor (Siapa, 2014).

2.1.6 Problemáticas

Es importante abordar las problemáticas por la falta de agua suficiente para satisfacer las necesidades de la población.

- Problemas de traslado desde el punto de captación, hasta el punto de consumo tales como:
    - No hay suficiente agua en el punto de captación.
    - Problemas de Bombeo en el área de captación.
    - Fugas en los ductos de conducción.
    - Falta de agua suficiente en el tanque antes de llegar al usuario final.

- La falta de acceso al agua tiene efectos graves en la salud pública, ya que limita la capacidad de mantener una higiene adecuada, lo que aumenta la propensión a contraer enfermedades.

2.2 Métodos de llenado de tinacos y medición de nivel en depósitos

Existen dos métodos de llenados, el llenado manual que consiste en el llenado del depósito proveniente de la llave de la red pública, o bien encendiendo la bomba de manera tradicional. Y los sistemas de llenado automático los cuales garantizan un proceso seguro con la intervención de electrónica.

2.2.1 Sistema de llenado automático

Electronivel. Es un sensor que detecta cuando el nivel del agua sube o baja tanto en la Cisterna como en el Tinaco e indica a la Bomba de agua cuándo encender y cuándo detenerse de forma automática y confiable. (Rotoplas, 2023)

La operación del electronivel se realiza por medio del posicionamiento. Cuando se encuentra de forma vertical o inclinado hacia arriba abre el circuito para evitar el paso del agua. Al estar de manera horizontal o inclinada hacia abajo enciende la bomba y permite el ingreso de agua para llenar por completo el depósito. (MN Del Golfo, 2023)

Para el funcionamiento óptimo del Tinaco o Cisterna se recomienda el uso de un Electronivel en el Tinaco y otro en la Cisterna como se indica en la ilustración 2.1. La utilización de un solo electronivel en un sistema Tinaco-Cisterna, implica posibles riesgos que van desde tirar el agua cuando el Electronivel se coloca solamente en la Cisterna, hasta quemar la Bomba en el caso de ubicarlo únicamente en el Tinaco (Rotoplas, 2023)

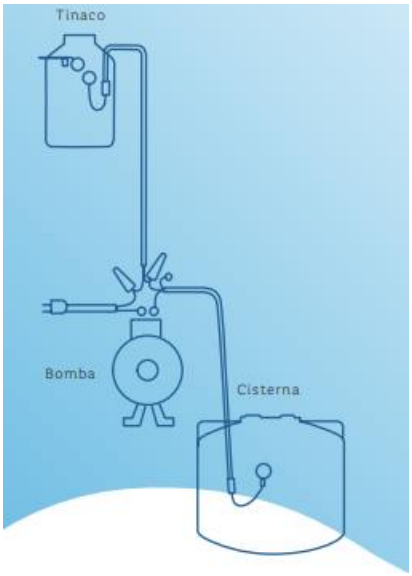


Ilustración 2.1 Instalación de electronivel.

Electrodos. Un electronivel con electrodos es un dispositivo que controla automáticamente el nivel en depósitos de agua u otros líquidos, accionando la bomba cuando el nivel está por debajo de lo deseado; también puede ayudar a proteger a la bomba por bajo nivel de succión. (Menecaxa, 2021).

Basa su funcionamiento en la detección de la fuente de agua (Cisterna, pozo, etcétera) para inhibir la salida del relevador en caso de no haber agua en el origen.

Como se muestra en la ilustración 2.2 al suministrar energía al control, se checa el estado del suministro u origen (sección A) en caso de estar en nivel bajo el control espera hasta que el agua toque el sensor nivel lleno. Una vez que la cisterna está en nivel lleno se checará el nivel del depósito destino o tinaco (sección B), si está en nivel bajo el relevador se activará y se detendrá hasta que el agua haya tocado el sensor de nivel lleno si en el proceso de llenado el agua llegará al sensor nivel bajo únicamente entonces la salida se apagará y solo se activará nuevamente hasta que la cisterna este en nivel lleno.

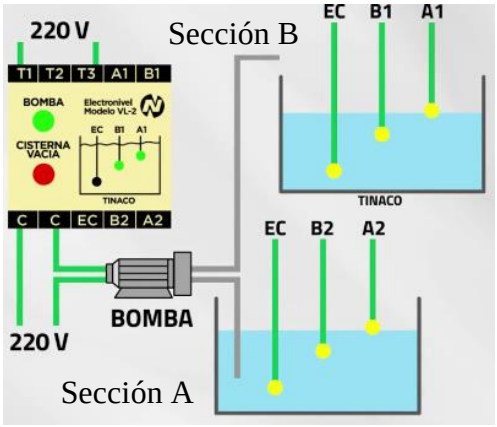


Ilustración 2.2 Funcionamiento de llenado con electrodos.

Sensor ultrasonido. Este método es utilizado para circuitos experimentales donde se ocupan microcontroladores como se muestra en la ilustración 2.3.

El sistema va a medir el nivel del tanque y a partir de dos parámetros prefijados para el nivel alto y el nivel bajo, va a tomar la decisión de encender o apagar la bomba. En otras palabras, si por ejemplo, se toma como nivel bajo el 40% del total de la capacidad del tanque y como nivel alto el 90%, el sistema va a mandar a encender la bomba (por medio del relé) cuando el nivel sea inferior a 40% hasta que llegue al 90% donde será apagada.

Paralelamente a este proceso, mientras esté encendida la bomba, el sistema va a chequear que exista flujo a su salida (que la bomba esté "jalando" agua) y en caso de que no exista flujo a la salida de la bomba, la va a mandar a parar, esto es para evitar que se queme, pues una bomba encendida sin bombear agua un tiempo relativamente corto se quema, pues se basa en el agua para su propia lubricación y enfriamiento (Alexander, Rivas Alpizar, 2020).

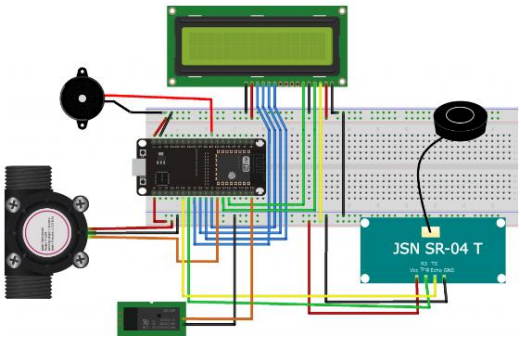


Ilustración 2.3 Circuito llenado automático con sensor de ultrasonido.

2.2.2 MEDICIÓN DE NIVEL EN DEPÓSITOS

Para visualizar el nivel de agua en los depósitos, hay dos métodos comunes los electrodos y el sensor de ultrasonido.

Medición con electrodos. Para su visualización en la ilustración 2.4 se ocupa un circuito ULN2803 el cual tiene en su interior un conjunto de ocho puertas inversoras implementado con transistores NPN tipo Darlington.

Cuando el terminal de entrada (pin B) se conecta a una tensión positiva de entre 3 y 5 voltios, los transistores conducen y quedan eléctricamente conectados los terminales de salida (pin C) y el negativo de la alimentación. Por lo tanto, el LED enciende al encontrarse directamente polarizado.

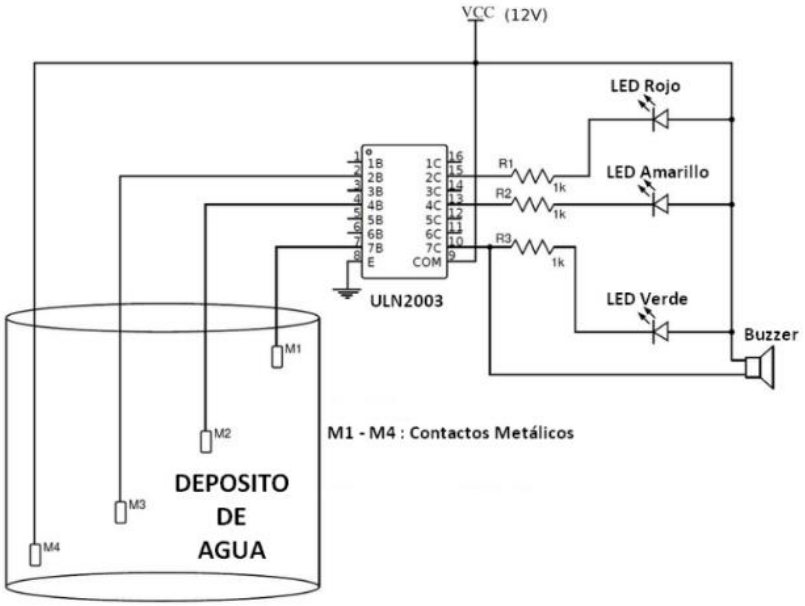


Ilustración 2.4 Conexión electrodo con ULN2803.

Cuando retiramos la tensión positiva del terminal de entrada, los transistores dejan de conducir y el LED se apaga. El terminal positivo de la alimentación está conectado al "común" (patilla 10) del integrado para evitar, a través del diodo en cada una de las salidas, que cualquier flujo negativo transitorio de corriente pueda dañar sus transistores cuando se producen las conmutaciones de conducción a no conducción (Hernandez, 2015).

Medición con sensor ultrasonido. El sistema se basa principalmente en medir el nivel del tanque por medio del sensor ultrasónico impermeable JSN SR-04T el cual va a estar ubicado en la tapa del tanque apuntando hacia el agua. En la figura que se muestra se aprecia cómo se logra medir el nivel de agua (N) con este sensor, a partir de tener la altura del tanque (H) (que es constante) y la distancia medida por el sensor (S), el nivel de agua sería la resta de la altura del tanque menos la distancia medida por el sensor: N = H – S. Entonces con esa sencilla ecuación es posible conocer el nivel existente en nuestro tanque. Este debería ser el primer paso, como se ve en la ilustración 2.5 se debe colocar el sensor en la tapa del tanque y visualizar por la pantalla LCD el nivel de agua existente en el tanque (Alexander, Rivas Alpizar, 2020)

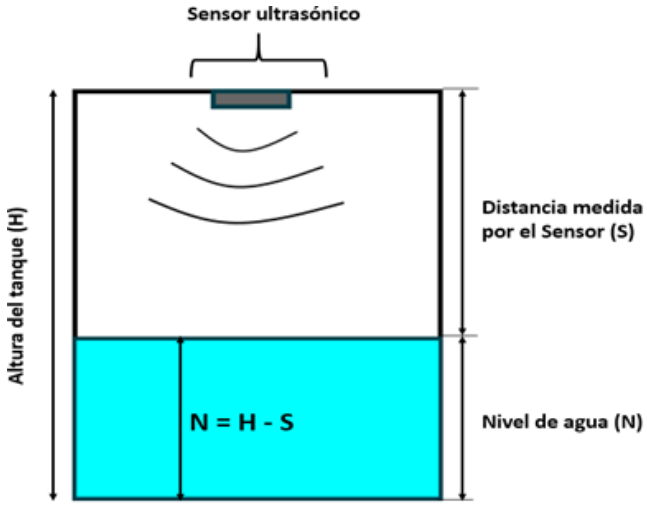


Ilustración 2.5 Medición de nivel de agua con sensor ultrasonido.

2.3 Bombas de agua

Una bomba de agua es una máquina que transforma la energía mecánica en energía de fluido (Auto Solar, 2023). Su funcionamiento es sencillo y básico, el agua es aspirada por el tubo de entrada de la bomba, para luego ser impulsada por el motor que crea un campo magnético con las bobinas e imanes, y así lograr que el impulsor gire de una manera continuada. A medida que gira el rotor, se mueve el fluido que alimenta la bomba. Las palas son las encargadas de impulsar los fluidos fuertemente, haciendo que el agua pase rápidamente de la entrada a la salida. El objetivo de estas palas del impulsor es que el agua entre al centro del rotor haciendo que la fuerza centrífuga sea tan fuerte por la compresión del fluido, que genere una presión haciendo que el fluido salga con rapidez y gran caudal, siendo este el propósito final. No solo se trata de pasar agua de un lado a otro sino de ahorrar tiempo, y muchas veces elevar ese fluido de una zona de menor presión o altitud a otra de mayor presión o altitud (ELECTROBOMBAS JAVEA TECNOLOGIAS DEL AGUA, 2023).

2.3.1 Bomba sumergible

Una bomba sumergible está diseñada para trabajar con todo el conjunto, que consiste en bomba y motor, completamente sumergidos en el líquido o medio a procesar. Este tipo de bomba tiene un motor herméticamente sellado que está estrechamente acoplado al cuerpo de la bomba. La carcasa estanca alrededor del motor generalmente se llena con aceite para protegerlo de daños al evitar la entrada de cualquier líquido que pueda causar un cortocircuito.

Cuando una bomba está sumergida hay presión positiva del fluido en la entrada de la bomba. Esta condición puede crear una mayor eficiencia debido a la menor energía requerida para mover el fluido a través de la trayectoria del líquido de la bomba.

Una bomba sumergible funciona empujando, en lugar de extraer, líquido durante el proceso de bombeo. Esto es extremadamente eficiente porque la bomba utiliza el cabezal de líquido en el que está sumergida para operar y no se gasta energía en atraer el líquido a la bomba. Un efecto positivo de la bomba sumergida es que el motor es enfriado por el líquido que lo rodea, evitando el sobrecalentamiento (EDDYPUMP CORPORATION, 2023).

Instalación bomba sumergible doméstica

1. La motobomba debe ser instalada en posición vertical.
2. Sujete el flotador de la bomba en el seguro para ajustar el encendido y apagado de la bomba. (Se sugiere aproximadamente 5 a 10 cm de longitud del cable del seguro al flotador).
3. Es necesario controlar la bomba (encendido y apagado) con el flotador del tinaco o depósito de agua donde se va a bombear el agua.
4. Para mejores resultados, calcule correctamente la demanda de su sistema y el diámetro correcto de la tubería.
5. Para máxima eficiencia de la descarga, utilice tubería de por lo menos el mismo diámetro de la descarga de la bomba o el diámetro superior.
6. Es necesaria la instalación de una válvula check de 1" en la descarga de la bomba para evitar que la tubería del servicio se descargue y la bomba trabaje sin control, además de reducir el golpe de ariete. (EVANS, 2023)

En la ilustración 2.6 se muestra el diagrama de la correcta instalación de la bomba

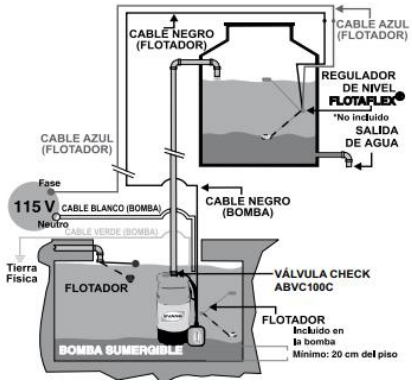


Ilustración 2.6 Diagrama de instalación Bomba sumergible.

Instalación bomba sumergible tipo bala.

1. La bomba debe ser instalada en vertical.
2. calcular correctamente la demanda del sistema y el diámetro correcto de la tubería, para evitar el sobrecalentamiento en la etapa de potencia.

En la ilustración 2.7 se muestra instalada la bomba tipo bala.

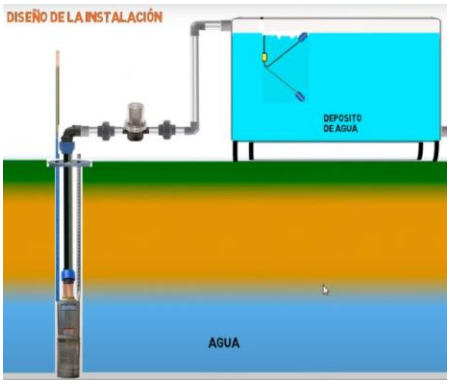


Ilustración 2.7 Diagrama de instalación bomba tipo bala.

2.3.2 Bombas centrífugas.

Una bomba centrífuga es una máquina que consiste en convertir la energía en velocidad y posteriormente en energía a presión. Es decir, transforman la energía mecánica en energía hidráulica. De esta manera, puede mover el mayor volumen de líquido posible.

El funcionamiento de una bomba centrífuga es especialmente sencillo. En primer lugar, el líquido entra por el rodete o impulsor, donde a través de unos álabes se dirige el fluido, y gracias a la fuerza centrífuga, se expulsa dicho líquido o fluido hacia el exterior. Una vez en el exterior es la carcasa la encargada de recogerlo. (fluideco, 2019)

En la figura 2.8 se muestra el funcionamiento de una bomba centrífuga.

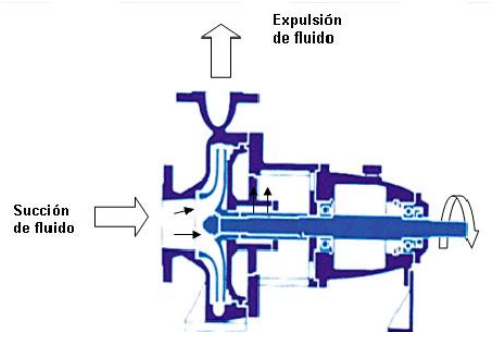


Ilustración 2.8 principio de funcionamiento de una bomba centrifuga.

Instalación.

En la ilustración 2.9 se muestra detalladamente la correcta posición de una bomba centrifuga, así como todos los componentes que debe llevar, en la parte inferior del tubo de succión va ubicado la válvula de pie, antes de arrancar la bomba se debe abrir el tapón y cebar la bomba (ingresar agua hasta que esté completamente lleno).

Las uniones universales nos sirven para cuando queramos reemplazar o darle un mantenimiento a la bomba y las superiores para cambiar las válvulas ya sea de paso o la de check. La tubería que se está usando para esta bomba es de 1 pulgada, ya que en la bomba sus medidas de succión y de descarga son esas. No es recomendable reducir esta medida ya que lo que va a ocurrir es perder presión del agua sobre todo lo que es en la parte de succión. (Electrotec, 2023)

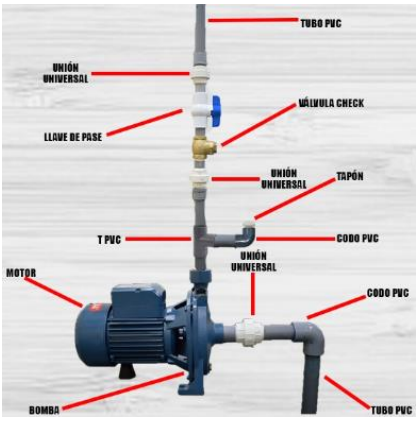


Ilustración 2.9 Instalación de bomba centrifuga.

2.3.3 Bomba booster o presurizadora.

La bomba presurizadora es un aparato que genera presión sobre el agua para facilitar el acceso a este líquido desde cualquier lugar que se necesite. Además, obtendrá un mejor aprovechamiento del agua al detener las goteras y evitar la humedad, así como también ayudará en el óptimo funcionamiento de las tuberías.

El motor se activa automáticamente cuando se abre el grifo e impulsa el agua hasta su lugar de destino. Esta función, no solo sirve para ahorrar en el consumo del líquido, sino también para darles mayor vida útil a las tuberías. (Ferremax, 2021).

Instalación

Si la bomba es instalada en una tubería donde se pueden producir bolsillos de aire, se recomienda instalar un sistema de venteo automático en la tubería, como se muestra en la ilustración 2.10 (Ferreabasto, 2020)

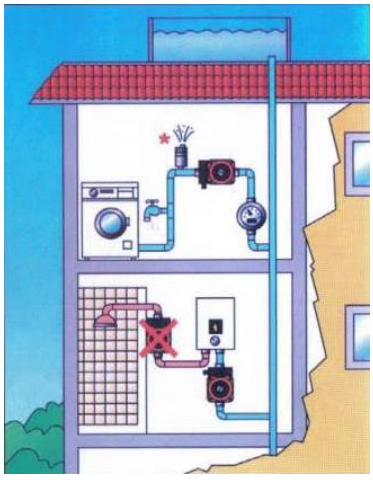


Ilustración 2.10 Ejemplo de instalación de una bomba presurizadora.

La bomba debe ser instalada con el eje del motor en posición horizontal, en la ilustración 2.11 se muestra las posiciones idóneas de instalación. (Ferreabasto, 2020)

Ilustración 2.11 posiciones de instalación.

2.4 sensores de flujo de agua.

Los sensores de flujo de agua son utilizados para el monitoreo de fluidos en tuberías

2.4.1 Sensor efecto Hall

El caudal de agua ingresa al sensor y hace girar una turbina, la turbina está unida a un imán que activa un sensor de efecto Hall, que a su vez emite un pulso eléctrico que puede ser leído por la entrada digital de un Arduino O ESP32 como se muestra en la ilustración 2.12. El sensor de efecto Hall está aislado del agua, de manera que siempre se mantiene seco y seguro.

Como el volumen de agua por cada pulso es fijo y de un valor conocido (promedio) podemos contar la cantidad de pulsos por unidad de tiempo (segundo o minuto), luego multiplicar el valor de volumen/pulso por la cantidad de pulsos y así determinar el caudal o flujo de agua. Se recomienda utilizar interrupciones por hardware en el Arduino para detectar o contar los pulsos del sensor. (NAYLAMP MECHATRONICS, 2022)

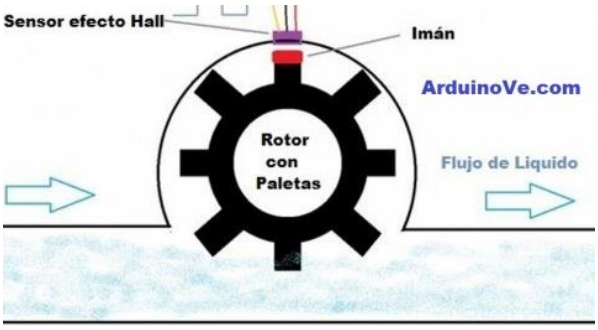


Ilustración 2.12 Sensor de flujo de agua de efecto hall.

2.4.2 interruptor de caudal

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 tal como lo muestra la ilustración 2.13. (eicos 40, 2022)

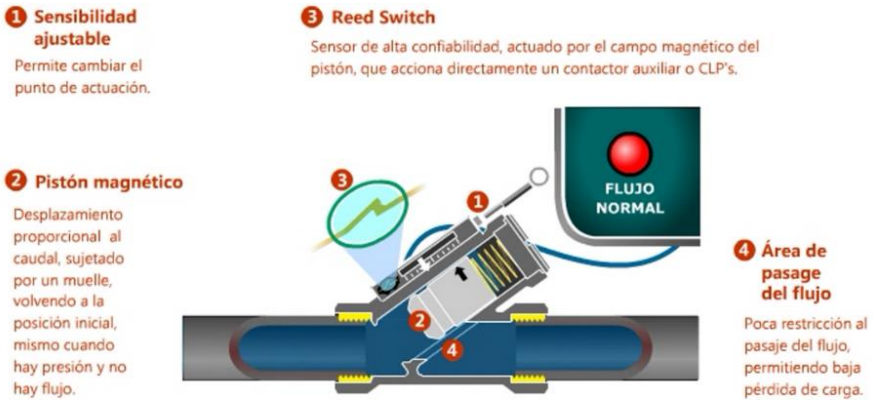


Ilustración 2.13 Funcionamiento Interruptor de caudal.

2.5 Reloj de tiempo real (RTC).

Un RTC tiene la función principal de realizar un seguimiento preciso del tiempo incluso cuando se apaga una fuente de alimentación o se coloca un dispositivo en modo de bajo consumo. Los RTC se componen de un controlador, un oscilador y un resonador de cristal de cuarzo integrado para contabilizar de forma correcta el paso del tiempo. Están diseñados como dispositivos todo en uno para brindar un mejor rendimiento que los componentes discretos, simplificar la integración en nuevos diseños.

Las funciones del RTC se denominan registros. Los datos de registro se programan en la memoria RAM. Los registros se actualizan periódicamente, incluso durante el funcionamiento normal del RTC. El diseño RTC también incluye una función de interruptor de encendido para el funcionamiento con batería u otra fuente de energía de respaldo de baja potencia. Esto permite que el RTC mantenga conteos de tiempo precisos y continuos incluso si la unidad entra en modo de suspensión o si se pierde la alimentación principal. También alivia la necesidad de que el usuario restablezca la hora y la fecha cada vez que se cicla el suministro del dispositivo.

Los RTC se utilizan en una variedad de aplicaciones en las que desempeñan un papel fundamental en el seguimiento preciso de la hora actual al mismo tiempo que proporcionan alarmas, temporizadores y funciones de interrupción y ayudan a reducir el consumo de energía. (ECS INC INTERNATIONAL, 2023)

2.5.1 Interrupciones y variaciones

Durante el funcionamiento, el RTC se puede programar para enviar una alarma o activar un indicador para proporcionar un reloj y un calendario completos. Estas funciones de interrupción están activas cuando el RTC está funcionando en la fuente de respaldo (estado de energía Vbat). Para cada ocurrencia y variación de interrupción, el RTC creará una marca de tiempo digital del evento para revisar las consistencias.

Los ejemplos de interrupciones RTC incluyen actualizaciones de tiempo periódicas, cuenta regresiva periódica para temporizadores, detectores de bajo voltaje, cambio automático de fuente de alimentación, reinicio de encendido y alarmas. Las alarmas se basan en la configuración de tiempo en los registros. Se genera una interrupción de alarma y luego el tiempo coincide con los registros de configuración, luego el pin /INT 10 pasa a un nivel bajo y activa la interrupción de alarma. Esto será lo mismo para los temporizadores de cuenta regresiva para una notificación de que se ha producido un evento. (ECS INC INTERNATIONAL, 2023)

2.5.2 Fuente de alimentación

Normalmente, un RTC estará alimentado por una fuente de alimentación del sistema principal durante el funcionamiento normal. Sin embargo, se requiere una fuente de energía de respaldo dedicada para que un RTC realice un seguimiento preciso del tiempo sin interrupción y mantenga el control del tiempo después de un corte de energía. Los RTC están diseñados para detectar un nivel bajo o falta de voltaje de suministro principal y cambiarán automáticamente a una batería o suministro secundario que mantiene el reloj interno,

Continuará recurriendo a la fuente de respaldo hasta que se restablezca la alimentación del sistema principal o se recargue la batería. (ECS INC INTERNATIONAL, 2023)

2.5.3 Modelos

RTC en hardware

Existen dos RTC habituales el DS1307 y el DS3231, ambos fabricados por Maxim (anteriormente Dallas Semiconductor). La comunicación en ambos modelos se realiza a través del bus I2C, por lo que es sencillo obtener los datos medidos. La tensión de alimentación es 4.5 a 5.5 para el DS1307, y 2.3 a 5.5V para el DS3231.

Frecuentemente estos módulos también incorporan una pequeña EEPROM AT24C32, que puede ser empleada para almacenar registros y mediciones. También incorporan una batería CR2032 para mantener el dispositivo en hora al retirar la alimentación. Esta batería debería ser capaz de mantener alimentado durante varios años al DS1307, y durante meses al DS3231. La tensión de alimentación de batería es de 2.0 a 3.5 para el DS1307 y de 2.3 a 5.0 para el DS3231. (Luis Llamas, 2016)

RTC en software

- Esp32. Si observamos en la ilustración 2.14, uno de los módulos internos del micro es un RTC funcional, el cual se utiliza en varios procesos, y en uno de ellos tenemos manejar hora y fecha con precisión. (Microelectronics, 2023)

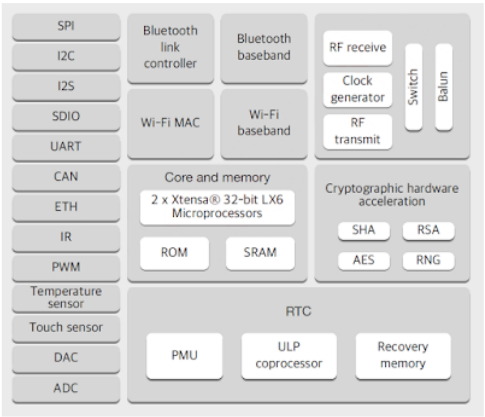


Ilustración 2.14 Funcionamiento interno ESP32.

- Blynk IOT ofrece una forma de obtener tiempo para usar en sus aplicaciones integradas que se ejecutan en dispositivos, la única desventaja de usar este RTC virtual es que en todo momento tiene que estar conectado a internet.

2.6 Transmisión de datos

"La comunicación serial es un protocolo estandarizado que permite el intercambio de información en forma de bits entre dos o más dispositivos. Existen dos modos básicos para realizar la transmisión de datos y son:

- Modo asíncrono.
- Modo síncrono.

Comunicación Sincrónica: es una técnica que consiste en el envío de una trama de datos (conjunto de caracteres) que configura un bloque de información comenzando con un conjunto de bits de sincronismo (SYN) y terminando con otro conjunto de bits de final de bloque (ETB). En este caso, los bits de sincronismo tienen la función de sincronizar los relojes existentes tanto en el emisor como en el receptor, de tal forma que estos controlan la duración de cada bit y carácter. En la ilustración 2.15 podemos ver un ejemplo de comunicación sincrónica.

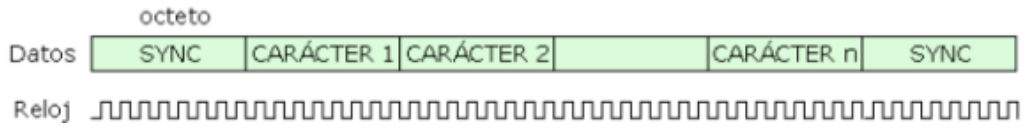


Ilustración 2.15 EJEMPLO DE TRANSMISIÓN SINCRÓNICA.

Comunicación Asíncrona: En esta transmisión el emisor decide cuándo va a enviar el mensaje por la red, mientras que el receptor no sabe en qué momento le puede llegar dicho mensaje, para esto se utiliza un bit de cabecera que va al inicio de cada carácter y uno o dos bits de parada que van al final de ese mismo carácter, esto se hace con la finalidad que tanto el emisor como el receptor puedan sincronizar sus relojes y poder decodificar el mensaje. Podemos ver un ejemplo de transmisión asincrónica en la ilustración 2.16 En este tipo de transmisión no se maneja mucha velocidad ya que cada carácter es transmitido de uno en uno y por lo tanto puede ser un poco lenta.

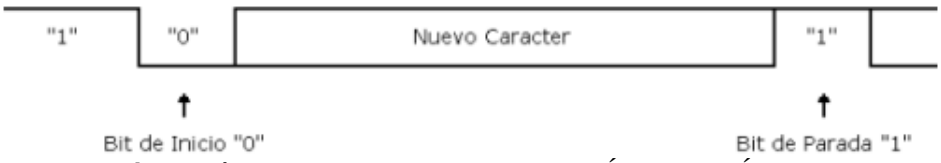


Ilustración 2.16 EJEMPLO DE TRANSMISIÓN ASINCRÓNICA.

Protocolo RS-232. Fue introducido por primera vez en 1962 por el sector de radio de la Alianza de Industrias Electrónicas (EIA). (ARC Electronics, 2010). Este protocolo en sus comienzos se utilizó para la comunicación entre dispositivos, conocidos en la jerga como dispositivos DTE (data terminal equipment) y dispositivos DCE (data communication equipment). Un equipo DTE es un equipo que convierte la información del usuario en señales, o convierte las señales recibidas. Los dispositivos DTE originales eran teletipos, y los dispositivos DCE originales eran usualmente módems, que a su vez transmitían los datos por la línea telefónica o por transmisores de radio para hacer packet. La versión actual de este protocolo es la TIA-232-F "Interfaz Between Data Terminal Equipment and Data Circuit-Terminating Equipment Employing Serial Binary Data Interchange" lanzada en 1997.

(CAMI Research, 2023)

Protocolo RS-485. También conocido como EIA-485, que lleva el nombre del comité que lo convirtió en estándar en 1983. Es un estándar de comunicaciones en bus de la capa física del modelo OSI. Está definido como un sistema en bus de transmisión multipunto diferencial, es ideal para transmitir a altas velocidades sobre largas distancias (35 Mbit/s hasta 10 metros y 100 kbit/s en 1200 metros) y a través de canales ruidosos, ya que reduce los ruidos que aparecen en los voltajes producidos en la línea de transmisión. El medio físico de transmisión es un par entrelazado que admite hasta 32 estaciones en 1 solo hilo, con una longitud máxima de 1200 metros operando entre 300 y 19 200 bit/s y la comunicación half-duplex(semi

duplex) soportando hasta 32 transmisiones y 32 receptores. La transmisión diferencial permite múltiples drivers dando la posibilidad de una configuración multipunto. Al tratarse de un estándar bastante abierto permite muchas y muy diferentes configuraciones y utilizaciones, y su resistencia al ruido lo hace ideal en ambientes sensibles o industriales. (Texas Instruments, 2010)

Protocolo Bus I2C es el acrónimo de Bus Inter-IC. El bus I2C fue desarrollado en los comienzos de los años ochenta, por Phillips Semiconductors. El propósito original de este protocolo era proveer una manera sencilla de conectar la CPU con los chips periféricos en los televisores y demás sistemas de complejidad media-alta. Hoy por hoy, este bus es aceptado por la industria como un estándar de-facto. Asimismo, fue adoptado por los fabricantes líderes como ST Microelectronics, Atmel, Texas Instruments entre otros. (Embedded Systems Academy, 2015). Actualmente se encuentra en la revisión 4, del 13 de febrero del 2012. (NXP, 2015)

Protocolo USB. El Universal Serial Bus más conocido por la sigla USB, es un bus estándar industrial que define los cables, conectores y protocolos usados en un bus para conectar, comunicar y proveer de alimentación eléctrica entre computadoras, periféricos y dispositivos electrónicos. (Allusb.com, 2023)

Bus SPI (del inglés Serial Peripheral Interfaz) es un estándar de comunicaciones, usado principalmente para la transferencia de información entre circuitos integrados en equipos electrónicos desarrollados en un principio por Motorola. El bus de interfaz de periféricos serie o bus SPI es un estándar de facto para controlar casi cualquier dispositivo electrónico digital que acepte un flujo de bits serie regulado por un reloj. Es un protocolo full-duplex que funciona bajo un paradigma maestro-esclavo. Incluye una línea de reloj, dato entrante, dato saliente y un pin de chip select, que conecta o desconecta la operación del dispositivo con el que uno desea comunicarse. De esta forma, este estándar permite multiplexar las líneas de reloj. (Microchip Technology, 2014)" (La comunicacion Serie, 2023).

2.7 Etapas de potencia

Dentro de los dispositivos electrónicos de potencia, podemos citar: los diodos y transistores de potencia, el tiristor, así como otros derivados de éstos, tales como los triac, diac, conmutador unilateral o SUS, transistor uniunión o UJT, el transistor uniunión programable o PUT y el diodo Shockley (Aguilar Peña & Montejo Ráez, 2023).

El componente básico del circuito de potencia debe cumplir los siguientes requisitos:

- Tener dos estados claramente definidos, uno de alta impedancia (bloqueo) y otro de baja impedancia (conducción).
- Poder controlar el paso de un estado a otro con facilidad y pequeña potencia.
- Ser capaces de soportar grandes intensidades y altas tensiones cuando está en estado de bloqueo, con pequeñas caídas de tensión entre sus electrodos, cuando está en estado de conducción. Ambas condiciones lo capacitan para controlar grandes potencias.
- Rapidez de funcionamiento para pasar de un estado a otro (Aguilar Peña & Montejo Ráez, 2023).

2.7.1 Tiristores.

Este tipo de semiconductores se compone de cuatro capas P-N-P-N y ofrece dos estados estables, uno de conducción y otro de bloqueo.

Entre los diferentes tipos, el rectificador controlado de silicio (SCR) es el más utilizado.

El tiristor consta habitualmente de tres electrodos: un ánodo (terminal positivo), cátodo (terminal negativo) y por último la puerta (gate) como se muestra en la ilustración 2.17. (Universidad Nacional de PIura, 2023)

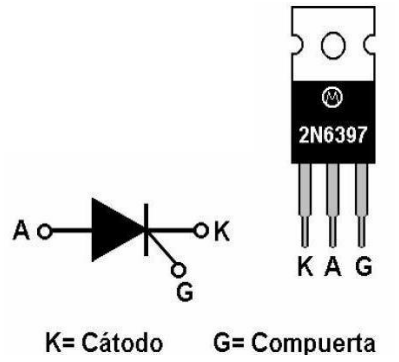


Ilustración 2.17 2N6397 Y Símbolo de un tiristor.

Tipos de tiristor:

- **Rectificador controlado de silicio (SCR):** interruptor unidireccional capaz de conmutar cantidades importantes de energía, se cierra con un pulso de corriente de puerta (disparo) y se abre cuando la corriente pasa por cero.
- **Tiristor GTO (Gate Turn-Off):** se enciende por un solo pulso de corriente positiva en la terminal puerta (G), al igual que el SCR, pero permite el apagado con un pulso negativo en el mismo terminal de puerta.
- **Interruptor controlado de silicio (SCS):** es un componente similar a un SCR, con la diferencia que este tiristor cuenta con un terminal de puerta adicional. Este terminal permite ejercer más control sobre el dispositivo, particularmente en el modo de conmutación forzada, donde una señal externa lo obliga a apagarse mientras que la corriente principal a través del dispositivo aún no ha caído por debajo del valor de la corriente de retención.
- **TRIAC:** Los triac son similares a los SCR, pero conducen en ambas direcciones como se muestra en la ilustración 2.18, lo cual significa que puede conmutar corrientes AC y DC. (Universidad Nacional de PIura, 2023)

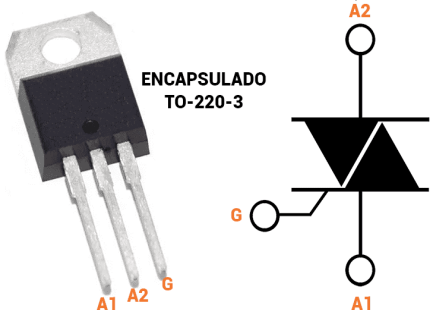


Ilustración 2.18 BTA24 y símbolo de un triac.

2.7.2 Diodos

Son dispositivos unidireccionales, no pudiendo circular la corriente en sentido contrario al de conducción. El único procedimiento de control es invertir el voltaje entre ánodo y cátodo como se muestra en la ilustración 2.19.

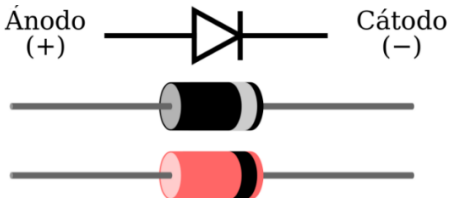


Ilustración 2.19 simbología del Diodo.

Los diodos de potencia se caracterizan porque en estado de conducción, deben ser capaces de soportar una alta intensidad con una pequeña caída de tensión. En sentido inverso, deben ser capaces de soportar una fuerte tensión negativa de ánodo con una pequeña intensidad de fugas. (uv.es, 2023)

- **Diodo Zener:** El diodo Zener es un semiconductor de silicio con una unión p-n que está diseñado específicamente para trabajar en condiciones de polarización inversa. Cuando está en polarización directa, se comporta como un diodo de señal normal, pero cuando se le aplica tensión inversa, el voltaje permanece constante para un amplio rango de corrientes.
  Debido a esta característica, se utiliza como regulador de voltaje en circuitos de corriente continua. El principal objetivo del diodo Zener como regulador de voltaje es mantener un voltaje constante. Digamos que si se utiliza un voltaje Zener de 5 V, el voltaje se mantiene constante en 5 V y no cambia. (ELECTRÓNICAONLINE, 2023) En la ilustración 2.20 se muestra su simbología.

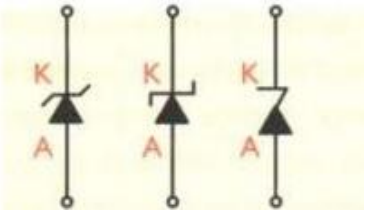


Ilustración 2.20 Simbología del Diodo Zener.

- **Diodo de cuatro capas:** este componente cuenta con un ánodo y un cátodo como se muestra en la ilustración 2.21. Cuando se aumenta la tensión entre ambos polos y se supera la tensión de ruptura, el diodo conmuta del estado de bloqueo al estado de conducción.

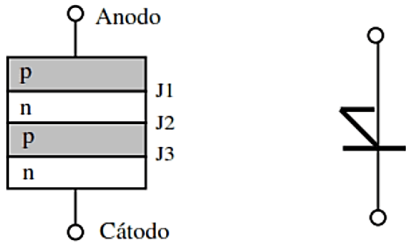


Ilustración 2.21 Estructura y símbolo Diodo cuatro capas.

- **DIAC:** Diac es una mezcla de las palabras "diodo" e "interruptor ac", es similar al diodo de cuatro capas, pero puede conducir en ambas direcciones, lo que significa que puede usar en aplicaciones de corrientes AC y DC (RS, 2023) tal como se muestra en la ilustración 2.22.

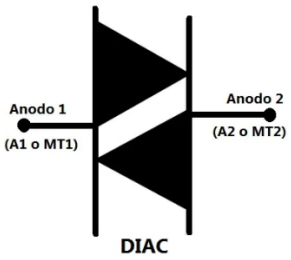


Ilustración 2.22 Símbolo DIAC.

2.7.3 Transistores de potencia.

Un transistor es un equipo electrónico que tiene como objetivo realizar la entrega de una señal de salida con relación a una señal de entrada. Los **transistores de potencia** son dispositivos semiconductores que tiene una estructura de funcionamiento igual a los transistores normales, pero con la característica de poder percibir y generar altas potencias. En la ilustración 2.23 se muestra la simbología del transistor en su configuración PNP y NPN.

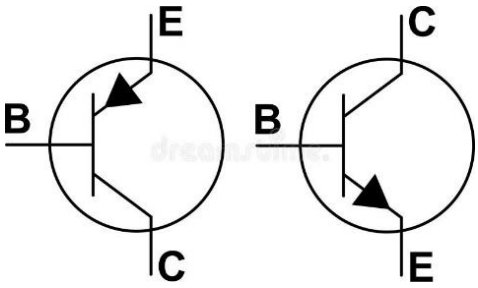


Ilustración 2.23 simbología del transistor en configuración "PNP" "NPN".

Transistores de potencia bipolar: Es un instrumento con dos uniones (PN) que se encuentran muy unidas. Este tipo de transformador permite aumentar la corriente, disminuyendo el voltaje y puede controlar la energía por medio de sus terminales.

Esta clasificación se subdivide en transistores bipolares (NPN y PNP) y su diferenciación radica en los materiales con los cuales son construidos, así como el sentido de la corriente de polarización, como se observa en la ilustración 2.24.

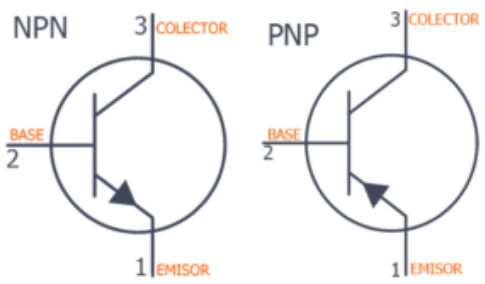


Ilustración 2.24 simbología del transistor bipolar en configuración "PNP" "NPN".

Transistores de potencia unipolares: Este dispositivo está compuesto por una capa semiconductora tipo N sobre un material tipo P. Su campo eléctrico puede controlar la conducción de un canal (GSL Industrias, 2021). Los transistores unipolares cuentan con cuatro terminales: puerta (representado con la G de "gate"), drenador (D de "drain"), fuente (S de "source") y sustrato (B de "substrate"), pero el sustrato generalmente está conectado internamente al terminal de fuente y por este motivo se pueden encontrar casi siempre los dispositivos FET con tres terminales. La puerta está separada del cuerpo por medio de una capa de aislante (parte en blanco), (Solectro, 2022) como se muestra en la ilustración 2.25.

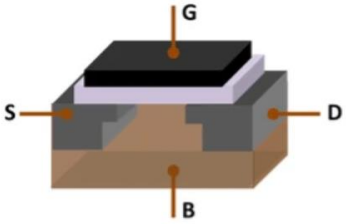


Ilustración 2.25 Transistor Unipolar.

2.7.4 Relevadores.

Son dispositivos electromagnéticos que se encargan de abrir y cerrar el paso de la corriente eléctrica y son accionados bajo este mismo tipo de energía.

Los relevadores tienen una bobina conectada a una corriente, cuando esta se activa produce un campo electromagnético, el cual provoca que el contacto del relé que se encuentra normalmente abierto se cierre y de esta forma, permite el paso de la corriente por un circuito para ejercer cierta acción, como arrancar un motor.

Cuando se deja de proveer corriente a la bobina, el campo electromagnético se retira y el contacto del relé se vuelve a abrir, dejando sin corriente al circuito eléctrico que iba al motor o al objeto en cuestión.

Tipo armadura. Un electroimán causa la basculación de una armadura al ser activado, cerrando o abriendo los contactos dependiendo de si está normalmente abierto o cerrado como se muestra en la ilustración 2.26.

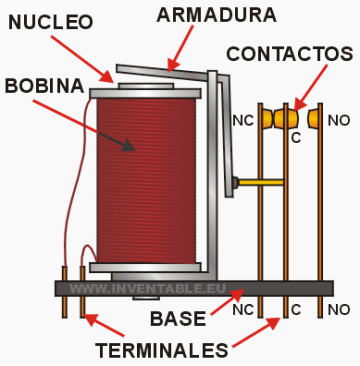


Ilustración 2.26 Componentes del relevador tipo armadura.

Núcleo móvil. Por su mayor fuerza de atracción, se usa un solenoide para cerrar sus contactos y se utiliza para el control de altas corrientes a continuación en la ilustración 2.27 se muestra el diseño del relevador.

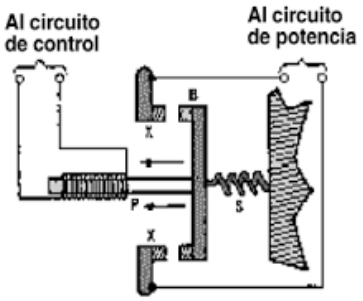


Ilustración 2.27 Diseño de relevador de Núcleo móvil.

De lengüeta. Estos relevadores están conformados por una ampolla de vidrio con contactos en la parte interior y se encuentran montados sobre delgadas láminas de metal como se muestra en la ilustración 2.28.

Ilustración 2.28 Relevador de lengüeta DIP05.

Polarizados. Conformados por una armadura pequeña, solidaria a un imán permanente. El extremo inferior gira dentro de los polos de un electroimán, mientras que el otro lleva una cabeza de contacto. Al excitar el electroimán, se mueve la armadura y causa el cierre de los contactos. Si se polariza de forma contraria, el giro ocurrirá al revés, abriendo los contactos o cerrando otro circuito. En la ilustración 2.29 se muestra un ejemplo del relevador polarizado.

Ilustración 2.29 Relevador polarizado físico.

De estado sólido. Se trata de un circuito híbrido, conformado por un optoacoplador que aísla la entrada, un circuito de disparo que detecta el paso por cero de la corriente de línea y un triac que actúa de interruptor de potencia. En la ilustración 2.30 se muestra un relevador de estado sólido.

Ilustración 2.30 Relevador de estado sólido SSR-40DD.

De corriente alterna. Cuando se excita la bobina de los relevadores con corriente alterna, el flujo magnético en el circuito magnético también es alterno, ocasionando una fuerza pulsante con frecuencia doble sobre los contactos. En la ilustración 2.31 se muestra un relevador de corriente alterna en donde se observa de manera externa, donde son sus respectivas conexiones.

Ilustración 2.31 Relevador de corriente alterna STW1000.

Con retardo a la conexión. El retardo a la conexión se consigue de manera mecánica, por medio del aumento en la masa de la armadura para obtener mayor inercia del sistema móvil. También se consigue un efecto parecido de retardo utilizando C. C. para alimentar al relé. En la ilustración 2.32 se muestra el ejemplo de un relevador de retado a la conexión de la marca Weg.

Ilustración 2.32 Relevador con retardo a la conexión Weg

Con retardo a la desconexión. El retardo a la desconexión también se consigue de manera mecánica, pero disminuyendo la presión de los resortes del relé. En la ilustración 2.33 se observa un relevador de la marca Lexo.

Ilustración 2.33 Relevador con retardo a la desconexión Lexo.

Con retención de posición. Los relevadores cuentan con un diseño que contiene remaches de elevada remanencia y se encuentran dentro de orificios practicados en el núcleo y la armadura de estos con exacta coincidencia. (SDI Industrial, 2022)

2.9 plataformas IoT

Dentro de los tipos de plataformas para proyectos IoT se encuentran cuatro grupos distintos. El primer grupo, son las plataformas que están orientadas a startups y empresas pequeñas. En este tipo he incluido todas aquellas que nos permiten su uso de una manera gratuita, pero con limitaciones en cuanto al número de mensajes enviados y de dispositivos conectados.

Están enfocadas exclusivamente a dispositivos u objetos conectados. Estas características las hacen ideales para utilizarlas en nuestros proyectos del IoT con Arduino.

El segundo grupo engloba a plataformas que también ofrecen servicios gratuitos o versiones de prueba. Están más centradas en ofrecer servicios globales a sistemas basados en el IoT. Ya no es solo recibir datos, en estas plataformas nos permiten almacenar webs, API para móviles, bases de datos, etc… Podemos decir que son el paso intermedio entre las enfocadas claramente al IoT y las plataformas de las grandes corporaciones.

El siguiente grupo serían las plataformas que ofrecen las grandes empresas y corporaciones como Google, Amazon, Microsoft, IBM, etc… Están orientadas sobre todo al sector industrial y a grandes proyectos del IoT, donde se ven involucrados cientos o miles de dispositivos.

El último grupo englobaría las plataformas de código abierto. Son todas aquellas que nos dan acceso al código sin restricciones. Podemos descargarlas e instalarlas en nuestras máquinas de forma local. (del Valle Hernández, 2023)

2.9.1 Esquema general para proyectos IoT

En este sistema intervienen tres elementos principales:

- El dispositivo conectado o del IoT
- La plataforma en la nube
- Los dispositivos que consumen la información en la plataforma del IoT

Cada uno de estos sistemas se trata por separado y se comunican entre ellos a través de protocolos de comunicación. Estos protocolos deberían ser un estándar para que, independientemente de la plataforma, se puedan comunicar. (del Valle Hernández, 2023)

Dispositivos del IoT

En este sentido, tenemos una amplia gama de microcontroladores. Los más comunes son Arduino, Raspberry y la gama de ESP, mediante sus placas se configuran, se crean circuitos y se conectan mediante wifi, Ethernet, etc. (del Valle Hernández, 2023)

API de acceso a los datos

Esto es algo muy importante que hay que valorar a la hora de crear proyectos IoT con Arduino. Tener una API (Application Programming Interface) nos permitirá consultar, modificar y borrar la información desde otros dispositivos.

Al final tenemos que entender que es una capa de comunicación estándar para conectarnos a los datos. Existen diferentes protocolos y estándares. El más utilizado sería a través de servicios web RESTful.

El acceso a dicha API dependerá del software desde donde nos conectemos. Si se trata de una aplicación web, frameworks como jQuery, AngularJS o React, nos facilitan esta tarea enormemente. (del Valle Hernández, 2023)

Plataformas del IoT

Dentro de esta categoría encontramos plataformas orientadas exclusivamente a proyectos del IoT y otras plataformas que ofrecen múltiples servicios para todo un sistema del IoT.

Los protocolos de comunicación más utilizados son HTTP, MQTT y CoAP. Además, para comunicarnos, existen diferentes redes como LoRa o SigFox. Son redes WAN para el IoT y una alternativa a los sistemas tradicionales de comunicación. (del Valle Hernández, 2023)

2.9.2 Tipos de plataformas

ThingSpeak (Ilustración 2.34). Está enfocado exclusivamente a la construcción de aplicaciones del IoT. Permite almacenar datos, visualizarlos y exponerlos a otras APIs.

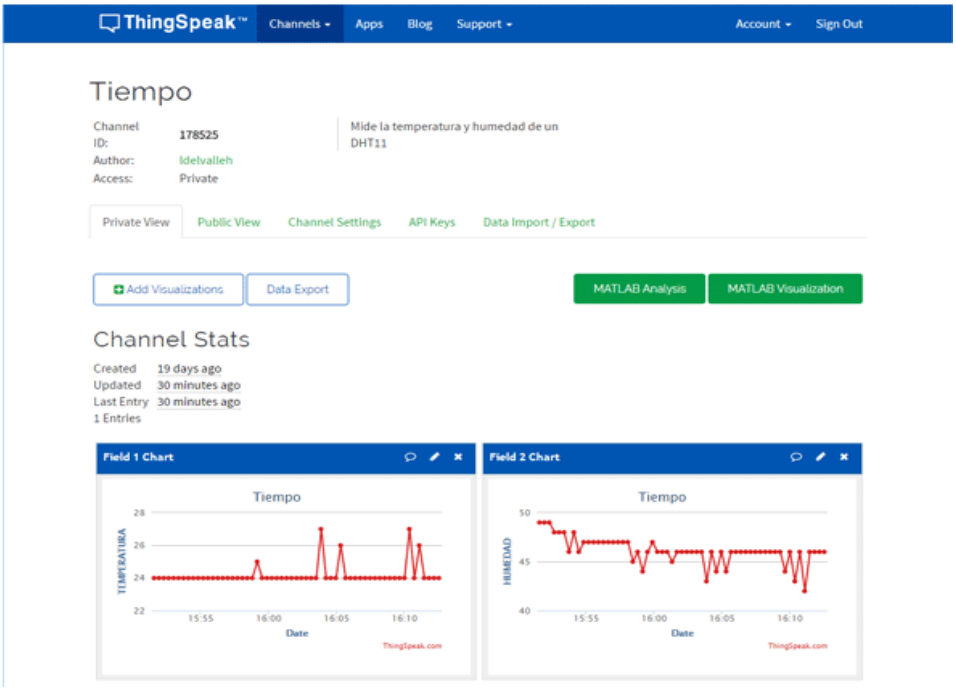


Ilustración 2.34 Visualización de ThingSpeak.

Esta aplicación es de código abierto, se puede descargarla de GitHub y utilizarla en proyectos locales.

La base de esta plataforma son los canales. En ellos se almacenan los datos que se le envía y se compone de 3 elementos:

- 8 campos para almacenar datos de cualquier tipo.
- 3 campos para almacenar la ubicación, latitud, longitud y elevación. Por supuesto que necesitaríamos un componente que nos diera esta información.
- 1 campo para almacenar el estado.

Cada uno de estos campos puede ser actualizado cada 15 segundos. Dentro del código hay librerías para muchos lenguajes de programación y por supuesto, para Arduino. Podemos encontrar esta librería para utilizarla en proyectos del IoT con Arduino, dentro del entorno de desarrollo oficial. (del Valle Hernández, 2023)

CAYENNE

El punto fuerte de Cayenne son las capacidades de IO para que pueda controlar de forma remota sensores, motores, actuadores, incluidas los puertos de GPIO con almacenamiento ilimitado de datos recogidos por los componentes de hardware, triggers y alertas, que proporcionan las herramientas necesarias para la automatización y la capacidad de configurar alertas. Además, también puede crear cuadros de mando personalizados para mostrar su proyecto con arrastrar y soltar widgets que también son totalmente personalizables (SOLO ELECTRONICOS, 2023). Tal como se observa en la ilustración 2.35.

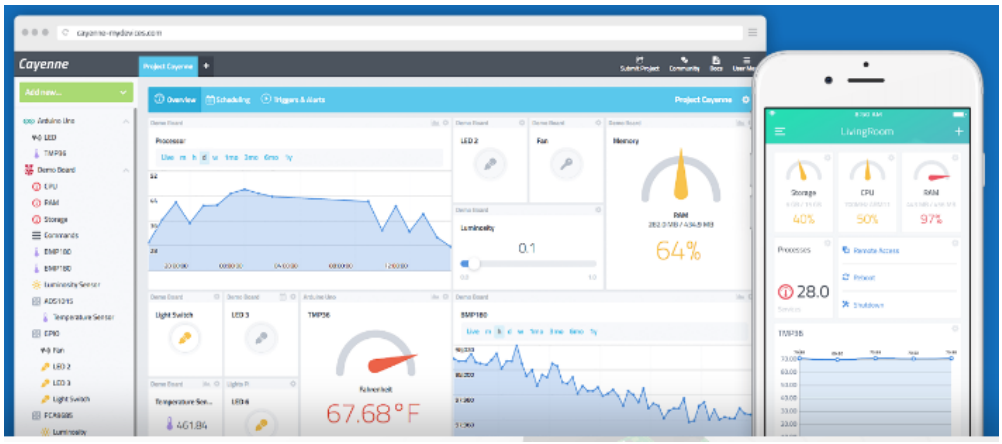


Ilustración 2.35 Visualización de Cayenne.

Resumidamente algunas características clave de esta novedosa plataforma son las siguientes:

- Una aplicación móvil para configurar, el monitor y los dispositivos de control y sensores desde cualquier lugar.
- Motor de reglas para desencadenar acciones a través de dispositivos.
- Panel personalizable con widgets de visualización de arrastrar y soltar.
- Programación de las luces, motores y actuadores
- Control de GPIO que se pueden configurar desde una aplicación móvil o desde un navegador
- Acceso remoto instantáneo desde su smartphone o con un ordenador
- Para construir un proyecto de la IO a partir de cero se ha logrado el objetivo de proporcionar un Proyecto Generador de IO que reduce el tiempo de desarrollo de horas en lugar de meses.

Blynk

Es una plataforma diseñada específicamente para interactuar con proyectos maker desde un teléfono móvil. Con Blynk se puede controlar el hardware remotamente, almacenar o mostrar datos de sensores.

La ilustración 2.36 muestra cómo funciona Blynk. En el smartphone (ya sea Android o iPhone). Como primer paso se instala la aplicación Blynk que se comunica con un servidor intermedio que interactúa con los proyectos y dispositivos IoT haciendo uso de las bibliotecas de Blynk.

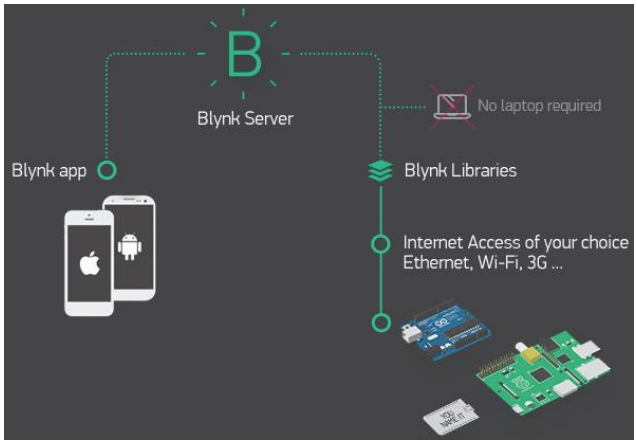


Ilustración 2.36 Comunicación Blynk con smartphone.

En la aplicación Blynk se puede construir la interfaz de usuario de acuerdo con sus necesidades, haciendo uso de widgets prehechos como botones e indicadores que simplemente se arrastra hacia la interfaz. La figura 2.37 muestra un ejemplo de cómo se construye la interfaz.

Por otro lado, el Blynk server es open source así que puedes descargarlo e instalarlo en la infraestructura propia, sin embargo, también se puede utilizar Blynk cloud, que provee un nivel básico gratuito y niveles con mayor capacidad con costo. (Sabas, 2023)

Ilustración 2.37 aplicación Blynk en Smartphone

Capítulo 3. Desarrollo experimental

En este apartado se explicará cómo se ha desarrollado el sistema para el abastecimiento de agua de manera automática, el cual funciona de la siguiente manera: todo el tiempo el sistema revisa los niveles de agua del depósito (cisterna) y del flujo de agua que lo llena, así como del nivel del tinaco. Cuando el depósito aún no llega a su nivel máximo y hay flujo de agua, se activa una bomba presurizadora, con el afán de acelerar el proceso de llenado, y que al lograrse se apaga la presurizadora. Adicionalmente, se incorporó un sistema de cierre por flotador, lo que permite tener abierta todo el tiempo de llave de agua proveniente de la red de agua. Para el bombeo al tinaco (con bomba sumergible), el sistema permite mediante una aplicación (web y móvil) establecer el nivel para encendido y apagado de bomba, el cual se activa siempre y cuando exista agua disponible en el depósito. La aplicación también muestra el nivel del tinaco, el cual también se puede observar en el módulo primario (que se instala cerca del depósito), mediante indicadores LED. Otra funcionalidad que se puede establecer mediante la aplicación es la de configurar horarios de trabajo para la bomba de agua, con el afán de no perturbar las horas de sueño, por ejemplo.

Para su implementación se desarrollaron dos módulos, el denominado primario, que monitorea el nivel de agua del depósito, y gestiona el encendido de las bombas (presurizadora y sumergible), y el módulo secundario, se monta en el tinaco, y realiza el monitoreo de nivel de agua del mismo. Para la explicación del desarrollo del sistema, se hará de manera independiente para cada módulo, y en ambos casos se expondrá su diagrama de bloques, diagrama esquemático, diagrama de flujo, codificación, tarjeta PCB y resultado final de los módulos.

3.1 Módulo primario

Este módulo es el encargado de monitorear el flujo de agua que va de la llave conectada a la red pública y que llega a la cisterna, también controla el encendido de la bomba sumergible para llenar al tinaco.

3.1.1 Diagrama de bloques

La ilustración 3.1, representa los bloques que componen el módulo primario, indicando sus componentes más importantes.

Donde:

- Sensor de nivel: Está compuesto por tres electrodos, uno funciona como referencia, otro indica el nivel considerado bajo, el cual sirve para apagar la bomba sumergible y evitar dañarla, y el otro electrodo es para determinar un nivel alto de agua, que se usa para apagar la bomba presurizadora. Además de los electrodos, esta etapa incorpora un circuito integrado LM2003, el cual contiene transistores en Darlington que sirve para energizar a los electrodos y como método de acoplamiento a la etapa de control.
- Sensor flujo: Se utilizó un sensor modelo FS300A, cuenta con capacidad de 60 L/m de flujo, tienen conectores de ¾ de pulgada, y tiene salida digital por efecto hall, para

cuantificar el flujo de agua, se hace un conteo de pulsos, y sobre ello se realizan la toma de decisión para encendido/apagado de bomba presurizadora.

- RTC: Se utilizó un Reloj de tiempo real modelo DS3231, este módulo es el encargado de activar los módulos primario y secundario de acuerdo en el horario preestablecido por el usuario.
- RS485: el módulo que se uso es el max485 rs485 el cual se configuro para trabajar como receptor del módulo secundario mediante el protocolo de comunicación RS485 simplex.
- Microcontrolador: El microcontrolador ESP32 es el encargado de la etapa de control, las etapas de potencia y reflejar los valores en la aplicación remota.
- Etapa de potencia: Para lograr la activación de la bomba presurizadora y sumergible se utilizaron dos relevadores de estado sólido de 50 A, los pines de la ESP32 solo entregan 3.3v lo cual no es suficiente para lograr activar el relevador, para ello colocamos un transistor 2n2222 configurado como interruptor.
- Bomba presurizadora: Actuador encargado del llenado de la cisterna cuando el flujo de agua de la red pública sea débil.
- Bomba sumergible: Actuador que suministra agua cuando el tinaco lo necesite de acuerdo con lo requerido por el usuario.
- Aplicación remota: Se utilizó la plataforma Blynk, en el sitio web se pueden visualizar y modificar valores requeridos mientras que en la aplicación de celular es encargado de mostrar los valores del electronivel, y los datos recibidos del microcontrolador secundario.

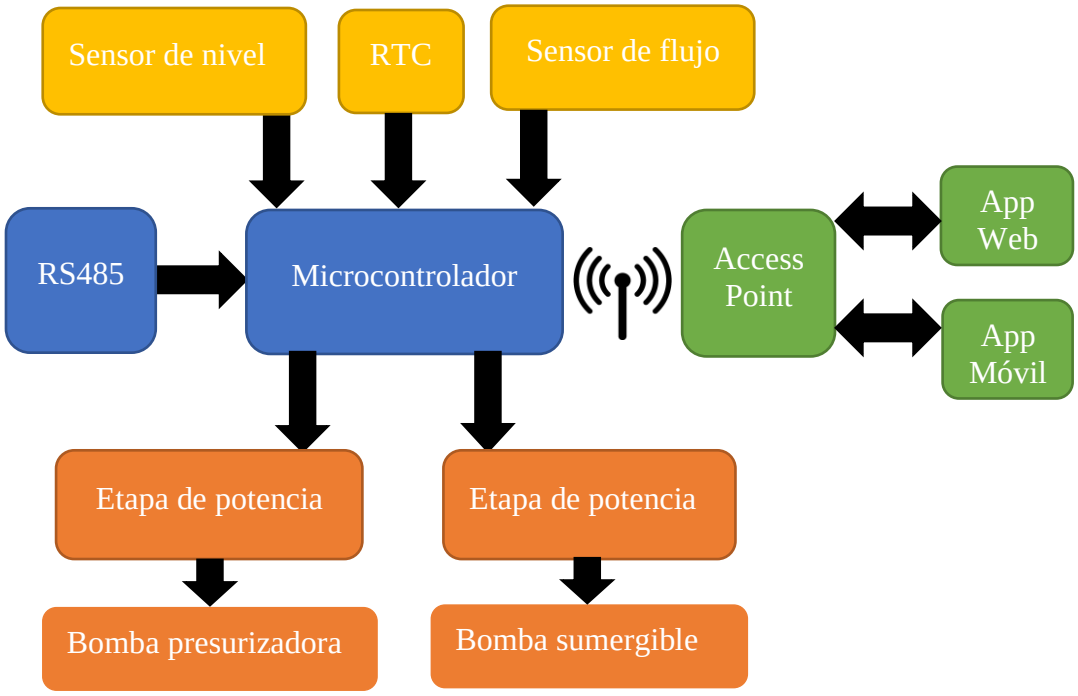


Ilustración 3.1 Diagrama de Bloques módulo del módulo primario.

3.1.2 Diagrama esquemático

En la ilustración 3.2 se muestra el esquemático que se realizó para este módulo. Los componentes que se utilizaron para formar el circuito fueron los siguientes:

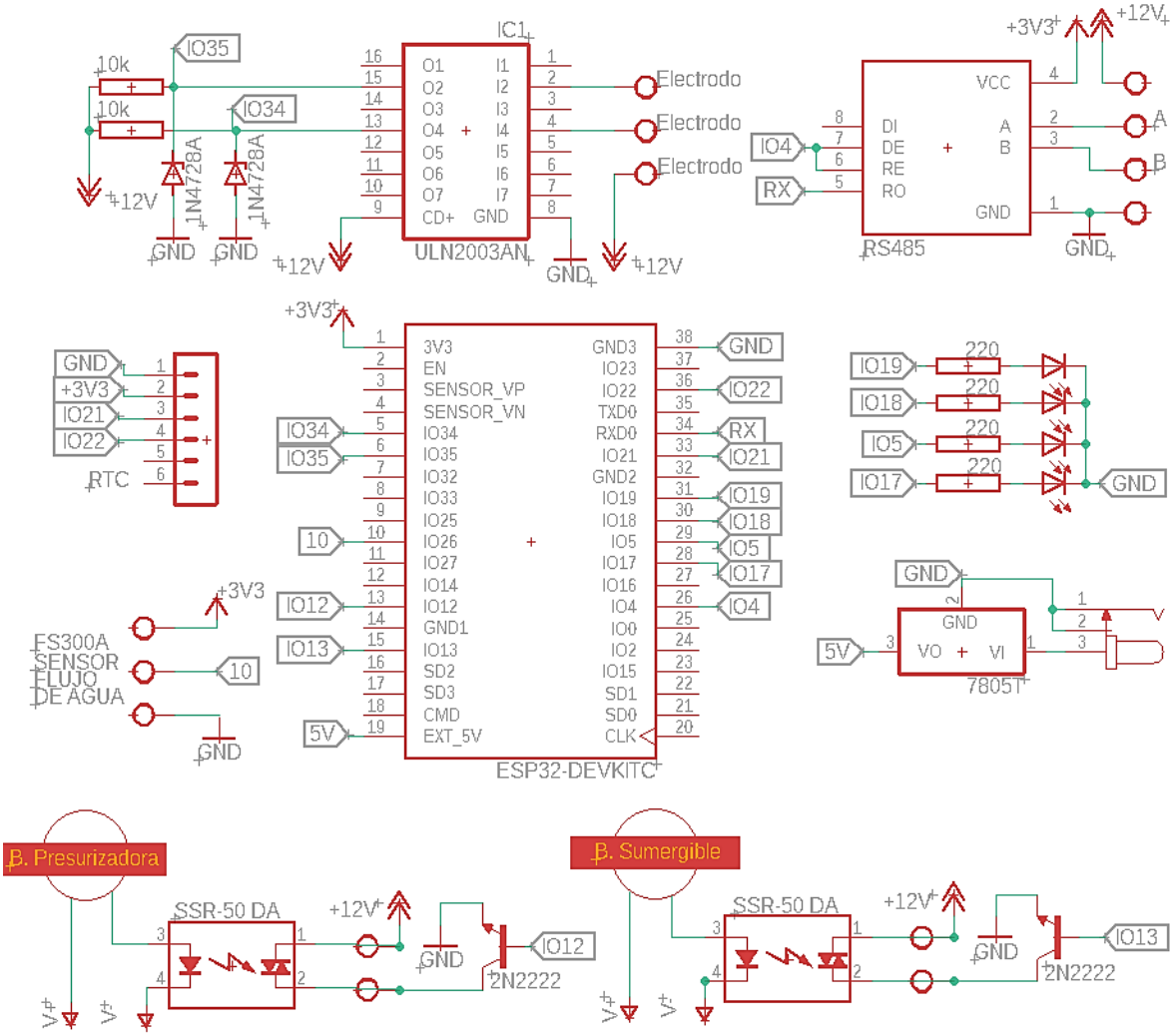


Ilustración 3.2 Diagrama de esquemático del módulo primario.

- 1 Esp32 38 Pin Devkit.
- 3 Electrodos para electronivel.
- 1 Sensor flujo de agua FS300A.
- 2 Transistor 2N2222.
- 1 ULN2003.
- 1 L7805.
- 1 Módulo Interfaz Rs-485 Max485.
- 1 Reloj de tiempo real Ds3231.

- 2 Resistencias 10k.
- 2 diodos Zener 1N4728A.
- 4 Resistencias 220.
- 4 luces led.
- 1 Molex 4 pines.
- 2 Molex 3 pines.
- 2 Molex 2 pines.
- 2 Relevadores de estado sólido 50 A.
- 1 Bomba presurizadora.
- 1 Bomba sumergible.

El microcontrolador ESP32 es el encargado de realizar las operaciones de control, puesto que tiene los elementos necesarios para el circuito y conexión WI-FI. Para el suministro de energía necesita una fuente de energía de 5 V, se le colocó un regulador LM7805 para obtener una tensión de 5 V, puesto que para el diseño del circuito se utilizó una fuente de 12 V, ya que algunos componentes requieren ese nivel.

En la ilustración 3.3 se muestra la etapa de medición del nivel del depósito, para ello se optó por un ULN2003AN, un circuito integrado con transistores en Darlington que es utilizado para la medición de nivel de agua, por medio de electrodos. Dado que la señal que entrega es incompatible con la ESP32, se optó por colocar en un Diodo Zener como regulador de voltaje para lograr tener 3.3 V y no dañar la ESP32.

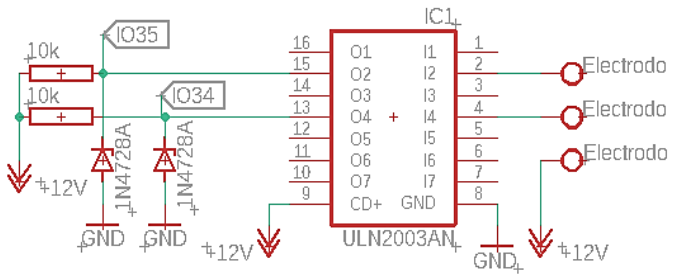


Ilustración 3.3 Conexión de electrodos.

En la ilustración 3.4 se muestra el módulo RS485 se conectó el pin 6 y 7 con el pin 4 de la ESP32, el cual se configuró para que funcione como tierra, esto se lleva a cabo para qué el módulo funcione como receptor, el pin RO es el encargado de transmitir la información recibida, provenientes de los pines A y B.

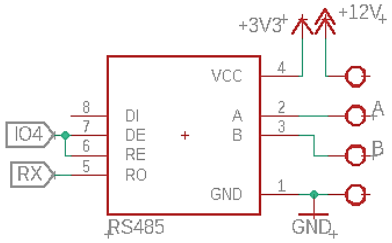


Ilustración 3.4 Conexión módulo RS485.

3.1.3 Diagrama de flujo

A continuación, se muestra en la ilustración 3.5 el diagrama de flujo que describe la programación del módulo del encendido/apagado de la bomba presurizadora:

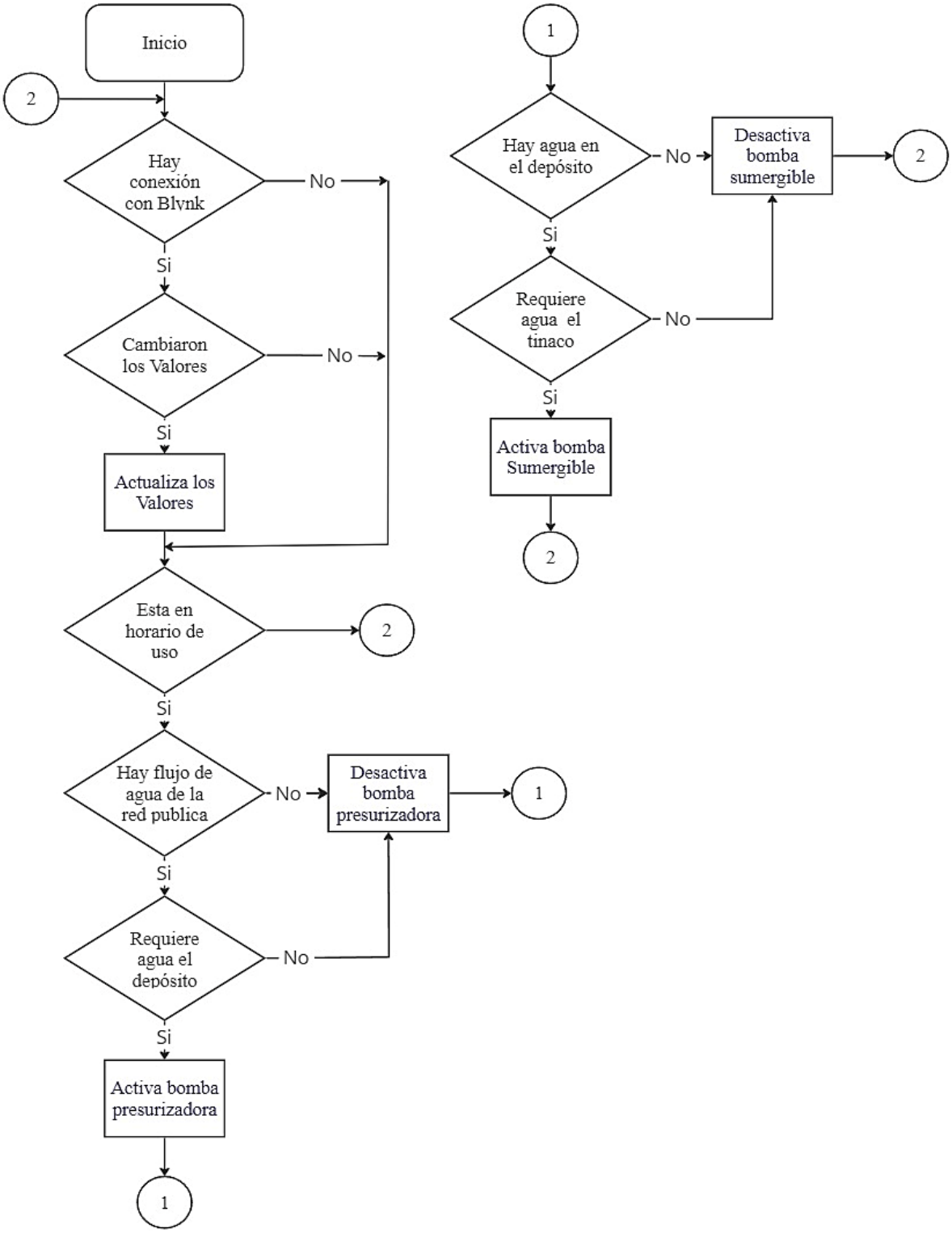


Figura 3.5. Diagrama de flujo principal.

Una vez definido el diagrama de flujo, se realizó la programacion del módulo, como primer paso se incluyeron las bibliotecas de Blynk en las líneas de código de 1 al 9 corrresponden a la inicializacion básica del programa para su comunicación con las bibliotecas.

En la líneas de código del 1-6 se usaron para la comunicación de la ESP32 con Blynk donde en la primera línea de la programación se tiene un código id para tener acceso, la línea 2 es el nombre del programa que se le dio en Blynk.

```
1     #define BLYNK_TEMPLATE_ID "TMPLrNJsLD3Q"
2     #define BLYNK_DEVICE_NAME "Distribución de agua"
3     #define BLYNK_FIRMWARE_VERSION        "0.1.0"
4     #define BLYNK_PRINT Serial
5     #define APP_DEBUG
6     #define USE_ESP32S2_DEV_KIT       //Nombre de la placa que se
ocupará
```

Para la solución del diagrama de flujo con respecto a la conexión Blynk se solucionó con la biblioteca de "BlynkEdgent.h", la biblioteca se encarga de la comunicacion de los datos enviados y recibidos por Blynk, para guardar los datos recibidos, se usa una función de la biblioteca “preferences.h”

```
8     #include "BlynkEdgent.h"
9     #include <Preferences.h>        //Librería par emplear como
memoria.
```

Para activar los horarios de uso se ocupó un reloj de tiempo real DS3231, para facilitar su uso se incluyeron las siguientes bibliotecas:

```
10    #include <RTClib.h>
11    #include <Wire.h>
12    #include <time.h>
```

Se ocuparon dos pines virtuales (V7, V8) para que el usuario a través de la aplicación de Blynk defina los horarios para que el sistema no sea activado.

```
128   BLYNK_WRITE(V7)                          //Pin Virtual de Blynk
129   {
130     apagado = param.asInt();               //Recibe el valor de
Blynk
131     preference.putUInt("apagado", apagado); //Guarda el valor
obtenido en la variable apagado
132   }
133
134   BLYNK_WRITE(V8)                          //Pin Virtual de Blynk
135   {
136     encendido = param.asInt();             //Recibe el valor de
Blynk
137     preference.putUInt("encendido", encendido); //Guarda el valor
obtenido en la variable encendido
138   }
```

Para la toma de decisión de activar al sistema en horario de uso, se realizó una toma de decisión (línea 105) que compara la información del RTC y las configuraciones recibidas desde la aplicación de Blynk. De estar fuera de uso, el sistema no realiza acción alguna, de estar en horario de uso, se colocó una restricción para solo trabajar en intervalos de cada cinco segundos, ya que no es necesario o incluso conveniente hacer un monitoreo cada fracción de segundo, debido a las limitaciones del sensor ultrasónico. Y lo que se hace cada que se cumple en tiempo indicado son varias acciones, que son invocadas con las funciones "programadeposito()" y "programatinaco()", las cuales serán explicadas más adelante.

```
105   if(hora >= encendido && hora < apagado)         //Intervalo para el
                                                  funcionamiento del sistema
106   {
107    if((millis()-tempRef1)>5100) //Temporizador 5s
108     {
109       tempRef1=millis();                      //se actualiza cada 5.1s
110       programadeposito();                     //Subrutina del programa
115       programatinaco();                       //Subrutina del programa
                                                                Principal
111     }
117   }
```

La función "programadeposito()", tiene como finalidad verificar el nivel de agua en el depósito, saber si hay flujo de agua proveniente de la red pública, y con esa información tomar decisiones, que incluyen activar o desactivar la bomba presurizadora y enviar ese condición a la aplicación de Blynk.
Es importante señalar que se colocó una llave con flotador en el depósito, y que al llenarse cierra el paso, y que ademas o por cuestiones de seguridad se pusieron sensores de nivel en el depósito para saber si esta lleno o vacío, se utilizan dos pines de la ESP32 para dicho monitoreo. Si se detecta nivel máximo de agua, o que no hay flujo de agua de la red pública, se apaga la bomba presurizadora, de lo contrario se activa para acelerar el llenado del depósito. Para saber si hay flujo de agua se hace el monitoreo mediante la función "contarpulsos()" que se explicará más adelante. Ahora se presentan las líneas de código de la función "programadeposito()":

```
144   void programadeposito()
145   {
146     lleno = digitalRead(34);                  //Nivel del electrodo
147     float frecuencia = ObtenerFrecuencia();  //obtenemos la
frecuencia de los pulsos en Hz
148     if (frecuencia >= 20) //Toma de decisión para evaluar si hay
flujo de agua
149     {
150       if (lleno == 0) //Toma de decisión si necesita agua el deposito
151       {
152         Blynk.virtualWrite(V6, 1);       //Enciende un led en Blynk
      para avisar que se encuentra encendida la bomba
```

```
153         digitalWrite(13, HIGH);                    //Enciende la bomba
154       }
155       else                  //si no se cumple el segundo if entonces:
156       {
157         Blynk.virtualWrite(V6, 0); //Mantiene el led de Blynk apagado
158         digitalWrite(13, LOW);           // Mantiene la bomba apagada
159       }
161     }
162     else                      //si no se cumple el primer if entonces:
163     {
164       Blynk.virtualWrite(V6, 0);   //Mantiene el led de Blynk apagado
165       digitalWrite(13, LOW);             // Mantiene la bomba apagada
166     }
167   }
```

La funcion “programatinaco()” tiene como finalidad verificar el nivel de agua en el depositó, asi como los niveles de agua del tinaco, y con esa información tomar decisiones, que incluyen activar o desactivar la bomba sumergible y enviar esa condición a la aplicación de Blynk.
Para observar los niveles de agua en el depósito se colocarón sensores de nivel utilizando dos pines de la ESP32 para saber si el deposito esta lleno o vacío, mientras que para tener los niveles de agua del tinaco se puso un sensor ultrasónico.
Si se detecta nivel mínimo de agua en el depósito y el nivel de agua del tinaco esta en el rango puesto por el usuario, se enciende la bomba sumergible; en caso contrario, se apaga.
Para visualizar el nivel del tinaco se le envían los datos leídos por el sensor ultrasónico a la aplicación de Blynk, en caso de no tener acceso a la aplicación se le colocaron luces LED al sistema a modo de inficación de nivel de agua. Ahora se presentan las líneas de código de la función “programatinaco()”:

```
170   void programatinaco()
171   {
172     mitad = digitalRead(35);          //Nivel del  segundo electrodo
173     int med = Serial.parseInt();      //Lectura del sensor proveniente
del módulo secundario
174     float porcentaje = map(med, 20, 130, 100, 0); //Transformación de
valores para visualizarse en porcentaje
175     Blynk.virtualWrite(V0, porcentaje); //Transmisión del porcentaje
para visualizarse en Blynk
176
177     if (mitad == 0) //Toma de decisión  para evaluar si hay agua en
el deposito
178     {
179       Blynk.virtualWrite(V4, 0);//Mantiene el led apagado
correspondiente a la bomba sumergible en Blynk
180       digitalWrite(12, LOW);   //Mantiene la bomba sumergible apagada
181     }
```

```
182
183     else
184     {
185       if (porcentaje <= (referencia - tolerancia)) //Toma de decisión
para evaluar si el valor porcentaje se encuentra por debajo del nivel de
referencia-tolerancia
186       {
187         Blynk.virtualWrite(V4, 1);  );//Mantiene el led encendido de
la bomba sumergible en Blynk
188         digitalWrite(12, HIGH); //Mantiene la bomba sumergible
encendido
189       }
190
191       else(porcentaje >= (referencia + tolerancia)) //cuando el
porcentaje es superior a la referencia + tolerancia entonces:
192       {
193         Blynk.virtualWrite(V4, 0); );//Mantiene el led apagado de la
bomba sumergible en Blynk
194         digitalWrite(12, LOW); //Mantiene la bomba sumergible apagada
195       }
196     }
220     int nivel = map(med, 20, 130, 3, 0);
220     switch(nivel)
220      {
220        case 0:
220        digitalWrite(17, 0);
220        digitalWrite(5, 0);
220        digitalWrite(18, 0);
220        break;
220      case 1:
220        digitalWrite(17, 1);
220        digitalWrite(5, 0);
220        digitalWrite(18, 0);
220        break;
220      case 2:
220        digitalWrite(17, 1);
220        digitalWrite(5, 1);
220        digitalWrite(18, 0);
220        break;
220      case 3:
220        digitalWrite(17, 1);
220        digitalWrite(5, 1);
220        digitalWrite(18, 1);
214      }
215  }
```

La códificación completa del diagrama de flujo se presenta en el anexo.

3.1.4 Fabricación de la placa PCB

Para el diseño de la PCB se utilizó el software Eagle, se creó una PCB de tal modo que fuera una placa de 95 x 95 mm, se configuró en el software que las áreas sin pistas funcionaran como el común de la placa (tierra) teniendo como diseño final la figura 3.6.

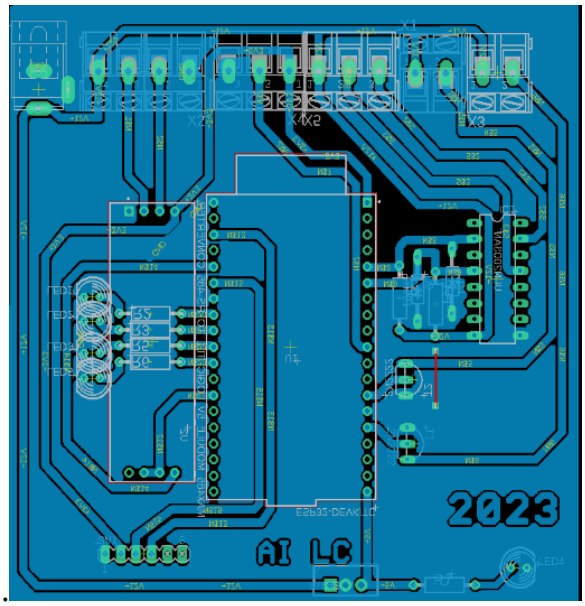


Ilustración 3.6 Diseño PCB módulo primario.

Una vez concluido el diseño se procedió a exportar el archivo Gerber RS-274x para generar la PCB en una placa fenólica en una CNC por medio de Flatcam. Posteriormente se procedió a convertir el archivo Gerber a código G, se hicieron dos archivos: uno para el trazado de las pistas, (ilustración 3.7), mientras que el segundo es para el barrenado de los orificios (figura 3.8).

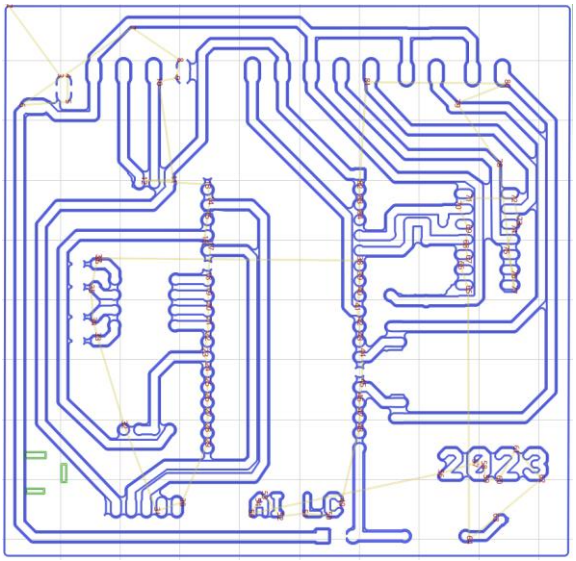


Ilustración 3.7 Código g del trazado de pistas.

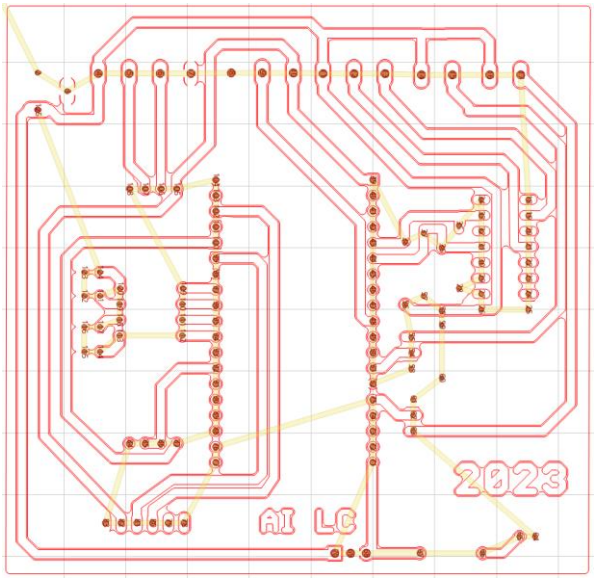


Ilustración 3.8 Código g del barrenado de orificios.

Como último paso, se soldaron los componentes, para la protección de los circuitos integrados, y para facilitar su remplazo, se colocaron zócalos, así como un disipador de calor para el regulador L7805.

En la ilustración 3.9 se muestra la PCB en donde se puede observar en la parte posterior los componentes ya soldados.

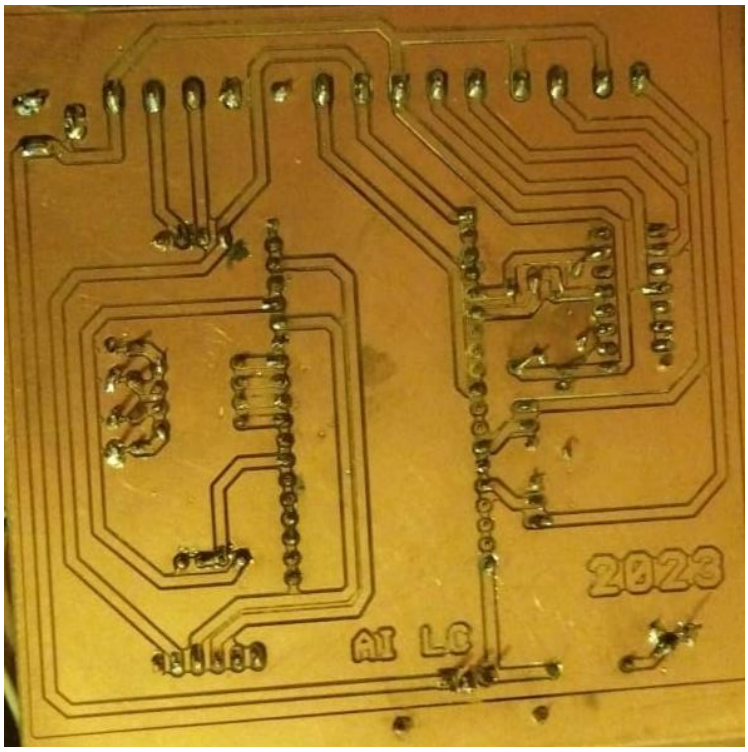


Ilustración 3.9 PCB Módulo primario.

En la ilustración 3.10 se muestra la parte superior.

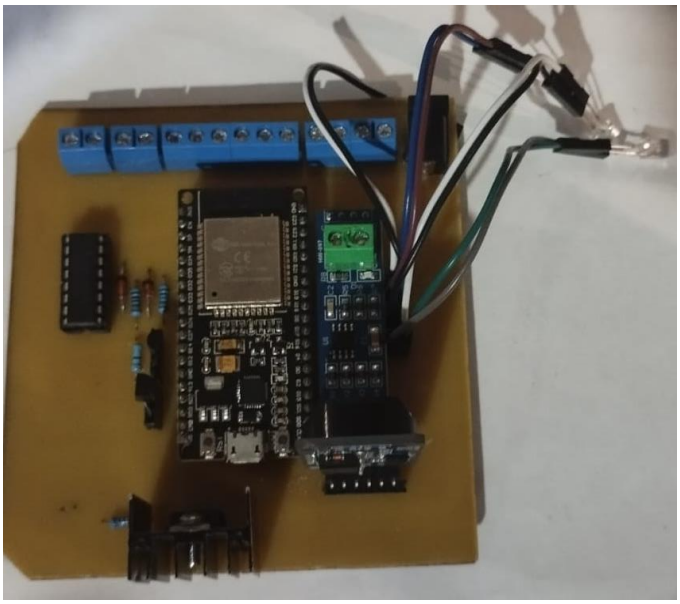


Ilustración 3.10 PCB parte superior del módulo primario.

3.1.5 Modelo 3D y real

Como parte del módulo primario se diseñó su respectiva carcasa 3D, se utilizó el programa Thinkercad para el modelado, el cual conllevo a hacer dos piezas para la carcasa.

Primero se realizó la base, esta fue hecha a partir de un cuadrado de 110 mm x 110 mm de altura de 60 mm, en las esquinas se le dejó columnas con los agujeros para la entrada de tornillos de 2 mm, tiene cuatro agujeros de 10 mm para los cables de los sensores, y otros dos para la conexión proveniente de los relevadores, como se muestra en la ilustración 3.11.

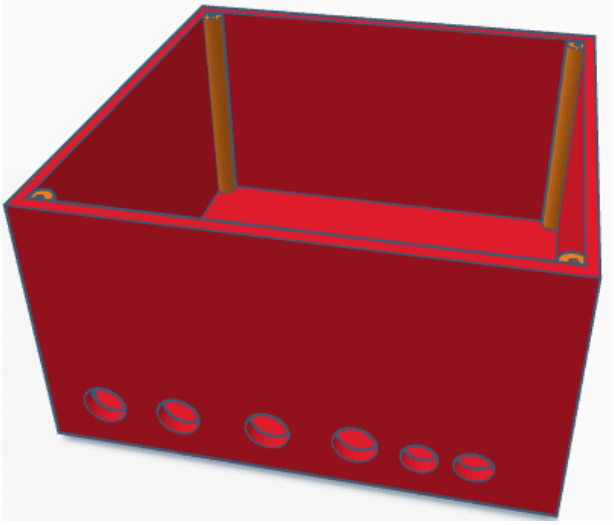


Ilustración 3.11 Caja módulo primario.

Para la tapa se realizó de la misma medida que la pieza de la caja con un grosor de 2 mm, se le colocó una imagen que representa un tinaco el cual tiene 3 niveles, en cada nivel se le

agregó un agujero para posteriormente colocarle un LED, como se muestra en la ilustración 3.12.

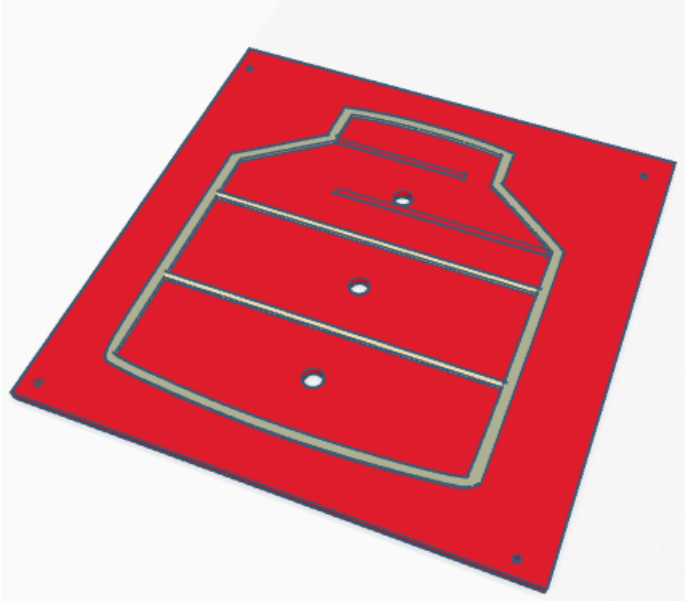


Ilustración 3.12 Tapa de la caja del módulo primario.

Ya terminadas las piezas modeladas, se procedió a exportar en formato “stl” para posteriormente imprimirlas; después, con las piezas ya listas se ensambló la caja con la PCB, como se muestra en la ilustración 3.13.

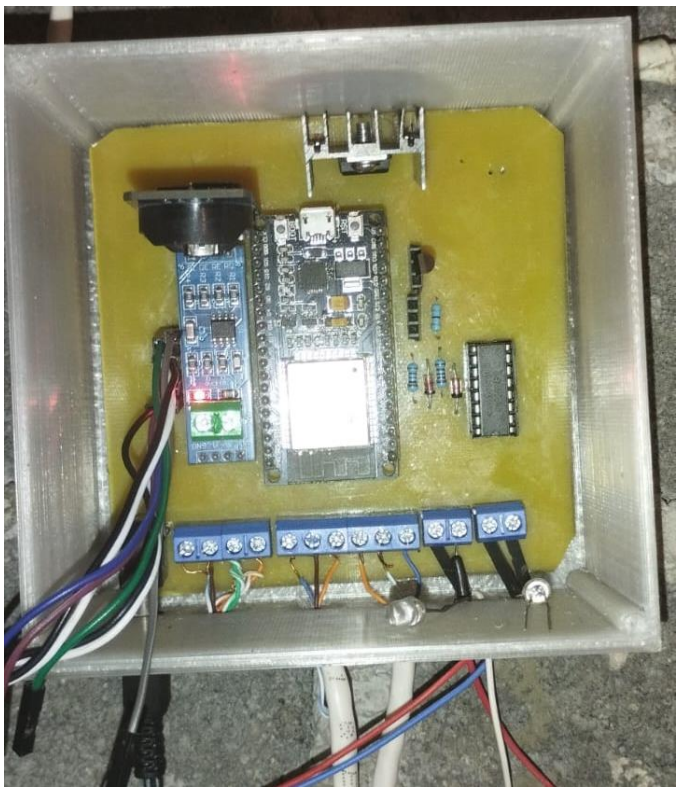


Ilustración 3.13 Ensamble de la caja con la PCB y sus respectivas conexiones.

Con los componentes adheridos, se ensambló la tapa con la caja, sujetándola con tornillos por medio de las columnas que se encuentran en las esquinas. En la ilustración 3.14 se muestra el modelo final del módulo primario.

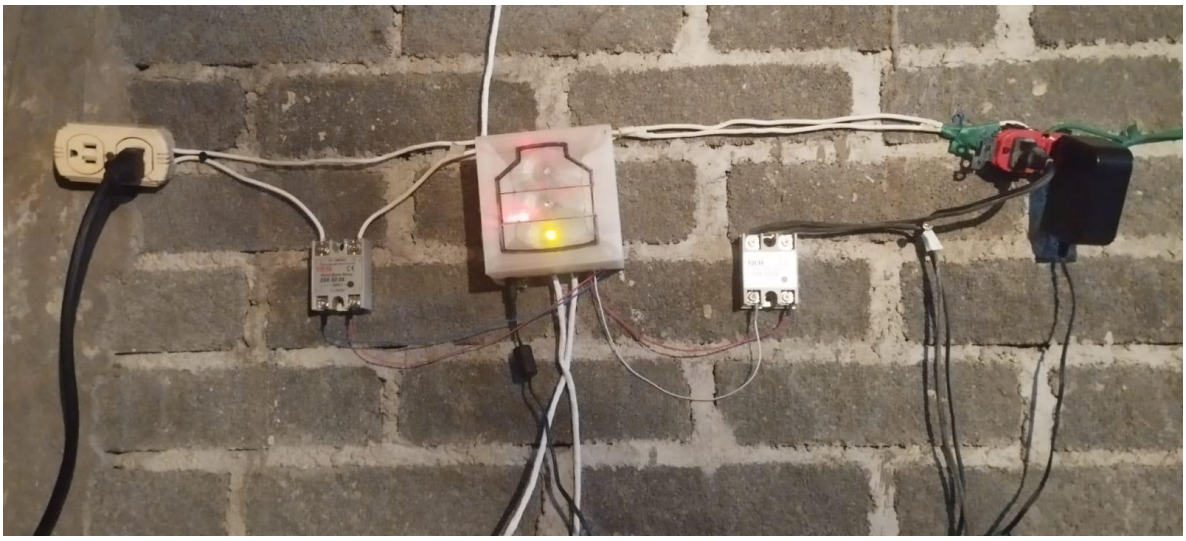


Ilustración 3.14 conexiones del módulo primario.

Como se muestra en la ilustración 3.14, las conexiones realizadas en el módulo primario son las siguientes:

- El cable negro es el encargado de suministrar la alimentación del circuito, proporcionando 12 V de energía.
- El siguiente cable, de color blanco, está destinado para la comunicación con el módulo secundario.
- El tercer cable está conectado con los electrodos de la cisterna, lo que permite medir el nivel de agua en el recipiente.
- El último cable blanco está conectado con el sensor de flujo de agua, que monitorea el caudal de agua.
- El enchufe blanco de la izquierda se utiliza para alimentar la bomba sumergible, y esta se activa a través del relevador situado a la izquierda del módulo primario.
- En cuanto al relevador ubicado a la derecha que se encuentra conectado al enchufe naranja, su función es energizar la bomba presurizadora.

3.2 Módulo secundario

Este módulo es el encargado de monitorear y transmitir el nivel del agua en el tinaco al módulo primario.

3.2.1 Diagrama de bloques

La ilustración 3.15 representa los bloques que componen el módulo secundario, indicando sus componentes que lo conforman.

Ilustración 3.15 Diagrama de Bloques módulo del módulo secundario.

Donde:

- Microcontrolador: El microcontrolador ESP32 se encarga de obtener y transmitir los valores del sensor ultrasónico.
- Sensor ultrasónico: se utilizó el módulo JSN-SRO4T aprueba de agua, el cual tiene un rango de medición de 20-600 cm.
- RS485: el módulo que se ocupo es el max485 rs485 similar al del módulo primario; sin embargo, se configuró para trabajar como transmisor al módulo secundario mediante el protocolo de comunicación RS485 simplex.

3.2.2 Diagrama esquemático

En la ilustración 3.16 se muestra el esquemático que se realizó para este módulo secundario.

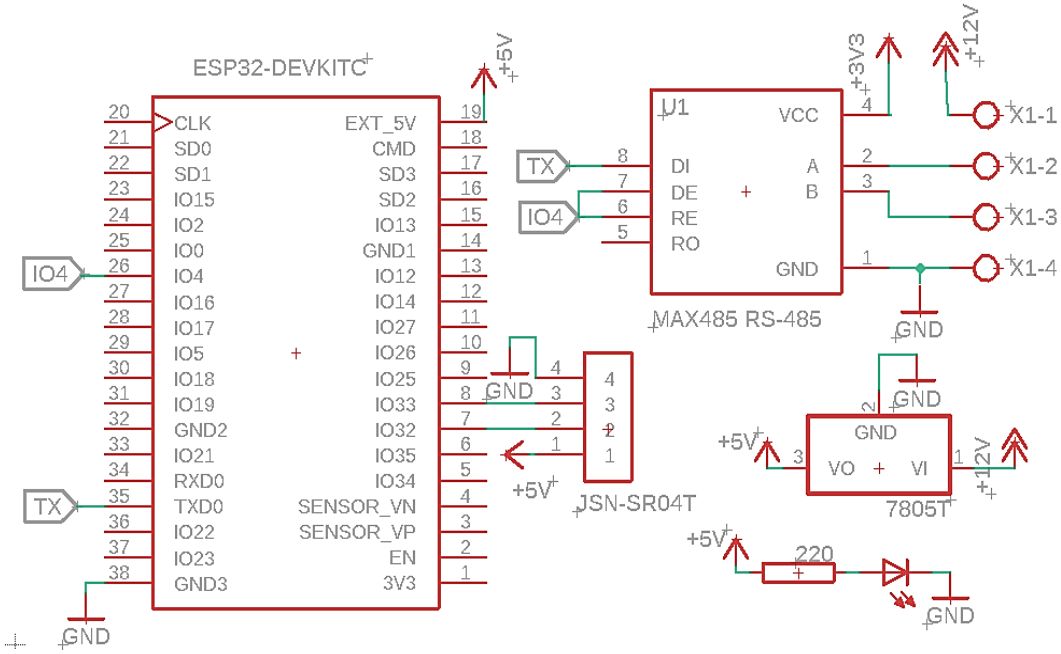


Ilustración 3.16 Diagrama de esquemático del módulo secundario.

Los componentes que se utilizaron para formar el circuito fueron los siguientes:

- 1 Esp32 38 Pin Devkit.
- 1 L7805.
- 1 Resistencia 220.
- 1 LED.
- 1 Módulo Rs-485 Max485.
- 1 Sensor ultrasónico JSN-SR04T

El microcontrolador ESP32 es el encargado de recibir los datos del sensor ultrasónico JSN-SR04T y convertirlos en la unidad de cm, para posteriormente enviar los valores al módulo primario.

En el módulo RS485 se conectó el pin 6 y 7 con el pin 4 de la ESP32, el cual se configuró para que funcione como emisor, esto se lleva a cabo para qué el módulo funcione como transmisor, el pin DI es el encargado de recibir la información, y transmitirla por los pines A y B. como se muestra en la ilustracion 3.17

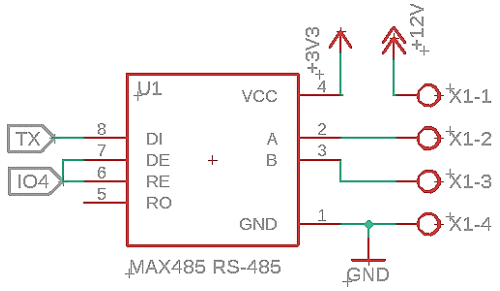


Ilustración 3.17 Conexión módulo RS485 secundario.

La alimentación es a través del módulo primario recibiendo 12 V, puesto que para este diseño solo se necesita el voltaje para el funcionamiento de la ESP32, se le colocó un regulador LM7805 para obtener una tensión de 5 V. Para saber si al circuito se le agrego un LED, tal como se muestra en la ilustración 3.18.

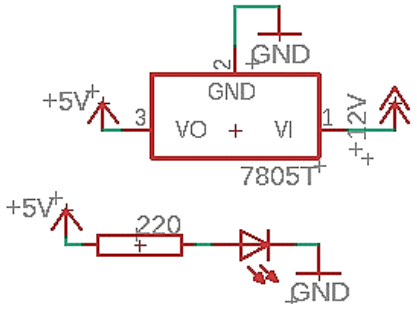


Ilustración 3.18 Conexión fuente de voltaje.

3.2.3 Diagrama de flujo

A continuación, se muestra en la ilustración 3.19 el diagrama de flujo que describe la programación del módulo del sensor ultrasónico:

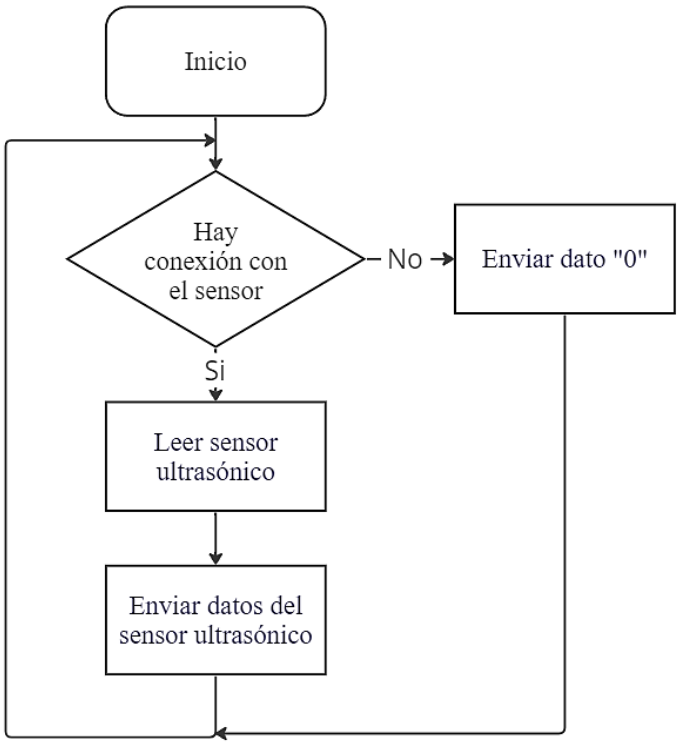


Ilustración 3.19 Diagrama de flujo del programa secundario.

Una vez definido el diagrama de flujo, se realizó la programacion del módulo, para la solución del diagrama de flujo se basa en las siguientes líneas de código, donde se recopilan y envían los datos cada 4 segundos, cuando el sensor está fallando, el programa manda un 0, por no tener ningún dato.

```
16    if((millis()-tempRef1)>4000)
17      {
18        tempRef1=millis();
19      digitalWrite(trigPin, LOW);          // Borra la condición trigPin
20      delayMicroseconds(2);
21      digitalWrite(trigPin, HIGH);         // Establece la trigPin HIGH
       (ACTIVA) durante 10 microsegundos
22      delayMicroseconds(10);
23      digitalWrite(echoPin, LOW);          // Lee el echoPin, devuelve
el tiempo de viaje de la onda de sonido en microsegundos

25      duration = pulseIn(echoPin, HIGH);   // Calculando la distancia
26      distance = duration * 0.034 / 2;     // Velocidad de la onda de
sonido dividida por 2 (ida y vuelta)

28      Serial.println(distance);              // Manda el valor de la
distancia en el puerto serial
29      }
```

La códificación completa del diagrama de flujo se presenta en el anexo.

3.2.4 PCB

Para el diseño de la PCB del módulo secundario, se ordenaron los componentes para que el tamaño de la placa quedara de 85 mm x 85 mm como se muestra en la ilustración 3.20.

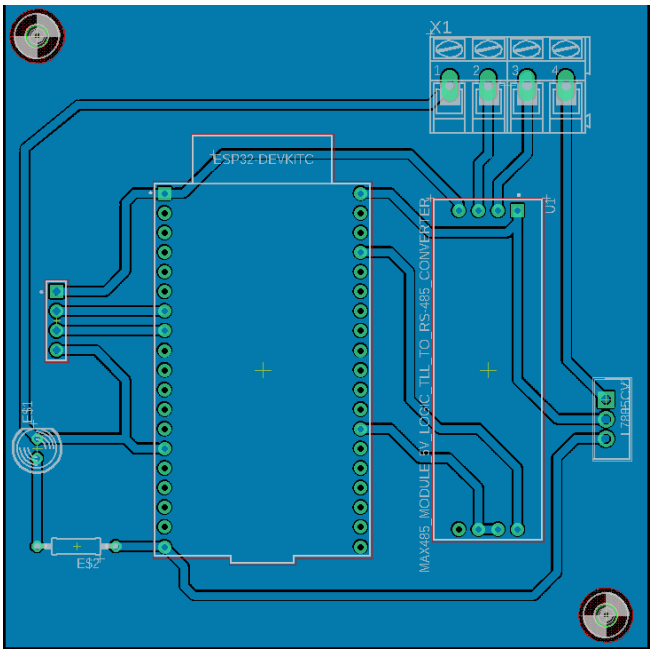


Ilustración 3.20 Diseño PCB en Eagle.

En la ilustración 3.21 se muestra la parte posterior de la PCB donde se observa la placa con sus componentes soldados, así como en las esquinas dos orificios, en las cuales van dos tornillos para su fijación con la caja que lo resguarda.

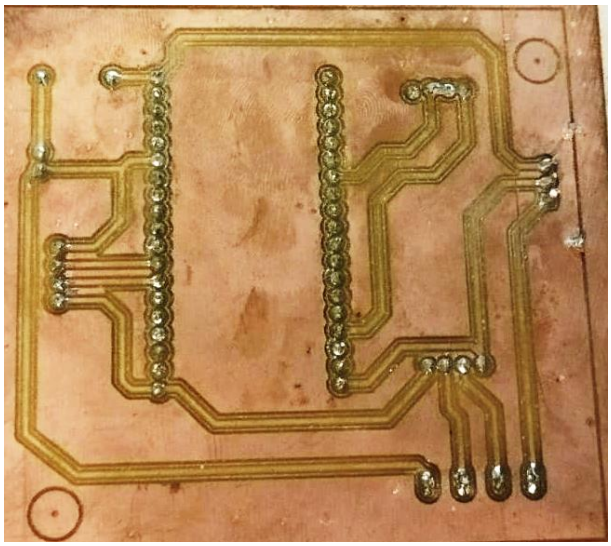


Ilustración 3.21 PCB Módulo secundario.

En la ilustración 3.22 se muestra la parte superior, donde están los componentes, se colocaron zócalos para cambiarlos cuando estos requieran ser reemplazados.

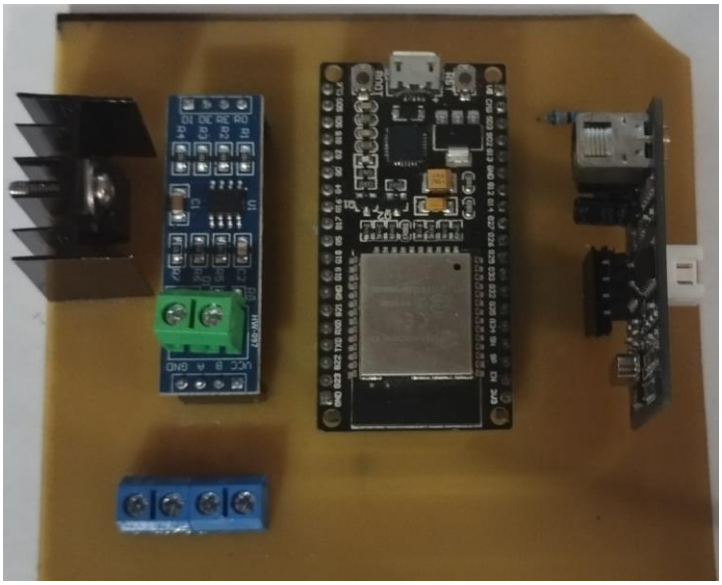


Ilustración 3.22 PCB parte frontal módulo secundario.

3.2.5 Modelo 3D y real

Al igual que el módulo anterior este modelo 3D fue hecho con la aplicación en línea Thinkercad.

Como primero se realizó la base, esta fue hecha a partir de un cuadrado de 110 mm x 110 mm, de altura de 60 mm. En las esquinas se le dejaron columnas con agujeros para la entrada de tornillos de 2 mm, en dos costados, tiene agujeros de 13 mm, uno para el cable de la transmisión de datos y alimentación, y otro para el cable proveniente del sensor ultrasónico, como se muestra en la ilustración 3.23.

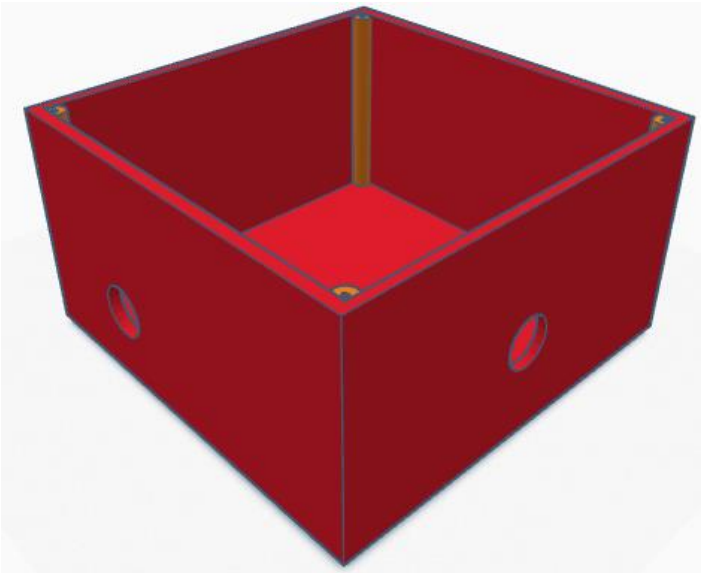


Ilustración 3.23 Caja módulo secundario.

Para la tapa se realizó de la misma medida que la pieza de la caja, tomando las dimensiones de donde se colocan los tornillos, el grosor de la tapa, así como de las paredes de la caja es de 2 mm, esto se muestra en la ilustración 3.24.

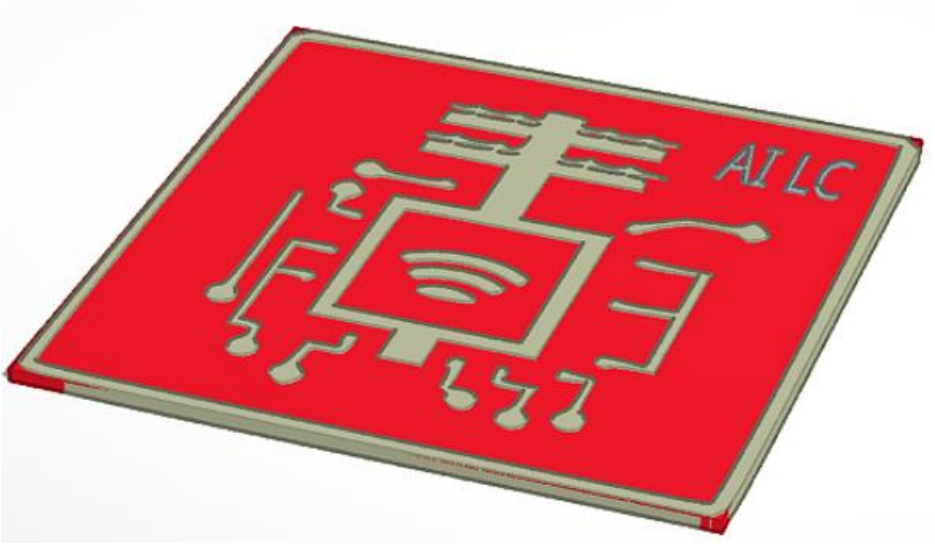


Ilustración 3.24 Tapa módulo secundario.

En la ilustración 3.25 se muestra ya colocada en la azotea junto al tinaco el cual estará monitoreando, el cacle negro que sale de la parte izquierda de la caja va dirigido hacia el sensor ultrasónico, mientras que el blanco es el cable encargado de enviar los datos obtenidos, así como de la alimentación del módulo secundario.

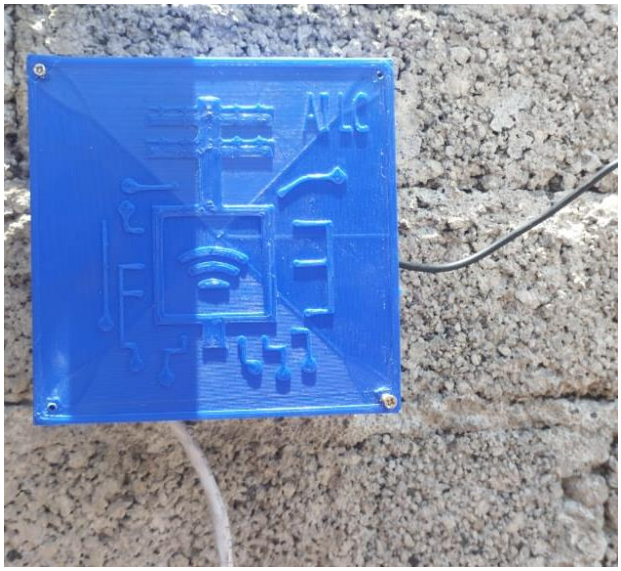


Ilustración 3.25 conexión módulo secundario.

En la ilustración 3.26 se puede apreciar la colocación del sensor ultrasónico dentro del tinaco. Se ha instalado cuidadosamente, teniendo en cuenta la medición mínima de 20 cm para futuras mediciones. Para asegurar su posición y evitar que se mueva de su lugar, se ha amarrado con su cable.

Ilustración 3.26 Colocación del sensor ultrasónico.

Una instalación adecuada y segura del sensor ultrasónico contribuye significativamente al correcto funcionamiento del sistema de monitoreo y control del nivel de agua. Al mantener el sensor en su posición óptima, se asegura que las mediciones sean consistentes y confiables, lo que es fundamental para el buen desempeño y eficiencia del sistema en general.

3.3 Aplicación web y móvil en Blynk

La aplicación se desarrolló en Blynk

3.3.1 Aplicación web

En la aplicación web, se muestra la pantalla inicial (ilustración 3.27) la cual contiene un indicador del porcentaje que hay en el nivel de agua en el tinaco, así como dos visualizadores que muestran el estado de las bombas para el depósito y el tinaco.

Se colocaron dos barras deslizantes para que el usuario establezca los niveles de agua para activar/desactivar la bomba del tinaco, un valor es para un punto medio deseado, y el otro es un margen de trabajo; es decir, se puede establecer un valor medio del 60% con un umbral del 20%, esto implica que la bomba se encenderá al estar por debajo del 40% y se apagará al rebasar; para evitar derrames de agua, se aplicó una restricción, la cual establece que la referencia más la tolerancia juntas no pueden rebasar más del 95 %.

De igual modo se le colocaron dos barras deslizantes para establecer valores en los que se desea que se active/desactive el sistema, es decir al valor de encendido se le pueden configurar a partir de las 5:00 y hasta las 10:00 hrs, mientras que la barra para el apagado se puede establecer desde las 20:00 y hasta las 23:00. Por ejemplo, se puede establecer que el sistema funcione desde las 5:00 hasta las 23:00, o que trabaje solo de 10:00 a 20:00, con ello se logró que el sistema de bombeo no provoque ruido que altere el sueño de los residentes.

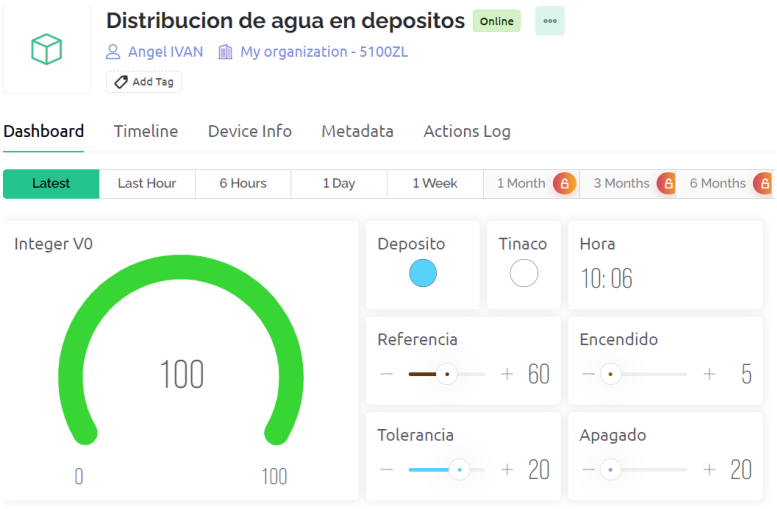


Ilustración 3.27 Blynk aplicación web.

3.3.2 Aplicación móvil

En la aplicación móvil mostrada en la ilustración 3.28, se presenta la pantalla principal, en ella hay un visualizador de porcentaje del nivel de agua en el tinaco, así como dos LED virtuales para mostrar el estado de trabajo de las bombas (encendidas o apagadas).

Se colocó un widget que muestra la hora del RTC de la ESP32, para conocer si tiene alguna imprecisión significativa, con respecta a la hora actual, de ser así el usuario puede hacer que esto se corrija simplemente reiniciando al módulo primario.

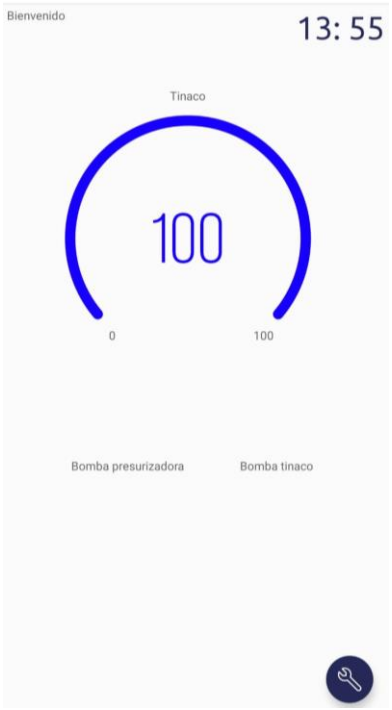


Ilustración 3.28 aplicación móvil Blynk.

Capítulo 4. Pruebas y resultados

En este apartado se describe el desarrollo de las pruebas realizadas para la creación del módulo primario y secundario, también se documentan algunas de las pruebas, y la solución cuando no se obtuvieron los resultados deseados.

4.1 Sensor ultrasónico

El experimento consiste en medir el nivel de agua utilizando el sensor ultrasónico JSN-SR04T. La prueba tiene como objetivo verificar la precisión y fiabilidad de las mediciones antes de implementar el sensor en el tinaco. Una vez que se ha confirmado la exactitud de las mediciones, se procederá a utilizar el sensor en el tinaco para determinar el porcentaje de llenado del mismo. De esta manera, se obtendrá información útil sobre la cantidad de agua almacenada en el tinaco, lo que facilitará la gestión y uso eficiente de este recurso.

La implementación del circuito se llevó a cabo mediante el siguiente programa en el cual se conectó el pin echo del sensor ultrasónico al pin 32 de la placa ESP32, mientras que el pin trig se conectó al pin 33 de la placa.

```
#define echoPin 32
#define trigPin 33
long duracion;  // variable para la duración del viaje de la onda de
sonido
int distancia;  // variable para la medida de la distancia
void setup()
{
  pinMode(trigPin, OUTPUT);  //Establece el trigPin como una salida
  pinMode(echoPin, INPUT);   //Establece el echoPin como una entrada
  Serial.begin(9600);
}
void loop()
{
  digitalWrite(trigPin, LOW);  // Borra la condición trigPin
  delayMicroseconds(2);
  digitalWrite(trigPin, HIGH);  // Establece la trigPin HIGH (ACTIVA)
durante 10 microsegundos
  delayMicroseconds(10);
  digitalWrite(echoPin, LOW);  // Lee el echoPin, devuelve el tiempo de
viaje de la onda de sonido en microsegundos
  duracion = pulseIn(echoPin, HIGH);  // Calculando la distancia
  distancia = duracion * 0.034 / 2;   // Velocidad de la onda de sonido
dividida por 2 (ida y vuelta)
  Serial.println(distancia);  // Manda el valor de la distancia en el
puerto serial
}
```

Se alambró el circuito y se comprobó su funcionamiento con un bote de 20 litros de agua. La ilustración 4.1 muestra al circuito en una tableta de prototipos.

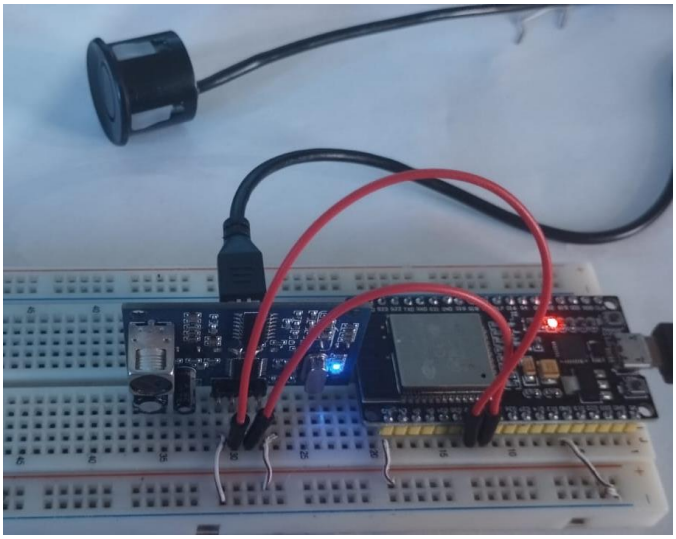


Ilustración 4.1 prueba sensor ultrasónico.

El problema que se encontró es que se debe dejar un intervalo entre cada lectura, ya que de realizarse muy rápido hay errores en la medición. La variabilidad en las mediciones del sensor ultrasónico puede ser resultado de realizar las lecturas muy rápidamente, lo cual puede generar errores debido a reflejos o rebotes del sonido ultrasónico en el objeto a medir o en el entorno. Estos reflejos causan valores muy dispares, incluso cuando no hay movimiento en el objeto bajo medición. Tal situación se puede observar en la ilustración 4.2, donde se obtienen lecturas con distancias indicadas de 18 cm y hasta 37 cm, lo que no debería suceder.

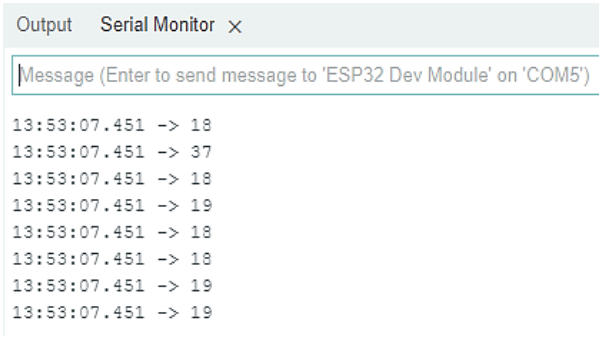


Ilustración 4.2 Lecturas sensor ultrasónico.

Para la solución se implementó un intervalo de cinco segundos entre cada lectura:

```
#define echoPin 32
#define trigPin 33

// Definición de variables
unsigned long tempRef1 = millis();
long duracion;  // variable para la duración del viaje de la onda de
sonido
int distancia;   // variable para la medida de la distancia
```

```
void setup()
{
  pinMode(trigPin, OUTPUT);  //Establece el trigPin como una salida
  pinMode(echoPin, INPUT);   //Establece el echoPin como una entrada
  Serial.begin(9600);
}

void loop()
{
  if ((millis() - tempRef1) > 4000)
  {
    tempRef1 = millis();
    digitalWrite(trigPin, LOW);  // Borra la condición trigPin
    delayMicroseconds(2);
    digitalWrite(trigPin, HIGH);  // Establece la trigPin HIGH (ACTIVA)
durante 10 microsegundos
    delayMicroseconds(10);
    digitalWrite(echoPin, LOW);  // Lee el echoPin, devuelve el tiempo de
viaje de la onda de sonido en microsegundos

    duracion = pulseIn(echoPin, HIGH);  // Calculando la distrancia
    distancia = duracion * 0.034 / 2;    // Velocidad de la onda de
sonido dividida por 2 (ida y vuelta)

    Serial.println(distancia);  // Muestra el valor de la distancia en el
puerto serial
  }
}
```

Como se muestra en la ilustración 4.3 al darle un espacio de tiempo entre cada medición, los valores de las lecturas se no presentaron las inconsistencias observadas con anterioridad.

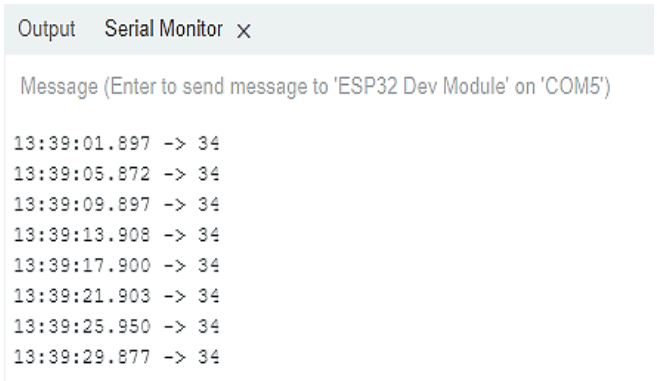


Ilustración 4.3 Lecturas sensor ultrasónico con intervalo de 4seg.

4.2 Nivel de agua con electrodos

La prueba realizada con los electrodos consiste en la verificación de nivel de agua, la cual fue realizada en un recipiente de 500 ml, para posteriormente llevarlo a cabo en un depósito.

Se identificó el inconveniente en el circuito, que consistía en que la fuente de alimentación era de 12 V, lo cual podría dañar la ESP32. Para evitar este problema, se decidió colocar un regulador de voltaje Zener de 3.3 V, como se muestra en la ilustración 4.4. De esta manera, se garantiza que la ESP32 reciba la tensión de alimentación adecuada y se previene cualquier daño potencial. Cómo se muestra en la ilustración 4.4.

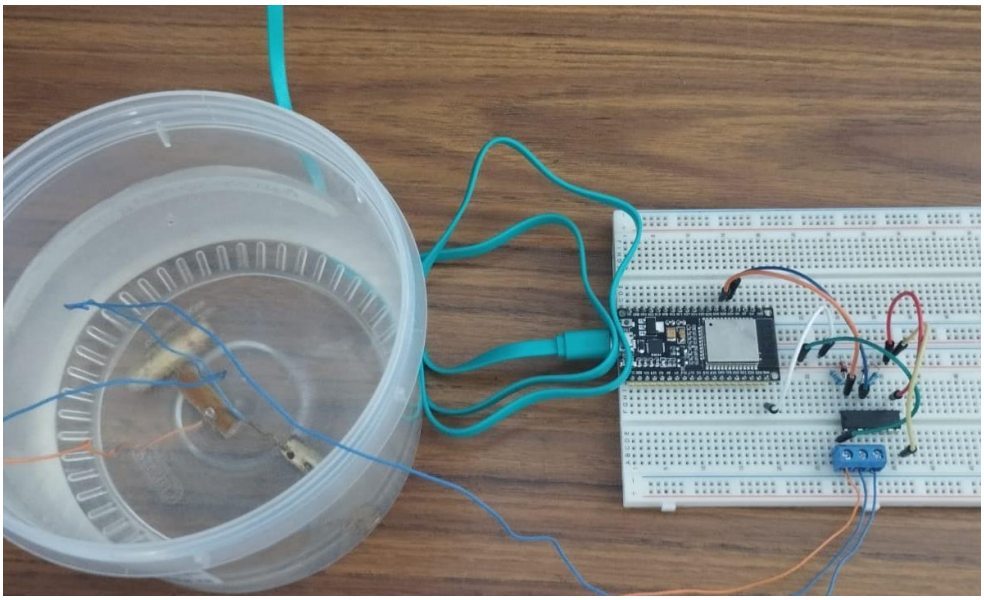


Ilustración 4.4 Circuito de nivel de agua con electrodos.

El agua del servicio público, al ser un conductor de corriente, permite la creación de un circuito cerrado cuando los electrodos están sumergidos en el agua. Esto significa que la corriente eléctrica puede fluir a través del agua y completar el circuito eléctrico. En contraste, cuando no hay agua entre los electrodos, el circuito se abre, lo que significa que la corriente no puede fluir y el circuito queda interrumpido.

El sistema utiliza dos electrodos en un recipiente con agua para determinar los niveles de agua y un tercer electrodo para el común, el cual proviene de la fuente de voltaje de 12 V.

El comportamiento del sistema es el siguiente:

1. El primer electrodo, que actúa como común, se mantiene siempre en contacto con el agua en el recipiente.
2. Cuando se inicia el llenado del recipiente con agua, el segundo electrodo (que está en el fondo del recipiente) hará contacto con el agua, y se formará un circuito cerrado entre el segundo electrodo y el común (primer electrodo). En esta etapa, si se cumplen las condiciones establecidas en la función IF (por ejemplo, si el valor de lectura es inferior a un umbral predeterminado), se mostrará el mensaje "Poca agua" en el monitor serial. Esto indica que el nivel de agua es bajo, ya que el primer electrodo está en contacto con el agua y cierra el circuito.

3. A medida que se sigue llenando el recipiente, el tercer electrodo (que está más alto en el recipiente) hará contacto con el agua, formando también un circuito cerrado entre el tercer electrodo y el común (primer electrodo).
4. Cuando ambos electrodos están en funcionamiento como circuito cerrado debido a que ambos hacen contacto con el agua, se considerará que el recipiente está "Lleno".
5. Si los dos electrodos no hacen contacto con el agua, el circuito se abrirá y se mostrará el mensaje "SIN AGUA" en el monitor serial.

De esta manera, el sistema puede detectar y comunicar los diferentes niveles de agua en el vaso a través del monitor serial, brindando información útil sobre la cantidad de agua presente en el contenedor.

Obteniendo el código del programa del siguiente modo:

```
int poca;
int lleno;

void setup()
{
  Serial.begin(9600);
  pinMode(32, INPUT);
  pinMode(33, INPUT);
}

void loop()
{
  poca=digitalRead(32);
  lleno=digitalRead(33);
  if (lleno==0 && poca ==0)
  {
   Serial.println("lleno");
   delay(1000);
  }

  if (lleno==1 && poca ==0)
  {
   Serial.println("poca agua");
   delay(1000);
  }

  if (lleno==1 && poca ==1)
  {
   Serial.println("vacio");
   delay(1000);
  }
}
```

En la simulación realizada, los resultados obtenidos coincidieron con los resultados esperados y se muestran en la ilustración 4.5.

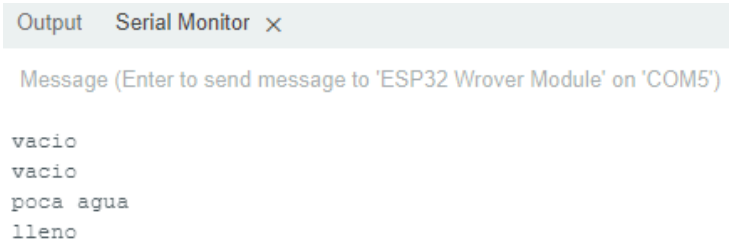


Ilustración 4.5 Resultados de electrodos en un recipiente de medio litro de agua.

Una vez que se obtuvieron buenos resultados, se procedió a realizar la verificación utilizando el circuito mostrado en la ilustración 4.4, el cual se implementó en un contenedor de mil litros, tal como se muestra en la ilustración 4.6. Para realizar la simulación, se utilizó un cable de red de tres metros, ya que esta distancia corresponde a la ubicación planificada para el módulo primario en la implementación real del sistema.

Ilustración 4.6 Electrodos en un depósito de mil litros de agua.

En esta prueba realizada se obtuvieron resultados satisfactorios mostrados en la ilustración 4.7 en donde se puede observar los tres estados del depósito reflejados en el monitor serial. lo que demuestra su eficiencia y funcionalidad para la gestión automatizada del nivel de agua.

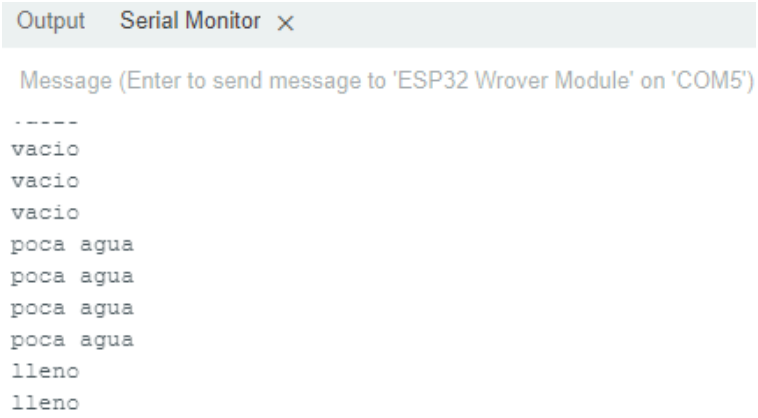


Ilustración 4.7 Resultados de electrodos en un recipiente de mil litros.

4.3 Sensor flujo de agua

Para saber si hay flujo de agua de la red pública se hizo una prueba con el sensor de flujo de agua FS300A.

El circuito sensor flujo de agua (Ilustración 4.8), incorpora una conexión a la terminal del sensor que cambia de nivel cada vez que hay un giro provocado por el flujo de agua, dicha terminal fue conectada al pin 2 de la ESP32, para alimentar el sensor se utilizó la terminal de 3.3 V de la tarjeta de control. Para las pruebas, se hicieron experimentos haciendo pasar agua de manera controlada proveniente de un recipiente, para ello se hizo circular un litro de agua, simulando el flujo de agua de la red pública.

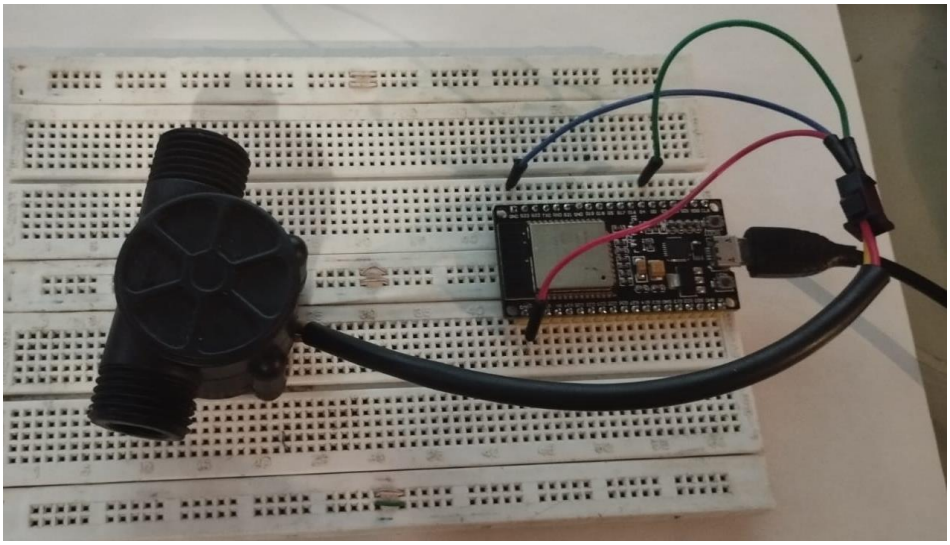


Ilustración 4.8 circuito sensor flujo de agua.

El código del programa se obtuvo de (NAYLAMP MECHATRONICS, 2023), se le hicieron algunas modificaciones teniendo el código del siguiente modo:

```
volatile int NumPulsos; //variable para la cantidad de pulsos recibidos
int PinSensor = 2;    //Sensor conectado en el pin 2
//---Función que se ejecuta en interrupción---------------
void ContarPulsos ()
{
  NumPulsos++;  //incrementamos la variable de pulsos
}
//---Función para obtener frecuencia de los pulsos--------
int ObtenerFrecuencia()
{
  int frecuencia;
  NumPulsos = 0;   //Ponemos a 0 el número de pulsos
  interrupts();    //Habilitamos las interrupciones
  delay(1000);   //muestra de 1 segundo
  noInterrupts(); //Desabilitamos las interrupciones
  frecuencia=NumPulsos; //Hz(pulsos por segundo)
```

```
  return frecuencia;
}

void setup()
{
  Serial.begin(9600);
  pinMode(PinSensor, INPUT);
  attachInterrupt(0,ContarPulsos,RISING); //(Interrupcion
0(Pin2),funcion,Flanco de subida)
}

void loop ()
{
  float frecuencia=ObtenerFrecuencia(); //obtenemos la Frecuencia de los
pulsos en Hz
  //-----Enviamos por el puerto serie---------------
  Serial.print("Frecuencia/Pulsos: ");
  Serial.println (frecuencia,0);
}
```

El resultado de la simulación se puede observar en la ilustración 4.9, donde al inicio de se ve una frecuencia de 60 pulsos, y que, al irse terminando el litro de agua, la cantidad de pulsos también se redujo. También se hicieron pruebas haciendo circular flujos de agua diferentes, y se pudo observar que cuando el agua circulante es muy poca, el sensor no la percibe.

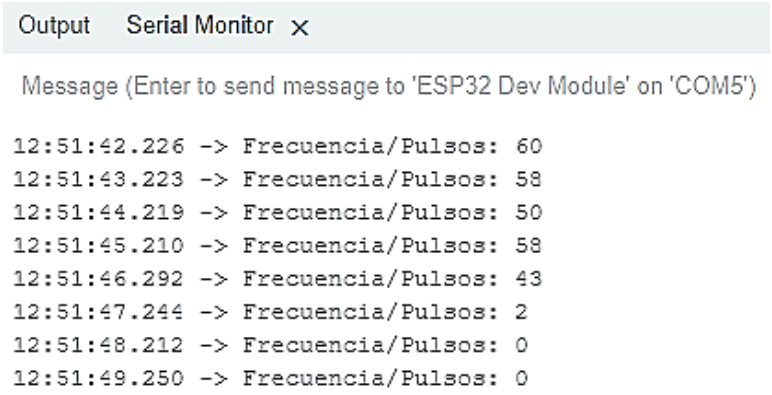


Ilustración 4.9 Resultados monitor serial.

4.4 RTC

En la prueba con el RTC se buscó tener la hora exacta, aun cuando el sistema no tuviera acceso a internet, obteniendo distintos resultados mostrados a continuación.

4.4.1 RTC DS3231

La primera prueba fue realizada con el módulo RTC3231, para su implementación se le suministro 3.3 V, se conectó el pin SDA y el SCL a los pines 21 y 22 de la placa ESP32, como se muestra en la ilustración 4.10.

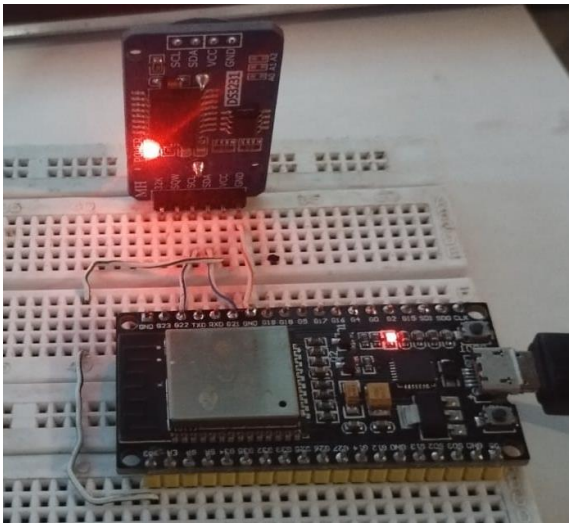


Ilustración 4.10 Circuito RTC.

Se procedió a realizar un experimento para "cargar" la hora actual al RTC, y comprobar que mantuviese la hora, al quitar y volver a polarizar.

Para ello, se utilizó un método de la biblioteca RTC[1], en el cual se incorporó directamente en el código de programación la hora de ese instante y se cargó inmediatamente el programa al sistema. El cual fue el siguiente:

```
#include "RTClib.h"
RTC_DS3231 rtc;
void setup ()
{
  Serial.begin(9600);
  if (! rtc.begin())
  {
    Serial.println("Couldn't find RTC");
    Serial.flush();
    while (1) delay(10);
  }
 rtc.adjust(DateTime(2023, 6, 22, 17, 28, 0));
}
void loop ()
{
    DateTime now = rtc.now();
    Serial.println(" ");
    Serial.print(now.hour(), DEC);
    Serial.print(':');
    Serial.print(now.minute(), DEC);
    delay(5000);
}
```

[1] Biblioteca accesible desde: https://github.com/adafruit/RTClib

El problema es que cada vez que se energiza al circuito se carga la hora que se puso en el código, en la ilustración 4.11, se muestra el monitor serial donde se puede visualizar la hora actual, contra la hora del RTC, siendo evidente el problema mencionado.

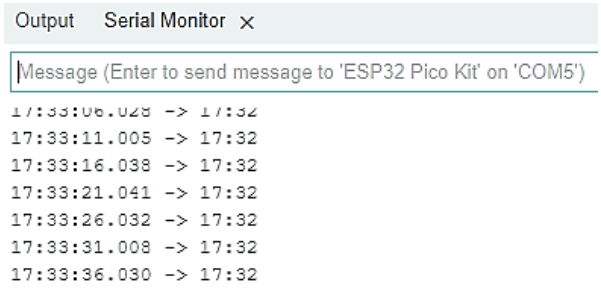


Ilustración 4.11 Resultado RTC externo.

Se tomó en consideración utilizar el código anterior para solo cargarle la hora al módulo RTC, sin embargo, para el usuario final no es conveniente puesto que el módulo RTC se llega a desfasar la hora, por lo cual al tratar de poner la hora actual sería un problema.

4.4.2 RTC blynk

Para abordar el problema con el RTC, se hizo otra prueba en esta ocasión se utilizó el RTC interno de la ESP32, en lugar de incluir en el código la hora de ese instante, se trabajó sobre un ejemplo de la biblioteca de Blynk[2] para RTC, para actualizar la hora del RTC, el código utilizado se da a continuación:

```
#define BLYNK_TEMPLATE_ID "TMPLrNJsLD3Q"
#define BLYNK_DEVICE_NAME "sensor ultrasonico"
#define BLYNK_FIRMWARE_VERSION        "0.1.0"
#define BLYNK_PRINT Serial
#define APP_DEBUG
#define USE_ESP32S2_DEV_KIT
#include "BlynkEdgent.h"
#include <Wire.h>
#include <TimeLib.h>
0#include <WidgetRTC.h>
#include <SPI.h>

BlynkTimer timer;
WidgetRTC rtc;

void clockDisplay()
{
  String currentTime = String(hour()) + ":" + minute() + ":" + second();
  Serial.print("Current time: ");
  Serial.print(currentTime);
  Serial.print(" ");
```

[2]Biblioteca accesible desde: https://github.com/blynkkk/blynk-library

```
  Blynk.virtualWrite(V1, currentTime); // Envia la hora a la app
}
BLYNK_CONNECTED() {// Sincroniza el tiempo de conexión
  rtc.begin();
}
void setup()
{
  Serial.begin(9600);
  BlynkEdgent.begin();
  Wire.begin();
  setSyncInterval(10 * 60);
  timer.setInterval(10000L, clockDisplay);
}
void loop()
{
  Blynk.run();
  timer.run();
}
```

En esta prueba, se obtuvieron inicialmente buenos resultados, como se muestra en la ilustración 4.12; la hora del sistema juntó con la hora del RTC son iguales

```
18:06:03.048 -> Current time: 17:6:2 23 6 2023
18:06:03.172 -> Current time: 17:6:2 23 6 2023
18:06:03.804 -> Current time: 17:6:2 23 6 2023
18:06:03.805 -> Current time: 17:6:2 23 6 2023
```

Ilustración 4.12 Resultado RTC Blynk.

Sin embargo, al reiterar pruebas se observó que cada vez que se energiza y no hay conexión a internet, no se carga al RTC la hora actual. En la figura 4.13 se observa dicha situación.

```
18:07:01.326 -> [75893] Connecting to WiFi: INFINITUMA693_2.4
18:08:03.655 -> Current time: 0:0:50 1 1 1970
```

Ilustración 4.13 Resultado RTC sin conexión con Blynk.

4.4.3 RTC DS3231 configurada con Blynk

Para la tercera prueba, se optó por tener una combinación de ambos programas, teniendo como RTC al módulo DS3231 donde al cargarle la hora actual se utilizó el método de la biblioteca de Blynk, para su posterior visualización, la hora obtenida del RTC se envía a la aplicación móvil y web de Blynk.

```
#define BLYNK_TEMPLATE_ID "TMPLrNJsLD3Q"
#define BLYNK_TEMPLATE_NAME "sensor ultrasonico"
#define BLYNK_FIRMWARE_VERSION        "0.1.0"
#define BLYNK_PRINT Serial
#define APP_DEBUG
#define USE_ESP32S2_DEV_KIT
#include "BlynkEdgent.h"
#include <TimeLib.h>
```

```
#include <WidgetRTC.h>
BlynkTimer timer;
WidgetRTC rtc;
void clockDisplay()
{
  String currentTime = String(hour()) + ":" + minute() + ":" + second();
  String currentDate = String(day()) + " " + month() + " " + year();
  Serial.print("Current time: ");
  Serial.print(currentTime);
  Serial.print(" ");
  Serial.print(currentDate);
  Serial.println();
  Blynk.virtualWrite(V1, currentTime);
}
BLYNK_CONNECTED()
{
  rtc.begin();
}
void setup()
{
  Serial.begin(115200);
  BlynkEdgent.begin();
   timer.setInterval(10000L, clockDisplay);
}
void loop()
{
  BlynkEdgent.run();
  clockDisplay();
}
```

En esta prueba se obtuvieron buenos resultados, al energizar el circuito se pudo observar en el monitor serial (ilustración 4.14), la hora del sistema y la del módulo RTC son iguales.

```
20:07:08.276 -> Hora: 19: 07
20:07:08.368 -> Hora: 19: 07
```

Ilustración 4.14 Hora configurada con Blynk.

Al desconectarse de internet sigue funcionando con el horario previamente establecido, inclusive si llega a tener alguna interrupción de energía eléctrica el módulo DS3231 sigue teniendo el registro de la hora. Como se muestra en la ilustración 4.15.

```
20:06:42.836 -> [118278] CONNECTING_NET => CONNECTING_CLOUD
20:06:42.902 -> Hora: 19: 06
20:06:42.902 -> [118289] Connecting to blynk.cloud:443
20:06:49.310 -> [124783] Secure connection failed
20:06:49.343 -> [124793] Connecting to blynk.cloud:443
20:06:56.372 -> [131784] Secure connection failed
20:06:56.434 -> [131784] CONNECTING_CLOUD => CONNECTING_NET
20:06:56.434 -> Hora: 19: 06
```

Ilustración 4.15 Hora sin internet

En la ilustración 4.16 se puede observar que una vez que se logró conexión a internet, el RTC se vuelve a sincronizar con la hora de la aplicación de Blynk teniendo los resultados esperados al inicio de la prueba.

```
20:06:56.469 -> [131786] Connecting to WiFi: INFINITUMA69B_2.4
20:06:59.351 -> [134819] Using Dynamic IP: 192.168.1.189
20:06:59.384 -> [134819] CONNECTING_NET => CONNECTING_CLOUD
20:06:59.420 -> Hora: 19: 06
20:06:59.448 -> [134830] Connecting to blynk.cloud:443
20:07:06.319 -> [141787] Secure connection failed
20:07:06.352 -> [141797] Connecting to blynk.cloud:443
20:07:07.721 -> [143219] Certificate OK
20:07:07.938 -> [143421] Ready (ping: 200ms).
20:07:08.062 -> [143558] CONNECTING_CLOUD => RUNNING
20:07:08.126 -> Hora: 19: 07
20:07:08.168 -> Hora: 19: 07
```

Ilustración 4.16 Hora con reconexión del circuito.

4.5 Comunicación

La elección de una comunicación inalámbrica inicialmente parecía una solución conveniente para conectar el módulo primario y secundario encargado del monitoreo del nivel de agua en el tinaco. Sin embargo, durante la implementación, surgieron desafíos significativos relacionados con la señal del Wifi. El módulo secundario se encontraba fuera del rango de la señal, lo que resultó en la incapacidad de comunicarse con el módulo primario. Además, la dependencia del acceso a internet también presentó un inconveniente, ya que cualquier pérdida de conexión dejaba el sistema inoperativo hasta que se restableciera la conexión.

Para superar estos obstáculos y garantizar una comunicación confiable y continua entre ambos módulos, se optó por utilizar una solución de comunicación alámbrica. La elección recayó en el protocolo de comunicación RS485, que ofrece una conexión estable y robusta. Este protocolo involucra el uso de dos módulos Max485 configurados como emisor y receptor, conectados mediante el puerto serial de la ESP32, como se detalla en la ilustración 4.17.

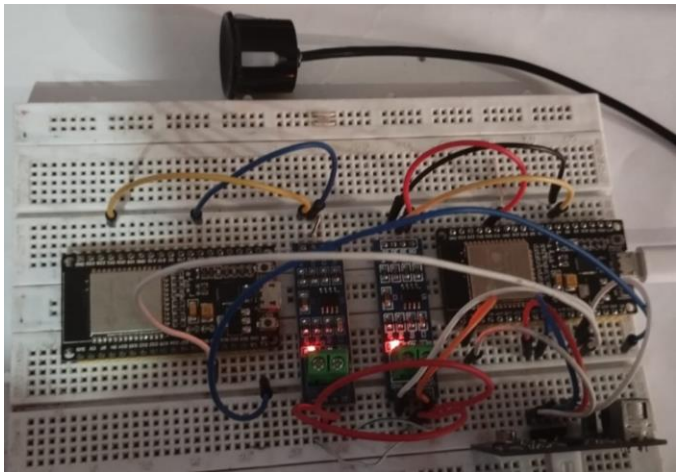


Ilustración 4.17 Comunicación serial.

Se realizó un código para la transmisión de los datos obtenidos por el sensor ultrasónico, el cual se muestra a continuación.

```
#define echoPin 32
#define trigPin 33
long duracion;  // variable para la duración del viaje de la onda de sonido
int distancia;   // variable para la medida de la distancia
int Enable_pin = 4;
void setup()
{
  Serial.begin(9600);
  delay(100);
  pinMode(trigPin, OUTPUT);  //Establece el trigPin como una salida
  pinMode(echoPin, INPUT);   //Establece el echoPin como una entrada
BlynkEdgent.begin();
  pinMode(Enable_pin, OUTPUT);
  digitalWrite(Enable_pin, HIGH);        //  (pin configurado en HIGH para
enviar el valor de la placa Maestro)
}
void loop()
{
  BlynkEdgent.run();
  Blynk.virtualWrite(V4, 1);
  digitalWrite(trigPin, LOW);  // Borra la condición trigPin
  delayMicroseconds(2);
  digitalWrite(trigPin, HIGH);  // Establece la trigPin HIGH (ACTIVA)
durante 10 microsegundos
  delayMicroseconds(10);
  digitalWrite(echoPin, LOW);  // Lee el echoPin, devuelve el tiempo de
viaje de la onda de sonido en microsegundos
  duracion = pulseIn(echoPin, HIGH);  // Calculando la distrancia
 distancia = duracion * 0.034 / 2;    // Velocidad de la onda de sonido
dividida por 2 (ida y vuelta)
  float porcentaje = map(distancia, 20, 130, 100, 0);
  Blynk.virtualWrite(V0, porcentaje);
  Serial.print("Porcentaje: ");
  Serial.println(porcentaje);
}
```

Y el código para el receptor es el siguiente:

```
int Enable_pin = 4;

void setup()
{
  Serial.begin(9600);
  pinMode(Enable_pin, OUTPUT);
```

```
  digitalWrite(Enable_pin, LOW);        //  (pin configurado en LOW para
recibir valor de la placa Maestro)
}

void loop()
{
  int PWM_master = Serial.parseInt();
  Serial.print("porcentaje:  ");
  Serial.println(PWM_master);
}
```

Se obtuvieron resultados esperados en la comunicación serial los cuales se muestran en la ilustración 4.18 en donde se observa el monitor serial de la ESP32 receptor, los datos obtenidos del sensor ultrasónico proveniente de la ESP32 emisor.

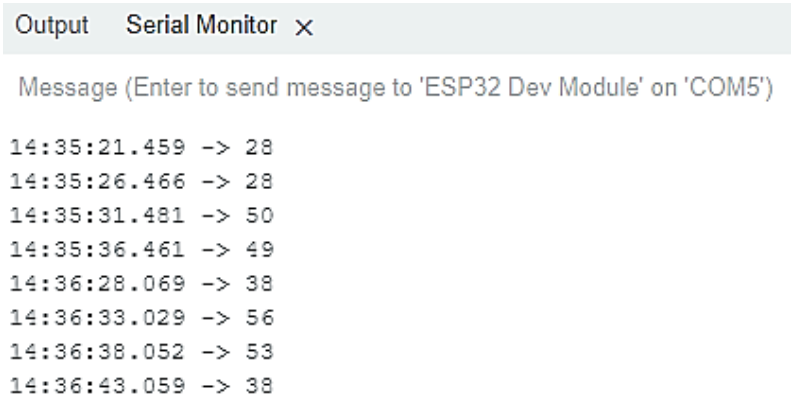


Ilustración 4.18 Resultado comunicación serial.

Con la adopción de la comunicación RS485, se logró una conexión más confiable y consistente entre los módulos primario y secundario. Ya no había problemas de alcance de señal y, al ser una comunicación alámbrica, no dependía del acceso a internet, lo que evitaba que el sistema quedara inoperativo por cualquier pérdida de conexión.

4.6 Comunicación con Blynk

Para la comunicación del módulo primario con la plataforma de Blynk se llevaron a cabo dos experimentos, el primero fue la transmisión de datos de la aplicación de Blynk a la ESP32 mientras que el segundo experimento fue él envió de datos provenientes de la placa para su posterior visualización en Blynk

4.6.1 TRANSMISION DE BLYNK A ESP32

En la transmisión de datos se hizo mediante dos barras deslizantes con los nombres: referencia y tolerancia, en los cuales, si se llega a cambiar su valor, este se transmite y se guarda en una localidad de memoria de la ESP32, quedando el código del siguiente modo:

```
#define BLYNK_TEMPLATE_ID "TMPLrNJsLD3Q"
#define BLYNK_TEMPLATE_NAME "sensor ultrasonico"
```

```
#define BLYNK_FIRMWARE_VERSION        "0.1.0"
#define BLYNK_PRINT Serial
#define APP_DEBUG
#define USE_ESP32S2_DEV_KIT
#include "BlynkEdgent.h"
#include <Preferences.h>
#include <Wire.h>
Preferences preference;
int referencia;
int tolerancia;

BLYNK_WRITE(V1)
{
  referencia = param.asInt();
  preference.putUInt("referencia", referencia);
}

BLYNK_WRITE(V2)
{
  tolerancia = param.asInt();
  preference.putUInt("tolerancia", tolerancia);
}

void setup()
{
  Serial.begin(115200);
  BlynkEdgent.begin();
  preference.begin("referencia", false);
  preference.begin("tolerancia", false);
  referencia = preference.getUInt("referencia", false);
  tolerancia = preference.getUInt("tolerancia", false);
}

void loop()
{
  BlynkEdgent.run();
  Serial.print("Referencia:");
  Serial.println(referencia);
  Serial.print("Tolerancia:");
  Serial.println(tolerancia);
  delay(1000);
}
```

En la ilustración 4.19 se muestra los valores establecidos en la aplicación de Blynk

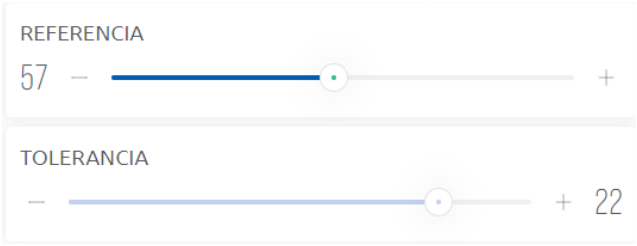


Ilustración 4.19 Barras deslizantes de la aplicación de Blynk.

El resultado obtenido se muestra en la ilustración 4.20, donde se observan los mismos valores provenientes de la aplicación de Blynk.

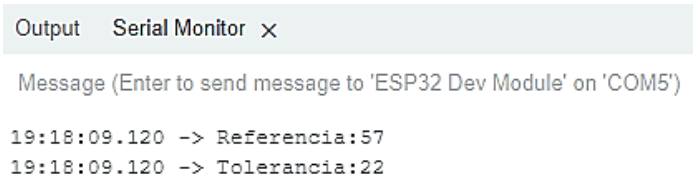


Ilustración 4.20 Resultados monitor serial.

4.6.2 Transmisión de ESP32 a Blynk

En la transmisión de valores de la ESP32 se utilizó el circuito del sensor ultrasónico, el propósito de este experimento es el visualizar por medio de la aplicación el nivel de agua en el depósito, el código se muestra a continuación:

```
#define BLYNK_TEMPLATE_ID "TMPLvVQr35tG"
#define BLYNK_DEVICE_NAME "RTC "
#define BLYNK_FIRMWARE_VERSION "0.1.0"
#define BLYNK_PRINT Serial
#define APP_DEBUG
#define USE_ESP32S2_DEV_KIT
#include "BlynkEdgent.h"
#define echoPin 32
#define trigPin 33
long duracion;  // variable para la duración del viaje de la onda de sonido
int distancia;   // variable para la medida de la distancia

void setup()
{
  Serial.begin(9600);
  delay(100);
  pinMode(trigPin, OUTPUT);  //Establece el trigPin como una salida
  pinMode(echoPin, INPUT);   //Establece el echoPin como una entrada
BlynkEdgent.begin();
}

void loop()
{
```

```
  BlynkEdgent.run();
  digitalWrite(trigPin, LOW);  // Borra la condición trigPin
  delayMicroseconds(2);
  digitalWrite(trigPin, HIGH);  // Establece la trigPin HIGH (ACTIVA)
durante 10 microsegundos
  delayMicroseconds(10);
  digitalWrite(echoPin, LOW);  // Lee el echoPin, devuelve el tiempo de
viaje de la onda de sonido en microsegundos
  duracion = pulseIn(echoPin, HIGH);  // Calculando la distancia
 distancia = duracion * 0.034 / 2;    // Velocidad de la onda de sonido
dividida por 2 (ida y vuelta)
  float porcentaje = map(distancia, 20, 130, 100, 0);
  Blynk.virtualWrite(V0, porcentaje); //envió de datos a la aplicación de
Blynk por el pin virtual V0
}
```

Para la visualización se utilizó una gráfica con los datos del 0 al 100, en la ilustración 4.21 se puede observar los datos enviados por la ESP32.

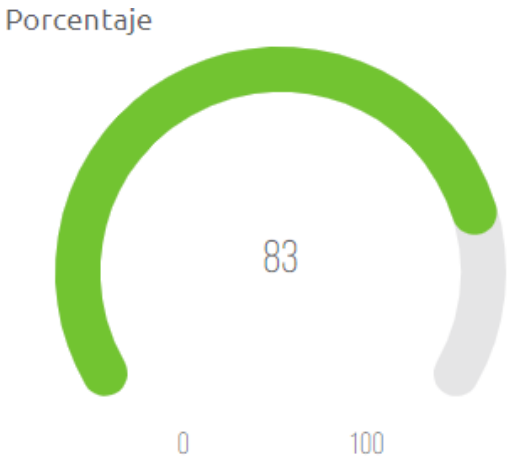


Ilustración 4.21 porcentaje de la medición del sensor ultrasónico.

Para corroborar que los datos obtenidos en la aplicación de Blynk son los mismos que los de la lectura del sensor ultrasónico, se procedió a visualizar los datos en el monitor serial, como lo muestra la ilustración 4.22.

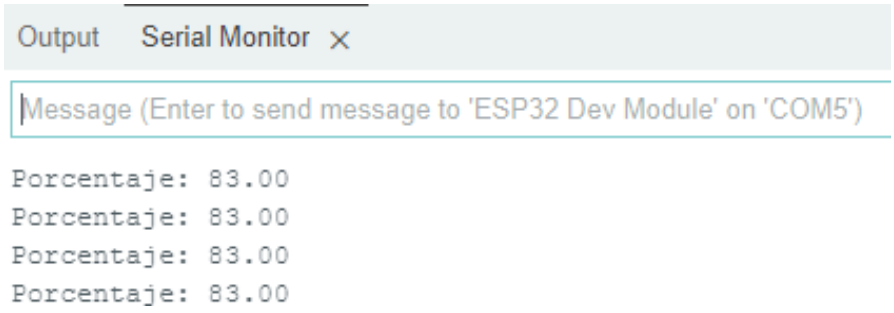


Ilustración 4.22 Resultados de medición por el sensor ultrasónico.

4.7 Pruebas Modulo primario

Las pruebas realizadas se llevaron a cabo juntando los códigos de las pruebas antes mencionadas, el cual es mostrado en el anexo. Los módulos primario y secundario están instalados en el lugar previsto mostrados en la ilustración 3.14 y 3.25.

El proceso de integración y prueba en el lugar donde será colocado es fundamental para asegurarse de que el sistema funcione correctamente en el contexto específico donde será implementado. Con esto, se garantiza que el sistema cumpla con los requerimientos establecidos y se logre una gestión efectiva del agua.

4.7.1 Encendido automático de la bomba sumergible

La primera prueba consistió en el encendido automático de la bomba sumergible, la cual fue conectada mediante el relevador de estado sólido. Para verificar su correcto funcionamiento, se observó el relevador, que cuenta con un LED de color rojo. Si el relevador no está siendo energizado, el LED permanece apagado; mientras que, si está activo, el LED se enciende, como se muestra en la ilustración 4.23.

El LED del relevador proporciona una indicación visual de si la bomba sumergible está siendo energizada adecuadamente o no. Esta verificación permite asegurar que el sistema está activando correctamente la bomba sumergible y, por lo tanto, se está logrando el encendido automático de la misma.

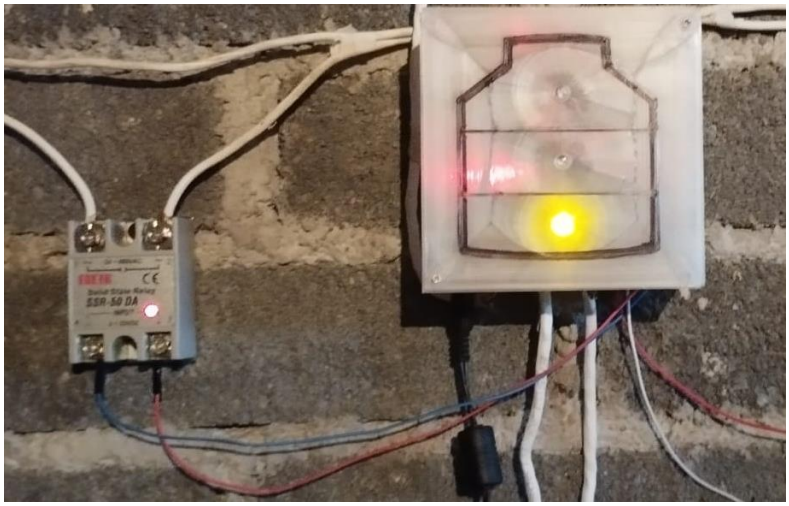


Ilustración 4.23 Encendido de bomba sumergible.

Las condiciones que hicieron que se encendiera el relevador de la bomba sumergible fueron las siguientes:

- El nivel de agua en el tinaco estaba por debajo de la referencia establecida. Cuando esto ocurrió, el sistema activó la bomba sumergible a través del relevador para rellenar el tinaco hasta alcanzar la referencia deseada siempre y cuando se cumpla la siguiente condición.
- El depósito se encontraba con agua disponible para ser bombeada.

Se llevaron a cabo dos pruebas más en el cual se observó el momento en el cual se apagó la bomba sumergible, las cuales fueron:

- Al pasar el límite de tolerancia establecido. Una vez que el nivel de agua en el tinaco alcanzaba el límite de tolerancia superior definido, el sistema apagaba la bomba sumergible para evitar un sobrellenado del tinaco.
- No contar con agua disponible en depósito. Si no había agua disponible en el depósito para ser bombeada hacia el tinaco, el sistema apagaba la bomba sumergible para evitar un funcionamiento ineficiente y garantizar que la bomba solo funcionara cuando hubiera suficiente agua para ser utilizada

Durante una de las pruebas realizadas, se observó que mientras se llenaba el tinaco, el depósito se quedaba sin agua, lo que provocaba que la bomba se llegara a apagar, como era lo esperado. Sin embargo, el depósito se mantenía llenando con agua proveniente de la red pública, lo que llevaba a creer que la bomba se encendería nuevamente al no haber llenado completamente el tinaco. No obstante, esto no ocurrió, y la bomba se volvió a encender solo cuando el agua regresó al punto de referencia en el tinaco donde debe ser activada.

Este comportamiento se debe a la lógica de control implementada en el sistema. Cuando el nivel de agua en el tinaco se encuentra por debajo del punto de referencia, la bomba se activa para rellenarlo hasta el nivel deseado. Si durante este proceso el depósito se queda sin agua, la bomba se apaga como es esperado para evitar que funcione sin agua y se dañe.

La bomba solo se volverá a encender únicamente cuando el agua regresa al punto de referencia en el tinaco donde debe ser activada. Esto es necesario para asegurar que el tinaco esté lleno y que el sistema de suministro de agua funcione adecuadamente.

4.7.2 Encendido automático bomba presurizadora

La segunda prueba realizada fue el encendido automático de la bomba presurizadora para el llenado del depósito. Como se muestra en la ilustración 4.24 el relevador encargado del encendido de la bomba se encuentra activo.

Ilustración 4.24 Encendido bomba presurizadora.

Las condiciones que hicieron que se encendiera el relevador de la bomba presurizadora fueron:

- Disponibilidad de agua de la llave pública. Cuando había disponibilidad de agua proveniente de la red pública, el sistema activaba el relevador de la bomba

presurizadora para aumentar la presión del agua, sin embargo, debería cumplir con la siguiente condición para ser activado.

- El depósito se encuentra sin disponibilidad de agua

Las pruebas para que se mantuviera apagado fueron:

- Sin disponibilidad de agua proveniente de la red publica
- El depósito se encuentra lleno. Si el depósito estaba lleno, el sistema también mantenía apagado el relevador de la bomba presurizadora, ya que no era necesario aumentar la presión del agua en el sistema debido a que el depósito ya estaba en su capacidad máxima.

4.7.3 Verificación de nivel de agua por medio de LED

Se observó los distintos niveles de agua en el tinaco mediante la información obtenida por el módulo secundario, los niveles de agua son reflejados en la tapa del módulo primario mediante la iluminación de los LED, donde:

En la ilustración 4.25, se muestra el nivel de agua dentro del rango del 67% al 100%. En este caso, todos los LED se encuentran encendidos, lo que indica que el tinaco está casi lleno o en su nivel máximo de capacidad.

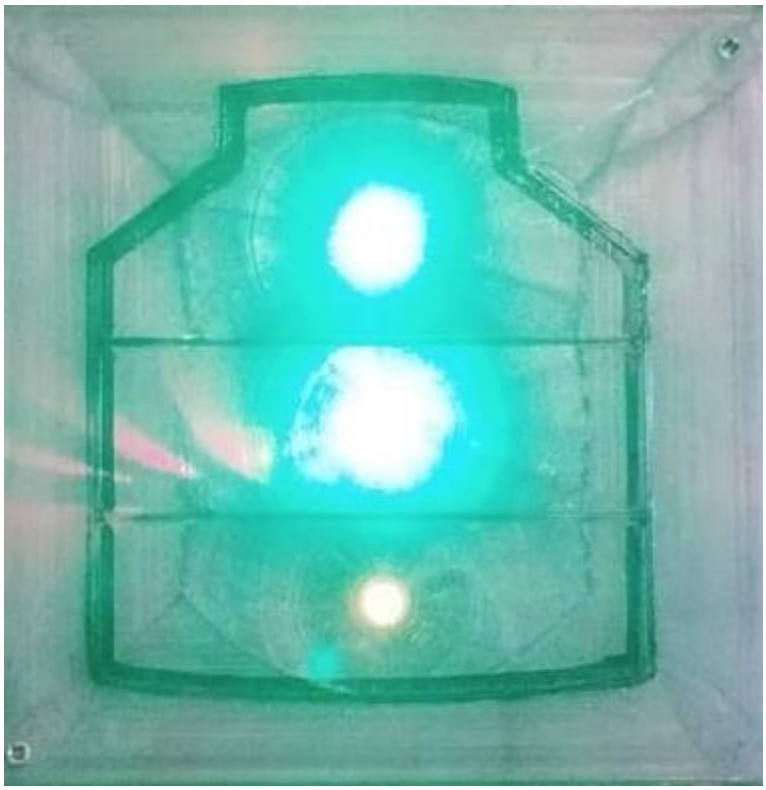


Ilustración 4.25 visualización del nivel de agua en el tinaco 67% al 100%.

En la ilustración 4.26, se observa el nivel de agua en el rango del 34% al 66%. En esta situación, solo dos LED están encendidos, lo que sugiere que el tinaco tiene una cantidad de agua moderada.

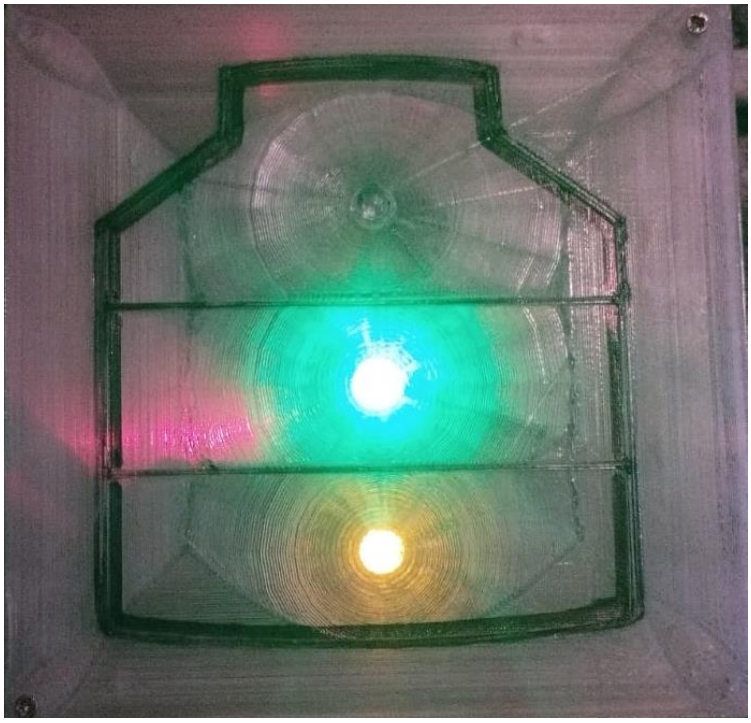


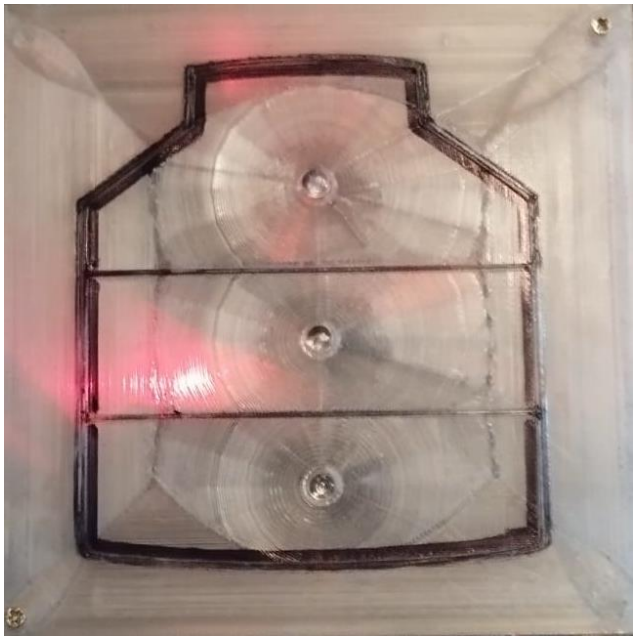
Ilustración 4.26 visualización del nivel de agua en el tinaco 34% al 66%.

En la ilustración 4.27, se observa el nivel de agua dentro del rango del 1% al 33%. En este caso, solo un LED está encendido, indicando que el tinaco tiene una cantidad de agua baja.

Ilustración 4.27 visualización del nivel de agua en el tinaco 1% al 33%.

Sin embargo, en la ilustración 4.28, se puede apreciar que los tres LED están apagados, y solo hay un LED rojo dentro de la caja. Esto señala claramente que no hay agua en el tinaco,

lo que implica que se ha alcanzado un nivel crítico y que es necesario tomar medidas para rellenar el tinaco lo antes posible.



4.28 visualización del nivel de agua en el tinaco 0%.

Conclusiones

El proyecto ha sido un éxito al lograr su objetivo principal de mejorar la eficiencia en el llenado de los depósitos y proporcionar un control fácil y accesible para los usuarios.

Para la activación de la bomba presurizadora, se utilizó un sensor de flujo de agua y un circuito detector de nivel, cuando el depósito requiere agua, y se detecta flujo proveniente de la red de agua pública, se activa la bomba presurizadora, y en las pruebas se pudo comprobar que el tiempo de llenado fue inferior, comparado al tiempo necesario para llenado cuando no se cuenta con la bomba.

La implementación de dos barras deslizantes en la plataforma web de Blynk para definir los horarios de encendido y apagado del sistema ha sido una solución eficiente y versátil. El rango de configuración proporciona flexibilidad al usuario, permitiéndole ajustar los horarios de funcionamiento de acuerdo con sus necesidades específicas.

Una de las ventajas clave de esta funcionalidad es evitar interrupciones durante las horas de descanso de los residentes. Al permitir que el usuario programe los horarios de encendido y apagado, el sistema puede adaptarse a las rutinas diarias y asegurarse de que no haya activaciones indeseadas que puedan perturbar el descanso o generar molestias innecesarias.

La utilización de un RTC DS3231 para mantener actualizada la hora del sistema de Blynk es una estrategia para lograr independencia respecto a la disponibilidad de conexión a internet. El RTC (Real Time Clock) es un dispositivo que permite contar con un reloj preciso incluso cuando no hay conexión a la red. De esta manera, el sistema puede seguir funcionando de manera autónoma, asegurando que los horarios programados por el usuario se respeten correctamente, sin depender de la conectividad constante a internet.

La combinación de las barras deslizantes y el RTC DS3231 proporciona una solución robusta para gestionar el horario de funcionamiento del sistema sin inconvenientes. Los usuarios pueden configurar los horarios de encendido y apagado según sus preferencias, y el RTC garantiza que el sistema mantenga la hora actualizada, incluso en caso de interrupción temporal de la conexión a internet. Para la activación en función de un horario, el sistema consulta la hora del RTC DS3231, y si de acuerdo con las configuraciones establecidas por medio de la aplicación, se está en un momento determinado para trabajar, el sistema funciona sin restricciones; de lo contrario, las bombas de agua quedan inhabilitadas.

Para la configuración de niveles de encendido-apagado de la bomba sumergible se colocaron dos barras deslizantes en la plataforma web de Blynk llamados referencia y tolerancia. La combinación de estas dos barras deslizantes brinda al usuario un control completo sobre los umbrales de funcionamiento de la bomba sumergible, lo que se traduce en un sistema altamente adaptable y personalizable adaptados a sus necesidades específicas y a las condiciones de la cisterna de agua.

La incorporación de dos electrodos para el monitoreo de la cisterna ha sido una medida esencial para asegurar un uso eficiente y seguro de las bombas de agua. Estos electrodos, ubicados a diferentes niveles en la cisterna, permiten obtener información en tiempo real sobre el nivel de agua, lo que resulta en dos beneficios clave:

- Protección de la bomba presurizadora: Debido al monitoreo constante del nivel de agua en la cisterna, el sistema puede tomar decisiones para proteger las bombas. Si el nivel de agua alcanza el nivel alto (indicando que la cisterna está llena), la bomba presurizadora se apaga automáticamente para evitar el sobrellenado y posibles daños a la bomba.
- Evitar el funcionamiento en seco: El otro electrodo que detecta el nivel bajo (indicando que la cisterna está vacía) es igualmente importante. Cuando el nivel de agua cae por debajo de este punto, el sistema no permitirá que la bomba sumergible funcione para evitar el bombeo en seco. Esta función de protección es esencial, ya que el funcionamiento en seco puede causar daños graves y reducir significativamente la vida útil de la bomba.

La incorporación del sensor ultrasónico en el tinaco para monitorear con precisión los niveles de agua es una solución práctica que proporciona información valiosa sobre el estado del tinaco en tiempo real.

La disponibilidad de visualización de los valores del nivel de agua en la aplicación web y móvil de Blynk agrega un elemento de conveniencia y accesibilidad al sistema. Los usuarios pueden monitorear la cantidad de agua en el tinaco desde cualquier lugar utilizando sus dispositivos móviles o computadoras, lo que les permite tomar decisiones rápidas y adecuadas sobre el uso del recurso hídrico.

Además, se incluyeron tres luces LED en el módulo primario para indicar el nivel de agua en el tinaco proporcionando una interfaz simple y rápida para obtener información visual sobre su estado. Esta característica puede ser particularmente útil cuando los usuarios deseen verificar rápidamente el nivel de agua sin necesidad de acceder a la aplicación.

La combinación de la visualización en la aplicación web y móvil de Blynk junto con las luces LED crea un sistema integral y completo de monitoreo de los niveles de agua en el tinaco. Los usuarios tienen múltiples formas de obtener información sobre el estado del tinaco, lo que les brinda flexibilidad y opciones para acceder a los datos de manera conveniente.

Para lograr que el sistema funcione con independencia de la comunicación con la aplicación web, se ha empleado la biblioteca "BlynkEdgent.h". Esta elección permite que el sistema siga operando incluso en ausencia de señal Wi-Fi, lo que es esencial para mantener la funcionalidad y la operatividad del sistema en situaciones de interrupción temporal de la conexión a Internet.

Además de proporcionar independencia de la conexión a Internet, la biblioteca "BlynkEdgent.h" también se ha utilizado para la comunicación con la tarjeta ESP32 y las aplicaciones web y móvil de Blynk. Esto facilita la interacción entre el usuario y el sistema, permitiendo configurar y supervisar el funcionamiento de manera efectiva y amigable.

El microcontrolador ESP32 ha sido seleccionado como el microcontrolador del sistema debido a sus numerosas ventajas, incluida su relación costo-beneficio. Además, la integración de un módulo Wi-Fi en el ESP32 ha sido fundamental para permitir la comunicación del sistema con la plataforma de Blynk y, en última instancia, facilitar la gestión y el control remoto del sistema.

La implementación de las diferentes visualizaciones y funciones en las aplicaciones web y móvil demuestra una consideración cuidadosa de accesibilidad para los usuarios.

En la aplicación web, se ha proporcionado una visualización completa del sistema con tres aspectos clave: el monitoreo del nivel de agua en el tinaco, la hora actual del módulo RTC y el estado de las bombas (presurizadora y sumergible).

Las barras deslizantes utilizadas tanto para configurar los horarios de encendido y apagado de las bombas como para ajustar la referencia y tolerancia de los niveles de encendido-apagado, ofrecen una forma flexible y precisa para que los usuarios personalicen el funcionamiento del sistema según sus necesidades específicas.

En cuanto a la aplicación móvil, se ha optado por una interfaz más sencilla y enfocada en la información esencial para una rápida visualización. Los widgets que muestran el nivel de agua en el tinaco y los estados de las bombas (activado/desactivado) permiten a los usuarios acceder rápidamente a la información más relevante sin necesidad de ajustar configuraciones.

La decisión de no incluir las barras deslizantes en la aplicación móvil es una elección para evitar cambios accidentales en las configuraciones. De esta manera, se garantiza que las configuraciones más detalladas y personalizadas se realicen en la aplicación web, donde los usuarios pueden tener una experiencia más completa y controlada.

Si bien el proyecto ha alcanzado sus objetivos principales de manera satisfactoria, siempre existen oportunidades de mejora. Una posible propuesta para el futuro es la integración de asistentes virtuales como Alexa, que permitirían un mayor nivel de interacción para activar o desactivar las bombas y proporcionar información sobre los niveles de agua disponibles en los depósitos, así como emplear un enlace inalámbrico para conectar los dos módulos en lugar de utilizar cables de conexión.

En resumen, el proyecto ha demostrado una implementación sólida y da soluciones para mejorar la eficiencia en el uso del agua, así como brindar un control intuitivo y accesible a los usuarios. La combinación de sensores, barras deslizantes, RTC y conectividad inalámbrica ha logrado un sistema completo y funcional que cumple con su objetivo principal y puede adaptarse a diferentes necesidades y condiciones.

Referencias

Aguilar Peña , J. D., & Montejo Ráez, M. Á. (27 de Mayo de 2023). *Introducción a los dispositivos electrónicos de potencia*. Obtenido de https://www.ugr.es/~amroldan/enlaces/dispo_potencia/introd.htm

Alexander, Rivas Alpizar. (10 de Marzo de 2020). *Controlador de nivel de agua en tanque con sensor ultrasonico y Arduino.* Obtenido de SYSADMINS: https://www.sysadminsdecuba.com/2020/03/controlador-de-nivel-de-agua-en-tanque-con-sensor-ultrasonico-y-arduino/

Allusb.com. (26 de Mayo de 2023). *USB History*. Obtenido de http://www.allusb.com/usb-history

Alonso, R. (12 de agosto de 2021). *HZ hardZone*. Obtenido de https://hardzone.es/reportajes/que-es/reloj-tiempo-real-rtc-hardware/

AM GROUP. (20 de 04 de 2023). *Aristegui Maquinaria*. Obtenido de https://www.aristegui.info/como-funciona-una-red-de-abastecimiento-de-agua-potable/

Auto Solar. (12 de 03 de 2023). *AutoSolar*. Recuperado el 12 de 03 de 2023, de Bombas de agua, tipología y funciones: https://autosolar.pe/aspectos-tecnicos/bombas-de-agua-tipologia-y-funciones

CAMI Research. (10 de Mayo de 2023). *The RS232 STANDARD.* Obtenido de https://www.camiresearch.com/Data_Com_Basics/RS232_standard.html

Comisión Nacional para el Uso Eficiente de la Energía. (15 de Julio de 2014). *GOBIERNO DE MEXICO*. Obtenido de https://www.gob.mx/conuee/acciones-y-programas/sistemas-de-agua-potable-sistemas-de-agua-potable-bombeo-de-agua-potable-municipal-estados-y-municipios?state=published#:~:text=El%20proceso%20del%20suministro%20de,aguas%20superficiales%20o%20aguas%20subterr

del Valle Hernández, L. (07 de Abril de 2023). *programarfacil.com*. Obtenido de #101. Proyectos IoT con Arduino, las plataformas más importantes: https://programarfacil.com/podcast/proyectos-iot-con-arduino/

Duran Juárez, J. M., & Torres Rodríguez, A. (2006). Los problemas del abastecimiento de agua potable en una ciudad media. *Espiral*, 129-162.

ECS INC INTERNATIONAL. (04 de Abril de 2023). Obtenido de WHAT IS A REAL TIME CLOCK (RTC)?: https://ecsxtal.com/what-is-a-real-time-clock-rtc/

EDDYPUMP CORPORATION. (24 de FEBRERO de 2023). *EDDYPUMP CORPORATION*. Obtenido de Lo que necesita saber sobre las bombas sumergibles: https://eddypump.com/es/educacion/lo-que-necesita-saber-sobre-las-bombas-sumergibles

eicos 40. (19 de Octubre de 2022). *eicos40*. Obtenido de ¿Qué es un Sensor de Flujo?: http://www.eicos.com/datos-tecnicos/que-es-un-sensor-de-flujo/

ELECTROBOMBAS JAVEA TECNOLOGIAS DEL AGUA. (12 de 03 de 2023). *Novedades en instalaciones y servicios de riego y desarrollo tecnológico*. Recuperado el 12 de 03 de 2023, de ELECTROBOMBAS JAVEA TECNOLOGIAS DEL AGUA: https://electrobombasjavea.com/blog/que-es-una-bomba-de-agua-y-como-funciona-una-electrobomba#:~:text=Su%20funcionamiento%20es%20sencillo%20y,fluido%20que%20alimenta%20la%20bomba.

ELECTRÓNICAONLINE. (14 de MAYO de 2023). *ELECTRÓNICAONLINE*. Obtenido de Diodo Zener como Regulador de Voltaje: https://electronicaonline.net/componentes-electronicos/diodo/diodo-zener-como-regulador-de-voltaje/

Electrotec. (01 de 04 de 2023). Obtenido de COMO INSTALAR UNA BOMBA DE AGUA: https://electrotec.pe/blog/InstalacionBombaDeAgua

EVANS. (12 de Marzo de 2023). *BOMBA SUMERGIBLE DOMESTICA PARA CISTERNA.* Obtenido de EVANS: https://evans.com.mx/media/manuales/MP_SPM1.pdf

Ferreabasto. (27 de abril de 2020). *Ferreabasto*. Recuperado el 25 de enero de 2023, de Instalacion de bomba presurizadora Igoto: https://ferreabasto.com/blogs/tips-para-uso-e-instalacion-de-sus-productos/instalacion-de-bomba-presurizadora-igoto

Ferremax. (13 de Enero de 2021). *Grupo Ferre-Max*. Obtenido de Bomba presurizadora, ¿Qué es y cuál es su funcionamiento?: https://distribuidortruper.mx/bomba-presurizadora-de-agua-que-es-como-funciona/

fluideco. (20 de Agosto de 2019). *fluideco*. Obtenido de ¿Qué es una bomba centrífuga?: https://fluideco.com/que-es-una-bomba-centrifuga/

GSL Industrias. (20 de Julio de 2021). *GSL Industrias*. Obtenido de TRANSISTORES DE POTENCIA: https://industriasgsl.com/blogs/automatizacion/transistores_de_potencia

Hernandez, A. (10 de Marzo de 2015). *TALLERELECTRONICA.COM*. Obtenido de https://tallerelectronica.com/2015/03/10/indicador-electronico-de-nivel-de-agua/

La comunicacion Serie. (02 de Mayo de 2023). *Dispositivos logicos programables*. Obtenido de http://sedici.unlp.edu.ar/bitstream/handle/10915/72131/Tesis.%20Protocolos%20de%20comunicaci%C3%B3n%20entre%20microcontroladores.pdf-PDFA.pdf?sequence=2&isAllowed=y

Luis Llamas. (18 de octubre de 2016). *Ingenieria,Informatica y diseño*. Obtenido de Reloj y calendario en Arduino con los RTC DS1307 y DS3231: https://www.luisllamas.es/reloj-y-calendario-en-arduino-con-los-rtc-ds1307-y-ds3231/

Menecaxa. (14 de Enero de 2021). *¿Qué es un control de nivel electrónico o electronivel?* Obtenido de Necaxa: https://www.menecaxa.mx/post/que-es-un-control-de-nivel-de-control-electronico-o-electronivel#:~:text=Un%20electronivel%20es%20un%20dispositivo,por%20bajo%20nivel%20de%20succi%C3%B3n.

Microchip Technology. (27 de Septiembre de 2014). *Overview and Use of the PICmicro Serial Peripheral*. Obtenido de http://ww1.microchip.com/downloads/en/devicedoc/spi.pdf

Microelectronics. (13 de Abril de 2023). *#ESP32 - REAL-TIME CLOCK #RTC INTERNO*. Obtenido de https://vasanza.blogspot.com/2021/08/esp32-real-time-clock-rtc-interno.html

MN Del Golfo. (24 de 05 de 2023). *¿Cómo funciona un electronivel?* Obtenido de Home Center Tu asesor confiable: https://www.mndelgolfo.com/blog/reportaje/como-funciona-un-electronivel/

NAYLAMP MECHATRONICS. (19 de Octubre de 2022). *NAYLAMP MECHATRONICS*. Obtenido de SENSOR DE FLUJO DE AGUA 1/2" YF-S201: https://naylampmechatronics.com/sensores-liquido/108-sensor-de-flujo-de-agua-12-yf-s201.html

NAYLAMP MECHATRONICS. (11 de 08 de 2023). *TUTORIAL SENSOR DE FLUJO DE AGUA*. Obtenido de https://naylampmechatronics.com/blog/47_tutorial-sensor-de-flujo-de-agua.html

NXP. (23 de Junio de 2015). *I2C-bus specification and user manual*. Obtenido de http://www.nxp.com/acrobat/usermanuals/UM10204_3.pdf

Rivas Alpizar, A. (10 de Marzo de 2020). *SYSADMINS*. Obtenido de https://www.sysadminsdecuba.com/2020/03/controlador-de-nivel-de-agua-en-tanque-con-sensor-ultrasonico-y-arduino/

Rodriguez Gonzalez , F., & Cortez Guardiola, B. A. (07 de Octubre de 2020). *studocu*. Obtenido de https://www.studocu.com/es-mx/document/universidad-autonoma-de-nuevo-leon/diseno-de-sistemas-de-abastecimiento-de/act-1-dsa-bacg-nota-10/12479787

Rotoplas. (24 de 04 de 2023). Electronivel para Tinaco y Cisterna. *electronivel-ficha-tecnica*. Obtenido de Rotoplas: https://rotoplas.com.mx/rtp-resources/productos/electronivel/electronivel-ficha-tecnica.pdf

RS. (27 de Mayo de 2023). *RS*. Obtenido de Tiristores: https://es.rs-online.com/web/c/semiconductores/semiconductores-discretos/tiristores/

Sabas, A. (30 de Mayo de 2023). *Plataformas IoT para Dummies*. Obtenido de SG: https://sg.com.mx/revista/57/plataformas-iot-dummies

SDI Industrial. (15 de Mayo de 2022). *SDI*. Obtenido de Relevadores: https://sdindustrial.com.mx/blog/relevadores/#:~:text=%C2%BFQu%C3%A9%20son%20los%20relevadores%3F,este%20mismo%20tipo%20de%20energ%C3%ADa.

SHI Servicio Hidraulico Industrial. (s.f.). *SHI Servicio Hidraulico Industrial*. Recuperado el 06 de 01 de 2023, de SHI Servicio Hidraulico Industrial: https://www.bombas-hidraulicas.com.mx/bombas-hidraulicas/

Siapa. (15 de Febrero de 2014). *CRITERIOS Y LINEAMIENTOS TÉCNICOS PARA FACTIBILIDADES*. Obtenido de Sistemas de Agua Potable.:

https://www.siapa.gob.mx/sites/default/files/capitulo_2._sistemas_de_agua_potable-2a._parte.pdf

Siapa. (15 de Febrero de 2014). Sistemas de Agua Potable. *CRITERIOS Y LINEAMIENTOS TÉCNICOS PARA FACTIBILIDADES*, págs. 1-36. Obtenido de Sistemas de Agua Potable.: https://www.siapa.gob.mx/sites/default/files/capitulo_2._sistemas_de_agua_potable-1a._parte.pdf

Solectro. (21 de Febrero de 2022). *Solectro*. Obtenido de Transistores unipolares (MOSFET): https://solectroshop.com/es/blog/transistores-unipolares-mosfet-n115

SOLO ELECTRONICOS. (08 de ABRIL de 2023). *SOLO ELECTRONICOS*. Obtenido de CAYENNE: https://soloelectronicos.com/tag/cayenne/

Tecnologia Hidrotermica. (21 de Octubre de 2022). Obtenido de Tecnologia Hidrotermica: Tecnologia Hidrotermica

Texas Instruments. (26 de Mayo de 2010). Obtenido de RS-422 and RS-485 Standards Overview and: https://www.ti.com/lit/an/slla070d/slla070d.pdf

Universidad Nacional de PIura. (11 de 08 de 2023). *Studocu.* Obtenido de https://www.studocu.com/pe/document/universidad-nacional-de-piura/ingenieria-de-control-i/descripcion-tiristor/54830563

UNIVERSIDAD TEC Virtual DEL SISTEMA TECNOLÓGICO DE MONTERREY. (2022). *Centros Comunitarios de Aprendizaje.* Recuperado el 24 de 06 de 2022, de http://www.cca.org.mx/ps/lideres/cursos/av_a/html/materiales/t1.pdf

uv.es. (27 de mayo de 2023). *Diodos de Potencia*. Obtenido de https://www.uv.es/marinjl/electro/diodo.html#:~:text=Los%20diodos%20de%20potencia%20se,una%20peque%C3%B1a%20intensidad%20de%20fugas.

Fuentes de imágenes

Ilustración 2.1 Instalación de electronivel Recuperada de: https://rotoplas.com.mx/rtp-resources/productos/electronivel/electronivel-ficha-tecnica.pdf, el: 4 de marzo de 2023

Ilustración 2.2 Funcionamiento de llenado con electrodos. Recuperada de: https://www.youtube.com/watch?v=ltfPpsGwkxI&ab_channel=NassarElectronics, el: 4 de marzo de 2023

Ilustración 2.3 Circuito llenado automático con sensor de ultrasonido. Recuperada de: https://www.sysadminsdecuba.com/2020/03/controlador-de-nivel-de-agua-en-tanque-con-sensor-ultrasonico-y-arduino/, el: 10 de marzo de 2023

Ilustración 2.4 Conexión electrodo con ULN2803. Recuperada de: https://tallerelectronica.com/2015/03/10/indicador-electronico-de-nivel-de-agua/, el: 10 de marzo de 2023

Ilustración 2.5 Medición de nivel de agua con sensor ultrasonido. Recuperada de: https://www.sysadminsdecuba.com/2020/03/controlador-de-nivel-de-agua-en-tanque-con-sensor-ultrasonico-y-arduino/, el: 10 de marzo de 2023

Ilustración 2.6 Diagrama de instalación Bomba sumergible. Recuperada de: https://evans.com.mx/media/manuales/MP_SPM1.pdf, el: 12 de marzo de 2023

Ilustración 2.7 Diagrama de instalación bomba tipo bala. Recuperada de: https://www.youtube.com/watch?v=TNRRJerjYLo&ab_channel=KelvinJr.FerrerasM, el: 12 de marzo de 2023

Ilustración 2.8 Principio de funcionamiento de una bomba centrifuga. Recuperada de: http://www.unet.edu.ve/~maqflu/doc/LAB-1-95.htm, el: 12 de marzo de 2023

Ilustración 2.9 instalación de bomba centrifuga. Recuperada de: https://www.google.com/search?q=instalacion+de+una+bomba+centrifuga&tbm=isch&ved=2ahUKEwiRhdLisKuAAxVhLN4AHUXcBvUQ2-cCegQIABAA&oq=instalacion+de+una+bomba+centrifuga&gs_lcp=CgNpbWcQAzIFCAAQgAQyBggAEAcQHjoICAAQBRAHEB46CAgAEAgQBxAeUABY-ipg6S1oAXAAeACAAV6IAYcIkgECMTOYAQCgAQGqAQtnd3Mtd2l6LWltZ8ABAQ&sclient=img&ei=RIzAZJHTL-HY-LYPxbibqA8&bih=750&biw=1522&hl=es&safe=active&ssui=on#imgrc=UGBgtROUQ75h_M, el: 12 de marzo de 2023

Ilustración 2.10 Ejemplo de instalación de una bomba presurizadora. Recuperada de: https://ferreabasto.com/products/presurizador-automatico-1-3hp-igoto, el: 13 de marzo de 2023

Anexo

Programa modulo primario

```
#define BLYNK_TEMPLATE_ID "TMPLrNJsLD3Q"
#define BLYNK_DEVICE_NAME "sensor ultrasonico"
#define BLYNK_FIRMWARE_VERSION        "0.1.0"
#define BLYNK_PRINT Serial
#define APP_DEBUG
#define USE_ESP32S2_DEV_KIT

#include "BlynkEdgent.h"
#include <Preferences.h>
#include <RTClib.h>
#include <Wire.h>
#include <time.h>

Preferences preference;
RTC_DS3231 RTC;

int lleno;
int mitad;
int referencia;
int tolerancia;
int apagado;
int encendido;

int PinSensor = 26;    //Sensor conectado en el pin
int Enable_pin = 4;
volatile int NumPulsos; //variable para la cantidad de pulsos recibidos
unsigned long tempRef1=millis();
unsigned long tempRef2=millis();

//---Función que se ejecuta en interrupción---------------
void ContarPulsos ()
{
  NumPulsos++;  //incrementamos la variable de pulsos
}

//---Función para obtener frecuencia de los pulsos--------
int ObtenerFrecuencia()
{
  int frecuencia;
  NumPulsos = 0;  //Ponemos a 0 el número de pulsos
  interrupts();   //Habilitamos las interrupciones
  delay(1000);      //muestra de 1 segundo
  noInterrupts(); //Desabilitamos las interrupciones
  frecuencia = NumPulsos; //Hz(pulsos por segundo)
  return frecuencia;
}

```

```
48  BLYNK_WRITE(InternalPinRTC)
49  {
50    time_t t = param.asLong();
51    RTC.adjust(DateTime(t));      //establece la hora del rtc DS3231
52  }
53
54  void clockDisplay()
55  {
56    char currentTime[10];
57    DateTime now = RTC.now();
58
59    sprintf(currentTime, "%02d: %02d", now.hour(), now.minute());
60    printf("Current time: %s\n", currentTime);
61    String hora actual = String(currentTime);
62    //envia la hora a la aplicación
63    Blynk.virtualWrite(V5, horaActual);
64  }
65
66  BLYNK_CONNECTED()
67  {
68    Blynk.sendInternal("rtc", "sync");
69  }
70
71  void setup()
72  {
73    Serial.begin(9600);
74    delay(100);
75    pinMode(5, OUTPUT);
76    pinMode(12, OUTPUT);
77    pinMode(13, OUTPUT);
78    pinMode(17, OUTPUT);
79    pinMode(18, OUTPUT);
80    pinMode(19, OUTPUT);
81    pinMode(34, INPUT);
82    pinMode(35, INPUT);
83    BlynkEdgent.begin();
84    Wire.begin();
85    RTC.begin();
86    pinMode(PinSensor, INPUT);
87    pinMode(Enable_pin, OUTPUT);
88    digitalWrite(Enable_pin, LOW);
      attachInterrupt(26, ContarPulsos, RISING);
89    noInterrupts();
90    preference.begin("referencia", false);
91    preference.begin("tolerancia", false);
92    referencia = preference.getUInt("referencia", false);
93    tolerancia = preference.getUInt("tolerancia", false);
94    preference.begin("apagado", false);
95    preference.begin("encendido", false);
96    apagado = preference.getUInt("apagado", false);
97    encendido = preference.getUInt("encendido", false);
98  }
```

```
 99
100  void loop()
101  {
102    int hora =now.hour();
103    if(hora >= encendido && hora < apagado)
104    {
105      BlynkEdgent.run();
106      if((millis()-tempRef1)>4100)
107      {
108        tempRef1=millis();
109        programadeposito();
110        programatinaco();
111      }
112    }
113  }
114
115  BLYNK_WRITE(V1)
116  {
117    referencia = param.asInt();
118    preference.putUInt("referencia", referencia);
119  }
120
121  BLYNK_WRITE(V2)
122  {
123    tolerancia = param.asInt();
124    preference.putUInt("tolerancia", tolerancia);
125  }
126
127  BLYNK_WRITE(V7)
128  {
129    apagado = param.asInt();
130    preference.putUInt("apagado", apagado);
131  }
132
133  BLYNK_WRITE(V8)
134  {
135    encendido = param.asInt();
136    preference.putUInt("encendido", encendido);
137  }
138
139  void programatinaco()
140  {
141    mitad = digitalRead(35);
142    int med = Serial.parseInt();
143    float porcentaje = map(med, 20, 130, 100, 0);
144    Blynk.virtualWrite(V0, porcentaje);
145
146    if (mitad == 1)
147    {
148      Blynk.virtualWrite(V4, 0);
149      digitalWrite(12, LOW);
150    }
```

```
151     else
152     {
153       if (porcentaje <= (referencia - tolerancia) )
154       {
155         Blynk.virtualWrite(V4, 1);
156         digitalWrite(12, HIGH);
157       }
158       else if (porcentaje >= (referencia + tolerancia))
159       {
160         Blynk.virtualWrite(V4, 0);
161         digitalWrite(12, LOW);
162       }
163     }
164     int nivel = map(med, 20, 130, 3, 0);
165     switch(nivel)
166       {
167         case 0:
168           digitalWrite(17, 0);
169           digitalWrite(5, 0);
170           digitalWrite(18, 0);
171           break;
172         case 1:
173           digitalWrite(17, 1);
174           digitalWrite(5, 0);
175           digitalWrite(18, 0);
176           break;
177         case 2:
178           digitalWrite(17, 1);
179           digitalWrite(5, 1);
180           digitalWrite(18, 0);
181           break;
182         case 3:
183           digitalWrite(17, 1);
184           digitalWrite(5, 1);
185           digitalWrite(18, 1);
186       }
187   }
188   ////////////////////////////////////////////////////////////////////////////////
189   void programadeposito()
190   {
191     lleno = digitalRead(34);
192     float frecuencia = ObtenerFrecuencia();
193     if (lleno == 0)
194     {
195       Blynk.virtualWrite(V6, 0);
196       digitalWrite(13, LOW);
197     }
198     else
199     {
200       if (frecuencia >= 20)
201       {
202         Blynk.virtualWrite(V6, 1);
```

```
203         digitalWrite(13, HIGH);
204       }
205       else
206       {
207         Blynk.virtualWrite(V6, 0);
208         digitalWrite(13, LOW);
209       }
210     }
211   }
212
213
```

Programa modulo secundario

```
#define Chopin 32
#define trigPin 33

// defines variables
unsigned long tempRef1=millis();
long duration;
int distance;

void setup()
{
  pinMode(trigPin, OUTPUT);           //Establece el trigPin como una
salida
  pinMode(echoPin, INPUT);            //Establece el echoPin como una
entrada
  Serial.begin(9600);
}

void loop()
{
  if((millis()-tempRef1)>4000)
  {
    tempRef1=millis();
  digitalWrite(trigPin, LOW);         // Borra la condición trigPin
  delayMicroseconds(2);
  digitalWrite(trigPin, HIGH);        // Establece la trigPin HIGH
(ACTIVA) durante 10 microsegundos
  delayMicroseconds(10);
  digitalWrite(echoPin, LOW);         // Lee el echoPin, devuelve el
tiempo de viaje de la onda de sonido en microsegundos

  duration = pulseIn(echoPin, HIGH);  // Calculando la distrancia
```

```
  distance = duration * 0.034 / 2;     // Velocidad de la onda de sonido
dividida por 2 (ida y vuelta)

  Serial.println(distance);              // Manda el valor de la distancia
en el puerto serial
  }
}
```