ESCUELA POLITÉCNICA NACIONAL

FACULTAD DE INGENIERÍA ELÉCTRICA Y ELECTRÓNICA

IMPLEMENTACIÓN DE UN PROTOTIPO PARA EL MONITOREO DE UNA ESTACIÓN TRANSMISORA DE TV ANALÓGICA

TRABAJO DE TITULACIÓN PREVIO A LA OBTENCIÓN DEL TÍTULO DE INGENIERO EN ELECTRÓNICA Y TELECOMUNICACIONES

JONATHAN ALEXANDER NASIMBA LLUMIQUINGA

DIRECTOR: ING. CARLOS ROBERTO EGAS ACOSTA, MSC.

Quito, Abril 2023

AVAL

Certifico que el presente trabajo fue desarrollado por Jonathan Alexander Nasimba Llumiquinga, bajo mi supervisión.

Ing. Carlos Roberto Egas Acosta, Msc.
DIRECTOR DEL TRABAJO DE TITULACIÓN

DECLARACIÓN DE AUTORÍA

Yo Jonathan Alexander Nasimba Llumiquinga, declaro bajo juramento que el trabajo aquí descrito es de mi autoría; que no ha sido previamente presentado para ningún grado o calificación profesional; y, que he consultado las referencias bibliográficas que se incluyen en este documento.

A través de la presente declaración dejo constancia de que la Escuela Politécnica Nacional podrá hacer uso del presente trabajo según los términos estipulados en la Ley, Reglamentos y Normas vigentes.

JONATHAN ALEXANDER NASIMBA LLUMIQUINGA

DEDICATORIA

Dedico este trabajo a mis padres Alicia y César por su amor y cuidado durante toda mi vida, por su guía y esfuerzo por ofrecerme una vida llena de felicidad.

A mis hermanas Jeaneth, María, Marisol, Angelica, Erika, Milena y a mi hermano Patricio que me apoyaron durante mis estudios, que me cuidaron, protegieron y educaron a lo largo de mi vida.

AGRADECIMIENTO

A mis padres, hermanas y hermano que me brindaron su amor, sabiduría y paciencia, para mi han sido una fuente de amor e inspiración.

Les agradezco por su apoyo incondicional, por no dejarme solo en momentos difíciles de mi vida, por nunca dejarme rendir e incentivarme a conseguir mis metas personales.

A mi tutor Carlos Egas por orientar y supervisar este proyecto.

A Hernán Mora jefe de departamento de radio frecuencia en Teleamazonas y Christian Yanangómez técnico de radio frecuencia que guiaron el desarrollo de este proyecto.

ÍNDICE DE CONTENIDO

RESUMEN

Este trabajo presenta la implementación de un prototipo para el monitoreo de una estación repetidora de TV analógica, el cual se conforma por un circuito electrónico para la recolección de señales de voltaje DC, corriente DC, potencia directa, potencia reflejada y temperatura en un amplificador de potencia. Estas señales analógicas se procesan digitalmente con un módulo NodeMCU, el cual se programa empleando el IDE de Arduino, para ser mostradas en una pantalla LCD y enviadas inalámbricamente a un segundo módulo basado en ESP-32 que se denomina TCALL V1.4, el cual se encarga de enviar estos datos empleando WiFi o la red GSM.

Se debe mencionar que el módulo NodeMCU envía alertas hacia el módulo TCALL V1.4 cuando la temperatura y potencia reflejada sobrepasan límites preestablecido.

EL módulo TCALL V1.4 incorpora un banco de energía, además de un sensor PIR, el cual genera alertas cuando detecta movimiento al interior de la caseta de la estación repetidora de TV.

Todos los datos recolectados por el NodeMCU son almacenados en una base de datos de un servidor web compartido y son mostrados en una página web con control de usuarios para permitir solo el ingreso de los usuarios previamente registrados.

En el capítulo 1, se presentan las características de los dispositivos usados y sus principios de funcionamiento.

En el capítulo 2, se muestra la implementación del circuito recolector de señales analógicas, el diseño de los programas de los módulos ESP32 empleando el IDE de Arduino y la implementación de la base de datos y la página web.

En el capítulo 3, se realizan las pruebas de funcionamiento del sistema, la recolección de señales analógicas y cómo son enviadas hacia la base de datos, además de la generación de alertas y su envío por mensajes de texto usando la red GSM.

En el capítulo 4, se presentan las conclusiones y recomendaciones del proyecto.

PALABRAS CLAVE: ESP32, alertas, Arduino, base de datos, señales analógicas.

ABSTRACT

This work presents the implementation of a prototype for the monitoring of an analog TV repeater station, which is made up of an electronic circuit for the collection of signals of DC voltage, DC current, direct power, reflected power and temperature in a power amplifier. These analog signals are digitally processed with a NodeMCU module, which is programmed using the Arduino IDE, to be displayed on an LCD screen and wirelessly sent to a second module based on ESP-32 called TCALL V1.4, which is responsible for sending this data using Wi-Fi or the GSM network.

It should be mentioned that the NodeMCU module sends alerts to the TCALL V1.4 module when the temperature and the reflected power exceed pre-established limits.

The TCALL V1.4 module incorporates a power bank, as well as a PIR sensor that generates alerts when motion is detected inside the TV repeater station.

All the data collected by the NodeMCU is stored in a shared web server database and displayed on a web page with user control to allow only previously registered users to enter.

In chapter 1, the characteristics of the devices used and their operating principles are presented.

In chapter 2, the implementation of the signal collector circuit is shown, the design of the ESP32 module programs using the Arduino IDE and the implementation of the database and the web page.

In chapter 3, the functional tests of the system are carried out, the collection of analog signals and how they are sent to the database, in addition to the generation of alerts and their sending by text messages using the GSM network.

In chapter 4, the conclusions and recommendations of the project are presented.

KEYWORDS: ESP32, alerts, Arduino, database, analog signals.

1. INTRODUCCIÓN

Teleamazonas, una cadena de televisión ecuatoriana, cuenta con una estación matriz y 30 estaciones repetidoras para la transmisión y retransmisión de su señal de TV analógica. Estas estaciones están ubicadas en todo el territorio nacional.

Las estaciones repetidoras reciben la totalidad de la programación transmitida desde la estación matriz y la retransmiten simultáneamente para la recepción directa por el público en general [1]. La programación llega a las repetidoras a través de enlaces satelitales o enlaces de microondas. Luego, utilizando un transmisor, se modula y amplifica la señal en el canal correspondiente y se transmite por el aire en modalidad broadcast.

Para verificar el correcto funcionamiento de las estaciones repetidoras de TV analógica, se necesita monitorear señales de potencia directa, potencia reflejada, voltaje DC, corriente DC y temperatura del amplificador de potencia.

Debido a que geográficamente las repetidoras de televisión están separadas por grandes distancias de la estación matriz, se requiere un sistema de recolección de los datos antes mencionados para monitorear el funcionamiento adecuado de los equipos y enviar mensajes de alerta cuando los parámetros de las señales recolectadas salgan de los umbrales adecuados.

La transmisión de los datos recolectados se enviará de forma inalámbrica utilizando la red celular GSM o redes WiFi. Para esto, se ha seleccionado módulos de desarrollo ESP32 que incorporan un microcontrolador un módulos WiFi y un módulo GSM.

1.1 OBJETIVOS

El objetivo general es:

- Implementar un prototipo de adquisición y monitoreo basado en un módulo TCALL V1.4 en una estación de TV analógica, datos que se almacenaran en un servidor web.

Los objetivos específicos son:

- Estudiar las tecnologías requeridas para la implementación del prototipo.
- Implementar un prototipo electrónico basado en un ESP32 para la adquisición y procesamiento de señales del amplificador de potencia.
- Implementar un servidor web al cual los usuarios puedan acceder desde internet.

- Desarrollar la interfaz web en la que se represente de manera adecuada los datos almacenados en la base de datos.

1.2 ALCANCE

Este trabajo de titulación se enfoca en la implementación de un prototipo para la recolección de datos de una estación de TV analógica. En primer lugar, se desarrolla el hardware necesario para la recolección de las señales analógicas de los equipos de la estación transmisora de TV hacia el microcontrolador, el cual será un ESP32 con CPUs Xtensa LX6 de 32 bits [2]. Se recolectan señales de voltaje DC, corriente DC, potencia directa, potencia reflejada y temperatura del amplificador de potencia de la estación repetidora. Además, se controla el ingreso a la caseta de la repetidora. Todas estas señales analógicas son procesadas por el microcontrolador. Para la programación del módulo ESP32, se utiliza el entorno de desarrollo integrado (IDE) de Arduino. Estos datos se envían de forma inalámbrica usando el módulo SIM800 para comunicación GSM o a través de una conexión WiFi.

Todos los datos recolectados se almacenan en una base de datos de un servidor web. Para ello, se contrata un servicio de alojamiento web compartido. Se implementa una página web con control de usuarios para permitir el acceso solo a usuarios previamente registrados.

1.3 MARCO TEÓRICO

1.3.1 SISTEMAS TRANSMISORES DE TELEVISIÓN ANALÓGICA

El transmisor de señales de televisión se encarga de recibir la señal analógica en banda base, trasladarla a un canal de radiofrecuencia, amplificarla y finalmente transmitirla al aire. Está compuesto por los bloques excitador y amplificador de potencia, como se muestra en la Figura 1.1 [1].

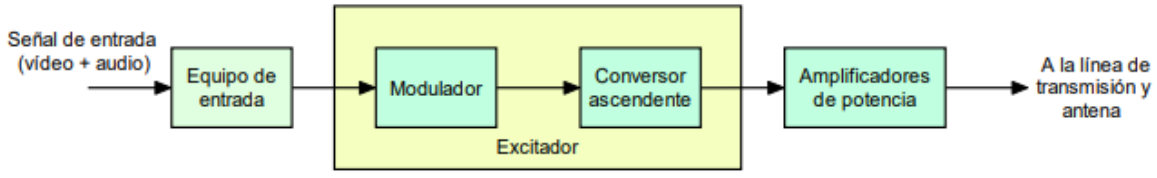


Figura 1.1 Sistema transmisor de amplificación común [1]

- **Equipo de entrada**

En el caso digital, la señal de entrada es un flujo binario único, mientras que en el caso analógico son una señal de video y una señal de audio.

- **Modulador**

El modulador se encarga de trasladar la señal de banda base a una frecuencia superior, que puede ser la frecuencia del canal de radio frecuencia o una frecuencia intermedia. La potencia obtenida a la salida del modulador es muy pequeña y se encuentra en el orden de fracciones de vatios.

- **Conversor ascendente**

Sirve para trasladar la señal a la frecuencia del canal de radiofrecuencia cuando se emplea una modulación en frecuencia intermedia. La salida del conversor es filtrada a la banda de paso de radio frecuencia deseada y se amplifica hasta un nivel de varios vatios. El conjunto de modulador y conversor ascendente se denomina excitador.

- **Amplificador de Potencia**

Se encarga de amplificar la señal de radio frecuencia entregada por el excitador. Para ello, se utilizan tecnologías de estado sólido o de válvulas de vacío, siendo estas últimas empleadas generalmente para potencias superiores a 10 KW. A la salida del amplificador de potencia es conectada una línea de transmisión para conducir la señal hasta la antena.

1.3.2 HERRAMIENTAS DE SOFTWARE

1.3.2.1 Arquitectura cliente-servidor

Se refiere a un conjunto de peticiones y respuestas generadas al utilizar una aplicación web o un servicio web, que el servidor procesa y resuelve para devolver una respuesta al cliente.

Cuando se aplica esta arquitectura en un sitio o aplicación web, el cliente es el navegador y en el lado del servidor pueden existir diferentes tecnologías para procesar y generar las respuestas que serán enviadas al cliente.

Algunas tareas realizadas por el servidor pueden incluir la consulta de datos almacenados en una base de datos o la obtención de información usando lenguajes de programación como Perl, ASP.NET o PHP [3].

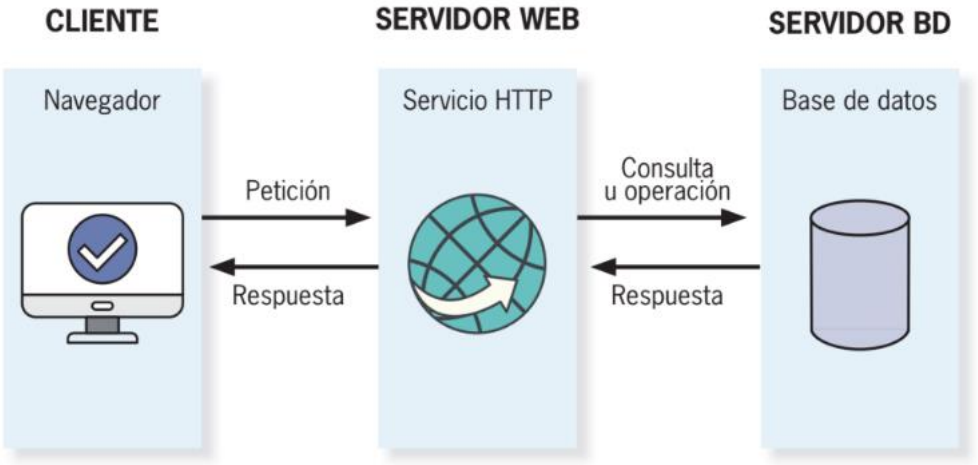


Figura 1.2 Representación cliente-servidor [3]

- **Cliente**

Es el encargado de enviar peticiones al servidor y mostrar las respuestas al usuario. Para desarrollar aplicaciones o páginas que se ejecutan en el cliente, se utilizan lenguajes y tecnologías como HTML, JAVASCRIPT, VISUAL BASICT SCRIT, CSS entre otras [3].

- **Servidor Web**

Es un programa informático encargado de procesar aplicaciones del lado del servidor, utilizando conexiones unidireccionales o bidireccionales con el cliente.

En el lado del servidor, se utilizan diferentes lenguajes como CSI, ASP, JSP, PHP entre otros [3].

1.3.2.2 Lenguaje PHP

Es un lenguaje de programación interpretado de alto nivel utilizado en el lado del servidor, Permitiendo procesar información en formularios, desarrollar páginas web, aplicaciones web entre otros. Es un lenguaje de código abierto y muy popular en el desarrollo web [3].

- **Estructura de una página PHP**

Un archivo .PHP puede tener solo código PHP o puede contener secciones de código HTML y secciones de código PHP, motivo por el cual se muestra la estructura básica de una página HTML, la cual comienza con la etiqueta <html> y termina con la etiqueta </html>. Dentro de estas etiquetas se encuentra la cabecera de la página, entre las etiquetas <head></head>, donde se coloca contenido que no pertenece a la página, por ejemplo, los estilos de la misma, seguida por la sección donde se coloca el contenido de la página, la cual se encuentra entre las etiquetas <body> </body>, en la que se coloca, por

ejemplo, texto, formularios, imágenes o código PHP el cual debe estar delimitado entre la etiqueta de apertura (<?php) y la etiqueta de cierre (?>).

- **Ejecución de un archivo PHP**

Para que el servidor ejecute un archivo con código PHP, este debe tener la extensión .PHP. De esta manera, el servidor ejecutará lo que se encuentra dentro de las etiquetas (<?php ; ?>) y el resultado se devolverá al navegador en forma de código HTML.

1.3.2.3 MySQL

Es un sistema de gestión de base de datos relacional que utiliza el lenguaje de consulta estructurado (SQL) y basa su funcionamiento en la arquitectura cliente-servidor .

Para crear una tabla se utiliza el comando CREATE TABLE, seguido por el nombre de la tabla, el cual debe comenzar con letras y no utilizar caracteres especiales, excepto el guion bajo.

En el marco de CREATE TABLE, entre paréntesis, se declaran primero los campos que conforman la tabla con nombres que empiezan con letras y sin caracteres especiales, excepto el guion bajo, seguido por el tipo de campo. Los principales tipos de campo son [4]:

- INT: admite valores numéricos enteros.
- REAL: permite valores numéricos con decimales.
- MONEY: el campo toma valor como moneda.
- CHAR: el campo tiene cadena de caracteres.
- VARCHAR: el campo puede tener letras y números.
- DATE: el campo puede tener valores con formatos de fecha.
- TIME: el campo puede tener valores con formato de tiempo que son representados por defecto , horas, minutos y segundos.
- BOOL: el campo puede tomar dos valores.

A continuación, se presenta un ejemplo de creación de tabla llamada Usuarios, la cual contienen los campos nombre y edad:

CREATE TABLE Usuarios (

nombre VARCHAR(10),

edad VARCHAR(10) [3].

)

- **Insertar datos en MySQL con PHP**

El siguiente ejemplo muestra cómo insertar datos en MySQL, para lo cual se considera la creación de una tabla en la base de datos llamada *Sensor,* que contiene los *campos value1, value2, value3*, tal como se muestra en la Figura 1.3 [5].

```
CREATE TABLE Sensor (
    id INT(6) UNSIGNED AUTO_INCREMENT PRIMARY KEY,
    value1 VARCHAR(10),
    value2 VARCHAR(10),
    value3 VARCHAR(10),
    reading_time TIMESTAMP DEFAULT CURRENT_TIMESTAMP ON UPDATE CURRENT_TIMESTAMP
)
```

Figura 1.3 Tabla SQL [5]

En la Figura 1.4 se muestra el uso del comando *mysqli,* el cual establece una conexión entre PHP y una base de datos MySQL [6]. Este comando requiere requiere los parámetros *$servername* (nombre del servidor donde se aloja la base de datos), *$username* (nombre de usuario de la base de datos), *$password* (contraseña de la base de datos) y *$dbname* (es el nombre de la base de datos).

```
// Create connection
$conn = new mysqli($servername, $username, $password, $dbname);
```

Figura 1.4 Comando de conexión [5]

En la Figura 1.5 se muestra cómo se insertan los datos en la tabla *Sensor* de la base de datos MySQL al emplear la declaración INSERT INTO [7]. Los valores de las columnas de la tabla *Sensor* (val1, val2, val3) son ingresados con los valores (‘” . $value1”’,’ ” . $value2”’,’ ” . $value3”’)

```
$sql = "INSERT INTO Sensor (value1, value2, value3)
VALUES ('" . $value1 . "', '" . $value2 . "', '" . $value3 . "')";
```

Figura 1.5 Código SQL para ingresar valores en un tabla [5]

1.3.2.4 Sistema de gestión de contenido

Un sistema de gestión de contenido (CMS), es una herramienta de software que ayuda a los usuarios a crear, administrar y modificar el contenido en un sitio web sin necesidad de escribir todo el código desde cero, los CMS están compuestos de dos partes principales, la primera es una aplicación de gestión de contenido (CMA) la cual permite agregar y administrar contenido en el sitio web y la segunda parte es una aplicación de entrega de contenido (CDA) que realiza un proceso backend que toma el contenido que ingresa en el CMA, lo almacena y lo hace visible para sus visitantes [8].

Algunos sistemas de gestión de contenido más populares son, WordPress, Joomla, Drupal, Magento, Squarespace y Wix.

1.3.2.5 IDE de Arduino

El entorno de desarrollo integrado (IDE) de Arduino es el conjunto de herramientas de software que permite a los programadores escribir y probar sus programas, permitiendo la identificación de errores durante la compilación del código para posteriormente grabarlo en la memoria permanente del microcontrolador.

La compilación en el IDE significa que el código escrito en lenguaje de Arduino es convertido en lenguaje de máquina el cual en esencia es un conjunto de impulsos eléctricos entendibles por el microcontrolador.

Un sketch es la denominación de un programa diseñado para ejecutarse sobre Arduino y consta de tres secciones, la sección de declaración de variables globales en la cual se escriben todas las diferentes clases de variables que se utilizaran en las dos siguientes secciones, la sección del *void setup()* que se ejecuta solo una vez al encender o resetear el microcontrolador y es empleada para realizar preconfiguraciones iniciales, y la tercera sección es el *void loop()* donde se coloca el programa principal el cual se ejecutará una y otra vez indefinidamente hasta que se apague o reinicie el microcontrolador [9].

El lenguaje de programación en el IDE de Arduino en una adaptación de lenguaje C++ resultante de avr-libc la cual proporciona una biblioteca de lenguaje C que en conjunto con los compiladores GCC considerados el estándar de los sistemas operativos derivados de UNIX son las herramientas necesarias para programar los microcontroladores de arquitectura AVR de Microchip [10] [11].

1.3.2.6 ESP-NOW

Es una tecnología de comunicación inalámbrica que opera en la frecuencia de 2.4 GHz fue desarrollada por espressif para la transmisión de paquetes cortos de hasta 250 bytes con velocidad de bits predeterminada a 1 Mbps.

Antes de la comunicación se debe realizar el emparejamiento de dispositivos, obteniendo una comunicación entre iguales de forma segura [12].

1.3.3 HERRAMIENTAS DE HARDWARE

1.3.3.1 Módulo NodeMCU

Es un módulo de propósito general basado en un ESP32-WROOM-32 que contiene un microprocesador de doble núcleo de 32 bits que emplea la frecuencia de reloj entre 80 y 240 MHz, el cual incluye Wi-Fi, bluetooth y bluetooth de baja energía.

El Wi-Fi tiene una potencia de salida de 20 dBm que proporciona gran alcance físico y conexión directa a internet a través de un enrutador, el uso del bluetooth permite la conexión a teléfonos o transmitir señales de baja energía para su detección [13], la distribución de pines del módulo NodeMCU se muestra en la Figura 1.6.

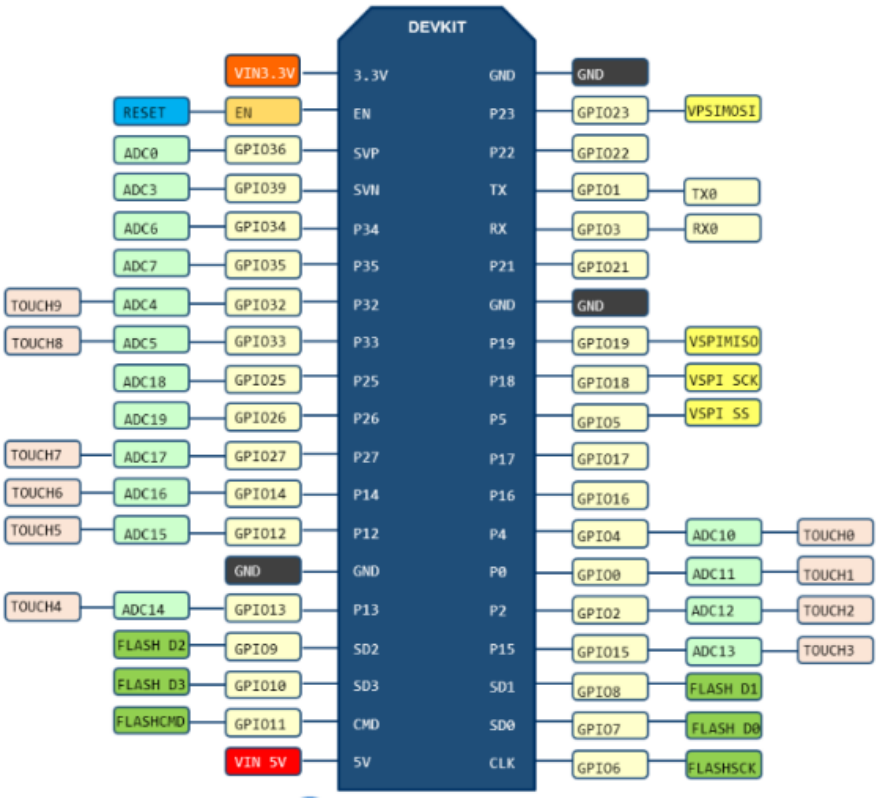


Figura 1.6 Definición de pines módulo NodeMCU [14]

Tabla 1.1 Principales especificaciones ESP32-WROOM-32 [13]**.**

Categoría	Ítem	Especificaciones
Wi-Fi	Protocolos	802.11 b/g/n (802.11 n hasta 150 Mbps)
	Rango de frecuencia central del canal de operación	2412 ~ 2484 MHz
Hardware	Interfaces	SD card, UART, SPI, I2C, PWM,
	Cristal integrado	40MHz
	Flash SPI integrada	4 MB
	Voltaje operativo	3.0V ~ 3.6V
	Corriente Operativa	Promedio: 80 mA
	Temperatura de trabajo	–40 °C ~ +85 °C

1.3.3.2 Módulo TCALL V1.4

Es un módulo incorpora el ESP32-WROVER-B y el SIM800 para realizar conexiones inalámbricas GSM.

El ESP32-WROVER-B incluye Wi-Fi, Bluetooth y Bluetooth de baja energía (BLE), lo que permite su uso en diversas aplicaciones, desde sensores de baja potencia hasta tareas más exigentes como son la codificación de voz, transmisión de música y decodificación MP3. La distribución de pines del módulo se muestran en la Figura 1.7 [15]**.**

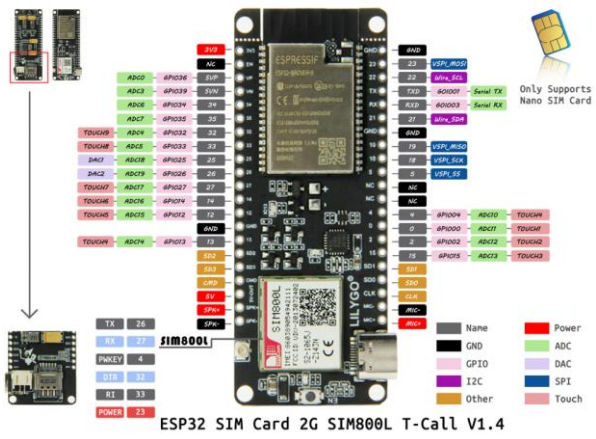


Figura 1.7 Distribución de pines módulo TCALL V1.4 [2]

Tabla 1.2 Principales especificaciones técnicas del ESP32-WROVER-B [2]

Especificaciones de Hardware	
Microprocesador	Espressif ESP32 240MHz Xtensa de 32bits LX6
FLASH	QSPI flash 4MB / PSRAM 8MB
SRAM	520 KB SRAM
Interfaces	UART, SPI, I2C, PWM
Cristal integrado	40MHz
Voltaje de trabajo	2.7V-3.6V
Corriente de trabajo	Promedio 70mA
Corriente en dormido	Promedio 1.1mA
SIM CARD	Soporta solo tarjeta nano SIM
Rango de temperatura de trabajo	-40℃ ~ +85℃
Especificaciones de la fuente de alimentación	
Fuente de alimentación	USB 5V/1A
Corriente de carga	500mA
Batería	Batería de litio de 3.7V
WiFi	
Protocolo	802.11 b/g/n(802.11n con velocidad de hasta 150 Mbps)
Rango de frecuencia	2.4GHz~2.5GHz(2400M~2483.5M)
Potencia de transmisión	22dBm
Distancia de comunicación	300m

1.3.3.3 Sensor de temperatura LM35

Es un sensor de temperatura de precisión cuyo voltaje de salida es directamente proporcional a la temperatura expresada en grados centígrados, a temperatura ambiente tiene una precisión de ± 1⁄4 ˚C mientras que en el rango de -55 a +150 ˚C la precisión es de ± 3⁄4 ˚C, en funcionamiento consume 60 µA por lo que el autocalentamiento es muy bajo, de alrededor de 0.1 ˚C cuando no hay circulación de aire [16].

Tabla 1.3 Características técnicas LM35 [16]**.**

Factor de escala	+10.0 mV / ˚C
Rango de operación	-55˚ a +150 ˚
Voltaje de operación	4 a 30 voltios
Impedancia de salida	0.1 Ω para una carga de 1 mA

1.3.3.4 Sensor de movimiento PIR HC-SR501

El sensor pasivo infrarrojo (PIR) detecta el movimiento basándose en la variación de las radiaciones infrarrojas que los cuerpos emiten en función de la temperatura que producen, esta radiación infrarroja es capturada y transformada en señal eléctrica. El sensor PIR consta de dos áreas para la recepción de la radiación, si ambas áreas reciben la misma cantidad de luz infrarroja, la señal resultante es nula. Por el contrario, si son diferentes, se genera un desequilibrio entre ambas áreas de recepción y se produce una señal eléctrica [17].

Tabla 1.4 Características técnicas del sensor PIR HC-SR501 [18]**.**

Voltaje de alimentación	5 a 12 voltios
Consumo promedio	Menor a 1 miliamperio
Rango de distancia de detección	De 3 a 7 metros
Angulo de detección	110°
Voltaje de salida	3.3 voltios
Temperatura de operación	-15° a +70°

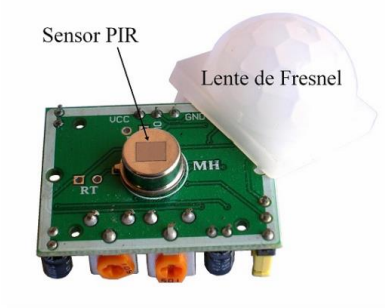


Figura 1.8 Sensor de movimiento PIR HC-SR501 [18]

1.3.3.5 Pantalla LCD

Es una pantalla LCD de 16x2 líneas con control de contraste ajustable, con luz led de fondo y emplea la interfaz de comunicación I2C.

La interfaz I2C fue diseñada para permitir la comunicación entre múltiples circuitos digitales, denominados “esclavos”, con uno o más circuitos digitales denominados “maestros”. Es una comunicación de corta distancia que utiliza dos señales: la primera es la señal de reloj (SCL) y la segunda es la señal de datos (SDA), el protocolo permite un espacio de direccionamiento de hasta 10 bits y una velocidad de transferencia entre 100Hz y 5KHz [19].

Figura 1.9 Pantalla I2C 16x02 [20]

2. METODOLOGÍA

2.1 REQUERIMIENTOS DEL SISTEMA

2.1.1 HARDWARE

Para el diseño del hardware se considera el monitoreo de los parámetros de voltaje DC, corriente DC, potencia directa, potencia reflejada y temperatura en el amplificador de potencia, además de la notificación cuando se detecte movimiento en la caseta de la estación repetidora de TV.

El monitoreo de las señales antes mencionadas servirá para controlar que sus parámetros estén dentro de los umbrales de funcionamiento permitidos, con el propósito de que el módulo NodeMCU ejecute acciones dependiendo de los datos obtenidos durante el monitoreo de estas señales analógicas.

El NodeMCU debe recolectar las señales analógicas, procesarlas y compararlas con umbrales predefinidos para determinar el funcionamiento correcto del amplificador de potencia o ejecutar una acción para evitar daños en el equipo.

Debido a que las repetidoras se ubican generalmente en cerros y montañas de difícil acceso, el envío de datos se realizará a través de las redes GSM de telefonía celular usando el SIM800 que ofrece esta comunicación inalámbrica, o mediante redes WiFi disponibles en determinadas estaciones repetidoras.

2.1.2 BASE DE DATOS

El muestreo de las señales analógicas antes mencionadas se realiza de forma constante una tras otra, sin embargo, los datos recolectados solo serán enviados a la base de datos una vez por día y cuando estas señales salgan de los umbrales preestablecidos y se generen alertas.

El almacenamiento en la base de datos y hosting será contratado en un servidor de hosting compartido, datos que estarán disponibles desde cualquier lugar con conexión a internet.

2.1.3 SITIO WEB

El sitio web debe contener una página de inicio y una página para mostrar la información de la base de datos, dicha página debe tener un control de usuarios para permitir el ingreso solo de usuarios registrados previamente.

2.2 ARQUITECTURA GENERAL DEL SISTEMA

En la Figura 2.1 se muestra el diagrama de bloques del prototipo de monitoreo de estación repetidora de TV analógica. El primer bloque, PCB1, mostrado en la Figura 2.3, recolecta las señales a monitorear del amplificador de potencia, las cuales son: voltaje DC, corriente DC, potencia directa, potencia reflejada y temperatura. Se emplea el conversor analógico digital a través de las entradas analógicas del módulo NodeMCU, y las señales analógicas son procesadas digitalmente para determinar que se encuentren dentro de los umbrales de funcionamiento predefinidos. En caso contrario, se generan alertas a través de los pines digitales del NodeMCU.

Los datos recolectados se envían hacia el bloque PCB2 empleando el protocolo esp-now, una vez por día y cuando se generen alertas debido a temperatura y potencia reflejada elevadas.

El bloque PCB2 incluye un sensor de movimiento y el módulo TCALL V1.4 mostrado en la Figura 2.2. Este último incorpora un ESP32-WROVER-B y un módulo SIM800 para conexión inalámbrica GSM. Los datos recibidos mediante el protocolo de comunicación inalámbrica esp-now son reenviados hacia la base de datos mediante las tecnologías WiFi o GSM.

El bloque PCB2 se energiza con una fuente de 15 voltios y cuenta con una batería de respaldo. El módulo TCALL V1.4, al recibir las alertas desde el módulo NodeMCU, las reenvía como mensajes de texto empleando el módulo SIM800. Además, envía mensajes de texto de alerta cuando se ha detectado movimiento al interior de la caseta de la estación repetidora.

El bloque “Sitio Web” de la Figura 2.1 representa la página web de inicio y la página web donde se muestran los valores de la base de datos que son recolectados por los módulos NodeMCU y TCALL V1.4. La página debe tener un control de usuarios para permitir el ingreso solo de usuarios registrados previamente.

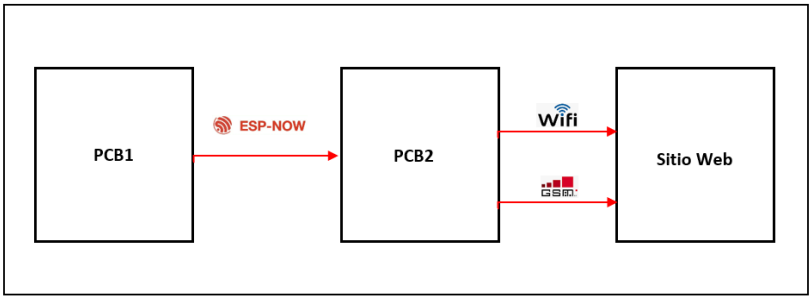


Figura 2.1 Arquitectura general del proyecto

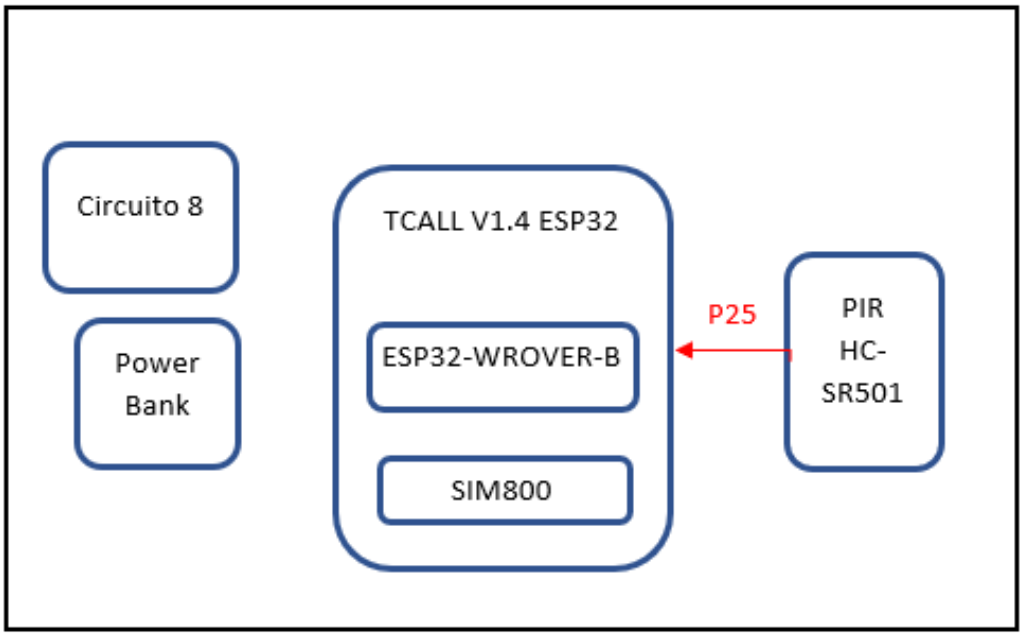


Figura 2.2 Bloque PCB2 de la arquitectura general del proyecto

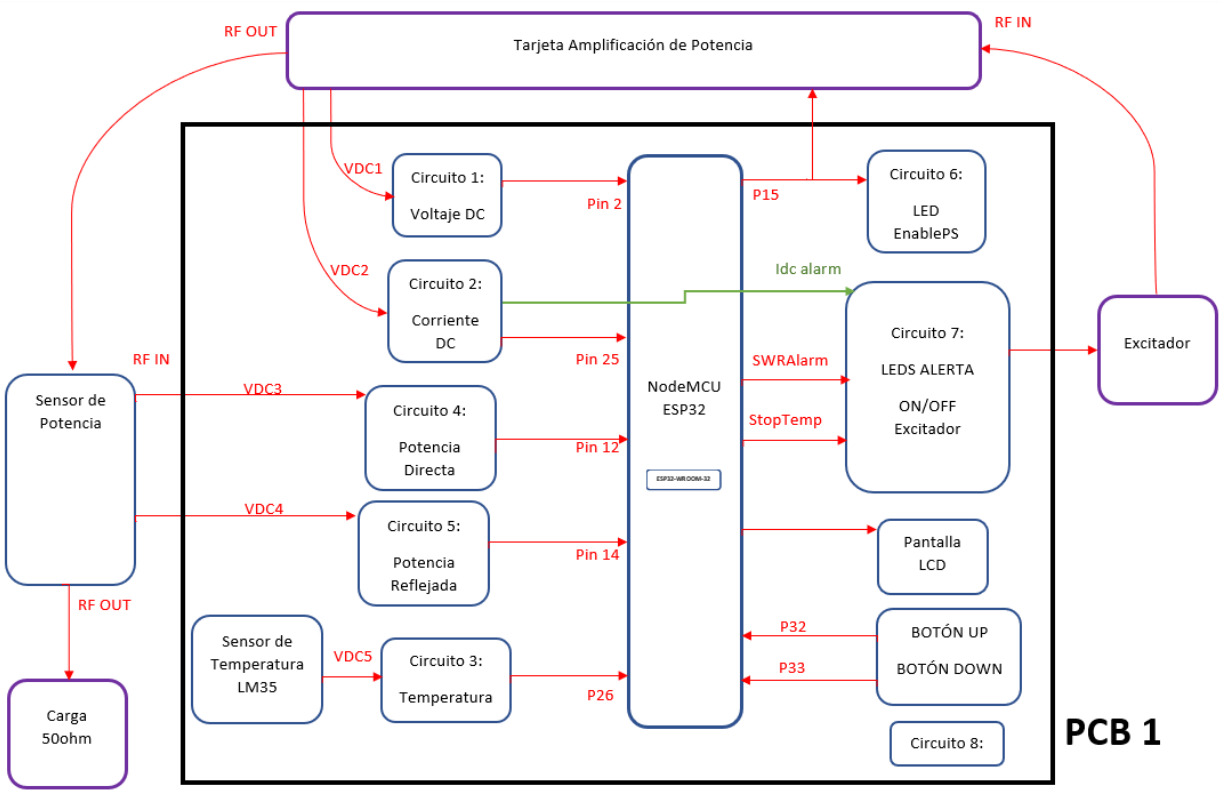


Figura 2.3 Bloque PCB1 de la arquitectura general del proyecto

2.3 DISEÑO DEL HARDWARE DEL SISTEMA

El diseño de los circuitos electrónicos se implementó en el software ISIS, mientras que el diseño de las tarjetas de circuito impreso (PCB) se realizó en el software ARES.

Para la implementación de los circuitos PCB1 y PCB2 se utilizó como referencia el circuito multímetro del transmisor de estado sólido modelo S/FNL-1000/3 y el circuito 405300012 para medir corriente del transmisor ITELCOT313 que se muestran en el ANEXO A.

2.3.1 DISEÑO DEL CIRCUITO PCB1

En la Figura 2.4 muestra la conexión de las entradas analógicas 2, 25, 12, 14 y 26 del NodeMCU a las señales de voltaje DC (VOLTAJEDC), corriente DC (IDC), potencia directa (DIRECTA), potencia Reflejada (REFLEJADA) y temperatura (TEMPERATURA) respectivamente.

Las conexiones a Los pines digitales del NodeMCU son las siguientes:

UP y DOWN: entradas digitales 32 y 33 configuradas como interrupciones, sirven para desplazarse por el menú que se muestra en la pantalla LCD.

SDA y SCL: pines para el control de la pantalla de cristal líquido (LCD) mediante el protocolo I2C.

EnablePS: pin 15 controla el arranque del amplificador de potencia.

STOP TEMP: pin 4 salida digital que genera una señal lógica en alto cuando la temperatura es igual o mayor a 70 grados centígrados.

SWR ALARM: pin 13 genera una señal en alto cuando la potencia reflejada es igual o mayor a 50 vatios.

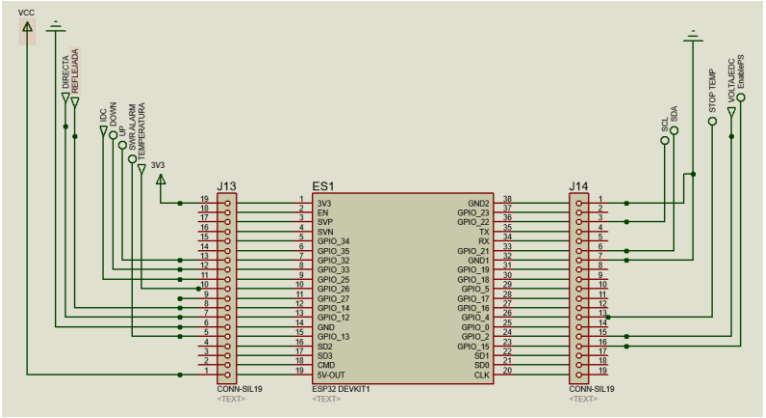


Figura 2.4 Conexión de entradas analógicas y salidas digitales del NodeMCU

El circuito PCB1 está energizado por una fuente de voltaje DC de 15 voltios y se emplea los reguladores de voltaje L7812 y L7805 para proporcionar voltajes de alimentación de 12 y 5 voltios, respectivamente, como se muestra en la Figura 2.5.

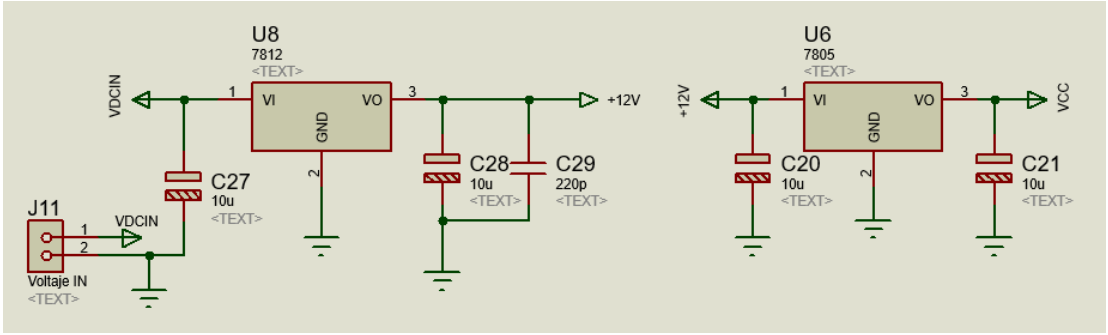


Figura 2.5 Circuito 8 de la PCB1

El circuito 6 de la PCB1, mostrado en la Figura 2.6, tiene como función indicar mediante un LED el estado de la tarjeta electrónica de amplificación de potencia.

Cuando el pin EnablePS está en estado bajo, el LED se enciende para mostrar que el funcionamiento es correcto. Por otro lado, cuando el pin está en estado alto, el LED se apaga, indicando que la tarjeta de amplificación de potencia no está trabajando.

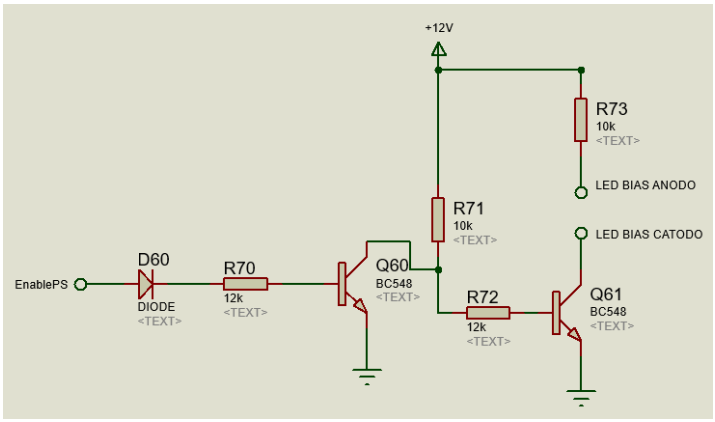


Figura 2.6 Circuito 6 de la PCB1

El circuito 1 de la PCB1 es un circuito divisor de voltaje que reduce el voltaje de entrada proveniente de la fuente de 50 voltios de la tarjeta de amplificación de potencia, permitiendo que sea medida por el pin analógico del NodeMCU. Esto garantiza que el voltaje máximo que ingresa por el pin del microcontrolador no supere los 3.6 voltios, que es el voltaje máximo de operación del ESP32-WROOM-32 que está integrado en el módulo NodeMCU.

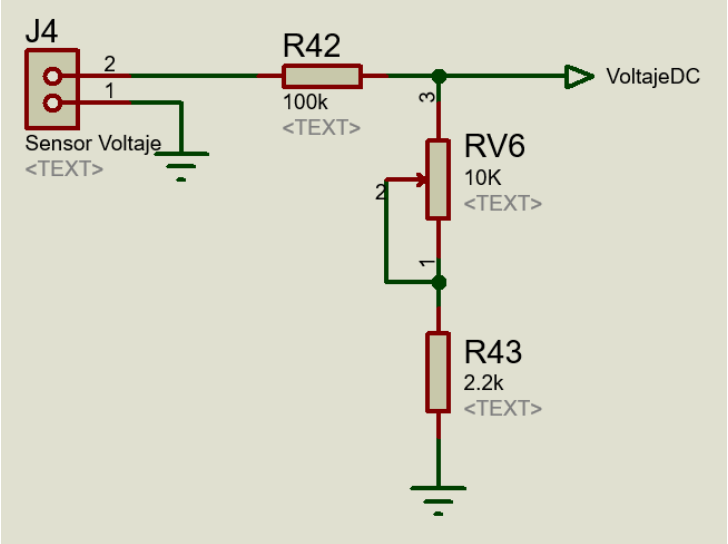


Figura 2.7 Circuito 1 de la PCB1

El circuito 3 de la PCB1, mostrado en la Figura 2.8, es el circuito encargado de amplificar el voltaje DC proporcionado por el sensor LM35. El sensor se conecta al conector J12, cuya salida está conectada al pin no inversor del amplificador operacional.

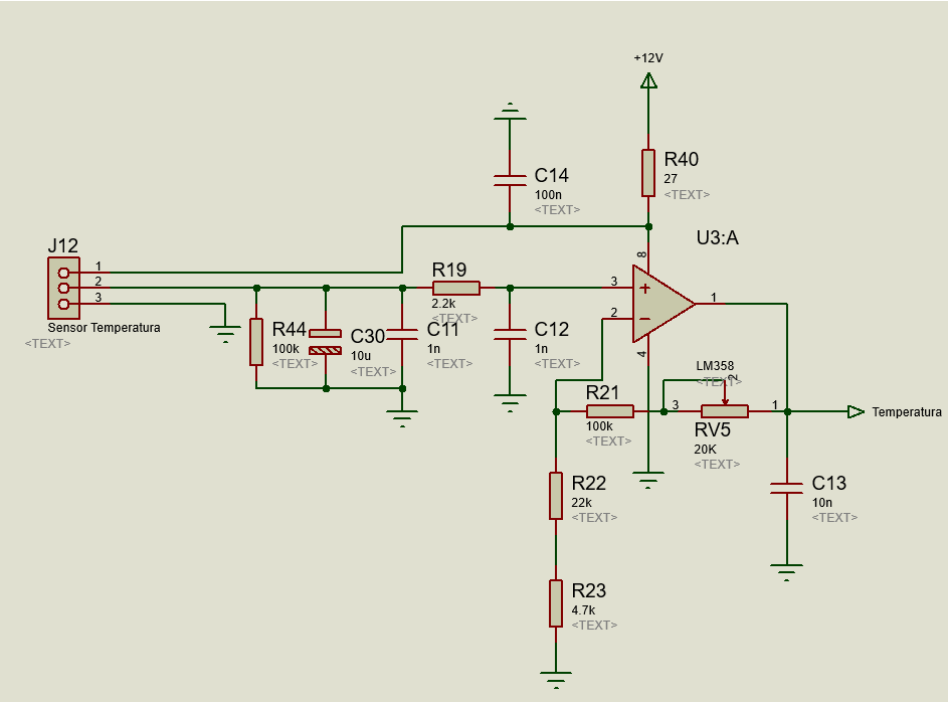


Figura 2.8 Circuito 3 de la PCB1

El circuito 2 de la PCB1, mostrado en la Figura 2.9, consta de dos partes. La primera es un circuito para la medición de corriente que utiliza un resistor derivador de 10 mili ohmios colocado en los terminales J6 y J7, que produce un voltaje proporcionalmente bajo. Para

amplificar este voltaje DC, se utiliza un amplificador operacional CA3240. Este voltaje amplificado entra al pin 25 del NodeMCU.

La segunda parte es un amplificador operacional CA3240 que compara el voltaje de salida del amplificador U7:A con un voltaje de referencia que ingresa por el pin inversor del amplificador operacional, cuyo valor es configurado con el potenciómetro RV8. El propósito de esta segunda parte es generar una alarma cuando se supere el voltaje de referencia.

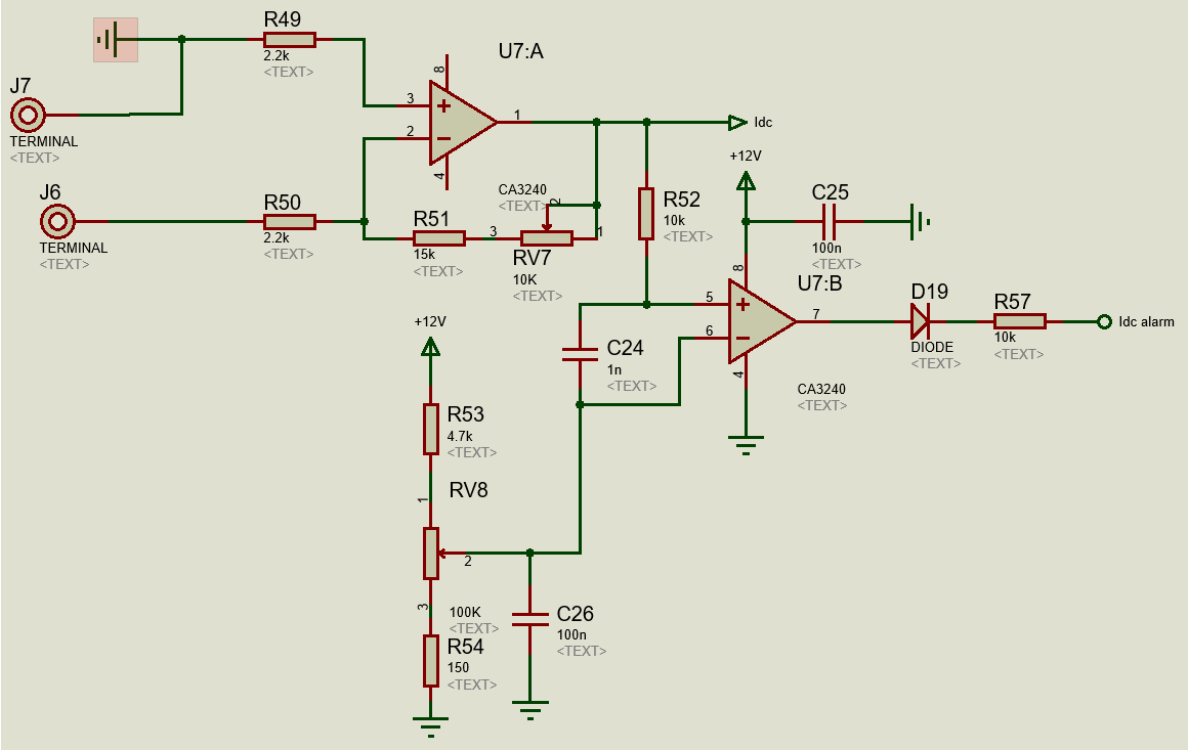


Figura 2.9 Circuito 2 de la PCB1

En la Figura 2.10 se muestra el circuito 4 de la PCB1, el cual utiliza dos amplificadores operacionales para obtener una realimentación negativa, lo que permite estabilizar la ganancia y la respuesta en frecuencia de la señal obtenida del sensor de potencia correspondiente a la potencia directa.

En la Figura 2.11 se muestra el circuito 5 de la PCB1, el cual tiene el mismo funcionamiento que el circuito de la Figura 2.10 y se utiliza con la señal obtenida del sensor de potencia correspondiente a la potencia reflejada.

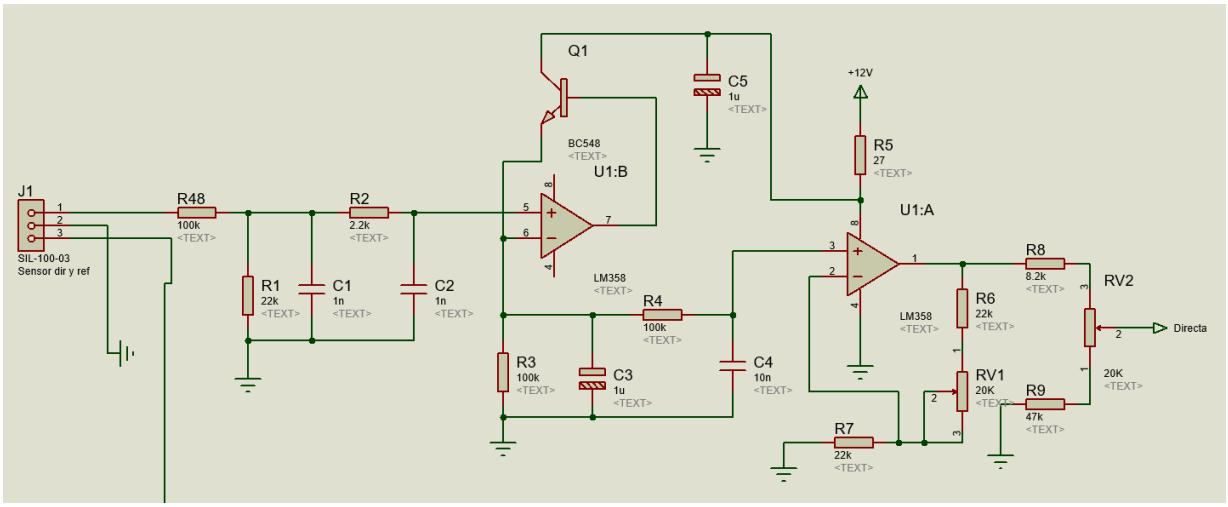


Figura 2.10 Circuito 4 de la PCB1

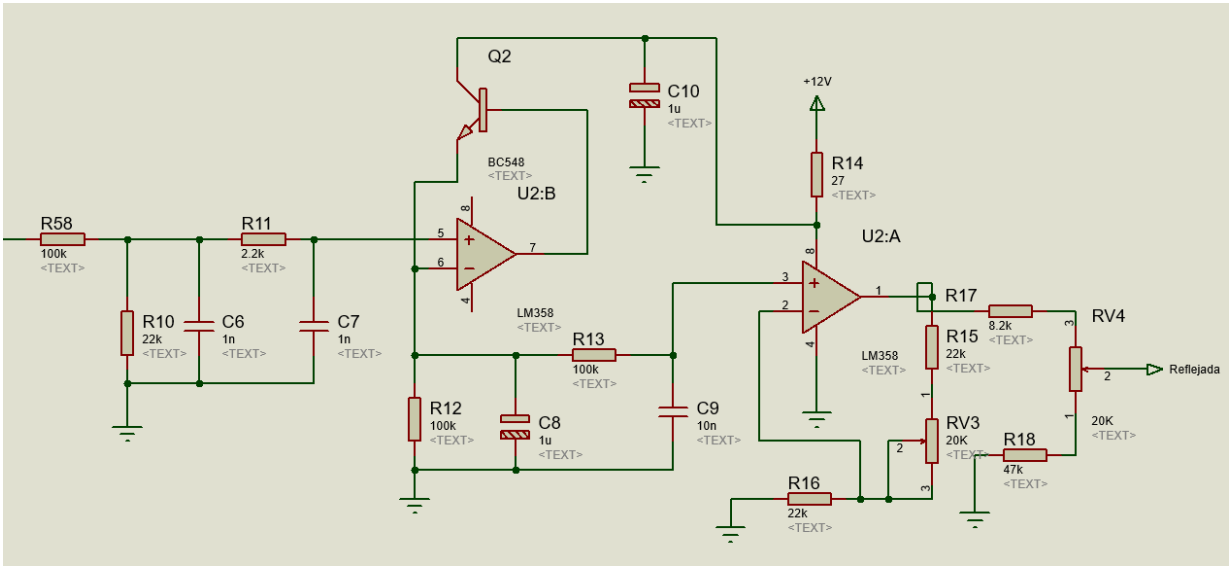


Figura 2.11 Circuito 5 de la PCB1

El circuito 7 de la PCB1, mostrado en la Figura 2.12, representa el esquema de las alarmas. Cuando se produce una alarma de temperatura elevada (StopTemp), corriente elevada (idc alarm) o potencia reflejada elevada (SWRAlarm), se deshabilita el excitador. Además, al generarse las alertas de temperatura y potencia reflejada, también se desactiva la tarjeta de amplificación de potencia.

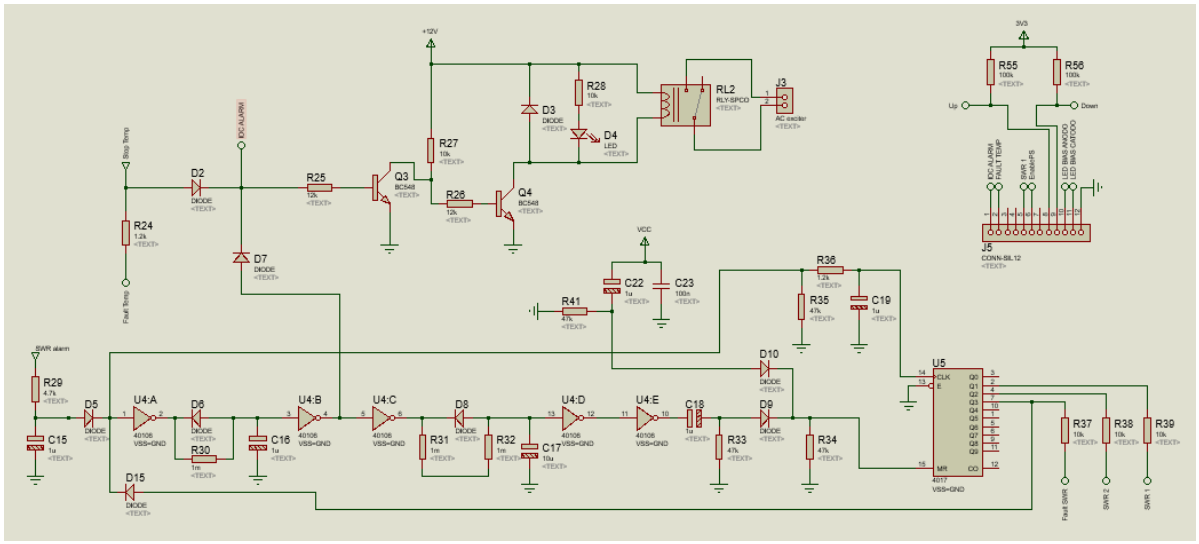


Figura 2.12 Circuito 7 de la PCB1

2.3.2 DISEÑO DEL CIRCUITO PCB2

La Figura 2.13 muestra el diagrama esquemático para las conexiones de la PCB2, que contiene el TCALL V1.4 con el sensor PIR y el sistema de respaldo de batería. Esta PCB está conectada simultáneamente a la fuente de alimentación de 15 voltios y a la batería. Cuando la fuente de voltaje de 15 voltios no está funcionando, se realiza la conmutación hacia la batería para energizar la PCB2.

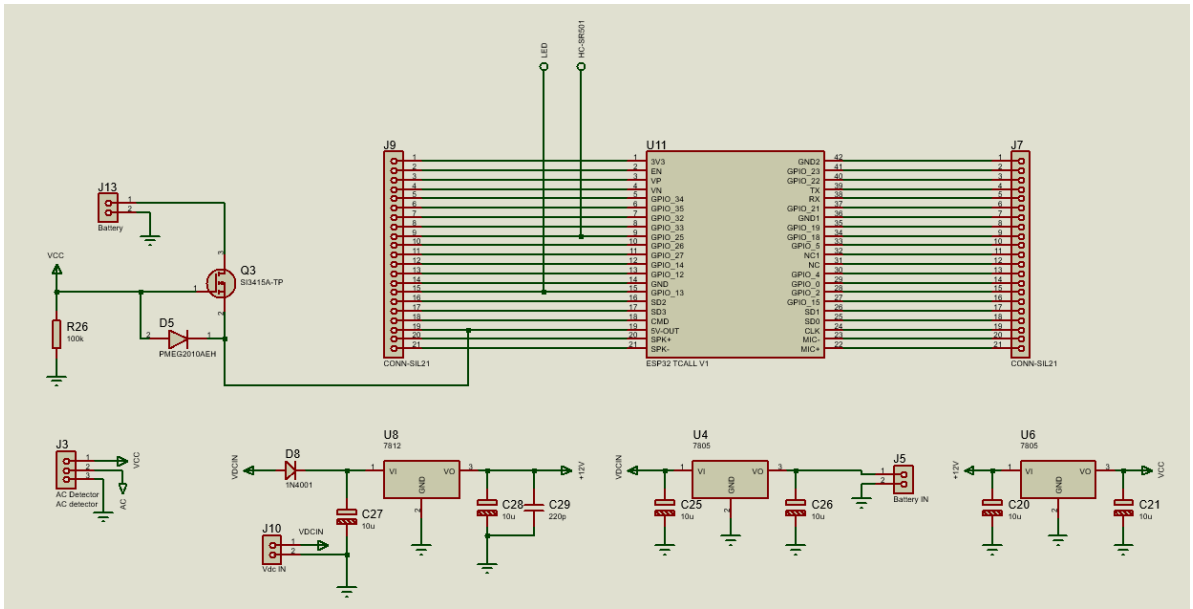


Figura 2.13 Circuito PCB2

2.4 DISEÑO DEL SOFTWARE DEL SISTEMA

2.4.1 DIAGRAMA DE FLUJO DEL PROGRAMA DEL MÓDULO NODEMCU

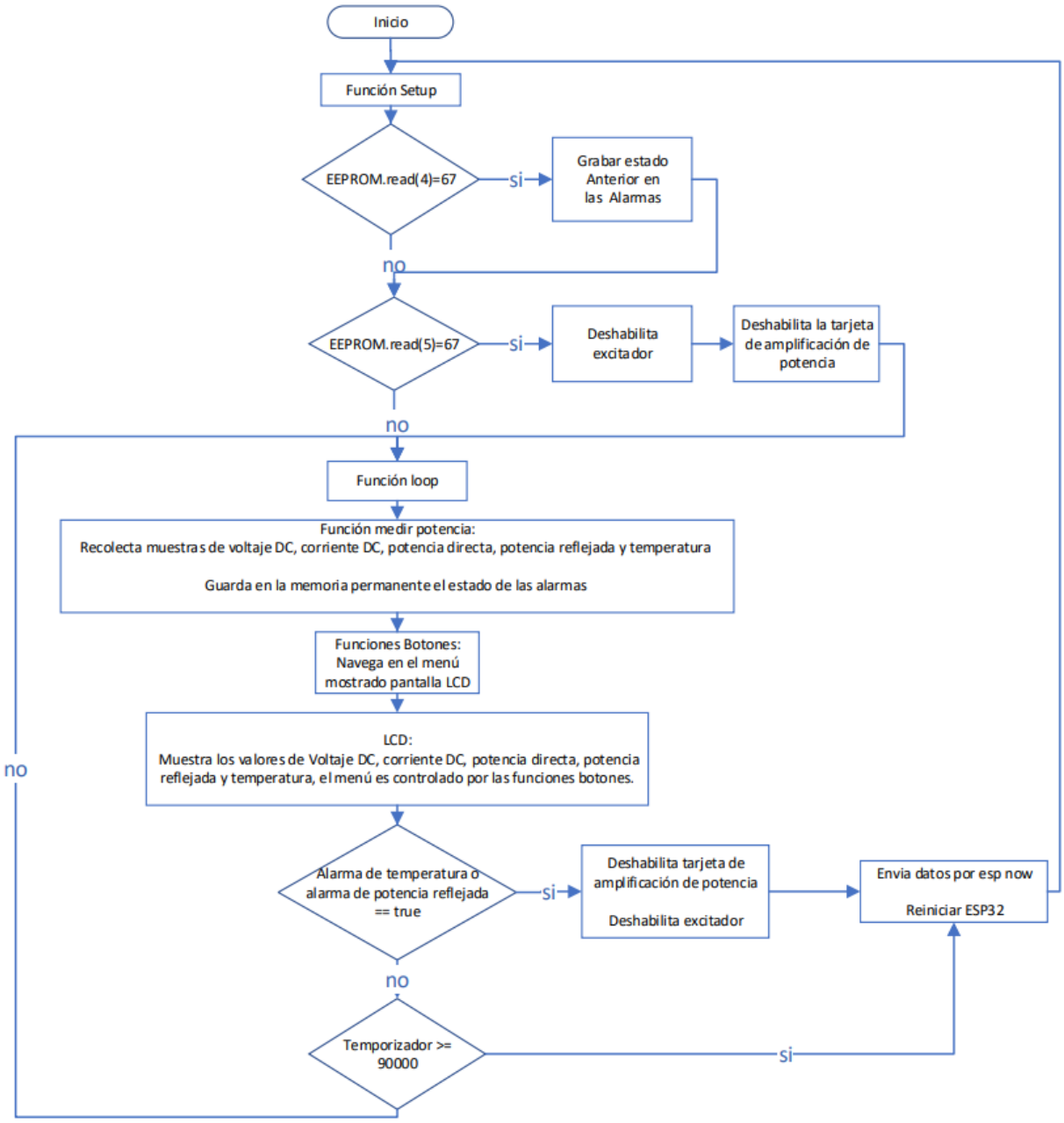


Figura 2.14 Diagrama esquemático del programa implementado en el IDE de Arduino para el módulo NodeMCU

2.4.2 PROGRAMACIÓN DEL NODEMCU

Para programar el módulo NodeMCU, es necesario descargar e instalar los complementos del ESP32 para el IDE de Arduino.

El código completo del programa implementado en el módulo NodeMCU se muestra en el ANEXO B.

Para instalar y configurar el paquete ESP32 compatible con el IDE de Arduino se debe seguir el siguiente proceso.

- Descargar la versión de Arduino 1.8 o versiones superiores desde la página oficial https://www.arduino.cc.
- Instalar el archivo ejecutable del ide de Arduino descargado.
- Iniciar el IDE de Arduino y abrir la ventana de *preferencias*.
- En el campo *Gestor de URLs Adicionales de Tarjetas* ingresar el siguiente enlace: https://raw.githubusercontent.com/espressif/arduino-esp32/gh-pages/package_esp32_index.json
- Seleccionar la pestaña *Herramientas*, después *Placa*, luego *Gestor de tarjetas* e introducir en el buscador *esp32* y presionar en el botón instalar.
- Seleccionar la pestaña de *Herramientas,* luego *Placa* y seleccionar *ESP32 Arduino* y finalmente seleccionar la tarjeta *ESP32 Dev Module.*

En general, el programa ejecutado en el módulo NodeMCU recolecta continuamente valores de voltaje DC, corriente DC, temperatura, potencia directa y potencia reflejada. Estos valores son mostrados al usuario mediante una pantalla LCD de 16x2 segmentos, además de ser enviados a través del protocolo esp-now al módulo TCALL V1.4 en un intervalo de tiempo definido por el usuario. Por motivo de pruebas y validación del funcionamiento del programa, los datos se enviarán cada 90 segundos y también se enviarán cuando se generen las alarmas de temperatura y potencia reflejada. Para realizar la comunicación empleando el protocolo esp-now, es necesario conocer la dirección MAC del módulo receptor, que en este caso es el módulo TCALL V1.4 como se indica en la Figura 2.15.

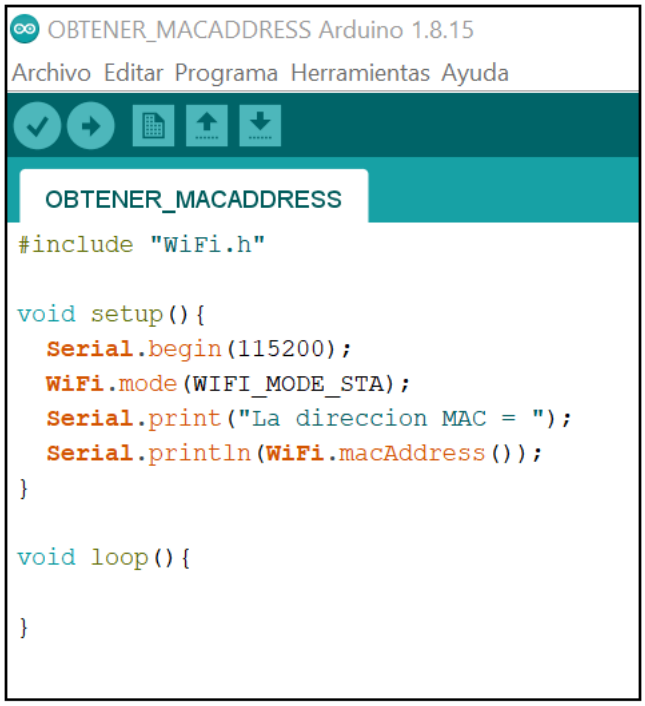


Figura 2.15 Código para la obtención de la dirección MAC del módulo TCALL V1.4

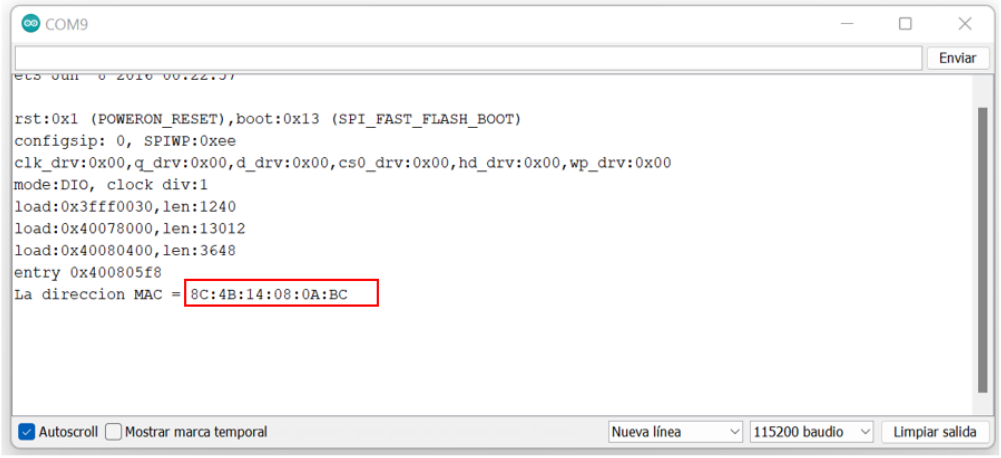


Figura 2.16 Dirección MAC del módulo TCALL V1.4

En la Figura 2.17 se muestran las librerías necesarias para el desarrollo del programa, las cuales son:

```
#include <esp_now.h>
#include <WiFi.h>
#include <Wire.h>
#include <EEPROM.h>
#include <LiquidCrystal_I2C.h>
```

Figura 2.17 Librerías empleadas en el programa del NodeMCU

- #include <esp_now.h>: librería para la comunicación inalámbrica entre dispositivos de igual a igual de Los módulos ESP32.
- #include <WiFi.h>: librería para controlar el módulo WiFi del ESP32.
- #include <Wire.h>: permite la comunicación a través del protocolo I2C.
- #include <EEPROM.h>: permite mantener los datos en la memoria permanente al desconectar la placa de la fuente de alimentación o después de reiniciar la placa ESP32.
- #include <LiquidCrystal_I2C.h>: librería que permite controlar pantallas con el protocolo I2C.

La Figura 2.18 muestra la asignación de pines del módulo NodeMCU, los pines 32 y 33 son usados como interrupciones externas empleadas por dos pulsadores que permiten moverse entre el menú indicado en la pantalla LCD, los pines 4 y 13 son salidas digitales, que se utilizan para desactivar el excitador cuando la temperatura y potencia reflejada sobrepasan los límites máximos de 70 °C y 50 W respectivamente.

El pin 15 es una salida digital que permite deshabilitar la tarjeta de amplificación de potencia cuando la temperatura y potencia reflejada sobrepasan los limites predefinidos.

Los pines 26, 2, 25, 12 y 14 son entradas para la recolección de los datos de temperatura, voltaje DC, corriente DC, potencia directa y potencia reflejada respectivamente.

```
// Variables para las interrupciones
#define PIN_INT_UP            32
#define PIN_INT_DOWN          33

// PINES DE LAS SALIDAS DIGITALES
#define StopTemp              4
#define SWRAlarm              13
#define EnablePS              15

//Define entradas analógicas
#define PIN_TEMPERATURA       26
#define PIN_VOLTAJE           2
#define PIN_IDC               25
#define PIN_POT_DIRECTA       12
#define PIN_POT_REFLEJADA     14
```

Figura 2.18 Asignación de pines en el módulo NodeMCU

La Figura 2.19 muestra la configuración del ESP32 para trabajar con el protocolo esp-now [21]:

- WiFi.mode(WIFI_STA): configura el módulo WiFi como estación.
- esp_now_init(): inicializa la función esp-now y devuelve ESP_OK cuando la inicialización fue exitosa y devuelve ESP_ERR_ESPNOW_INTERNAL cuando hay un error interno.
- esp_now_register_send_cb(OnDataSent): registra la función de devolución de llamada de envió del dato esp-now.
- esp_now_add_peer(&peerInfo): añade un par a la lista de pares.

```
WiFi.mode(WIFI_STA);
// Init ESP-NOW
if (esp_now_init() != ESP_OK) {
  Serial.println("Error inicializando ESP-NOW");
  lcd.clear();
  lcd.print("ESP NOW ERROR");
  delay(500);
  return;
}
esp_now_register_send_cb(OnDataSent);
memcpy(peerInfo.peer_addr, broadcastAddress, 6);
peerInfo.channel = 0;
peerInfo.encrypt = false;

if (esp_now_add_peer(&peerInfo) != ESP_OK) {
  Serial.println("Failed to add peer");
  return;
}
```

Figura 2.19 Configuración esp-now

La Figura 2.20 muestra el envío de los datos recolectados por los pines analógicos hacia el módulo TCALL V1.4.

- esp_now_send(broadcastAddress, (uint8_t *) &sensordatos, sizeof(sensordatos)): La función envía los datos de la estructura *sensordatos* por el protocolo esp-now a la dirección MAC del módulo TCAL V1.4, y se imprime mensajes en el monitor serial y en la pantalla LCD informando si el envío fue exitoso o fallido.

```
//Envia los datos medidos
esp_err_t result = esp_now_send(broadcastAddress, (uint8_t *) &sensordatos, sizeof(sensordatos));

if (result == ESP_OK) {
  Serial.println("Enviado con exito");
  lcd.clear();
  lcd.print("SENT ESP NOW");
  delay(500);
}
else {
  Serial.println("Erros enviando los datos");
  lcd.clear();
  lcd.print("FAIL SENT ESP NOW");
  delay(500);
}
```

Figura 2.20 Envío de mensaje por esp-now

La Figura 2.21 muestra el código para la recolección y procesamiento de los datos a través de las entradas analógicas del módulo NodeMCU. El proceso consiste en tomar 100 muestras empleando la instrucción *analogRead(),* de las 100 muestras se obtiene el promedio para evitar valores altos instantáneos de las muestras que no corresponden a los valores reales.

```
for (int i = 0; i < 100; i++)
{
  MEDICION_POT_DIRECTA = (analogRead(PIN_POT_DIRECTA) * 3.3) / 4095.0;
  MEDICION_POT_REFLEJADA = (analogRead(PIN_POT_REFLEJADA) * 3.3) / 4095.0;
  MEDICION_TEMPERATURA = (analogRead(PIN_TEMPERATURA) * 3.3 * 20) / 4095.0;
  MEDICION_IDC = (analogRead(PIN_IDC ) * 3.3 ) / 4095.0;
  MEDICION_VOLTAJE = (analogRead(PIN_VOLTAJE) * 3.3 ) / 4095.0;

  MAX_POT_REFLEJADA += MEDICION_POT_REFLEJADA * MEDICION_POT_REFLEJADA;
  MAX_POT_DIRECTA += MEDICION_POT_DIRECTA * MEDICION_POT_DIRECTA;
  MAX_TEMPERATURA += MEDICION_TEMPERATURA * MEDICION_TEMPERATURA;
  MAX_IDC += MEDICION_IDC * MEDICION_IDC;
  MAX_VOLTAJE += MEDICION_VOLTAJE * MEDICION_VOLTAJE;
}
PROMEDIO_POT_REFLEJADA = sqrt(MAX_POT_REFLEJADA / 100);
```

Figura 2.21 Recolección de valores de señales analógicas

Los valores promedio obtenidos son:

- PROMEDIO_VOLTAJE
- PROMEDIO_IDC
- PROMEDIO_TEMPERATURA
- PROMEDIO_POT_DIRECTA
- PROMEDIO_POT_REFLEJADA

Para obtener los valores finales de voltaje DC, corriente DC y temperatura que se muestran en la pantalla LCD se emplea una relación lineal en la cual se multiplica los valores promedio por las constantes K1=56, K2=10 y K3=20 respectivamente.

Para obtener el valor K1, se midió el voltaje en el pin 3 del potenciómetro RV6 de la Figura 2.7 con el multímetro y se comparó con el valor PROMEDIO_VOLTAJE. De acuerdo al valor obtenido, se varió el valor del potenciómetro RV6 hasta que los dos valores sean iguales. A continuación,se midió el valor de la fuente de voltaje DC y se obtuvo la relación:

$$\frac{\text{Voltaje DC de la fuente}}{\text{PROMEDIO\_VOLTAJE}} = K1 \tag{2.1}$$

Para obtener el valor de K2, se midió el voltaje del pin 1 del integrado U7:A de la Figura 2.9 con el multímetro y se comparó con el valor PROMEDIO_IDC. Posteriormente, se varió el valor del potenciómetro RV7 hasta que ambos valores fueran iguales. Luego, se midió la corriente (Corriente IDC) en la tarjeta de amplificación de potencia para obtener la relación.

$$\frac{\text{Corriente IDC}}{\text{PROMEDIO\_IDC}} = K2 \tag{2.2}$$

Para obtener el valor de K3, se midió el voltaje en el pin 1 del integrado U3:A de la Figura 2.8 con el multímetro y se lo comparo con el valor de voltaje PROMEDIO_TEMPERATURA. Luego, se varió el valor del potenciómetro RV5 para que ambos valores fueran iguales, y se midió el valor de la temperatura (Valor temperatura) en las cercanías de la tarjeta amplificadora de potencia con la termocupla del multímetro para obtener la relación.

$$\frac{\text{Valor temperatura}}{\text{PROMEDIO\_TEMPERATURA}} = K3 \tag{2.3}$$

Para obtener los valores finales de la potencia directa y la potencia reflejada que se muestran en la pantalla LCD, se realizó la interpolación de Lagrange con los datos obtenidos usando el vatímetro.

En la Figura 2.22 se muestra el código para realizar la interpolación de Lagrange, para lo cual se necesita obtener los valores *x* y *f*. El valor *x* representa el valor del voltaje obtenido en los pines 12 y 14 del microcontrolador, estas señales de voltaje provienen del sensor de potencia. Los datos son procesados y almacenados en las variables PROMEDIO_POT_DIRECTA y PROMEDIO_POT_REFLEJADA respectivamente. Mientras que el valor *f* representa el valor de potencia pico ($Potencia_{peak-sync}$), el cual está dado por la fórmula 2.4 [22].

$$f = Potencia_{peak-sync} = Potencia_{promedio} * 1.68 \tag{2.4}$$

```
x = [0, 0.38, 0.61, 0.89, 1.13];
f = [0, 50.4, 126, 252, 420];
n=length(x);
syms t;
p=0;
figure (5)
plot(x,f, "g")
for i=1:n
    L=1;
    for j=1:n
        if i~=j
            L=L*(t-x(j))/(x(i)-x(j));
        end
    end
    p=p+L*f(i)
end
p=expand(p)
hold on
ezplot(p, [0, 1.2])
```

Figura 2.22 Interpolación de Lagrange potencia directa

La Figura 2.23, presenta las curvas de potencia directa, donde la curva verde es la representación de los valores (*x,f*). Para determinar los valores de $Potencia_{peak-sync}$ (*f = [0, 50.4, 126, 252, 420])*, se midió la potencia directa colocando el vatímetro entre la salida del amplificador de potencia y una carga de 50 ohmios. El valor obtenido corresponde a la potencia promedio ($Potencia_{promedio} = [0,\ 30,\ 75,\ 150,\ 250]$).

La curva de color rojo representa la interpolación de Lagrange de los valores (*x,f*) cuyo polinomio se muestra en la ecuación 2.5.

$$P = 369.72t^4 - 788.97t^3 + 825.84t^4 - 87.54t \qquad (2.5)$$

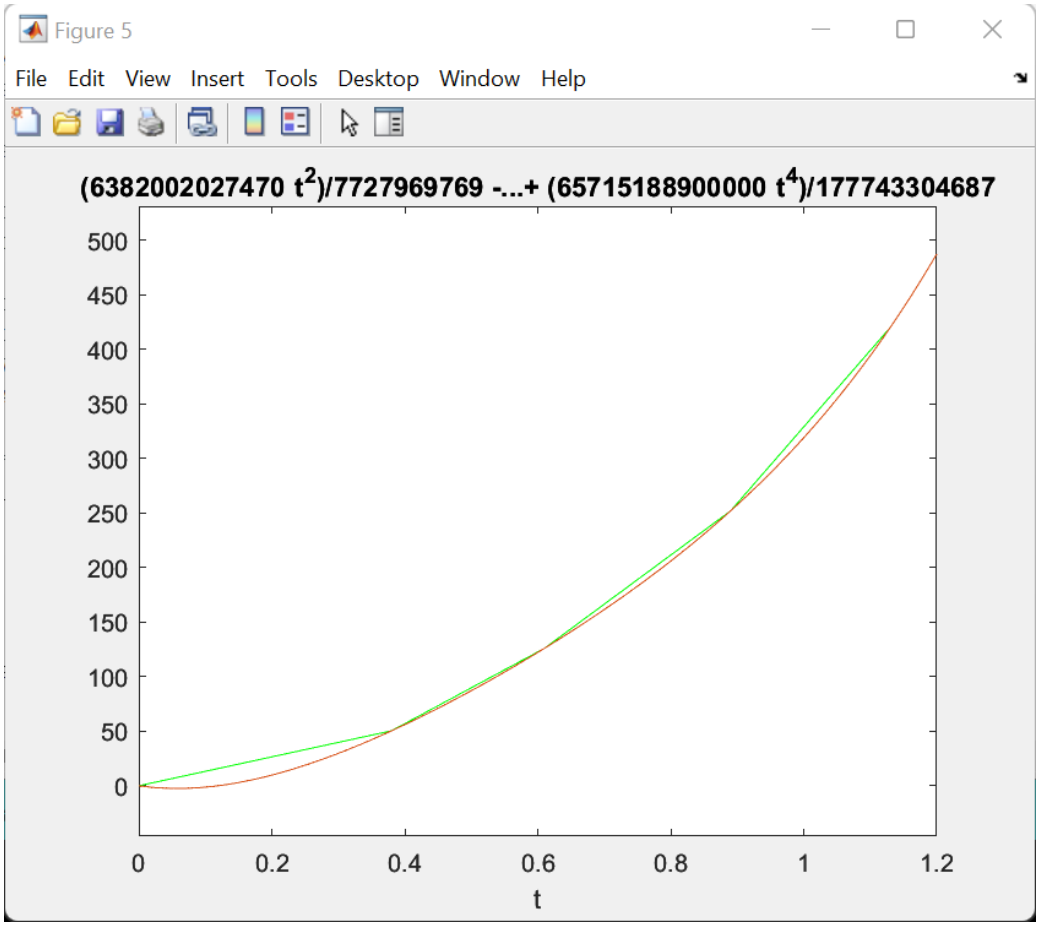


Figura 2.23 Curvas de potencia directa

En la Figura 2.24, se presenta las curvas de potencia reflejada, donde la curva verde representa los valores (*x,f*). Los valores del voltaje DC correspondientes a la potencia reflejada obtenidos del sensor de potencia y procesados por el microcontrolador son: *x* = [0, 0.18, 0.28, 0.46]. Para determinar los valores de $Potencia_{peak-sync}$ (*f = [0, 25.2, 42, 84])* se utilizó los valores de potencia *promedio* ($Potencia_{promedio} = [0,\ 15,\ 25,\ 50]$), obtenidos con el vatímetro, para lo cual en cada medición se desconecta la carga de 50 ohmios.

Al desconectar la carga se obtiene un voltaje DC en el sensor de potencia reflejada el cual ingresa al pin 3 del conector J1 como se muestra en el circuito 4 de la Figura 2.10 y desde este pin ingresa al circuito 5 de la Figura 2.11.

La curva de color rojo representa la interpolación de Lagrange de los valores (*x,f*) cuyo polinomio se muestra en la ecuación 2.6.

$$P = 289.86t^3 - 33.33t^2 - 136.61t \tag{2.6}$$

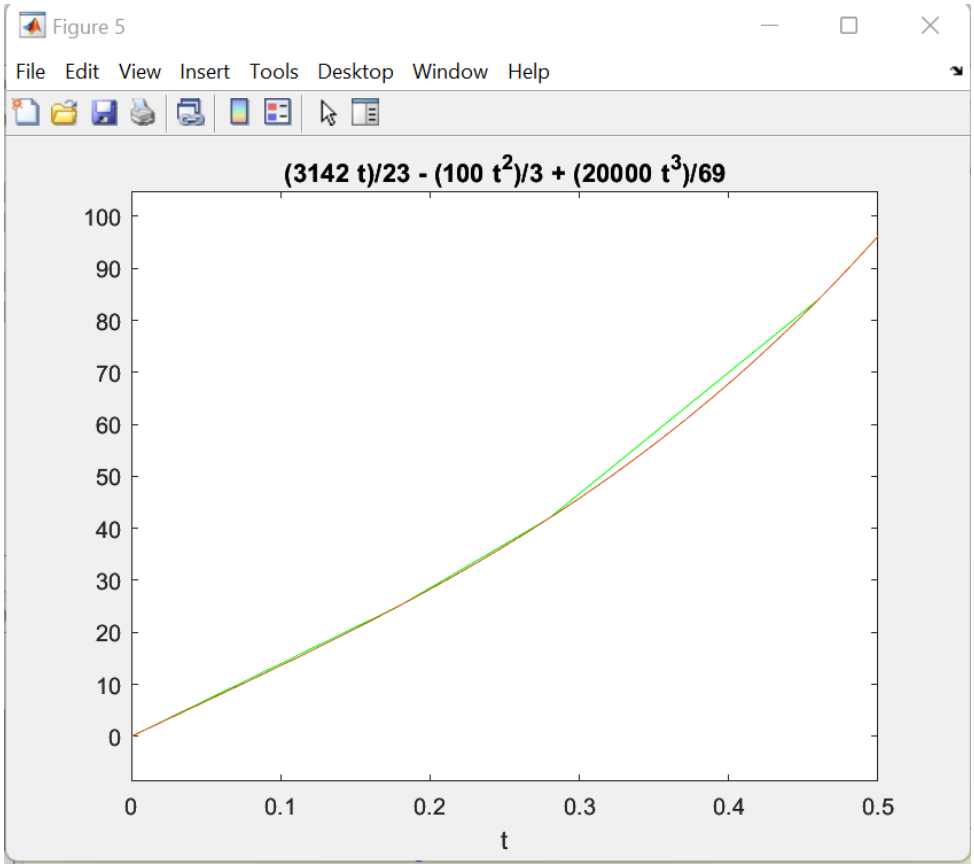


Figura 2.24 Curvas de potencia reflejada

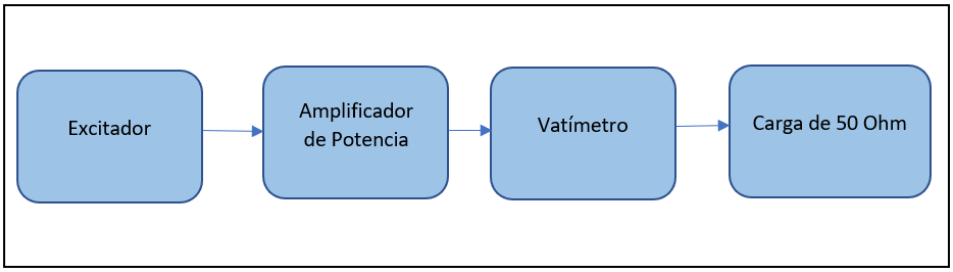


Figura 2.25 Diagrama de bloques para la medición de potencia

La Figura 2.26 muestra el código que se ejecuta cuando la potencia reflejada sobrepasa los 50 vatios, en este caso el NodeMCU coloca en alto la salida SWRAlarm, deshabilita la tarjeta de la amplificación de potencia colocando en alto la salida EnablePS y se habilita el envío de la alarma colocando un valor verdadero a la variable ENVIAR_ALARMAS.

La variable ACTIVAR_ALERTAS2 al detectar el valor de potencia reflejada elevada cambia su valor y se guarda en la memoria permanente del ESP32 con el propósito que solo se envíe una vez la alarma, de esta manera al resetearse el ESP32 o por un corte de energía esta variable no permitirá el envío consecutivo de la alerta.

Para que se pueda volver a enviar una alerta de potencia reflejada elevada el NodeMCU debe detectar que la potencia bajo de 50 vatios y por lo tanto a la siguiente vez que se detecte que la potencia sobrepaso este umbral se volverá a enviar la alarma.

```
if (POT_REFLEJADA_FINAL >= 50)
{

  if (ACTIVAR_ALERTA2 == 78)
  {
    ACTIVAR_ALERTA2 = 88;
    ENVIAR_ALARMAS = true;
    EEPROM.write(1, ACTIVAR_ALERTA2);
    EEPROM.commit();
    delay(6);
    ALERTA_POT_REFLEJADA = true;
    digitalWrite(SWRAlarm, HIGH);
    digitalWrite(EnablePS, HIGH);

    REINICIAR_VALOR_P = 98;
    EEPROM.write(3, REINICIAR_VALOR_P);
    EEPROM.commit();
    EEPROM.write(4, 67);
    EEPROM.commit();
    EEPROM.write(5, 67);
    EEPROM.commit();

  }
  delay (1000);
}
```

Figura 2.26 Código de alarma de potencia reflejada elevada

La Figura 2.27 muestra cómo, al detectarse que la potencia reflejada sobrepasó los 50 vatios y posterior se detecta que la potencia reflejada baja de ese umbral máximo, las variables y salidas digitales son restablecidas a sus estados iniciales. Esto permite que las alertas sean enviadas nuevamente cuando la potencia reflejada vuelva a sobrepasar los 50 vatios.

```
if (POT_REFLEJADA_FINAL <= 49)
{
  if (REINICIAR_VALOR_P == 98)
  {
    REINICIAR_VALOR_P = 0;
    EEPROM.write(3, REINICIAR_VALOR_P);
    EEPROM.commit();
    delay(6);

    ACTIVAR_ALERTA2 = 78;
    EEPROM.write(1, ACTIVAR_ALERTA2);
    EEPROM.commit();

    EEPROM.write(5, 77);
    EEPROM.commit();

    digitalWrite(SWRAlarm, LOW);
    digitalWrite(EnablePS, LOW);
    ALERTA_POT_REFLEJADA = false;

  }
}
```

Figura 2.27 Código para reestablecer variables de potencia reflejada

Las alertas de temperatura elevada son determinadas de manera similar a las alarmas de potencia reflejada, considerando que la temperatura máxima antes de generar la alarma es de 69 ° C.

Cuando se envía una alarma, también se transmiten los valores de voltaje DC, corriente DC, temperatura, potencia directa y potencia reflejada al módulo TCALL V1.4. Esto se logra mediante el uso de una estructura de datos denominada *sensordatos.*

La Figura 2.28 muestra los tipos de datos y como se conforma la estructura.

```
typedef struct struct_message {

  float a;
  float b;
  float c;
  float d;
  float e;
  boolean f;
  boolean g;

} struct_message;



 sensordatos.a = VOLTAJE_FINAL;
 sensordatos.b = IDC_FINAL;
 sensordatos.c = POT_DIRECTA_FINAL;
 sensordatos.d = POT_REFLEJADA_FINAL;
 sensordatos.e = PROMEDIO_TEMPERATURA;
 sensordatos.f = ALERTA_TEMPERATURA;
 sensordatos.g = ALERTA_POT_REFLEJADA;
```

Figura 2.28 Estructura de datos recolectados por el NodeMCU

2.4.3 DIAGRAMA DE FLUJO DEL PROGRAMA DEL TCALL V1.4

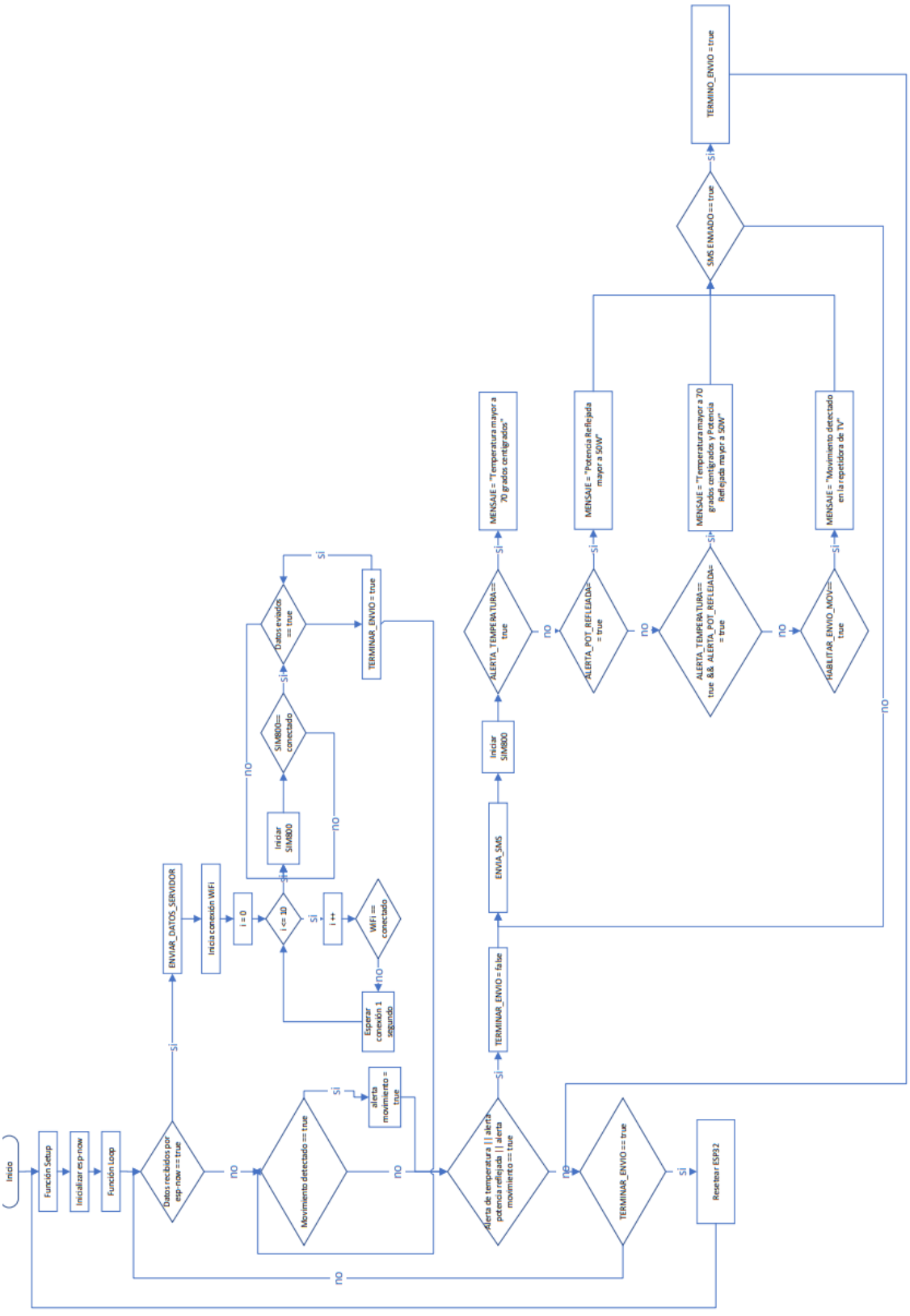


Figura 2.29 Diagrama de flujo del módulo TCALL V1.4

2.4.4 PROGRAMACIÓN DEL TCALL V1.4

El módulo TCALL V1.4 recibe los valores enviados por el módulo NodeMCU mediante el protocolo esp-now y los reenvía a la base de datos de la página web. Si se reciben las alarmas de temperatura, potencia reflejada o se genera la alarma de movimiento, el módulo envía mensajes de texto a través de las redes GSM hasta un número de teléfono celular.

A continuación, se explican las partes más relevantes del código implementado en el IDE de Arduino para el módulo TCALL V1.4. El código completo se encuentra en el ANEXO C**.**

La Figura 2.30 muestra las librerías empleadas para el funcionamiento del programa del módulo TCALL V1.4.

```
#include <esp_now.h>
#include <WiFi.h>
#include <HTTPClient.h>
#include <Wire.h>
#include <TinyGsmClient.h>
```

Figura 2.30 Librerías empleadas en el programa del módulo TCALL V1.4

- #include <HTTPClient.h>: librería utilizada para realizar solicitudes HTTP GET y POST a un servidor web.
- #include <TinyGsmClient.h>: librería empleada para el control del módulo SIM800 incorporado en el módulo TCALL V1.4.

La Figura 2.31 muestra la utilización de pines 5, 4, 23, 27 y 26 para el control del SIM800. El pin 25 se utiliza como entrada del sensor PIR de movimiento, mientras que el pin 13 se utiliza como salida digital para indicar cuando se ha detectado movimiento en la caseta de la estación repetidora de TV.

```
#define PIN_RST                 5
#define PIN_PWKEY               4
#define PIN_ON                  23
#define PIN_M_TX                27
#define PIN_M_RX                26

#define LED_MOV                 13
#define SENSOR_MOVIMIENTO       25
```

Figura 2.31 Pines utilizados en el módulo TCALL V1.4 para el SIM800

La Figura 2.32 muestra las constantes:

- apn[ ]: es el punto de acceso por el cual se conecta a internet empleando el SIM800. En este caso, se usa la tarjeta nano SIM de Tuenti, por lo que el valor asignado es *internet.tuenti.ec*.
- server[ ]: es la constante donde se almacena el nombre del dominio del sitio web, que en este caso es *radiofrecuenciateleamazonas.com*.
- resource[ ]: es el nombre del del archivo que realiza la conexión entre el módulo NodeMCU y la base de datos. En este proyecto, el nombre del archivo para la conexión con la base de datos es *datos-esp32.php.*
- port: constante que identificar que se está empleando el puerto 80 para la navegación HTTP.
- SMS_TARGET: indica el número de teléfono celular en formato internacional al que se enviaran los mensajes de alerta. En este proyecto, se emplea el prefijo para Ecuador +593.
- TINY_GSM_MODEM_SIM800: selecciona el modelo SIM800 para utilizarlo con la librería TinyGSMClient.h.
- TINY_GSM_BUFFER: configura el buffer de recepción a 1 Kb.

```
#define SMS_TARGET  "+593XXXXXXXXX"
#define TINY_GSM_MODEM_SIM800
#define TINY_GSM_RX_BUFFER   1024

const char apn[]      = "internet.tuenti.ec";
const char server[] = "radiofrecuenciateleamazonas.com";
const char resource[] = "/datos-esp32.php";
const int  port = 80;
```

Figura 2.32 Parámetros de conexión con el módulo SIM800

La Figura 2.33 muestra los parámetros para la conexión mediante una red WiFi.

- ssid: es el nombre de la red WiFi a la cual se conecta el módulo TCALL V1.4.
- password: es la contraseña para conectarse a la red WiFi.

- serverName: indica que se utiliza el recurso datos-esp32.php que está ubicado en el dominio radiofrecuenciateleamazonas.com.

```
const char* ssid     = "NOMBRE_DE_RED";
const char* password = "CONTRASEÑA";
const char* serverName = "http://radiofrecuenciateleamazonas.com/datos-esp32.php";
```

Figura 2.33 Conexión a la red WiFi

En la figura 2.34 se muestra la función OnDataRecv, la cual recibe los datos enviados a través del protocolo esp-now y los muestra en el monitor serial. Además dichos datos son asignados a variables para su posterior procesamiento.

```
void OnDataRecv(const uint8_t * mac, const uint8_t *incomingData, int len) {
  memcpy(&sensordatos, incomingData, sizeof(sensordatos));
  Serial.print("Bytes recibidos: ");
  Serial.println(len);
  //Imprime valores recibidos de la estructura
  Serial.print("a: ");
  Serial.println(sensordatos.a);
  Serial.print("b: ");
  Serial.println(sensordatos.b);
  Serial.print("c: ");
  Serial.println(sensordatos.c);
  Serial.print("d: ");
  Serial.println(sensordatos.d);
  Serial.print("e: ");
  Serial.println(sensordatos.e);
  Serial.print("f: ");
  Serial.println(sensordatos.f);
  Serial.print("g: ");
  Serial.println(sensordatos.g);

  VOLTAJE_FINAL = sensordatos.a;
  IDC_FINAL = sensordatos.b;
  POT_DIRECTA_FINAL = sensordatos.c;
  POT_REFLEJADA_FINAL = sensordatos.d;
  PROMEDIO_TEMPERATURA = sensordatos.e;
  ALERTA_TEMPERATURA = sensordatos.f;
  ALERTA_POT_REFLEJADA = sensordatos.g;
}
```

Figura 2.34 Asignación de datos recibidos por esp-now

La Figura 2.35 muestra la función set_up(), en la cual se realiza las siguientes preconfiguraciones.

- WiFi.mode(WIFI_STA): inicializa el módulo WiFi como estación.
- esp_now_init(): inicializa el protocolo esp-now.
- esp_now_register_recv_cb(OnDataRecv): registra la función de devolución de llamada para recibir datos a través del protocolo esp-now.

```
void setup() {

  Serial.begin(115200);
  delay(10);

  pinMode(SENSOR_MOVIMIENTO, INPUT_PULLUP);
  pinMode(LED_MOV, OUTPUT);
  digitalWrite(LED_MOV, LOW);
  delay(4000);
  //Inicializacion ESP NOW
  WiFi.mode(WIFI_STA);
  if (esp_now_init() != ESP_OK) {
    Serial.println("Error initializing ESP-NOW");
    return;
  }

  esp_now_register_recv_cb(OnDataRecv);

  Serial.println("termino INIT");
}
```

Figura 2.35 Función set_up

La Figura 2.36 muestra la función *loop(),* la cual monitorea la recepción de mensajes mediante el protocolo esp-now. Si la variable VOLTAJE_FINAL tiene un valor diferente de cero, se habilita el envío de los datos al servidor.

Los mensajes de alerta de movimiento se habilitan cuando la variable HABILITAR_ENVIO_MOV es verdadera, lo cual ocurre cuando el sensor PIR detecta movimiento tres veces. La alerta de temperatura elevada se envía cuando la variable booleana ALERTA_TEMPERATURA tiene un valor verdadero, de la misma manera que la

alerta de potencia elevada se habilita cuando el valor de la variable booleana ALERTA_POT_REFLEJADA es verdadero.

```
void loop() {

  if (VOLTAJE_FINAL != 0) {
    ENVIAR_DATOS_SERVIDOR();
  }
  if (DETECTAR_MOVIMIENTO == true)
  {
    ESTADO_MOVIMIENTO = digitalRead(SENSOR_MOVIMIENTO);
    if (ESTADO_MOVIMIENTO == true)
    {
      cont_mov++;

      if (cont_mov == 3)
      {
        HABILITAR_ENVIO_MOV = true;
      }
    }
  }

  if ((ALERTA_TEMPERATURA == true) || (ALERTA_POT_REFLEJADA == true) || (HABILITAR_ENVIO_MOV == true))
  {
    TERMINAR_ENVIO = false;
    //Configurar MENSAJES
    if (INIT_SMS == true)
    {
     FUNCION_INIT_SMS();
    }
    ENVIAR_SMS();
  }

  if (TERMINAR_ENVIO == true)
  {
    TERMINAR_ENVIO = false;
    ESP.restart();
  }
}
```

Figura 2.36 Función *loop* del TCALL V1.4

La función ENVIAR_DATOS_SERVIDOR se encargar de enviar los datos obtenidos a través del protocolo esp-now hacia la base de datos. Para ello, intenta realizar una conexión con la red WiFi, si la conexión no es exitosa, realiza 11 intentos de conexión. Si no puede realizar la conexión, enciende el módulo SIM800 para enviar los datos mediante la red GSM.

Las Figuras 2.37 y 2.38 muestran el envío de datos mediante la conexión WiFi, y a continuación se describe los principales comandos usados [5]:

- WiFi.begin(ssid, password): inicializa la configuración de la librería WiFi. El primer argumento, *ssid*, es el nombre de la red WiFi a la que se va a conectar, y el segundo argumento, *password*, es la contraseña de la red WiFi a la que se quiere conectar.
- WiFi.status(): retorna el estado de conexión.

- WiFi.localIP(): obtiene la dirección IP.
- http.begin(client, serverName): inicializa la conexión con el servidor.
- http.addHeader("Content-Type", "application/x-www-form-urlencoded"): define el tipo de contenido y el encabezado, donde *application/x-www-form-urlencoded* es el valor por defecto en el que los datos se codifican como una pareja clave-valor.
- Solicitud_datos: datos a ser enviados en la solicitud HTTP POST
- http.POST(Solicitud_datos): envia solicitud HTTP POST

```
void ENVIAR_DATOS_SERVIDOR() {
  WiFi.begin(ssid, password);
  Serial.println("CONECTANDO WIFI");
  delay(5);
  for (int i = 0; i <= 10; i++) {
    if (WiFi.status() != WL_CONNECTED) {
      delay(1000);
      Serial.print(".");
      ERROR_CONECT = true;
    }
    if (WiFi.status() == WL_CONNECTED) {
      i = 11;
      ERROR_CONECT = false;
    }

  }
  if (ERROR_CONECT == false)
  {
    Serial.println("");
    Serial.print("WIFI CON IP: ");
    Serial.println(WiFi.localIP());
    if (WiFi.status() == WL_CONNECTED) {
      WiFiClient client;
      HTTPClient http;
      http.begin(client, serverName);
      http.addHeader("Content-Type", "application/x-www-form-urlencoded");
      String Solicitud_datos = "ESTACION_1=" + UBICACION
                               + "&Voltaje_DC=" + String(VOLTAJE_FINAL)
                               + "&IDC=" + String(IDC_FINAL)
                               + "&Potencia_Directa=" + String(POT_DIRECTA_FINAL)
                               + "&Potencia_Reflejada=" + String(POT_REFLEJADA_FINAL)
                               + "&Temperatura=" + String(PROMEDIO_TEMPERATURA) + "";
```

Figura 2.37 Función para enviar datos por WiFi parte 1

```
    Serial.print("Solicitud_datos: ");
    Serial.println(Solicitud_datos);
    int respuesta_codigo = http.POST(Solicitud_datos);

    if (respuesta_codigo > 0) {
      Serial.print("HTTP Response code: ");
      Serial.println(respuesta_codigo);
    }
    else {
      Serial.print("Error code: ");
      Serial.println(respuesta_codigo);
    }
    http.end();

    Serial.println("TERMINO ENVIO");

    delay(100);
  }
  WiFi.disconnect();

  TERMINAR_ENVIO = true;

}
if (ERROR_CONECT == true)
{

  ERROR_CONECT = false;
  Serial.println("FALLO CONET WIFI");
  HABILITAR_SIM800 = true;
}
```

Figura 2.38 Función para enviar datos por WiFi parte 2

En las Figuras 2.39 y 2.40 se muestra el código para enviar datos al servidor usando el módulo SIM800, los principales comandos se detallan a continuación [23].

- SerialAT.begin(115200, SERIAL_8N1, MODEM_RX, MODEM_TX): configura la velocidad de señal del módulo GSM y configura los pines para la comunicación con el módulo SIM800.
- modem.gprsConnect(apn, gprsUser, gprsPass): realiza la conexión a internet empleando la *apn* internet.tuenti.ec.
- client.connect(server, port): conecta el cliente HTTP al servidor a través del puerto 80 usando el módulo SIM800.
- Solicitud_datos: datos a ser enviados con HTTP POST
- modem.gprsDisconnect(): desconecta el módulo SIM800.

```
//ENVIAR DATOS POR SIM800L INICIA LA CONEXION
if (HABILITAR_SIM800 == true)
{
  HABILITAR_SIM800 = false;
  TERMINAR_ENVIO = true;
  Serial.println("CONECT SIM800L");
  pinMode(PIN_PWKEY , OUTPUT);
  pinMode(PIN_RST, OUTPUT);
  pinMode(PIN_ON, OUTPUT);
  digitalWrite(PIN_PWKEY , LOW);
  digitalWrite(PIN_RST, HIGH);
  digitalWrite(PIN_ON, HIGH);
  SerialAT.begin(115200, SERIAL_8N1, PIN_M_RX, PIN_M_TX);
  delay(3000);
  modem.restart();
  delay(2000);
  if (strlen(simPIN) && modem.getSimStatus() != 3 ) {
    modem.simUnlock(simPIN);
  }
  Serial.print("CONECT APN: ");
  Serial.print(apn);
  if (!modem.gprsConnect(apn, gprsUser, gprsPass)) {
    Serial.println(" fail");
  }
  else {
    Serial.println(" OK");
    Serial.print("Connecting to ");
    Serial.print(server);
    if (!client.connect(server, port)) {
      Serial.println(" fail");
    }
    else {
      Serial.println(" OK");
```

Figura 2.39 Código para enviar datos por GSM parte 1

```
      Serial.println("Performing HTTP POST request...");

      String Solicitud_datos = "ESTACION_1=" + UBICACION + "&Voltaje_DC=" + String(VOLTAJE_FINAL)
                             + "&IDC=" + String(IDC_FINAL)
                             + "&Potencia_Directa=" + String(POT_DIRECTA_FINAL)
                             + "&Potencia_Reflejada=" + String(POT_REFLEJADA_FINAL)
                             + "&Temperatura=" + String(PROMEDIO_TEMPERATURA) + "";
      client.print(String("POST ") + resource + " HTTP/1.1\r\n");
      client.print(String("Host: ") + server + "\r\n");
      client.println("Connection: close");
      client.println("Content-Type: application/x-www-form-urlencoded");
      client.print("Content-Length: ");
      client.println(Solicitud_datos.length());
      client.println();
      client.println(Solicitud_datos);
      unsigned long timeout = millis();
      while (client.connected() && millis() - timeout < 10000L) {
        while (client.available()) {
          char c = client.read();
          Serial.print(c);
          timeout = millis();
        }
      }
      Serial.println();

      client.stop();
      Serial.println(F("Server disconnected"));
      modem.gprsDisconnect();
      Serial.println(F("GPRS disconnected"));
    }
  }

}
```

Figura 2.40 Código para enviar datos por GSM parte 2

La función en la Figura 2.41 es la encargada de enviar mensajes de texto en caso de que se generen alertas. Si la alerta se genera debido a temperatura elevada, se envía el mensaje de texto "**Temperatura mayor a 70 grados centígrados**".

Si la alerta se genera debido a una potencia reflejada elevada, se envía el mensaje de texto "**Potencia Reflejada mayor a 50W**".

Si ambas alertas se generan al mismo tiempo, se envía el mensaje de texto "**Temperatura mayor a 70 grados centígrados y Potencia Reflejada mayor a 50W**".

Si la alerta se genera debido a movimiento, se envía el mensaje de texto "**Movimiento detectado en la repetidora de TV**"

Cuando se envían los mensajes de alerta, las variables se establecen en sus valores iniciales para permitir el envío de una nueva alerta en caso de que se genere otra. Esto asegura que las alertas no se envíen continuamente.

```
void ENVIAR_SMS() {
  if ((ALERTA_TEMPERATURA == true) && (ALERTA_POT_REFLEJADA == true)){
    DOBLE_ALARMA = true;
    smsMessage = "Temperatura mayor a 70 grados centigrados  y Potencia Reflejada mayor a 50W";
  }
  if (DOBLE_ALARMA == false)
  {
    if (ALERTA_TEMPERATURA == true ){
      smsMessage = "Temperatura mayor a 70 grados centigrados";
    }
    if (ALERTA_POT_REFLEJADA == true){
      smsMessage = "Potencia Reflejada mayor a 50W";
    }
  }
  if (HABILITAR_ENVIO_MOV == true){
    smsMessage = "Movimiento detectado en la repetidora de TV";
    DETECTAR_MOVIMIENTO = false;
    HABILITAR_ENVIO_MOV = true; //mantiene en alta hasta que se envie el mensaje de movimiento
    digitalWrite(LED_MOV, HIGH);
  }
  if (modem.sendSMS(SMS_TARGET, smsMessage)) {
    Serial.println(smsMessage);
    INIT_SMS = true;
    TERMINAR_ENVIO = true;
    ALERTA_TEMPERATURA = false;
    ALERTA_POT_REFLEJADA = false;
    DOBLE_ALARMA = false;
    DETECTAR_MOVIMIENTO = true;
    HABILITAR_ENVIO_MOV = false;
    cont_mov = 0;
    digitalWrite(LED_MOV, LOW);
  }
  else {
    Serial.println("Error enviando SMS");
  }
```

Figura 2.41 Función para enviar mensajes de texto

2.5 DISEÑO DE LA PÁGINA WEB

Para el diseño del sitio web se contrató un hosting con la empresa Hostinger que ofrece diferentes planes y funciones para alojar sitios web, incluyendo el acceso a la base de datos MySQL, espacio de almacenamiento, un dominio gratuito por un año, compatibilidad

con WordPress y servicio al cliente en español las 24 horas del día. Además, sus planes de hosting son económicos en comparación con otras empresas del mercado.

Con los beneficios y facilidades que ofrece un hosting web compartico en Hostinger, se consideró que es una mejor opción para las necesidades del presente proyecto en lugar de implementar un servidor dedicado.

2.5.1 IMPLEMENTACIÓN DE LA BASE DE DATOS

Para la creación de la base de datos se utiliza PHPMYADMIN ubicado en el servidor web compartido, a continuación se indican los pasos para crear una tabla:

1. Ingresar a la base de datos de MySQL del servidor web compartido.

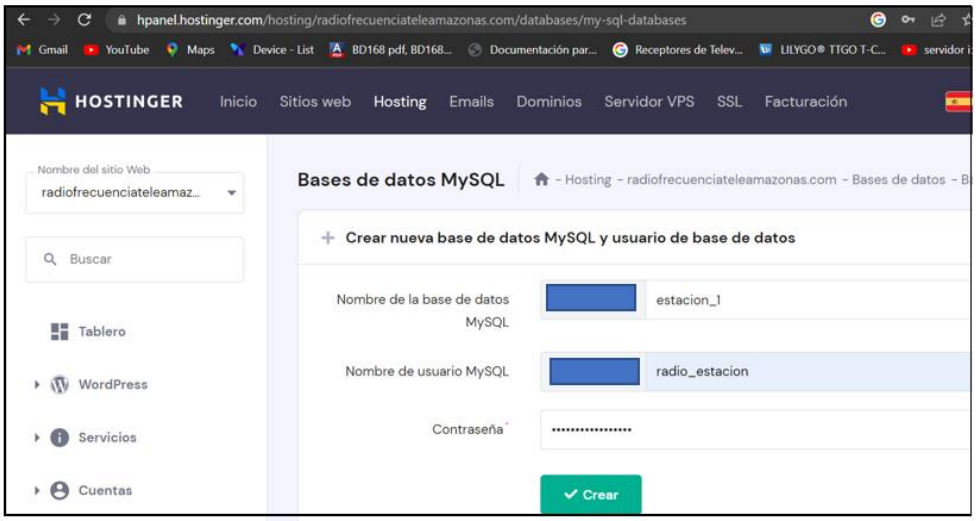


Figura 2.42 Creación de la base de datos

2. creación de la tabla llamada “Tabla Valores” que tiene los siguientes campos:

    - **Lectura:** Campo que se autoincrementa para identificar cada registro en la tabla.
    - **ESTACION_1:** Campo para almacenar el nombre de la estación repetidora donde se recolectan los datos.
    - **Voltaje_DC:** Campo para almacenar el valor del voltaje DC medido.
    - **IDC:** Campo para almacenar el valor de corriente DC medida.
    - **Potencia_Directa:** Campo para almacenar el valor de potencia directa medida.

- **Potenica_Reflejada:** Campo para almacenar el valor de potencia reflejada medida.
- **Temperatura:** Campo para almacenar el valor de temperatura medida.
- **reading_time:** Campo para guardar la fecha y hora en la que se almacenaron los datos.

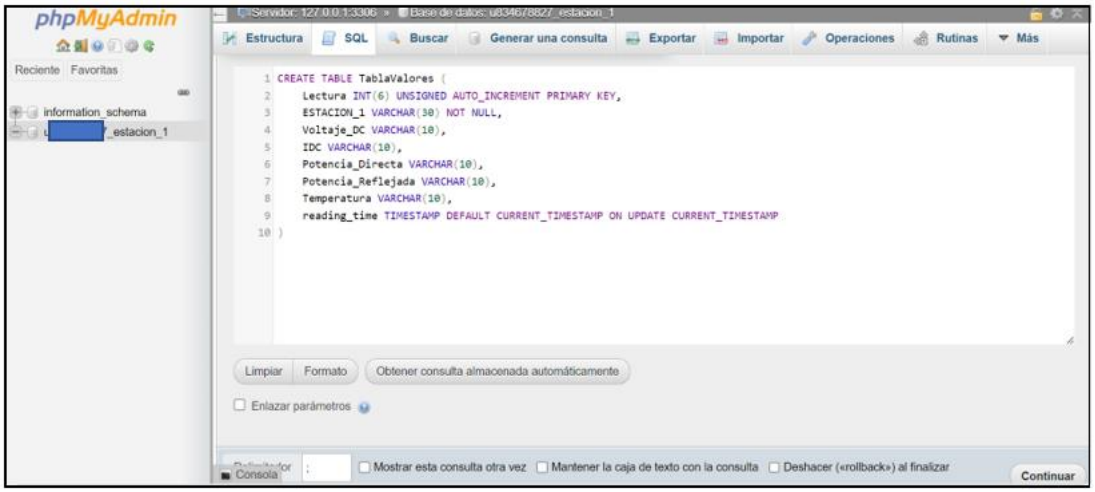


Figura 2.43 Creación de tabla “TablaValores”

2.5.2 INGRESAR VALORES EN LA BASE DE DATOS MYSQL

Para el ingreso de información en la base de datos se utiliza un archivo implementado con PHP, el cual está ubicado en la carpeta *public_html* del servidor web contratado como se muestra en la Figura 2.44.

```php
<?php
$servername = "localhost";
$dbname = [redacted]_estacion_1";
$username = [redacted]_radio_estacion";
$password = [redacted]";

$api_key= $ESTACION_1 = $Voltaje_DC = $IDC = $Potencia_Directa = $Potencia_Reflejada = $Temperatura = "";

if ($_SERVER["REQUEST_METHOD"] == "POST") {
    $api_key = test_input($_POST["api_key"]);
        $ESTACION_1 = test_input($_POST["ESTACION_1"]);
        $Voltaje_DC = test_input($_POST["Voltaje_DC"]);
        $IDC = test_input($_POST["IDC"]);
        $Potencia_Directa = test_input($_POST["Potencia_Directa"]);
        $Potencia_Reflejada = test_input($_POST["Potencia_Reflejada"]);
        $Temperatura = test_input($_POST["Temperatura"]);

        // Crea la conexion
        $conn = new mysqli($servername, $username, $password, $dbname);
        // Verifica la conexion
        if ($conn->connect_error) {
            die("Connection failed: " . $conn->connect_error);
        }

        $sql = "INSERT INTO TablaValores (ESTACION_1, Voltaje_DC, IDC, Potencia_Directa, Potencia_Reflejada, Temperatura)
        VALUES ('" . $ESTACION_1 . "', '" . $Voltaje_DC . "', '" . $IDC . "', '" . $Potencia_Directa . "', '" . $Potencia_Reflejada . "', '" . $Temperatura . "')";
```

Figura 2.44 Archivo para ingreso de datos en la base de datos MYSQL

El campo *dbname* corresponde al nombre de la base de datos, el *username* corresponde al nombre de usuario de la base de datos y el *password* corresponde a la contraseña de la base de datos, estos datos se muestran en la figura 2.42.

También se muestra en la Figura 2.44 que los argumentos del código *$sql*, corresponden al nombre de la tabla y campos mostrados en la figura 2.43.

2.5.3 PANTALLA DE INICIO DE LA PÁGINA WEB

Para la pantalla de inicio se utilizó el sistema de gestión de contenidos WordPress, el proceso se muestra a continuación.

1. Instalar WordPress en el hosting web.

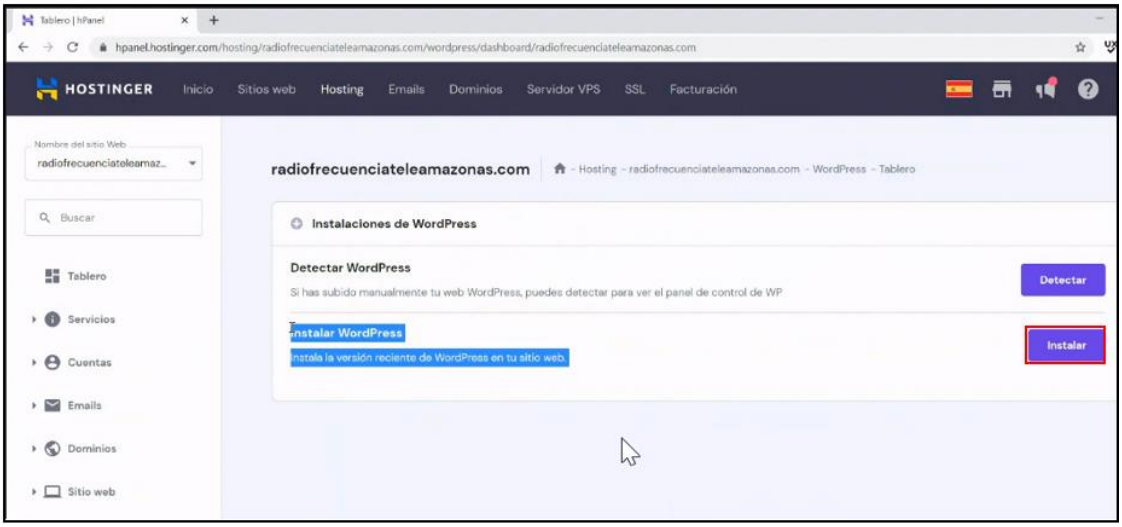


Figura 2.45 Instalar WordPress

2. Entrar en WordPress y presionar en el botón *Editar sitio web.*

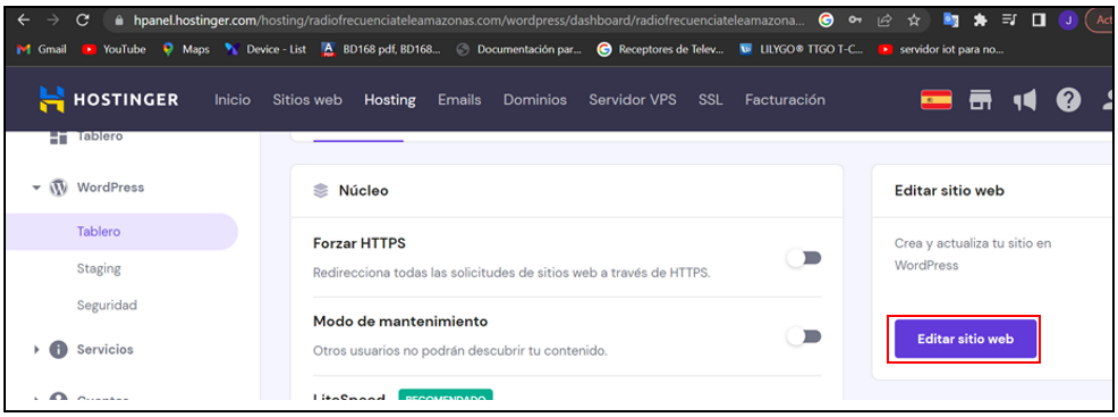


Figura 2.46 Tablero de WordPress

3. En la Figura 2.47 se muestra el entorno de edición en el cual se puede modificar el aspecto de la página.

Figura 2.47 Entorno de edición de WordPress

4. La Figura 2.48 muestra el aspecto final de la página web de inicio, la cual contiene un botón que redirige a la página web llamada *Datos obtenidos Estación Repetidora TV,* la cual tiene una conexión con la base de datos MySQL y tiene un control de usuarios.

Figura 2.48 Pantalla de inicio de página web

2.5.4 BASE DE DATOS DE MYSQL EN WORDPRESS

Para presentar los datos recolectados por el módulo NodeMCU, se ha creado una página web en el hosting compartido utilizando WordPress. En esta página se muestra la base de datos de MySQL, donde se almacena los valores enviados por el módulo TCALL V1.4.

El procedimiento detallado de la creación de la página web llamada *Datos obtenidos Estación Repetidora TV*, la conexión de información desde la base de datos MySQL con la página de WordPress y el control de usuarios de la misma, se muestra en detalle en el ANEXO D.

En la Figura 2.49 se muestra la conexión exitosa entre la base de datos de MySQL y la página *Datos obtenidos Estación Repetidora TV.*

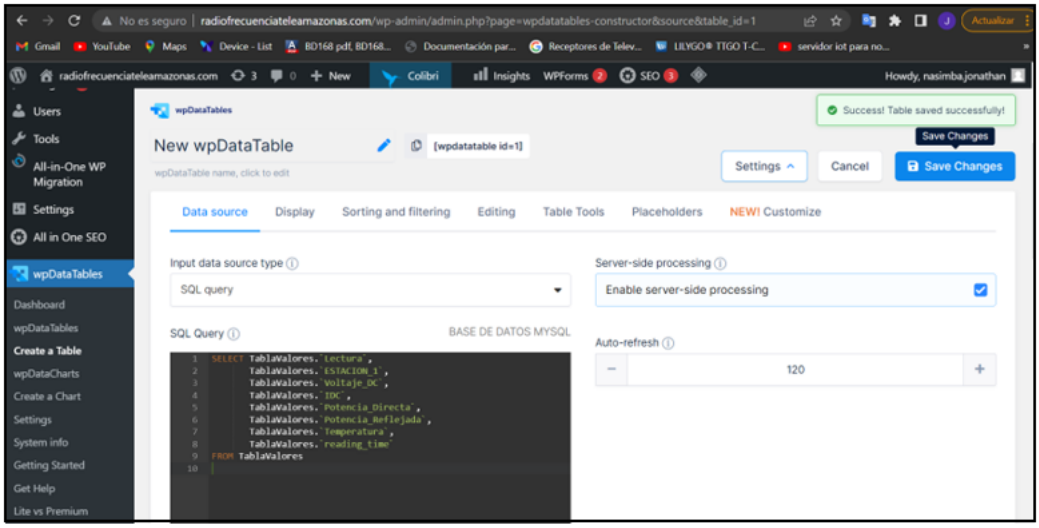


Figura 2.49 Conexión de MySQL con página de WordPress

En la figura 2.50 se muestra cómo se realiza el control de usuarios a la página *Datos obtenidos Estación Repetidora TV* para que solo los usuarios registrados y el administrador puedan acceder a su contenido.

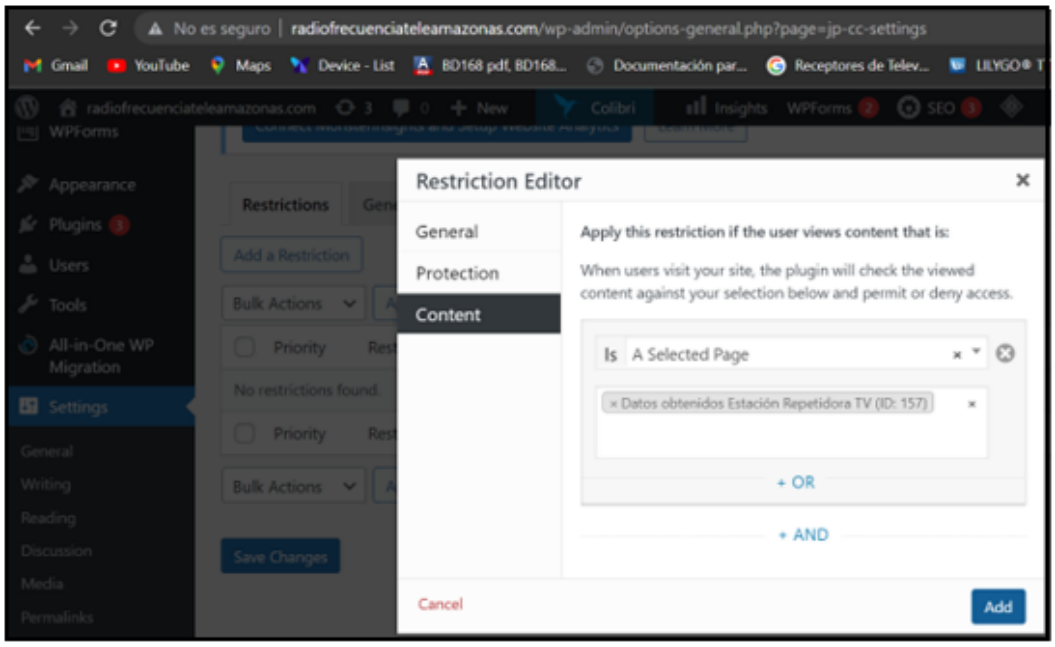


Figura 2.50 Restricción de usuarios a página de WordPress

La Figura 2.51 muestra cómo registrar un usuario y asignarle un rol, como subscriptor, editor de contenido, colaborador, autor, editor o administrador.

En este ejemplo, se le asigna al usuario el rol de subscriptor, lo que le permitirá acceder a la página web *Datos obtenidos Estación Repetidora TV.*

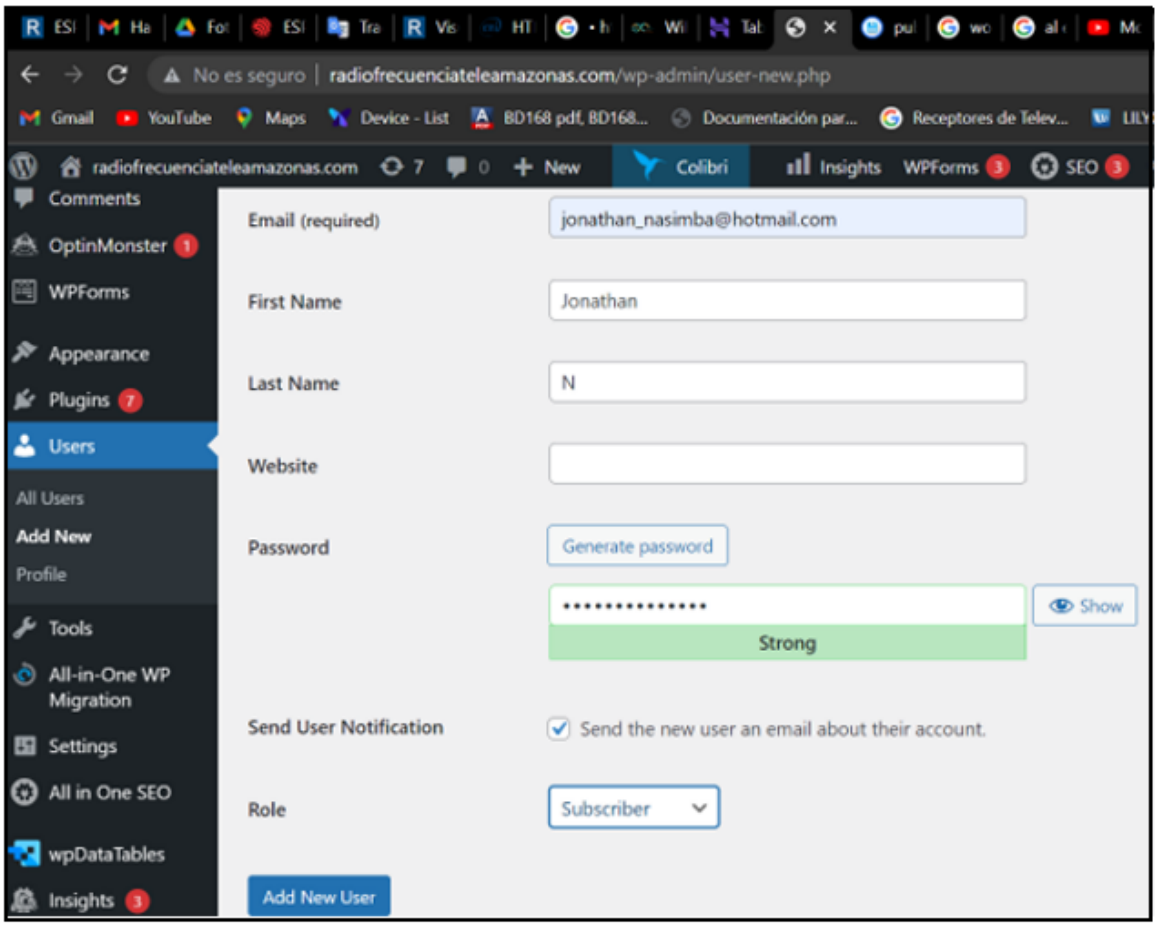


Figura 2.51 Registro de usuario

3. RESULTADOS Y DISCUSIÓN

3.1 IMPLEMENTACIÓN DE LAS PCB1 Y PCB2

Para realizar las pruebas de funcionamiento se mandaron a fabricar las placas de circuito impreso PCB1 y PCB2 en la empresa JLCPCB. Los materiales se adquirieron de forma local y la soldadura de los componentes electrónicos se llevó a cabo por el autor del proyecto.

La figura 3.1 muestra la PCB1 antes de soldar los componentes electrónicos.

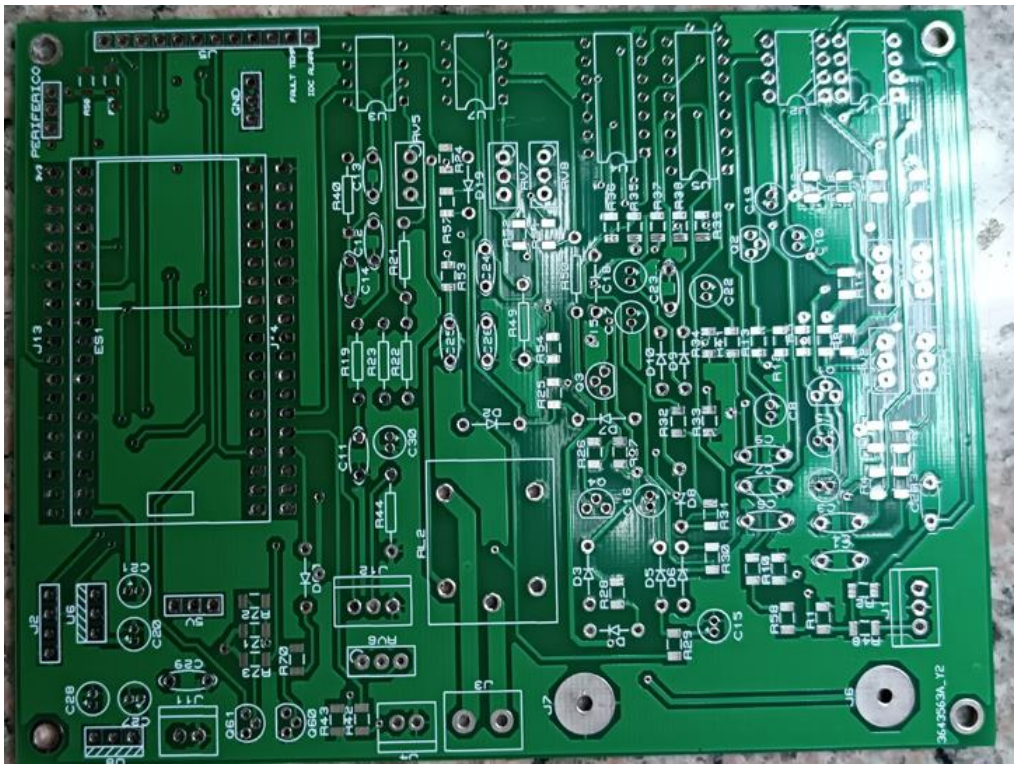


Figura 3.1 Placa de circuito impreso PCB1

En la Figura 3.2 se muestra la placa de circuito impreso PCB1, a la cual se le ha modificado las conexiones de los pines STOPTEMP (pin 34), SWRALARM (pin 35) y TEMPERATURA (pin 0), remplazandolos por los pines 4, 13 y 26, respectivamente. Esto se debe a que los pines 34 y 35 solo funcionan como entradas, mientras que el pin 0 se utiliza para el arranque del esp32-wroom-32 que se encuentra en el módulo NodeMCU.

Los diagramas esquemáticos completos de las tarjetas de circuito impreso PCB1 y PCB2 fabricadas se muestran en el ANEXO E.

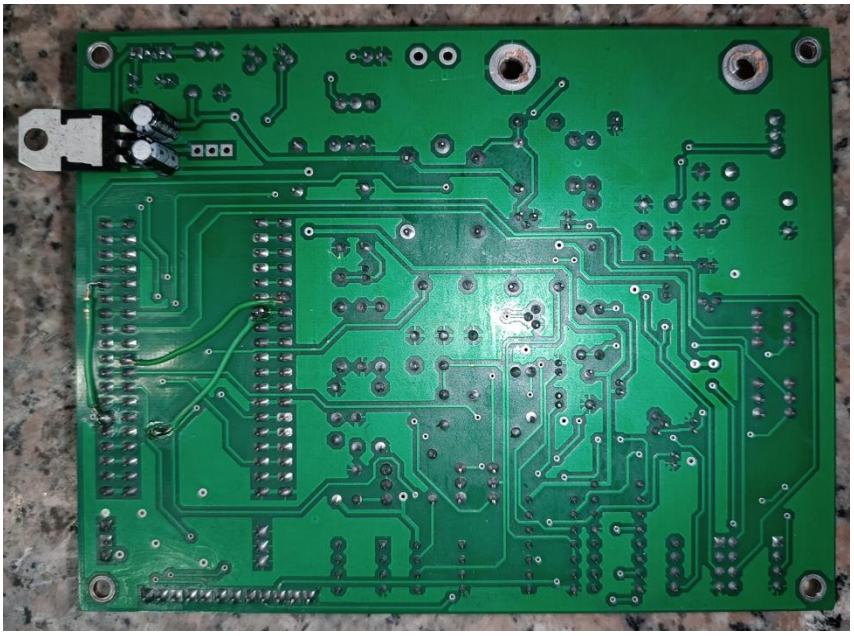


Figura 3.2 Corrección de pines para el módulo NodeMCU

La Figura 3.3 muestra la placa de circuito impreso de la PCB2 que implementa circuitos para la recolección de señales analógicas y un sistema de suministró de energía con un respaldo de batería, sin embargo, en este proyecto solo se utilizó el circuito de suministro de energía, el cual se muestra en la Figura 2.13.

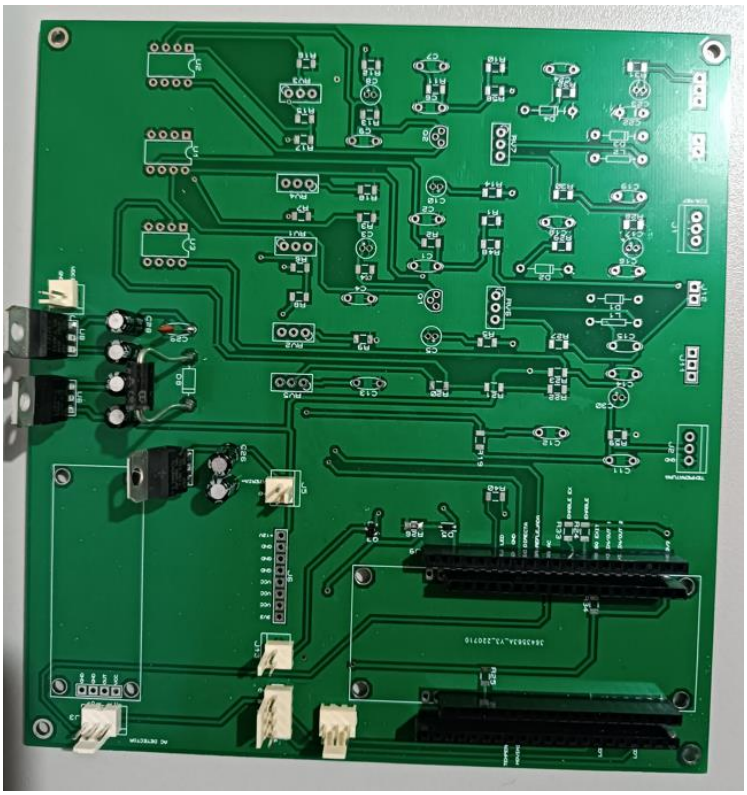


Figura 3.3 Placa de circuito impreso PCB2

3.2 PRUEBAS DE RECOLECCIÓN DE SEÑALES ANALÓGICAS

La Figura 3.4 muestra las conexiones de los sensores a la placa de circuito impreso del módulo NodeMCU ESP32.

1. Módulo NodeMCU.
2. Entrada de la fuente de alimentación de 15 voltios a la PCB1 a través del pin 1 del conector J11, como se muestra en el circuito 8 de la Figura 2.5.
3. Entrada de voltaje DC del amplificador de potencia a través del pin 2 del conector J4, como se muestra en el circuito 1 de la Figura 2.7.
4. Entrada del sensor de temperatura LM35 a través del pin 2 del conector J12, como se muestra en el circuito 3 de la Figura 2.8.
5. Resistencia por la cual atraviesa la corriente de la tarjeta de amplificación de potencia conectada a los conectores J6 y J7 del circuito 2, como se muestra en la Figura 2.9.
6. Punto de conexión de la salida del sensor de potencia con el conector J1. En el pin 1 se conecta la potencia directa y al pin 2 se conecta la potencia reflejada, como se muestra en el circuito 4 de la Figura 2.10. Se debe mencionar que la entrada en el sensor de potencia es de radiofrecuencia y a su salida se tienen señales de voltaje DC.
7. Relé para desactivar el excitador, como se muestra en el circuito 7 de la figura 2.12.
8. Pantalla LCD.

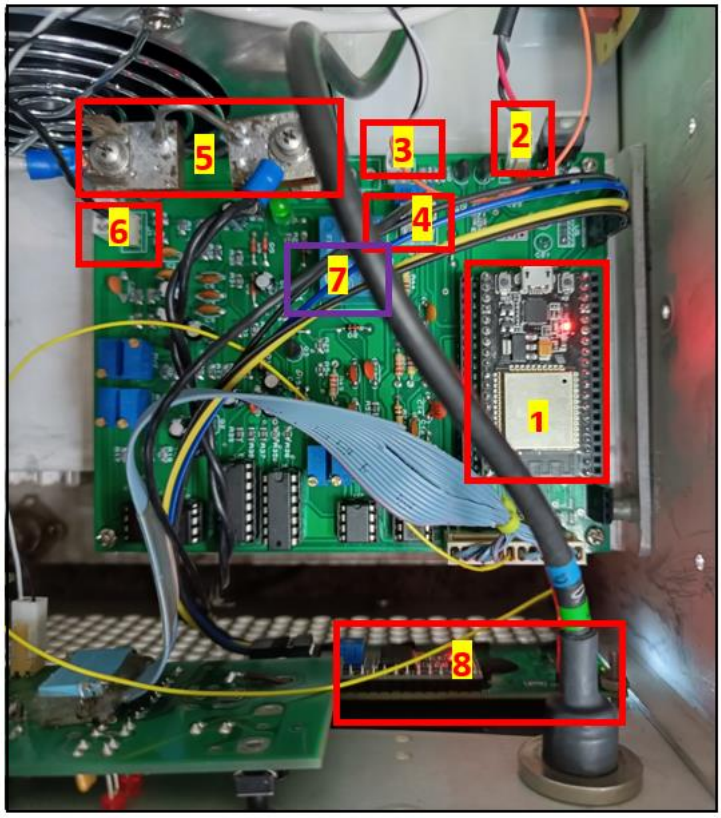


Figura 3.4 Conexión de sensores al circuito del módulo NodeMCU

En la Figura 3.5 se muestra el circuito para el envío de datos y alarmas, el cual está conformado por el módulo TCALL V1.4 y el sensor de movimiento HC-SR501.

El sistema se energiza mediante una fuente de 15 voltios conectada en el terminal *Vin,* y además cuenta con una batería de respaldo para cuando la fuente de voltaje no está funcionando. En caso de que la fuente de alimentación falle, el circuito se alimentará automáticamente con la batería.

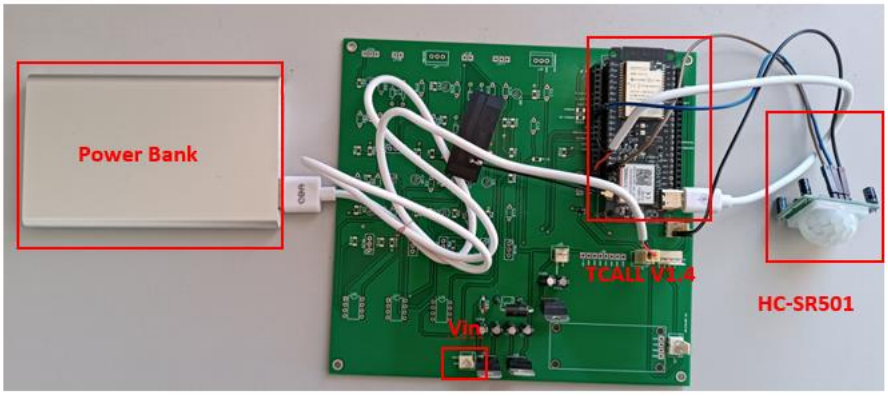


Figura 3.5 Circuito para enviar mensajes vía WiFi o GSM

3.2.2 MEDICIÓN DE VOLTAJE DC

En la Figura 3.6 se muestra la medición del voltaje de la fuente de 50 voltios que emplea la tarjeta de amplificación de potencia.

Figura 3.6 Medición de voltaje DC

3.2.3 MEDICIÓN DE CORRIENTE DC

En la Figura 3.7 se muestra la medición de la corriente DC de la tarjeta amplificadora de potencia cuando no está conectada la señal de radio frecuencia que proviene del excitador.

En la figura se muestras el valor de la corriente DC en miliamperios medida con el multímetro, mientras que el valor de la corriente medida por el NodeMCU se muestra en la pantalla LCD en amperios.

Figura 3.7 Medición de corriente DC

3.2.4 MEDICIÓN DE TEMPERATURA

En La Figura 3.8 se muestra el sensor LM35 empleado para medir la temperatura.

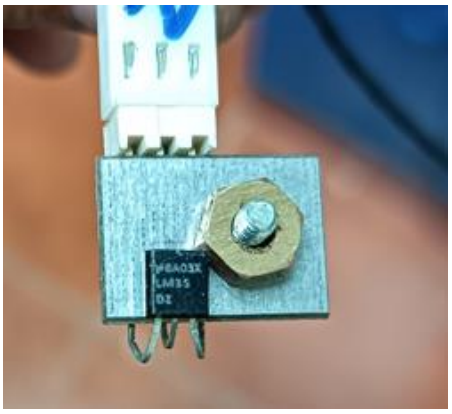


Figura 3.8 Sensor de temperatura LM35

En la Figura 3.9 se muestra la ubicación del sensor de temperatura en relación a la tarjeta de amplificación de potencia. Como se puede apreciar en la figura, el cable del sensor pasa muy cerca de la señal de radio frecuencia, lo que provoca que el circuito del módulo NodeMCU realice mediciones erróneas en el pin de entrada de temperatura, generando alertas de temperatura elevada de forma incorrecta. Para solucionar este problema, se separó el cable del sensor de temperatura de la parte de radio frecuencia.

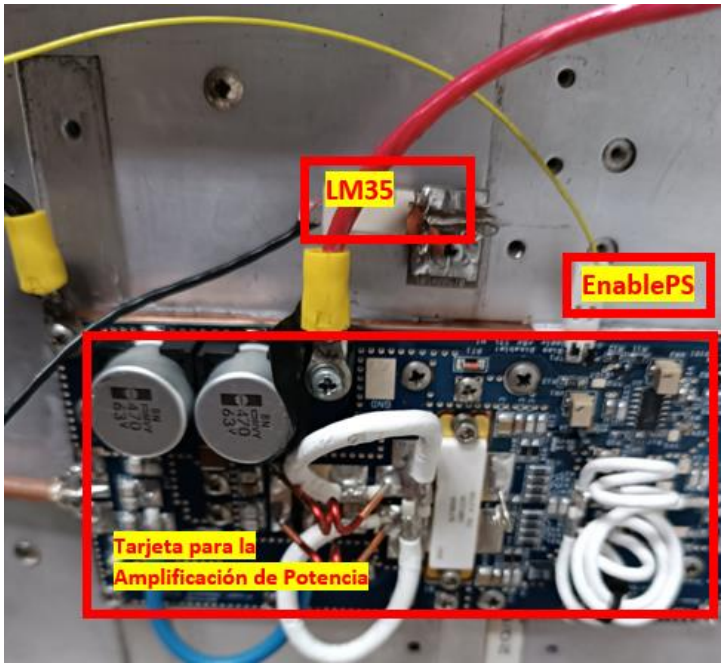


Figura 3.9 Ubicación de sensor de temperatura

La Figura 3.10 muestra la medición de temperatura utilizando una termocupla conectada al multímetro, y la medición de temperatura utilizando el módulo NodeMCU con el sensor LM35.

Figura 3.10 Medición de temperatura

3.2.5 MEDICIÓN DE POTENCIA

La Figura 3.11 muestra el esquema completo del transmisor de televisión analógica, en el cual se conecta el vatímetro entre la salida del amplificador de potencia y la carga de 50 ohmios.

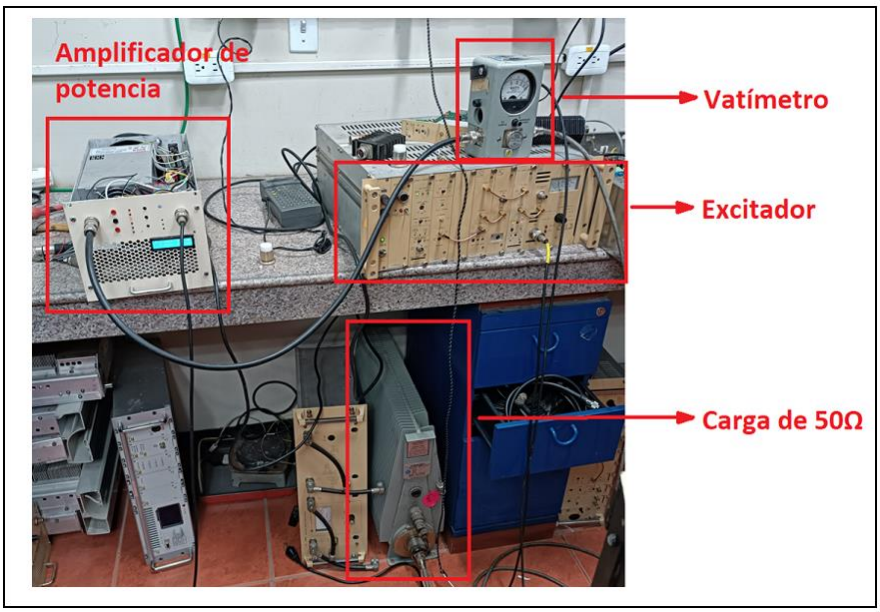


Figura 3.11 Sistema transmisor de televisión analógica

En la Figura 3.12 se muestra el ingreso de la señal de radio frecuencia desde el excitador (RF INPUT) y la salida de la señal de radio frecuencia (RF OUTPUT) en la cual se encuentra el sensor de potencia directa y reflejada.

Figura 3.12 Conexión de la señal de radio frecuencia al amplificador de potencia

En la Tabla 3.1 y 3.2 se muestran los valores medidos y calculados de la potencia directa y reflejada respectivamente. Con estos datos, se calcularon el error absoluto y error relativo.

Tabla 3.1 Mediciones de potencia directa

Potencia Directa				
Valor promedio medido (W)	Valor pico medido (W)	Valor pico calculado ESP32 (W)	Erro absoluto	Error relativo (%)
250	420	444	24	5,71
200	336	364	28	8,33
150	252	275	23	9,13
100	168	185	17	10,12
75	126	140	14	11,11
50	84	94	10	11,90
25	42	47	5	11,90

Tabla 3.2 Mediciones de Potencia reflejada

Potencia Reflejada				
Valor promedio medido (W)	Valor pico medido (W)	Valor pico calculado ESP32 (W)	Erro absoluto	Error relativo (%)
5	8,4	4	4,4	52,38
15	25,2	25	0,2	0,79
25	42	41	1	2,38
50	84	84	0	0,00

3.2.6 PRUEBAS DE ENVÍO DE DATOS AL SERVIDOR WEB

En la figuras 3.13 se muestra la pantalla de inicio y al presionar el botón “INGRESAR” se redirige a la página *datos obtenidos estación repetidora tv.* Esta página tiene un control de acceso de usuarios, por lo que se solicita ingresar el correo y la contraseña como se muestra en la Figura 3.14.

La Figura 3.15 muestra la página *datos obtenidos estación repetidora tv,* la cual está conectada con la base de datos de MySQL donde se almacenan los valores de los datos enviados por el módulo TCALL V1.4.

Figura 3.13 Pantalla de inicio de página web

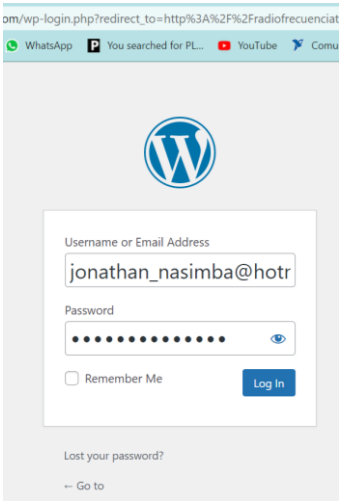
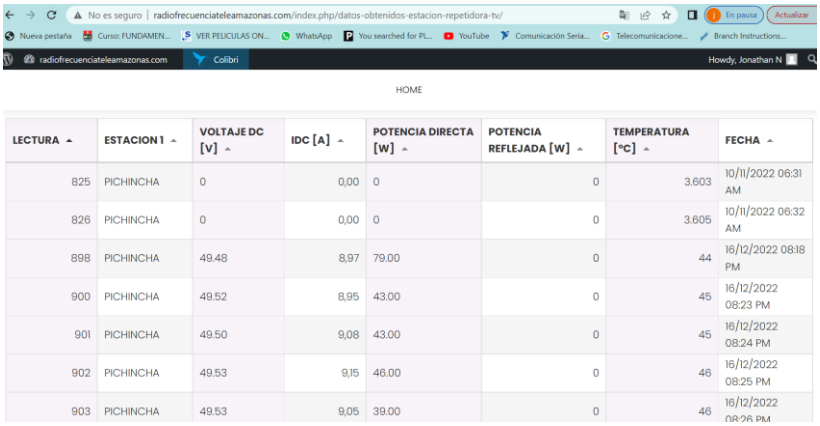


Figura 3.14 Acceso con control de usuarios



LECTURA	ESTACION 1	VOLTAJE DC [V]	IDC [A]	POTENCIA DIRECTA [W]	POTENCIA REFLEJADA [W]	TEMPERATURA [°C]	FECHA
825	PICHINCHA	0	0,00	0	0	3.603	10/11/2022 06:31 AM
826	PICHINCHA	0	0,00	0	0	3.605	10/11/2022 06:32 AM
898	PICHINCHA	49.48	8,97	79.00	0	44	16/12/2022 08:18 PM
900	PICHINCHA	49.52	8,95	43.00	0	45	16/12/2022 08:23 PM
901	PICHINCHA	49.50	9,08	43.00	0	45	16/12/2022 08:24 PM
902	PICHINCHA	49.53	9,15	46.00	0	46	16/12/2022 08:25 PM
903	PICHINCHA	49.53	9,05	39.00	0	46	16/12/2022 08:26 PM

Figura 3.15 Datos mostrados en la página web

3.2.7 PRUEBAS DE ENVÍO DE ALERTAS

La Figura 3.16 muestra cómo se envía el menaje de alerta de temperatura elevada. Al detectarse que la temperatura sobrepaso el límite establecido, el módulo NodeMCU envía la alerta junto con los datos de voltaje DC, corriente DC, potencia directa, potencia reflejada y temperatura medidos cuando se generó la alerta.

Si la conexión por WiFi fue exitosa, envía los datos al servidor y luego envía el mensaje de texto empleando el SIM800.

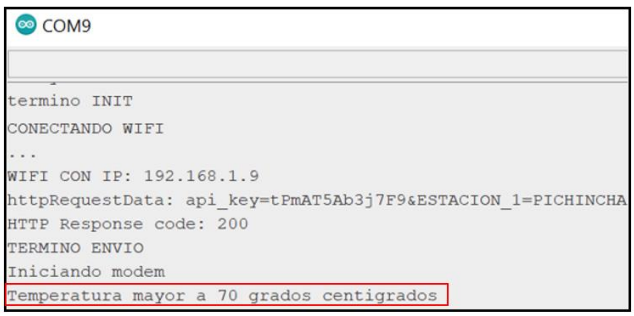


Figura 3.16 Mensaje de alerta de temperatura elevada

La Figura 3.17 muestra el envío del mensaje de alerta de potencia reflejada. En la figura se observa que no se pudo establecer la conexión con la red WiFi, por lo cual se activa el módulo SIM800 para enviar la información recolectada a la base de datos del servidor web.

Después de enviar los datos al servidor, se procede a enviar el mensaje de texto con el mensaje, *Potencia Reflejada mayor a 50w*. Si el mensaje no fue enviado exitosamente, se seguirá intentando hasta que se logre enviar correctamente.

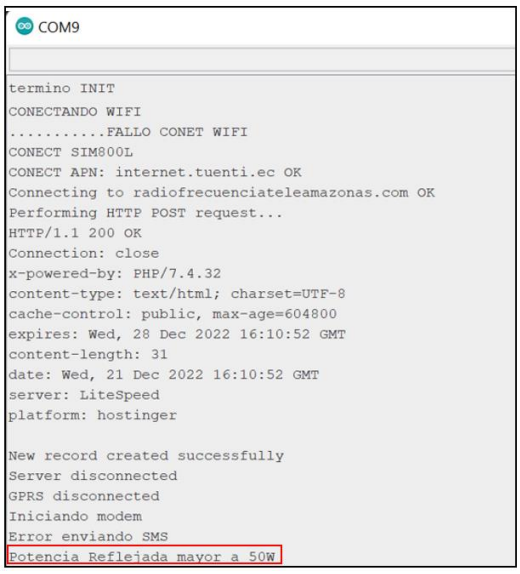


Figura 3.17 Mensaje de alerta de potencia reflejada elevada

La Figura 3.18 muestra cómo se envía el mensaje de texto de alerta de movimiento. En este caso, no se envían datos obtenidos del módulo NodeMCU.

Si el envío de mensaje no fue exitoso, se seguirá intentando hasta que se pueda enviar el mensaje.

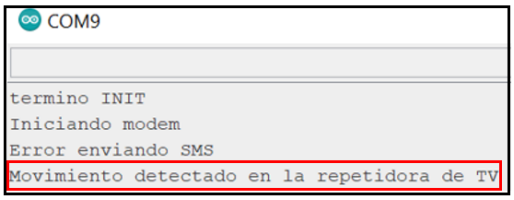


Figura 3.18 Mensaje de alerta de movimiento

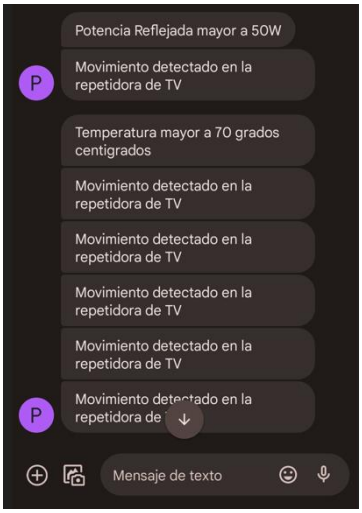


Figura 3.19 Mensajes recibidos desde el módulo TCALL V1.4

3.3 PRESUPUESTO REFERENCIAL

El presupuesto del proyecto comprende los componentes electrónicos de hardware y software utilizados para el desarrollo del prototipo.

Tabla 3.3 Costo hardware y software

Cantidad	Descripción	V. Unit. [$]	Total [$]
21	Condensador electrolítico	0,1	2,1
15	Condensador cerámico	0,1	1,5
8	Resistencia through-hole	0,04	0,32
50	Resistencia SMD	0,1	5
3	Amplificador operacional LM358	0,42	1,26
1	Amplificador operacional CA3240	3,48	3,48
1	Inversor schmitt-trigger 40106	0,62	0,62
1	Circuito contador 4017	0,82	0,82
2	Regulador de voltaje L7812	0,69	1,38
3	Regulador de voltaje L7805	0,7	2,1
1	Sensor de temperatura LM35	2,49	2,49
6	Transistor bipolar BC548	0,36	2,16
11	Diodo 1N4148	0,1	1,1
1	Diodo 1N4001	0,1	0,1
1	Diodos schottky	0,4	0,4
1	Mosfet tipo P SI2305TP	0,5	0,5
7	Led	0,1	0,7
8	potenciómetro	0,5	4
1	Relé 12 voltios	0,8	0,8
5	Conector Molex 2 pines	0,5	2,5
2	Conector Molex 3 pines	0,5	1
1	Conector Molex 12 pines	2	2
4	Conectores hembra 2.54mm	0,5	2
2	Pulsadores	0,1	0,2
1	Cable UTP (1 metro)	0,5	0,5
1	Kit cables micro USB y USB C	2	2
1	Power bank	4	4
1	Sensor PIR HC-SR501	2,9	2,9
1	Pantalla LCD 16X2	6,5	6,5
1	Módulo NodeMCU	13	13
1	Módulo TCALLV1.4	29	29
1	Fabricación de PCB y transporte	120	120
1	Hosting 1 año con dominio	23	23
TOTAL (incluye IVA)			**239,43**

3.4 ANÁLISIS Y DISCUSIÓN DE RESULTADOS

Al recolectar las muestras de las señales analógicas de voltaje DC, corriente DC, potencia directa, potencia reflejada y temperatura, la presencia de ruido provoca que las lecturas en los pines analógicos del módulo NodeMCU varie mucho. Por este motivo, al implementar el promedio de las muestras, se puede corregir en gran medida este error, pero no se lo puede eliminar por completo.

Durante las pruebas, los efectos de la radio frecuencia sobre el cable del sensor de temperatura provocaron lecturas erróneas por parte del módulo NodeMCU, lo que generó la alerta por temperatura elevada. Para resolver este inconveniente, se separó el cable del sensor de la parte de radio frecuencia, permitiendo que la lectura de las muestras sea adecuada.

Las mediciones de potencia directa presentan errores relativos de alrededor del 10 por ciento para potencias bajas, como se aprecia en la Figura 2.23, la curva de interpolación en valores de potencia inferiores a 50 vatios difiere mucho con respecto a la curva realizada con los valores medidos con el vatímetro. Se podría mejorar cambiando el tipo de interpolación. También se podría cambiar los sensores de potencia por otros más exactos.

Durante las pruebas de envío de las señales de alerta, se determinó la conveniencia de enviar las alarmas al inicio del evento, guardando el estado de las alarmas en la memoria permanente. Para que el sistema vuelva a enviar un mensaje de alerta del mismo tipo, el módulo NodeMCU debe registrar que los valores de las señales volvieron a sus rangos nominales y posterior a ello volvieron a sobrepasar el límite máximo definido.

El empleo de WordPress mejoro la interacción que tiene el usuario con la base de datos, sin necesidad de programar desde cero todas las páginas web empleando HTML y PHP.

La página web implementa condiciones básicas de seguridad que se pueden mejorar, pero para el desempeño de esta aplicación de monitoreo es funcional.

4. CONCLUSIONES Y RECOMENDACIONES

4.1 CONCLUSIONES

- La radio frecuencia sobre el circuito electrónico empleado para la toma de muestras con el módulo NodeMCU causó mediciones erróneas en algunos datos debido a que no se utilizaron normas para el diseño de tarjetas de circuito impreso.
- Se puede reducir el número de muestras erróneas en los pines analógicos del NodeMCU realizando un promedio al obtener múltiples muestras.
- El monitoreo de la potencia puede ser mejorado incluyendo sensores de mejor calidad que proporcionen datos más exactos.
- Las pruebas en laboratorio mostraron que la comunicación inalámbrica entre módulos ESP32 a través del protocolo esp-now es factible. Considerando que en el ambiente de las estaciones repetidoras hay más señales de radio frecuencia, se debe realizar pruebas de comunicación inalámbrica entre los módulos ESP32 en una estación repetidora para analizar los efectos en la comunicación.
- El sistema implementado cumple el objetivo de desarrollar el monitoreo de la estación repetidora de televisión analógica.
- Es conveniente incluir operaciones de forma remota que ejecuten acciones de control sobre el sistema de monitoreo, para lo cual se debe mejor la seguridad en la página web implementada.
- Utilizar el sistema de gestión de contenido WordPress facilita la implementación de las páginas web para el proyecto, mejorando la interacción del usuario final con la página web.

4.2 RECOMENDACIONES

- Para la recolección de datos por los pines analógicos del NodeMCU, es necesario colocar los sensores cerca de los pines analógicos. Si se utiliza cables para los sensores, estos deben tener apantallamiento.
- Es recomendable diseñar las tarjetas de circuito impreso basadas en normas de diseño, recomendaciones de montaje y ensamble de componentes, como por ejemplo las normas IPC.
- Mejorar el procedimiento de toma de datos de potencia con el vatímetro para asegurar que los valores de referencia para la interpolación sean adecuados.

- Se debe seleccionar el hosting web adecuado considerando el alcance del proyecto, la seguridad y el presupuesto.
- Es importante utilizar el protocolo HTTPS para establecer una conexión segura entre el cliente y el servidor.
- Utilizar el entorno de desarrollo de software de espressif ESP-IDF para aprovechar todos los componentes de hardware que tienen los módulos ESP32.
- Implementar el control remoto del transmisor de televisión analógica para ejecutar acciones preventivas o correctivas.

5. REFERENCIAS BIBLIOGRÁFICAS

[1] C. Pérez Vega, *Introducción a los sistemas transmisores de TV,* Santander: Gráficas Calima SA, 2005.

[2] LILYGO, “TTGO T-Call V1.4 ESP32 Wireless Module SIM Antenna SIM Card SIM800L Module, ” [En línea]. Available: http://www.lilygo.cn/prod_view.aspx?TypeId=50033&Id=1127&FId=t3:50033:3. [Último acceso: 13 Octubre 2022].

[3] C. P. Millahual, PHP 7-Sitios Dinámicos: Aprenda a programar sin conocimientos previos, RedUsers, 2019.

[4] Á. Arias, Bases de Datos con MySQL: 2\textordfeminine Edición, IT Campus Academy, 2014.

[5] R. Santos, “Visualize Your Sensor Readings from Anywhere in the World (ESP32/ESP8266 + MySQL + PHP), ” 2022. [En línea]. Available: https://randomnerdtutorials.com/visualize-esp32-esp8266-sensor-readings-from-anywhere/. [Último acceso: 17 Noviembre 2022].

[6] T. P. Group, “php, ” 2022. [En línea]. Available: https://www.php.net/. [Último acceso: 20 Noviembre 2022].

[7] G. B., “Cómo usar PHP para insertar datos en MySQL, ” 12 Diciembre 2022. [En línea]. Available: https://www.hostinger.es/tutoriales/como-usar-php-para-insertar-datos-en-mysql/#Solucion_de_errores_comunes. [Último acceso: 20 Diciembre 2022].

[8] “¿Qué Es un Sistema de Gestión de Contenidos (CMS)?, ” 19 Noviembre 2020. [En línea]. Available: https://kinsta.com/es/base-de-conocimiento/sitema-de-gestion-de-contenido/. [Último acceso: 20 Noviembre 2022].

[9] Ó. T. Artero, ARDUINO. Curso práctico de formación, RC libros, 2013.

[10] “Lenguaje de programación de Arduino, estructura de un programa, ” [En línea]. Available: https://aprendiendoarduino.wordpress.com/2015/03/26/lenguaje-de-programacion-de-arduino-estructura-de-un-programa/. [Último acceso: 12 Agosto 2022].

[11] “AVR Libc Home Page, ” 09 Febrero 2016. [En línea]. Available: https://www.nongnu.org/avr-libc/. [Último acceso: 12 Agosto 2022].

[12] Espressif, “ESP-NOW User guide, ” 2016. [En línea]. Available: https://www.espressif.com/en/support/documents/technical-documents?keys=esp+now. [Último acceso: 12 Noviembre 2022].

[13] E. Sistems, *ESP32-WROOM-32 Datasheet,* 2022.

[14] Ai-Thinker, *Nodemcu-32s Datasheet,* Shenzhen, 2019.

[15] E. Systems, *ESP32WROVERB & ESP32WROVERIB Datasheet,* 2022.

[16] T. Instruments, *LM35 Precision Centigrade Temperature Sensors datasheet,* Texas, 2000.

[17] A. Moreno-Muñoz y S. C. Córcoles, Arduino: curso práctico, Ra-Ma, 2018.

[18] P. F. S.A, *Sensor infrarrojo de movimiento PIR HC-SR501 Datasheet,* 2017.

[19] J. P. López, GUÍA PRÁCTICA PARA RASPBERRY PI Y BEAGLEBONE, España: MARCOMBO, S.A., 2018.

[20] H. Technology, *I2C Serial Interface 1602 LCD Module User Guide.*

[21] R. Santos, “ESP-NOW with ESP32: Receive Data from Multiple Boards (many-to-one), ” RANDOM NERD TUTORIALS, 2020. [En línea]. Available: https://randomnerdtutorials.com/esp-now-many-to-one-esp32/. [Último acceso: 20 Julio 2022].

[22] Harris Corporation, *Television Transmiters Technical Manual,* 2000.

[23] R. Santos, “ESP32 Publish Data to Cloud without Wi-Fi (TTGO T-Call ESP32 SIM800L), ” RANDOM NERD TUTORIALS, [En línea]. Available: https://randomnerdtutorials.com/esp32-sim800l-publish-data-to-cloud/. [Último acceso: 21 Julio 2022].

[24] “Lilygo, ” ESP32 WROVER B, [En línea]. Available: http://www.lilygo.cn/claprod_view.aspx?TypeId=62&Id=1403&FId=t28:62:28. [Último acceso: 15 Agosto 2022].

[25] STMelectronics, *POSITIVE VOLTAGE REGULATORS DATASHEET,* 2004.

ANEXOS

ANEXO A: CIRCUITOS DE REFERENCIA (ANEXO DIGITAL)

ANEXO B: CÓDIGO IMPLEMENTADO EN EL NODEMCU (ANEXO DIGITAL)

ANEXO C: CÓDIGO IMPLEMENTADO EN EL TCALL V1.4 (ANEXO DIGITAL)

ANEXO D: CONEXIÓN DE BASE DE DATOS MYSQL CON PÁGINA WEB CON CONTROL DE USUARIOS EN WORDPRESS (ANEXO DIGITAL)

ANEXO E: DIAGRAMAS ESQUEMÁTICOS DE LAS PCB1 Y PCB2 (ANEXO DIGITAL)

ORDEN DE EMPASTADO