TRABAJO ESPECIAL DE GRADO

DISEÑO E IMPLEMENTACIÓN DE UNA RED WIFI MALLADA QUE SOPORTE PROTOCOLO MODBUS PARA EQUIPOS DE CONTROL INDUSTRIAL

Presentado ante la Ilustre
Universidad Central de Venezuela
por el Br. Adrian Vazquez
para optar al título de
Ingeniero Electricista.

Caracas, Mayo de 2020

TRABAJO ESPECIAL DE GRADO

Diseño e implementación de una red WiFi mallada que soporte protocolo Modbus para equipos de control industrial.

TUTOR ACADÉMICO: Profesor José Alonso

Presentado ante la ILUSTRE
Universidad Central de Venezuela
por el Br. nombres y apellidos para optar
al título de Ingeniero Electricista.

Caracas, Junio de 2020

A quien desees dedicar este trabajo

RECONOCIMIENTOS Y AGRADECIMIENTOS

Adrian Vazquez

Diseño e implementación de una red WiFi mallada que soporte protocolo Modbus para equipos de control industrial.

Tutor Académico: José Alonso. Tesis. Caracas, Universidad Central de Venezuela. Facultad de Ingeniería. Escuela de Ingeniería Eléctrica.Opción Electrónica, Computación y Control. Año 2020. Pág. 82 + anexos.

Palabras Claves: WiFi, Modbus, mallada, inalámbrica, nodo, enrutamiento, microcontrolador, ESP32, interfaz serial, RS-485.

Resumen. Este trabajo presenta la implementación y el diseño de una red mallada inalámbrica WiFi para la transmisión del protocolo Modbus a modo de sustitución del bus RS-485 entre el maestro y los esclavos. Se documentan los tipos de red malladas mas comunes que usen microcontroladores o estén orientadas al área industrial, se hace una comparación. Posteriormete se diseña una estructura de red mallada con enrutamiento estático que soporte la transmisión del protocolo Modbus entre los nodos de la red. Luego, se diseña el hardware que con el chip de acople serial RS-485, el microcontrolador ESP32 y un regulador lineal. Finalmente se ejecutan pruebas de rendimiento contando con cuatro nodos, variando la tasa de baudios, la topología, el número de variables Modbus y el tiempo de muestreo. Se logró implementar satisfactoriamente la red con compatibilidad con equipos Modbus RTU serial con interfaz serial RS-485 y presentar análisis de respecto a los parámetros que afectar su eficacia.

ÍNDICE GENERAL

LISTA DE FIGURAS

LISTA DE TABLAS

LISTA DE ACRÓNIMOS

BSS: Conjunto de servicios básicos.

MAC: Control de acceso al medio.

NIC: Controladores de interfaz de red.

AP: Punto de acceso.

STA: Estación.

WMN: Red mallada inalámbrica.

WLAN: Red de área local inalámbrica.

ADU: Unidad de datos de aplicación.

PDU: Unidad de datos de protocolo.

CCMP: Protocolo de código de autenticación de mensaje de encadenamiento de bloque de cifrado en modo contador.

DS: Sistema de distribución.

FCS:Trama de chequeo de errores.

PMK: Llave maestra primaria.

LMK: Llave maestra local.

MIFA: Antena F invertida.

EVM: Magnitud del vector de error.

UART: Transmisor-receptor asíncrono universal.

INTRODUCCIÓN

Las telecomunicaciones se pueden clasificar en dos grupos: las alámbricas y las inalámbricas. En el primer caso el medio es tangible estando, generalmente, compuesto por varios conductores metálicos o fibra óptica y en el segundo no hay medio físico. En las comunicaciones inalámbricas existen diversos estándares que señalan su comportamiento y permiten la interoperabilidad de equipos que usen la misma tecnología, algunos de ellos son: IEEE 802.15.1 (Bluetooth), IEEE 802.11 (WiFi) y GSM (telefonía celular).

Las tecnologías WiFi han ganado popularidad actualmente en el mundo de las comunicaciones, usando para la interconectividad entre clientes (PC, laptops, smartphones) y puntos de acceso. Las redes que usan dicha tecnología se pueden clasificar en centralizadas o descentralizadas, dependiendo si la conectividad entre clientes depende o no de un punto de acceso (Sharma y Singh, 2016).

Dentro de las redes descentralizadas tenemos a las redes de topología mallada, donde los clientes están en capacidad de ser, simultáneamente, puntos de acceso brindando servicio a otros clientes y servir de puente a otros nodos. Además, las redes WiFi malladas difieren de las convencionales en que no es obligatorio que todos los nodos estén conectados al nodo central, en su lugar, cada nodo se puede conectar a el nodo vecino. Esto abre la posibilidad de ampliar la cobertura manteniendo la interconectividad de la red.

Las redes WiFi brindan la posibilidad de que la adquisición de información pueda estar presentes en virtualmente cualquier cosa de manera inalámbrica, poseyendo potencialidad en el monitoreo y control de procesos. No obstante, los inconvenientes de ruido, seguridad y distancia han hecho que se hayan desarrollado

arquitecturas que vayan superando estas limitaciones.

Aunque el estándar IEEE 802.11s establece las normas generales de la comunicación WiFi mallada (Hiertz y cols., 2010), todavía es un terreno actualmente se encuentra en exploración, especialmente en los entornos electromagneticamente ruidosos o congestionados. En el mismo orden de ideas, existen reservas respecto al manejo de datos críticos o sensibles en un proceso industrial, debido a la confiabilidad y seguridad de los datos(Cheng y cols., 2018).

Modbus es el protocolo de comunicación industrial estándar de facto desde 1979 (Modbus Organization, 2017). Al emplear un protocolo Modbus se establece un sistema de comunicación de tipo maestro/esclavo. En este tipo de sistemas un nodo principal o maestro envía una solicitud específica a un esclavo y este genera la respuesta. Los esclavos no transmiten datos sin una instrucción previa y no se comunican con ningún otro esclavo. En la capa física del sistema de comunicación, el protocolo Modbus puede emplear diversas interfaces, entre las que se encuentran la RS-485 y RS-232, donde la interfaz TIA/EIA-485 (RS-485) de dos cables es la más común.

En el ámbito industrial la reducción de costos es siempre una meta y las redes inalámbricas podrían ser una alternativa frente a las alámbricas, en las que las grandes distancias cubiertas por conductores representan un costo relativamente elevado (Cheng y cols., 2018). Así mismo, los conductores instalados son vulnerables a hurtos, lo que se traduce en pérdidas económicas para las industrias.

Tomando en cuenta lo anterior, se propone el diseño de una red inalámbrica WiFi mallada con una comunicación basada en el protocolo Modbus orientada a la implentación en un entorno industrial, cuyos nodos estarán constituidos por microcontroladores ESP32.

En el capítulo I se expone el problema se da la justificación y alcance del trabajo. Los objetivos también de presentan en este capítulo. Más adelante en el capítulo II, se desarrollan las bases teóricas de sirven como apoyo para explicar los conceptos de las redes WiFi malladas, el protocolo Modbus y el microcontrolador ESP32.

El capítulo III se encarga de la descripción de las metodologías empleadas para alcanzar los objetivos del trabajo, profundizando en las etapas del diseño e implementación de hardware y software.

El capítulo IV se presentan los resultados y análisis de las pruebas de rendimiento efectuadas a la red. Finalmente, se exponen las conclusiones obtenidas mediante los estudios realizados en el trabajo y se efectuan recomendaciones para futuros estudios relacionados.

CAPÍTULO I

PRESENTACIÓN Y DESCRIPCIÓN

Planteamiento del problema

La creciente popularidad de las redes inalámbricas en casi todos los sectores ha hecho que las infraestructuras asociadas, dispositivos y protocolos se vean en la necesidad de mejorar constantemente para manejar la creciente cantidad de usuarios de manera segura y eficiente. Así mismo, las redes WiFi centralizadas están limitadas por la capacidad del punto de acceso, en el ámbito de cobertura y cantidad de clientes(Singh, 2019) , restricciones que se pudiesen superar con las redes WiFi malladas.

Actualmente la redes malladas están en una etapa de desarrollo, por lo que no existe un estándar sólido para la implementación de toda la red, lo que causa reservas en las industrias, especialmente debido a la vulnerabilidad de los datos, la confiabilidad en ambientes electromagnéticamente ruidosos y el rendimiento (Milic, Brack, y Naumann, 2014). Es por eso que una red mallada WiFi con comunicación basada en el protocolo Modbus podría representar una solución a este problema.

Justificación

Se plantea una red WiFi lo cual representaría una reducción de costos en la implementación de un sistema de control y adquisición de datos, dado que se necesitan menos conductores. Además, se explorará el campo popular hoy en día de las redes WiFi y su rendimiento como red mallada. Dicha red representa una alternativa a superar la limitaciones de número de clientes y área de cobertura presentes en las redes WiFi centralizadas, y más aún, supone una solución a llegar a lugares lejanos de un nodo central sin la necesidad de agregar puntos de acceso adicionales.

La constitución de la red mallada se elaborará basados en comunicación WiFi a través de microcontroladores ESP32, que poseen características de tamaño, potencia y costos aunado a las ventajas principales de las redes mallada de cobertura y conectividad. También se sustentará la transmisión de información en el protocolo Modbus debido a su confiabilidad todos los sectores aplicables, especialmente el industrial.

Alcance y limitaciones

La red se compondrá de al menos cuatro nodos, donde cada nodo tendrá conexión con al menos otro nodo usando red WiFi y estarán constituidos por microcontroladores ESP32 con módulo WiFi integrado y el programa asociado a la red. Los nodos deben estar apropiadamente alimentados, cuyo diseño no forma parte del proyecto.

El programa se diseñará para que se logre transportar la información bajo el protocolo Modbus por la red inalámbrica, considerando que solo en un nodo está conectado el maestro. Así mismo, algunos de los nodos restantes poseerán

esclavos. Cabe resaltar que las unidades que generan los datos del protocolo Modbus (maestro y esclavos) no forman parte de la red a diseñar, ya que se asume que se recibe información en los nodos sin tener en cuenta mayor detalle de su origen.

1.1. Objetivos

1.1.1. Objetivo General

Diseñar e implementar una red WiFi mallada que soporte protocolo Modbus usando microcontroladores ESP32 para equipos de control industrial.

1.1.2. Objetivos específicos

1. Documentar el funcionamiento de las redes WiFi malladas.
2. Diseñar una red mallada basada en el microcontrolador ESP32.
3. Implementar el módulo del programa para el manejo del protocolo Modbus en el bus RS-485.
4. Implementar el programa de la red diseñada que soporte la transmisión del protocolo Modbus.
5. Diseñar el circuito de un nodo para una red WiFi mallada basada en el microcontrolador ESP32.
6. Implementar la red mallada diseñada.
7. Analizar el rendimiento de la red de acuerdo a variaciones en los parámetros de transmisión de datos.

1.2. Antecedentes

Se tiene el trabajo de grado realizado por por Alejandro J. Temprano G., titulado *Diseño de una Red Industrial Modbus RTU* inalámbrica, en cuyo trabajo diseño e implementó una red usando la banda de radiofrecuencia de 433MHz basándose en el transceptor RFM69HW (Temprano G., 2017).

Este trabajo contemplo la transmisión inalámbrica del protocolo Modbus, sin embargo, la realizó en una topología radial, donde todas las compuertas de enlace con esclavos están al alcance de las señales provenientes del maestro. Por lo que no existe la posibilidad de múltiples saltos. No obstante, provee observaciones en relación a la comunicación con equipos Modbus que fueron tomadas en cuenta para el diseño de la red que contempla este trabajo.

CAPÍTULO II

MARCO TEÓRICO

En este capítulo se tratarán los fundamentos de las redes WiFi malladas, protocolo Modbus y del microcontrolador ESP32.

2.1. Fundamentos de las redes WiFi malladas

Una red WiFi mallada (WMN) se puede definir como una red que permite la comunicación entre nodos a través de múltiples saltos en una topología mallada (Bahr, 2016). Los nodos son los intermediarios encargados de la formación de la red y la vinculación entre los dispositivos que la usan. Una WMN usualmente poseen clientes, enrutadores y puertas de enlace. Los clientes son dispositivos electrónicos, sistemas embebidos o sensores que pueden comunicarse con otros en la red. El enrutador es un dispositivo electrónico que sirve como un intermediario entre dos o más redes para transportar los datos de una red a otra. Y las compuertas de enlace son dispositivos electrónicos que conectan la red con Internet.

Cuando un nodo no puede operar, el resto de los nodos en la WMN aún pueden comunicarse con los otros, bien sea, directa o indirectamente a través de uno o más nodos intermediarios (Rifki Muhendra, 2016).

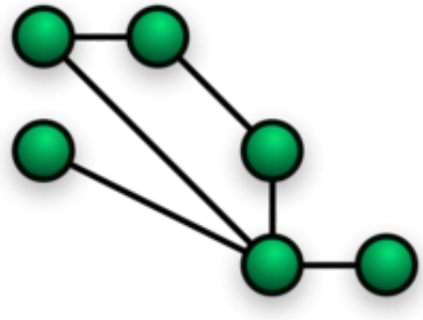


Figura 2.1: Topología Mallada

Una red WiFi mallada se establece en la banda para comunicación WiFi (sea 2,4 GHz o 5GHz), donde hay estaciones (nodos) que soportan comunicación de múltiples saltos (multi-hop) para transferir información en la red. Así mismo se tiene que, el enrutamiento y la capacidades de reenvios de datos residen en la capa de Control de Acceso al Medio (MAC) (Hiertz y cols., 2010).

La capa de Control de Acceso al Medio (MAC)

La capa MAC administra el acceso a un medio compartido, proveyendo sincronización entre diferentes nodos para permitir la transmisión inalámbrica. Dicha sincronización es cada vez más esencial para la red en tanto el método de acceso es más complejo. Como un ejemplo, la sincronización puede ser necesaria entre nodos de un sistema que emplee un espectro abierto. Otro ejemplo, nodos individuales pudiesen necesitar permiso de un controlador en una red inalámbrica para transmitir en un canal dado. La capa MAC administra las negociaciones con un nodo de control para el acceso al medio.

Las funciones específicas que están encapsuladas en la capa MAC, varían de un protocolo a otro. Incluyen, pero no están limitadas a, técnicas de acceso múltiple, sincronización un medio abierto y corrección de errores (Chew, 2018).

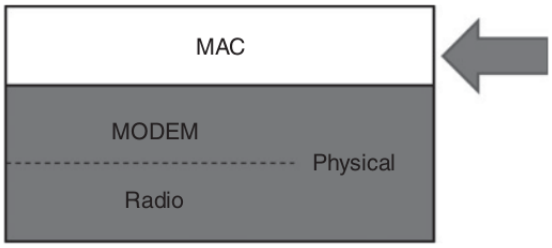


Figura 2.2: Localización de la MAC sobre la interfaz física

Dirección MAC

En una red de computadoras, la dirección MAC es un valor único asociado a un adaptador de red. La dirección MAC también es conocida como dirección de hardware o dirección física y se ubica en la capa de vínculo del modelo OSI.

Las direcciones MAC se componen de doce número hexadecimales (48 bits de longitud). Por convención las direcciones MAC son escritas en uno de los dos formatos siguientes:

$$MM:MM:MM:SS:SS:SS \quad o \quad MM-MM-MM-SS-SS-SS \quad (2.1)$$

La primera mitad de la dirección MAC contiene el número identificador del fabricante del adaptador(i.e. 00:A0:C9:14:C8:29). Dichos identificadores están regulados por un estándar de Internet. La segunda parte de la dirección MAC representa el número de serial asignado al adaptador por el fabricante. La suplantación de MAC es equivalente a hacerce cargo de los controladores de interfaz de red (NIC). La unicidad de la dirección MAC es esencial en todas la fases de la comunicación de red porque mapea todos los identificadores de las capas superiores (Al-Husainy, 2013).

2.2. La Referencia del Modelo OSI

El modelo del sistema de interconexión abierta (OSI) está basado en una propuesta del la Organización Internacional de estándares (ISO,) para la la estandarización de pilas de protocolos. Para clasificar un protocolo usando el modelo OSI se deben cumplir ciertos requerimientos (Chew, 2018).

1. Una capa debe existir para cada nivel de abstracción.
2. Cada capa debe ejecutar una función bien definida.
3. La función de cada capa debe formar parte de un estándar internacional.
4. Los límites de cada capa deben minimizar el flujo de información a través de las interfaces.
5. El número de capas debe de suficientemente grande para no forzar múltiples funciones en una sola capa, pero lo suficientemente pequeña para que la arquitectura no sea incómoda.

El modelo consiste en siete (7) capas, juntas son solo un modelo de referencia, y no arquitectura de red. Los estándares de la ISO existe para varios niveles y no son parte de este modelo. Las capas son:

1. La capa de aplicación.
2. La capa de presentación.
3. La capa de sesión.
4. La capa de transporte.
5. La capa de red.

6. La capa de vínculo de datos.

7. La capa física.

Para explicar las funcionalidades y la interrelación entres estas capas, es beneficioso estudiarlas desde la capa física.

2.2.1. La capa Física

A la capa capa física le concierne la transmisión de los bits de data cruda sobre el canal de comunicación. Es responsable de la integridad de dichos bits, tanto por la entrega como por la interpretación. Los detalles específicos de cuantos Volts representan el cero lógico y cuantos representan el uno, la duración de la señal, el mecanismo de conexión y desconexión, etc., son dependientes de los medios físicos y los dispositivos empleados.

2.2.2. La capa de vínculo de datos

La capa de vínculo de datos provee la primera capa de abstracción en la pila. Esta protege la capa de red de detalles de nivel bajo y errores de la capa física. Esto es logrado agrupando los bits crudos en una unidad de nivel más alta llamada trama de datos, la cual puede ser usada en la capa de red.

La trama de datos consiste en un grupo de bytes. Patrones especiales de bits delimitan la carga, para que la trama de datos sea reconocida. Esto significa que especial cuidado se debe posee para asegurar que estos patrones especiales no ocurren dentro de la carga, en cuyo caso la trama se perdería. Un mecanismo apropiado debe existir para notificar que la fuente retransmite la trama.

Otra característica en esta capa es la inclusión del control de flujo y los agradecimientos. En las redes que están basadas en un canal compartido, una subcapa ha sido introducida en la capa de de vínculo de datos, para manejar el control de acceso, con el nombre de subcapa de control de acceso al medio (MAC).

2.2.3. La capa de red

Esta capa es la responsable por controlar la operación de la subred. La carga de la trama de datos, en esta capa, es llamada paquete y determina cómo moverlo desde la fuente al destino usando las rutas apropiadas, la determinación de dichas rutas puede puede ser estática o dinámica. Esta capa también maneja la congestión de la red.

La capa de red tiene que lidiar con problemas relacionados con las diferentes arquitecturas de red, diferentes direccionamientos, y diferentes condiciones de operación y restricciones, tanto en los sistemas de origen como en los de destino. La heterogeneidad de red es tomada en cuenta a este nivel.

2.2.4. La capa de transporte

La función básica de la capa de transporte es aceptar datos de un capa más alta, descomponerla en unidades más pequeñas, si es necesario y asegurarse que estas piezas llegar correctamente al destino. Para propósitos de eficiencia, se puede multiplexar varias conexiones de transporte.

Es la primera capa fin-a-fin de la pila. En las capas más bajas, la interacción real no necesitaba estar entre los sistemas de de fuente y destino. Enrutadores o sistemas intermedios podían ser parte de la transacción. La interacción en este nivel, sin embargo, es siempre entre puntos finales.

El control de flujo juega un rol importante en esta abstracción.

2.2.5. Capa de sesión

La capa de sesión provee algunos servicios adicionales comparados a la capa de transporte. Como por ejemplo incluye el manejo de suscripción, transferencia de archivos y manejo de tokens.

Otro servicio de sesión es la sincronización, y proveer las funciones para la inserción de puntos de revisión dentro de los datos que se están transmitiendo, para que la reanudación o reconexión de datos pueda llevarse a cabo.

2.2.6. La capa de presentación

La capa de presentación es la encargada de la sintaxis y la semántica de la información transmitida. Un ejemplo típico es la codificación y decodificación de los datos. Para que los datos sean correctamente interpretados en cada punto, debe haber una codificación estándar. La capa de presentación provee los servicios para manejar la conversión de las estructuras de datos del usuario a la red, y *vice versa*.

2.2.7. La capa de aplicación

Esta es la capa más familiar para el usuario, la cual comprende varios protocolos. El ejemplo más famoso incluye los clásicos protocolos de terminales. Las definiciones de protocolos en esta capa son un nivel alto, normalmente entendibles para el usuario.

Protocolos de comando-respuesta y basados en texto, forman parte de esta

capa.

2.3. Funcionamiento de las redes WiFi malladas

La implementación de la topología mallada ha encontrado problemas con la necesidad de procedimientos adicionales relacionados con el enrutamiento. Hay algunos protocolos que soportan el servicio de red mallada sobre la red IP, por ejemplo: B.A.T.M.A.N. (Better Approach To MobiLle Adhoc Networking), Babel (Protocolo de distancia de vector para IPv6 y IPv4 con propiedades de convergencia rápida), HWMP (Protocolo Híbrido Inalámbrico Mallado). El uso de estos protocolos requiere la completa implementación de la pila TCP/IP y un poder de computación significativo, lo cual limita sus implementaciones. Sin embargo, no todos los equipos (para comunicaciones WiFi) soportan un protocolo particular(Lech y Wlodarski, 2017).

El uso de microcontroladores avanzados incrementa altamente el costo de la construcción de la red, que muchas veces no son necesarios para aplicaciones donde se necesita obtener información sobre los procesos lentos a través de mediciones periódicas. Un amplio rango de módulos simples WiFi hechos como Sistemas en un Chip (SoC), que además de manejar estándares de comunicación, pueden adquirir datos a través de entradas y salidas de propósito general. Estos abren la posibilidad para la construcción de sensores de red de bajo costo en la ampliamente usada WMN. Sin embargo, cuando muy poco poder de procesamiento no permite la implementación de algoritmos avanzados que soporten el enrutamiento IP en la topología mallada, es posible crear una red simplificada mientras se mantienen las características principales de las redes malladas.

A continuación se describe el funcionamiento de algunas de redes WiFi malladas, es de observar que existe una amplia gama de funcionamientos, cada una

caracterizada por su aplicación, se expondrán algunas relacionadas con el objetivo del trabajo.

2.3.1. Red mallada para Internet de las cosas (IoT)

Pior Lech and Przemyslaw Wlodarski en su artículo *Analysis of the IoT WiFi Mesh Network* (Análisis de las redes WiFi malladas IoT) (Lech y Wlodarski, 2017), llevan a cabo un análisis estadístico de rendimiento sobre una red WiFi mallada. Para lograrlo usan la versión de desarrollo del módulo de comunicación NodeMCU ESP8266. La operación básica de cada nodo en la red es el modo AP+STA (Punto de acceso y estación),todos los nodos posee un número fijo asignado que crece desde la fuente (RPi 1) en la dirección de la estación destino (RPi 2). La estrategia de entre los nodos está basada en la transmisión del mensaje a los nodos con un número mayor al asociado a la estación transmisora.

De acuerdo a la figura 2.3 se pueden seleccionar las siguientes rutas: 1-2-5, 1-2-3-5, 1-2-4-5, 1-3-5 y 1-4-5. El número asignado esta estrechamente relacionado con las direcciones IP. Los mensajes son enviados a través del protocolo UTP. La fuente de los mensajes es la microcomputadora Raspberry Pi v.2 (Rpi 1) la cual envía mensajes a el nodo 1. Los módulos NodeMCU duplican el mensaje y lo reenvían acorde a la estrategia antes mencionada. Todo el tráfico de datos termina en el segundo Raspberry Pi (RPi 2) a través del nodo 5. Los nodos que llevan a cabo la duplicación del mensaje, envían estos en un ciclo, del menor al mayor número asociado con el nodo.

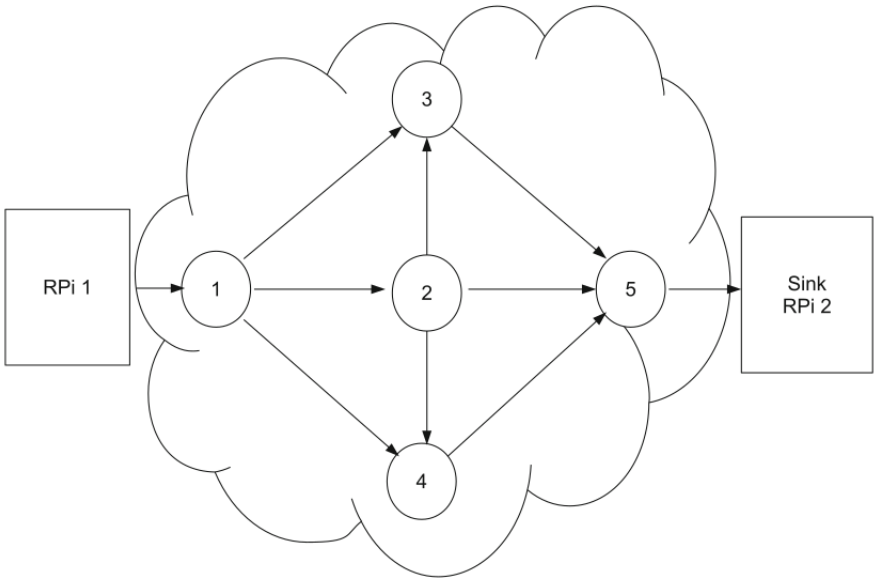


Figura 2.3: Topología usada por Lech y Wlodarski (Lech y Wlodarski, 2017)

2.3.2. Red mallada y el balaceo de carga

Yujun Cheng, Dong Yang y Huachun Zhou en su artículo *A Load Balancing Approach in 802.11 Wireless Networks for Industrial Soft Real-Time Applications* (Cheng y cols., 2018) (Un Enfoque de Carga Equilibrada en Redes Inalámbricas 802.11 para Aplicaciones Industriales Ligeras en Tiempo Real) propone una arquitectura basada en el que distribución de los nodos esta directamente relacionada con la cantidad de enlaces que posee cada uno, de manera de distribuirlos equitativamente.

El estándar 802.11 no define ningún mecanismo para el balanceo de carga. Casi todos los adaptadores 802.11 se asocian con el punto de acceso que posea la mayor intensidad de señal, por que las redes son propensas a una distribución desigual de recursos, lo que significa que algunas AP exceden o se acercan a la capacidad de carga máxima, mientras que la de otras permanecen relativamente baja. En este

enfoque, el proceso de asociación de las estaciones no está simplemente relacionado con la intensidad de señal, sino que también está basado en la carga que posee cada AP. El algoritmo provee una compensación entre la intensidad de señal y la carga, cambiando las estaciones de los punto de acceso sobrecargados con una intensidad de señal alta, a un punto de acceso vecino menos cargado y la intensidad de señal que pudiese ser más débil.

En la red enfocada al balanceo, una unidad central llamada controlador de red es usada para administrar el balanceo de las cargas. El controlador de red pudiese actuar como un simple punto de acceso o como una entidad independiente directamente conectada a la central cableada. Cada punto de acceso envía su información al controlador de red, y así el controlador conoce la condición básica global de la red. La arquitectura jerárquica de la red del enfoque se muestra en la figura 2.4.

Un algoritmos es el encargado de balancear la red, el cual está basado en revisiones métricas y un proceso de distribución de carga. Tomando la topología de la Figura 2.4 como un ejemplo donde hay más de dos estaciones conectadas a AP2 y AP3 comparadas con la situación de carga de la AP1 y AP4. Así, la carga de la red esta relativamente desbalanceada; si las características de tráfico de cada estación son similares, por lo tanto, la red requiere un algoritmo de balanceo específico. El algoritmo de revisión métrica comienza cuando una estación (STA1) es alertada de una situación de potencial desbalanceo. Este mismo algoritmo verifica las métricas designadas y decide si el nodo en cuestión está o no sobrecargado. Si la métrica se encuentra más allá de determinado límite, entonces la estación STA1 debe decidir si abandonar la actual AP (AP2) y enviar una solicitud de disociación a el controlador, o por otro lado mantener la conexión. Si STA1 decide desconectarse, entonces el controlador de red ejecuta el algoritmo de distribución de carga y distribuye la estación a otra AP en el

área de solapamiento que posee menor carga, tal como AP1. Antes de que el algoritmo de distribución de carga termine, la estación STA1 deberá poseer un mejor rendimiento, así como las estaciónes que aún estarías asociadas con AP2.

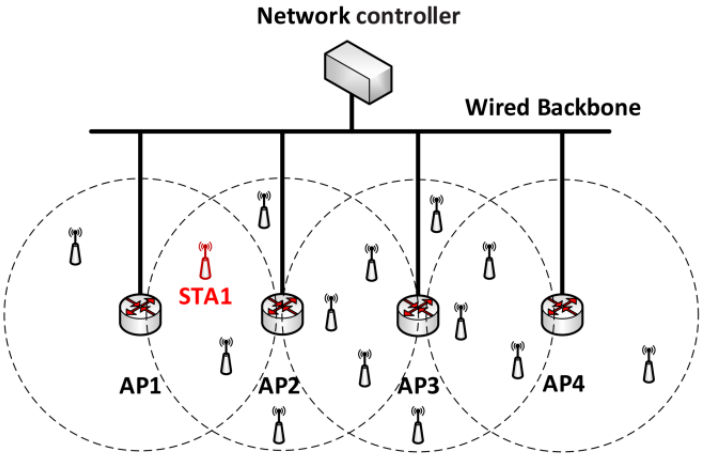


Figura 2.4: Topología usada por Yujun Cheng, Dong Yang y Huachun Zhou (Cheng y cols., 2018)

2.3.3. El protocolo ESP-MESH de Espressif (Systems, 2016a)

ESP-MESH es un protocolo de red construido sobre del protocolo WiFi. Dicho protocolo permite que numerosos dispositivos(Nodos) posicionados sobre un área física estén interconectados bajo única Red de Área Local Inalámbrica (WLAN). Las redes ESP-MESH son auto-organizadas y auto-reparables, es decir, que la red pueden ser construidas y mantenidas de manera autónoma (Systems, 2016a).

ESP-MESH permite que los nodos actúen simultáneamente como estación y como punto de acceso (AP). Por lo tanto un nodo puede poseer múltiples conexiones de estaciones a su punto de acceso, mientras que su estación posee una única conexión a un punto de acceso de una capa superior. Lo que naturalmente resulta en una topología de árbol de múltiples capas con una jerarquía de padre-hijo(Observe Figura 2.5).

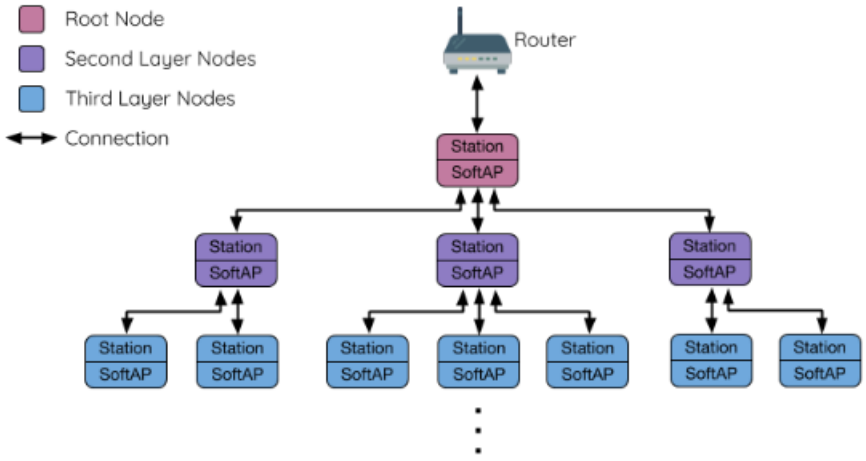


Figura 2.5: Topología de árbol de ESP-IDF (Systems, 2016a)

Tramas de faro y límite de RSSI

Cada nodo que pueda formar conexiones *downstream* (desde el nodo a hijos) transmite periódicamente una trama de faro, para comunicar su presencia, estado o para formar nuevas conexiones.

La intensidad de señal de una potencial conexión *upstream*(desde algún hijo a un padre) esta representada por RSSI (Indicación de la Intensidad de Señal Recibida) en la trama de faro. Esta se una para prevenir que los nodos formen enlaces débiles.

La selección del nodo padre

Cuando un nodo posee varios candidatos de nodos padre la selección se lleva a cabo tomando en cuenta en cual capa se encuentran y la cantidad de conexiones *downstream* que posee cada uno de los candidatos; con prioridad en el que este en

una capa más baja, esto se hace para minimizar el número de capas que posee la red.

Las tablas de enrutamiento

Cada nodo dentro de una red ESP-MESH mantiene su tabla enrutamiento individual esta contiene de las direcciones MAC de todos los nodos en la subred del nodo particular (incluyendola dirección MAC del nodo en cuestión). Cada tabla de enrutamiento es particionada internamente en las subtablas de enrutamiento de sus hijos. Las tablas de enrutamiento determinan si los paquetes deben ser reenviados *upstream* o *downstream*, basados en las siguientes reglas:

1. Si la dirección MAC del nodo destino se encuentra en la tabla de enrutamiento y no es el nodo en cuestión, entonces reenvía los paquetes *downstream* al el hijo correspondiente en la tabla de enrutamiento.

2. Si la dirección MAC destino o esta en la tabla de enrutamiento, reenvia los paquetes *upstream* al correspondiente nodo padre. De esta manera el mensaje terminaría en el nodo raíz, cuya tabla de enrutamiento debe contener todos los nodos de la red.

2.4. Protocolo MODBUS

Modbus es un protocolo de mensajes de capa de aplicación, posicionada en el nivel 7 del modelo OSI, el cual provee comunicación cliente/servidor entre dispositivos conectados en diferentes tipos de buses o redes. (Modbus Organization, 2006)

El protocolo Modbus define la unidad de datos de protocolo (PDU) independiente de las capas inferiores de comunicación. El mapeo del protocolo MODBUS en buses o redes específicas pueden introducir campos adiciones en la unidad de datos de aplicación (ADU).

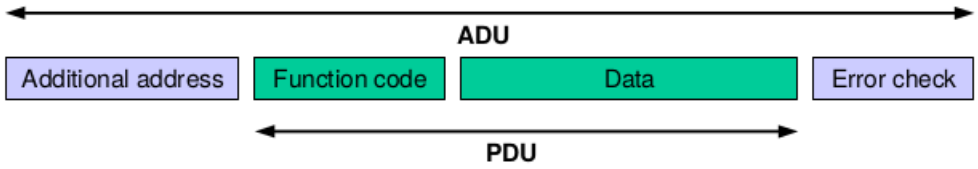


Figura 2.6: Trama general de procolo MODBUS

La ADU es construida por el cliente que inicializa la transacción Modbus, con una forma específica defina por el protocolo. El protocolo posee códigos en las PDU, llamados códigos de funciones, que son los elementos a través de los cuales el protocolo ofrece diferentes servicios, es decir, la función indica al servidor que tipo de acción llevar a cabo.

El código de función ocupa un byte,con valores válidos entre 1 y 255 (el rango de 128 - 255 está reservado para respuesta de excepciones). Además, se pueden agregar códigos de sub-funciones que contienen información adicional que el servidor usa para ejecutar la acción definida por el código de la función, como direcciones de registros, cantidad de ítems y número de bytes en un campo. El campo del código de la función se utiliza también para indicar si hubo una respuesta normal (sin errores) o si hubo errores (repuesta de excepción). Para respuesta normales, el servidor simplemente responde con un eco del código de función en la respuesta, mientras que para repuestas de excepción este campo posee el código asociado.

El tamaño de la ADU está limitado en una línea serial a 256 bytes, por lo tanto, si se le resta un byte para la dirección del servidor y dos bytes de chequeo

de errores, se tiene que el tamaño de la PDU es de 253 bytes.

El protocolo Modbus defines tres tipos de PDU:

1. PDU de solicitud Modbus: Se compone de un byte del código de función, más n bytes que contiene información adicional de los datos solicitados, como desplazamientos, códigos de sub-funciones, etc.

2. PDU de respuesta Modbus: También posee un byte del código de la función más n bytes de información de respuesta asociada a la función ejecutada.

3. PDU de excepción Modbus: Un byte del código de la función de excepción y otro byte del código de excepción.

2.4.1. Modelo de datos Modbus

Modbus basa su modelo de datos en una serie de tablas que tienen una características que las distinguen. Las cuatro tablas primarias son :

Tablas Primarias	Tipo de objeto	Definición
Entradas discretas	Único bit	Solo leer
Bobinas	Único bit	Leer y escribir
Registros de entrada	Word de 16-bits	Solo leer
Registros de retención	Word de 16-bits	Leer y escribir

Tabla 2.1: Tablas primarias de los modelos de datos Modbus

Toda los datos manejados vía Modbus (bits y registros) deben estar localizados en la memoria de aplicación del dispositivo interrogado; la memoria física no debe ser confundida con las referencia de los datos. El único requerimiento es la vinculación de la referencia de los datos con la memoria física.

2.4.2. Protocolo Modbus sobre línea serial

El estándar del protocolo Modbus define una capa de aplicación posicionada en el nivel siete del modelo OSI. También estandariza un protocolo específico en la línea serial para el intercambio de solicitudes Modbus entre el maestro y uno o varios esclavos, que se ubica en las capas uno y dos del modelo OSI.

Un sistema del tipo maestro-esclavo tiene un nodo maestro que envía comandos a los nodos esclavos. Los nodos esclavos no transmiten datos si una solicitud y no se comunican con otros esclavos.

En nivel físico, el protocolo Modbus sobre linea serial puede usar diferentes interfaces, sin embargo, la TIA/EIA-485 (RS-485) es la más común.En la linea serial solo un maestro puede estar conectado en un dado momento y un límite de 247 esclavos.

Modos de transmisión serial

Existen dos modos transmisión: el modo RTU y el modo ANSII. Estos definen cómo la información es empaquetada en los campos del mensaje y cómo es decodificada. Por lo tanto, el modo de transmisión debe ser el mismo para todos los dispositivos Modbus en la linea serial.

Para la lograr la interoperativilidad entre los equipos Modbus, el modo RTU tiene que ser implementado y debe estar configurado por defecto; el modo ANSII es opcional.

Cuando la comunicación se ejecuta usando el modo RTU, cada byte en un mensaje contiene dos caracteres hexadecimales de 4-bits. La principal ventaja de este modo que se logra una mayor densidad de caracteres que el modo ANSII para

la misma tasa de baudios. El formato para cada byte en el modo RTU es binario de 8 bits como sigue:

1. 1 bit de inicio.
2. 8 bits de datos, se envía el menos significativo primero.
3. 1 bit de paridad.
4. 1 bit de parada.

El bit de paridad es requerido, para asegurar la máxima compatibilidad. Es recomendado soportar también el modo sin paridad. El modo por defecto de paridad es la paridad par.

Toda la trama Modbus debe ser enviada de forma continua sin superar silencios de un tiempo equivalente de 1,5 caracter entre dos caracteres, de otra manera se originaria un error de trama. El espaciado entre tramas es marcado por silencios mayores entre 3,5 caracteres.

2.5. Microcontrolador ESP32 (Systems, 2019)

El ESP-32 es un chip con integración WiFi y Bluetooth diseñado con la tecnología de ultra bajo consumo de 40 nm. Este microcontrolador esta diseñado para aplicaciones móviles, electrónicos personales, y de Internet de las cosas (IoT). Posee características de bajo consumo, incluyendo reloj de alta precisión, múltiples estados de energía, y escalamiento de consumo dinámico(Systems, 2019).

Además es una solución integrada, que ya posee WiFi, Bluetooth, junto con alrededor de 20 componentes externos. El chip incluye una interruptor de antena,

acoplador de radio-frecuencia, amplificador de potencia, amplificador de recepción de bajo ruido, filtros, y módulos de administración de consumo.

Este microcontrolador posee dos núcleos CPU que pueden ser individualmente controlados, con un reloj ajustable entre 80 MHz y 120 MHz. Además, usa el sistema operativo en tiempo real freeRTOS y tensión de alimentación nominal de 3,3 V.

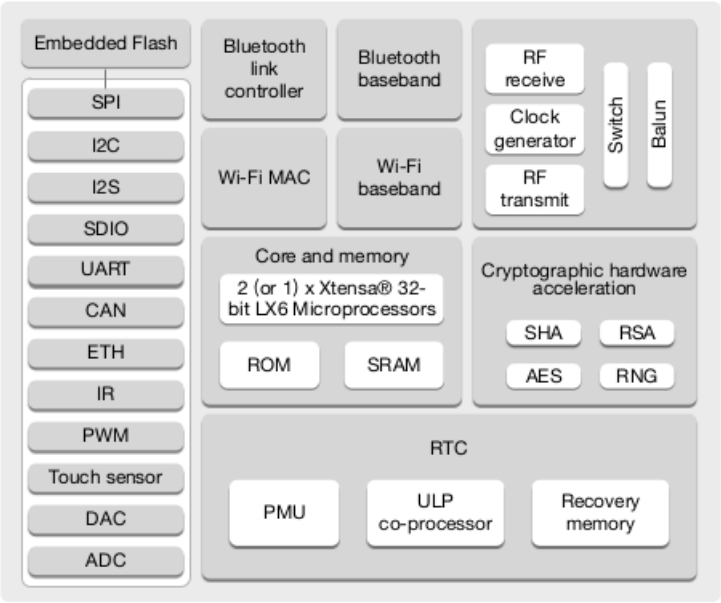


Figura 2.7: Diagrama funcional del ESP-32(Systems, 2019)

2.5.1. Características principales del WiFi

- 802.11 b/g/n.
- 802.11 n (hasta 150Mbps).
- WMM.
- TX/RX A-MPDU, RX A-MSDU.
- Bloque de ACK inmediato.

- Defragmentación.
- Monitorización de faro automático (Hardware TSF).
- 4 interfaces virtuales WiFi.
- Soporte simultaneo para estación, Punto de acceso y modo promiscuo.
- Diversidad de antena.

2.5.2. Características principales de CPU y memoria

- Doble núcleo Xtensa de 32 bits, hasta 600MIPS.
- 448 KB ROM.
- 520 KB SRAM.
- 16 KB SRAM en RTC.
- Memoria flash embebida hasta 4MB.

2.5.3. Relojes y Temporizadores

- Oscilador interno con calibración de 8MHz
- Oscilador interno RC
- Oscilador externo de cristal desde 2 a 60MHz.
- Dos grupos de temporizadores, incluyendo 2x64-bits con un perro guardián en cada uno.
- Un temporizador RTC
- Perro guardián RTC

2.5.4. Intefaces de periféricos avanzadas

- 34 GPIO mapeables.
- Convertidor analógico digital de 12-bits de hasta 18 canales.
- Dos convertidores digital analógicos.
- 10 sensores táctiles.
- 4 SPI.
- 2 I^2C.
- 3 UART.
- 1 host (SD/eMMC/SDIO).
- Interfaz de MAC Ethernet con DMA dedicado y soporte IEEE 1588.
- CAN 2.0.
- IR (TX/RX).
- Motor PWM.
- LED PWM hasta de 16 canales.
- Sensor Hall.

2.5.5. Seguridad

- Boot seguro.
- Encriptación de flash.
- 1024-bits OTP, hasta 768-bit clientes.

- Aceleración de criptografía por hardware .
    - AES.
    - Hash (SHA-2).
    - RSA.
    - ECC.
    - Generador de números aleatorio.

2.6. Sistema Operativo en Tiempo Real

Un sistema operativo es un programa de comptadora que soporta las funciones básicas de la misma, y provee servicios a otros programas o aplicaciones que corren en la computadora. Las aplicaciones ejecutan las funcionalidades que el usuario quiere o necesita. Los servicios del sistema operativo hace el desarrollo de aplicaciones rápido, simple, y mantenible.

La mayoría de los sistemas operativos permiten que se ejecuten múltiples programas al mismo tiempo. Es es llamado multi-tarea. Aunque, en realidad, cada procesador solo puede correr una tarea. Una parte del sistema operativo es llamado planificador,y es responsable de la decisión cual programa correr y cuando, así da la ilusión de la ejecución simultanea de tareas mientras conmuta rápidamente entre cada programa (Services, 2020a).

2.6.1. El FreeRTOS

FreeRTOS es una clase de RTOS que está diseñado para ser lo sufientemente liviano para correr en un microcontrolador. Este provee planificador en tiempo real, comunicación entre tareas, temporizadores y sincronización. Esto significa

que es descrito más exactamente como un kernel en tiempo real. El planificador usado en el FreeRTOS, logra determinismo, proporcionando a los usuarios asignar prioridad a cada tarea de ejecución (Services, 2020b). Cabe resaltar algunos elementos resaltantes:

- **Tareas:** Las tareas se implementan como funciones.Cada tarea es un pequeño programa en sí mismo. Tiene un punto de entrada, se ejecuta normalmente de forma continua en un bucle infinito y no se cierra.
- **Colas:** Las colas proporcionan mecanismo de comunicación de una a otra tarea, de una tarea a una interrupción y de una interrupción a una tarea. La comunicación se basa en la transferencia en un dato ubicado en la cola.

CAPÍTULO III

MARCO METODOLÓGICO

En este capítulo se describirá el diseño de la red, detalles de software y hardware de los nodos.

3.1. DISEÑO DE LA RED MALLADA

Tomando en cuenta la investigaciones realizadas se encontró que en el campo de las redes WiFi malladas no existe un protocolo de aplicación definido, en su lugar, las redes son ajustadas a las necesidades específicas del campo en cual se vaya a implementar, esto trae como consecuencia que las especificaciones se concentren en el máximo rendimiento en las áreas críticas del problema.

Dentro de los funcionamientos descritos anteriormente, se tiene que el descrito en la sección 2.3.1 se concentra en el algoritmo de optimización de la topología de la red, sin tomar en consideración el hardware de implementación. Por otro lado, en la sección 2.3.2, se provee una aproximación más realista, donde existen especificaciones del hardware, sin embargo, no hay justificación para el enrutamiento usado y las medidas de rendimiento aplicadas no son de interés para el área industrial. Por último, se tiene que la ESP-MESH parecía brindar extensibilidad y rápido desarrollo, no obstante, posee la desventaja de la necesidad de un enrutador WiFi para la formación, que brinde las direcciones IP a la red mallada, lo que deriva en un centralización, pues si el router falla, la red perece.

Por lo tanto, es conveniente poseer una arquitectura especifica de la red mallada para el funcionamiento dentro de una red Modbus . Así se puede adaptar y configurar de manera más óptima y confiable para las funciones y características del protocolo Modbus .

Para la creación de redes inalámbricas WiFi usando el microcontrolador ESP-32,*Espressif Systems* a través del *ESP-IDF* (Espacio de trabajo para desarrollo de Internet de las cosas) provee dos librerías: la ya mencionada ESP-MESH y ESP-NOW. La librería ESP-NOW fue ideada para comunicaciones punto a punto en esquemas maestro/esclavo, no para creaciones de redes multi-salto, sin embargo, posee la expansibilidad de manipular el direccionamiento de los datos, brindado la potencialidad de crear redes de diversas arquitecturas.

3.1.1. ESP-NOW

ESP-NOW es un tipo de protocolo de comunicación WiFi definido por *Espressif Systems* donde los datos de aplicación son encapsulados en una trama específica por el proveedor y es transmitida de un dispositivo WiFi a otro. Para proteger la trama de acción por seguridad se usa el protocolo CCMP (Protocolo de código de autenticación de mensaje de encadenamiento de bloque de cifrado en modo contador). ESP-NOW es ampliamente usada en luces inteligente, control remoto y sensores.

ESP-NOW permite desarrollar redes según la arquitectura definida en la capa de aplicación, por lo que es ideal para la implementación de nuevas topologías y procesos de red. Además, debido a que la comunicación es punto a punto, la red no se centraliza. Así mismo, posee una capacidad de carga aceptable para la transmisión de tramas Modbus, por lo que fue la librería utilizada para el diseño e implementación de la red.

3.1.2. Requerimientos de diseño

Principalmente debe ser una red confiable, estable, y eficaz; aprovechando las principales características de las redes malladas, como saltos múltiples y descentralización. La nodos deben soportar al menos la configuración básica descrita por Modbus para linea serial garantizando la interoperatividad con otros equipos Modbus. Los nodos deben ser configurables, pudiendo ajustar las rutas de la red y la tasa de baudios de la interfaz serial, para así, obtener una red extensible.

3.1.3. Diseño Básico de la red

Un nodo es una estación de comunicación con una dirección MAC bajo el estándar 802.11 y una implementación de capa física (En este caso el ESP-32) para la comunicación bajo este protocolo, además, constituye la entidad básica de la red. La red más elemental, llamada conjunto de servicios básicos (BSS), puede ser formada por dos nodos. Para que otros nodos se unan, los integrantes del BSS provee de servicios a través de un sistema de distribución (DS) que permiten la comunicación con con dispositivos fuera del BSS para su integración. Los enlaces se forman a partir del nodo que genere la trama (salto inicial) e indique el nodo siguiente (nodos intermedios) hasta llegar al nodo final (salto final).Los nodos involucrados en el proceso forman un BSS que permanece establecido.

Estructura de la trama

Actualmente, el estándar 802.11 clasifica las tramas como de datos, de control, o de gestión. La trama de datos transmite la información en las capas más altas. Las tramas de control son usadas para agradecimientos y reservaciones. Finalmente, los dispositivos usan las tramas de gestión para configurar, organizar

y mantener el enlace local.

ESP-NOW usa una trama de acción (Trama de control) específica de proveedor para transmitir datos. La tasa de transmisión por defecto es de 1 Mbps y formato de la trama es como se muestra en las tablas 3.1 y 3.2.

Tabla 3.1: Trama de acción ESP-NOW

Dirección MAC	Código de categoría	Identificador de organización	Valores aleatorios	Contenido de provedor	FCS
24 bytes	1 byte	3 bytes	4 bytes	7-255 bytes	4 bytes

1. Código de categoría: es establecido en 127 para indicar la trama específica de proveedor.
2. Identificador de organización: contiene un identificador único (0x18fe34), los cuales son las primeros tres bytes de la MAC aplicada por *Espressif*.
3. Valores aleatorios: son usador para prevenir ataques de retransmisión.
4. Contenido de proveedor: Contenido de específico de proveedor, contiene datos adicionales.
5. FCS: Trama de chequeo de errores.

Tabla 3.2: Contenido de específico de proveedor

ID de elemento	Longitud	ID de organización	Tipo	Versión	Cuerpo
1 byte	1 byte	3 bytes	1 byte	1 bytes	9 - 250 bytes

1. ID de elemento: Es establecido en 221, para indicar elemento de específico de proveedor.
2. Longitud: Es la longitud total de ID de organización, tipo, version y cuerpo.
3. Tipo: Es establecido en 4, para indicar ESP-NOW .
4. Versión: Versión de ESP-NOW .

5. Cuerpo: El cuerpo contiene los datas ESP-NOW .

En el protocolo ESP-NOW la cabecera MAC varia respecto a las convencionales, el primer campo es dedicado a la dirección destino, la segunda la dirección de origen y la tercera es usada para establecer la dirección broadcast (0xFF:0xFF:0xFF:0xFF:0xFF:0xFF).En las tramas ESP-NOW los bits de control de tramas *FromDS* y *ToDS* son ambos cero.

Los campos de gestión que indican las direcciones MAC son usados para mantener la red, y el servicio de distribución para ingresar nuevos nodos, se logra a partir de la dirección broadcast.

Los campos de cuerpo de la trama también son usados para manejar la comunicación en la red mallada en la capa de aplicación, cada nodo posee una lógica dependiente de la tramas recibidas. La estructura de datos del cuerpo toma la forma definida en la tabla 3.3.

Tabla 3.3: Estructura de datos envidada en el cuerpo de la trama ESP-NOW

type	Broadcast o unicast
state	Indica si se han recibido datos broadcast
seq_num	Numero de secuencia
crc	Crc de los datos
dir	Dirección desde o hacia el maestro
Nodeid	Identificador del nodo
payload	Carga
data_len	Longitud de los datos

Gestión y formación mallada

Una señal broadcast proveniente de un nodo se usa para detectar la red y obtener información sobre su configuración. Los nodos se detectan entre sí en un escaneo activo. La trama de escaneo se identifica debido a que es generada me-

diante broadcast. La información recibida posee tres campos claves: información del identificador del nodo, si se ha recibido o no información broadcast antes del nodo por el que se está buscando y un campo adicional Modbus. El escaneo se realiza cuando un nodo no posee la dirección MAC de otro nodo en su tabla de enrutamiento y se requiere enviar datos a este. La fuente de datos posee el esclavo Modbus destino, lo que es usado para registrar solo aquellos nodos que tengan relación con el esclavo particular. Una vez los nodos se han registrado unos a otros la relación permanece, incluso si algún nodo es apagado, una vez sea se reinicie la red permite una rápida reconexión.

Los nodos solo usan un transceptor simple, por lo tanto, la red mallada opera en un solo canal de frecuencia WiFi. Esto indica que se puede colocar una red en cada canal WiFi en un mismo espacio físico, sin llegar a tener conflictos de este tipo.

Seguridad

Cada nodo ESP-NOW usa el método CCMP, el cual está descrito en el estándar 802.11- 2012, para proteger la trama de acción específica del proveedor. Esto junto con un algoritmo de cifrado mutuo, los nodos emparejados usan una llave maestra primaria (PMK) que es usada para cifrar las llaves de cifrado locales (LMK) con el algoritmo AES-128. Este método de permite la autenticación entre dos nodos emparejados. Debido a que es un cifrado por emparejamiento, cada vínculo es independientemente asegurado, como consecuencia, ESP-NOW no provee cifrado fin-a-fin.

El cifrado para comunicación broadcast no esta soportado. Las llaves locales son usadas por el método CCMP y cada nodo puede contener hasta seis llaves diferentes, si la llave local no se establece la trama de acción permanece sin cifrado.

Selección de rutas

Las rutas no son determinadas por la red, si no por el usuario que la implemente. Además, son estáticas y no se reasignan automáticamente, estas rutas permanecen en una tabla unidimensional, con los identificadores de los nodos destinos. El enrutamiento está estrechamente relacionado con los esclavos Modbus, y están representados por las posiciones en la tabla de enrutamiento.

La red no propaga información de las rutas, por lo que cada nodo solo conoce la ruta para una salto en cualquier sentido. Sin embargo, es posible solicitar y configurar las rutas de los nodos inalanbricamente.

La tabla de enrutamiento posee 256 casillas (una para cada esclavo Modbus), donde la posición está asociada al esclavo Modbus y el valor en la casilla se relaciona con la ubicación de dicho esclavo. Cabe resaltar que el identificador al que se refiere la tabla de enrutamiento como destino no es la localización del esclavo en la red. En la figura 3.1 se ilustra el la arquitectura e interpretación de la tabla de enrutamiento asociada al nodo 255.

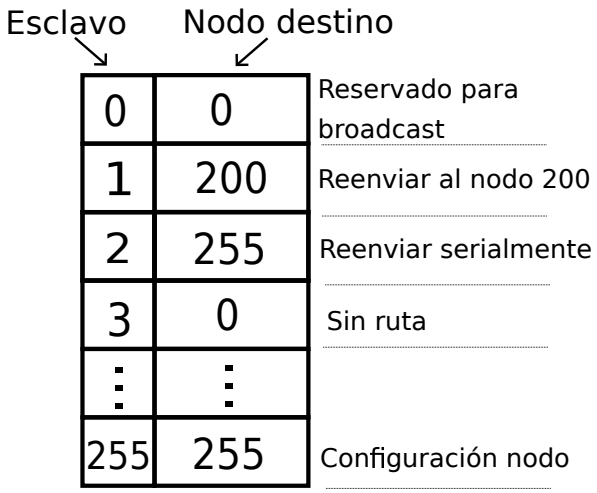


Figura 3.1: Ilustración de la estructuras de la tabla de enrutamiento

Existe otra tabla de enrutamiento para las respuestas, de manera tal que las rutas sean bidireccionales y únicas. La tabla de respuesta es configurada y mantenida automáticamente por cada nodo. Cuando se reciben solicitudes Modbus, se guarda el identificador del nodo del cual provino dicha información, y dicho identificador se usa para enrutar la trama respuesta.

Un esquema del funcionamiento se puede apreciar en las figuras 3.2 y 3.3.

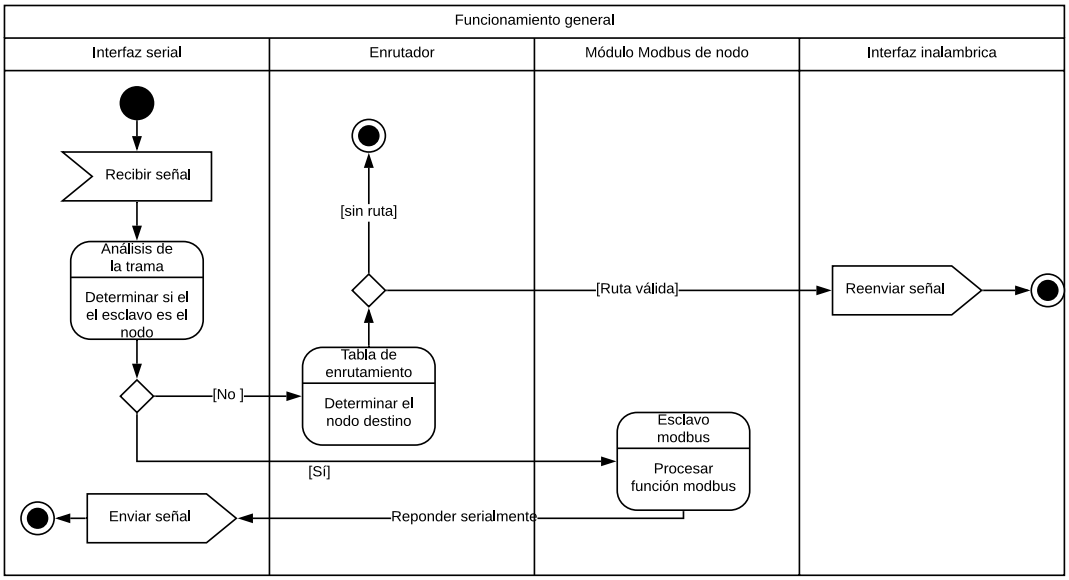


Figura 3.2: Diagrama de flujo del funcionamiento general de un nodo de la red recibiendo información WiFi

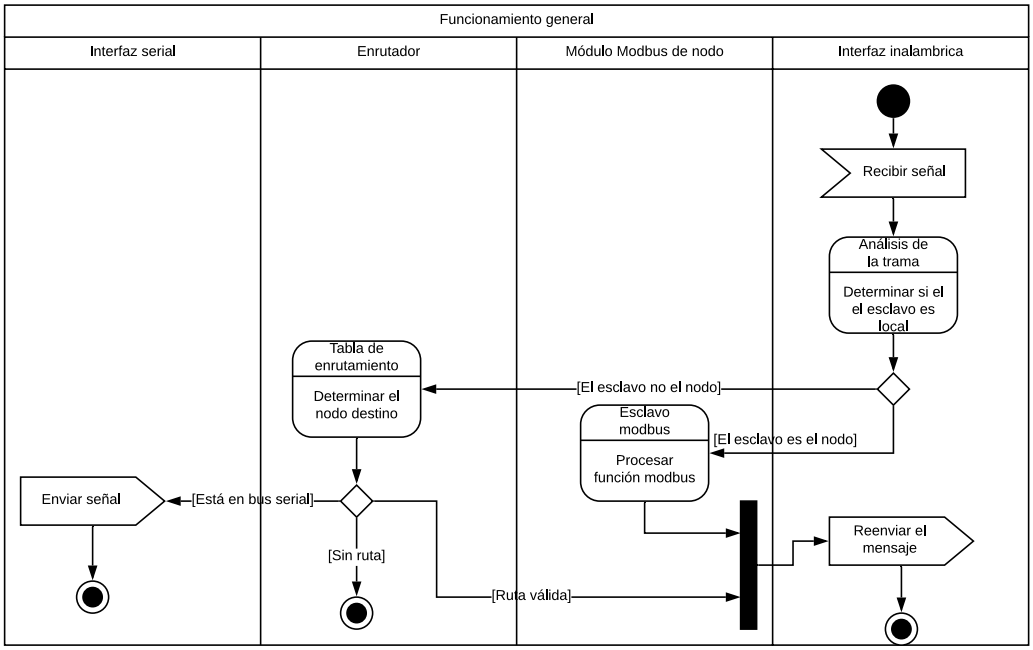


Figura 3.3: Diagrama de flujo del funcionamiento general de un nodo de la red recibiendo información serial

Características Modbus de los nodos

Los nodos de la red son esclavos adicionales de la red Modbus, con direcciones reservadas para ellos desde el identificador (101). El modo de operación Modbus en los nodos permite que se puedan obtener y configurar parámetros de la red, como la tabla de enrutamiento de los nodos. Además el proceso es realizado mediante un protocolo estandarizado y que esta relacionado con la red.

El funcionamiento como esclavo Modbus en los nodos es permanente, incluso cuando ya se ha implementado la red con equipos Modbus seriales externos, permitiendo actualización de parámetro de la red en cualquier momento.

Los nodos poseen un diagnóstico visual que indica cuando se ha recibido o enviado correctamente una trama serial o inalambricamente. Dicho diagnóstico de

basa en un LED que emite un pulso. Otro diagnóstico visual para indicar que se ha encendido el equipo, del mismo tipo se encuentra en la tarjeta de alimentación. No hay diagnostico visual para errores de comunicación.

Los registros disponibles para acceso son los registros de retención desde la dirección cero hasta la 512, y también las diez primeras bobinas. Para las bobinas solo es soportada la función de escribir una bobina simple. La tabla 3.4 describe la asignación de los registros.

Tabla 3.4: Tablas primarias de los modelos de datos Modbus

Dirección	Tipo de registro	Descripción
01	Registros de retención	Identificador
02	Registros de retención	Tasa de baudios
256-512	Registros de retención	Tabla de enrutamiento
0	Bobina	Reinicio
1	Bobina	Guardar valores Modbus para ejecución
2	Bobina	Guardar valores Modbus en la memoria Flash

La tabla 3.5 posee una descripción de la parámetros Modbus.

Tabla 3.5: Parámtros Modbus

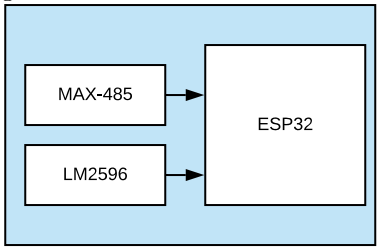
Dirección	Por defecto	Opciones
ID Modbus	255	101 a 255
Tasa de baudios serial	19200	300 a 115200
Tabla enrutamiento	Nula	Valores válidos de ID Modbus
Modo	RTU	Ninguna
Paridad	Ninguna	Ninguna
Bits de parada	1	Ninguna

Cabe resaltar que estos nodos pueden ser interrogados tanto serial como inalambricamente, es decir, con la red ya instalada. Las funciones no usadas proveerán la excepción Modbus "Función ilegal".

El hardware

El hardware del nodo se puede dividir en tres bloques: el microcontrolador, la alimentación y la interfaz serial. Esto se puede ilustrar en el esquema de la figura 3.4.

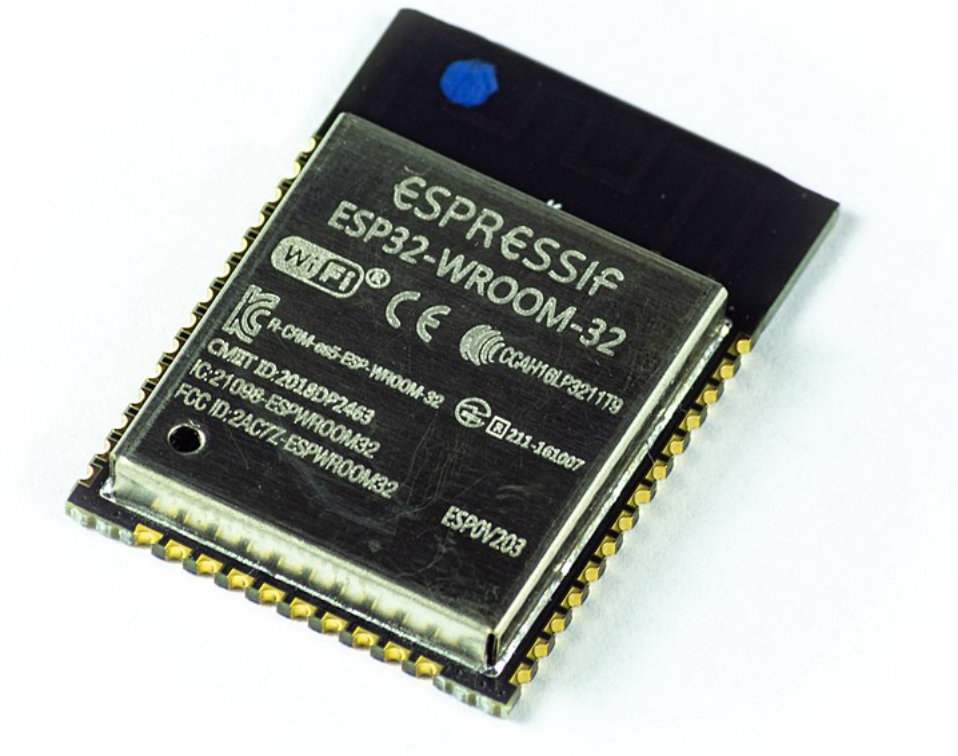
Figura 3.4: Diagrama de bloques para el hardware del nodo



Microcontrolador

Para la aplicación se usó el encapsulado ESP-WROOM-32, debido a disponibilidad y encapsulado con escudo WiFi. Se debe tener en cuenta que se dejaron disponibles los pines del UART 0 para la descarga del software, además de dos pines adicionales para inducir el estado de programación, los cuales son el pin *EN* y el pin de propósito general cero.

Figura 3.5: Encapsulado ESP-WROOM-32 del ESP32



Alimentación La alimentación del microcontrolador es de 3,3 V y además los equipos Modbus especifican un rango de tensión de funcionamiento entre 5 V y 25 V. Por lo tanto se escogió el regulador conmutado LM2596S que brinda estas características y además existe una tarjeta pre-fabricada para rápida implementación. La figura 3.1.3 muestra la forma de dicha tarjeta. Además posee un LED testigo de su encendido, que es usado como señal de diagnóstico Modbus.

Tabla 3.6: Especificaciones de la fuente de alimentación

Voltaje de entrada	4 V - 35 V
Voltajes de salida	1,23 V 30 V
Corriente de entrada	3 A (Máxima)
Eficiencia de conversión	92 % (Máxima)
Frecuencia de conmutación	150 kHz
Ripple de salida	30 mA (Máximo)

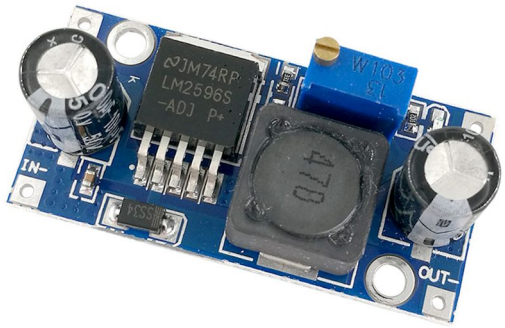


Figura 3.6: Tarjeta con regulador DC-DC para alimentación

Las características eléctricas son presentadas en la tabla 3.6

Interfaz serial

Modbus en sus clases básica dicta que los equipos Modbus seriales tienen que poseer la interfaz eléctrica RS-485 de dos cables de acuerdo con el estándar EIA/TIA-485. Por esto se seleccionó para la implementación en el nodo, asegurando la máxima compatibilidad.

El chip MAX3485 de *Maxim Integrated* provee de las características antes descritas y admite tensión de alimentación de 3,3 V, lo cual hace que se posea un solo nivel de tensión para la alimentación del nodo.

Este chip soporta hasta 32 dispositivos,sin embargo, el uso de nodos distribuye la carga de los esclavos y la utilización de repetidores se vería disminuida, y se soportarían 32 dispositivos por cada nodo.

La topología RS-485 Modbus sin repetidor posee cables truncados en un par trenzado, sobre los cuales los dispositivos están conectados bien sea directamente o a través de cables de derivación. Los cables truncados también llamado bus, cuya longitud depende de la tasa de baudios, los parámetros del cable y la carga de dispositivos, para 9600 baudios y un cable calibre de 26 AWG, la máxima longitud es de 1000 m.

Las derivaciones no deben estar a más de 20 m. En caso del uso de una derivación de puertos múltiples con n derivaciones, cada una tiene que respetar una distancia máxima de 40 m dividida por n.

El circuito común de cada bus debe estar conectado a tierra, preferiblemente a un solo punto para todo el bus. Esto no aplica para la red, pues evidentemente la comunicación inalámbrica brinda aislamiento galvánico.

Para minimizar las reflexiones del final del cable RS-485, se requiere colocar un terminador de linea cerca del final de cada punto del bus (Resistencia de 150 Ω y 0,5 W). No se deben colocar más de dos terminadores de linea en el bus, ni colocar terminadores de linea en los cable de derivación.

Si el sistema requiere polarización esta de debe implementar en cada bus serial del sistema, los nodos inalámbricos no requieren polarización.

La interfaz mecánica consta de de Bloques terminales, la identificación RS-485 se encuentra en la el circuito impreso.

Los cables de la linea serial deben estar blindados, deben formar una par

balanceado y deben poseer un tercer cable (Común). Estos deben posee el calibre suficiente para permitir la máxima distancia. Un calibre de 24 AWG es siempre suficiente. Además, preferiblemente se debe seleccionar cables con una impedancia características mayor a 100 Ω, especialmente para tasas de Baudios mayores a 19200.

3.2. Detalle del Diseño

3.2.1. Programación del ESP-32

El fabricante del ESP-32 *Espressif Systems* especifica los recursos básicos de hardware y software relacionados con el desarrollo de aplicaciones que usen la serie de hardware del ESP32.

El hardware necesario son: una tarjeta de desarrollo con un ESP32, un cable USB y computador que posea Windows, Linux o MACOS. Para el desarrollo del programa se usaron tarjetas de desarrollo ESP32-DevKitC V1 y un computador con sistema operativo Ubuntu 18.04 LTS.

En cuanto al software se puede representar por capas:

1. Un **Toolchain** para compilar el código para el ESP32: Xtensa-ELF 1.22.0.
2. **Herramientas de construcción** para construir toda la aplicación para el microcontrolador, en este caso se usó *GNU Make* 4.1.
3. ESP-IDF que contiene esencialmente las librerias y los scripts para operar con el **Toolchain**, versión 4.0.
4. **Editor de texto** para escribir los programas en C, en este caso *Eclipse IDE* 2019-12.

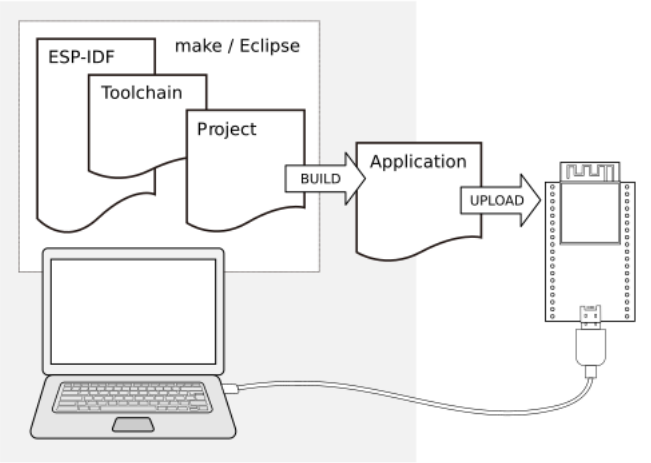


Figura 3.7: Ilustración del paquete para el desarrollo de aplicaciones en el ESP32

Los detalles de la instalación de cada uno de estas herramientas se llevo a cabo siguiendo la metodología propuesta por *Espressif* en el su pagina web `https://docs.espressif.com/projects/esp-idf/en/latest/esp32/get-started/index.html`.

El sistema operativo en tiempo real

El microcontrolador ESP32 está diseñado para correr un sistema operativo en tiempo real(RTOS) basado en el *FreeRTOS* desarrollado por *Amazon*..La lógica funcional del método de programación RTOS es mediante establecimiento de tareas e interrupciones controladas por orden de prioridad y por el temporizador principal del sistema.

Dicho sistema operativo permite que se creen y desarrollen programas en lenguaje ANSI C, implementado las librerías para el manejo del microcontrolador proveídas por ESP-IDF de *espressif*. Por lo tanto, el desarrollo del programa se basó en la creación de tareas y la comunicación entre ellas basada en colas. Se

crearon tareas dos tareas para el manejo de los datos inalámbricos, una para la recepción de datos mediante el UART y otras tareas con funciones de monitorización.

3.2.2. Programa de la red diseñada que soporte la transmisión del protocolo Modbus

El rol de cada nodo en la red es determinado por la secuencia de estados durante operación, por lo que el programa es idéntico para todos los microcontroladores, incluido el que posee conectado el maestro. Cualquier nodo puede enviar datos, incluso aquellos que posee esclavos Modbus en su interfaz serial.

En primer lugar se definen en la aplicación principal (*app_main*) las funciones que se llevaran a cabo al iniciar el microcontrolador. La inicialización de la flash, el módulo WiFi, el UART y la creación de tareas toma lugar en esta parte. Esta sección se ejecuta solo una vez.

Librerías

En el programa se incluyen la librerías estándar de C <stdlib.h>, <time.h>, <string.h>, y <assert.h>, para el uso y manejo de las principales funciones de C. Luego tenemos que freertos/file.h provee del las funciones necesarias parar la ejecución y uso del sistema operativo en tiempo real. Restando las inclusiones asociadas al ESP-IDF para el uso de las librerías de funciones del microcontrolador.

La librería nvs_flash.h, habilita para el uso de la funciones de acceso a la flash, necesarias para guardar y cargar la configuración a los nodos.

Las inclusiones esp_log.h, esp_system.h, esp32/rom/ets_sys.h, esp32/rom/ets_sys.h

y esp32/rom/crc.h permiten el uso del sistema del ESP-32 para el acceso a estado del sistema como estado de la pila, reinicio, etc.

Finalmente, para el uso de las funciones para la construcción de la red mallada son necesarias las inclusiones siguientes: tcpip_adapter.h, esp_wifi.h, esp_now.h, espdefine.h

Inicialización

La rutina para las tareas de la red mallada comienzan con la inicialización del periferico WiFi y funciones ESP-NOW mediante las funciones *wifi_ init()* y *espnow_ init()* respectivamente.

Con la función *wifi_ init()* se inicializan el adaptador TCP/IP, el manejador de eventos WiFi y se carga la confiuración, en este caso como sigue:

1. Modo de punto de acceso
2. Canal cero
3. Modo de largo alcance
4. Soporte de los protocolos 802.11b,802.11g y 802.11n.

Luego tenemos la función *espnow_ init()* que prepara las variables iniciales, crea las colas de recepción y envío, registra las funciones de *callback* de recepción y envío, establece las claves de encriptación, registra los nodos y crea la tareas.

Creación de colas (*Queue*)

En la inicialización se crean tres colas:

1. *espnow_ queue* que se usa para transmitir los datos de las funciones de *callback* a las tareas que manejan la red.

2. *espnow_ Squeue* se usa para la transmisión de los datos desde la tarea asociada al UART.

3. *espnow_ Rqueue* se usa para transmitir los datos hacia la tarea que envía información inalambricamente.

Las funciones de callback

Estas son funciones escritas por el programador que son llamadas dentro del pila WiFi automáticamente, y permite que la información del proceso de enviar y recibir se pueda manejar desde otras funciones o tareas más complejas. El fabricante recomienda que se mantengan cortas dichas funciones, que en su lugar se use una cola y se maneje la información desde otras funciones.

Estas funciones son activadas cuando se envía o recibe información mediante ESP-NOW , proveen información de la MAC, bandera de agradecimiento, longitud de los datos y los datos. Básicamente vinculan la comunicación WiFi a la aplicación principal.

Registro de puntos

En la librería *espnow* existen unos entes llamados *Peers* (Puntos), los cuales conservan la información de emparejamiento. Si un peer no está registrado, es imposible enviar información a el o recibirla. Hay un peer virtual para enviar información brodacast cuya característica es la dirección MAC, la cual es la mayor posible (FF:FF:FF:FF:FF:FF).

Los campos de cada puntos son los descritos en la tabla 3.7

Tabla 3.7: Tablas primarias de los modelos de datos Modbus

Parámetro	Descripción
peer address	Dirección MAC
lmk	Clave maestra local que es usada para cifrar los datos
channel	Canal WiFi
ifidx	Interface que usa el WiFi para enviar y recibir datos
encrypt	Habilitador de cifrado
*priv	Datos privados de la librería

La información de los puntos es guardada mediante la función *esp_now_add_peer()*.

Estructuras de datos relevantes

Se crearon ciertas estructuras usando el lenguaje C, para manejar los datos con mayor facilidad y orden.

1. *espnow_send_param_t*: esta estructura esta diseñada para contener los datos en la transmisión de los datos. Indica si la trama es unicast o bradcast. La longitud de los datos, los datos a enviar, el número de secuencia, el estado de la transmisión, etc.

2. *espnow_event_t* : es la estrutura usada para manejar los estados nodo. Permite que posea información relacionada con los datos trasmitidos, la dirección y el estado.

3. *esp_uart_data_t*: Esta estructura de datos se creó para comunicar las tareas del UART y las de ESP-NOW, no solo posee los datos recibidos si no también el sentido en la red de estos.

3.2.3. Tarea esp_now_manage_task

Es la tarea principal del sistema de comunicación inalámbrica, y es administrada por la cola *espnow_queue*. Cabe mencionar que las colas permiten que la tarea se ejecute solo si se recibió un dato en ella, lo cual ayuda mantener la eficiencia del procesador.

En esta tarea se verifica si se estaba enviando o recibiendo datos. Y existen ciertos procesos que se ejecutan en cada caso, considerando si es broadcast o unicast.

La cola se habilita por las funciones de *callback* y por la tarea del UART.

A continuación se presenta pseudocódigo de la ejecución de la tarea.

Algoritmo 1: Algoritmo tarea esp_now_manage_task

```
Data: ADU MODBUS, EVENT_DATA
Result: ADU MODBUS ADMINISTRADA
Inicialización;
while Se recibe algo en la cola do
    switch Event ID do
        case Desde enviar do
            switch Origen do
                case De la función de Callback do
                    if desMAC == broadMAC then
                        if contador ==0 then
                            break
                        contador−;
                        Enviar broadcast;
                    else
                        if estado de envío == exito or intentsos == 5 then
                            break;
                        else
                            Intentos ++
                            Enviar a la cola espnow_Squeue;
                case De la tarea UART do
                    Enviar a la cola espnow_Squeue;
            case Desde recibir do
                if Broadcast then
                    if Nodo no está registrado then
                        Registrar nodo;
                else
                    if Unicast then
                        UnicastAlgo();
                    else
                        Error
```

[H]

Algoritmo 2: Algoritmo para datos recibidos unicast

```
Data: ADU MODBUS, EVENT_DATA
Result: ADU MODBUS Direcionada
Inicialización;
if Dirección hacia adelante then
    /* Guardar el esclavo del cual provino                    */
    TablaEnrutamiento[Esclavo+offset] = NodoIDRecivido;
    /* Crear una ruta para el nodo                            */
    TablaEnrutamiento[NodoIDRecivido] = NodoIDRecivido;
switch Dirección do
    case Hacia adelante do
        switch Modo do
            case Serial do
                Envia datos serialmente;
            case Configuración Nodo do
                Usa la trama para configuración propia;
            case Salto do
                Reenvia a la tarea espnow_send();
    case Hacia atrás do
        desMAC = RoutingTable[esclavo+offset];
        if desMAC == broadMAC then
            /* Se trata una respuesta en el maestro           */
            Enviar a la interfaz serial;
```

El *offset* en la tabla de enrutamiento representa a la posición a partir de la cual las rutas hacia atrás son automáticamente llenadas, para que la ruta de información sea la misma hacia adelante que hacia atrás.

Cuando se envía información se entra a la primera *SEND_CB*, si es unicast se realizan cinco intentos de envío o hasta sea recibida la trama por el nodo destino. Solo cuando se inicializa en nodo se envían tramas broadcast para indicar a los nodos vecinos su presencia.

En la sección de recepción tenemos que cuando se recibe broadcast se registra

el nodo si no esta registrado y responde con un broadcast. Si se recibe unicast entonces se pasa a una máquina de estados que determina el destino de la información.

3.2.4. Tarea espnow_send

Esta tarea se encarga de preparar, enrutar y enviar los datos unicast. Es activada mediante la cola *espnow_Squeue*. El algoritmo bajo el cual se rige es el 3.

Algoritmo 3: Algoritmo de enrutamiento de los datos en la tarea *espnow_send()*

```
/* idNode: ID nodo actual                                    */
/* direcciónDe: Dirección recibida en la data                */
/* dataRecibida: Carga de de datos                           */
Data: idNode, direcciónDe, dataRecivida
Result: Data enviada al nodo respectivo
while Se recibe información en la cola espnow_queue do
    desNode = TabladeEnrutamiento[Esclavo];
    direccionA = hacia adelante;
    if desNode == idNode or direcciónDe == hacia atrás then
        desNode == TablaEnrutamiento[offset + Esclavo];
        direcciónA = hacia atrás;
    dataAEnviar = dataRecivida;
    espnow_send(dataAEnviar, desNode ,direcciónA);
```

Como se observa en el algoritmo 3, se toma en cuenta si los datos van hacia adelante o hacia atrás, para elegir la sección de la parte de enrutamiento a usar.

Direcciones MAC

Como se dijo anteriormente, la libreria ESP-NOW usa las direcciones MAC para direccionar los datos. Ya que la tabla de enrutamiento considera solo identificadores referenciales, es necesaria la obtención de las MAC por otro método.

Cuando se envía datos broadcast las MAC son registradas en todos los dos que alcance la señal, y dicho identificador se encuentre en la tabla de enrutamiento respectiva (Observe algoritmo 1). Al mismo tiempo se guarda la dirección MAC respectiva en un arreglo de $(255\,x\,6)$ casillas, es decir, la cantidad de nodos por los campos que ocupa una dirección MAC.

Para acceder a las posiciones solo basta usar el identificador del nodo del que se desee obtener la dirección MAC, multiplicarla por seis y usar el número resultante como la posición a partir de la cual se tomarán seis campos.

3.2.5. Programa para el manejo del protocolo Modbus en el bus RS-485.

El ESP-IDF provee un manejador de eventos para el UART, así como estructuras de datos diseñadas para su configuración. Por lo tanto, se configuró el UART 1 para la comunicación RS-485 *half-duplex*. El UART 0 esta reservado para la quema del microcontrolador y notificaciones de información y error.

El manejador de eventos proporciona una arquitectura de máquina de estados donde se puede aceptar los datos si estos se reciben correctamente. Una vez esto pasa se crea una máquina de estados adicional para determinar el destino de la trama, que es básicamente hacia el nodo en cuestión o para un nodo externo.

En el algoritmo 4 describe el funcionamiento de la rutina de la tarea *rx_ task()*.

Algoritmo 4: Algoritmo de la tarea que administra el UART

```
Data: ADU MODBUS serial
Result: Datos direccionada
UARTinit();
while Se recibe información en la cola uart1_queue do
    Limpia el buffer;
    switch Tipo de evento do
        case Datos recibidos do
            tramaModbus = recibirDatos();
            dirección = obtenerDireccion(tramaModbus);
            switch dirección do
                case Nodo externo do
                    eventoID = desdeUART; colaEnviar(espnow_queue,
                     eventoID, tramaModbus);
                case Configuración Nodo do
                    configurarNodo(tramaModbus);
            Limpiar buffer;
            Resetear Cola;
        case Excepciones UART do
            Limpiar buffer;
            Resetear Cola;
```

La función *UARTinit()* se encarga de incializar el uart con la configuración adecuada.

Librerías

Para el uso del UART y las estructuras de datos relevante asociadas de usan las librerias de ESP-IDF *driver/uart.h*, *uart_func.h* y *soc/uart_struct.h*.

3.2.6. Funciones auxiliares

Función *vNotiLEDinit*

Tabla 3.8: Descripción de la función *vNotiLEDinit*

Nombre	vNotiLEDinit
Descripción	Parpadeo de encendido
Parámetro	Vacío
Resultado	Crea que en un pin 3 parpadeo de 1s

Ejecuta un parpadeo en el GPIO 2 del microcotrolador al que se le conectó un LED para indicar el que el nodo se ha encendido correctamente.

Función *FormatFactory*

Tabla 3.9: Descripción de la función *FormatFactory*

Nombre	FormatFactory
Descripción	Función de la tarea de reseteo de fábrica
Parámetro	Vacío
Resultado	Limpia la configuración del nodo

Esta tarea recibe una activación mediante un semáforo binario proveniente de una interrupción por hardware en el GPIO 0, también usado como $BOOT$; en caso de que se desee formatear el nodo a sus parámetros por defecto. Se debe mantener presionado por al menos cinco segundos.

Función *vConfigLoad*

Tabla 3.10: Descripción de la función *vConfigLoad*

Nombre	vConfigLoad
Descripción	Función para cargar la configuración al nodo
Parámetro	Vacío
Resultado	Carga datos desde la flash a la RAM

Esta función carga los parámetros de funcionamiento e identificación del nodo, como lo son las tablas de enrutamiento, las direcciones MAC, el identificador del nodo y la configuración del UART. Se ejecuta al energizar el nodo.

Función *vConfigGetNVS*

Tabla 3.11: Descripción de la función *vConfigGetNVS*

Nombre	vConfigGetNVS
Descripción	Función para obtener datos en la flash
Parámetros	Arreglo, String
Resultado	Carga información asociada al String en el Arreglo

Se usa para obtener la configuración desde flash, y solo funciona para los registros de retención, tabla de MAC's y tabla de enrutamiento. Esta información de carga el parámetro del arreglo.

Análogamente con la función *vConfigSetNVS* que en ese caso carga información en la flash desde la RAM.

Función *vConfigSetNode*

Tabla 3.12: Descripción de la función *vConfigGetNVS*

Nombre	vConfigSetNode
Descripción	Función para procesar tramas Modbus
Parámetros	Trama Modbus, origen (serial o inalambrico)
Resultado	Respuesta Modbus

Es la función asociada al Modbus de cada nodo, puede ejecutar las funciones de lectura y escritura de los registros de retención y escritura de la bobinas. Posee la capacidad de generar la respuesta inalambricamente si la solicitud fue inalámbrica o serial si la solicitud fue serial.

Función *uComDirection*

Tabla 3.13: Descripción de la función *vConfigGetNVS*

Nombre	uComDirection
Descripción	Comparar el esclavo de la trama con el identificador del nodo
Parámetros	Esclavo
Resultado	Nodo externo o Nodo en cuestión

Se usa en la tarea del UART para dirigir la información Modbus que se recibe, comparando el esclavo recibido con el identificador del nodo que recibió la trama. Posee una salida binaria: va o no la trama dirigida el nodo en cuestión.

Función *uComGetTransData*

Tabla 3.14: Descripción de la función *uComGetTransData*

Nombre	uComGetTransData
Descripción	Máquina de estados para determinar destino de la trama
Parámetros	Esclavo
Resultado	Serial, Reenvío o configuración

Es una función similar a la descrita en la sección 3.13, sin embargo, esta arroja tres posibilidades. Determina si la trama recibida debe ser enviada seriamente, reenviarse a otro nodo o simplemente configurarse a sí mismo.

Función *vEspnowGetOldPeers*

Tabla 3.15: Descripción de la función *vEspnowGetOldPeers*

Nombre	vEspnowGetOldPeers
Descripción	Registrar los nodos al iniciar
Parámetros	Vacío
Resultado	Serial, Reenvío o configuración

Junto con la función *RegisterPeer* registra MAC de los nodos al energizar el nodo. Pues el registro se hace en la memoria volátil.

Función *espnow_data_prepare*

Tabla 3.16: Descripción de la función *espnow_data_prepare*

Nombre	espnow_data_prepare
Descripción	Preparar los datos a enviar
Parámetros	variable *send_param*
Resultado	cargar datos de trama *espnow*

Esta función prepara los trama inalámbrica, que contiene el chequeo de error, el identificador de nodo, si es unicast o broadcast, la dirección, etc.

Función *espnow_ data_ parse*

Tabla 3.17: Descripción de la función *espnow_ data_ parse*

Nombre	espnow_data_parse
Descripción	Descompone los datos recibidos
Parámetros	datos, longitud de datos, estado
Resultado	cargar datos de trama *esp_ now*

Cumple la función complementaría a la función 3.2.6, descompone los datos recibidos, comprueba el chequeo de error, etc. Se usa en la máquina de estado de recepción inalámbrica.

3.2.7. Circuito de un nodo para una red WiFi mallada basada en el microcontrolador ESP32.

El desarrollo del software de la red ser realizó de forma iterativa sobre las tarjetas de desarrollo, por lo que la red fue implementada mientras era desarrollada.

Se consideran las especificaciones del diseño en la sección 3.1.3 y las recomendaciones técnicas realizadas por *Espressif* para el diseño de PCB con el ESP32.

Posicionamiento del ESP32 en el PCB

El encapsulado ESP-WROOM-32 está diseñada para ser soldado en una PCB huésped. El ESP-WROOM-32 usa una antena de F invertida (MIFA), para la banda de WiFi 2,4 GHz, con un ganancia de 20dBi. La figura 3.8 muestra las seis

opciones de colocación que son comúnmente usadas (la opción uno es usada como referencia). Los resultados de las pruebas ejecutadas por *Espressif* muestran que las opciones 2 y 3 tienen los mejores rendimientos, mientras el de las otras son sub-óptimos(Systems, 2016b).

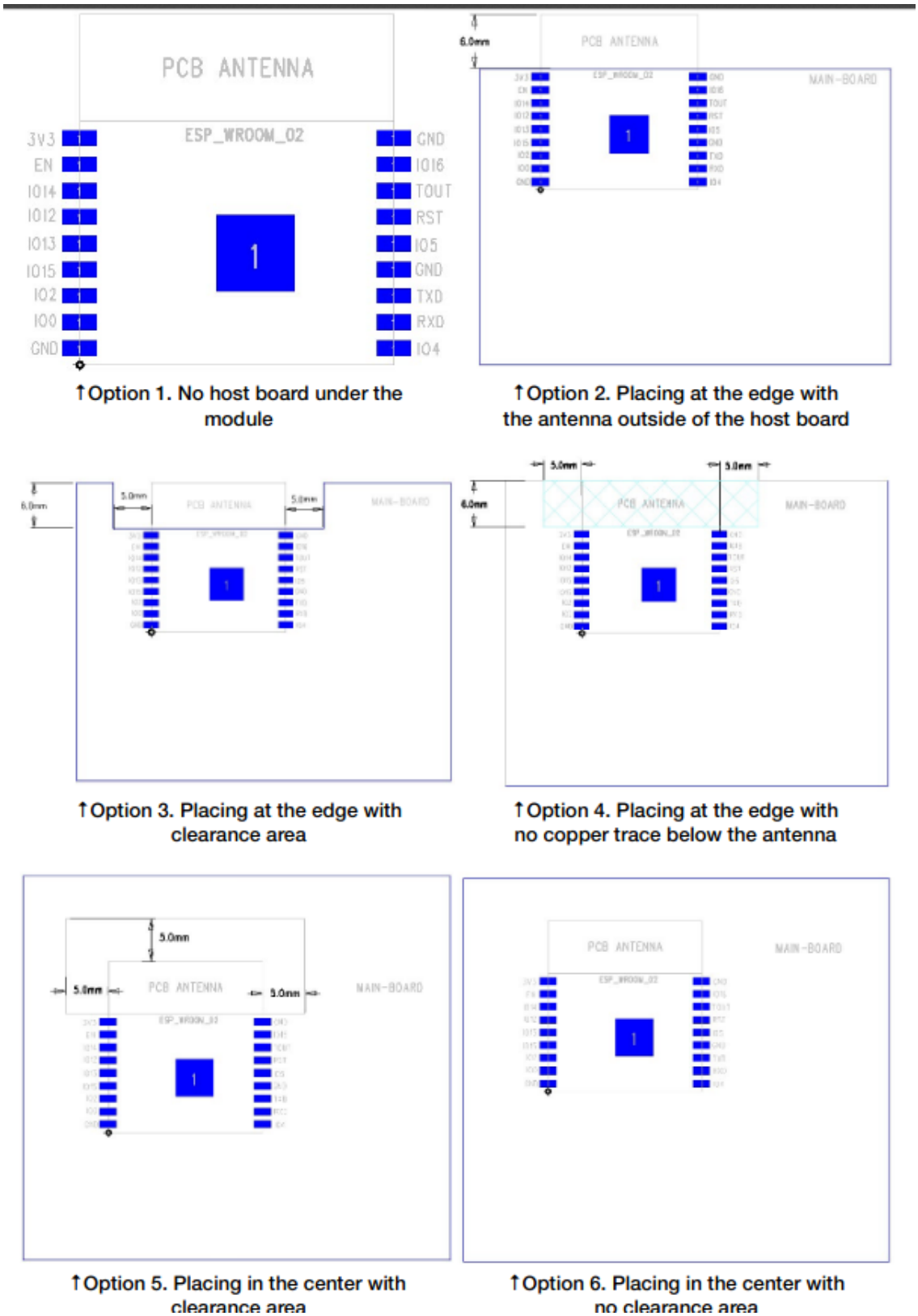


Figura 3.8: Opciones para colocación en PCB

Los resultados del estudio de basan en las mediciones de potencia WiFi y parámetros EVM (magnitud del vector de error)para varios canales para verificar el rendimiento del la antenas en diferentes posiciones. Los resultados fueron los siguientes:

1. Las opciones de colocación 1,2 y 3 básicamente no se muestran afectadas en el rendimiento de la antena siempre y cuando esta este enfrente de espacio abierto. Es recomendado proveer una espacio de al menos 5 mm alrededor de antena en cada dirección.

2. Si el PCB de la antes tiene que ser colocado en una tarjeta huesped, es recomendada la opción 4, que no posee plano de cobre bajo la antena, así hayan algunas pérdidas del rendimiento.

3. La opción 6 posee el peor rendimiento tanto en la transmisión como la recepción, pues se ve afectado por por la tarjeta huésped.

Por lo tanto se tomo la recomendación **??** eligiendo la opción 4 para el diseño.

El chip MAX-485

Espressif provee una ejemplo de conexión para el uso del chip MAX-3485, en cual se indica la topología para usar el UART como interfaz. Se agrega, sin embargo, una resistencia de *pull-up* en el terminal asociado a TDX para evitar niveles indeterminados en dicho pin, causados por el estado en alta impedancia durante recepción.

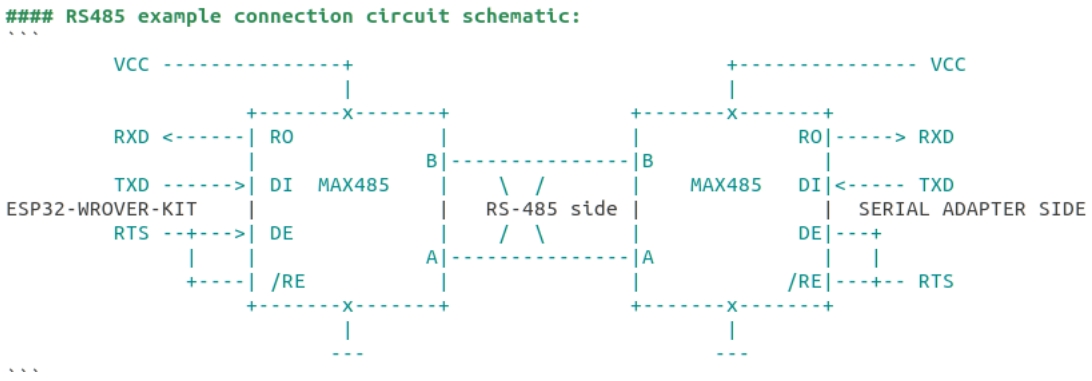


Figura 3.9: Ejemplo de conexión entre MAX-485 y UART

Alimentación

Como se mencionó anteriormente, se usó la fuente LM2596S debido que se ajusta a las especificaciones del diseño. Sin embargo, la tarjeta posee un tamaño comparable al del nodo, lo que fue determinante en la arquitectura del PCB.

Estructura del Circuito Impreso

Se considera una capa con elementos de superficie (resistencia y condensadores). La fuente se adapta a través de pin *headers* en la parte posterior de la capa. Se agregan los bloques terminales para el acceso de la alimentación externa y el bus RS-485. Así mismo, se agregaron 3 pin *headers* para el acceso al UART 0 y dos pares de pin header adicionales para el accionamiento de las funciones de *BOOT* y *EN*, en el ingreso al modo de programación del microcontrolador. El detalle del esquemático de observa en la figura 3.10.

Los terminales se agregaron en la parte posterior de la capa del microcontrolador para evitar obstruir el espacio libre de la antena WiFi. observe las figuras 3.12 y 3.11.

Adicionalemente, se agregó un LED (GPIO 19)con el propósito de que sirva como testigo de comunicación en el encendido y durante la instalación. La tabla 3.18 refleja la lista de materiales empleados y sus características.

Tabla 3.18: Lista de materiales del circuito

Nombre	Descripción	Encapsulado
D2	Diodo emisor de luz enviar	1206
R1	Resistencia 10k Ω	1206
R2	Resistencia 10k Ω	1206
R3	Resistencia 1k Ω	01 x 03 de 2,54 mm
R5	Resistencia 10k Ω	1206
C2	Condensador no polarizado, 100 uF	1206
C4	Condensador no polarizado, 0,1 uF	1206
J1	Pin header	01 x 03 de 2,54 mm
J2	Bloque terminal	01 x 04 de 5.00 mm ,Altech AK300
J4	Pin header	01 x 02 de 2,54 mm
J5	Pin header	01 x 02 de 2,54 mm
U1	Microcontrolador	ESP32-WROOM-32
U2	Covertidor RS-485 TTL (MAX485E)	SOIC-8
DCDC1	Convertidor DC-DC LM2596	pendiente xxx

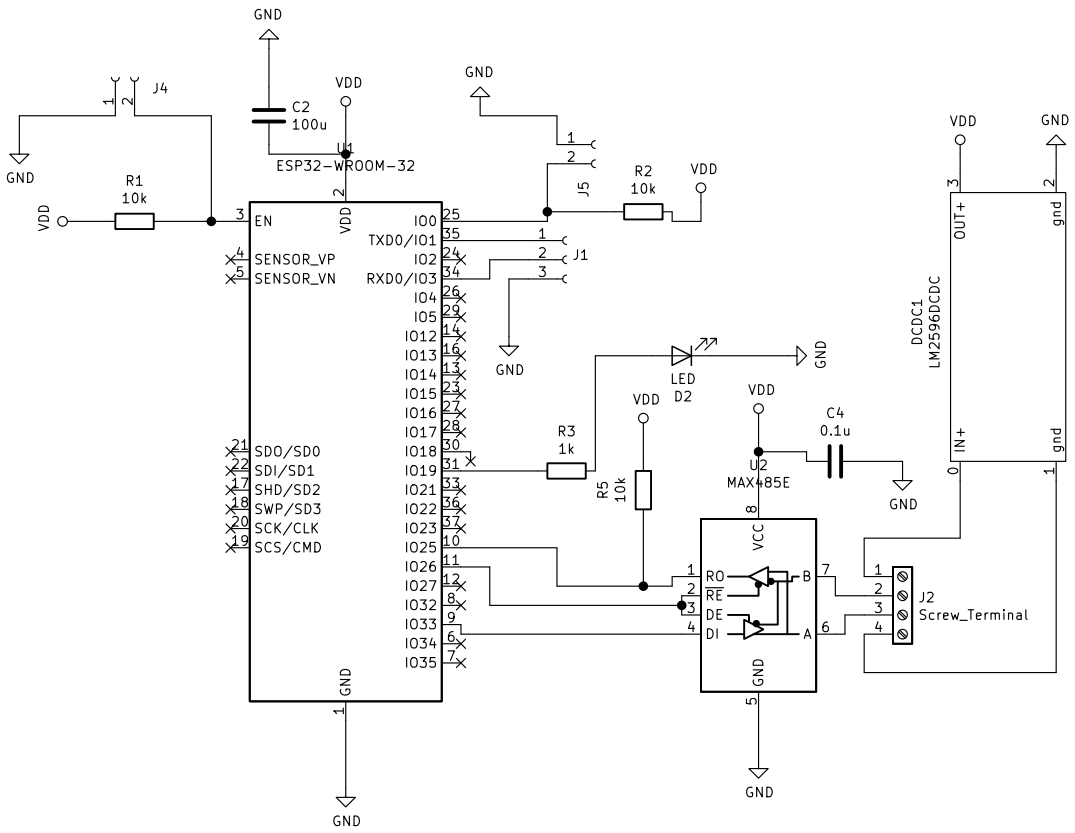


Figura 3.10: Esquemático del nodo

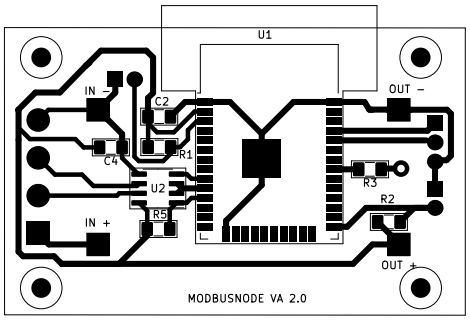


Figura 3.11: Diagrama de circuito impreso capa superior

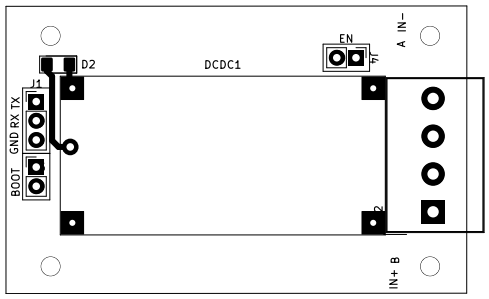


Figura 3.12: Diagrama de circuito impreso capa inferior

3.2.8. Implementación la red mallada diseñada

La red mallada se implementó usando como nodos tarjetas de desarrollo junto con tarjetas para conversión RS-485 a TTL basadas en chip MAX-3485. Cada tarjeta se programó y configuró de acuerdo a la tabla 3.19. Se usaron un total de cuatro nodos con la topología de la figura 3.13 como base. En el bus RS-485 del maestro se conectó un maestro Modbus proveniente de un computador a través de un convertido USB a RS-485 para generar las tramas.

Nodo	Descripción	Tasa de baudios serial
150	Maestro conectado	Variable
151	Esclavos conectados	9600
152	Modo de salto	No aplica
153	Esclavos conectados	9600

Tabla 3.19: Configuración de los esclavos para la implementación

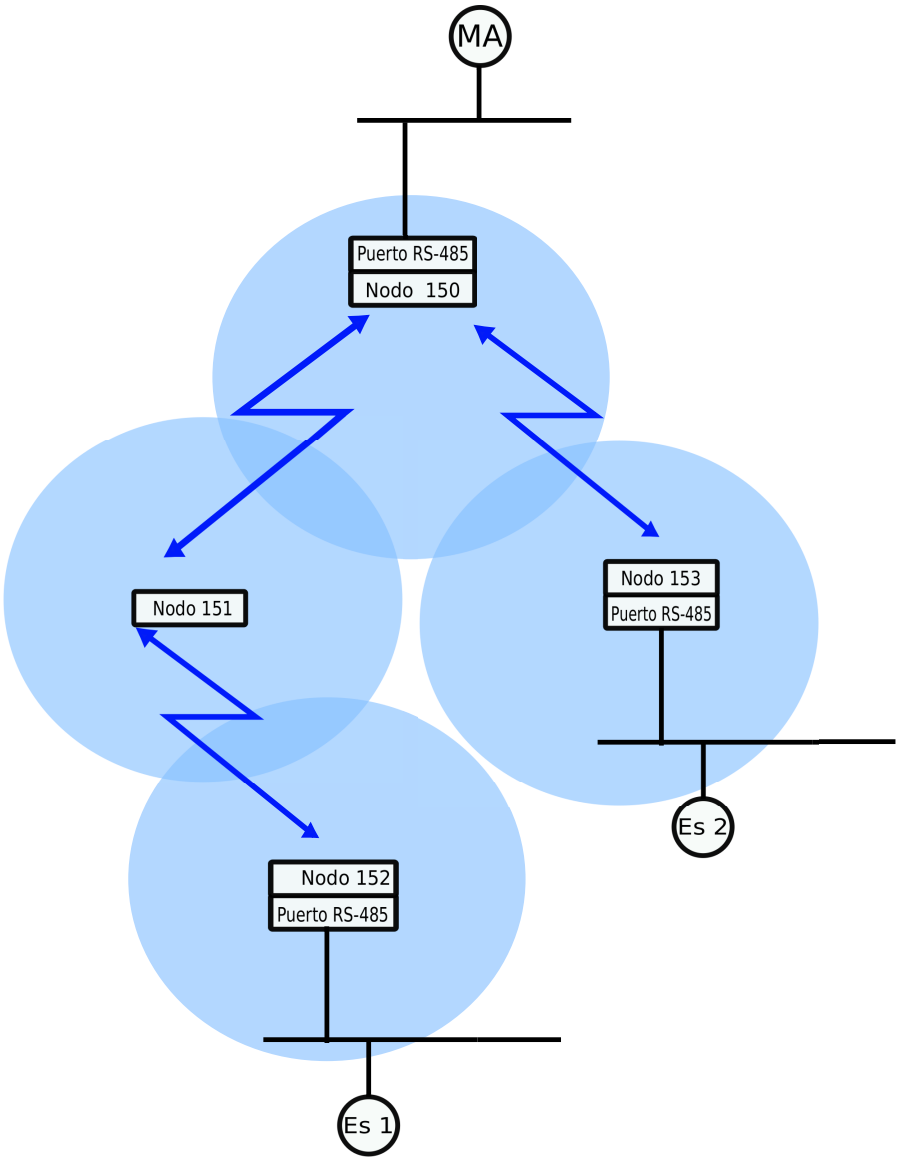


Figura 3.13: Topología de implementación

Por otro lado, en el bus serial de los esclavos se conectaron unas unidades terminales remotas con capacidad de comunicación Modbus. En este caso se conectaron solo una remota por nodo.

Las características de los equipos Modbus usados fueron los descritos en la tabla (Falta agregar). Adicionalmente la lista de materiales usada para la

implementación de la red usando la tarjeta de desarrollo se presenta en la tabla 3.20.

Tabla 3.20: Lista de materiales del circuito de implementación

Nombre	Descripción	Detalle
D2	Diodo emisor de luz enviar	TH 5mm
R3	Resistencia 1k Ω	TH
U1	Tarjeta de desarrollo	ESP32-DevKit-V1
U2	Covertidor RS-485 TTL (MAX485E)	SOIC-8
Alimentación	Cable USB y AC/DC 120VAC/5VDC	-
Cables	cables protoboard	Cantidad 14, macho-macho
Protoboard	cables protoboard	Elego

Ilustración de los nodos

A continuación se presentan imágenes (figuras 3.14 y 3.15) de los nodos donde se implementó la red.

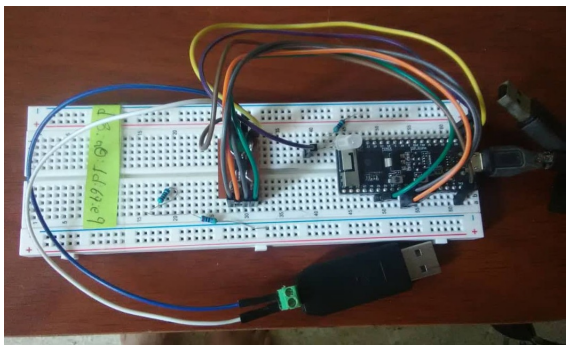


Figura 3.14: Hardware del nodo maestro.

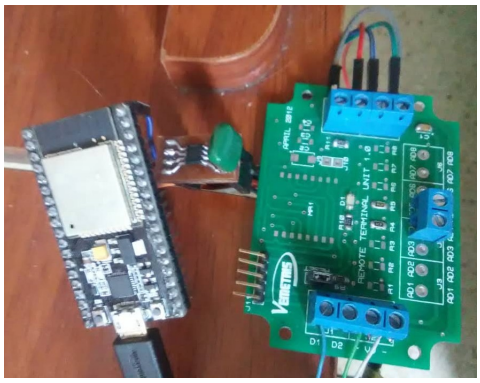


Figura 3.15: Hardware de un nodo con un esclavo Modbus

3.2.9. Análisis el rendimiento de la red de acuerdo a variaciones en los parámetros de transmisión de datos

Las características principales de funcionamiento de la red Modbus son: distancia entre nodos, topología, tasa de baudios serial, velocidad de muestreo, número de variables Modbus y topología. Para la evaluación del rendimiento se varió un parámetro respecto a a la vez. Así se observa la degradación de la calidad de servicio sin caer en numerosas pruebas redundantes. Las variaciones toman los valores expresados en la tabla 3.21.

Tabla 3.21: Variación de parámetros en la pruebas de rendiemietno

Tasa serial [Baudios]	Número de variables	velocidad de muestreo [s]
2400	50	10,0
4800	10	5,0
9600	150	2,0
19200	200	1,0
38400	250	-

Se estimará el error en función de la cantidad de excepciones Modbus que se

obtengan en el maestro por cada esclavo, considerando la ecuación 3.1.

$$error = \frac{cantidad\ de\ excepciones}{cantidad\ de\ solicitudes} \cdot 100\,\% \tag{3.1}$$

Se toma la topología de la figura 3.13 como base para todas las pruebas, excepto, por supuesto, la prueba de variación de topología. Se colocó el nodo 150 como el nodo que posee el maestro conectado, el nodo 151 para ejecutar reenvío de datos y los nodo 153 y 152 con un esclavo Modbus conectados.

Se conectaron los nodos con los nodos inalámbricos a una distancia menor a 10 cm. Se colocó el nodo 150 en la habitación del primer piso de una casa de paredes de concreto, el nodo 153 en otra habitación en la planta baja a una distancia recta del nodo maestro de aproximadamente 15 m, sin linea de vista. El nodo 152 se dispuso en el primer piso a aproximadamente 8 m, con dos paredes intermedias. Y el nodo 151 en la misma habitación del maestro.

Los nodos se colocaron con el plano de la antena perpendicular a la horizontal, sin una rotación específica. Las demás topologías de prueba descritas más adelante, se llevaron a cabo con esta misma disposición física.

Para las variaciones en la tasa de baudios, estás se cambiaron en todas las interfaces seriales, es decir, todas los buses posee la mima velocidad.

En la tabla 3.22 se presentan las condiciones bajo las cuales se hicieron cada una de las pruebas.

Tabla 3.22: Condiciones para las pruebas de rendimiento, la prueba cuatro se asocia a la variación de las topologías.

Prueba	Baudarate [Baudios]	Variables[n]	Tiempo de muestreo[s]
1	9600	50	Variable
2	9600	Variable	5
3	Variable	50	5
4	9600	50	5

CAPÍTULO IV

PRUEBAS EXPERIMENTALES

En este capítulo de detallan las pruebas de rendimiento de la red, basado en los parámetros más relevantes de Modbus sobre línea serial y la topología mallada.

4.1. Resultados y análisis

Se toma como métrica de rendimiento el porcentaje de respuesta Modbus, es decir, la relación entre el número de excepciones Modbus y el número total de solicitudes hechas por el maestro. Las excepciones de no respuesta (*Timeout*) son las que se consideran para el estudio de rendimiento, pues otros tipos de excepciones pueden estar asociadas al mal funcionamiento de los esclavos, y no la red inalámbrica.

Se asume que los esclavos cumplen las especificaciones de equipos Modbus para línea serial con un porcentaje de menor al 1 % para entrega y menor a 2 % para respuesta. Además, se asume que las antenas son omnidireccionales dispuestas verticalmente.

4.1.1. Rendimiento y tiempo de muestreo

En un primer escenario se tiene una red donde se toman variaciones para el tiempo de interrogación Modbus de los esclavos, incluidos los nodos de la red. Se observa que no existe un patrón de degradamiento de rendimiento producto del aumento en la frecuencia de tráfico en la red mallada, quedándose con una tasa de error promedio por debajo de del 1 %. La tabla 4.1 presenta los resultados de las mediciones para distintos tiempos de muestreo del maestro.

Tabla 4.1: Porcentajes de error para variaciones en el tiempo de muestro

	% de error por esclavos					
Tiempo de Muestro [s]	ID 1	ID 2	ID 150	ID 151	ID 153	**Promedio** %
10	0,21	0,74	0,00	0,11	0,89	**0,39**
5	0,10	0,00	0,00	0,10	0,10	**0,06**
2	0,00	0,10	0,00	0,10	0,30	**0,10**
1	0,20	0,00	0,00	0,30	0,30	**0,16**
Promedio por ID	**0,12**	**0,21**	**0,00**	**0,17**	**0,39**	**0,18**

4.1.2. Rendimiento y número de variables

En un segundo escenario se tiene que la red es sometida a variaciones en el número de variables solicitadas (ver 4.2), aumentado gradualmente el tamaño de la trama serial e inalámbrica. En ese caso al igual que el anterior, no hubo una degradamiento notable con las variaciones tomadas.

Tabla 4.2: Porcentajes de error para variaciones en el tiempo de muestro

	% de error por esclavos					
Número de variables [s]	ID 1	ID 2	ID 150	ID 151	ID 153	**Promedio** %
50	0,10	0,00	0,00	0,10	0,10	**0,06**
100	0,34	0,25	0,00	0,08	0,34	**0,20**
150	0,19	0,00	0,09	0,00	0,00	**0,06**
200	0,30	0,30	0,00	0,15	0,23	**0,20**
250	0,20	0,29	0,00	0,10	0,59	**0,23**
Promedio por ID	**0,23**	**0,17**	**0,02**	**0,09**	**0,25**	**0,15**

La máxima cantidad de registros en una solicitud es de 50, más allá del cual los bytes siguientes de pierden y el maestro arroja la excepción *Byte missing error* (Error de bytes faltantes). Esto debe a que la trama inalámbrica se sobrecarga. En el nodo maestro si se puede consultar la cantidad máxima permitida por Modbus de 250 registros, lo que sugiere que la limitación ocurre en la interfaz inalámbrica.

4.1.3. Rendimiento y tasa de baudios

La tabla 4.3 presenta los resultados de las mediciones para distintas tasa de baudios, donde se observan variaciones en la tasa error de casi dos puntos porcentuales en promedio entre algunas tasas de baudios, especialmente para la tasa de 2400 baudios y 19200 baudios. Sin embargo, no existe un patrón específico para atribuir el incremento a una causa específica.

Tabla 4.3: Porcentajes de error para variaciones en la tasa de baudios

	% de error por esclavos					
Tasa de baudios [s]	ID 1	ID 2	ID 150	ID 151	ID 153	**Promedio** %
2400	0,86	2,67	1,62	0,48	4,29	**1,98**
4800	0,32	0,24	0,49	0,41	1,78	**0,65**
9600	0,17	0,67	0,42	0,17	3,74	**1,03**
19200	0,19	4,38	0,37	0,19	4,29	**1,88**
38400	0,29	0,46	0,09	0,19	0,56	**0,30**
Promedio por ID	**0,34**	**1,68**	**0,60**	**0,28**	**2,93**	**1,17**

4.1.4. Rendimiento y topología

Para las topologías presentadas en las figuras 4.1, 4.2 y4.3. Se presentan los resultados de la tabla 4.4, donde se observan que las topologías 1 y 2, poseen la mayor tasa de error promedio. En la topología 1, el error se concentra en el esclavo 1, debido a que es el esclavo más lejano del maestro y al poseer un mayor número de saltos, la probabilidad de que ocurra perdida de trama se acumula con cada

reenvió. En la topología 2 el error es mayor en el esclavo 153, lo cual es producto de su distanciamiento físico del maestro. Ya que las solicitudes de todos los esclavos son realizadas en un periodo corto de tiempo, la última respuesta en llegar seria las provenientes del nodo 153. Considerando que el canal este congestionado por la tramas que llegaron primero, se podrían perder de paquetes.

La topología 3 posee la menor tasa de error promedio, podría ser causado a que posee una distribución más balanceada a las otras topologías, manteniendo una relación equitativa entre el número de enlaces y cantidad de saltos. Ya que la topologías 1, carga los nodos con numerosos reenvíos. Por otro lado, la topología 2 carga el nodo 150 con numerosos enlaces.

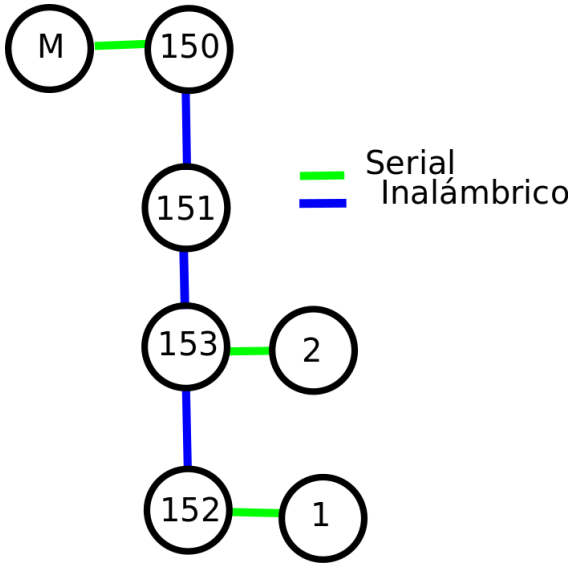


Figura 4.1: Topología 1 de prueba

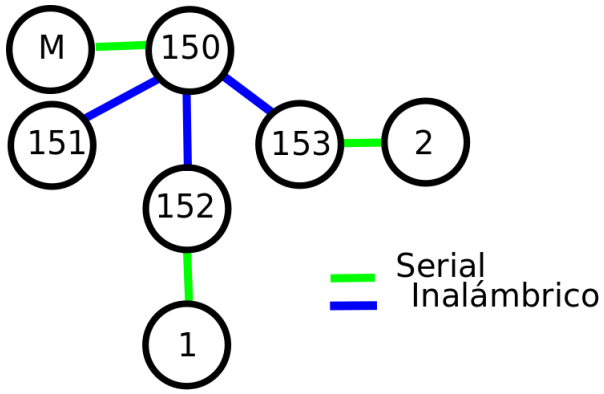


Figura 4.2: Topología 2 de prueba

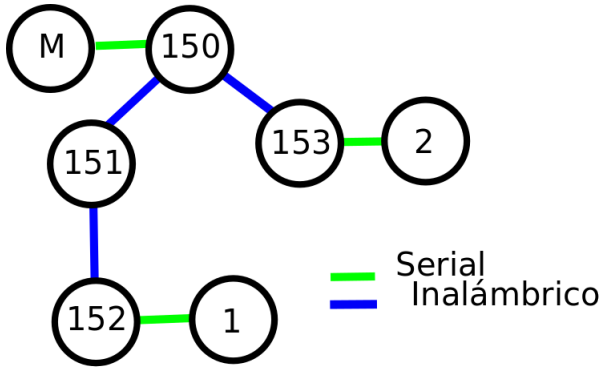


Figura 4.3: Topología 3 de prueba

La tabla 4.4 presenta los resultados de las mediciones para distintas topologías.

Tabla 4.4: Porcentajes de error para variaciones en la topolgía

	% de error por esclavos					
Topología [s]	ID 1	ID 2	ID 150	ID 151	ID 153	**Promedio %**
1	12,20	2,60	0,25	0,25	2,60	**3,58**
2	3,24	0,16	0,08	0,32	11,35	**3,03**
3	0,07	0,14	0,00	0,14	0,00	**0,07**
Promedio por ID	**5,17**	**0,97**	**0,11**	**0,24**	**4,65**	**2,23**

CAPÍTULO V

CONCLUSIONES

Mediante las revisiones bibliográficas se observó que existe gran cantidad de documentación actual respecto a las redes WiFi malladas, a su implementación en distintas áreas y sobre distintos hardware, sin embargo, se identificó como aspecto resaltante que cada una ajusta su funcionamiento al área de aplicación, lo que fue de ayuda para restringir el alcance de la red diseñada.

Se logró implementar una red mallada usando el microcontrolador ESP32, haciendo uso librería de funciones brindadas por el fabricante que agilizan el desarrollo de aplicaciones. Además, el uso del sistema operativo en tiempo real permitió que se implementaran programas en un esquema sencillo y organizado.

Con el uso del chip3485 y el programa de manejo de UART en el microcontrolador ESP32, se logró establecer una comunicación serial entre el microcontrolador y un equipo Modbus. Dicha comunicación estuvo soportada sobre una interfaz RS485 half-duplex de dos conductores.

La red ESP-NOW se ajustó a el protocolo Modbus satisfactoriamente, debido a la bajas tasas de transmisión, tamaño tramas dentro de la capacidad de red y tiempo entre solicitudes. Por lo que obtuvo una red que soportó la transmisión del protocolo Modbus inalabricamente, sin que representará una degradación en la eficacia de comunicación respectó a linea serial, dentro de ciertos límites.

Se obtuvo un diseño del hardware de un nodo que funcione en la red diseñada. Dicho diseño contempló el uso del microcontrolador ESP32, el chip MAX3485, la alimentación y la interfaz física de la conexiones.

Finalmente, se ejecutaron pruebas variando parámetros de comunicación Modbus, donde se logró observar aquellas características cuya modificación altera en mayor grado la eficacia de la red para la transmisión de datos, como lo es la topología.

CAPÍTULO VI

RECOMENDACIONES

Se recomienda ampliar la capacidades de comunicación la red implementando un modo de compuerta en TCP/IP en en el nodo maestro, para abrir la posibilidad de acceder remotamente a esta, incluso Internet, creando una característica relacionada al Internet de la cosas en el área Industrial (IIoT).

Dada la relevancia de la topología en el rendimiento de la red, debido a la congestión de información en los nodos con múltiples conexiones, se propone implementar un control centralizado de gestión de datos y de balanceamiento de red para disminuir la pérdida de paquetes en nodos congestionados y que establezca estructuras de red mas eficaces, de forma dinámica.

En cuanto al enrutamiento se recomienda realizar estudios para la implementación de un gestor de rutas de red que use algoritmos del camino corto, de acuerto a las conexiones establecidas por el usuario, tomando como base el algoritmo de Dijsktra o el algoritmo de Prim. Suponiendo en cada caso que el peso de cada enlace en la intensidad de señal recibida.

También proponen hacer estudios de rendimiento la propagación de las tramas broadcast usando algoritmos heurísticos constructivos, observado este proceso como el problema del vendedor viajero (TPS). Esto para la aplicación de actualizaciones de configuraciones comunes en los nodos que componen la red o los equipos Modbus involucrados.

Apéndice I

Datos recolectados

xxx anexo datos crudos

Apéndice II

Código fuente

codigo fuente

Apéndice III

TÍTULO DEL ANEXO

REFERENCIAS

Al-Husainy, M. (2013, 11). *Mac address as a key for data encryption.*

Bahr, M. (2016, 10). Update on the hybrid wireless mesh protocol of ieee 802.11s. *Siemens Corporate Technology, Information and Communications*, 1.

Cheng, Y., Yang, D., y Huachun, Z. (2018, 02). Det-lb: A load balancing approach in 802.11 wireless networks for industrial soft real-time applications. *IEEE Access*, *PP*, 1-1. doi: 10.1109/ACCESS.2018.2802541

Chew, D. (2018, 10). *Mac layer.* doi: 10.1002/9781119260608.ch5

Hiertz, G., Denteneer, T., Max, S., Taori, R., Cardona, J., Berlemann, L., y Walke, B. (2010, 03). Ieee 802.11s: the wlan mesh standard. *Wireless Communications, IEEE*, *17*, 104 - 111. doi: 10.1109/MWC.2010.5416357

Lech, P., y Wlodarski, P. (2017, 04). Analysis of the iot wifi mesh network. En (p. 272-280). doi: 10.1007/978-3-319-57264-2_28

Milic, B., Brack, S., y Naumann, R. (2014, 05). Hierarchical configuration system for wireless mesh networks in manufacturing industry. En (p. 1-8). doi: 10.1109/WMNC.2014.6878867

Modbus Organization, I. (2006, 10). *Modbus messaging on tcp/ip implementation guide v1.0b.*

Modbus Organization, I. (2017, Abril). *Modbus application protocol specification.* ([Internet; accedido el 25 de Octubre del 2019] Disponible en: `http://modbus.org/docs/Modbus_Application_Protocol_V1_1b3.pdf`)

Rifki Muhendra, M. B., Aditya Rinaldi. (2016). Development of wifi mesh infrastructure for internet of things applications. *Engineering Physics International Conference, EPIC 2016*, 331.

Services, A. W. (2020a, 05). *What is a general purpose operating system?*

([Internet; accedido el 08 de Mayo del 2020] Disponible en: https://www.freertos.org/about-RTOS.html)

Services, A. W. (2020b, 05). *What is freertos?* ([Internet; accedido el 08 de Mayo del 2020] Disponible en: https://www.freertos.org/about-RTOS.html)

Sharma, P., y Singh, G. (2016, 10). Comparison of wi-fi ieee 802.11 standards relating to media access control protocols. *International Journal of Computer Science and Information Security,*, *14*, 856-862.

Singh, M. (2019, 01). Wireless mesh networks architecture. En (p. 11-14). doi: 10.1007/978-981-13-0674-7_2

Systems, E. (2016a, 06). *Esp-wroom-02pcb design and module placement guide.* ([Internet; accedido el 27 de Abril del 2020] Disponible en: https://docs.espressif.com/projects/esp-idf/en/latest/esp32/api-guides/mesh.html)

Systems, E. (2016b, 06). *Esp-wroom-02pcb design and module placement guide.* ([Internet; accedido el 27 de Abril del 2020] Disponible en: https://www.espressif.com/sites/default/files/documentation/esp-wroom-02_pcb_design_and_module_placement_guide_0.pdf)

Systems, E. (2019, 05). *Esp32-wroom-32 datasheet.* ([Internet; accedido el 14 de Mayo del 2020] Disponible en: https://www.espressif.com/sites/default/files/documentation/esp32-wroom-32_datasheet_en.pdf)

Temprano G., A. J. (2017, 11). *Diseño de una red industrial modbus rtu inalámbrica.*