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, Abril 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, Abril 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. Mención Electrónica. Año 2020, xvii, 144 pp.

Palabras Claves: Palabras clave.

Resumen.- Escribe acá tu resumen

ÍNDICE GENERAL

LISTA DE FIGURAS

LISTA DE TABLAS

LISTA DE ACRÓNIMOS

INTRODUCCIÓN

En el área de la ingeniería eléctrica, las comunicaciones 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. Las comunicaciones inalámbricas se clasifican básicamente según la banda de frecuencia o el estándar que satisface, así las tecnologías IEEE 802.15.1 (bluetooth), IEEE 802.11 (WiFi) y GSM (telefonía celular).

Las tecnologías basadas en el estandar IEEE 802.11, también conocidas como WiFi, han ganado popularidad en el mundo de las comunicaciones proveyendo interconectividad entre clientes (PC, laptops, smartphones) y puntos de acceso. Las redes que usan dicha tecnología se clasifican (según la topología) en centralizadas o descentralizadas dependiendo si la conectividad entre clientes posee como intermediario o no 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 datosCheng, Yang, y Huachun (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 físicas, entre las que se encuentras la RS-485 y RS-232, en la cual 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 a un entorno industrial, cuyos nodos estarán constituidos por microcontroladores ESP32. Este trabajo tiene por finalidad presentar la metodología

para el diseño e implementación de la red , además se presentan los detalles del planteamiento del problema y la factibilidad de proyecto así como el cronograma para la ejecución de actividades para lograr los objetivos planteados .

Objetivos

Objetivo General

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

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.

CAPÍTULO I

MARCO HISTÓRICO

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, 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. 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.

CAPÍTULO II

MARCO TEÓRICO

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

2.1. Fundamentos de las redes WiFi malladas

Las redes WiFi malladas(WMN) se pueden definir como una red que permite la comunicaion entre nodos a través de múltiples saltos en una topología mallada Bahr (2016). Los nodos intermedios son capaces de reenviar los datos hasta llegar al nodo destino. Las WMN usualmente se componen de 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 en un dispositivo electrónicos que sirve como un intermediario entre dos o mas redes para transportar los datos de una red a otra. Y las compuertas de enlace es un dispositivo electrónico que conecta 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).

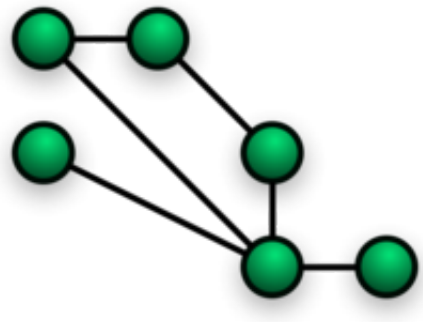

Figura 2.1: Topología Mallada

Una red WiFi mallada se establece en la banda para comunicación WiFi (sea 2,4 GHz o 5GHz), donde hay estaciones 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. Como 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 estan encapsuladas en la capa MAC, varian de un protocolo a otro. Estas funciones incluyen, pero no estan limitadas a, técnicas de acceso múltiple, sincronización un medio abierto y correccción de errores. Chew

(2018)

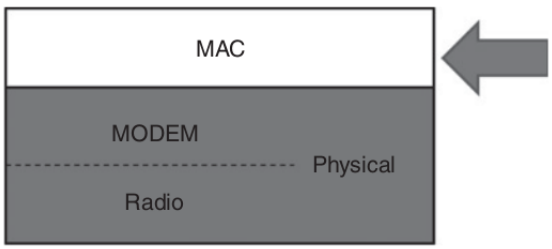

Figura 2.2: El propósito de la capa MAC es administrar el espectro

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. La dirección MAC trabaja en la segunda capa del modelo OSI, siendo identificada como capa de vínculo.

Las direcciones MAC se componen de 12 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 estan regulados por un estadar 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) esta basado en una propuesta del la Organización Internacional de estándares (ISO) para la la estandarización de pilas de protocolos. El modelo consiste en siete (7) capas.

1. Una capa debe existir para cada nivel de abstracción.
2. Cada para 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 sufientemente pequeña para que la arquitectura no sea incómoda.

Las siete capas del modelo OSI juntas son solo un modelo de referencia, y no son una 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 y,

7. La capa física.

Para entender las funcionalidades y la interrelación entres estas capas, es beneficioso estudiarla 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 por asegurarse de la integridad de dichos bits tanto por la entrega como por la interpretación. Los detalles específicos de cuantos Volts representan el "0"lógico y cuantos representan el "1", 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. Redes broadcast 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 a canales compartidos, con el nombre de subcapa de control de acceso al medio (MAC).

2.2.3. La capa de red

La capa de red en la responsable por controlar la operación de la subred. La carga de la trama de datos, en esta capa, es llamada paquete. Esta capa determina como mover el paquete desde la fuente al destino usando las rutas apropiadas. La determinación de dichas rutas puede puede ser estático o dinámico. La capa de red 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 en esta capa.

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, para pasarlas a la capa de red, y asegurarse que estas piezas llegar correctamente al destino. Para propósitos de eficiencia, la capa de transporte puede multiplexar varias conexiones de transporte en una sola conexión.

La capa de transporte es la primera capa fin-a-fin de la pila. En las capas más bajas, la intección 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 esta capa, sin embargo, es siempre entre puntos finales.

El control de flujo juega un rol importante en la capa de transporte (Así como en las otras capas).

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 (a distance-vector routing protocol for IPv6 and 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 una poder de computación significativo, lo cual limita sus implementaciones. Sin embargo, es de notar que no todo los equipos (para comunicaciones WiFi) soportan un protocolo particular como es el caso de HWMP. 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 resaltan las más relacionadas con el objetivo del trabajo.

2.3.1. Pior Lech and Przemyslaw Wlodarski

En su artículo llamado *Analysis of the IoT WiFi Mesh Network* (Análisis de las redes WiFi malladas IoT), llevan a cabo un análisis estadístico de rendimiento sobre una red WiFi mallada. Para lograr eso usan la version de desarrollo del módulo de comunicación NodeMCU ESP8266. La operación básica en la red de cada node es en el modo AP+STA(Punto de acceso y estación). La estrategia de entre los nodos esta basada en la transmisión de un único mensaje a los nodos con número mayor de el asociado a la estación receptora. Todos los nodos posee un número fijo asignado que crece desde la fuente (RPi 1) en la dirección de la estación destino (RPi 2).

De acuerdo a la figura 2.3.1 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 reenvian 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.

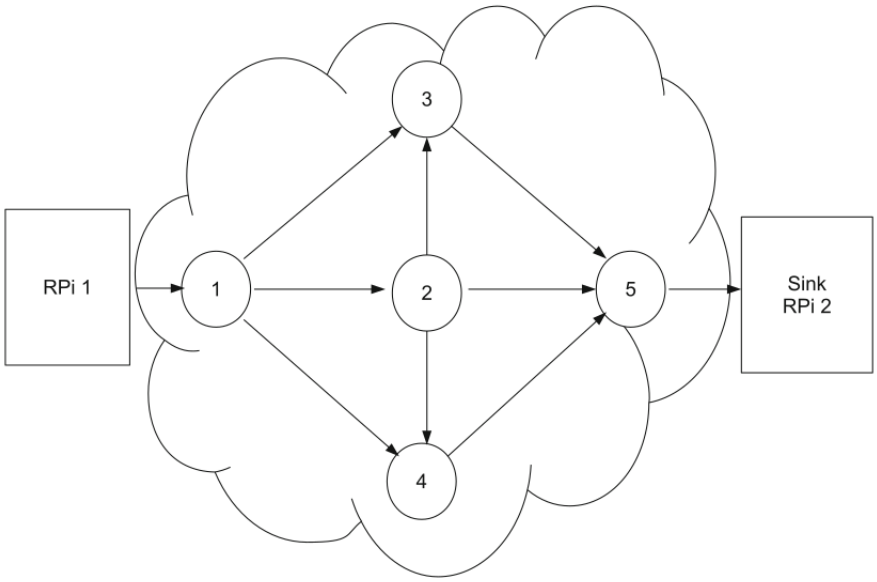

Figura 2.3: Topología usada por Lech y Wlodarski

2.3.2. Yujun Cheng, Dong Yang y Huachun Zhou

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* (Un Enfoque de Carga Equilibrada en Redes Inalámbricas 802.11 para Aplicaciones Industriales Ligeras en Tiempo Real) propone un 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 balaceamiento de carga. Casi todos los adaptadores 802.11 se asocian con el punto de acceso que posea la mayor intensidad de señal. Basados en la mayor intensidad señal las redes son propensas a una distribución desigual de recursos, lo que significa que algunas APs 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 esta simplemente relacionado con la intensidad de señal, sino que también esta 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 sobre cargados 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 2.3.2.

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 2.3.2 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.

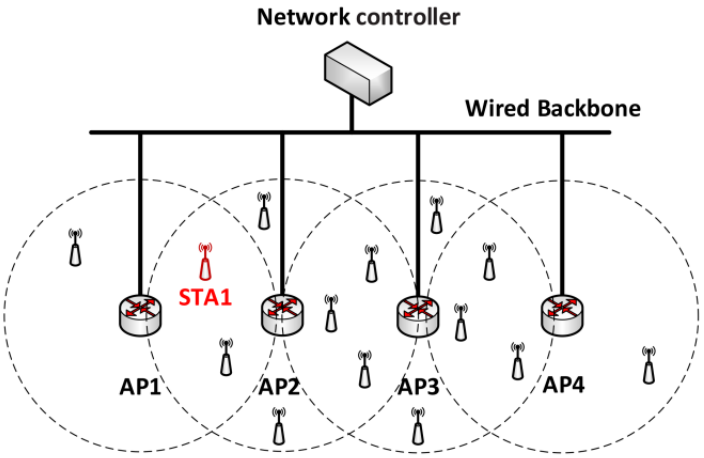

Figura 2.4: Topología usada por Yujun Cheng, Dong Yang y Huachun Zhou

2.3.3. El protocolo ESP-MESH de Espressif

ESP-MESH es un protocolo de red construido encima 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.

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 2.3.3).

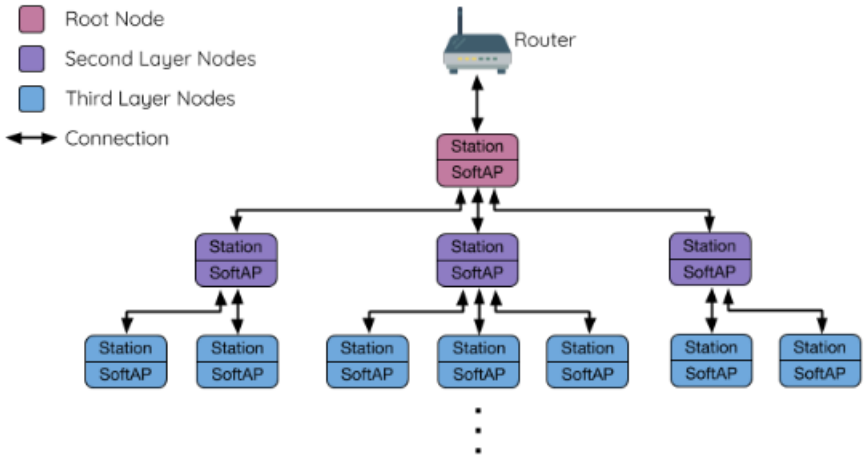

Figura 2.5: Topología de árbol de ESP-IDF

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, usada para enrutar correctamente los paquetes al node destino correcto. La tabla de enrutamiento 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.

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).

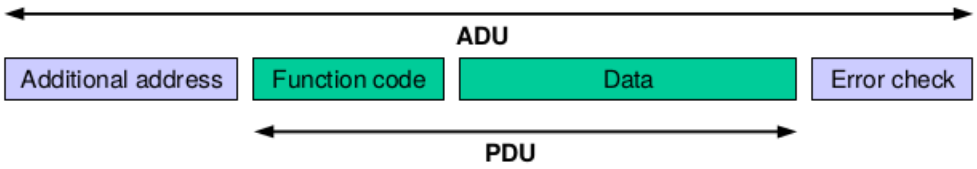

Figura 2.6: Trama general de procolo MODBUS

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 servidos que tipo de acción llevar a cabo.

El código de función ocupa un byte, poseyendo 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 items 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 hubieron 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 a la excepción.

El tamaño de la ADU está limitado en una línea serial a 256 bytes, por lo tanto, si le restamos un byte para la dirección del servidos 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.

2.5. Microcontrolador ESP32

El ESP32 es un chip con integración WiFi y Bluetooth diseñado con la tecnología de ultra bajo consumo de 40 nm. Está diseñado para alcanzar desempeño importante de energía sobre radio frecuencia.

El ESP32 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.

Además es una solución integrada, ya que 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.

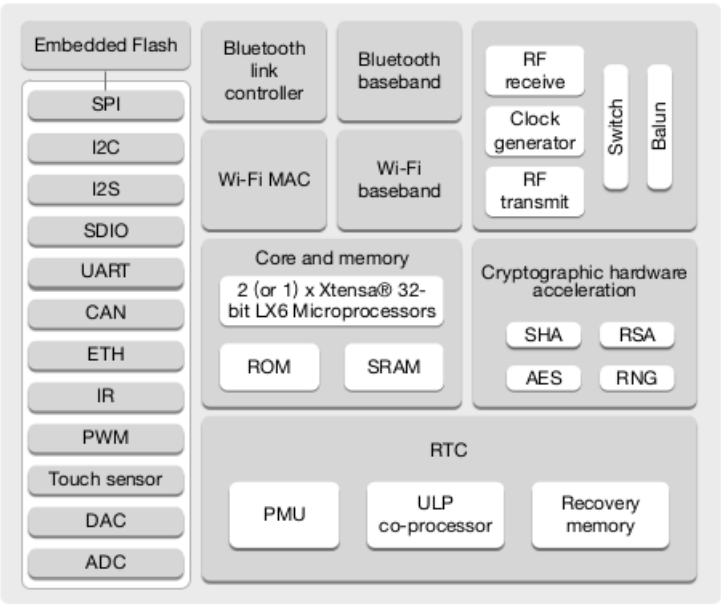

Figura 2.7: Diagrama funcional del ESP-32

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

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 guardian en cada uno.
- Un temporizador RTC
- Perro guardian RTC

2.5.4. Intefaces de periféricos avanzadas

- 34 GPIO programables.
- Convertidor analógio 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
- Encripcion de flash
- 1024-bits OTP, hasta 768-bit clientes.
- Aceleración de criptografía por hardware
  - AES
  - Hash (SHA-2)
  - RSA
  - ECC
  - Generador de números aleatorio

CAPÍTULO III

MARCO METODOLÓGICO

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 se ajustas 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 funcionamiento descritos anteriormente, se tiene que la primera se concentra en el algoritmo de optimización de la topología de la red, sin tomar en consideración el hardware de implentación. La segunda, provee una aproximación más realista, proveyendo las 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 que para la formación de la red es necesario un enrutador WiFi externo que brinde las direcciones IP a la red mallada, lo que deriva en un centralización, pues si el router falla, la red perece.

Sabiendo que la implementación se realizará en el microcontrolador ESP-32, se encontró que la librería ESP-NOW provee comunicación WiFi punto a punto entre microcontroladores, con posibilidades de encriptación, comunicación broadcast,

multicast y unicast. Además, provee de direccionamiento por direcciones MAC. Dicha librería no está orientada a redes malladas, sin embargo, posee las características para la contruccion de una, ya que se puede formar de forma descentralizada, extensible, y ajustable a los protocolo de aplicación.

Por otro lado tenemos que la interfaz serial puede ser lograda usando un UART del MCU configurado para el funcionamiento *half-duplex* de RS-485. Existen diferentes en el mercado variedad de chips para la conversión de niveles TTL a RS-485, dentro de los más usados se encuentra el MAX-485 de *Maxim Integrated*.

Las aplicaciones del procotocolo MODBUS son numerosas, sin embargo, la variación MODBUS RTU provee alto aprovechamiento de la memoria, ya que la información ocupa solo los bytes necesario y puede ser directamente interpretada por el microcontrolador a diferencia de la versión ANSII.

3.1.1. Requerimientos de diseño

Principalmente debe ser una red confiable, estable, y eficaz; aprovechando las principales características de las redes malladas, como saltos múltiples y un enrutamiento flexible.

La red mallada tiene que ser aplicable a dispositivos existentes que trabajen sobre linea serial MODBUS RTU, sin ninguna modificación, es decir, que la red sería transparente para los equipos Modbus, para permitir que sea de rápida instalación y no altere la infraestructa ya implenetada en la industria. Así mismo, debe soportar la transmisión todas las funciones Modbus entre los equipos.

Los nodos deben poder ser configurables, pudiendo ajustar las rutas de la red y la tasa de baudios de la interfaz serial. Así, se crea un red adaptable a los parámetros disponibles en las aplicaciones.

El ESP-32 brinda la posibilidad de llevar a cabo las mencionadas características a cabalidad, sin embargo, solo bajo pruebas es posible determinar la calidad de las características.

3.1.2. Diseño Básico de la red

En primer lugar se estableció que la rutas de la información fueran estáticas, esto permite la implentación de algoritmos más simples que sean más confiables sobre la arquitectura del microcontrolador. El establecimientos de la rutas se ejecuta mediante la implentación de tablas de enrutamiento en cada uno de los nodos.

Se estableció que una serie de identificadores Modbus esten reservados para los nodos de la red, es decir, los nodos serán esclavos adicionales de la red. Esto permite que se use el protocolo Modbus para el acceso a la configuración del nodo, como lo es establecer la tabla de enrutamiento, la velocidad de los baudios de la linea serial y el identificador del mismo.

Características de los nodos

Se realizó el programa para que todos los nodos tuviesen el mismo programa sin importar el rol que posea en la red (maestro, intermedio o final). Se llama nodo maestro a el que posee el maestro Modbus conectado, nodo intermedio a aquel cuya funcionalidad solo en reenviar datos inalámbricos y nodo final a el que posee esclavos Modbus. Sin embargo, los nodos finales también pueden reenviar datos.

Los nodos en sí soportan el protocolo Modbus para ser interrogados y configurados. Por lo que los se reservaron direcciones para los nodos, en este caso los identificadores desde 101 a 255. Más concretamente, los nodos de la red son

esclavos adicionales de la red Modbus.

Cada nodo tiene la posibilidad de ofrecer el servicio de configuración de su identificador Modbus, la tasa de baudios de la interfaz RS-485, y la tabla de enrutamiento. Esto a través del acceso a los registros de retención. Además de la funcionalidad de reinicio y reseteo de fabrica(hardware). Las direcciones de los registros con sus funcionalidades se expresan en la tabla 3.1.

Dirección	Tipo de registro	Descripción
01	Registros de rentención	Identificador
02	Registros de rentención	Tasa de baudios
256-512	Registros de rentención	Tabla de enrutamiento
0	Bobina	Reinicio

Tabla 3.1: Tablas primarias de los modelos de datos Modbus

Un esquema del funcionamiento se puede apreciar en la figura 3.1.2.

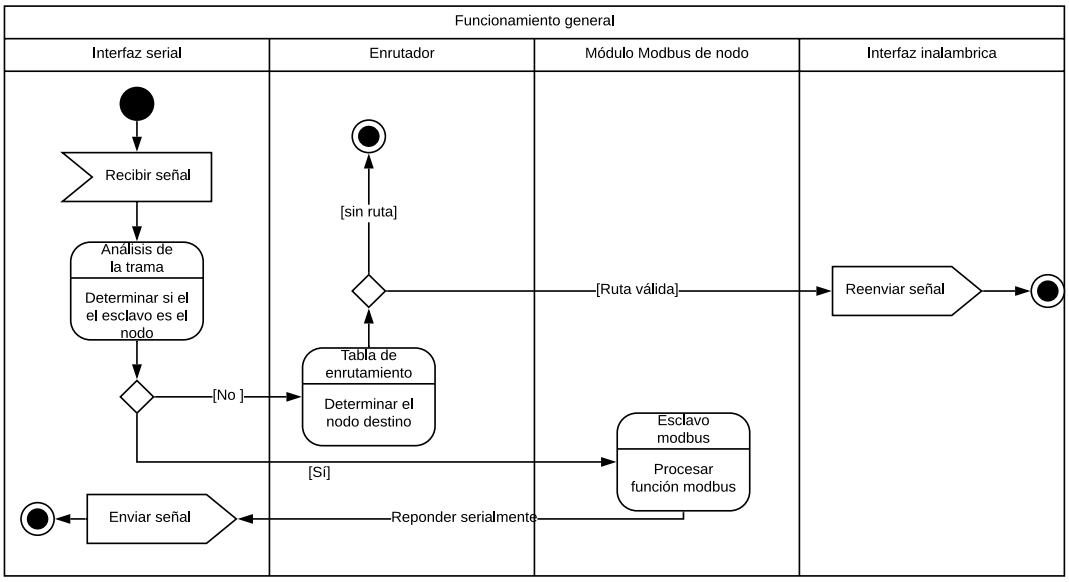

Figura 3.1: Diagrama de flujo del funcionamiento general de un nodo de la red

Características de enrutamiento

La tabla de enrutamiento consisten en un vector de 256 casillas, donde la posición está asociada a el esclavo Modbus y el valor en la casilla se relaciona con la ubicación de dicho esclavo. Cabe resaltar que el identificador a la que se refiere la tabla de enrutamiento como destino no es la localización del esclavo en la red, en su lugar es el siguiente nodo a reenviar los datos para llegar al esclavo en cuestión, es decir, los nodos solo conocen un salto hacia adelante y un salto hacia atrás para cada esclavo.

La tabla de enrutamiento se llena mediante los registros de retención, a partir de la dirección 256, la cual se asocia al esclavo 1. Esta debe ser determinada por el usuario que conoce la posición de los nodos en el área a implementar.

Cuando se recibe un trama, se extrae el identificador del esclavo y surgen dos casos: corresponde a un número diferente de cero ó es cero. Si es cero, entonces el nodo no tiene ruta a ese nodo, en caso contrario el número puede corresponder al identificador del nodo en cuestión o a otro nodo al cual se reenviarán la trama. En caso de que sea el nodo en cuestión entonces ha llegado la trama ha llegado a su destino.

En la figura 3.1.2 se muestra un ejemplo de la tabla de enrutamiento. En el primer lugar supondremos que se trata de la tabla de enrutamiento del esclavo 255. Luego, se observa que para cuando el identificador del nodo coincide con elemento de la tabla de enrutamiento, se dice que el esclavo se encuentra en la linea serial de dicho nodo, así el esclavo coincide con el nodo destino significa que el esclavo a interrogar es el nodo.

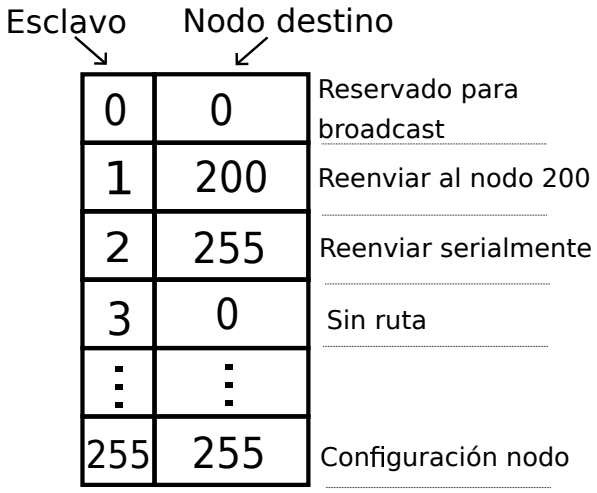

Figura 3.2: Ilustración de la estructuras de la tabla de enrutamiento

En la ADU de la comunicación inalámbrica existe un dato de la trama que identifica el sentido de la trama (hacia o desde el maestro). Cuando se recibe información inalambricamente y se debe reenviar, la aplicación usa la información del sentido para asignar la dirección del nodo destino, es decir, si va hacia adelante (desde el maestro), entonces se usa la tabla de enrutamiento de manera convencional y si hacia atrás se usa el nodo de donde provino la trama inicialmente; por ejemplo, si la trama hacia el esclavo provino desde el nodo 200, cuando vaya hacia el maestro se reenviará la trama a través del nodo 200. Concretamente, las tramas posee el mismo camino en ambos sentidos.

Existe una diferencia cuando se reciben datos serialmente, porque no hay una manera directa de determinar el sentido de la información simplemente analizando la trama recibida, sin embargo, es posible asignarla con el apoyo de la tabla de enrutamiento, pues si el nodo destino es el nodo en cuestión significa que es una respuesta Modbus.

Características Modbus de los nodos

La capacidad del nodo para funcionar como esclavo Modbus dentro de la red permite que sea configurable los parámetros de funcionamiento como la velocidad de comunicación serial, identificadores Modbus respecto y la tabla de enrutamiento.

Estos parámetros se le asignaron direcciones de aplicación Modbus en los registros de retención, para que pudiesen ser leídos y escritos desde el maestro. Otras funciones binarias como la de reinicio se ejecuta a través de la escritura de la bobina cero.

- La dirección cero corresponde al identificador del nodo.
- La dirección uno corresponde a la tasa de baudios.
- Desde la dirección 256 hasta 512 corresponde a la tabla de enrutamiento.

Cabe resaltar que estos nodo pueden ser interrogados tanto serialmente como inalambricamente, es decir, con la red ya instalada. Las funciones no usadas proveerán la respuesta de "Función ilegal".

El hardware

El hardware del nodo se puede dividir en tres bloques: el microcontrolador, la alimentación y la interfaz RS-485. Esto de puede ilustrar en el esquema de la figura 3.1.2.

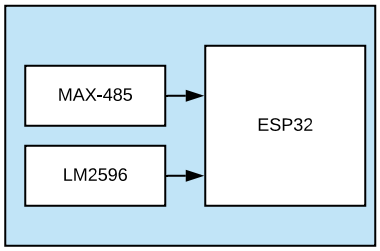

Figura 3.3: Diagrama de bloques para el hardware del nodo

Microcontrolador Para la aplicación se usó el encapsulado ESP-WROOM-32, debido a disponibilidad. Se debe tener en cuenta que se deben dejar diponibles los pines del UART cero para la programación del, además de dos pines adicionales para inducir el estado de programación, los cuales son el pin *EN* y el pin de propósito general cero.

Alimentación Para proveer los 3,3 V necesarios se usó la tarjeta pre-fabricada basada en el regulador lineal LM259S, que posee la capacidad de salida ajustable a la tensión requerida con tensiones de entrada de hasta 30 V. Lo cual es beneficioso ya que en el contexto industrial es común las tensiones del 24 V y 5 V.

Intefaz RS-485 Usando el chip d *MAX-485* de *MAXIM* es posible obtener niveles de RS-485 a partir de una interfaz serial TLL, proveniente del UART1.

3.2. Detalle del Diseño

3.2.1. Programación del ESP-32

Espressif provee 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.

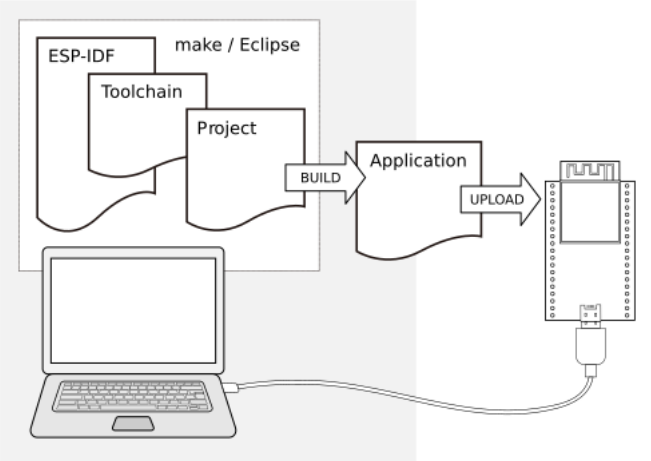

Figura 3.4: Ilustración del combo para el desarrollo de aplicaciones en el ESP32

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*.Un sistema operativo de tiempo real es una herramienta eficiente para dispositivos con poca capacidad de procesamiento. Es decir, aquellos dispositivos que no cuentan con una variedad de núcleos o kernels. RTOS trata de implementar en estos microsistemas una similud de comportamiento de múltiples unidades de procesamiento aún estando contenidas en un solo núcleo. Lo cuál es una ventajaimportante para la implementación de sistemas de mayor complejidad de código.La lógica funcional del método de programación RTOS es mediante establecimiento de tareas e interrupciones controladas por órden de prioridad y por el timer principal del sistema. Y soporta las casi las misma características desde el punto de vista de creación y manejo de tareas, así como de las demás herramientas que ofrece un sistema operativo en tiempo real.

Dicho sistema operativo permite que se creen y desarrollen programas en lenguaje ANSI C, implementado las librerias para el manejo del microcontrolador proveidas 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 monitores fueron añadidas.

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

En primer lugar se definen el la aplicación principal (app main) las funciones que se llevaran a cabo al iniciar el microcontrolador. La incialización 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 libreria 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 incialización 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 mantenan 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. La de envío provee información de la MAC y el estado (envío exitoso o no) y la de recepción la MAC, los datos y la longitud de la trama.

En estas funciones basicamente se pasan los datos a la aplicación principal junto con una variable que ayuda a conocer el estado del sistema del tipo *espnow_event*.

Registro de puntos

En la librería *espnow* existen unos entes llamados *Peers* (Puntos), los cuales conservan la información de cada dispositivo con el que se pueda interactuar. 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 3.2

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

Tabla 3.2: Tablas primarias de los modelos de datos Modbus

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, es administrada por la cola *espnow_queue*. Cabe mencionar que las colar 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 exiten 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 envia cinco veces o hasta sea recibida la trama por el nodo destino. Solo cuando se incializa en nodo se envia tramas broadcast para indicar a los nodos vecinos su precencia.

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 maquina de estado 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 siguiente:

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*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, multificarla 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 estruturas de datos diseñadas para su configuración. Por lo tanto, se configuró el UART1 para la comunicación RS-485 *half-duplex*. El UART0 esta reservado para la quema del microcontrolador y logs de información y error.

El manejador de eventos proporciona una arquitectura de maquina de estados donde se puede aceptar los datos si estos se reciben correctamente. Una vez esto pasa que crea una maquina 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. Por defecto 115200 baudios, paridad desactivada, un bit de parada y modo RS-485 half-duplex.

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*

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

Tabla 3.3: Descripción de la función *vNotiLEDinit*

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

Función *FormatFactory*

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

Tabla 3.4: Descripción de la función *FormatFactory*

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*

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

Tabla 3.5: Descripción de la función *vConfigLoad*

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 ejcuta al energizar el nodo.

Función *vConfigGetNVS*

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

Tabla 3.6: Descripción de la función *vConfigGetNVS*

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.

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

Función *vConfigSetNode*

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

Tabla 3.7: Descripción de la función *vConfigGetNVS*

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*

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

Tabla 3.8: Descripción de la función *vConfigGetNVS*

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*

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

Tabla 3.9: Descripción de la función *uComGetTransData*

Es una función similar a la descrita en la sección 3.8, sin embargo, esta arroja tres posibilidades. Determina si la trama recibida debe ser enviada seriamente, reenviarse a otro nodo o simplemente configurarse a si mismo.

Función *vEspnowGetOldPeers*

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

Tabla 3.10: Descripción de la función *vEspnowGetOldPeers*

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*

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

Tabla 3.11: Descripción de la función *espnow_data_prepare*

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*

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

Tabla 3.12: Descripción de la función *espnow_data_parse*

Cumple la función complementaría a la función 3.2.6, descompone los datos recibidos, comprueba el chequeo de error, etc. Se usa en la maquina 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 fue desarrollado de forma iterativa sobre las tarjetas de desarrollo, por lo que la red fue implementada miestras era desarrollada.

Aunque el desarrollo se realizó en las tarjetas de desarrollo, y la implementación con estas es totalmente válido para realizar pruebas se diseño un circuito más específico para el nodo de la red.

Se consideran las especificaciones del diseño en la sección 3.1.2 y las recomendaciones recomendacionestécnicas realizadas por Espressif para el diseño de PCB con el ESP32.

Posicionamiento del ESP32 en el PCB

El encapsulado ESP-WROOM-32D está diseñada para ser soldado en una PCB huésped. La PCB de la antena usado en el ESP-WROOM-32D es una antena meandered inverted F (MIFA), para la banda de WiFi 2,4 GHz, con un ganancia de antena de 2dBi. La figura 3.2.7 muestra las seis opciones de colocación que son comunmente 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.

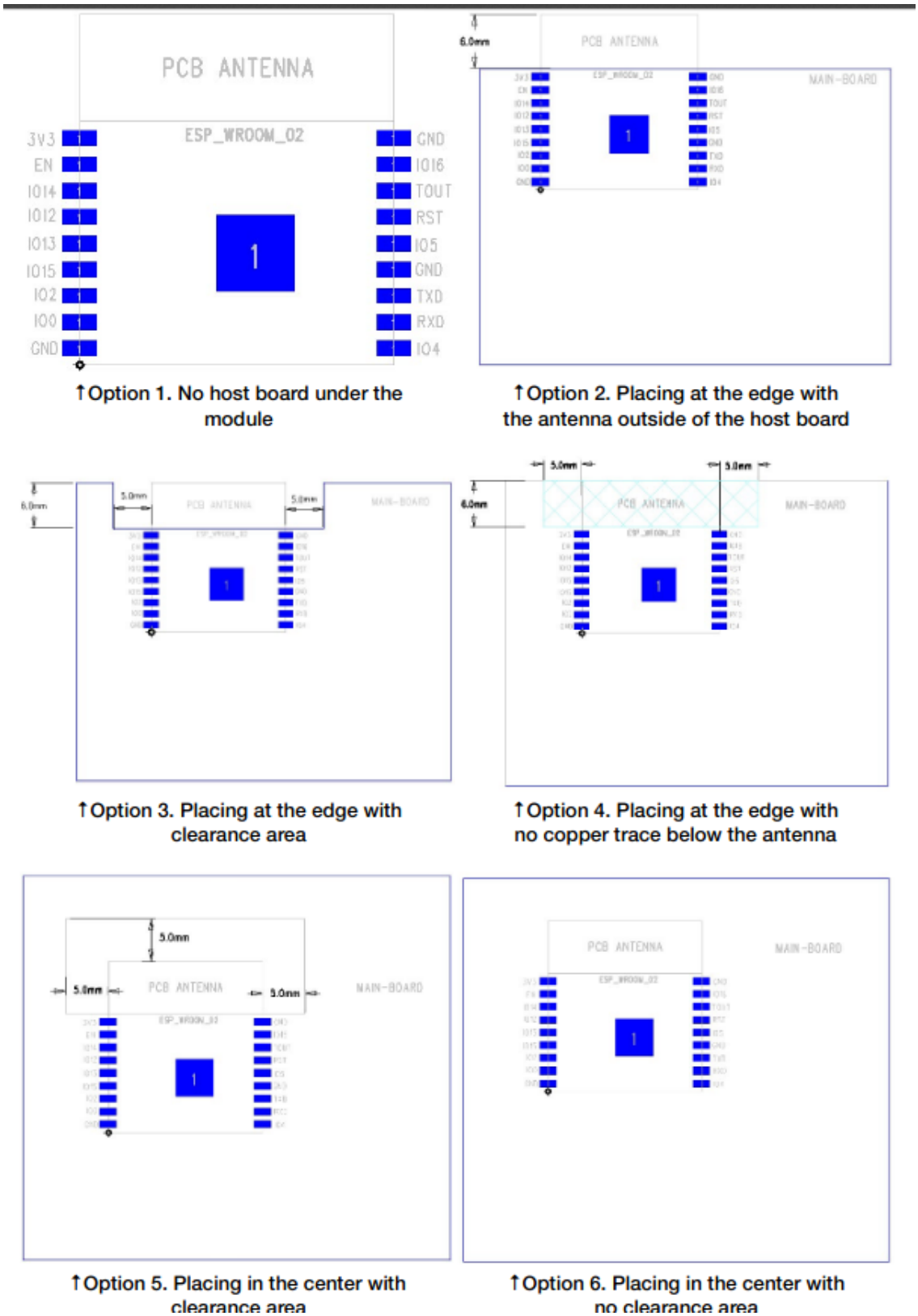

Figura 3.5: Opciones para colocación en PCB

Los resultados del estudio de basan en las mediciones de potencia WiFi y parámetros EVM para varios canales de 802.11 OFDM (MCS1-7) para verificar el rendimiento del la antenas en diferentes posiciones. Los resultados fueron los siguientes:

1. Las opciones de colocación 1,2 y 3 basicamente no se muestran afectadas en el redimiento RF 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 RF.

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

Por lo tanto se tomo la recomentación 2 eligiendo la opción 4 para el diseño.

El chip MAX-485

Espressif provee una ejemplo de conexión para el uso del chip MAX-485, 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.

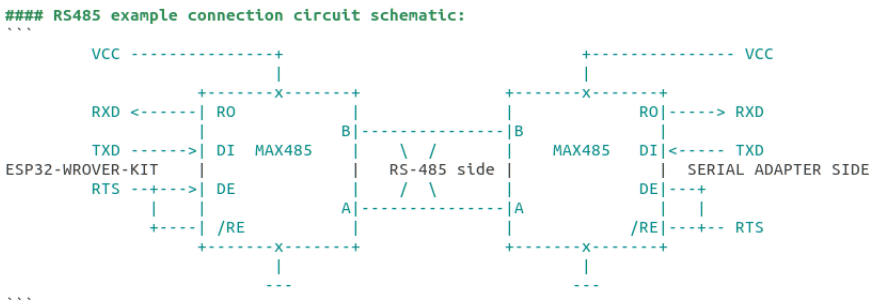

Figura 3.6: Ejemplo de conexión entre MAX-485 y UART

Alimentación

Como se mencionó anteriormente, se usó la fuente LM2596S debido a una potencial rápida implementación, bajo costo y flexibilidad. Sin embargo, la tarjeta posee un tamaño comparable al del nodo, lo que es deteminante en la arquitectura del PCB.

Arquitectura 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 header 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.

Los terminales se agregaron en la parte posterior de la capa de trama para evitar obstruir el espacio libre de la antena WiFi. En la figuras

Adicionalemente se agregó un LED con el propósito de que sirva como testigo de comunicación en el encendido y durante la instalación. La tabla **??** refleja la lista de materiales empleados y sus características.

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

Tabla 3.13: Descripción de la función *espnow_data_prepare*

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, donde aplicaba. Cada tarjeta se programó y configuró de acuerdo a la tabla 3.14. Se usaron un total de cuatro nodos con la topología de la figura 3.2.8. 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 enviar las tramas Modbus generadas por el software.

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 3.14: Configuración de los esclavos para la implementación

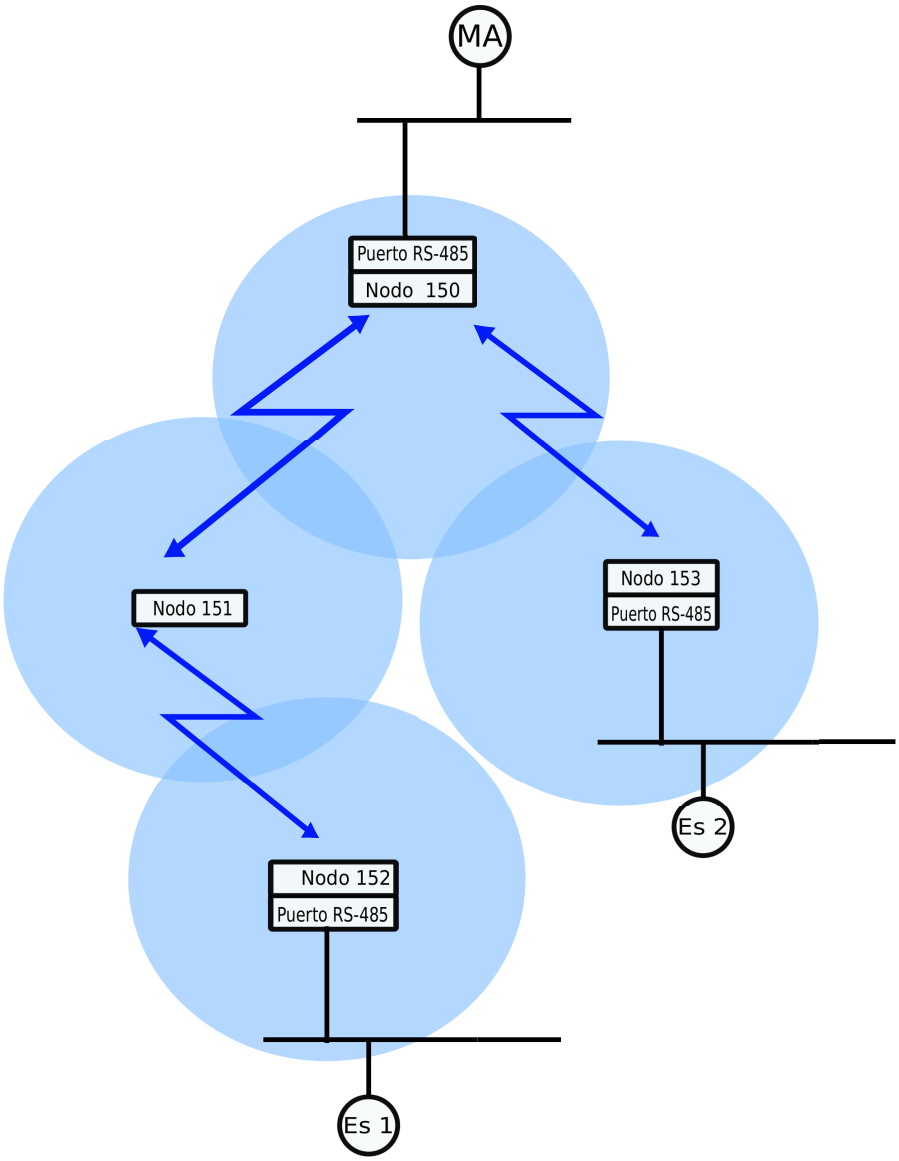

Figura 3.7: Topología de implementación

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 elementos equipos Mosbus usados fueron los descritos en la tabla

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 de la interfaz serial, velocidad de muestreo Modbus, número de variables Modbus. Para la evaluación del rendimiento se variará un parámetro respecto al que tenga mejor desempeño del otro. Así se observa la degradación de la calidad de servicio sin caer en numerosas pruebas redundantes.

Se comenzó con la topología descrita en la figura 3.2.8, y para variaciones en los parámetros Modbus: velocidad de muestreo, número de variables y tasa de baudios.

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	-	0,5
115200	-	0,2

Tabla 3.15: Variación de parámetros en la pruebas de rendiemietno

Se estimará el error en función de la cantidad de excepciones Modbus que se obtengan en el maestro por cada esclavo, considerando la ecuacion 3.1

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

CAPÍTULO IV

DESCRIPCIÓN DEL MODELO

CAPÍTULO V

PRUEBAS EXPERIMENTALES

CAPÍTULO VI

RESULTADOS

CAPÍTULO VII

CONCLUSIONES

CAPÍTULO VIII

RECOMENDACIONES

Apéndice I

TÍTULO DEL ANEXO

Apéndice II

TÍTULO DEL ANEXO

Apéndice III

TÍTULO DEL ANEXO

REFERENCIAS

Al-Husainy, M. (2013, 11). Mac address as a key for data encryption. , 1.

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. En (p. 139-171). 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

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.

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.