Ecosistema del Microcontrolador ESP32 Análisis y Aplicación Práctica en la Simulación de un Sistema para un Cuarto de Control Industrial

(Junio 2024)

Omar A. Torres, Yuly Alvear Romo y Leidy Castaño

***Resumen*—La presente investigación se enfoca en realizar un análisis exhaustivo de la literatura oficial existente del microcontrolador ESP32. El objetivo principal es conocer y entender el microcontrolador ESP32, aprender sobre sus características eléctricas, sus funcionalidades, el mapeo de sus pines, la manera en que se programa, se investigara el IDE que se requiere para su programación. Si bien no es un chip que requiera de un sistema operativo tradicional como el que tiene una computadora personal, si tiene un sistema operativo llamado ROT (Es un "Sistema operativo en tiempo real") que permite la multitarea, pues bien, nos proponemos entender cómo funciona el RTO durante la corrida de un programa. Finalmente, se realizará un programa que gestione de manera remota equipos aplicados a un proceso industrial.**

I. Introducción

El mundo actual experimenta cambios constantes y revoluciones tecnológicas aparentemente interminables. Apenas si se comprende una revolución cuando ya están en marcha otras transformaciones en el ámbito de la ciencia y la tecnología. Estas revoluciones se entrelazan y se complementan para dar lugar a nuevos productos, servicios, tecnologías y avances continuos en la ciencia. La revolución de la electrónica se vincula con el desarrollo del software, y ambas convergen con la inteligencia artificial. En la actualidad, la creciente demanda en el campo de la Internet de las cosas (IoT) refleja la automatización de diversos aspectos de nuestro entorno.

En el ámbito de la IoT y la industria, los microcontroladores, como dispositivos programables, son fundamentales para diseñar sistemas que permitan por ejemplo el control de la temperatura en el hogar, el ahorro de energía eléctrica, la creación de equipos de propósito general para la industria de la automatización, la electromedicina y diversas aplicaciones.

En Colombia, el entendimiento y el uso práctico de los microcontroladores están limitados principalmente al ámbito educativo, durante los procesos de formación académica. Sin embargo, el desarrollo y el uso de los microcontroladores en nuestro país permanecen en gran medida inexplorados, esperando la iniciativa de los emprendedores.

El propósito de este trabajo es establecer un primer contacto con la tecnología electrónica, centrándose particularmente en el microcontrolador ESP32. Se busca elaborar un documento que puedan funcionar como herramienta para el aprendizaje en lo relativo al ecosistema electrónico del ESP32, su sistema operativo y su integración con otros dispositivos. Además, se llevará a cabo una práctica sencilla para programar el chip con el objetivo de simular la operación de un cuarto de control central que permita la gestión de equipos y procesos industriales. La idea central detrás del control es que el dispositivo funcione usando una arquitectura cliente servidor, de tal forma que un operador pueda conectarse remotamente y gestionar los equipos aplicados a un proceso.

Este trabajo adquiere relevancia desde la perspectiva de la carrera de ingeniería de sistemas, ya que proporciona a los ingenieros una visión más amplia y menos genérica del mundo de los sistemas. Implica comprender que el concepto de '1' o '0' lógicos no es simplemente una abstracción, sino que detrás de ello existe un ente real, donde un '1' lógico representa un motor en pleno funcionamiento y un '0' lógico es una máquina apagada.

El estudio del microcontrolador y su dominio abre las

Manuscript received Jun, 9 de 2024. Los materiales para este proyecto fueron adquiridos por los autores, quienes son estudiantes de la Facultad de Ingeniería, Programa de Ingeniería de Sistemas, asignatura de Sistemas Operativos, Universidad de Antioquia.

Omar Alberto Torres, Yuly Alvear Romo y Leidy Castaño, estudiantes de la Facultad de Ingeniería, Universidad de Antioquia, Medellín, Colombia.

puertas a un mundo de oportunidades cercanas y accesibles. Este conocimiento no solo enriquece la formación académica, sino que también proporciona las habilidades necesarias para explorar las diversas posibilidades que ofrece el mundo de la tecnología y la electrónica.

II. MARCO TEÓRICO

A. *Microcontrolador vs Microprocesador*

Las diferencias entre un microcontrolador y un microprocesador se centran en su diseño, funciones y aplicaciones. Un microcontrolador es un dispositivo integrado que combina un procesador central, memoria y periféricos de entrada/salida en un solo chip, diseñado para ejecutar tareas específicas en sistemas embebidos. Por otro lado, un microprocesador es un circuito integrado que incluye solo la unidad central de procesamiento (CPU) y depende de componentes externos como memoria y periféricos. Mientras que los microcontroladores se utilizan en una amplia gama de dispositivos embebidos y sistemas automatizados, los microprocesadores son comunes en computadoras personales y sistemas informáticos que requieren capacidades de procesamiento generales. Además, los microcontroladores tienden a ser más eficientes en cuanto al consumo de energía, lo que los hace adecuados para dispositivos alimentados por batería o sistemas con requisitos de eficiencia energética, mientras que los microprocesadores suelen consumir más energía y se utilizan en sistemas con acceso a una fuente de energía constante, como la red eléctrica.

B. *Microcontrolador ESP32*

El ESP32, desde la perspectiva de la electrónica, se presenta como un microcontrolador integral con un diseño eficiente y potente. Al integrar un procesador dual-core, memoria, módulos WiFi y Bluetooth, así como diversos periféricos, este microcontrolador simplifica el diseño de circuitos y proporciona un rendimiento sólido a 240 MHz. Su La conectividad inalámbrica del ESP32 facilita la creación de equipos que se comunican entre sí y la transferencia de datos sin cables. Por otro lado, su eficiencia energética lo hace idóneo para aplicaciones alimentadas por batería. Con su versatilidad en periféricos, como puertos SPI, I2C y UART, y su compatibilidad con Arduino, el ESP32 se convierte en una opción atractiva para proyectos de IoT. Además, ofrece a los diseñadores electrónicos una herramienta versátil respaldada por una activa comunidad de desarrolladores y un amplio soporte en línea.

Categories	Items	Specifications
Certification	RF certification	FCC/CE-RED/IC/TELEC/KCC/SRRC/NCC
	Wi-Fi certification	Wi-Fi Alliance
	Bluetooth certification	BQB
	Green certification	RoHS/REACH
Test	Reliablity	HTOL/HTSL/uHAST/TCT/ESD
Wi-Fi	Protocols	802.11 b/g/n (802.11n up to 150 Mbps)
		A-MPDU and A-MSDU aggregation and 0.4 μs guard interval support
	Frequency range	2.4 GHz ~ 2.5 GHz
Bluetooth	Protocols	Bluetooth v4.2 BR/EDR and BLE specification
	Radio	NZIF receiver with –97 dBm sensitivity
		Class-1, class-2 and class-3 transmitter
		AFH
Categories	Items	Specifications
	Audio	CVSD and SBC
Hardware	Module interfaces	SD card, UART, SPI, SDIO, I²C, LED PWM, Motor PWM, I²S, IR, pulse counter, GPIO, capacitive touch sensor, ADC, DAC
	On-chip sensor	Hall sensor
	Integrated crystal	40 MHz crystal
	Integrated SPI flash	4 MB
	Operating voltage/Power supply	3.0 V ~ 3.6 V
	Operating current	Average: 80 mA
	Minimum current delivered by power supply	500 mA
	Recommended operating temperature range	–40 °C ~ +85 °C
	Package size	(18.00±0.10) mm × (25.50±0.10) mm × (3.10±0.10) mm
	Moisture sensitivity level (MSL)	Level 3

Fig.1 Especificaciones técnicas del ESP32

C. *Arquitectura del ESP32*

El ESP32-WROOM-32 es un módulo MCU Wi-Fi+BT+BLE potente y versátil diseñado para una amplia variedad de aplicaciones, desde redes de sensores de bajo consumo hasta tareas más exigentes como la codificación de voz, transmisión de música y decodificación de MP3. En el núcleo de este módulo se encuentra el chip ESP32-D0WDQ6, diseñado para ser escalable y adaptable. Cuenta con dos núcleos de CPU que pueden controlarse de forma independiente. La frecuencia del reloj de la CPU es ajustable desde 80 MHz hasta 240 MHz.

El ESP32 integra un conjunto completo de periféricos, que incluyen sensores táctiles capacitivos, sensores Hall, interfaz de tarjeta SD, Ethernet, SPI de alta velocidad, UART, I²S e I²C. Ofrece conectividad Wi-Fi 802.11 b/g/n integrada y también incorpora Bluetooth clásico y Bluetooth de baja energía (BLE), lo que lo hace versátil para diversas aplicaciones de IoT. Además, proporciona una amplia variedad de periféricos, incluyendo GPIO, UART, SPI, I²C y PWM, así como convertidores analógicos a digitales (ADC) para medir señales analógicas. Se incluyen funciones de seguridad, la operación y gestión de la cpu la realiza el sistema operativo en tiempo real (RTOS), como FreeRTOS para multitarea y gestión eficiente de recursos. Diseñado para ser eficiente en términos de consumo de energía, el ESP32 es adecuado para dispositivos alimentados por batería.

La programación se puede realizar en lenguajes como C o C++ utilizando el entorno de desarrollo integrado (IDE) de Arduino o el Espressif IDF (IoT Development Framework).

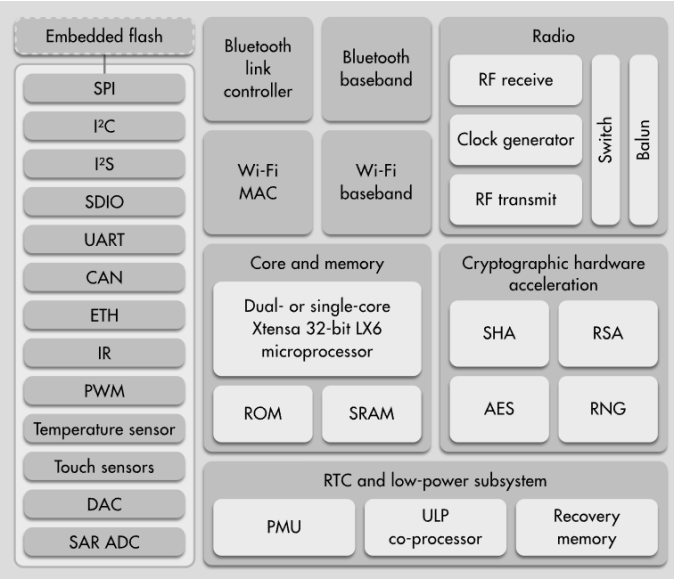

Fig. 2 Arquitectura del ESP32

D. Mapeo de los pines ESP32

El ESP32 utiliza técnicas de mapeo y multiplexado de pines para lograr la multifuncionalidad de sus pines GPIO (General Purpose Input/Output). Esto significa que un mismo pin puede asumir diferentes roles o funciones según la configuración y necesidades específicas del programa. A través del sistema de configuración de pines, se asignan funciones específicas a los pines GPIO, como entrada analógica, salida digital, interfaz I²C, interfaz SPI, UART, entre otras. Este proceso se realiza mediante el ajuste de registros de multiplexación (MUX) asociados a cada pin. En resumen, gracias a estas técnicas de mapeo y multiplexado, un mismo pin en el ESP32 puede adaptarse dinámicamente para realizar distintas tareas, proporcionando flexibilidad y eficiencia en la gestión de recursos en entornos con limitaciones de pines.

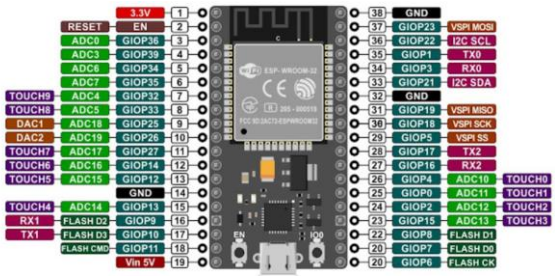

Fig. 3 Mapeo de pines en el ESP32

En cuanto a los pines de entrada y salida del ESP32, se destacan los pines del grupo 34 al 39 como pines de entrada, y del 6 al 11 como pines de salida. Además, el ESP32 dispone de 10 sensores táctiles capacitivos que pueden utilizarse como entradas táctiles.

El ESP32 también ofrece conversores analógico-digital y digital-analógico en varios de sus pines, lo que permite la digitalización y conversión de señales analógicas.

Por otro lado, el ESP32 integra un módulo WiFi que proporciona capacidades de conectividad inalámbrica, así como soporte para el sistema operativo en tiempo real (RTOS) FreeRTOS, que ofrece una gestión eficiente y predecible de tareas en entornos embebidos.

E. Programación del ESP32

Para programar el ESP32, existen varias herramientas de desarrollo que facilitan la tarea de escribir, compilar y cargar código en el microcontrolador. Algunas de las herramientas más comunes para programar el ESP32 son:

Arduino IDE: una herramienta de desarrollo ampliamente utilizada para programar microcontroladores, incluido el ESP32. Puedes programar el ESP32 utilizando la plataforma oficial de ESP32 para Arduino, que proporciona una interfaz fácil de usar y una amplia comunidad de usuarios.

PlatformIO: es una plataforma de desarrollo que se integra con diferentes IDEs, como Visual Studio Code y Atom. Ofrece soporte para una variedad de placas, incluido el ESP32, y proporciona una interfaz más avanzada con funciones de desarrollo más potentes.

Espressif IDF (IoT Development Framework): es el marco de desarrollo oficial de Espressif, el fabricante del ESP32. Es más avanzado que las soluciones de Arduino y proporciona acceso directo a las funciones y características específicas del ESP32. Es adecuado para desarrolladores que buscan un mayor control sobre el código que están desarrollando y el hardware.

MicroPython: es una implementación de Python diseñada para microcontroladores, incluido el ESP32. Permite programar el ESP32 en Python, lo que puede ser beneficioso para aquellos familiarizados con este lenguaje.

NodeMCU/Lua: para aquellos que prefieran el uso de Lua, NodeMCU es una plataforma que permite programar el ESP32 utilizando lenguaje de scripting.

Cada herramienta tiene sus propias ventajas y desventajas, y la elección depende en gran medida de tus preferencias y requisitos de desarrollo. El Arduino IDE es popular para principiantes debido a su simplicidad, mientras que Espressif IDF proporciona un mayor control y acceso a funciones avanzadas para desarrolladores más experimentados. La elección también puede depender de la comunidad de usuarios, la documentación disponible y las características específicas que necesite el proyecto.

F. ESP32 como servidor

El ESP32, gracias a su capacidad de conectividad mediante

el protocolo TCP/IP, puede desempeñarse como un servidor web. Esto significa que puede recibir solicitudes y enviar respuestas a través de la red, lo que lo convierte en una herramienta flexible para aplicaciones de Internet de las Cosas (IoT) y control remoto.

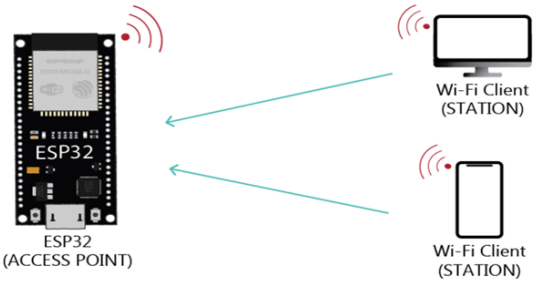

Fig. 4 Conectividad del ESP32

III. Metodología

La metodología seguida para el análisis y simulación de un cuarto de control industrial con el ESP32 incluyó varias etapas claves. En primer lugar, se llevó a cabo una revisión exhaustiva de la documentación disponible, incluidos datasheets y manuales, para comprender la arquitectura y funcionalidades del ESP32. Luego, se procedió a configurar el entorno de desarrollo utilizando el Arduino IDE 2.3.2, lo que permitió el desarrollo del programa para el ESP32. Posteriormente, se realizó el montaje del hardware utilizando una protoboard, donde se colocaron los componentes electrónicos y se probaron los circuitos controlados por el ESP32. La etapa final implicó la programación y o desarrollo del código en HTML para la interfaz gráfica más la lógica para controlar las máquinas de un cuarto de control industrial, escrita para que el dispositivo opere como un servidor.

El paso siguiente fue el desarrollo de las pruebas y ajustes necesarios para garantizar el funcionamiento adecuado del sistema. A renglón seguido, para grabar el programa en la memoria del ESP32, se destacó la necesidad de poner el dispositivo en modo de programación para cargar el nuevo firmware. Una vez en modo de programación, se utilizó el IDE de Arduino para escribir, compilar y cargar el programa. Finalmente se realizaron pruebas exhaustivas del sistema como fué la conexión, como clientes al dispositivo para acceder a la aplicación del cuarto de control Virtual, verificando que se pudiera tener tanto el arranque y parada de equipos deseados, y se realizó pruebas a diferentes distancias del dispositivo.

A. Adquisición Inicial del ESP32

Cuando se adquiere un módulo ESP32 por primera vez, este generalmente llega en su estado predeterminado de fábrica, listo para ser programado según las necesidades específicas. A continuación, se describe cómo suele llegar y la necesidad de ponerlo en modo de programación.

Estado de Fábrica: Al recibir un ESP32 nuevo, este generalmente viene con un firmware de fábrica básico instalado. Este firmware puede variar según el fabricante, pero suele ser lo suficientemente simple como para permitir la programación del dispositivo.

Listo para Programar: El ESP32 no contiene código de usuario específico al principio, lo que significa que está listo para ser programado con el firmware que se desee cargar en él. Esto proporciona flexibilidad para adaptar el ESP32 a las necesidades específicas.

Modo de Programación: para cargar nuevo firmware en el ESP32, generalmente es necesario poner el dispositivo en modo de programación. Este modo permite la comunicación con la herramienta de programación y la carga del nuevo código en el ESP32.

Conexión Específica de Pines: para activar el modo de programación, se conecta o desconecta ciertos pines del ESP32 según la variante específica del módulo. Estos pines suelen incluir GPIO0 y GPIO2, entre otros. La combinación exacta de pines puede variar según el fabricante y el modelo.

Interfaz de Programación: Al poner el ESP32 en modo de programación, este entra en un estado que permite la comunicación con la herramienta de programación a través de interfaces como UART o USB. Este paso es esencial para cargar el firmware de usuario en el ESP32.

La necesidad de poner el ESP32 en modo de programación radica en la capacidad de permitir la carga de nuevo firmware de manera segura y controlada. Al conectar o desconectar ciertos pines, el ESP32 entra en un estado específico que habilita la interfaz de programación y facilita el proceso de carga del código del usuario. Este modo es una característica fundamental que asegura la flexibilidad y la capacidad de personalización del ESP32 para diferentes aplicaciones.

B. Modo programación del ESP32 W-ROOM-32

El procedimiento para poner el microcontrolador ESP32 en estado de programación, es el siguiente:

1. Abrir el IDE de Arduino.
2. Ir a la pestaña tools de la barra de menú y seleccionar la referencia de la tarjeta ESP32 que en nuestro caso fue ESP32 Dev Module.
3. Estando en tools seleccionar el puerto usb en el caso nuestro se seleccionó el puerto COMS 6
4. Compilar el programa a ejecutar.
5. Estando la tarjeta conectada al puerto COMS 6 de la computadora, y energizada, iniciar la carga del programa, El IDEE inicia la compilación al tiempo que genera mensajes en la consola de Arduino, justo cuando sale el mensaje de que se está estableciendo la comunicación, pulsar el botón boot que viene en la tarjeta del ESP32, se mantiene pulsado un instante y se suelta, el programa queda de esta forma cargado con el programa del usuario.

IV. Implementación

La implementación consistió en el desarrollo de un sistema integral que emula un centro de control en un entorno industrial, empleando el ESP32 como dispositivo principal de gestión y una interfaz web básica para la interacción con el sistema mediante un servidor local.

Los equipos sujetos a control comprenden: "Alumbrado", "Horno FC01", "Banda CV03", "Ventilador Fa32 Fondo del Horno", "Bomba PP321", "Bomba de Emergencia PP333", "Alarma Contra incendio" y "Emergencia CCC".

Se establecieron bucles de seguridad para garantizar un funcionamiento seguro del sistema. Por ejemplo, el encendido del horno solo es posible si el ventilador del fondo del horno (FA32) está activado y al menos una de las bombas de agua (PP321 o PP333) está en funcionamiento. Si alguno de estos equipos se apaga, el horno de fundición (FC01) debe detenerse automáticamente para prevenir situaciones de riesgo.

V. Resultados

A. *Interfaz gráfica*

Fig. 5 Interfaz gráfica que muestra el nombre de la máquina con su botón de encendido y apagado.

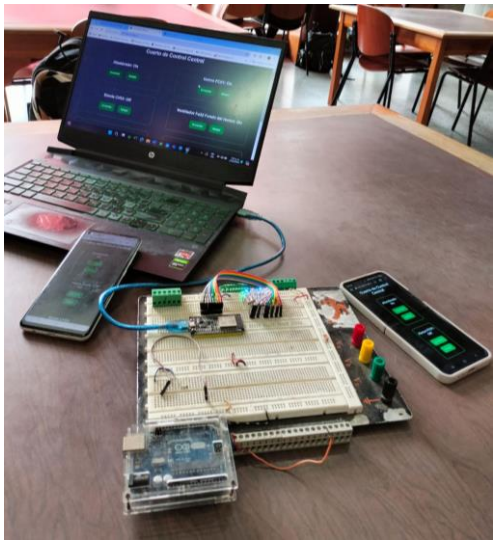

Fig. 6. Montaje en funcionamiento con 3 dispositivos conectados a la misma red wifi.

Los resultados obtenidos muestran que el ESP32 se desempeña eficazmente como un servidor web gracias a su capacidad de conectividad mediante el protocolo TCP/IP. En el ejercicio planteado, se programó el ESP32 para conectarse a una red Wi-Fi privada al ser energizado. Una vez conectado a la red, el microcontrolador está disponible para que los clientes accedan a él utilizando la dirección IP asignada.

Es importante destacar que la programación del ESP32 como servidor web resulta relativamente sencilla, en gran parte debido a la disponibilidad de la librería "<wifi.h>". Esto simplifica considerablemente el proceso de configuración y permite aprovechar rápidamente las capacidades de servidor del ESP32 en aplicaciones prácticas.

VI. Conclusiones

Durante el desarrollo de este trabajo, se lograron exitosamente los objetivos propuestos. Se exploró extensamente la literatura relacionada con el microcontrolador ESP32, proporcionada por los fabricantes, lo que nos permitió adentrarnos en el amplio ecosistema de hardware y software de este dispositivo.

En el proceso, adquirimos un conocimiento detallado sobre los protocolos de comunicación, analizamos las características eléctricas del chip y comprendimos a fondo el mapeo de los pines. Este mapeo nos permitió asignar funciones diversas a un mismo pin, ya sea como entrada, salida, salida PWM o para funciones de comunicación.

Asimismo, nos sumergimos en los elementos fundamentales de la programación del chip, estableciendo una conexión valiosa entre este conocimiento y nuestra formación como ingenieros de sistemas.

Es un logro destacado, implementar con éxito un sistema que simula la operación de un cuarto de control en una empresa industrial. La aplicación diseñada permite que un operador se conecte para monitorear y controlar un proceso.

Este enfoque de simulación encuentra su relevancia en el ámbito de la Internet de las Cosas (IoT), donde la interconexión y la gestión remota son fundamentales.

El logro exitoso de este trabajo fue posible gracias al apoyo y compromiso del equipo, subrayando la importancia de la colaboración en proyectos de esta índole.

En el horizonte de avances potenciales en este proyecto, se sugieren varias direcciones que podrían enriquecer y optimizar la simulación del cuarto de control industrial basada en el microcontrolador ESP32. Considerando el contexto académico y de investigación, las siguientes áreas de exploración podrían ser de interés:

A. Investigación en sensores y monitoreo:

Explorar la integración de sensores adicionales para ampliar la capacidad de monitoreo del entorno industrial simulado. Esto podría incluir investigaciones sobre la selección adecuada de sensores y sus protocolos de comunicación.

B. Desarrollo de protocolos de comunicación específicos:

Investigar y desarrollar protocolos de comunicación específicos para la gestión eficiente de una gama más amplia de dispositivos industriales. Esto podría involucrar la adaptación de estándares existentes o la creación de nuevos enfoques de comunicación.

C. Optimización de interfaces gráficas:

Explorar mejoras en las interfaces gráficas de usuario para aumentar la eficiencia y usabilidad. Esto podría incluir investigaciones en visualización de datos, representación gráfica de procesos y diseño centrado en el usuario.

D. Integración de tecnologías emergentes:

Investigar la viabilidad de incorporar tecnologías emergentes, como Machine Learning o Inteligencia Artificial, para mejorar la capacidad predictiva y adaptativa del sistema. Evaluar cómo estas tecnologías podrían contribuir a la eficiencia operativa y la toma de decisiones.

E. Desarrollo de medidas de seguridad avanzadas:

Investigar y desarrollar medidas de seguridad avanzadas, tales como autenticación multifactor, cifrado robusto y sistemas de monitoreo continuo. Estos aspectos son cruciales en entornos industriales donde la seguridad es una prioridad fundamental.

F. Investigación en estrategias de optimización de energía:

Explorar estrategias de optimización de energía para gestionar eficientemente el consumo de energía de los dispositivos controlados. Investigar posibles fuentes de energía alternativas y su implementación en el sistema.

G. Integración con Plataformas en la nube:

Investigar la viabilidad y los beneficios de la integración con plataformas de nube. Esto permitiría almacenar datos de manera centralizada, facilitando el análisis de tendencias a largo plazo y el acceso remoto a información crítica.

H. Desarrollo de Aplicaciones Móviles Complementarias:

Investigar y desarrollar aplicaciones móviles complementarias que extiendan la capacidad de monitoreo y control a dispositivos móviles. Evaluar la implementación de funcionalidades adicionales para mejorar la flexibilidad operativa.

Estas perspectivas ofrecen un marco para la investigación y desarrollo continuo de este proyecto, proporcionando oportunidades para contribuciones significativas en el ámbito académico y tecnológico.

REFERENCIAS

[1] Carrasco, D. (2021, abril 25). WLAN en ESP32: Primeros pasos con el Wifi. ElectroSoftCloud. https://www.electrosoftcloud.com/wlan-en-esp32-primeros-pasos-con-el-wifi/

[2] Barry, R., & The FreeRTOS Team. (n.d.). Mastering the FreeRTOS™ Real Time Kernel: A Hands-On Tutorial Guide. FreeRTOS. https://freertos.org/Documentation/Mastering-the-FreeRTOS-Real-Time-Kernel.v1.0.pdf

[3] Pinillos, J. M. (2020, mayo 7). Básicos ESP32: Mapeo de pines y sensores internos. Tecnotízate. https://tecnotizate.es/esp32-mapeo-de-pines-y-sensores-internos/

[4] Punto Flotante S.A. (2022). Manual básico NodeMCU ESP32 DevKit V1 Arduino IDE. https://www.puntoflotante.net/MANUAL-BASICO-NODEMCU-ESP32-ARDUINO.pdf

[5] Espressif Systems. (n.d.). ESP-WROOM-32 Datasheet. Retrieved from https://www.alldatasheet.com/view.jsp?Searchword=Esp-wroom-32%20datasheet&gad_source=1&gclid=CjwKCAiAopuvBhBCEiwAm8jaMUrA9d3ZAeRBJZGn3q_aJnhFT3ny3V6c93gDRD1OtHYEOarCMxLaWRoCNPYQAvD_BwE

[6] Electronics. (2021, julio). Especificaciones del módulo ESP32. Vasanza. https://vasanza.blogspot.com/2021/07/especificaciones-del-modulo-esp32.html

[7] Espressif Systems. (s.f.). ESP32. https://www.espressif.com/en/products/socs/esp32