PROYECTO INTEGRADOR

Materia: Proyecto Integrador

Docente: Gonzalo Cristian Vera

Alumno: Raúl Jara

Trabajo Práctico #2
Fundamentos de Programación IoT y Módulos de Desarrollo

Objetivos

1. Entender y aplicar los fundamentos de la programación en Arduino y MicroPython en el contexto de IoT, enfocándose en el manejo de entradas y salidas digitales y analógicas.
2. Desarrollar habilidades prácticas en el uso de módulos de desarrollo y shields para ESP32, explorando diversas aplicaciones de los módulos en proyectos IoT.
3. Implementar controladores básicos utilizando ESP32, que gestionen efectivamente entradas y salidas tanto digitales como analógicas.
4. Familiarizarse con las herramientas de desarrollo como PlatformIO y RT-Thread en Visual Studio Code y utilizar la simulación en Wokwi para validar los programas desarrollados.

Desarrollo

1. Investigación de las plataformas de desarrollo:

• Documentar las características principales de Arduino (PlatformIO) y MicroPython (RT-Thread), destacando sus diferencias y aplicaciones en IoT.

ARDUINO se trata de uno de los tipos de las placas más populares del mundo *maker*, pero que a diferencia de la Raspberry Pi no cuenta con un único modelo, sino que ofrece unas bases de hardware abierto para que otros fabricantes puedan crear sus propias placas.

Qué es Arduino

Arduino es una plataforma de creación de electrónica de código abierto, la cual está basada en hardware y software libre, flexible y fácil de utilizar para los creadores y desarrolladores. Esta plataforma permite crear diferentes tipos de microordenadores de una sola placa a los que la comunidad de creadores puede darles diferentes tipos de uso.

El software libre son los programas informáticos cuyo código es accesible por cualquiera para que quien quiera pueda utilizarlo y modificarlo. Arduino ofrece la plataforma Arduino IDE (Entorno de Desarrollo Integrado), que es un entorno de programación con el que cualquiera puede crear aplicaciones para las placas Arduino, de manera que se les puede dar todo tipo de utilidades.

Arduino con PlatformIO PlatformIO es un entorno de desarrollo integrado (IDE) para Arduino que proporciona soporte para múltiples plataformas de hardware, incluido Arduino.

Algunas de las características principales de Arduino con PlatformIO son:

- Multiplataforma: PlatformIO es compatible con varias plataformas de hardware, lo que permite a los desarrolladores trabajar con una amplia gama de dispositivos Arduino y no solo con los modelos estándar.
- Gestión de bibliotecas avanzada: PlatformIO facilita la gestión de bibliotecas y dependencias, lo que simplifica el proceso de agregar funcionalidades adicionales a los proyectos de Arduino.

- Soporte para múltiples lenguajes de programación: PlatformIO admite varios lenguajes de programación, como C/C++, Python, entre otros, lo que brinda a los desarrolladores una mayor flexibilidad en la elección del lenguaje según sus necesidades y preferencias.
- Integración con otras herramientas: PlatformIO se integra fácilmente con otras herramientas de desarrollo, como Git y sistemas de control de versiones, lo que facilita la colaboración en proyectos y la gestión del código fuente.

PlatformIO

Que es PlatformIO? PlatformIO es una IDE abierta de programación para C/C++, orientado al hardware. ¿Qué es una IDE? Es una sigla y acrónimo en inglés de integrated development environment, en español es un conjunto de herramientas para desarrollo (programación) generalmente agrupadas en una misma interfaz gráfica.

PlatformIO es una plataforma de código abierto para el desarrollo de proyectos de IoT y electrónica, compatible con una variedad de placas de desarrollo, incluido Arduino

MicroPython con RT -Thread

MicroPython es una implementación del lenguaje de programación Python que está diseñada para ser ejecutada en microcontroladores y sistemas embebidos. Está optimizado para funcionar en dispositivos con recursos limitados, como memoria y potencia de procesamiento, lo que lo hace ideal para aplicaciones en el Internet de las Cosas (IoT), robótica y sistemas embebidos en general.

MicroPython con RT-Thread: MicroPython es una implementación de Python 3 optimizada para microcontroladores y sistemas embebidos, mientras que RT-Thread es un sistema operativo en tiempo real de código abierto para dispositivos IoT y sistemas embebidos.

Algunas características principales de MicroPython con RT-Thread son:

• Python en dispositivos embebidos: MicroPython permite a los desarrolladores utilizar el lenguaje de programación Python en dispositivos con recursos limitados, lo que facilita el desarrollo de aplicaciones para IoT sin la necesidad de aprender un lenguaje de programación específico de bajo nivel.

• Soporte para hardware variado: MicroPython es compatible con una amplia gama de microcontroladores y placas de desarrollo, lo que permite a los desarrolladores utilizar Python en una variedad de dispositivos IoT.

• Integración con RT-Thread: RT-Thread proporciona un entorno de ejecución en tiempo real para aplicaciones de MicroPython, lo que permite a los desarrolladores crear aplicaciones IoT que requieran un funcionamiento en tiempo real y una gestión eficiente de los recursos del sistema.

Diferencias y aplicaciones en IoT

Las principales diferencias entre Arduino con PlatformIO y MicroPython con RTThread radican en el lenguaje de programación utilizado y el entorno de ejecución. Mientras que Arduino con PlatformIO se basa en C/C++ y proporciona un entorno de desarrollo integrado específico para Arduino, MicroPython con RT-Thread utiliza Python y se integra con un sistema operativo en tiempo real para aplicaciones IoT.

• Investigar sobre diferentes módulos shield disponibles para ESP32 y su aplicación en controladores IoT.

Shield Tarjeta de Expansion de Pines para Modulo ESP32 es una placa de expansión de entradas y salidas para el Modulo ESP32 de 30 Pines. Esta Shield expande todos los terminales del GPIO, proporcionando comodidad y una rapida conexion entre componentes para agregar funcionalidades adicionales y facilitar el desarrollo de proyectos IoT.

Ejemplos de las mas usadas:

ESP32-CAM: Aplicaciones:

Este módulo integra una cámara y es ideal para aplicaciones de videovigilancia, sistemas de seguridad doméstica, captura de imágenes para análisis de imágenes y reconocimiento facial en proyectos IoT. 2.

ESP32-LCD: Aplicaciones:

Equipado con una pantalla LCD, este módulo es ideal para proyectos que requieren una interfaz de usuario gráfica, como dispositivos de control doméstico inteligente, paneles de control IoT y dispositivos de visualización de datos en tiempo real. 3.

ESP32-LORA: Aplicaciones:

Con soporte para LoRa (Long Range), este módulo es ideal para aplicaciones IoT que requieren comunicación de larga distancia y bajo consumo de Proyecto Integrador • Romero Maria Lourdes energía, como sistemas de telemetría, redes de sensores inalámbricos y dispositivos de seguimiento de activos.

ESP32-ETH: Aplicaciones:

Este módulo agrega conectividad Ethernet al ESP32, lo que lo hace adecuado para aplicaciones IoT que requieren una conexión de red cableada más estable y confiable, como sistemas de automatización industrial, monitoreo remoto y control de dispositivos.

ESP32-Bluetooth: Aplicaciones:

Con soporte para Bluetooth, este módulo es ideal para proyectos IoT que requieren comunicación inalámbrica de corto alcance, como dispositivos de seguimiento de objetos, sistemas de posicionamiento en interiores y aplicaciones de proximidad.

ESP32-GPS: Aplicaciones:

Equipado con un módulo GPS, este módulo es ideal para proyectos IoT que requieren seguimiento de ubicación en tiempo real, como sistemas de navegación vehicular, dispositivos de seguimiento de activos y aplicaciones de geolocalización.

ESP32-Relay: Aplicaciones:

Este módulo agrega relés al ESP32, lo que lo hace adecuado para proyectos IoT que requieren control de dispositivos eléctricos, como sistemas de domótica, automatización industrial y dispositivos de control de acceso.

Estos son solo algunos ejemplos de módulos shield disponibles para ESP32 y sus posibles aplicaciones en controladores IoT. La elección del módulo adecuado dependerá de los requisitos específicos del proyecto, como la conectividad requerida, las funcionalidades adicionales necesarias y el presupuesto disponible.

Codigos fuentes de los controladores desarrollados:

Los códigos de fuentes de controladores desarrollados se refieren al conjunto de instrucciones escritas en un lenguaje de programación específico que se utilizan para crear y controlar los controladores de un dispositivo. Estos códigos son comprensibles para los humanos, lo que permite entender el funcionamiento del controlador y, cuando sea necesario, modificarlo.

El código fuente de un controlador es un elemento fundamental en el

desarrollo de software. Contiene instrucciones y lógica para llevar a cabo tareas específicas. Por ejemplo, si estás desarrollando un controlador para un dispositivo específico, como una tarjeta de red o una impresora, el código fuente de ese controlador contendrá las instrucciones específicas que el dispositivo necesita para funcionar correctamente.

Además, el código fuente de un controlador puede incluir detalles sobre lo que hace un controlador y cómo funciona internamente. Estos detalles permiten a los usuarios realizar pruebas exhaustivas antes del lanzamiento final, garantizando que el controlador sea fiable y seguro.

Es importante mencionar que antes de liberar un controlador basado en un código de ejemplo, se deben realizar cambios importantes en los controladores de ejemplo. Por ejemplo, si el controlador de ejemplo contiene identificadores únicos globales (GUID), nombres simbólicos de vínculos, nombre del objeto de dispositivo, etiquetas de grupo, definiciones de código de control de E/S (IOCTL), entre otros, estos deben ser cambiados para que se apliquen al controlador y al dispositivo

Demostración del funcionamiento de los controladores en wokwi