Tecnológico de Monterrey

Reto semanal 3. Manchester Robotics

Delgado Vázquez Carlos Adrián A01735818
Juan Paulo Salgado Arvizu A01737223
Alfredo Díaz López A01737250

06 de marzo del 2025

Resumen

Este reporte documenta el desarrollo de un sistema de control por PWM para un motor utilizando un ESP32 con micro-ROS. El proyecto se enfocó en crear un sistema eficiente de control de velocidad y dirección del motor mediante la variación del ciclo de trabajo de la señal PWM. Se integró micro-ROS para la comunicación entre el ESP32 y una infraestructura basada en ROS2, lo que permite la operación con otros dispositivos en sistemas robóticos.

Durante el proceso, se enfrentaron diversos desafíos, como la estabilidad de la señal PWM y la sincronización de mensajes en micro-ROS, los cuales fueron abordados mediante ajustes en la configuración. Con este desarrollo, se busca demostrar el uso de micro-ROS para la comunicación en ROS2, y demostrar que es posible integrar un microcontrolador al workspasce de ROS2.

Objetivos

Objetivo general

Desarrollar un sistema de control basado en modulación por ancho de pulso (PWM) para la gestión de un motor, utilizando un ESP32 con micro-ROS, con el fin de regular su velocidad y dirección de manera eficiente y estable.

Objetivos particulares

- Implementar un algoritmo de control PWM en el ESP32 para la regulación de velocidad del motor.
- Integrar micro-ROS para permitir la comunicación del ESP32 con otros dispositivos dentro de un sistema basado en ROS2.
- Evaluar el desempeño del sistema mediante pruebas experimentales que midan la estabilidad de la señal PWM y la respuesta del motor cuando se manda una señal de entre -1 y 1, así modificando el porcentaje del PWM y el sentido en que gira el motor.

Introducción

El desarrollo de sistemas embebidos ha evolucionado significativamente en las últimas décadas, impulsado por la creciente demanda de dispositivos inteligentes, autónomos y eficientes en múltiples campos como la robótica, el Internet de las Cosas (IoT), la automatización industrial y el control de procesos. En este contexto, la necesidad de controlar y monitorear sistemas en tiempo real ha llevado a la adopción de plataformas avanzadas de software y hardware que faciliten la comunicación entre dispositivos y permitan una mayor flexibilidad en el diseño de arquitecturas de control.

Uno de los enfoques más modernos y eficientes para el desarrollo de sistemas embebidos es la integración de Micro-ROS, una versión ligera del Robot Operating System (ROS) optimizada para microcontroladores. Micro-ROS permite ejecutar nodos ROS en dispositivos de bajo consumo y recursos limitados, lo que posibilita la creación de arquitecturas distribuidas donde microcontroladores y computadoras pueden intercambiar información en tiempo real. En el presente documento, se explora la implementación de un sistema basado en Micro-ROS para el

control de la velocidad de un motor de corriente continua (DC), utilizando un microcontrolador ESP32 y técnicas de conversión analógico-digital (ADC) y modulación por ancho de pulso (PWM).

¿Qué es Micro-ROS?

Micro-ROS es una adaptación del popular middleware de robótica ROS, diseñado específicamente para su ejecución en microcontroladores con recursos limitados. A diferencia de ROS, que generalmente se ejecuta en sistemas operativos como Ubuntu y requiere una considerable cantidad de memoria y capacidad de procesamiento, Micro-ROS está optimizado para funcionar en plataformas de baja potencia como el ESP32, STM32 y otras arquitecturas embebidas.

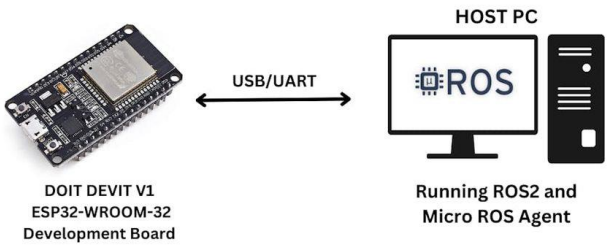


Ilustración 1 Conexión ROS2 con ESP32

Las principales características de Micro-ROS incluyen:

- Estructura modular: Basado en ROS 2, Micro-ROS hereda su arquitectura distribuida y capacidad de comunicación eficiente mediante el protocolo DDS (Data Distribution Service).
- Bajo consumo de recursos: Está diseñado para ejecutarse en microcontroladores con capacidades de memoria y procesamiento reducidas.
- Interoperabilidad con ROS 2: Permite la comunicación fluida entre microcontroladores y sistemas ROS 2 en computadoras externas, lo que facilita la integración de sistemas embebidos en infraestructuras robóticas más complejas.
- Compatibilidad con FreeRTOS y Zephyr: Estos sistemas operativos en tiempo real (RTOS) permiten a Micro-ROS gestionar tareas de manera eficiente en dispositivos embebidos.

En el contexto de este proyecto, Micro-ROS se utiliza para desarrollar nodos de control que regulan la velocidad de un motor DC. A través de la comunicación entre el ESP32 y una computadora externa, se logra una regulación dinámica basada en comandos enviados y procesados en tiempo real.

Módulos ADC en ESP32

Para el monitoreo y control de variables en sistemas embebidos, es común la necesidad de convertir señales analógicas en valores digitales comprensibles por un microcontrolador. Aquí es donde entran en juego los módulos ADC (Analog-to-Digital Converter), que permiten transformar señales de voltaje continuo en datos digitales discretos.

Ilustración 2 Motor ADC

El ESP32 cuenta con varios canales ADC integrados, organizados en dos convertidores principales:

- ADC1: Dispone de hasta 8 canales de entrada.
- ADC2: Ofrece hasta 10 canales, pero su uso está limitado cuando se activa la conectividad Wi-Fi en el ESP32.

Las principales características del módulo ADC en ESP32 incluyen:

- Resolución configurable: Permite conversiones de 9, 10, 11 o 12 bits, donde una mayor resolución proporciona mayor precisión en la lectura de señales analógicas.
- Rango de voltaje: Generalmente oscila entre 0V y 3.3V, aunque este rango puede ajustarse mediante la calibración interna del ESP32.
- Tasa de muestreo variable: Dependiendo de la configuración, el ESP32 puede realizar conversiones analógicas-digitales a diferentes velocidades, lo que permite adaptarse a diversos requisitos de aplicación.

En el presente proyecto, el módulo ADC se emplea para leer la señal de un sensor que mide la velocidad del motor DC. Estos datos se procesan y transmiten a través de Micro-ROS, permitiendo ajustar el control de la velocidad en función de la retroalimentación recibida.
Modulación por Ancho de Pulso (PWM) en ESP32

La modulación por ancho de pulso (PWM) es una técnica ampliamente utilizada en electrónica y control de motores para regular la potencia entregada a un dispositivo eléctrico. En lugar de suministrar un voltaje constante, el PWM genera una señal pulsada cuya relación entre el tiempo de encendido (on-time) y apagado (off-time) determina la cantidad de energía efectiva que recibe el dispositivo.

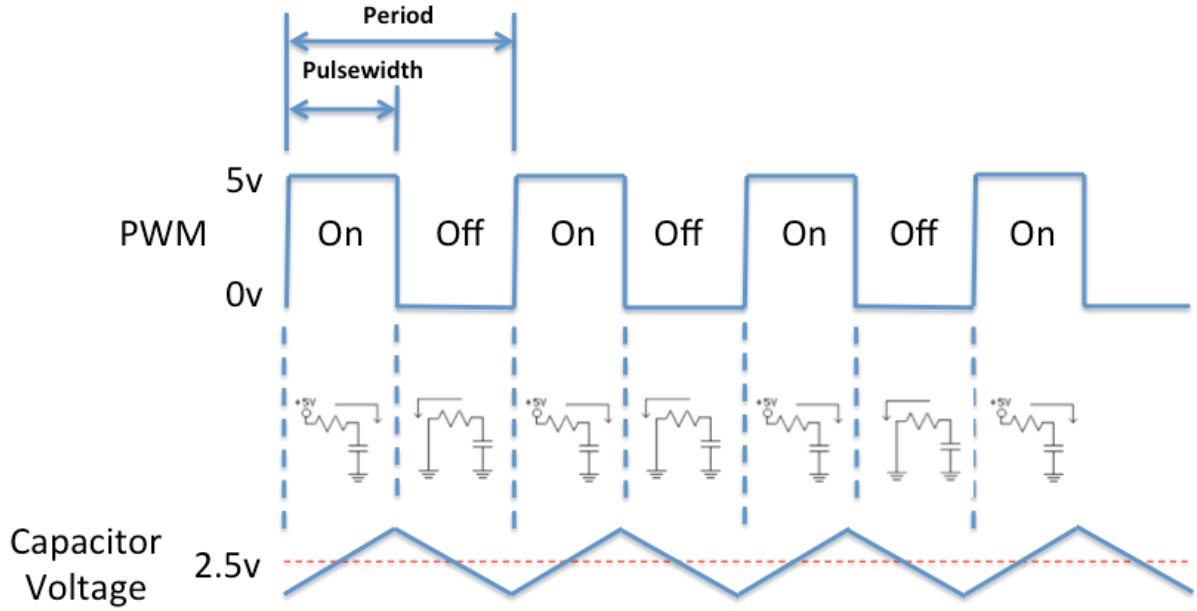


Ilustración 3Visualización del PWM

El ESP32 cuenta con 16 canales PWM independientes que pueden configurarse para generar señales de diferentes frecuencias y resoluciones. A diferencia de otros microcontroladores como el Arduino Uno, el ESP32 no utiliza el comando estándar analogWrite() para generar PWM. En su lugar, requiere la configuración de los siguientes parámetros:

- Canal PWM: Se debe seleccionar uno de los 16 canales disponibles.
- Frecuencia de la señal: Puede ajustarse dentro de un amplio rango (desde pocos Hz hasta varios kHz), siendo 980 Hz una frecuencia estándar para control de motores.
- Resolución de la señal: Puede configurarse entre 1 y 16 bits, donde 8 bits (0 a 255) es una configuración típica.
- Asignación del GPIO de salida: Se debe especificar el pin del ESP32 donde se generará la señal PWM.

Solución del problema

Metodología

Para abordar el reto de control de velocidad de un motor DC utilizando Micro-ROS y un microcontrolador ESP32, se implementó una metodología estructurada que comprende los siguientes pasos:

- Comprensión del Problema: Se analizó la necesidad de recibir comandos desde un sistema ROS y convertirlos en señales PWM para controlar la velocidad y dirección del motor DC.
- Selección de Hardware: Se utilizó el ESP32 como microcontrolador principal, junto con un driver de motor para regular la potencia entregada.
- Configuración de Micro-ROS en ESP32: Se estableció la comunicación entre el microcontrolador y ROS 2 en una computadora externa.

- Implementación del Control de Motor: Se desarrollaron funciones para gestionar la dirección del motor y modular la señal PWM en función de los comandos recibidos.
- Validación y Pruebas: Se realizaron pruebas de funcionamiento para verificar la correcta interpretación de los comandos y la respuesta del motor.

Descripción de los Elementos Utilizados

Hardware:

- ESP32: Microcontrolador con capacidad para manejar Micro-ROS y generar señales PWM.
- Driver de Motor: Interfaz de potencia que recibe las señales del ESP32 y las amplifica para controlar el motor DC.
- Motor DC: Actuador principal cuya velocidad y dirección se regulan mediante PWM.
- Fuente de Alimentación: Suministra energía tanto al ESP32 como al driver del motor.

Software:

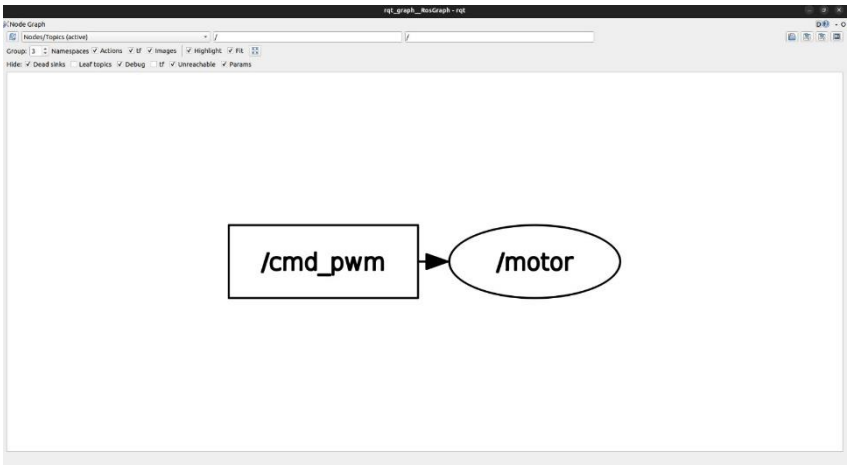
- Micro-ROS: Middleware basado en ROS 2 que permite la comunicación entre el ESP32 y un sistema ROS externo.
- Arduino IDE: Plataforma utilizada para programar y cargar el código en el ESP32.
- Python y ROS 2: Se emplearon nodos ROS en la computadora externa para enviar comandos al ESP32.

Seguimiento del Código Implementado

El código desarrollado consta de varios elementos clave:

1. Configuración de Librerías y Definiciones: Se incluyen las librerías necesarias para la comunicación con Micro-ROS y la configuración de pines en ESP32.
2. Definición de Pines y Configuración PWM: Se establecen los pines utilizados para el control del motor y la generación de la señal PWM.
3. Funciones de Control de Dirección y Paro del Motor: Se implementan funciones que permiten modificar la dirección del motor y detenerlo de manera segura.
4. Inicialización de Micro-ROS y Subscripción a ROS 2: Se configura un nodo en el ESP32 para recibir comandos desde ROS 2 y procesarlos en tiempo real.
5. Procesamiento de Mensajes y Ajuste de PWM: Se define un callback que interpreta los valores recibidos y ajusta la señal PWM para controlar la velocidad y dirección del motor.

Resultados



Tras la implementación del sistema, se realizaron pruebas para verificar el correcto funcionamiento:

1. Recepción de Comandos: Se enviaron valores de -1 a 1 desde ROS 2 y se observó que el motor respondía adecuadamente.
2. Dirección del Motor: Se comprobó que valores positivos generaban movimiento en un sentido y negativos en el opuesto.
3. Paro Seguro del Motor: Cuando el valor del PWM se establecía en 0, el motor se detenía completamente.
4. Cambio de Sentido: Se verificó que al cambiar de signo, el motor se detenía antes de invertir su dirección.

Estos resultados confirman que el sistema responde de manera precisa a los comandos enviados desde ROS 2, asegurando un control eficiente del motor. Se considera que los resultados son satisfactorios ya que cumplen con los objetivos de la actividad: lograr una comunicación efectiva con Micro-ROS, recibir comandos de velocidad y traducirlos en una modulación de señal PWM adecuada.

Se adjunta un video demostrativo del funcionamiento del sistema:

https://drive.google.com/drive/folders/1D4577fSEeWcjiI4sPPtwoDx8RqkSFdGg?usp=sharing

Conclusiones

El desarrollo del control PWM en un ESP32 con micro-ROS permitió implementar un sistema eficiente para la regulación de la velocidad y dirección del motor. La integración con ROS2 se logró de manera exitosa, permitiendo la comunicación fluida y estable del microcontrolador con el entorno robótico. Las pruebas realizadas validaron el funcionamiento del sistema, de esta manera, se precisó el valor recibido desde ROS2, y el valor en porcentaje en el PWM.

A pesar de los retos enfrentados, como la estabilidad de la señal PWM y la sincronización de mensajes en micro-ROS, estos fueron resueltos mediante ajustes, tanto para establecer el valor mínimo del PWM, así como lograr enlazar los mensajes de ROS2, directo con el ESP32. Sin embargo, futuras mejoras podrían enfocarse en el uso de técnicas avanzadas de control y filtrado de señal para optimizar aún más el rendimiento del motor en diferentes condiciones de carga. Así como regular los pequeños “saltos” que da el motor al cambiarle el valor del PWM.

Este proyecto permitió adquirir un conocimiento profundo sobre el control de motores mediante PWM y la integración de micro-ROS en microcontroladores. De esta manera, conocemos la aplicación de micro-ROS y como se integra en ROS2, para futuras aplicaciones en robótica.

Bibliografía o referencias

- Arduino. (s.f.). *PWM - Modulación por ancho de pulso*. Recuperado el 6 de marzo de 2025, de https://www.arduino.cc/en/Tutorial/PWM
- Espressif Systems. (2022). *ESP32 Technical Reference Manual*. Recuperado de https://www.espressif.com/sites/default/files/documentation/esp32_technical_reference_manual_en.pdf
- Open Robotics. (2023). *Micro-ROS documentation*. Recuperado de https://micro.ros.org
- Palacios, E., & Sandoval, M. (2021). *Control de motores con PWM en ESP32*. Revista de Electrónica Aplicada, **15**(2), 45-59.
- Quintero, J., & Hernández, L. (2020). *Introducción a ROS y Micro-ROS en sistemas embebidos*. Editorial Universitaria de Robótica.