UNIVERSIDAD NACIONAL DE INGENIERÍA

FACULTAD DE INGENIERÍA ELÉCTRICA Y ELECTRÓNICA

PROYECTO DE FIN DE CARRERA

"DISEÑO DE UN SISTEMA DE CONTROL DIFUSO AUTOAJUSTABLE EN FPGA PARA LA PROPULSIÓN Y ESTABILIZACIÓN DE UN ROV CON MOTORES BLDC EN AMBIENTES DINÁMICOS"

PARA OPTAR POR EL GRADO DE BACHILLER EN

INGENIERÍA ELECTRÓNICA

ELABORADO POR:

YAFHERS ALONSO MENDOZA CÉSPEDES

ASESOR:

DR. ING. FRANKLIN ALFREDO CABEZAS HUERTA

LIMA - PERÚ

2025

Dedicatoria

A mi madre, Natalia, y a mi hermana, Keony,

por estar siempre ahí, pase lo que pase.

Y a mi padre, Alonso, quien ya no está entre nosotros,

pero desde otro lugar sigue siendo mi fuerza.

Agradecimientos

Expreso mi agradecimiento a todas las personas que contribuyeron
a la realización de este trabajo.

A mi familia, por su apoyo incondicional durante mi
formación universitaria. En especial, a mi madre, Natalia,
cuyo apoyo constante fue fundamental para culminar este proceso.

A mis compañeros de carrera, con quienes compartí proyectos desafiantes,
largas horas de trabajo y momentos inolvidables que hicieron
de este camino uno más enriquecedor.

Al equipo YakuTronix, cuyo trabajo inspiró la concepción de
este proyecto y amplió mi perspectiva profesional.

A mi asesor, por su dedicación en cada corrección
y sugerencia que perfeccionaron este trabajo.

A todos, gracias por acompañarme en esta etapa fundamental de mi vida.

Resumen

Los ROV empleados en ambientes dinámicos enfrentan problemas de estabilidad y eficiencia energética debido a las condiciones variables del entorno. Los sistemas de control convencionales utilizados en ROV presentan limitaciones para adaptarse a cambios abruptos, lo que puede generar errores de posicionamiento, sobrecarga energética y desgaste en los motores BLDC. Además, estos sistemas implementan sus algoritmos en microcontroladores, cuyos límites arquitectónicos introducen latencias críticas al no permitir procesamiento paralelo, lo que afecta negativamente la respuesta dinámica del vehículo.

Esta investigación desarrolla un sistema de control difuso autoajustable, capaz de adaptar dinámicamente sus parámetros de control en tiempo real, según las condiciones del entorno. La arquitectura propuesta aprovecha el procesamiento paralelo de la FPGA para eliminar estas latencias críticas, mientras que la lógica difusa optimiza el rendimiento de los motores BLDC mediante reglas adaptativas. El sistema permitirá el control simultáneo de cuatro motores BLDC, sincronizando su operación con las órdenes de movimiento de la estación terrena y compensando, en tiempo real, las variaciones de inclinación del vehículo.

Palabras Clave: ROV, FPGA, BLDC, Control Difuso, Xilinx, PWM, IMU.

Abstract

ROVs used in dynamic environments face stability problems due to variable environmental conditions. Conventional control systems used in ROVs have limitations in adapting to abrupt changes, which can lead to positioning errors, energy overload, and accelerated wear on BLDC motors. In addition, these systems implement their algorithms in microcontrollers, whose architectural limits introduce critical latencies by not allowing parallel processing, which negatively affects the dynamic response of the vehicle.

This research develops a self-adjusting fuzzy control system capable of dynamically adapting its control parameters in real time according to environmental conditions. The proposed architecture takes advantage of the parallel processing capabilities of the FPGA to eliminate these critical latencies, while the fuzzy logic optimizes the performance of the BLDC motors through adaptive rules. The system will allow the simultaneous control of four BLDC motors, synchronizing their operation with the ground station's motion commands and compensating, in real time, for the vehicle's tilt variations.

Keywords: ROV, Fuzzy Control, FPGA, BLDC, Xilinx, PWM, IMU.

Prefacio

Este trabajo representa el fruto de todo lo aprendido a lo largo de mi formación durante la carrera de Ingeniería Electrónica en la Universidad Nacional de Ingeniería. Mi interés por las FPGA, la programación y el diseño electrónico fue el motor que impulsó la idea de desarrollar este proyecto, centrado en el control preciso del movimiento de un ROV en entornos dinámicos.

La motivación surgió durante mi octavo ciclo universitario, cuando, junto a mis compañeros, fundamos la agrupación YakuTronix con el objetivo de construir un robot submarino desde cero y llevarlo a competir en la competencia MATE ROV, en Michigan. Aunque el proyecto no logró concretarse por falta de financiamiento, decidí continuar con el sueño por mi cuenta, combinando mis conocimientos en robótica y FPGA para dar vida a un proyecto que reflejara esa visión.

A ti, lector, he redactado este documento buscando un equilibrio entre la profundidad técnica y la claridad. Mi propósito es explicar de forma completa y sencilla cada concepto y cada etapa del desarrollo del proyecto. Asimismo, con el objetivo de fomentar la reproducibilidad y el aprendizaje abierto, he incluido en los Apéndices los códigos fuente y protocolos desarrollados.

Índice

Índice de figuras

Índice de tablas

Lista de Acrónimos

ROV Remotely Operated Vehicle

BLDC Brushless DC Motor

FPGA Field Programmable Gate Array

PID Proporcional, Integral, Derivativo

BPNN Backpropagation Neural Network

MCU Microcontroller Unity

DSP Digital Signal Processor

PCB Printed Circuit Board

ESC Electronic Speed Controller

PWM Pulse Width Modulation

MSE Error Cuadrático Medio

PLC Controlador Lógico Programable

SMA Sliding Mode Adaptative

DC Direct Current

AC Alternating Current

BEMF Back Electromotive Force

1. Capítulo I: Antecedentes y Descripción del Problema

1.1. Introducción

En la actualidad, el uso de los Remotely Operated Vehicle (ROV) ha adquirido una gran relevancia en diversas áreas, como la exploración submarina, la inspección de infraestructuras, el monitoreo de ecosistemas marinos y las operaciones de rescate. Los entornos marinos en los que operan estos vehículos suelen estar caracterizados por corrientes y obstáculos impredecibles, lo que exige sistemas de control robustos que aseguren la estabilidad del ROV. Azis et al., 2012

Figura 1

Coxworth, B. (2024, 10 de abril). Cinematic ROV takes filmmaking to new depths. New Atlas. https://newatlas.com/marine/boxfish-luna-rov/

Los ROV utilizan motores Brushless DC Motor (BLDC) como propulsores, distribuidos alrededor de su estructura por su eficiencia, durabilidad y rápida respuesta. Sin embargo, su control en condiciones variables representa un desafío significativo, ya que requieren de señales de alta frecuencia y tiempos de reacción muy bajos, lo que requiere una plataforma de control con gran capacidad de procesamiento. Muchos ROV son implementados utilizando microcontroladores, los cuales presentan limitaciones para ejecutar múltiples tareas en paralelo. Esta restricción puede representar una desventaja frente a la complejidad es incertidumbre del entorno acuático, donde se requiere

procesamiento en tiempo real, alta velocidad de respuesta y elevada confiabilidad.

El control difuso representa una alternativa eficaz para sistemas complejos, ya que maneja adecuadamente la incertidumbre y la no linealidad sin requerir un modelo matemático exacto. Estos controladores se basan en reglas lingüísticas, este enfoque proporciona respuestas suaves y adaptativas ante cambios en el entorno. Actualmente, se aplica en diversos sistemas como cámaras, vehículos, drones y procesos industriales.

Para una aplicación en tiempo real, es esencial implementar el sistema de control en hardware especializado que garantice velocidad y procesamiento paralelo, como las Field Programmable Gate Array (FPGA). Estos dispositivos permiten diseñar circuitos digitales personalizados, brindando mayor rendimiento que los microcontroladores convencionales, lo que los hace ideales para sistemas exigentes como los ROV.

La presente tesis propone el diseño de un sistema de control difuso autoajustable implementado en una FPGA *Artix-7*, orientado a la propulsión y estabilización de un ROV impulsado por motores BLDC, enfrentando las variaciones propias de ambientes dinámicos. Se presentan los resultados obtenidos mediante simulaciones y pruebas experimentales, comparándolos con otros sistemas de control convencionales, lo que demuestra la superioridad del enfoque propuesto en cuanto a capacidad de adaptación.

1.2. Antecedentes

A lo largo de los años, múltiples investigaciones han abordado problemáticas similares a los planteados en este trabajo, aunque desde enfoques y metodologías diferentes.

En los últimos años, los sistemas de control difuso han mostrado grandes avances, especialmente por su adaptabilidad ante condiciones cambiantes. Estudios como el de (H. C. Huang et al., 2019) validan su eficacia en sistemas dinámicos complejos aplicada a vehículos. Sin embargo, su aplicación en la propulsión y estabilización de ROV aún es un campo emergente. Investigaciones como la de (Rehman et al., 2021) mediante microcontroladores, como el uso de Arduino Mega para el control de estabilidad.

En este sentido, el presente proyecto tiene como objetivo optimizar la eficiencia de

los enfoques anteriores mediante la implementación de un sistema de control difuso en FPGA, adaptado específicamente a los requerimientos de los motores BLDC en entornos dinámicos. Diversas aplicaciones han explorado el uso de FPGAs para aplicaciones de control en tiempo real, destacando su capacidad de procesamiento paralelo y alta velocidad de respuesta. Trabajos como el de (Wang et al., 2022) han demostrado que el uso de FPGAs para el control de sistemas, mejora significativamente la velocidad de respuesta y la estabilidad en comparación con otros métodos de control tradicionales.

Por otra parte, los motores BLDC han sido ampliamente investigados y aplicados en la propulsión de ROV debido a su alta eficiencia energética y fiabilidad en entornos marinos. Sin embargo, continúa representando un reto el control preciso de estos motores bajo condiciones dinámicas, propias de medios acuáticos. Investigaciones como la de (Sanchez Rosado, 2016) abordan estrategias de control específicas para estos motores, aunque aún existe una brecha significativa en la implementación de sistemas de control autoajustables que respondan en tiempo real a las variaciones del entorno.

Además, es importante destacar investigaciones como la de (Dong & Duan, 2023), que exploró el uso de técnicas de control alternativas aplicadas a ROV, enfocándose específicamente en el control de profundidad, en lugar del control de movimiento general.

A continuación, se presenta una breve descripción de los trabajos mencionados:

1.2.1. A Design of FPGA-Based Neural Network PID Controller for Motion Control System

En el trabajo de (Wang et al., 2022) se presenta un sistema de control de movimiento basado en un controlador Proporcional, Integral, Derivativo (PID) adaptativo utilizando una Red Neuronal de Retropropagación (Backpropagation Neural Network (BPNN)), implementado en una FPGA Xilinx. El objetivo del estudio es superar las limitaciones de los microcontroladores, que no cumplen con los requisitos de tiempo real en aplicaciones industriales actuales. En lugar de usar un Microcontroller Unity (MCU) convencional, el sistema propuesto se implementa en un FPGA, lo que permite realizar

ajustes adaptativos en tiempo real para los parámetros del controlador PID.

El sistema de control se divide en varios submódulos. Estos submódulos incluyen la propagación hacia adelante de la red neuronal, el módulo PID que mapea las operaciones aritméticas del PID al nivel de transferencia de registros, y un módulo de máquina de estados principal que gestiona la ejecución secuencial de los submódulos. Además, el sistema incluye un módulo para medir la velocidad del motor y otro para generar señales PWM que controlan la velocidad de rotación del motor.

El sistema fue simulado utilizando Modelsim y Simulink para verificar su funcionamiento. Los resultados mostraron que el sistema FPGA propuesto supera a los controladores basados en MCU en términos de velocidad de convergencia, siendo tres órdenes de magnitud más rápido. Además, el sistema demostró tener un alto rendimiento en tiempo real y una gran capacidad de resistencia a interferencias, lo que lo hace más adecuado para aplicaciones exigentes.

1.2.2. Análisis comparativo entre controladores basados en lógica difusa y control PID clásico, aplicados a sistemas de control de velocidad de motores DC sin escobillas

En el trabajo de (Sanchez Rosado, 2016) se compara dos técnicas de control: un controlador basado en lógica difusa y un controlador clásico PID, aplicados al control de velocidad de un motor BLDC. El objetivo principal de la investigación es analizar cuál de estos controladores presenta una mejor respuesta frente a cambios rápidos en la referencia de velocidad o en la carga aplicada al motor. Para ello, el autor desarrolla la teoría de ambos tipos de control y lleva a cabo pruebas experimentales utilizando un sistema de control de motores basado en procesadores Digital Signal Processor (DSP).

Los resultados experimentales demuestran que el controlador difuso proporciona una respuesta más estable y precisa frente a cambios repentinos en la carga y la referencia de velocidad, en comparación con el controlador PID clásico. El controlador PID, aunque efectivo en ciertas condiciones, no se adapta tan bien a las variaciones rápidas en el

sistema. El controlador difuso, en cambio, muestra una mayor capacidad de adaptación a estos cambios, lo que resulta en un mejor rendimiento general del sistema.

1.2.3. FPGA-Based Mechatronic Design and Real-Time Fuzzy Control with Computational Intelligence Optimization for Omni-Mecanum-Wheeled Autonomous Vehicles

En este estudio de (H. C. Huang et al., 2019), se presenta un sistema de control difuso en tiempo real optimizado con inteligencia computacional, aplicado a vehículos autónomos de ruedas Omni-Mecanum. Los autores utilizaron FPGA para implementar un sistema mecatrónico que integra el control difuso, optimizado por un algoritmo de búsqueda evolutiva llamado Cuckoo Search (**CS!** (**CS!**)). Este enfoque permite ajustar dinámicamente los parámetros del controlador difuso, proporcionando un control más adaptativo que los métodos tradicionales.

Los resultados mostraron que el sistema CS-fuzzy superó a los controladores tradicionales, particularmente en términos de rendimiento en tiempo real y capacidad de adaptación. El controlador difuso permitió ajustes en línea de los parámetros de control en tiempo real, lo que aseguró un rendimiento más preciso frente a los cambios dinámicos que podrían ocurrir en el entorno.

1.2.4. A Robust Control via a Fuzzy System with PID for the ROV

Este artículo de (Dong & Duan, 2023) presenta una propuesta de control robusto para el ROV, centrado en el control de profundidad en ambientes submarinos complejos. En este trabajo, se desarrolla un esquema de control utilizando una combinación de control PID con un Control Difuso tipo-2 (IT2FLC), lo que da lugar a un controlador híbrido denominado Control PID Difuso de tipo-2 (IT2FPID).

Para evaluar la efectividad del controlador, se utilizaron indicadores de desempeño que miden la inmunidad del sistema ante perturbaciones externas. Los resultados mostraron que el controlador IT2FPID tiene un sobreimpulso de solo 0.3 % y un tiempo de establecimiento de 7.5 segundos, lo que demuestra una mayor robustez en comparación con

el controlador PID tradicional y el controlador PID difuso de tipo-1 (T1FPID).

1.2.5. Prototyping and Stabilizing of Under-Actuated Remotely Operated Vehicle (ROV) using Fuzzy PID Control Algorithm

El artículo de (Rehman et al., 2021) aborda los desafíos asociados con la estabilización y el control de un ROV. Los ROV son inestables y no lineales, y cuando el grado de libertad del vehículo es mayor que el número de actuadores disponibles, la tarea de estabilizar el vehículo se vuelve aún más compleja.

En este trabajo, se presenta el diseño del controlador de un ROV, con el objetivo de abordar los problemas asociados. El ROV propuesto es eficiente en términos de energía y tiene un tiempo de respuesta rápido. Además, los autores desarrollaron un novedoso algoritmo híbrido Fuzzy-PID para estabilizar el control de inmersión de un ROV submarino con tres propulsores. Los resultados de las simulaciones demostraron la efectividad y validez del algoritmo propuesto.

1.3. Descripción del Problema

La operación de un ROV en ambientes dinámicos presenta diversas dificultades, debido a que las condiciones acuáticas son de naturaleza impredecible. Entre estos factores se incluyen las corrientes submarinas, los cambios de temperatura, las variaciones de presión y la presencia de obstáculos, los cuales pueden generar inestabilidad en la trayectoria del ROV, afectando su maniobrabilidad.

El control de la propulsión en los ROV depende del correcto funcionamiento de sus motores BLDC. No obstante, su manejo en entornos dinámicos presenta varios desafíos. Estos motores requieren señales de alta frecuencia para activar los transistores MOSFET, y las condiciones variables del agua dificultan mantener un control estable. Asimismo, la inestabilidad del medio exige respuestas rápidas ante cambios repentinos.

A pesar de los avances en sistemas de control, muchos ROV siguen utilizando microcontroladores que presentan limitaciones para gestionar múltiples tareas a alta velocidad, lo que provoca retrasos y dificulta mantener la estabilidad en entornos variables.

Estas limitaciones tecnológicas representan un obstáculo para alcanzar un control eficiente en ambientes dinámicos, lo que puede poner en riesgo operaciones críticas como inspecciones submarinas, rescates e investigaciones científicas.

1.4. Formulación del Problema

1.4.1. Problema General

¿Cómo diseñar un sistema de control difuso autoajustable, implementado en FPGA, que permita la propulsión y estabilización eficiente de un ROV impulsado por motores BLDC en ambientes dinámicos?

1.4.2. Problemas Específicos

- ¿Cómo diseñar una PCB para un controlador electrónico de velocidad (ESC) que, mediante la selección adecuada de componentes, permita controlar motores BLDC de una corriente determinada y reciba señales de control generadas por la FPGA?
- ¿Cuáles son las principales perturbaciones y variaciones presentes en ambientes acuáticos dinámicos que afectan el desempeño del ROV?
- ¿Qué estrategias de control difuso permiten una adaptación eficiente a condiciones cambiantes sin requerir un modelo matemático preciso?
- ¿Cómo implementar un sistema de control difuso autoajustable en FPGA, garantizando alta velocidad de procesamiento y ejecución en paralelo?
- ¿Cómo diseñar un sistema de control que optimice el consumo energético de la batería, maximizando el rendimiento de los motores BLDC?
- ¿De qué manera evaluar y comparar el desempeño del sistema diseñado frente a otras estrategias de control convencionales?

1.5. Justificación del Problema

Los ROV son utilizados en tareas de inspección de estructuras submarinas y monitoreo del medio ambiente, donde se requiere una maniobrabilidad precisa. En estas

situaciones, una falla en la estabilidad o en la respuesta del ROV puede provocar la pérdida del equipo, demoras en las operaciones e incluso poner en riesgo vidas humanas. Por eso, es esencial contar con un sistema de control que responda de inmediato y se ajuste automáticamente ante cualquier perturbación, garantizando así la eficiencia de las misiones.

La elección de un sistema de control difuso autoajustable se justifica por su capacidad para adaptarse a entornos inciertos y no lineales sin depender de modelos matemáticos complejos. Por otro lado, el uso de FPGA se justifica debido a su capacidad para ejecutar tareas en paralelo, lo que permite controlar de forma eficiente cada motor BLDC e implementar sistemas con tiempos de respuesta más rápidos. Esta combinación no solo mejora la precisión del control en tiempo real, sino que también optimiza el consumo de energía, un aspecto clave en misiones prolongadas.

1.5.1. Impacto Científico

Esta investigación aborda una brecha al integrar control difuso autoajustable con hardware reconfigurable basado en FPGA para ROV, un tema con escasa documentación pública y que no ha sido abordado previamente en Perú.

Al integrar estos avances en el control difuso, la FPGA y los sistemas de propulsión, este trabajo abre nuevas posibilidades para futuras investigaciones en cualquier sistema que utilice motores BLDC, ofreciendo soluciones innovadoras para enfrentar nuevos retos.

1.5.2. Impacto Social

La tecnología desarrollada en esta tesis puede mejorar la seguridad de comunidades que dependen de actividades marinas como pesca, turismo y conservación. Un ROV con control difuso podrían realizar inspecciones en infraestructuras, reduciendo riesgos en operaciones peligrosas como reparación de plataformas o cables submarinos. Además, su precisión contribuiría a preservar ecosistemas marinos vitales para la biodiversidad del país.

Los ROV también podrían usarse en misiones de rescate, proporcionando una herramienta valiosa para salvar vidas en situaciones de emergencia, como el rescate de personas atrapadas en naufragios o en accidentes en el mar.

1.5.3. Impacto Económico

Se busca ofrecer un ROV con un sistema de control más preciso, sin aumentar significativamente su costo frente a modelos comerciales. La adopción de ROV mejorados podría abrir nuevas oportunidades para los sectores energético, minero y pesquero. Así, Perú podría posicionarse en el creciente mercado global de robótica marina, desarrollando servicios tecnológicos innovadores y generando nuevas fuentes de ingreso.

Además, este proyecto fomentaría la creación de centros de investigación y fortalecería la formación de talento nacional en áreas estratégicas como robótica, control automático y FPGA. Esto impulsaría el avance científico-tecnológico y el crecimiento económico mediante la generación de empleo. La adopción de tecnologías avanzadas permitiría reducir costos operativos en minería submarina, energías renovables marinas e investigación científica, promoviendo una gestión más sostenible de los recursos naturales.

1.6. Objetivos

1.6.1. Objetivo General

Desarrollar un sistema de control difuso autoajustable para la propulsión y estabilización de un ROV impulsado por motores BLDC, utilizando una FPGA que garantice el procesamiento en tiempo real de cada motor, permitiendo su operación eficiente en ambientes marinos dinámicos.

1.6.2. Objetivos Específicos

- Diseñar una Printed Circuit Board (PCB) capaz de controlar motores BLDC de corriente específica, integrando señales de control generadas por la FPGA.
- Seleccionar un giroscopio adecuado e integrarlo en la FPGA mediante programación en VHDL, obteniendo datos precisos de movimiento.
- Desarrollar el control del Electronic Speed Controller (ESC) mediante señales Pulse Width Modulation (PWM) generadas desde la FPGA para regular la velocidad de los motores.

- Modelar las principales perturbaciones presentes en entornos acuáticos, como corrientes marinas y variaciones de temperatura.
- Diseñar un sistema de control difuso en Simulink, basado en reglas predefinidas, para simular y ajustar el comportamiento del ROV ante distintos escenarios.
- Implementar el sistema de control difuso autoajustable en FPGA, asegurando su adaptación a condiciones variables sin requerir un modelo matemático preciso.
- Optimizar el consumo energético del sistema, maximizando el rendimiento de los motores BLDC con el menor uso de energía posible.
- Evaluar el desempeño del sistema de control frente a estrategias convencionales y comparar su eficacia en diferentes condiciones operativas.
- Desarrollar una interfaz gráfica de usuario que permita visualizar datos del ROV y controlar su movimiento de forma intuitiva en tiempo real.

1.7. Hipótesis

1.7.1. Hipótesis Principal

Si se implementa un sistema de control difuso autoajustable en FPGA, junto con la optimización del consumo energético y la integración de sensores adecuados, se mejorará la estabilidad, maniobrabilidad y eficiencia energética de un ROV en ambientes marinos dinámicos, superando las limitaciones de los sistemas de control convencionales basados en microcontroladores.

1.7.2. Hipótesis Específicas

- Si se seleccionan componentes electrónicos adecuados según la corriente nominal de los motores BLDC e integran correctamente en el diseño de la PCB, será posible construir un ESC que gestione eficientemente la potencia requerida y reciba señales desde la FPGA sin fallos de comunicación ni sobrecalentamiento.

- Si se identifican y clasifican correctamente perturbaciones como corrientes, oleaje, presión y temperatura, será posible diseñar estrategias de control que mejoren la estabilidad y el rendimiento del ROV en entornos acuáticos dinámicos.

- Si se implementan estrategias de control difuso con reglas adaptativas y sistemas de inferencia flexibles, se logrará una adaptación eficiente a condiciones variables sin requerir un modelo matemático preciso del entorno o del ROV.

- Si se estructura el control difuso autoajustable utilizando arquitectura paralela dentro de la FPGA, entonces se podrá garantizar alta velocidad de procesamiento y ejecución en paralelo, mejorando la capacidad de respuesta del sistema en tiempo real.

- Si se optimiza el diseño del sistema de control y propulsión, tanto a nivel de lógica en FPGA como en la gestión dinámica de los motores BLDC, entonces se reducirá el consumo energético, permitiendo un mayor tiempo de operación del ROV.

- Si se emplean métricas de desempeño como tiempo de respuesta, consumo energético y estabilidad bajo perturbaciones para comparar el sistema de control difuso autoajustable frente a estrategias convencionales, entonces se podrá demostrar cuantitativamente las mejoras en eficiencia y robustez del sistema diseñado.

1.8. Variables

1.8.1. Variables Independientes

- **Tipo de controlador difuso**: El diseño del controlador difuso, con sus reglas y parámetros, es fundamental para que el ROV pueda adaptarse a los cambios del entorno sin perder estabilidad.

- **Plataforma de implementación**: La elección entre usar una FPGA o un microcontrolador afecta directamente la velocidad de respuesta del sistema, siendo la FPGA ideal para procesamiento en paralelo.

- **Ángulos de Inclinación del ROV**: Los datos del giroscopio, que miden la inclinación del vehículo en sus tres ejes, son cruciales para que el sistema de control pueda corregir su posición.
- **Perturbaciones ambientales**: Factores como corrientes marinas, cambios de temperatura y presión son variables externas que desafían constantemente la estabilidad del ROV.
- **Parámetros energéticos**: El voltaje de alimentación y la frecuencia de las señales PWM determinan qué tan eficientemente funcionan los motores y cuánta energía consume el sistema.

1.8.2. Variables Dependientes

- **Estabilidad del ROV**: Indica qué tan bien mantiene el vehículo su posición y orientación a pesar de las perturbaciones. Se mide por el error de movimiento y la capacidad de recuperación ante cambios bruscos.
- **Eficiencia Energética**: Refleja cuánta energía consume el sistema para mantener el ROV en operación. Un buen control optimiza el uso de la batería, prolongando su autonomía.
- **Tiempo de Respuesta**: Es la rapidez con la que el sistema reacciona a cambios en el entorno o en los comandos. Una FPGA reduce este tiempo, mejorando el control en tiempo real.
- **Desempeño de motores BLDC**: Evalúa qué tan bien funcionan los motores bajo diferentes condiciones, incluyendo su velocidad, precisión y resistencia al desgaste por uso continuo.

1.9. Indicadores

- El indicador para demostrar la primera hipótesis será la corriente máxima soportada por la PCB, la temperatura en operación continua y el porcentaje de errores en la

recepción de señales PWM desde la FPGA.

- El indicador para demostrar la segunda hipótesis será la cuantificación de perturbaciones (corrientes, oleaje, presión y temperatura) y su impacto en la estabilidad del ROV.
- El indicador para demostrar la tercera hipótesis será el número de reglas difusas, el tiempo de adaptación ante cambios y el error medio cuadrático (Error Cuadrático Medio (MSE)) del sistema.
- El indicador para demostrar la cuarta hipótesis será la latencia por ciclo de control, el nivel de paralelismo y el uso de recursos lógicos en la FPGA.
- El indicador para demostrar la quinta hipótesis será el consumo promedio de corriente, la relación consumo/desplazamiento y la reducción porcentual frente a un sistema convencional.
- El indicador para demostrar la sexta hipótesis será el tiempo de respuesta, el consumo energético global, la estabilidad y el error de posición comparado con métodos tradicionales.

2. Capítulo II: Marco Teórico y Conceptual

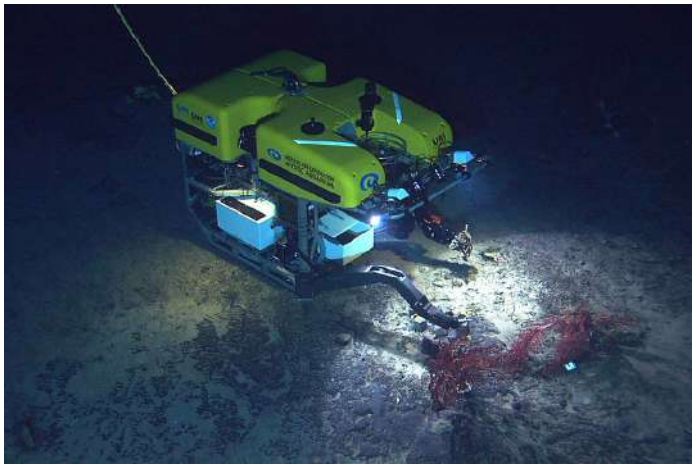
2.1. Vehículos Operados Remotamente

2.1.1. Introducción a los ROV

Un ROV (Vehículo Operado Remotamente) es un vehículo submarino controlado desde la superficie por uno o varios operadores, como se muestra en la figura 2. Estos pilotos suelen manejar el ROV desde una embarcación o estación (figura 3), a la que el vehículo está conectado mediante un cable conocido como *tether*.

Figura 2

Un ROV recolectando muestras con 2 manipuladores



Expedición del ROV Hercules, 2004

El *tether* suministra todas las señales necesarias para la alimentación y la transmisión de datos. Este cable contiene en su interior varios conductores de diferentes calibres, cada uno con sus respectivas protecciones e impermeabilidad.

Los ROV están diseñados para una amplia gama de tareas, incluyendo investigaciones submarinas, reparaciones, inspección de estructuras y recolección de datos. Emplear un ROV en misiones submarinas ofrece múltiples beneficios en comparación con el uso de buzos humanos. Entre estas ventajas, destaca su mayor seguridad y la capacidad de acceder a zonas profundas o de difícil alcance para los humanos. Además, un ROV puede

permanecer sumergido por periodos largos que un buzo, lo que facilita la ejecución de tareas complejas.

Figura 3

Sala de Control de un ROV

Tomado de MBARI y Monterey Bay Aquarium, 2024

2.1.2. Instrumentación de los ROV

La mayoría de los ROV incluyen cámaras y luces para capturar y transmitir imágenes y videos al operador en tiempo real. Algunos modelos incorporan equipos adicionales, como sensores para medir la claridad del agua, la temperatura u otros parámetros químicos y físicos. Además, pueden estar equipados con brazos robóticos (manipuladores) que permiten operar herramientas o recuperar objetos.

Los ROV varían en forma y tamaño según su aplicación, pero comparten componentes básicos esenciales. Esta sección describe las partes principales de un sistema ROV típico. La Figura 4 muestra un esquema general de estos elementos. A continuación, se detallan sus componentes principales:

- **Estructura**: Es el chasis que aloja todos los componentes. Generalmente se fabrica con materiales resistentes a la corrosión y capaces de soportar las altas presiones del

Figura 4

Partes Principales de un ROV

Tomado de Blue Robotics

agua. Entre los materiales más comunes utilizados en ROV se encuentran los materiales mostrados en la tabla 1.

Cuadro 1

Materiales utilizados en chasis de ROV

Material	Densidad	Profundidad	Desventaja
HDPE / Polímeros	0.95 g/cm^3	0-300 m	Se deforma bajo presión
Aluminio	2.70 g/cm^3	0-1000 m	Se corroe aceleradamente
Fibra de carbono	1.60 g/cm^3	0-2000 m	Frágil ante golpes
Acero inoxidable	8.00 g/cm^3	>4000 m	Muy pesado
Titanio	4.50 g/cm^3	>6000 m	Demasiado caro

- **Propulsores**: Son los motores que permiten el movimiento del ROV en el agua. Estos motores están conectados a hélices, las cuales posibilitan el control de la dirección y velocidad del vehículo. En cuanto a los motores empleados en los ROV comerciales, se han investigado los tipos mostrados en la tabla 2.

Cuadro 2

Motores utilizados en los propulsores de ROV

Tipo de Motor	Ventaja	Desventaja
Motor BLDC	Larga vida útil	Requiere controlador ESC
Motor Direct Current (DC)	Coste bajo	Se desgatan por sus escobillas
Motor Alternating Current (AC)	Alta Potencia	Son muy grandes y pesados
Motor Hidráulico	Ideales para > 4000 m	Son costosos y requieren mantenimiento

Para los ROV comerciales, el tipo de motor más utilizado es el motor BLDC. Es por ello que esta tesis se enfoca en el estudio de técnicas de control para este tipo de motor. La figura 5 muestra un vehículo empleando propulsores T200 de *Blue Robotics*, que son los motores BLDC más comunes en ROV comerciales.

Figura 5

ROV usando una configuración de 8 propulsores T200

Extraído del equipo Sea Robotics

- **Sensores**: Varían según la aplicación del ROV, pero los más comunes incluyen:
  - Profundidad/presión: Estos sensores calculan la profundidad exacta y aseguran que el vehículo opere dentro de sus límites estructurales.
  - Orientación (IMU): Integran giroscopios, acelerómetros y magnetómetros para

determinar la orientación y el movimiento del ROV. Esta información es vital para la navegación desde la estación terrestre.

- Parámetros del agua (pH, oxígeno, turbidez, conductividad): Miden el pH, el oxígeno disuelto, la turbidez y la conductividad del agua. Estos datos son fundamentales para estudios oceanográficos o para diversas misiones del ROV.
- Geolocalización (GPS, USBL): Calculan la ubicación relativa del ROV con respecto a la estación terrestre. Esto es crucial para operaciones de recuperación o para mantener la posición frente a corrientes fuertes.
- Sonar: Permiten mapear el lecho marino y detectar obstáculos. Funcionan emitiendo pulsos acústicos que rebotan en los objetos y regresan al sensor, creando imágenes detalladas del entorno submarino.

- **Cámaras**: Las cámaras permiten la visualización en tiempo real, facilitando la exploración o la manipulación de objetos. Estas imágenes son capturadas por la computadora a bordo y enviadas a la estación terrestre para que el piloto pueda navegar el ROV.

- **Manipuladores**: Estos son los brazos robóticos, equipados con distintos grados de libertad, que se utilizan para tareas como la recolección de muestras o el mantenimiento submarino. Existen modelos de ROV con uno, dos o más brazos, aunque no todos los ROV están necesariamente equipados con manipuladores.

- **Tether**: Es el cable que conecta el ROV a la estación de control. A través de este recubrimiento pasan los cables de energía, comunicación y datos de los sensores.

- **Sistema de Iluminación**: Son las luces que mejoran la visibilidad y el posicionamiento del ROV. Algunos modelos incorporan láseres para fijar un punto de referencia y optimizar su posición.

Figura 6

ROV Defender recuperando una caja metálica con muestras

Extraído de Reach Robotics

- **Sistema de control**: Es el cerebro del ROV. Este sistema gestiona los propulsores, sensores, manipuladores y la comunicación. Utiliza microcontroladores o computadoras a bordo que interactúan internamente en el ROV y establecen una comunicación bidireccional con la estación terrestre.

  En cuanto al sistema de control, los ROV emplean diferentes tipos de controladores según su complejidad y aplicación. Para modelos pequeños, los sistemas basados en Arduino y Raspberry Pi son los más comunes, debido a su bajo costo y facilidad de configuración. En aplicaciones industriales, se opta por controladores más robustos como los Controlador Lógico Programable (PLC), capaces de soportar grandes profundidades y condiciones extremas. Para tareas avanzadas que demandan procesamiento de imágenes o inteligencia artificial, se utilizan computadoras embebidas como la NVIDIA Jetson Nano. La tabla 3 presenta algunos modelos comerciales de ROV junto con los controladores que incorporan:

Cuadro 3

Controladores usados en ROV comerciales

ROV Comercial	Controlador	Aplicación
BlueROV 2	Arduino Mega2560 + Raspberry Pi4	Educación, investigación
Deep Trekker DTG3	Arduino Nano + Raspberry Pi4	Monitoreo Ambiental
OpenROV Nemo	ESP32-WROOM-32	Fotografía Submarina
Oceanbotics SRV-8	ARM Cortex-M4	Inspección de presas
Saab Seaeye Falcon	PLC Siemens S7-1200	Rescate Submarino
SeaBotix LBV300	PIC32	Arqueología Marina
TechnipFMC ROV	PLC Allen Bradley	Mantenimiento Marino
VideoRay Pro 5	ARM Cortex + NVIDIA JetsonTX2	Inspección con IA
Schilling UHD III	FPGA Xilinx Zynq UltraScale	Inspección petrolera
Saab Seaeye Leopard	FPGA Lattice ECP5	Rescate Submarino

2.1.3. Problemática de los ROV

Según la investigación de (Azis et al., 2012), se han identificado diversas problemáticas en el diseño y operación de un ROV. Los problemas detectados se presentan a continuación:

1. **Problema en el Sistema de Control**: El desarrollo de un sistema de control para ROV es complejo por la naturaleza no lineal del entorno marino, sus perturbaciones y la dificultad de modelar con precisión los parámetros hidrodinámicos. Aunque se han propuesto diversos métodos de control (tabla 4), muchos aún presentan deficiencias.

   Después de varios ensayos, se han explorado alternativas como el control adaptativo, que ajusta la respuesta del sistema a cambios en la dinámica del vehículo, o combinaciones híbridas de métodos de control, que han demostrado mejorar la robustez y la tolerancia a fallos. Actualmente, la estrategia más efectiva para controlar ROV es la combinación de sistemas de control, incluyendo el neuro-difuso

Cuadro 4

Limitaciones de los métodos de control en ROV

Método de Control	**Limitaciones**
PID	No puede adaptarse bien a fuerzas inesperadas del agua porque necesita un modelo muy preciso para funcionar correctamente.
Sliding Mode	Podría fácilmente provocar vibraciones del sistema y afectar la precisión del control.
Difuso	Es difícil ajustar correctamente las reglas difusas.
Redes Neuronales	No son lo suficientemente rápidas para responder en tiempo real; requiere tiempo de procesamiento.

(redes neuronales + lógica difusa) y el Sliding Mode Adaptativo. Sobre estos controles, se ha observado lo siguiente:

- Control Neuro-Difuso: Este control fusiona la capacidad de aprendizaje de las redes neuronales con la lógica difusa basada en reglas. Es ideal para sistemas no lineales que necesitan aprender y mejorar con el tiempo. Sin embargo, su ajuste de parámetros implica un alto costo computacional y es más complejo de implementar.
- Control Sliding Mode Adaptativo (Sliding Mode Adaptative (SMA)): Este control combina la robustez del control *Sliding Mode* ante incertidumbres con mecanismos adaptativos que ajustan los parámetros en tiempo real, lo que le confiere una alta estabilidad en condiciones extremas. Además, su diseño es más sencillo que el del control neuro-difuso. No obstante, sus desventajas incluyen la posible generación de vibraciones (*chattering*) si no se filtra adecuadamente, y la necesidad de una sintonización precisa, lo cual puede representar un desafío en plataformas con recursos computacionales limitados, como microcontroladores.

Dado que la implementación del sistema de control de esta tesis se realizará en una FPGA, se opta por un enfoque cercano al controlador Neuro-Difuso. Esta elección se debe a su notable capacidad de aprendizaje, adaptación y toma de decisiones en entornos no lineales y con incertidumbre. Aunque este tipo de controlador demanda mayor potencia computacional que otras estrategias como el Sliding Mode Adaptativo, la arquitectura paralela de la FPGA permite su ejecución eficiente en tiempo real, maximizando así el aprovechamiento de los recursos de hardware disponibles.

2. **Problema de Condición de Subactuación**: Un ROV se considera subactuado cuando tiene menos propulsores que grados de libertad, lo que dificulta mantener una posición o profundidad precisa. Esto implica que los propulsores restantes deben compensar cualquier fallo, lo que podría sobrecargar el sistema. Para mitigar esto, se sugiere incorporar un encoder a cada propulsor, permitiendo un control preciso de la velocidad y compensación en caso de fallos

3. **Problema de Estabilidad por Tether**: La interacción del *tether* con el cuerpo del ROV afecta su estabilidad, a pesar de ser crucial para la transferencia de energía y comunicaciones. El diseño del cable debe considerar su tamaño, peso, profundidad de operación y los motores del ROV. Dada su influencia en la dinámica del vehículo, se busca reducir su tamaño para optimizar el rendimiento.

4. **Problema de Comunicación**: La comunicación inalámbrica de video y monitoreo en ROV es un desafío subacuático. Las altas frecuencias tienen alcance limitado; el GPS, por ejemplo, solo funciona a menos de 8 cm de profundidad, y la calidad de la señal depende de la SNR y los satélites.

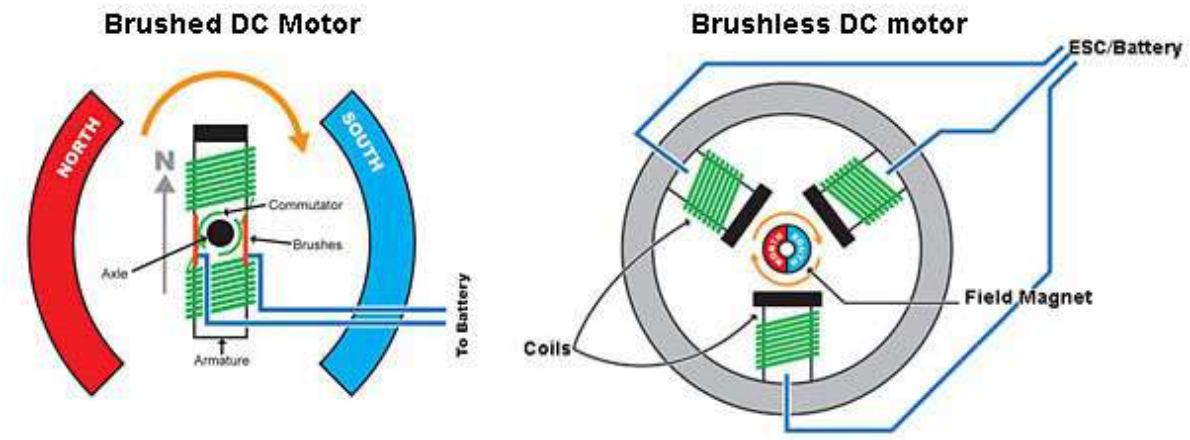
2.2. Motores Brushless DC (BLDC)

2.2.1. Definición de motor BLDC

Un motor *brushless* es un motor eléctrico de corriente continua que, a diferencia de los motores DC tradicionales, no utiliza escobillas (*brushes*) para conmutar el campo magnético. En su lugar, un controlador electrónico conmuta el flujo de corriente de las bobinas, creando electroimanes que hacen girar el rotor.

Figura 7

Diferencia entre motor DC y BLDC



Extraído de Battery Things

Los motores BLDC se emplean en diversas aplicaciones, como vehículos eléctricos, drones y robots, que demandan un control preciso de la velocidad y el par motor.

También se encuentran en electrodomésticos como ventiladores o aspiradoras, gracias a su capacidad de operar a altas velocidades con baja generación de calor y ruido. En el ámbito industrial, los motores BLDC se utilizan en servomecanismos, maquinaria CNC y sistemas de automatización, donde se requiere fiabilidad y una larga vida útil.

En la siguiente tabla 6 se presenta una comparación entre un motor DC sin escobillas y un motor DC con escobillas:

Cuadro 5

Comparación entre motores DC y motores BLDC

Motores DC	**Motores BLDC**
Usan una conmutación mecánica con escobillas.	Usan una conmutación electrónica sin escobillas.
Se desgastan más rápido debido a la fricción magnética de las escobillas.	Tienen mayor vida útil debido a la ausencia de fricción mecánica.
Requieren mantenimiento periódico para cambiar o limpiar las escobilla.	Requieren poco mantenimiento gracias a su diseño sin escobillas.
Su velocidad es limitada debido a la fricción de las escobillas	Pueden alcanzar velocidades más grandes debido a no tener escobillas.
Operan con mayor ruido eléctrico y mecánico.	Operan con menor ruido, siendo ideales para aplicaciones de precisión.
Su eficiencia energética es menor debido a las pérdidas por fricción.	Su eficiencia energética es mayor al evitar pérdidas por fricción.
El control es más simple y económico, pero menos preciso.	Requieren un ESC para su control obligatoriamente.
Su precio es económico,	Su precio es elevado.
Son más grandes y pesados para una misma potencia.	Ofrecen mayor potencia en menor tamaño y peso.

2.2.2. Componentes de un motor BLDC

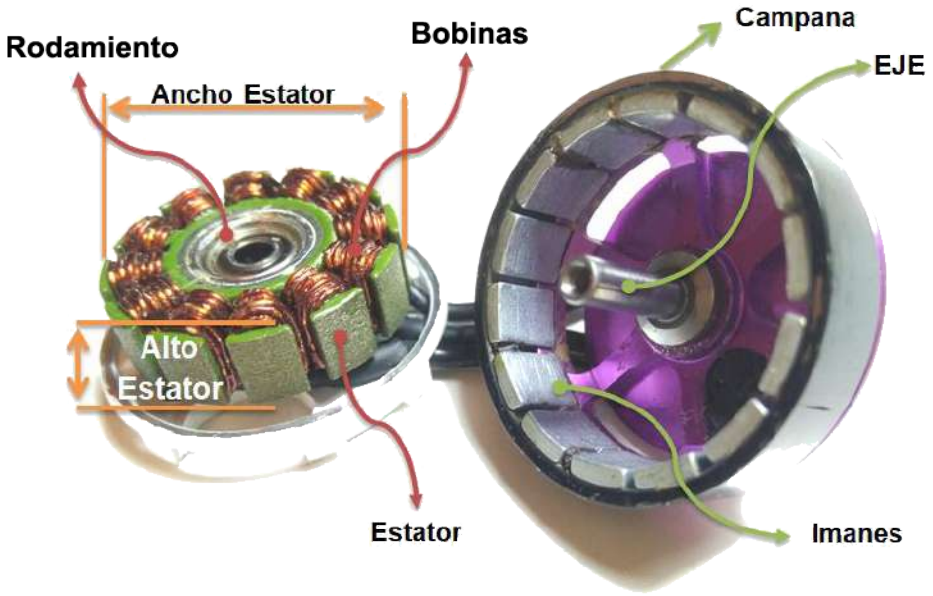
- **Rotor**: Es la parte giratoria que contiene imanes permanentes. Su movimiento se debe a las fuerzas de atracción y repulsión generadas por la secuencia de activación de las bobinas del estátor.

- **Estator**: Es la parte estática del motor. Contiene bobinas enrolladas en núcleos de hierro, que forman tres electroimanes (fases). Este componente es el encargado de

generar un campo magnético giratorio al recibir corriente eléctrica conmutada en secuencia.

Figura 8

Componentes de un Motor Brushless



Extraído de 3Bhobby 3B-R 2207 Pro

- **Sensores de Posición**: Algunos motores BLDC incluyen sensores de posición para determinar la ubicación del rotor y así ajustar la secuencia de activación de sus bobinas. Pueden utilizar sensores de efecto Hall (que detectan la posición del rotor midiendo el campo magnético de los imanes), encoders absolutos (que proporcionan información sobre la posición y velocidad del motor), o bien ser *sensorless* (sin sensores), estimando la posición mediante la fuerza contraelectromotriz (*Back Electromotive Force* - Back Electromotive Force (BEMF)).
- **Eje**: Es el eje que transmite el movimiento giratorio del rotor al sistema mecánico, ya sean hélices, ruedas, engranajes, etc.

- **Rodamientos**: Permiten que el eje gire suavemente con mínima fricción, al estar encajados en la carcasa alrededor del eje del motor.

- **Campana**: Es la estructura cilíndrica externa que gira junto con el eje del motor. Esta pieza es característica de los motores *outrunner* y se encarga de sostener los imanes permanentes del rotor.

2.2.3. Parámetros de un motor BLDC

- KV (RPM por voltio): Indica las Revoluciones por Minuto (RPM) que el motor genera por cada voltio aplicado (sin carga). Por ejemplo, un motor de 3000 KV alimentado a 12V puede alcanzar una velocidad de hasta 36 000 RPM. Es importante considerar que el KV es inversamente proporcional al torque y directamente proporcional a la velocidad máxima: un motor con un KV más alto puede alcanzar mayores velocidades, pero tendrá un menor torque.

- Tipo de rotor: El rotor puede configurarse de dos maneras principales dentro del motor:

  - Inrunner: Con el rotor en el interior, estos motores son más compactos y ligeros, pero ofrecen un menor torque.

  - Outrunner: Con el rotor en el exterior, pueden alcanzar un mayor torque, aunque son más pesados.

- Relación Bobinas vs Polos: Al construir o seleccionar un motor, es crucial considerar el factor de devanado. Este valor, entre 0 y 1, indica la proporción de la corriente de armadura que se convierte en par motor. Cada motor BLDC tiene una configuración fija de bobinas y polos. La ilustración 9, extraída del estudio de (Skaar et al., 2006), muestra cómo este factor varía según la combinación de polos y bobinas.

  Sobre esto, a mayor número de polos, se requiere una mayor cantidad de ciclos de conmutación por revolución, lo que permite generar más torque, pero reduce la

Ns/Nm	2	4	6	8	10	12	14	16	18	20	22	24	26	28	30	32	34
3	0.866	0.866	q<1/4														
6	q>1/2	0.866		0.866	q<1/4												
9		q>1/2	0.866	0.945	0.945	0.866	q<1/4										
12			q>1/2	0.866	0.933		0.933	0.866	q<1/4								
15				q>1/2	0.866		0.951	0.951		0.866	q<1/4						
18					q>1/2	0.866	0.902	0.945		0.945	0.902	0.866	q<1/4				
21						q>1/2	0.866	0.89		0.953	0.953		0.89	0.866	q<1/4		
24							q>1/2	0.866		0.933	0.949		0.949	0.933		0.866	q<1/4
27								q>1/2	0.866	0.877	0.915	0.945	0.954	0.954	0.945	0.915	0.877
30									q>1/2	0.866	0.874		0.936	0.951		0.951	0.936
33										q>1/2	0.866		0.903	0.928		0.954	0.954
36											q>1/2	0.866	0.867	0.902	0.933	0.945	0.953
39												q>1/2	0.866	0.863		0.917	0.936
42													q>1/2	0.866		0.89	0.913
45														q>1/2	0.866	0.858	0.886
48															q>1/2	0.866	0.857
51																q>1/2	0.866
54																	q>1/2

Figura 9

Valores factor de bobinado según número de bobinas y polos

velocidad máxima del rotor.

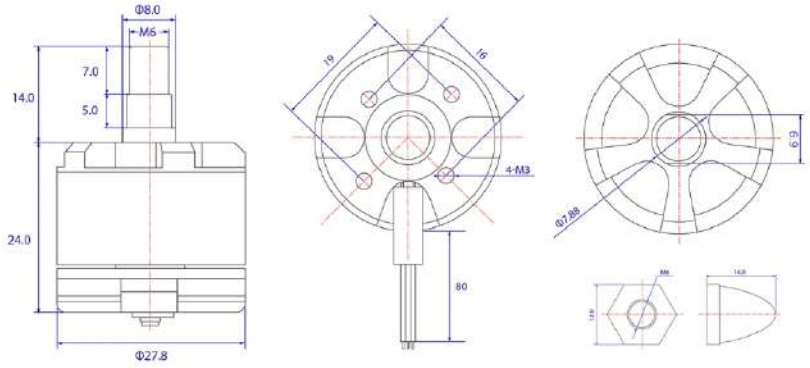
- Conexión de Electroimanes: El estátor contiene tres electroimanes distribuidos en varias bobinas. Estas bobinas, por lo general, pueden conectarse de dos maneras:
  - Delta: Esta conexión utiliza tres cables, donde cada fase del motor se conecta entre dos electroimanes. Permite alcanzar mayores velocidades, aunque con un menor torque.
  - Estrella: Esta conexión utiliza cuatro cables: tres se conectan directamente a los extremos de los tres electroimanes, y un cable adicional sirve como punto común o neutro. Permite obtener mayor torque a bajas velocidades con un bajo consumo.
- Eficiencia (g/W): La eficiencia se mide en gramos por *Watt* (g/W) en los manuales, indicando cuántos gramos puede levantar el motor por cada *Watt* de energía que consume. Por ejemplo, si un motor tiene una eficiencia de 6.1 g/W, significa que requiere 1W para levantar 6 gramos, o 166W para levantar 1 kilogramo.
- Voltaje Nominal (S): El voltaje de operación recomendado es el rango seguro para el motor, generalmente medido en número de celdas en serie. Cada celda, una batería de

Litio-Polímero o iones de litio, tiene un voltaje nominal de 3.7V. Por ejemplo, un motor con un rango de 3S a 5S puede operar entre 11.1V y 18.5V.

- Corriente Máxima (A): Otro parámetro crucial de un motor BLDC es la corriente máxima que puede consumir de la batería sin sufrir daños. Este dato es esencial para el diseño adecuado del controlador ESC.

Figura 10

Datasheet del motor brushless A2212 920KV



Moto	KV (PRM/V)	Voltage (V)	Prop (Inch)	Amps (A)	Thrust (g)	Watts (W)	Efficiency (g/W)	Throttle (%)
A2212	920	11.1	9450	1.5	180	16.65	10.81	50%
				4.2	370	46.62	7.94	75%
				7.8	600	86.58	6.93	100%
			1045	1.8	210	19.98	10.51	50%
				5.2	420	57.72	7.28	75%
				10	680	111	6.13	100%

2.2.4. Controlador Electrónico de Velocidad (ESC)

Un motor BLDC no puede operar sin un ESC (Controlador Electrónico de Velocidad). El ESC es el encargado de interpretar una señal PWM (Modulación por Ancho de Pulso) de un microcontrolador y transformarla en secuencias precisas de pulsos eléctricos trifásicos que hacen girar el motor. Su función es aplicar corriente a los electroimanes del motor siguiendo una estrategia de conmutación específica para regular la

velocidad en función de la señal PWM de entrada. Adicionalmente, el ESC ofrece protecciones contra sobrecorriente, sobretemperatura y bajo voltaje.

Figura 11

ESC Skywalker de 30A de HobbyWing

Existen varios modelos de ESC en el mercado; a continuación, se presentan los criterios clave para elegir el más adecuado:

- La elección de un ESC para un motor BLDC depende de la corriente máxima que el motor soporta, incluso bajo carga máxima o picos. Siempre se recomienda que el ESC tenga una capacidad de corriente superior a la máxima del motor. Por ejemplo, según la figura 10, si el motor A2212 920KV puede consumir hasta 10A, el ESC ideal para este motor debería ser de 15A o 20A.

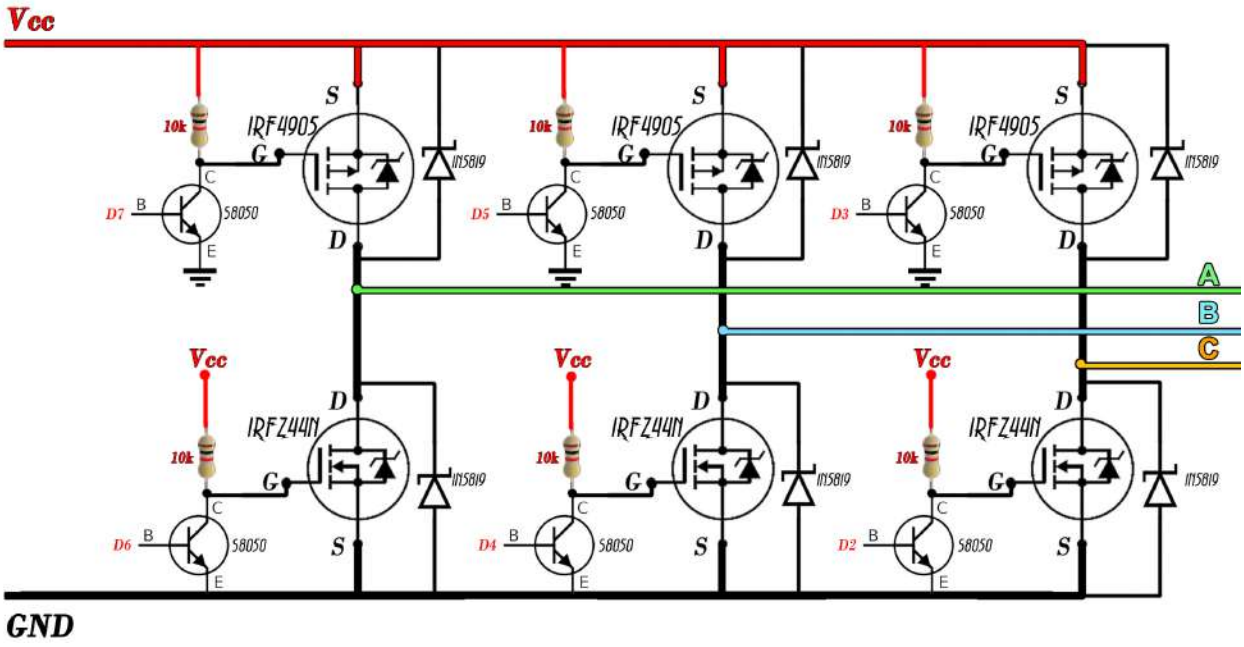
- El ESC debe ser compatible con el voltaje de la batería. Por ejemplo, si se utiliza una batería 3S, el ESC debe soportar ese rango de voltaje, de lo contrario podría dañarse o funcionar incorrectamente.

- Finalmente, considera si tu motor BLDC es *sensorless* o sensorizado. Si el motor incorpora sensores Hall, opta por un ESC sensorizado para un control más preciso. Si el motor no tiene sensores, puedes elegir un ESC más sencillo o uno que utilice la técnica BEMF.

En esta tesis, se desarrollará una PCB de un ESC personalizado para una FPGA, por lo que es fundamental comprender su circuito. La figura 12 muestra un circuito general de un ESC, que consiste en un arreglo de transistores MOSFET. La activación secuencial de estos transistores permite controlar los electroimanes del motor, logrando así su giro.

Figura 12

Etapa de Potencia de un ESC con control directo



Extraído de Electrotecnias

- Todos los ESC integran seis transistores MOSFET: tres configurados como High-Side (conectados a la alimentación de la batería) y tres como Low-Side (conectados a tierra). Estos MOSFET actúan como interruptores, recibiendo señales en sus compuertas en un orden específico para activar los electroimanes del motor.
- La activación de las compuertas de los MOSFET no se puede realizar directamente desde el microcontrolador, dado que requieren un voltaje superior. Por ello, en la figura 12 se optó por usar transistores BJT para proveer el voltaje necesario en las

compuertas. Otros diseños, sin embargo, emplean controladores (drivers) con protecciones adicionales, como los modelos IR2110, IR2101 o IRS2183.

2.2.5. Métodos de conmutación de un motor BLDC

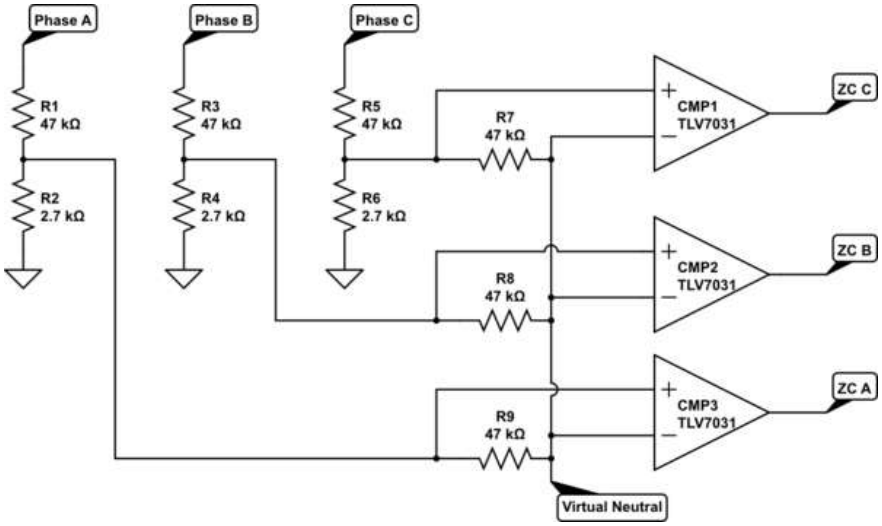
Los transistores MOSFET de un ESC deben conmutarse en un orden específico para controlar el movimiento del motor. Existen varios métodos de conmutación, entre ellos se encuentra:

- **Conmutación Trapezoidal**: Este es el método más simple, que consiste en alimentar los transistores MOSFET en una secuencia de seis pasos. La corriente resultante en los devanados sigue una forma trapezoidal. Sin embargo, los saltos directos en la conmutación pueden generar *torque ripple* (ondulación de par y vibraciones).
- **Conmutación Senoidal**: Se usa para obtener una mayor suavidad en la respuesta. Los transistores MOSFET se controlan con señales PWM moduladas sinusoidalmente, logrando que la corriente en los devanados siga una forma sinusoidal y reduciendo así el *torque ripple*.
- **Conmutación por Modulación de Vector Espacial (SVPWM)**: La Modulación por Ancho de Pulso Espacial Vectorial (SVPWM) es una técnica avanzada que permite controlar los transistores MOSFET de forma eficiente, generando una señal más cercana a una onda sinusoidal. Se basa en representar las tensiones trifásicas como un vector giratorio en un plano, optimizando los tiempos de conmutación. Esta técnica suele combinarse con el Control de Orientación de Campo (FOC), el cual separa la corriente del motor en componentes d-q para regular con precisión el par y el flujo magnético. La combinación de SVPWM y FOC mejora la eficiencia, reduce vibraciones y proporciona un control suave y dinámico.

En esta tesis se implementará un algoritmo de conmutación trapezoidal de seis pasos para el control de un ESC (Electronic Speed Controller) desarrollado desde cero.

Figura 13

Circuito de detección mediante BEMF - Oscar F. Becerra



Esta estrategia será empleada para accionar un motor BLDC (Brushless DC) sin sensores Hall, por lo que se integrará además la técnica de detección de la Fuerza Contraelectromotriz (Back Electromotive Force, BEMF). Esta técnica permite estimar la posición del rotor de forma indirecta, facilitando la conmutación precisa de las fases del motor sin requerir sensores físicos, lo cual resulta ideal para reducir costos y complejidad en aplicaciones embebidas.

2.3. Field-Programmable Gate Array (FPGA)

2.3.1. Definición de FPGA

Un FPGA (Field-Programmable Gate Array) es un conjunto de elementos lógicos reconfigurables que permiten implementar una gran variedad de circuitos digitales, desde una simple compuerta lógica hasta un procesador completo. A diferencia de un microcontrolador, que ejecuta instrucciones de manera secuencial en una arquitectura fija, un FPGA te da la libertad de diseñar directamente el hardware, creando estructuras que operan en paralelo. Por esta razón, los FPGA son muy utilizados en aplicaciones que demandan alta velocidad, precisión temporal y procesamiento paralelo, como las de telecomunicaciones, control industrial, visión artificial, aeroespacial, biomedicina o militar.

Figura 14

Placa de desarrollo DE10-lite con FPGA Altera MAX10 - Terasic

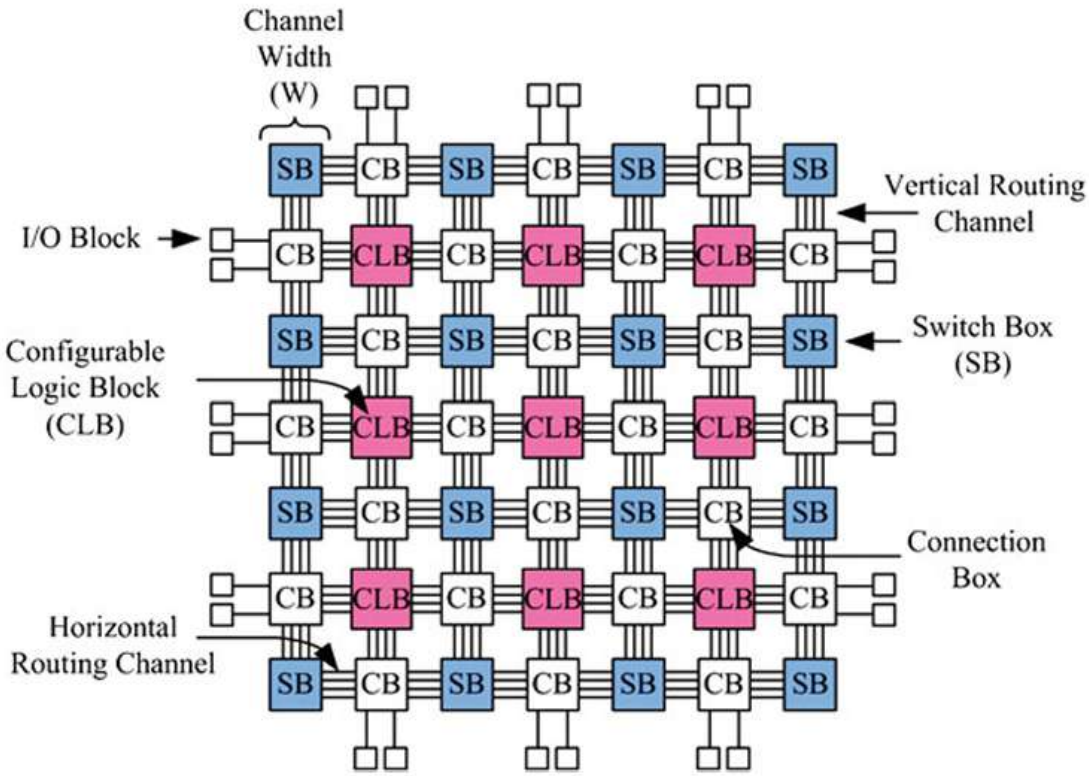
La característica principal del FPGA es su reprogramabilidad, lo que lo hace ideal para el prototipado e investigación, permitiendo probar y validar diseños repetidamente. En contraste, un Circuito Integrado de Aplicación Específica (ASIC) es un chip hecho a medida, diseñado para una tarea fija y que no se puede reprogramar. Los FPGA consumen más energía y operan a menor velocidad que los ASIC. Por esta razón, para aplicaciones finales, se prefiere el uso de un ASIC una vez que el diseño ha sido probado y finalizado con

un FPGA.

La arquitectura de una FPGA se basa en una matriz de elementos lógicos. Esta matriz está conformada por los *Configurable Logic Blocks* (CLB), como podemos observar en la figura 15:

Figura 15

Descripción de la Arquitectura de una FPGA - Springer, 2012



Entre los elementos presentes en la FPGA podemos mencionar:

- Configurable Logic Blocks (CLB): Es la unidad mínima de una FPGA. Está compuesta por *multiplexores*, *flip-flops* y *look-up tables* (LUTs) que se configuran según la tarea que queramos realizar.
- Connection Block (CB): Los bloques de conexión son los que conectan directamente los CLB. De los CLB, a su vez, salen los pines GPIO de una FPGA.
- Switch Block (SW): Estos bloques son los encargados de la interconexión general de

la FPGA. Su función es conectar varios *Connection Blocks* (CB) para facilitar la comunicación entre los CLB. A diferencia de los CB, los *Switch Blocks* no se conectan directamente a los CLB.

A continuación se presentan las ventajas y desventajas de un FPGA:

Cuadro 6

Ventajas y Desventajas de un FPGA

Ventajas	**Desventajas**
Pueden reprogramarse varias veces y probar varios diseños.	Sus curva de aprendizaje es lenta, es complejo de diseñar y requiere conocimiento de lenguajes HDL.
Permite ejecutar múltiples operaciones en paralelo.	Suelen consumir más energía que los microcontroladores.
Es ideal para sistemas que requieren de procesamiento en tiempo real.	Las FPGA son costosas, especialmente las de alta gama.
Permite diseñar bloques lógicos específicos a tus necesidades	El proceso de desarrollo y prueba es lento.

2.3.2. Lenguajes de Descripción de Hardware (HDL)

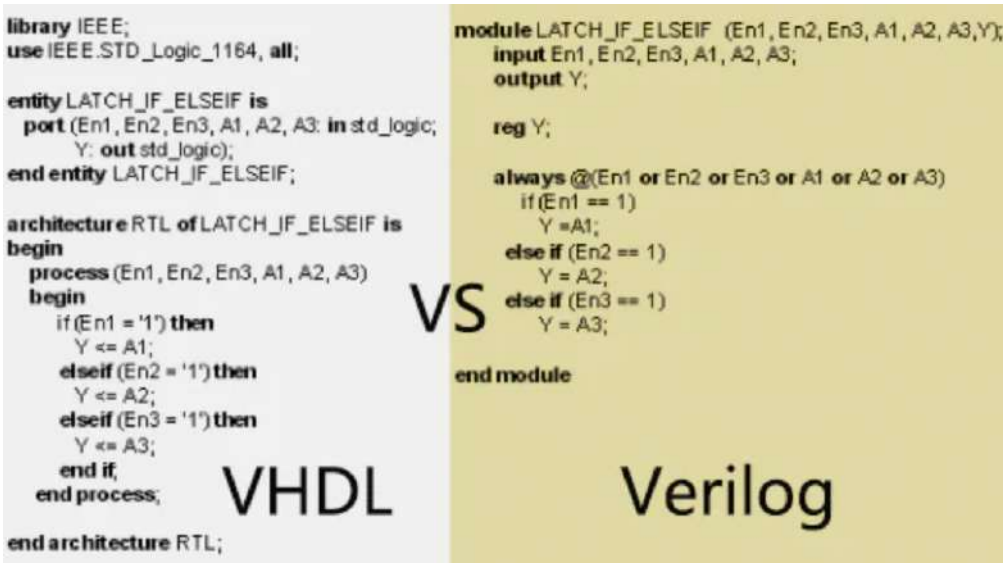
Un HDL (*Hardware Description Language*) se utiliza para describir la estructura de un sistema digital, que posteriormente se implementa en dispositivos como FPGA o ASIC. A través de un HDL, se puede definir cómo interactúan y se comportan los componentes de *hardware*, permitiendo la creación de diseños que luego pueden ser sintetizados en un circuito físico. Existen 3 lenguajes HDL principales:

- VHDL: Es un lenguaje más crudo de descripción de hardware, nos permite describir de manera muy detallada al hardware, fue desarrollado por la IEEE y se usa para el diseño y verificación de circuitos en FPGA.

- Verilog: Es un lenguaje desarrollado en 1980 y luego transferido a la IEEE, actualmente en su versión 2001. Su sintaxis es parecida a C, lo que puede hacerlo más accesible. Es más adecuado para proyectos donde la rapidez de desarrollo es clave.
- System Verilog: Es una mejora de Verilog que incluye nuevas funcionalidades con su versión 2009. Esta diseñado principalmente para la verificación de sistemas, además combina características de programación orientada a objetos y permite realizar tareas de verificación más complejas.

Figura 16

Comparación entre VHDL y Verilog - Electro Gadget



2.3.3. Entornos de Desarrollo

Las principales marcas de FPGA en el mercado son Altera (pertenecientes a Intel) y AMD (pertenecientes a Xilinx). Intel ofrece FPGA de familias Cyclone, Arria o Stratix; mientras que, Xilinx ofrece FPGA de familias Spartan, Artix, Kintex o Virtex. Estas no son las únicas marcas de FPGA, también existen otras marcas como Lattice Semiconductor o Microsemi.

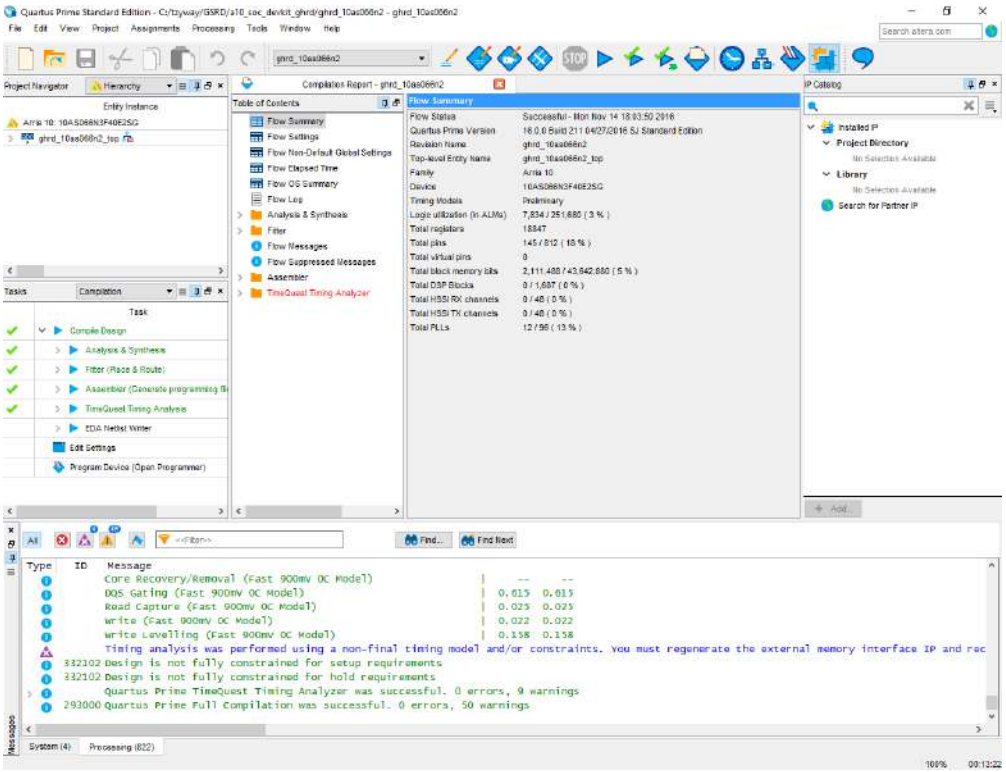
Cada una de estas empresas ofrece un entorno de programación para sus FPGA. A

continuación, veremos los softwares de desarrollo actuales de Intel y AMD:

- **Quartus Prime**: Este es el entorno de desarrollo para las FPGA de Intel. Nos permite la síntesis, simulación, análisis y programación de todas las familias de FPGA de Altera. Su interfaz, mostrada en la figura 17, ofrece herramientas para diseño RTL (*Register-Transfer Level*), análisis de temporización o asignación de pines. Además, incluye el Platform Designer, que nos permite integrar procesadores como Nios II junto con periféricos y bloques lógicos.

Figura 17

Quartus Prime - Intel

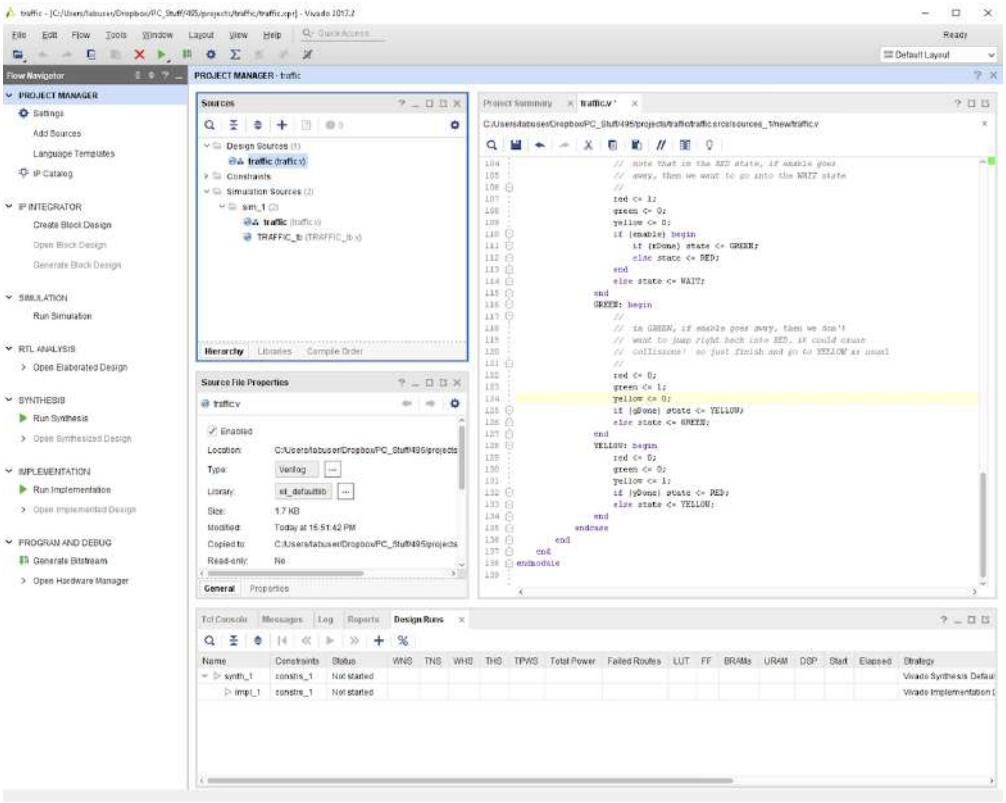


- **Vivado Design Suite**: Este es el entorno de desarrollo de AMD para sus FPGA. Nos permite la síntesis, simulación, análisis y programación de todas las FPGA de Xilinx. Su interfaz, mostrada en la figura 19, ofrece herramientas para diseño RTL (*Register-Transfer Level*), simulación y análisis de tiempos; prácticamente igual que

Quartus Prime, pero para FPGA Xilinx. También nos ofrece el Vivado IP Integrator para integrar el procesador Microblaze, el cual podemos personalizar a medida.

Figura 18

Vivado Design Suite - AMD



- **Modelsim**: Este no es un entorno de descripción de hardware, sin embargo, es uno de los simuladores más usados para VHDL y Verilog. Es compatible con muchos fabricantes de FPGA y se utiliza para la verificación y simulación de los *testbenchs* de diseños. Es muy poderoso para la depuración de código y análisis de tiempo de simulación.

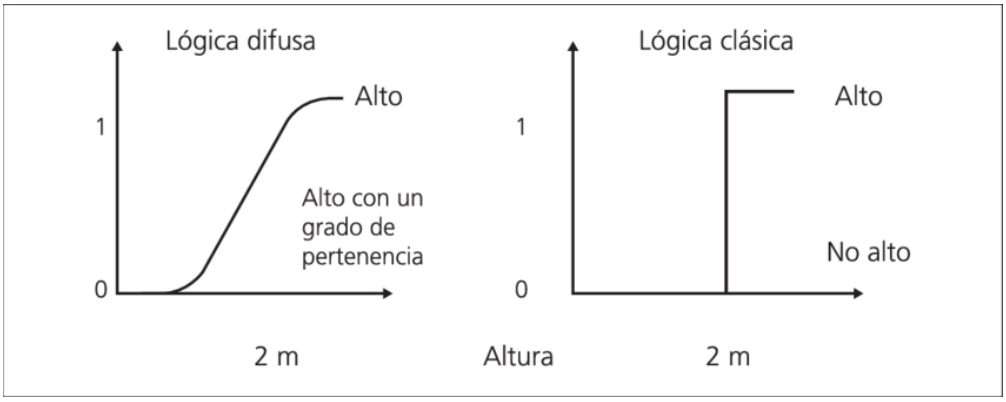
2.4. Fundamentos de Lógica Difusa

La lógica difusa fue propuesta en 1965 por Lotfi Zadeh (Zadeh, 1965). Es un tipo de razonamiento que busca imitar la capacidad del pensamiento humano para manejar información imprecisa. A diferencia de la lógica clásica donde una afirmación es estrictamente verdadera (1) o falsa (0), la lógica difusa permite una gama más amplia de veracidad, admitiendo valores intermedios entre 0 y 1.

En lugar de clasificar la estatura de una persona como estrictamente baja (0) o alta (1), la lógica difusa nos permite expresar grados de pertenencia, como que su estatura está un poco baja (0.25) o casi alta (0.9).

Figura 19

Lógica Difusa vs Lógica Clásica - María García B.



Gracias a su capacidad para trabajar con valores intermedios, que se asemejan al lenguaje humano, los controladores difusos pueden tomar decisiones en sistemas donde las condiciones no siempre son exactas, sino que operan con rangos y grados de verdad. Esto nos permite traducir la experiencia humana a un sistema computacional mediante reglas lingüísticas comprensibles, lo que simplifica el diseño del control. Una de sus mayores ventajas es que no requieren un modelo matemático preciso del sistema, lo que se traduce en un considerable ahorro de tiempo durante la etapa de desarrollo.

En la tabla 7 se resumen las ventajas y desventajas del uso de controladores difusos:

Cuadro 7

Ventajas y Desventajas de un controlador difuso

Ventajas	Desventajas
No necesita un modelo matemático exacto del sistema.	Requiere conocer las reglas lingüísticas de control basadas en la experiencia de una persona.
Permite tener múltiples entradas y salidas del controlador.	No garantiza estabilidad, ni un buen desempeño sin un buen diseño de las reglas.
Es muy intuitivo ya que usa lenguaje humano para tomar decisiones.	Puede requerir establecer una gran cantidad de reglas para que el sistema funcione correctamente.
Se adapta bien a sistemas no lineales o difíciles de controlar.	Requiere más recursos computacionales en implementaciones complejas.

2.4.1. Términos de la Lógica Difusa

- **Conjuntos Difusos**: En la lógica clásica, un elemento pertenece o no pertenece a un conjunto de forma binaria. Por el contrario, en la lógica difusa, todos los elementos poseen un grado de pertenencia que oscila entre 0 y 1 a un conjunto difuso. Este grado cuantifica qué tan fuertemente un elemento pertenece al conjunto y se establece mediante una función de membresía.

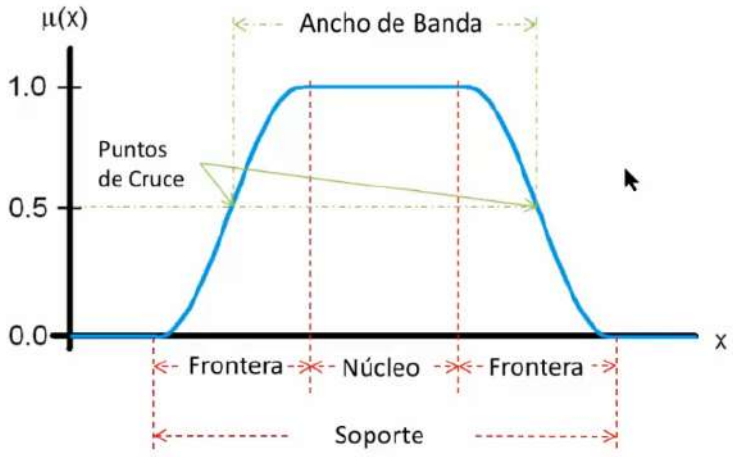
- **Universo de discurso**: Es el conjunto de todos los posibles valores que una variable puede asumir. Es fundamental conocerlo porque establece los límites dentro de los cuales se evaluarán los conjuntos difusos.

- **Funciones de membresía**: La función de membresía define el grado de pertenencia

de un elemento del universo de discurso a un conjunto difuso. Su dominio es el universo de discurso y su rango siempre se encuentra entre 0 y 1. En la gráfica de una función de membresía, como la mostrada en la figura 20, podemos observar varias partes:

Figura 20

Partes de una función de membresía - Hackeando Tec



- **Núcleo**: Esta es la zona donde el grado de pertenencia es igual a 1.
- **Frontera**: Son las áreas donde el grado de pertenencia se encuentra entre 0 y 1.
- **Soporte**: Es la zona donde el grado de pertenencia es mayor que 0. Se define también como la unión del núcleo y las fronteras.
- **Cruce**: Se refiere al elemento del universo de discurso que tiene un grado de pertenencia igual a 0.5.
- **Ancho**: Es la diferencia entre los dos puntos de cruce de la función de membresía.

- Variables Lingüísticas: Una variable lingüística permite representar magnitudes usando palabras en vez de números. Estas variables emplean conjuntos difusos para asignar un significado numérico a sus valores a través de funciones de membresía. Por

ejemplo, para la variable lingüística "temperatura", podríamos definir conjuntos difusos como: Frío o Caliente.

- **Reglas Difusas**: Se utilizan en la creación de sistemas de control difuso para conectar las entradas del controlador con sus salidas. Un ejemplo sería: "Si la temperatura es caliente y la velocidad es baja, entonces la potencia del motor es media."

- **Fuzzificación**: Consiste en convertir una entrada numérica en valores difusos, asignándole grados de pertenencia a diferentes conjuntos difusos. Por ejemplo, si la temperatura de entrada es 28°C, esta variable podría pertenecer en un 0.2 al conjunto "templadoz en un 0.8 al conjunto çaliente".

- **Defuzzificación**: Es el último paso del controlador difuso. Consiste en transformar una salida difusa en un único valor numérico, empleando alguna técnica de defuzzificación.

2.4.2. Tipos de Controladores Difusos

Existen varios tipos de control difuso, y la elección del tipo adecuado depende de la complejidad del sistema a controlar y de los requisitos de precisión. Los principales tipos son:

- Mamdani: Es el más común y uno de los más sencillos de comprender y aplicar. En este enfoque, la salida se expresa mediante conjuntos difusos, y se utiliza un método de defuzzificación para obtener un valor de control numérico.

- Sugeno: El control difuso de tipo Sugeno utiliza funciones matemáticas como salidas, en lugar de conjuntos difusos como lo hace Mamdani. Este enfoque es más adecuado para sistemas que requieren un modelo matemático más preciso, y elimina la necesidad de una técnica de defuzzificación, ya que las funciones directamente producen un valor numérico.

- Tsukamoto: Este control es una variante del tipo Mamdani, donde se calcula un valor numérico para cada regla usando funciones que transicionan suavemente de 0 a 1. Posteriormente, se realiza un promedio ponderado de esos valores para obtener la respuesta final.

- Control Difuso Híbrido: Este tipo de control combina el sistema de control difuso con otros métodos, como el control PID, el control adaptativo o el uso de redes neuronales. Se emplea para aprovechar las ventajas de múltiples enfoques y así mejorar el rendimiento general del sistema.

Cuadro 8

Comparación entre controladores difusos

Mamdani	**Sugeno**	**Tsukamoto**
Necesita técnica de defuzzificación.	Solo defuzzifica por regla, porque la salida ya es un número.	No necesito defuzzicar, porque la salida se obtiene por promedio.
Es más lento, porque maneja muchos cálculos y hace defuzzificación.	Es más rápido al tener funciones matemáticas	No es lento ni rápido, ya que se hacen algunos cálculos por cada regla.
Es el más fácil de diseñar, al ser más cercano al lenguaje humano.	Es más difícil de diseñar, al requerir modelos matemáticos precisos.	Su dificultad depende de las funciones definidas.
Es más preciso, pero usa potencia computacional.	Es menos preciso, pero más eficiente.	Ofrece un balance entre precisión y eficiencia.

Se eligió el control difuso tipo Sugeno por su eficiencia computacional y fácil implementación en FPGA, lo que permite respuestas rápidas y precisas en tiempo real. A diferencia del método Mamdani, Sugeno evita la defuzzificación compleja al usar salidas

matemáticas directas, siendo ideal para entornos marinos dinámicos donde el ROV debe adaptarse constantemente.

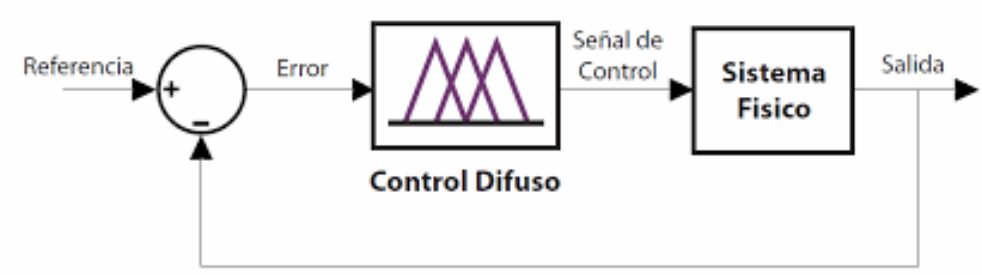
2.4.3. Métodos de Defuzzificación

La defuzzificación es el proceso de convertir una salida difusa en un valor numérico único. Entre los métodos para obtener ese valor numérico se tienen:

- Centro de Gravedad: Es el método más popular y consiste en calcular el centro de área bajo la curva de membresía difusa. Es decir, encuentra el "promedio ponderado" de todas las posibles salidas, donde las áreas más grandes tienen más peso. Este método es preciso, pero también es más costoso computacionalmente.
- Bisector del Área: Este método divide en dos partes iguales el área bajo la curva de membresía y busca el punto de equilibrio en el que la mitad del área está a la izquierda y la otra mitad a la derecha..
- Mayor de los Máximos (LOM): Elige el valor de salida con la mayor membresía. Es una técnica más simple y rápida que solo toma el valor con la máxima pertenencia, pero puede ser menos preciso en situaciones complejas.
- Menor de los Máximos (SOM): Elige el valor con la menor pertenencia entre los que tienen la mayor membresía.

Figura 21

Sistema de Control Difuso - Redalyc



2.5. Herramientas de Desarrollo

2.5.1. MatLab - Simulink

MatLab es un software desarrollado por MathWorks, y Simulink es su herramienta de MatLab para la simulación de sistemas dinámicos. Se usará para modelar y simular el sistema de control de manera gráfica, utilizando bloques que representan diferentes partes del sistema. Esto permite probar el comportamiento del sistema antes de implementarlo en hardware, asegurando que el diseño funcione correctamente.

2.5.2. MatLab - Fuzzy Logic Designer

Fuzzy Logic Designer, también de MathWorks, es una herramienta especializada en la creación de sistemas de lógica difusa. Se utilizará para diseñar el controlador difuso que gestionará el sistema de control. Con esta herramienta, es posible definir reglas lógicas y ajustar parámetros de manera fácil y visual, sin necesidad de escribir código.

2.5.3. MatLab - HDL Coder

MatLab HDL Coder es otra herramienta de MathWorks que convierte los modelos creados en Simulink en código de hardware, que puede ser VHDL, Verilog o System Verilog. Este código está en lenguaje HDL, que es el que entienden las FPGAs. Esto permite que el sistema de control se implemente rápidamente en la FPGA, optimizando el proceso de diseño y asegurando un buen rendimiento en el hardware.

2.5.4. Vivado Design Suite

Vivado Design Suite es una plataforma de diseño de Xilinx que se usa para crear y programar proyectos en FPGAs. En este proyecto, se utilizará Vivado para cargar el código generado en MatLab en la FPGA a usar en esta tesis, que es de Xilinx, y realizar pruebas de funcionamiento. También permite depurar y ajustar el sistema, asegurando que el diseño sea eficiente y cumpla con los requisitos del proyecto.

2.5.5. Altium Designer

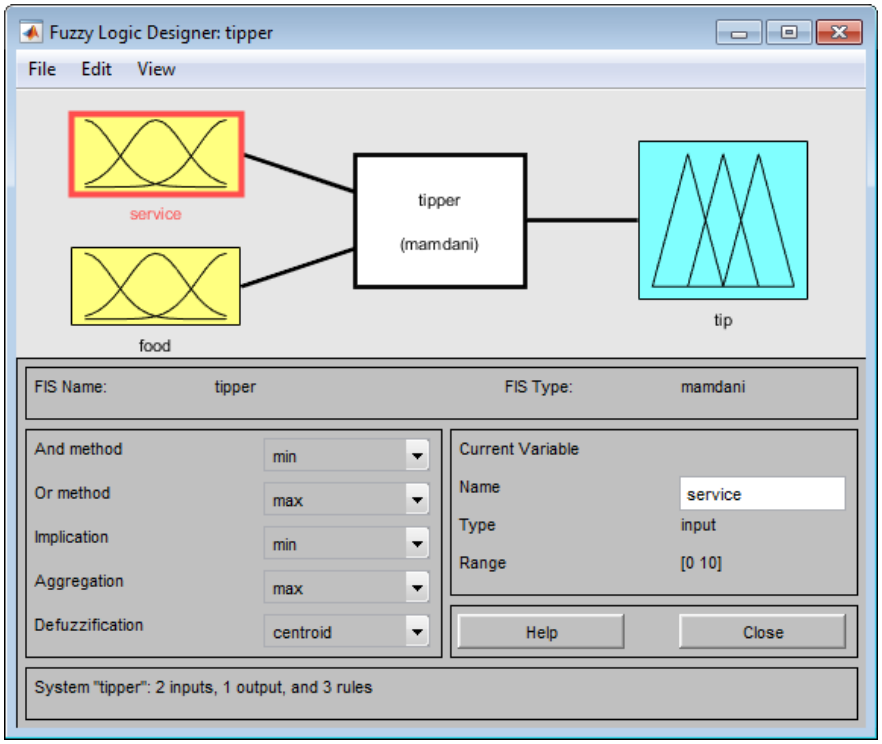
Altium Designer es un software de diseño de circuitos electrónicos que se usará para el diseño de la ESC de los motores BLDC. Con Altium, se pueden crear los esquemáticos y las PCB necesarias para la ESC, asegurando que todos los componentes estén conectados correctamente y que el diseño sea eficiente para el control de los motores.

2.5.6. NI LabVIEW

NI LabVIEW es una plataforma gráfica de programación desarrollada por National Instruments. Se utilizará para simular una pequeña estación terrena de control del ROV, donde habrán indicadores de los sensores y gráficas del control en tiempo real. LabVIEW permite crear interfaces gráficas de usuario, facilitando la visualización del rendimiento del sistema y permitiendo hacer ajustes durante las pruebas físicas del proyecto.

Figura 22

Fuzzy Logic Designer - MathWorks



3. Capítulo III: Desarrollo del Trabajo de Investigación

3.1. Arquitectura y selección de componentes del sistema

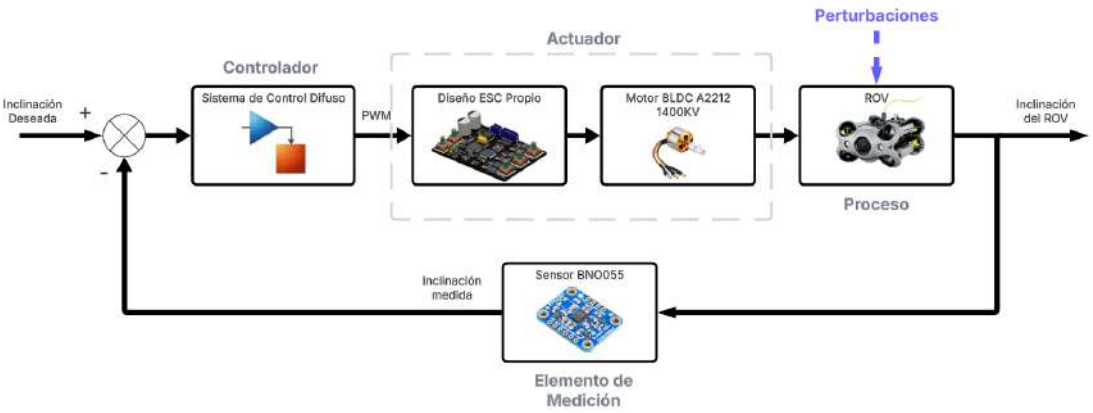
En esta sección se presenta el criterio de selección de los componentes que conforman el sistema de control de propulsión y estabilización de un ROV, a través de sus motores BLDC. Además, se incluye un esquema de la arquitectura general del sistema. La selección de estos componentes se basa en criterios de rendimiento, disponibilidad, costo y compatibilidad con la placa de desarrollo basada en FPGA.

3.1.1. Diagrama de bloques del sistema de control

El diseño del diagrama de bloques del sistema sigue una estructura modular utilizada en la ingeniería de control moderno, donde se representa de forma abstracta la relación entre entrada, planta, sensor, controlador y actuador (Ogata, 2010). Este enfoque permite una visión clara del funcionamiento general del sistema, facilitando su análisis.

Figura 23

Diagrama de Bloques del Sistema de Control de un ROV



Nota. Elaboración Propia usando la plataforma LucidChart

El diagrama presentado en la figura 23 ilustra el funcionamiento del sistema de control propuesto en esta tesis, mostrando los componentes seleccionados que conforman el sistema. A continuación, se describen las distintas partes que lo integran:

- ***Setpoint*:** La entrada del sistema de control son los ángulos de inclinación deseados

(yaw, pitch, roll) para el ROV. Estos ángulos son calculados según las instrucciones del piloto para realizar un movimiento al ROV.

- **Planta:** El proceso a controlar será el chasis del ROV que se ubicará en un entorno marino. El ROV cuenta con una determinada cantidad de motores que irán distribuidos alrededor del chasis del ROV.
- **Sensor:** El sensor encargado de medir la inclinación en tiempo real del ROV será la Unidad de Medición Inercial (IMU) BNO055, con el cuál podremos determinar las inclinaciones yaw, pitch y roll. El sensor entrega las señales medidas al controlador.
- **Controlador:** Es el encargado de comparar la señal medida con la señal deseada para aplicar una acción de control, para esta investigación, un control difuso. El sistema de control es implementado en una FPGA cuyo modelo es Artix XC7A100T.
- **Actuador:** Consiste de una determinada cantidad de motores BLDC, los cuales permiten mover al ROV. Para mover estos motores requerimos una señal PWM enviará por la FPGA, la cual irá a un Controlador Electrónico de Velocidad (ESC).

3.1.2. Unidad de procesamiento y lógica de control: FPGA

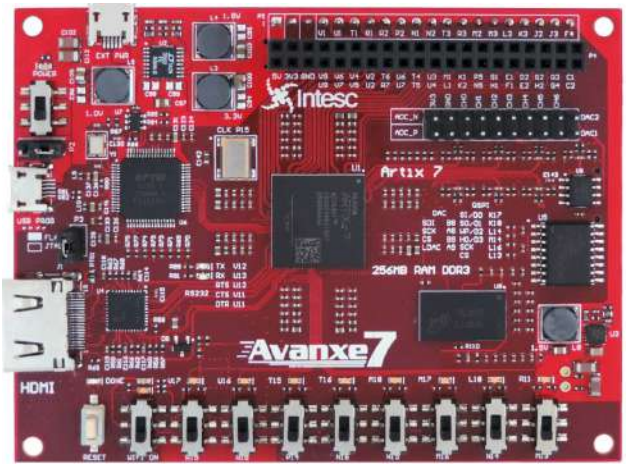
El sistema de control difuso se implementará en una placa de desarrollo basada en FPGA, la cual gestionará las señales de entrada y salida, aprovechando el paralelismo para realizar procesamiento en tiempo real y controlar los motores del ROV.

La tarjeta de desarrollo seleccionada para la implementación del sistema es la tarjeta Avanxe 7 100T, diseñada por la empresa mexicana Intesc (Embebidos, 2025). Esta tarjeta incorpora una FPGA Artix-7 XC7A100T de Xilinx, reconocida por su equilibrio entre capacidad lógica, bajo consumo de energía y versatilidad para aplicaciones embebidas en tiempo real. Se muestra esta tarjeta en la figura 24, la cual incorpora una amplia variedad de periféricos para realizar diversos proyectos embebidos. Su elección se fundamentó en su disponibilidad, tamaño compacto y en la integración de una FPGA con

una considerable cantidad de recursos lógicos, lo cual resulta ideal para aplicaciones de control en tiempo real.

Figura 24

Tarjeta de Desarrollo Avanxe 7 basada en una FPGA Artix XC7A100T



Nota. El gráfico muestra la vista frontal de Avanxe 7. Tomado de (Embebidos, 2025)

De acuerdo con el manual técnico elaborado por Intesc (Embebidos, 2021), es posible identificar los principales periféricos integrados en la tarjeta Avanxe 7:

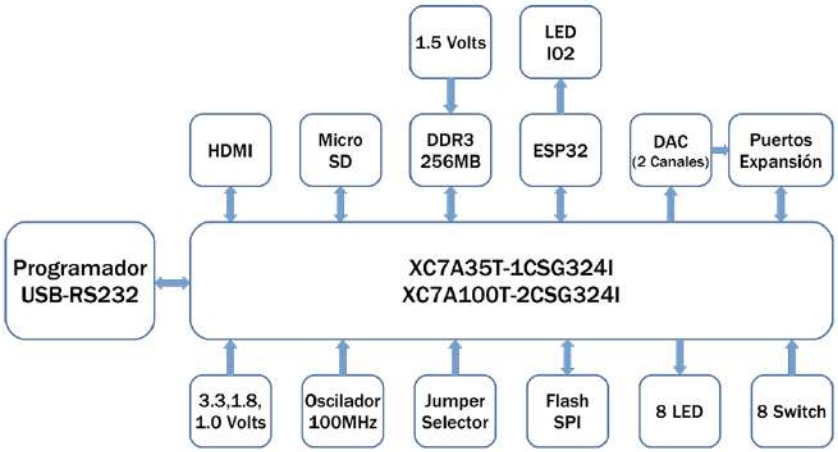
- Modelo de FPGA: XC7A100T-2CSG324I
- Elementos Lógicos: 101440
- Memoria Flash de 256Mb
- Memoria RAM de 4680Kb
- Oscilador de 100MHz
- ADC de 12 bits de resolución a 1SMPS
- DAC MCP4822E de 12 bits de resolución en 2 canales
- Programador JTAG FT2232H
- Puerto HDMI con acoplador TMDS141

- Módulo ESP32 con WiFi/Bluetooth
- Slot para memoria Micro SD
- 51 pines de entradas y salidas digitales
- 8 leds y 8 switches integrados

En la Figura 25 se ilustra el diagrama de bloques de la tarjeta Avanxe 7 donde se visualizan sus periféricos.

Figura 25

Diagrama de bloques de la tarjeta Avanxe 7



Nota. Tomado de (Embebidos, 2021)

3.1.3. Propulsores BLDC: T60 860KV

Los motores BLDC utilizados para el diseño serán los propulsores submarinos T60 KLV11100 (AliExpress, 2025) mostrados en la figura 26. Estos motores están diseñados para operar en entornos submarinos, por lo que traen materiales impermeables y su motor presenta un alto toque, siendo adecuado para accionar un ROV en condiciones exigentes.

Dado que los propulsores para ROV suelen tener un costo elevado, se optó por adquirir un motor proveniente de la tienda Aliexpress, el cual es una imitación china basada en modelos de propulsores comerciales. A pesar de ser una alternativa más

económica, se verificó que este motor es suficiente para llevar a cabo las pruebas del sistema de control difuso en tiempo real, cumpliendo con los requisitos funcionales de la aplicación.

Figura 26

Propulsor BLDC T60 860KV

Nota. Tomado de (AliExpress, 2025)

El motor BLDC T60 860KV es un motor sin escobillas de alta eficiencia y par, como se observa en la figura 27, diseñado para aplicaciones que requieren fuerza y estabilidad, como ROVs o drones de carga media. Su clasificación 860KV indica que entrega 860 revoluciones por minuto por cada voltio aplicado, lo que lo hace adecuado para configuraciones que priorizan el torque a bajas revoluciones. Para su control, se requiere un ESC compatible capaz de entregar corriente sostenida de hasta 30A, en función de la carga y el tipo de hélice utilizada. A continuación se presentan las características principales del propulsor BLDC:

- Voltaje de Alimentación: 2S - 4S
- Corriente sin carga: 0.7A
- Corriente de carga máxima: 23A
- Revoluciones por Voltio (KV): 860KV

- Tipo de Rotor: Outrunner
- Número de Bobinas e imanes: 12 - 14
- Peso: 51g
- Empuje: 1.65Kg (3S) o 2.10Kg (4S)
- Tamaño: 65mm x 26mm x 15.5mm

Figura 27

Motor BLDC T60 860KV por dentro

Nota. Tomado de (AliExpress, 2025)

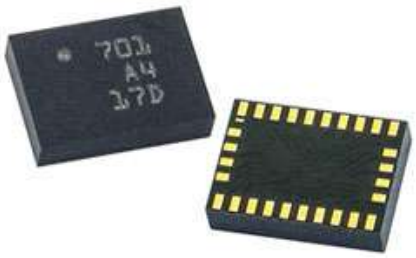
3.1.4. Sensor de Estabilidad BNO055

Para medir la inclinación del ROV en tiempo real es decidió utilizar la Unidad de Medición Inercial (IMU) BNO055 de la empresa Bosch Sensortec (GmbH, 2024), mostrado en la figura 28. Mediante este sensor lograremos determinar si el ROV es inestable o no, para luego aplicar las acciones de control correctivas.

El BNO055 es un MEMS que combina múltiples sensores para obtener directamente los datos de orientación 3D. Entre los sensores que integra el BNO055 se tiene un acelerómetro de 3 ejes (para detectar aceleración lineal), un giroscopio de 3 ejes (para detectar rotación), un magnetómetro de 3 ejes (para detectar la orientación respecto al campo magnético) y un microcontrolador ARM Cortex-M0, el cual fusiona los datos de los sensores y entrega información ya procesada usando el protocolo de comunicación I2C.

Figura 28

MEMS BNO055 desarrollado por Bosch Sensortec

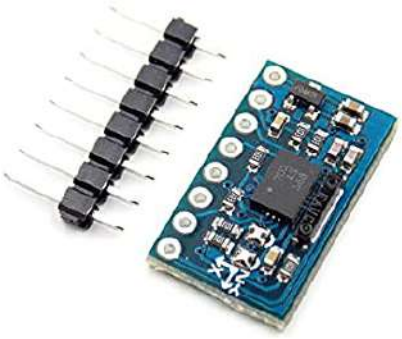


Nota. Tomado de (GmbH, 2024)

Para fines de prototipado, en este proyecto se optó por utilizar un módulo que integra el sensor BNO055, como se muestra en la figura 29. Esta versión es una réplica funcional del módulo comercialmente distribuido por Adafruit, seleccionada principalmente por su menor costo, lo que la convierte en una alternativa accesible para entornos académicos sin comprometer significativamente la funcionalidad requerida para la adquisición de datos de orientación.

Figura 29

Módulo GY-BNO055



Nota. Tomado de EGBO - Aliexpress

3.2. Diseño del Controlador Electrónico de Velocidad (ESC)

En esta sección se describe el diseño del circuito y la lógica de control del Controlador Electrónico de Velocidad (ESC), cuya función es regular la velocidad de giro de los motores BLDC a partir de señales de control generadas por la FPGA.

Se optó por desarrollar un ESC personalizado con el fin de aprovechar al máximo el paralelismo de la arquitectura de la FPGA, permitiendo así el control directo de las fases del motor mediante señales digitales que activan los transistores encargados de manejar los electroimanes. Además al controlar las fases con la FPGA tenemos la libertad de cambiar el sentido de giro del motor, característica que varios ESC comerciales no tienen incluido.

Todos los diseños se desarrollaron usando el software Altium Designer v25.1.2, una de las herramientas más reconocidas a nivel mundial para el diseño electrónico avanzado. El proceso de diseño se llevará a cabo siguiendo criterios técnicos establecidos, tales como el uso de modelos de cuadripolos para representar el comportamiento eléctrico de las conexiones, y cumpliendo con las normas IPC vigentes (IPC, 2019), que garantizan buenas prácticas de diseño y calidad en el ruteo, espaciado y manufactura de las PCBs.

A continuación, se presentan las distintas versiones del diseño de PCBs para el ESC, desarrolladas con el propósito de realizar pruebas experimentales, evaluar el comportamiento del motor BLDC ante diferentes configuraciones electrónicas y validar el diseño progresivamente. Estas placas fueron concebidas con una disposición tipo shield, lo que permite montarlas directamente sobre la tarjeta de desarrollo Avanxe 7, facilitando así la conexión, la depuración y la portabilidad del sistema completo.

3.2.1. Diseño de la versión 1 del shield ESC

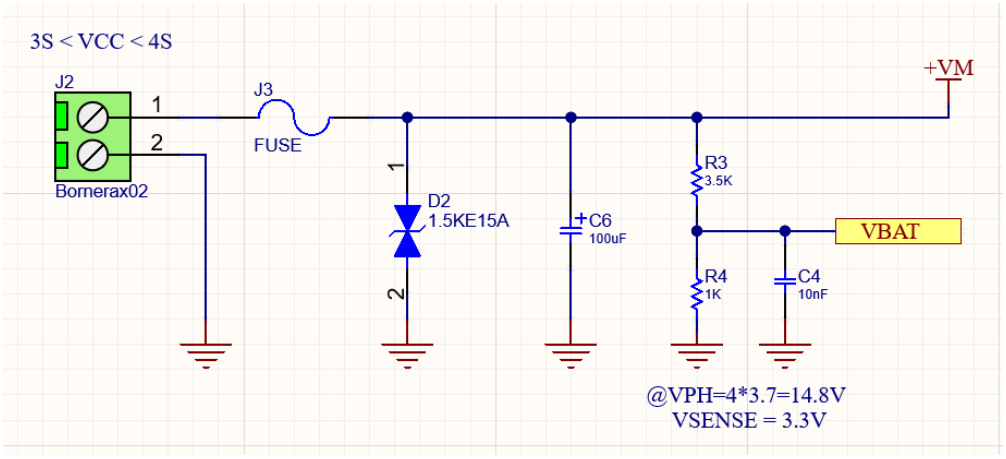
Esta primera versión del diseño fue realizada con el objetivo de efectuar pruebas iniciales de control sobre los motores BLDC T60 860KV. Se priorizó el uso de componentes que pudieran adquirirse localmente, sin necesidad de recurrir a importaciones, con el fin de facilitar la fabricación y el ensamblaje del prototipo. Por esta razón, se emplearon exclusivamente componentes de agujero pasante (THT), los cuales también simplifican las

tareas de soldadura y reemplazo durante la etapa de pruebas.

I. Etapa de Alimentación En esta etapa se incorpora la conexión de la batería encargada de alimentar los motores BLDC, la cual puede operar dentro de un rango de 3S a 4S (11.1V a 14.8V), incluyendo protecciones básicas para evitar sobrecargas o descargas profundas. Además, se integra un circuito de monitoreo de voltaje, que permite a la FPGA supervisar en tiempo real el estado de la batería. El diseño se muestra en la figura 30.

Figura 30

Etapa de Alimentación de la versión 1 del ESC



Nota. Elaboración Propia

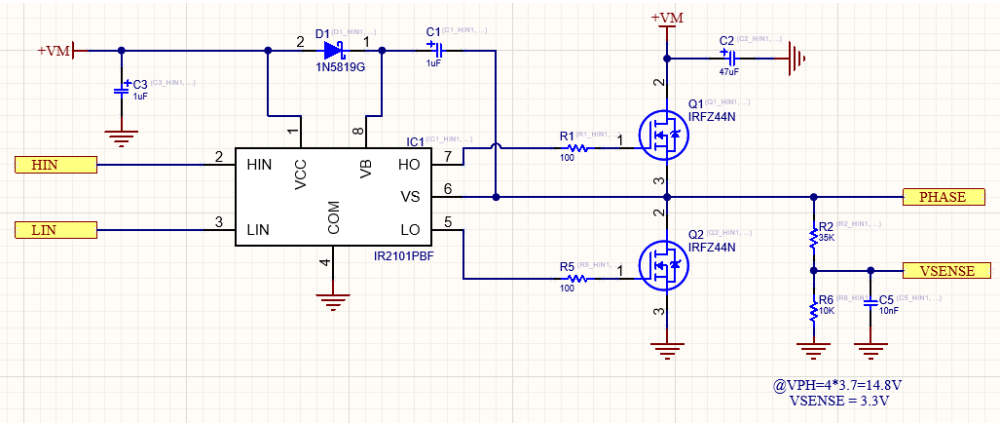
- Se eligió una bornera de tipo Terminal Barrier, debido a que son capaces de maneja altas corrientes y pueden sostener el cable de la fase del motor BLDC con un tornillo.
- Se eligió un porta-fusible para fusibles automotrices de tipo cuchilla, debido a su capacidad de manejar altas corrientes y ser cómodos de reemplazar.
- Se eligió un diodo TVS unidireccional 1.5KE15A de 15v 200A, el cual protege el circuito en caso de sobre-voltajes transitorios.
- Se eligió un capacitor electrolítico en la entrada para estabilizar el voltaje de la batería, además este capacitor permite filtrar el ruido de alta frecuencia.

- Se eligió un divisor de tensión mediante un par de resistencias para reducir el voltaje de la batería a un voltaje de 3.3V o inferior, el cuál puede ser leído por uno de los canales del ADC de la tarjeta de desarrollo Avanxe 7.

II. Etapa de Control de Fase En esta etapa se implementa el control de un par de transistores MOSFET, encargados de conmutar la corriente hacia los electroimanes asociados a una de las fases del motor BLDC. El esquemático presentado corresponde al control de una sola fase, por lo que deberá replicarse tres veces en el diseño final para cubrir las tres fases del motor. El diseño se muestra en la figura 31.

Figura 31

Etapa de Control de Fase de la versión 1 del ESC



Nota. Elaboración Propia

- Se eligió usar un circuito integrado IR2101 (Rectifier), 2019) para generar las señales de control de las compuertas de los transistores MOSFET, con este integrado ingresamos 2 señales (HIN y LIN) desde la FPGA y de esa manera conmutamos los transistores MOSFET con el voltaje de la batería.

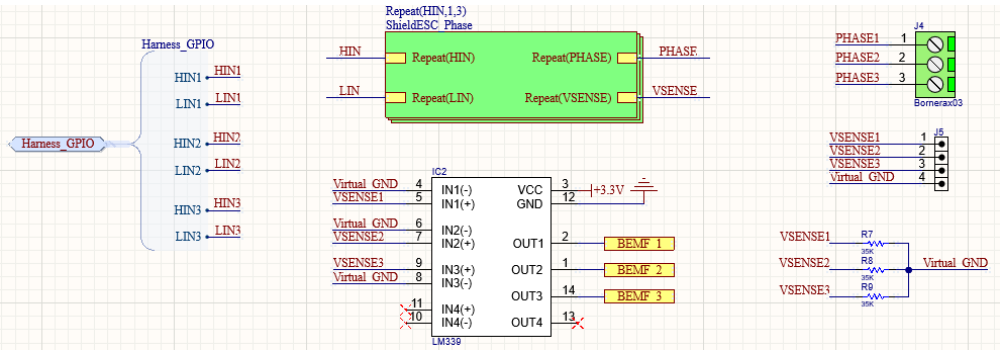
- Se eligió usar el transistor MOSFET de canal N IRFZ44N (Rectifier), 2021), el cual puede manejar corrientes de hasta 49A con un voltaje de encendido en la compuerta de 10V a 15V. Además este transistor es muy comercial y fácil de obtener en mi ciudad además de contar con una gran velocidad de conmutación.

- Para la lectura del BEMF, usamos un divisor de voltaje con 2 resistencias para de esta manera ingresar una señal a un comparador de señales de 3.3V.

III. Etapa de Comparación de BEMF En esta etapa se lleva a cabo la comparación de las señales de fuerza contraelectromotriz (BEMF) con una referencia de tierra virtual, con el fin de detectar el cruce por cero de la señal y así estimar la posición angular del rotor del motor BLDC. El diseño se muestra en la figura 32.

Figura 32

Etapa de Comparación de BEMF de la versión 1 del ESC



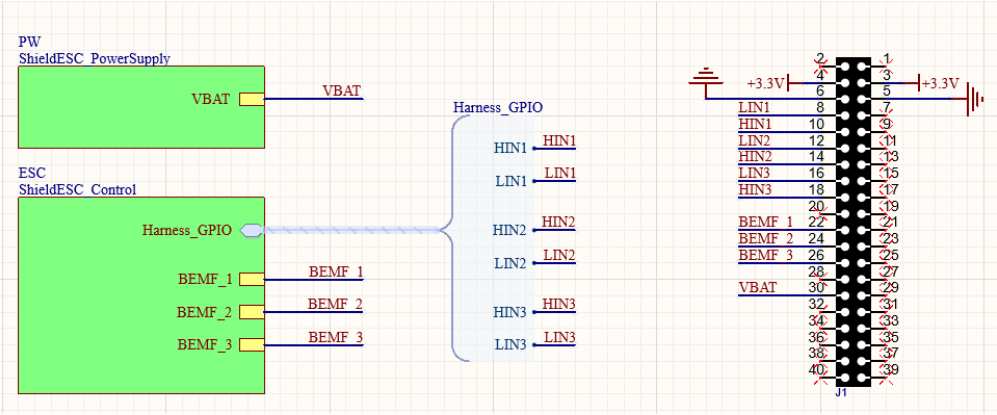
Nota. Elaboración Propia

- Para generar la tierra virtual usamos 3 resistencias de 35K a cada una de las fases.
- Para realizar la comparación de las señales BEMF usamos un comparador LM339, el cual internamente contienen 4 comparadores para señales desde 2V a 36V. En la salida del comparador se obtienen señales digitales que serán leídas por la FPGA.
- Se eligió también usar una bornera triple de tipo Terminal Barrier para las 3 fases del motor BLDC.
- En el esquemático también se ven otras herramientas de Altium como el usó de *Sheet Symbol* y *Harness*, con las cuales logró generar 3 veces el esquemático del control de fase, con lo cual puedo controlar las 3 fases del motor y con el *Harness* logró enviar varias señales al esquemático principal.

IV. Etapa de Conexión con la FPGA En esta etapa colocamos los espadines que conectarán el shield ESC con la tarjeta Avanxe 7 formando de esa manera una PCB multiboard. Además conectamos todas las señales necesarias para controlar los transistores MOSFET a pines de estos espadines.El diseño se muestra en la figura 33.

Figura 33

Etapa de Conexión con la FPGA de la versión 1 del ESC

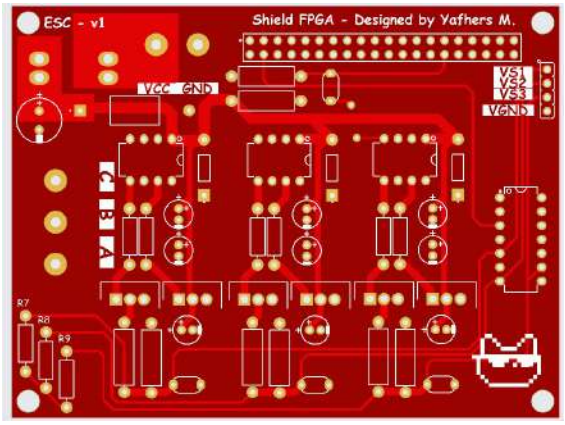


Nota. Elaboración Propia

Luego de completar los diseños esquemáticos, se procedió con el enrutamiento de la tarjeta PCB, obteniéndose como resultado el diseño mostrado en la figura 34.

Figura 34

Versión 1 del Shield ESC

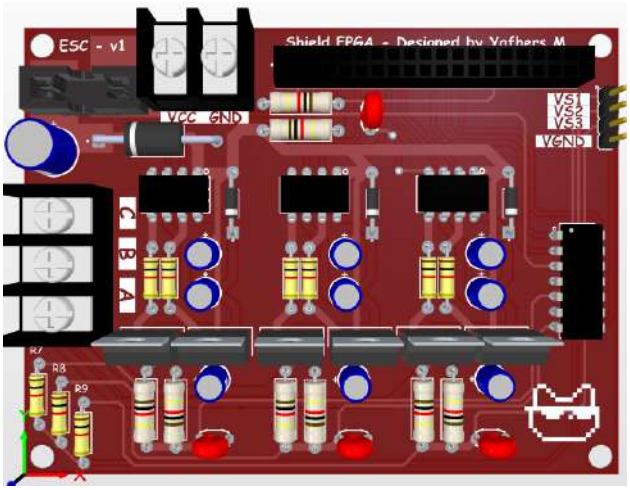


Nota. Elaboración Propia

Posteriormente, en la figura 35 se presenta la visualización tridimensional del PCB con los componentes ubicados, y en la figura 36 se muestra la implementación física de la tarjeta con los componentes ya soldados.

Figura 35

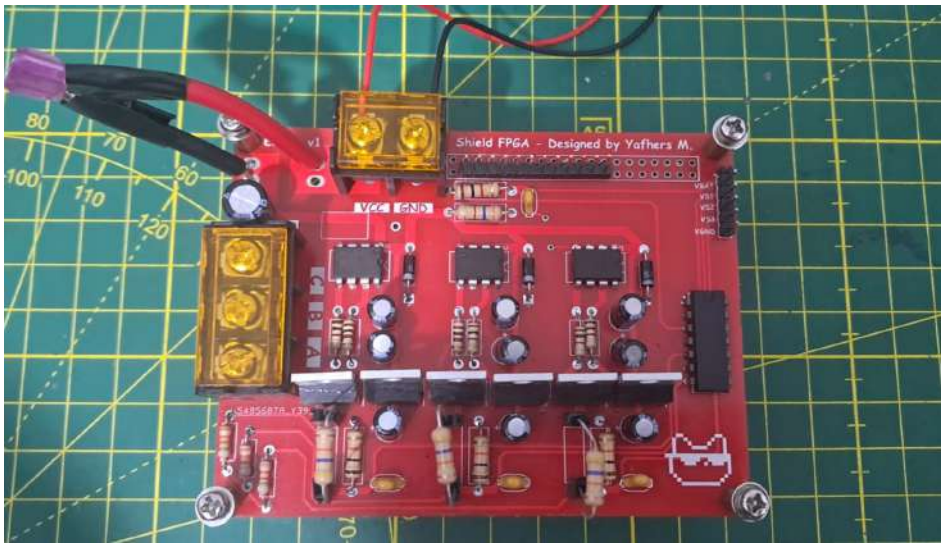
Versión 1 del Shield ESC con componentes en 3D



Nota. Elaboración Propia

Figura 36

Versión 1 del Shield ESC implementado en físico



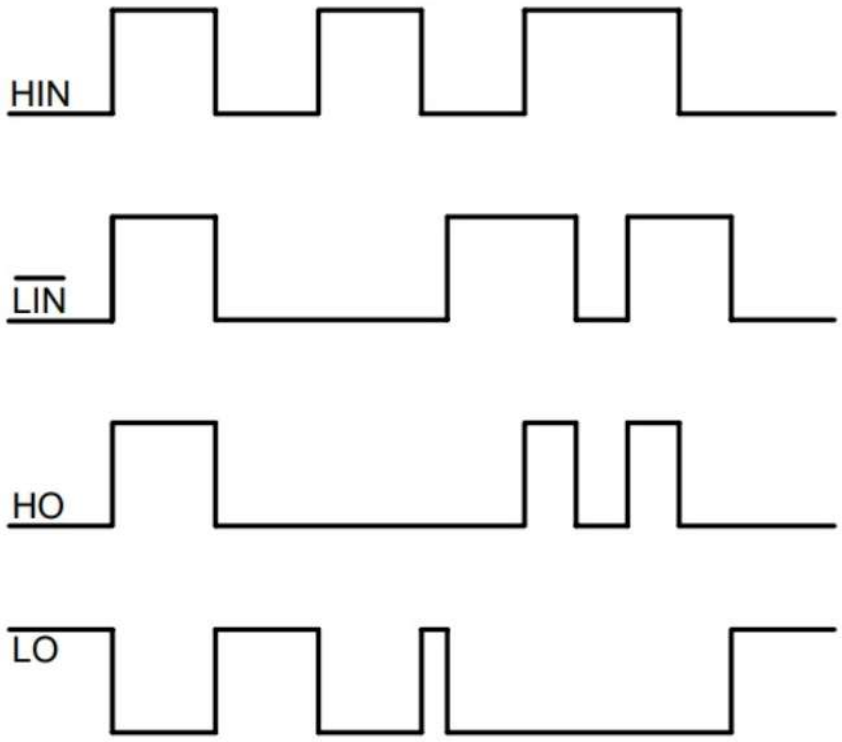
Nota. Elaboración Propia

3.2.2. Pruebas de la versión 1 del shield ESC

Con el diseño completo, se procedió a realizar las pruebas del sistema. Para ello, se consideró la hoja de datos del integrado IR2103 (Rectifier, 2009), el cual se encarga de controlar la activación de los transistores MOSFET según la lógica mostrada en la Figura 37. El objetivo del código desarrollado es controlar las compuertas de los transistores conectados a las salidas HO y LO del driver, mediante una secuencia de seis pasos correspondiente a la conmutación trapezoidal.

Figura 37

Diagrama de Tiempos de Entradas / Salidas para el IR2103



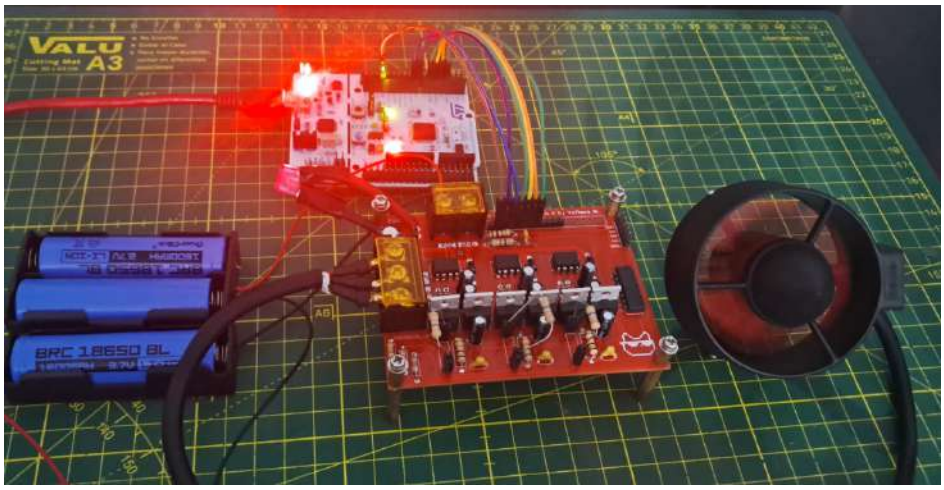
Nota. Extraído del datasheet del IR2103 de Infineon

En la primera etapa de pruebas, se buscó comprobar el correcto funcionamiento de la PCB antes de avanzar hacia un diseño en VHDL, el cual puede resultar más tedioso y demandar mayor tiempo de desarrollo. Por ello, se optó por iniciar utilizando un microcontrolador más sencillo. En este caso, se empleó una tarjeta Nucleo-64 F446, que

incorpora el microcontrolador STM32F446RET6. En dicha plataforma se desarrollaron una serie de funciones y la lógica secuencial (mostrada en el anexo correspondiente) para lograr la activación del motor representado en la Figura 38. Gracias a esta prueba preliminar, se pudo comprobar el correcto funcionamiento del sistema.

Figura 38

Funcionamiento del motor BLDC con el ESC personalizado mediante STM32



Nota. Elaboración Propia

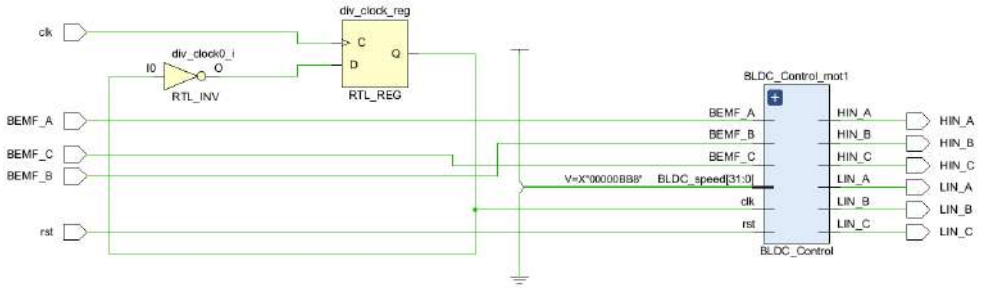
A continuación, se desarrolla un módulo en VHDL que permita replicar las acciones previamente implementadas en el STM32. Para ello, se plantea el diseño mostrado en la Figura 39, cuyo nombre de módulo es `BLDC_Motor`. Este módulo tiene como salidas los pines de activación de compuertas conectados al IR2103, y como entradas las señales de retroalimentación BEMF, además de un parámetro especial: un número entero en el rango de -1000 a 1000, que representa la velocidad deseada del motor BLDC. Un valor positivo indica un giro en sentido antihorario, mientras que un valor negativo corresponde a un giro en sentido horario.

En el diseño también se implementó el uso de paquetes, los cuales son equivalentes a las funciones en lenguaje C. A través de procedimientos (*procedures*) definidos en estos paquetes, se logró simplificar la sintaxis del código VHDL y estructurarlo de manera más clara y modular. Esto facilitó la implementación de la lógica de control mediante una

máquina de estados finita (FSM), mejorando así la legibilidad del diseño.

Figura 39

Funcionamiento del motor BLDC con el ESC personalizado mediante STM32

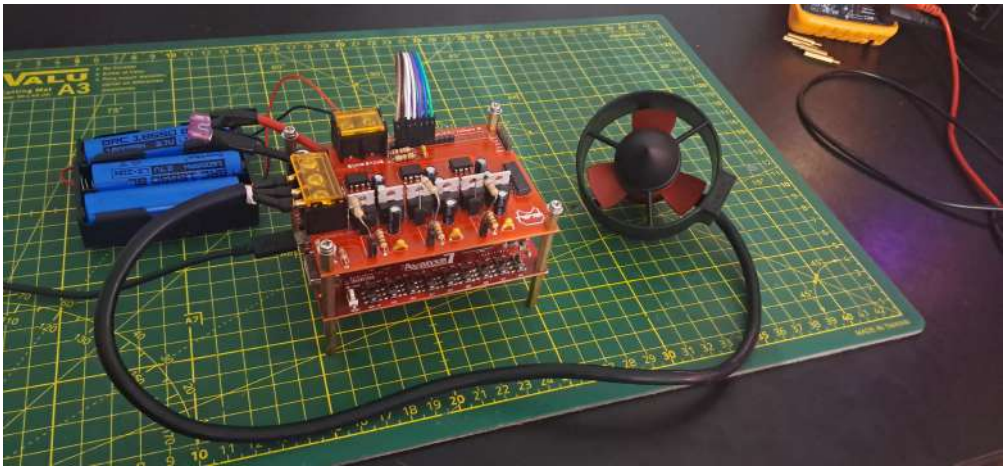


Nota. Elaboración Propia

Finalmente, con el diseño en HDL concluido, se procedió a la implementación en la FPGA, mostrado en la Figura 40. Para ello, se asignaron los pines correspondientes a cada señal del ESC personalizado y se montó la tarjeta sobre la placa de desarrollo Avanxe 7, utilizándola como un shield. Posteriormente, se conectó una batería de 3S y el propulsor BLDC T60 de 860KV, logrando así el correcto funcionamiento del sistema, evidenciado por el giro del motor de acuerdo con la velocidad proporcionada al módulo.

Figura 40

Funcionamiento del motor BLDC con el ESC personalizado mediante FPGA



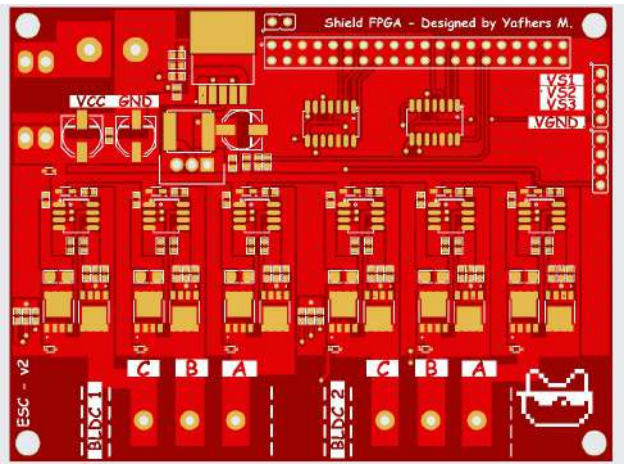
Nota. Elaboración Propia

3.2.3. Diseño y pruebas de la versión 2 del shield ESC

Luego de verificar el correcto funcionamiento de la primera versión del ESC basada en el método trapezoidal, se decidió realizar varias mejoras respecto al diseño anterior. En la versión inicial, el uso de componentes THT de gran tamaño limitó la implementación a un solo ESC por PCB. Sin embargo, considerando que la aplicación final está orientada al control de múltiples motores BLDC en un ROV, esta configuración resultaba insuficiente. Por ello, en esta nueva versión mostrada en la figura 41 se optó por el uso de componentes SMD, lo que permitió una significativa reducción del tamaño del circuito y facilitó la integración de dos ESC en una misma tarjeta PCB.

Figura 41

Versión 2 del Shield ESC



Nota. Elaboración Propia

Esta nueva versión implementa varias mejoras respecto a la versión anterior, entre ellas destacan las siguientes características:

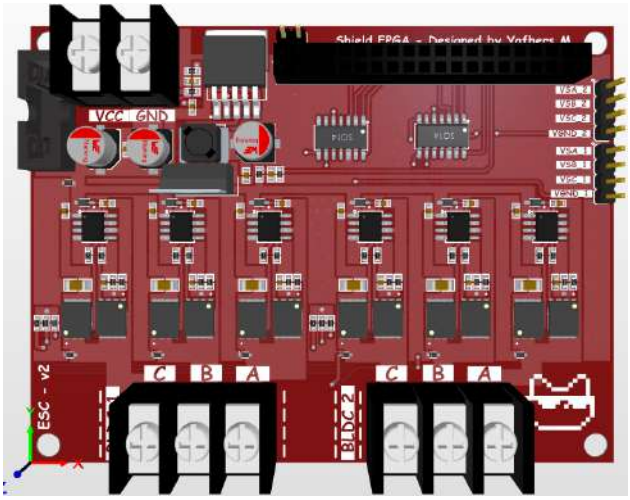
- En la versión 1 se utilizaron componentes THT, lo que limitó la posibilidad de incluir más de un ESC por PCB. En la versión 2, todos los componentes fueron reemplazados por sus equivalentes en formato SMD, permitiendo reducir el espacio ocupado y diseñar dos ESC en una misma tarjeta PCB.

- Se incorporó un convertidor reductor de voltaje (Buck Converter) basado en el XL4015E1, el cual permite reducir el voltaje de la batería a 5V de forma eficiente, lo que posibilita alimentar directamente la FPGA desde la misma batería. El diseño de esta etapa se realizó siguiendo las recomendaciones del fabricante XLSEMI.
- Se reemplazó el controlador de compuertas IR2101 por el IR2103, que ofrece un mejor rendimiento y características mejoradas.
- Se sustituyó el MOSFET IRFZ44N por el SIR182DP, el cual soporta hasta 60 A y presenta un formato más compacto, optimizando el diseño térmico y espacial.
- Se incorporó una funcionalidad adicional que permite alimentar la FPGA directamente desde la batería, mediante un espadín de selección que permite activar o desactivar esta opción según sea necesario.

En la figura 42 se presenta la visualización tridimensional del PCB con los componentes ubicados.

Figura 42

Versión 2 del Shield ESC implementado en físico



Nota. Elaboración Propia

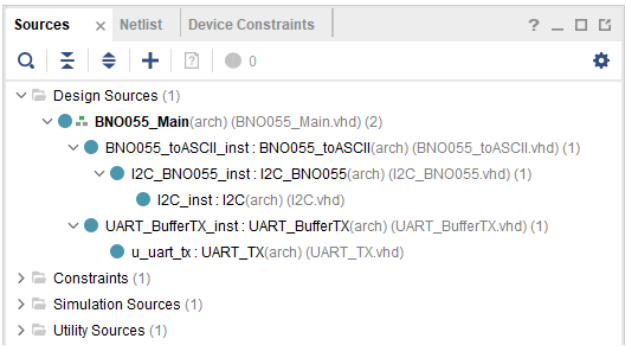
3.3. Diseño de módulos I2C en VHDL para lectura del sensor BNO055

En esta sección se desarrolla el diseño de módulos en VHDL necesarios para la lectura del sensor inercial BNO055. Este MEMS se comunica mediante el uso del protocolo I2C, por lo que es necesario diseñar módulos en VHDL capaces de leer los registros I2C del sensor BNO055 y extraer los valores requeridos para el cálculo de la inclinación según la hoja de datos del sensor (GmbH, 2024). Adicionalmente, con el fin de visualizar los datos desde la FPGA, se diseñarán también módulos que implementen la transmisión de datos a través de la interfaz de hardware UART.

En total, se diseñarán seis módulos destinados a la lectura de registros del sensor BNO055 mediante I2C y a la posterior transmisión de dichos datos mediante la interfaz de hardware UART. Estos módulos se presentan en la figura 43, y han sido implementados usando el entorno de desarrollo Vivado 2025.1. Al final de esta sección, se presentarán las pruebas experimentales realizadas con el sensor para validar su funcionamiento.

Figura 43

Módulos diseñados en VHDL para la lectura del sensor BNO055



Nota. Elaboración Propia. Captura tomada desde Vivado

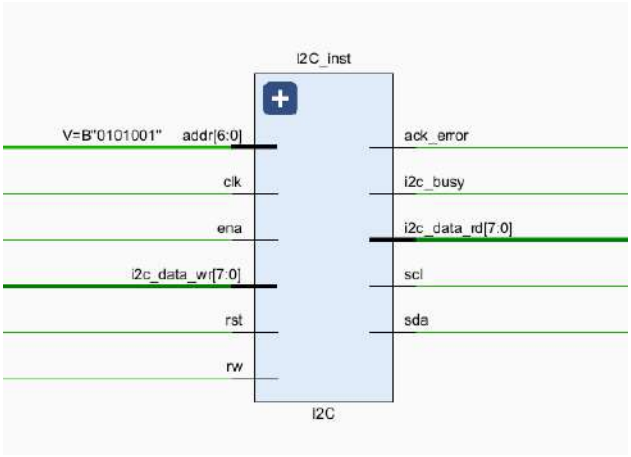
3.3.1. Módulos dedicados a la lectura del sensor BNO055 mediante el protocolo de comunicación I2C

Inicialmente, se diseñaron tres módulos fundamentales destinados a la lectura de los ángulos de inclinación proporcionados por el sensor BNO055. A continuación, se describen las funciones de estos módulos:

I. Módulo I2C: El módulo mostrado en la figura 44 implementa la comunicación I2C en modo maestro, permitiendo el envío y la recepción de datos con dispositivos esclavos a una velocidad de 500KHz. Para ello, genera la señal de reloj I2C a partir del oscilador de 100MHz de la tarjeta Avanxe 7. Además, gestiona el control de las líneas SDA y SCL, mientras que una máquina de estados finita (FSM) se encarga de coordinar las distintas fases del protocolo I2C, como el inicio, la dirección, la lectura/escritura y la detención de la transmisión.

Figura 44

Diseño RTL del módulo I2C



Nota. Elaboración Propia. Captura tomada desde Vivado

II. Módulo I2C_BNO055: El módulo mostrado en la figura 46 implementa una máquina de estados secuencial en VHDL para gestionar la comunicación con el sensor BNO055 mediante el protocolo I2C. Su función principal es inicializar el sensor en el modo NDOF (fusión completa de sensores) y realizar la lectura continua de los valores de orientación: yaw, roll y pitch. Para ello, este módulo se apoya en el módulo maestro I2C previamente desarrollado, sobre el cual construye la lógica de control necesaria para enviar comandos de configuración, acceder a los registros internos del sensor y extraer los datos de orientación en tiempo real.

Para realizar el diseño de este módulo se consulto a la hoja de datos del sensor

BNO055, obtenida en la página oficial de la empresa Bosch Sensortec (Sensortec, 2020). El diseño consiste en configurar el sensor BNO055 para luego leer los registros que contienen los valores de los ángulos de Euler.

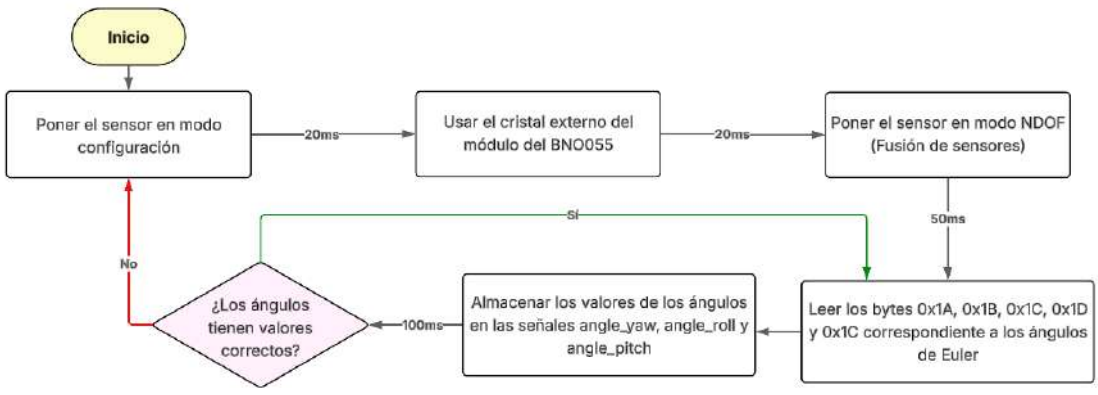
Cuadro 9

Registros I2C del sensor BNO055 - Bosch Sensortec

Dirección	Nombre	Contenido
$0x3D$	OPR_MODE	Modo de operación del sensor
$0x3F$	$SYS\_TRIGGER$	Configuración del cristal
$0x1A$	EUL_Yaw_LSB	Datos del ángulo YAW $< 7:0 >$
$0x1B$	EUL_Yaw_MSB	Datos del ángulo YAW $< 15:8 >$
$0x1C$	EUL_Roll_LSB	Datos del ángulo ROLL $< 7:0 >$
$0x1D$	EUL_Roll_MSB	Datos del ángulo ROLL $< 15:8 >$
$0x1E$	EUL_Pitch_LSB	Datos del ángulo PITCH $< 7:0 >$
$0x1F$	EUL_Pitch_MSB	Datos del ángulo PITCH $< 15:8 >$

Figura 45

Diagrama de flujo para lectura del sensor BNO055



Nota. Elaboración Propia, hecho en LucidChart

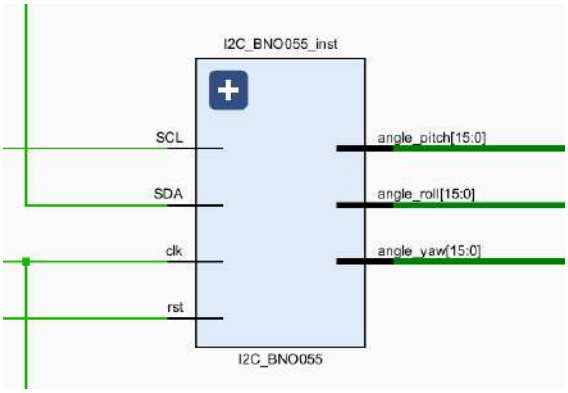
En el diseño del módulo se elaboró una máquina de estados para interactuar con cada uno de los registros mostrados en la tabla 9. La máquina de estados mostrada en la

figura 45 realiza la siguiente secuencia:

- Paso 1: Colocar el sensor en modo configuración escribiendo $0x00$ en el registro $0x3D$.
- Paso 2: Activar el uso del cristal externo escribiendo $0x80$ en el registro $0x3F$.
- Paso 3: Colocar el sensor en modo NDOF escribiendo $0x0C$ en el registro $0x3D$. En este modo activamos el giroscopio, el acelerómetro y el magnetómetro, este modo es conocido como modo de fusión de sensores.
- Paso 4: Leer 6 registros desde la dirección $0x1A$, a partir de este registro se encuentran los 3 ángulos de Euler, cada uno ocupa 2 registros ya que el valor del ángulo ocupa 16 bits.
- Paso 5: Comprobar si el sensor sigue funcionando correctamente, y volver a leer los ángulos secuencialmente durante todo el proceso para realizar el control de los motores.

Figura 46

Diseño RTL del módulo I2C_BNO055



Nota. Elaboración Propia. Captura tomada desde Vivado

III. Módulo BNO055_toASCII El módulo con diseño RTL mostrado en la figura 47 tiene como función principal leer los datos del sensor BNO055 a través de la interfaz I2C, utilizando para ello el módulo #2 previamente descrito. Posteriormente,

convierte los valores de los ángulos yaw, roll y pitch en cadenas de caracteres ASCII en formato hexadecimal. Esta conversión resulta especialmente útil para la visualización de los datos en un monitor serial, usado para la comunicación con la estación terrena.

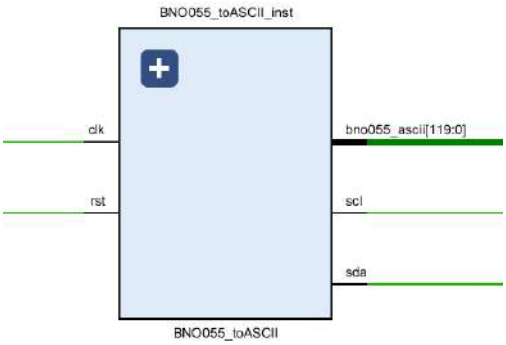
Para llevar a cabo el diseño, cada ángulo de 16 bits es dividido en 4 grupos de 4 bits (nibbles), los cuales se convierten a su representación en código ASCII hexadecimal de 8 bits, obteniendo así una cadena de 32 bits por cada ángulo. Al finalizar el proceso de conversión, se construye una trama con el siguiente formato:

$$< YAW\_HEX >, < ROLL\_HEX >, < PITCH\_HEX > \backslash \text{n}.$$

Esta trama estructurada está destinada a ser enviada a los módulos responsables de la transmisión de datos mediante la interfaz UART.

Figura 47

Diseño RTL del módulo BNO055_toASCII



Nota. Elaboración Propia. Captura tomada desde Vivado

3.3.2. Módulos dedicados a la exportación de datos del sensor BNO055 mediante la interfaz de hardware UART

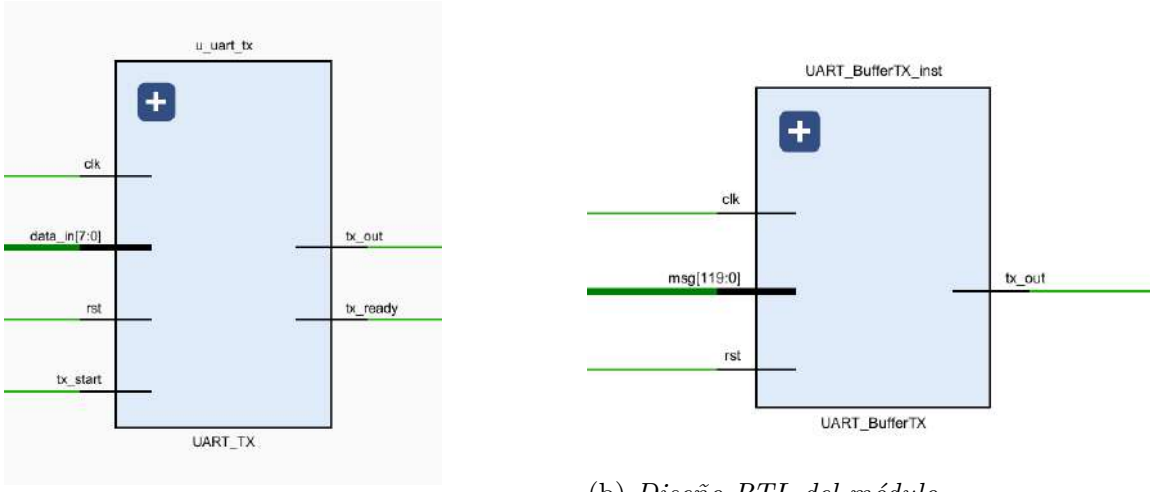
Luego del diseño de los módulos dedicados a la lectura de datos mediante el protocolo I2C desde el sensor BNO055, el siguiente paso consiste en extraer dicha información para su transmisión hacia una estación terrena o su visualización local. Para ello, se emplea la interfaz de comunicación UART, que permite el envío de los datos de forma serial.

IV. Módulo UART_TX El módulo con diseño RTL mostrado en la Figura 48a tiene como función implementar un transmisor UART en VHDL, configurado para una transmisión de 8 bits de datos, 1 bit de inicio, 1 bit de parada, y una velocidad de 9600 baudios. Su propósito es enviar un byte de manera serial a través de un pin de la FPGA. Además, este módulo incorpora señales de activación y de indicación de disponibilidad del transmisor.

Este módulo es importante para establecer comunicación serial entre la FPGA y periféricos UART, como microcontroladores o sensores. En este proyecto, se utiliza para enviar los datos al monitor serial y visualizar los ángulos de inclinación del sensor.

Figura 48

Diseño de módulos RTL dedicados a la transmisión UART



(a) *Diseño RTL del módulo UART_TX*

(b) *Diseño RTL del módulo UART_BufferTX*

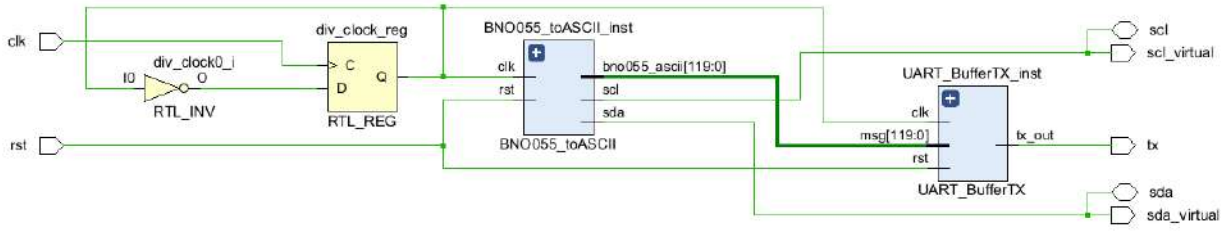
Nota. Elaboración propia. Captura tomada desde Vivado.

V. Módulo UART_BufferTX El módulo con diseño RTL mostrado en la Figura 48b tiene como objetivo transmitir de manera serial cadenas de texto, utilizando como base el módulo presentado previamente, el cual permite enviar un único byte. Este diseño extiende dicha funcionalidad, permitiendo la transmisión de múltiples bytes a partir de un vector de entrada (msg) que representa una secuencia de caracteres codificados en ASCII.

VI. Módulo BNO055_Main Este es el módulo principal, cuyo diseño RTL se muestra en la Figura 49. Su función es integrar y coordinar todos los módulos desarrollados previamente, gestionando la comunicación tanto por el bus I2C como por la interfaz UART. Además, se encarga de exportar las señales correspondientes hacia los pines físicos de la FPGA, permitiendo así la interacción con el sensor BNO055 y el sistema de visualización o transmisión de datos.

Figura 49

Diseño RTL del módulo BNO055_Main



Nota. Elaboración Propia. Captura tomada desde Vivado

Entre las señales utilizadas en la implementación sobre la FPGA usadas en Vivado como se muestra en la figura 50 se tienen las siguientes:

- clk: Señal de entrada proveniente del cristal de 100 MHz de la tarjeta Avanxe 7. Se utiliza para establecer la frecuencia de operación de los diferentes módulos.
- rst: Señal de entrada proveniente de uno de los botones de la tarjeta Avanxe 7. Permite reiniciar el funcionamiento del sistema al activarse.
- tx: Señal de salida correspondiente a la línea UART. Se emplea para enviar los bytes que contienen los ángulos de inclinación hacia un dispositivo externo.
- scl y sda: Señales utilizadas para establecer la comunicación I2C con el sensor BNO055. scl corresponde al reloj serial y sda a la línea de datos.

Figura 50

Asignación de pines en la tarjeta Avanxe 7 mediante Vivado

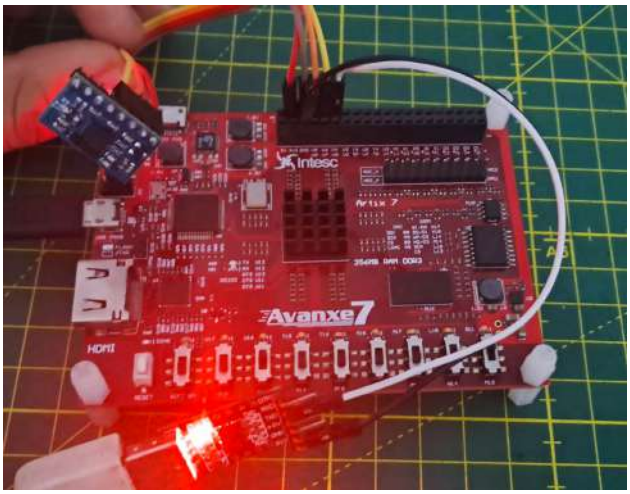
Nota. Elaboración Propia. Captura tomada desde Vivado

3.3.3. Pruebas de los módulos del sensor BNO055

Para verificar el funcionamiento del diseño RTL, se conectaron el sensor BNO055 y un adaptador USB-Serial con chip CH340, permitiendo la lectura de los bytes transmitidos por la FPGA vía UART. La conexión se muestra en la Figura 51.

Figura 51

Conexión del BNO055 y Adaptador CH340 a la tarjeta Avanxe 7

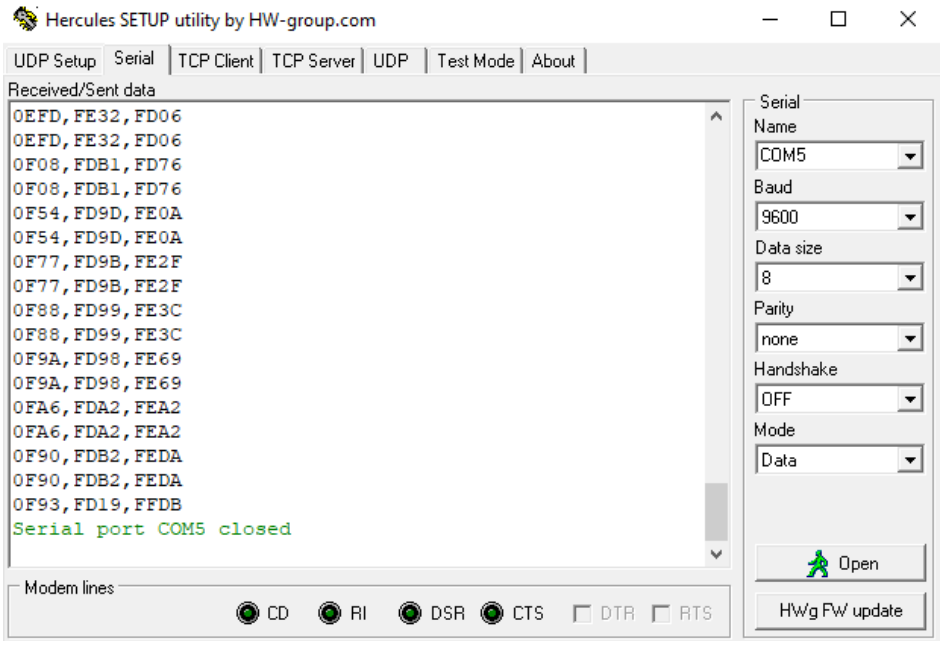


Nota. Elaboración Propia

Para visualizar los datos de inclinación obtenidos del sensor BNO055, se empleó el software Hercules, una herramienta de terminal serial que permite recibir y mostrar datos transmitidos mediante UART. A través de esta interfaz, fue posible observar en tiempo real los valores de ángulos —yaw, pitch y roll— enviados desde la FPGA en formato ASCII, tal como se muestra en la Figura 52. Esto facilitó la validación del correcto funcionamiento tanto del sistema de transmisión como del proceso de conversión de datos.

Figura 52

Visualización de ángulos del sensor BNO055 en Hercules



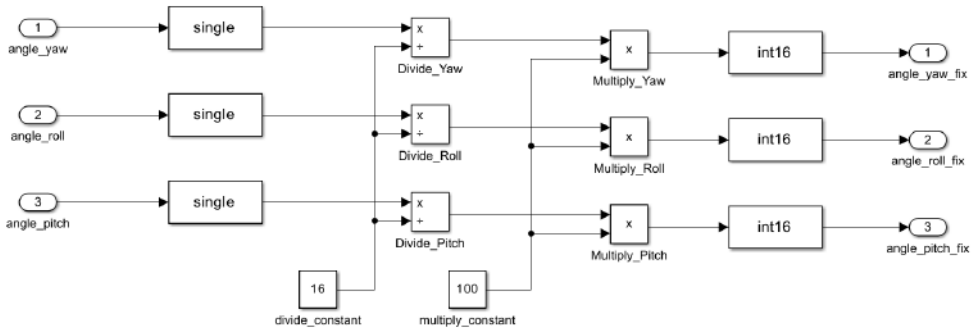
Nota. Elaboración Propia. Captura tomada desde Hercules

Los ángulos mostrados aún no se encuentran escalados, ya que, según la hoja de datos, es necesario dividir el valor de 16 bits entre 16 para obtener el ángulo en grados sexagesimales con parte decimal. El uso de números en punto flotante en una FPGA implica un consumo considerable de recursos, y la implementación de divisores también puede ser compleja. Por esta razón, se optó por utilizar herramientas de generación automática de código HDL. En este proyecto, se empleó HDL Coder de Simulink para describir el sistema y generar el código VHDL encargado del escalamiento de los ángulos.

Por ello, en primer lugar se diseñó el diagrama mostrado en la Figura 53.

Figura 53

Diseño de diagrama en Simulink para corregir los valores de los ángulos

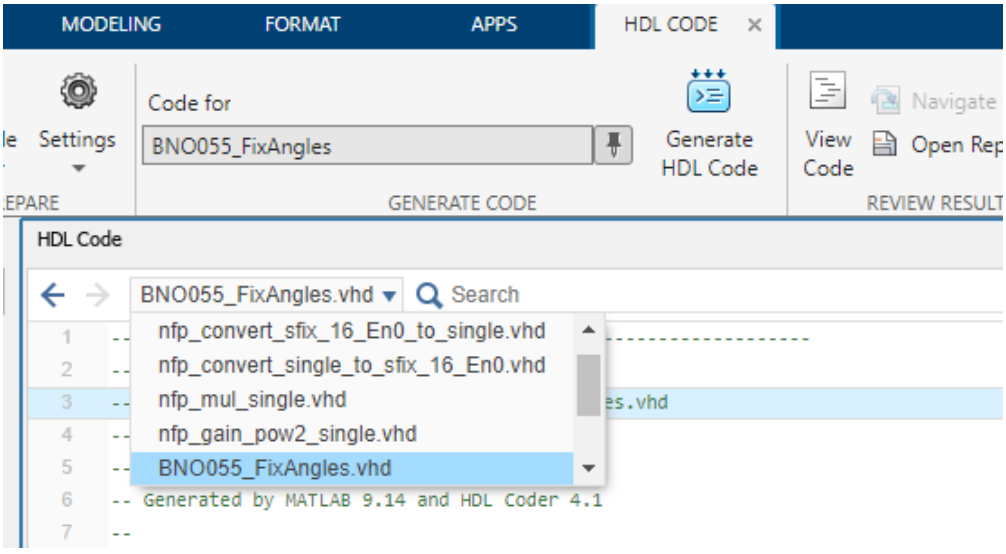


Nota. Elaboración Propia. Captura tomada desde Simulink

Una vez diseñado el diagrama en Simulink, se utilizó HDL Coder para generar los módulos HDL mostrados en la Figura 54. Estos pueden integrarse directamente al diseño en la FPGA para obtener los ángulos ya escalados. La implementación del módulo en Vivado, denominado BNO055_FixAngles, se muestra en la Figura 55.

Figura 54

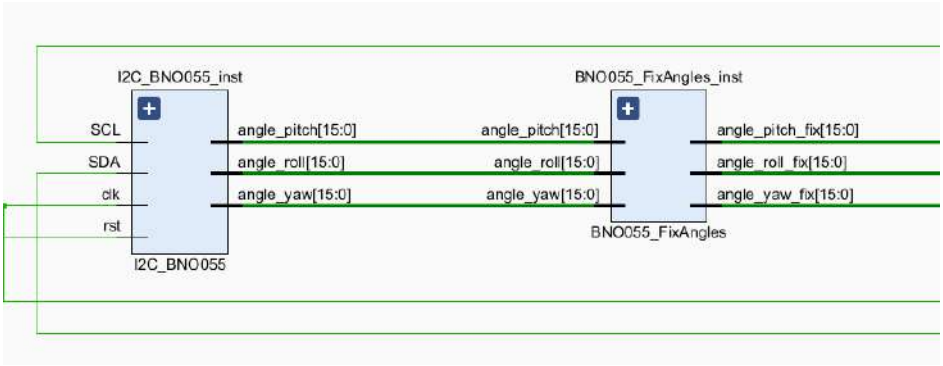
Archivos VHDL generador por HDL Coder para corrección de ángulos



Nota. Elaboración Propia. Captura tomada desde Simulink

Figura 55

Módulo de escalamiento de ángulos implementado en Vivado

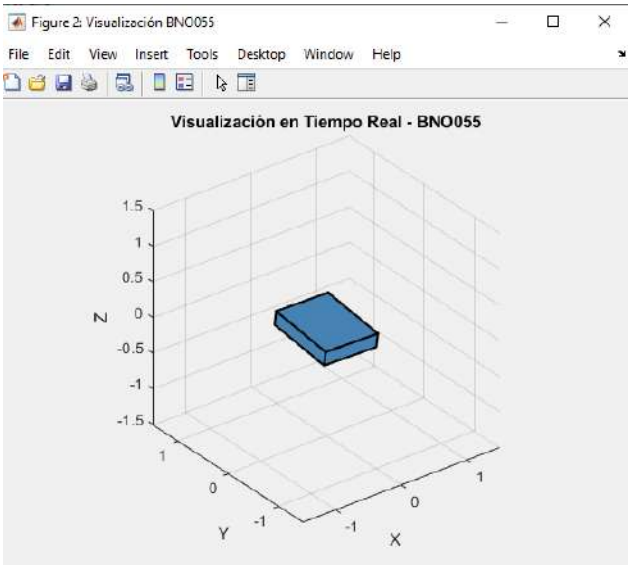


Nota. Elaboración Propia. Captura tomada desde Vivado

Finalmente, como prueba final de precisión, se desarrolló un script en MATLAB (mostrado en el anexo), el cual toma los ángulos en grados sexagesimales enviados desde la FPGA y, mediante una animación en tiempo real, permite visualizar un paralelepípedo que se inclina conforme lo hace el sensor BNO055 como se observa en la Figura 56.

Figura 56

Visualización en 3D de un paralelepípedo que se inclina según un sensor BNO055



Nota. Elaboración Propia. Captura tomada desde MatLab

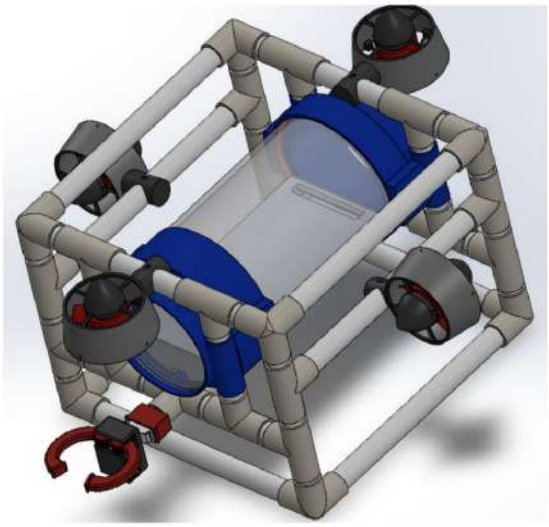
3.4. Desarrollo de interfaz de control para los movimientos del ROV

Todo ROV requiere de una estación en superficie donde un piloto se encarga de maniobrar y supervisar su funcionamiento. Por ello, en esta sección se desarrollará una interfaz de usuario didáctica que permitirá enviar instrucciones de movimiento a la FPGA. A partir de estas instrucciones, se definirán los ángulos de inclinación deseados, los cuales activarán el sistema de control de los propulsores.

Los movimientos que puede realizar un ROV dependen del número y la disposición de sus propulsores. En este proyecto se ha considerado un diseño con cuatro propulsores distribuidos alrededor del chasis en forma de cruz, como se muestra en la Figura 57. Esta configuración, conocida como "tipo cruz", es ampliamente utilizada en prototipos por su sencillez y facilidad de implementación (X. Huang et al., 2019).

Figura 57

ROV con 4 propulsores en configuración "tipo cruz"



Nota. Tomado de diseño hecho por el equipo Neptune Knights de la Universidad Central de Florida

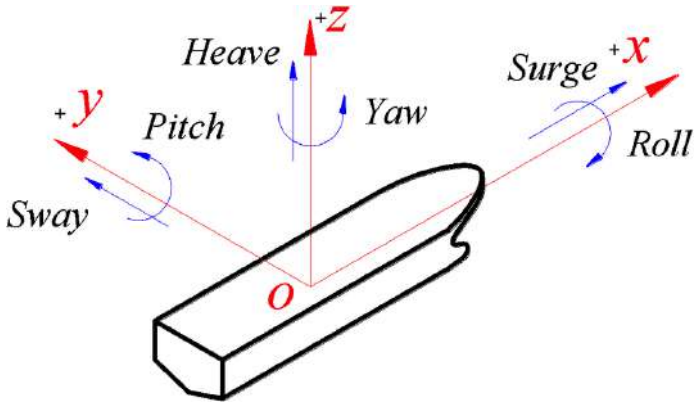
La cantidad de movimientos que un ROV puede realizar normalmente es de seis, los cuales se ilustran en la Figura 58.

- Surge: Movimiento hacia adelante / atrás (eje X)

- Sway: Movimiento hacia izquierda / derecha (eje Y)
- Heave: Movimiento hacia arriba / abajo (eje Z)
- Roll: Giro sobre el eje X
- Pitch: Giro sobre el eje Y
- Yaw: Giro sobre el eje Z

Figura 58

Movimientos Rotacionales y Translacionales



Nota. Tomado de publicación de Roy de Winter

Para que un ROV sea capaz de ejecutar los seis grados de libertad, se requerirían al menos seis propulsores. No obstante, en este diseño se ha optado por una configuración tipo cruz con solo cuatro propulsores, lo cual nos permite realizar únicamente cuatro de esos movimientos: surge, heave, pitch y yaw.

En este proyecto se ha decidido implementar el control únicamente sobre tres de ellos: surge, heave y yaw, con el fin de simplificar el sistema y optimizar el tiempo de desarrollo. Sin embargo, el sistema ha sido diseñado de manera modular, por lo que es posible incorporar más propulsores y extender el control a los otros grados de libertad, en caso se requiera una mayor maniobrabilidad.

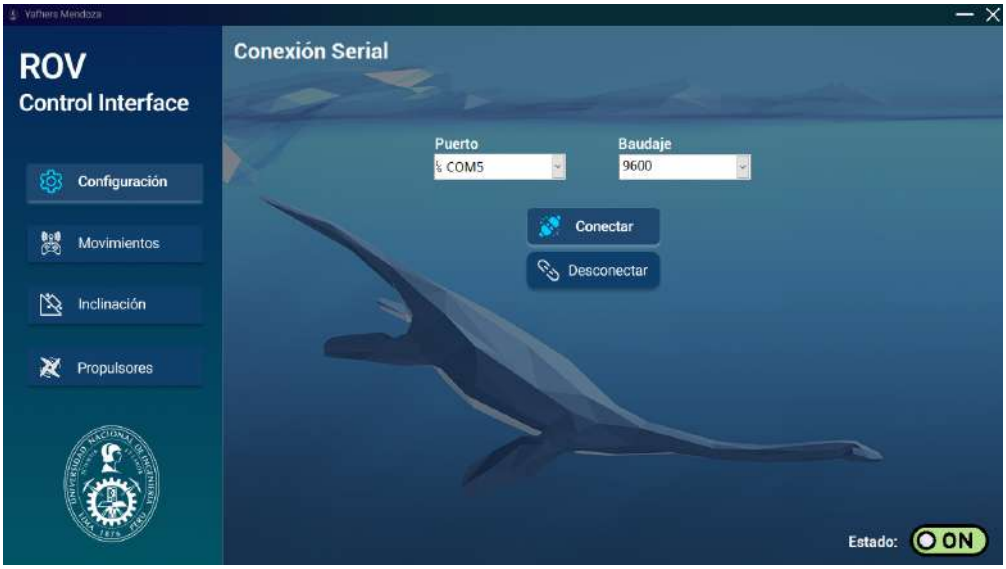
Con lo anteriormente mencionado, es necesario desarrollar una interfaz que permita ejecutar los tres movimientos seleccionados. Para ello, se utilizó el software NI LabVIEW,

versión 2024 Q1. Esta herramienta fue elegida por su capacidad para integrarse de forma eficiente con hardware y su rendimiento en aplicaciones de comunicación en tiempo real. A través de LabVIEW se diseñó una interfaz capaz de enviar comandos directamente a la FPGA, de acuerdo con las instrucciones proporcionadas por el operador del ROV.

El diseño final consta de cuatro ventanas, siendo las dos primeras las más relevantes para el control del sistema. En la primera ventana, ilustrada en la Figura 59, el usuario puede seleccionar el puerto serial correspondiente a la interfaz UART implementada en la FPGA. Esta sección también permite establecer la conexión o desconexión con dicho puerto, así como configurar diferentes velocidades de transmisión.

Figura 59

Ventana de configuración de interfaz de control del ROV



Nota. Elaboración propia. Tomado de NI LabVIEW

En la siguiente ventana, mostrada en la Figura 60, se presenta la interfaz principal de control del ROV. Para facilitar su uso y hacerlo más accesible, se optó por implementar el envío de instrucciones a través del teclado de la computadora, permitiendo al operador manejar el sistema de forma rápida e intuitiva.

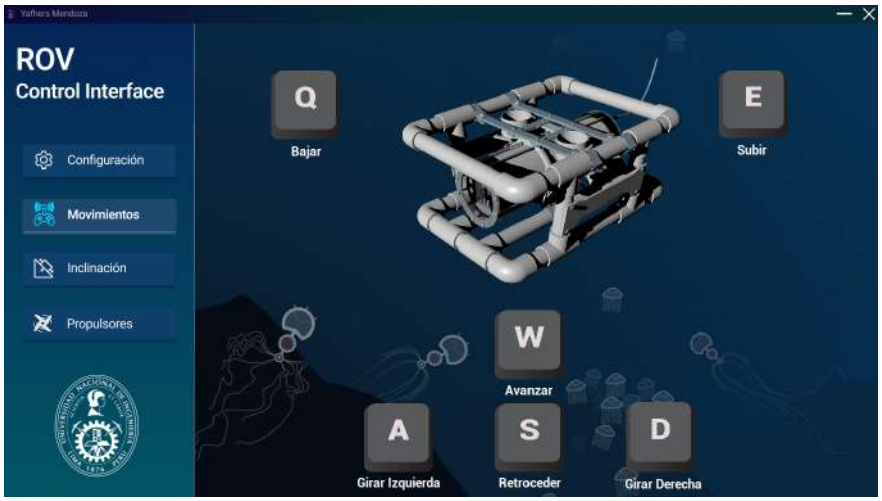
Se optó por utilizar seis teclas del teclado para controlar los tres movimientos seleccionados del ROV. Cada una de estas teclas está asociada a una acción específica,

como se detalla a continuación:

- **Tecla W:** Movimiento de *surge* hacia adelante (avanzar).
- **Tecla S:** Movimiento de *surge* hacia atrás (retroceder).
- **Tecla Q:** Movimiento de *heave* hacia abajo (descender).
- **Tecla E:** Movimiento de *heave* hacia arriba (ascender).
- **Tecla A:** Giro sobre el eje Z hacia la izquierda (*yaw* negativo).
- **Tecla D:** Giro sobre el eje Z hacia la derecha (*yaw* positivo).

Figura 60

Ventana de control de movimiento del ROV

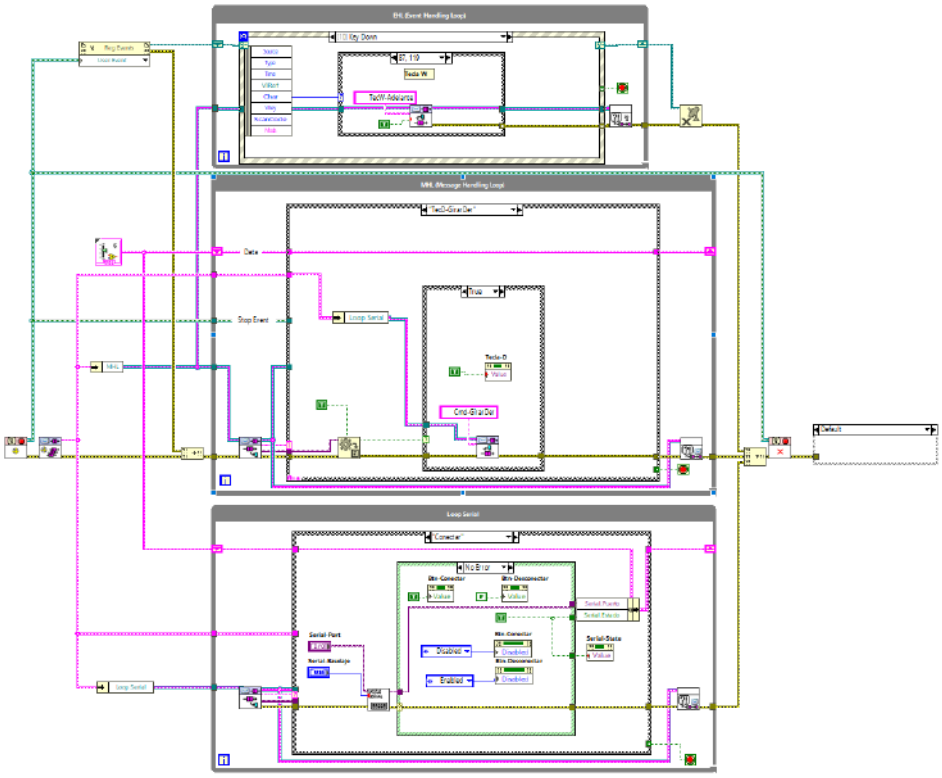


Nota. Elaboración propia. Tomado de NI LabVIEW

Todo el sistema ha sido desarrollado utilizando la arquitectura QMH (Queued Message Handler) en LabVIEW. Esta arquitectura, representada en la Figura 61, se basa en el uso de colas y el manejo estructurado de eventos, lo que permite una comunicación eficiente y de alta velocidad entre los diferentes módulos. Gracias a este enfoque, se logra una gestión simultánea de múltiples eventos con una mínima probabilidad de errores en la transmisión o procesamiento de datos (National Instruments, 2012).

Figura 61

Arquitectura QMH empleada en el diseño de la interfaz de control



Nota. Elaboración propia. Tomado de NI LabVIEW

A continuación, se calculan los ángulos de inclinación deseados según el movimiento indicado por el usuario. Estos permiten determinar el error y su variación, que son las entradas del sistema de control difuso. A partir de ellas, se estiman las velocidades de los motores BLDC para ejecutar el movimiento de forma estable. La incorporación del sistema de control permite que el ROV mantenga su estabilidad ante posibles perturbaciones externas. Por ello, es fundamental estimar correctamente los ángulos de inclinación deseados, ya que de estos depende la precisión y el desempeño del sistema de control.

La Tabla 10 muestra el criterio de acción correspondiente para cada comando recibido por la FPGA, especificando la respuesta del sistema ante cada instrucción de movimiento enviada desde la interfaz de control.

Cuadro 10

Ángulos Deseados para cada comando de movimiento del ROV

Comando	Movimiento	Ángulo deseado
W	Surge adelante	Yaw_Deseado = Yaw _Actual Pitch_Deseado = Pitch_Actual
S	Surge atrás	Yaw_Deseado = Yaw _Actual Pitch_Deseado = Pitch_Actual
A	Yaw izquierda	Yaw_Deseado = Yaw _Actual $-10°$ Pitch_Deseado = Pitch_Actual
D	Yaw derecha	Yaw_Deseado = Yaw _Actual $+10°$ Pitch_Deseado = Pitch_Actual
Q	Heave abajo	Yaw_Deseado = Yaw _Actual Pitch_Deseado = Pitch_Actual
E	Heave arriba	Yaw_Deseado = Yaw _Actual Pitch_Deseado = Pitch_Actual

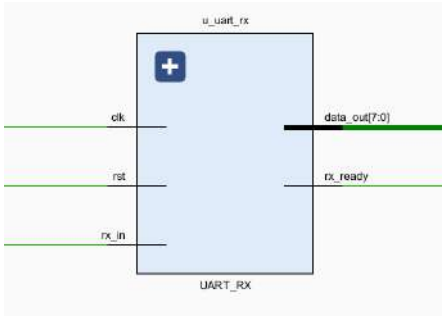
Antes de proceder con el diseño de los módulos en la FPGA, se implementó una funcionalidad adicional en la interfaz de LabVIEW: al presionar una tecla, se envía el carácter del comando en mayúscula, y al soltarla, se envía el mismo carácter en minúscula. Esta estrategia permite identificar de forma precisa el inicio y fin de cada movimiento solicitado por el usuario.

3.4.1. Módulos dedicados a la recepción de comandos de la estación terrena del ROV mediante el uso de la interfaz UART

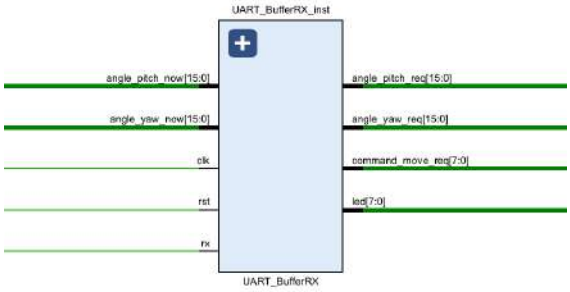
En esta sección se describe el funcionamiento de los módulos implementados para la recepción de instrucciones de movimiento. A partir de dichas instrucciones, el sistema realiza el cálculo de los ángulos de inclinación deseados que servirán como referencia para el sistema de control.

Figura 62

Diseño de módulos RTL dedicados a la recepción UART



(a) *Diseño RTL del módulo UART_RX*



(b) *Diseño RTL del módulo UART_BufferRX*

Nota. Elaboración propia. Captura tomada desde Vivado.

I. Módulo UART_RX El diseño RTL mostrado en la Figura 62a tiene como objetivo recibir un byte a través de la interfaz UART, correspondiente a un comando de movimiento según la Tabla 10. Incluye sincronización de baudaje y control de la línea RX mediante una máquina de estados.

II. Módulo UART_RX El diseño RTL mostrado en la Figura 62b tiene como objetivo gestionar la recepción de bytes mediante el módulo UART previamente descrito. Según el comando recibido, determina el ángulo de inclinación deseado y lo envía al sistema de control difuso.

Entre las funciones principales que realiza este módulo se encuentran:

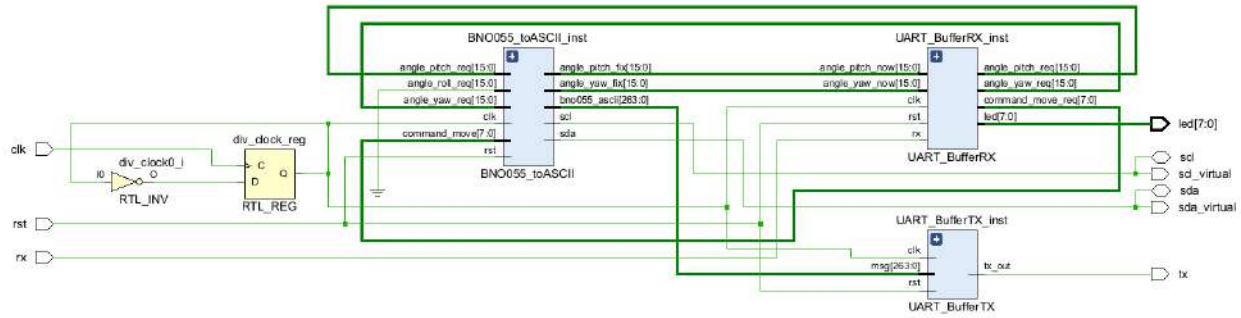
- Recibe los ángulos actuales del ROV, en caso de que el cálculo del ángulo deseado deba realizarse con respecto a su valor actual.
- Genera como salida los ángulos deseados junto con la instrucción correspondiente al comando actual. Estos cuatro valores son entregados al sistema de control difuso, el cual calculará las velocidades necesarias para cada motor BLDC.
- Activa los LED integrados en la tarjeta Avanxe 7 para indicar el comando activo.

- Se limita el control a los ángulos *yaw* y *pitch*, debido a la restricción en el número de propulsores disponibles en la estructura del ROV.

Como consecuencia de la inclusión de estos módulos, fue necesario modificar otros bloques del sistema para permitir la impresión de los cuatro nuevos valores correspondientes a los ángulos deseados a través del monitor serial. El diseño RTL completo de la FPGA hasta este punto se muestra en la Figura 63.

Figura 63

Diseño RTL completo con el módulo de recepción de comandos



Nota. Elaboración propia. Captura tomada desde Vivado

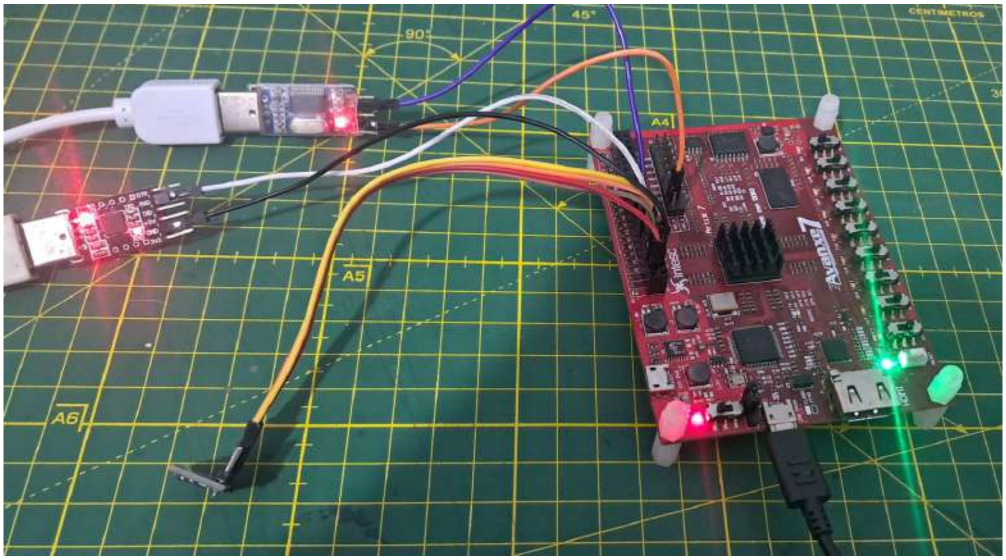
3.4.2. Prueba de los módulos de recepción de comandos

Para la validación de los módulos diseñados, se empleó *MatLab* como herramienta de verificación de los datos transmitidos. Las conexiones realizadas se ilustran en la Figura 64, donde se utilizó un puerto UART para la transmisión de datos hacia *MATLAB* y otro puerto UART dedicado a la recepción de comandos provenientes de la estación terrena.

Además, se desarrolló un *script* en *MatLab* (incluido en el Anexo), cuyo resultado se muestra en la Figura 65. Dicho *script* permite recibir las tramas enviadas por la FPGA a través del puerto serial e interpretar los datos en tiempo real. De esta manera, se pueden visualizar e imprimir los siguientes valores: los ángulos actuales obtenidos del sensor BNO055, los ángulos deseados generados por el HMI de control, y el comando recibido correspondiente al movimiento solicitado.

Figura 64

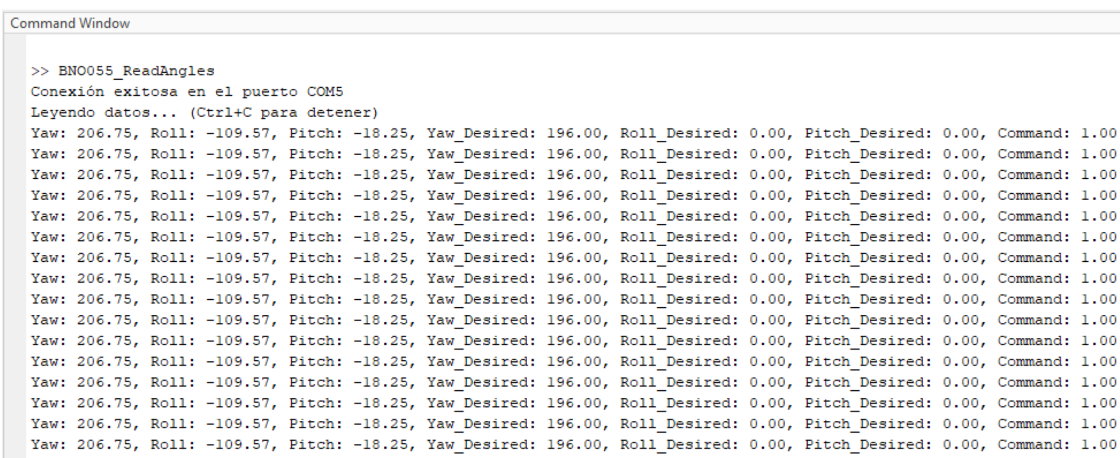
Conexiones realizadas para la prueba de recepción de datos con Avanxe 7



Nota. Elaboración propia

Figura 65

Script de lectura de datos del puerto serial de la FPGA en MatLab



```
Command Window

>> BNO055_ReadAngles
Conexión exitosa en el puerto COM5
Leyendo datos... (Ctrl+C para detener)
Yaw: 206.75, Roll: -109.57, Pitch: -18.25, Yaw_Desired: 196.00, Roll_Desired: 0.00, Pitch_Desired: 0.00, Command: 1.00
Yaw: 206.75, Roll: -109.57, Pitch: -18.25, Yaw_Desired: 196.00, Roll_Desired: 0.00, Pitch_Desired: 0.00, Command: 1.00
Yaw: 206.75, Roll: -109.57, Pitch: -18.25, Yaw_Desired: 196.00, Roll_Desired: 0.00, Pitch_Desired: 0.00, Command: 1.00
Yaw: 206.75, Roll: -109.57, Pitch: -18.25, Yaw_Desired: 196.00, Roll_Desired: 0.00, Pitch_Desired: 0.00, Command: 1.00
Yaw: 206.75, Roll: -109.57, Pitch: -18.25, Yaw_Desired: 196.00, Roll_Desired: 0.00, Pitch_Desired: 0.00, Command: 1.00
Yaw: 206.75, Roll: -109.57, Pitch: -18.25, Yaw_Desired: 196.00, Roll_Desired: 0.00, Pitch_Desired: 0.00, Command: 1.00
Yaw: 206.75, Roll: -109.57, Pitch: -18.25, Yaw_Desired: 196.00, Roll_Desired: 0.00, Pitch_Desired: 0.00, Command: 1.00
Yaw: 206.75, Roll: -109.57, Pitch: -18.25, Yaw_Desired: 196.00, Roll_Desired: 0.00, Pitch_Desired: 0.00, Command: 1.00
Yaw: 206.75, Roll: -109.57, Pitch: -18.25, Yaw_Desired: 196.00, Roll_Desired: 0.00, Pitch_Desired: 0.00, Command: 1.00
Yaw: 206.75, Roll: -109.57, Pitch: -18.25, Yaw_Desired: 196.00, Roll_Desired: 0.00, Pitch_Desired: 0.00, Command: 1.00
Yaw: 206.75, Roll: -109.57, Pitch: -18.25, Yaw_Desired: 196.00, Roll_Desired: 0.00, Pitch_Desired: 0.00, Command: 1.00
Yaw: 206.75, Roll: -109.57, Pitch: -18.25, Yaw_Desired: 196.00, Roll_Desired: 0.00, Pitch_Desired: 0.00, Command: 1.00
Yaw: 206.75, Roll: -109.57, Pitch: -18.25, Yaw_Desired: 196.00, Roll_Desired: 0.00, Pitch_Desired: 0.00, Command: 1.00
Yaw: 206.75, Roll: -109.57, Pitch: -18.25, Yaw_Desired: 196.00, Roll_Desired: 0.00, Pitch_Desired: 0.00, Command: 1.00
Yaw: 206.75, Roll: -109.57, Pitch: -18.25, Yaw_Desired: 196.00, Roll_Desired: 0.00, Pitch_Desired: 0.00, Command: 1.00
Yaw: 206.75, Roll: -109.57, Pitch: -18.25, Yaw_Desired: 196.00, Roll_Desired: 0.00, Pitch_Desired: 0.00, Command: 1.00
Yaw: 206.75, Roll: -109.57, Pitch: -18.25, Yaw_Desired: 196.00, Roll_Desired: 0.00, Pitch_Desired: 0.00, Command: 1.00
```

Nota. Elaboración propia. Captura tomada desde MatLab

De esta manera, se ha logrado establecer una interfaz entre el usuario y el sistema embebido, permitiendo definir con precisión los ángulos de inclinación deseados mediante comandos simples desde la HMI en LabVIEW.

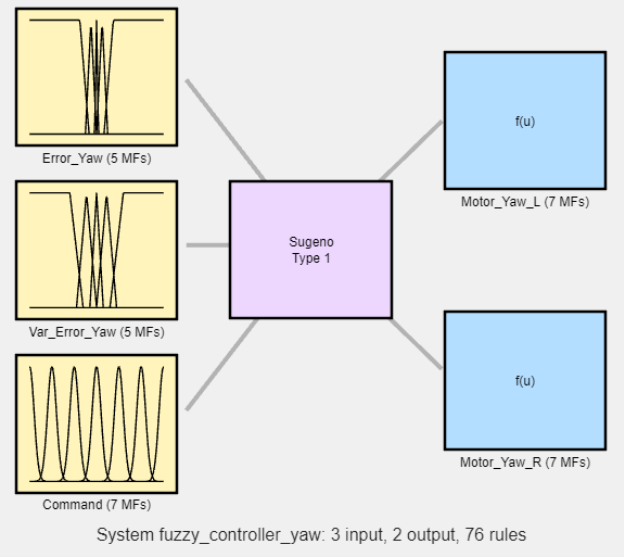
3.5. Diseño del Sistema de Control Difuso Autoajustable

El controlador difuso fue diseñado utilizando la versión más reciente de la herramienta Fuzzy Logic Designer de MatLab. Esta herramienta permite definir las funciones de membresía para cualquier número de entradas y salidas del sistema, así como establecer las reglas difusas necesarias. Finalmente, se genera un archivo con extensión .fis, el cual contiene la definición completa del controlador difuso.

Antes de diseñar el controlador, es importante mencionar que, con el fin de aprovechar el paralelismo que ofrecen las FPGAs, se plantea implementar un controlador independiente para cada ángulo de inclinación. En el caso del presente diseño, que emplea cuatro motores, se desarrollarán dos controladores difusos para Yaw y Pitch.

Figura 66

Entradas y Salidas del controlador difuso para el ángulo de inclinación YAW



Nota. Elaboración propia. Captura tomada desde MatLab

3.5.1. Funciones de Membresía del controlador

Se comenzará explicando el procedimiento seguido para el diseño del controlador difuso correspondiente al ángulo Yaw, debido a que el procedimiento para el controlador Pitch es similar. Para construir un controlador de este tipo, es fundamental definir

previamente la cantidad de entradas y salidas que tendrá el sistema. En este caso, se consideró la configuración mostrada en la figura 66, donde el controlador cuenta con tres entradas y dos salidas, las cuales representan las variables lingüísticas del sistema.

- **Entrada `Error_Yaw`**: Representa la diferencia entre el ángulo de yaw medido actualmente en el ROV y el ángulo de yaw deseado, definido por la instrucción de control enviada desde la estación terrena.
- **Entrada `Var_Error_Yaw`**: Corresponde a la variación del error de yaw, es decir, la diferencia entre el error actual y el error registrado en el intervalo anterior. Esta entrada permite estimar la velocidad con la que el error está cambiando.
- **Entrada `Command`**: Señal de control enviada desde la estación terrena, que indica la acción deseada. Está codificada mediante un valor numérico entre 0 y 6, representando diferentes tipos de movimiento del ROV.
- **Salida `Motor_Yaw_L`**: Controla la velocidad del motor brushless izquierdo del ROV, permitiendo así ajustar el ángulo de yaw.
- **Salida `Motor_Yaw_R`**: Controla la velocidad del motor brushless derecho del ROV, complementando el ajuste de la inclinación yaw.

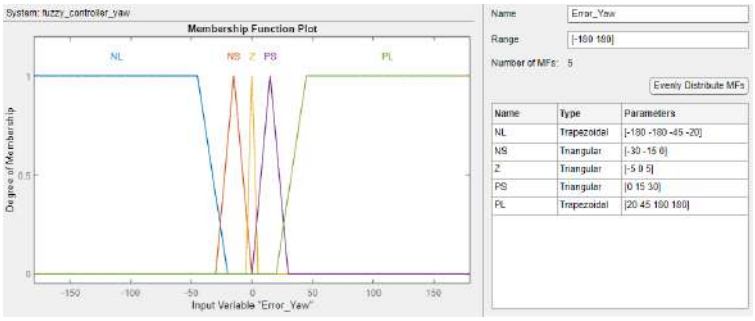
El siguiente paso consiste en definir las funciones de membresía para cada una de las entradas y salidas del controlador. Dado que un sistema de control difuso se basa en la experiencia humana, estas funciones fueron establecidas mediante un proceso de prueba y error, buscando una distribución adecuada de los valores.

Función de Membresía de `Error _Yaw`: El rango de esta variable va de $-180°$ a $180°$, lo que permite representar variaciones en el ángulo hacia la izquierda y la derecha. Se definieron cinco funciones de membresía lingüísticas mostradas en la figura 67: NL (Negativo Largo), NS (Negativo Corto), Z (Cero), PS (Positivo Corto) y PL (Positivo Largo). Las funciones extremas (NL y PL) utilizan el tipo trapmf para cubrir

adecuadamente los valores límite del error, mientras que las funciones centrales emplean trimf para facilitar transiciones suaves entre estados. Esta configuración permite que el sistema reaccione proporcionalmente a la magnitud y dirección del error de yaw, favoreciendo una corrección gradual y estable en el comportamiento del ROV.

Figura 67

Funciones de membresía de Error_Yaw

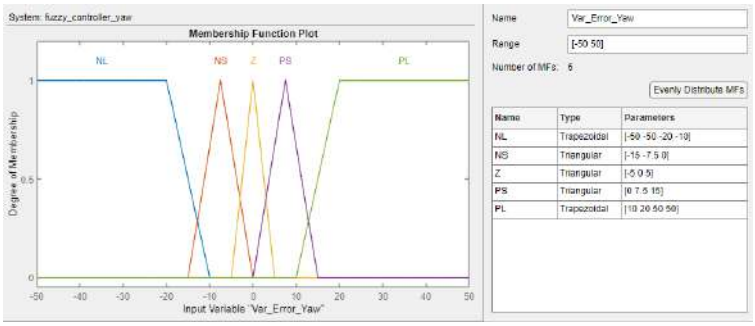


Nota. Elaboración propia. Captura tomada desde MatLab

Funciones de Membresía de `Var_Error_Yaw`: Esta variable, con un rango de $-50°$ a $50°$ por segundo, proporciona al controlador la velocidad del cambio de error para hacer predicciones. Al igual que `Error_Yaw`, se compone de cinco funciones de membresía lingüísticas mostradas en la figura 68: NL (Negativo Largo), NS (Negativo Corto), Z (Cero), PS (Positivo Corto) y PL (Positivo Largo), distribuidas simétricamente alrededor del cero.

Figura 68

Funciones de membresía de Var_Error_Yaw



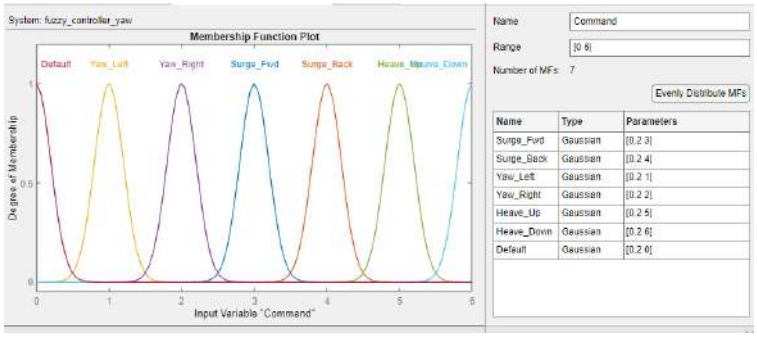
Nota. Elaboración propia. Captura tomada desde MatLab

Se utilizan funciones tipo trapmf en los extremos y trimf en las zonas intermedias, lo que permite una interpretación gradual del comportamiento del error, facilitando acciones correctivas anticipadas y mejorando la estabilidad del sistema.

Funciones de Membresía de `Command`: Define la acción de control deseada según el modo de operación actual del ROV, y permite al sistema diferenciar cuándo aplicar control de guiñada y cuándo no. Su rango es de 0 a 6 y está representada mediante siete funciones de membresía gaussianas (gaussmf) representadas en la funciona 69, cada una centrada en un valor entero correspondiente a un modo específico. Las funciones gaussmf permiten una transición suave entre modos, favoreciendo la adaptabilidad del sistema ante cambios dinámicos de operación. Esta variable actúa como un filtro lógico que habilita o inhibe el control de guiñada según el contexto operativo.

Figura 69

Funciones de membresía de Command



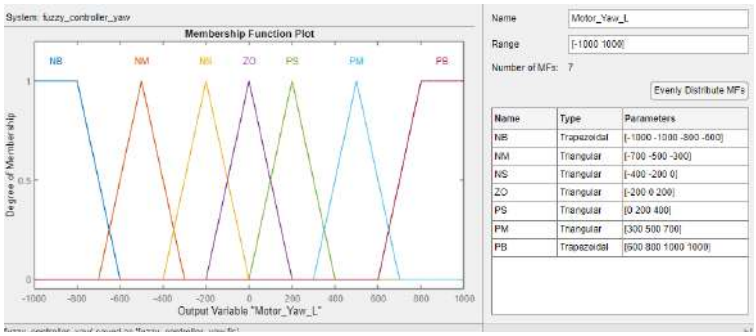
Nota. Elaboración propia. Captura tomada desde MatLab

Funciones de Membresía de `Motor_Yaw_L`: Tiene un rango de salida continuo que va de -1000 a 1000, representando desde la velocidad máxima del motor BLDC izquierdo en sentido antihorario hasta la máxima en sentido horario. Esta variable difusa está definida mediante siete funciones de membresía mostradas en la figura 70: NB (Negativo Grande), NM (Negativo Medio), NS (Negativo Pequeño), ZO (Cero), PS (Positivo Pequeño), PM (Positivo Medio) y PB (Positivo Grande). Las funciones ubicadas en los extremos son de tipo trapezoidal (trapmf), mientras que las intermedias son

triangulares (trimf).

Figura 70

Funciones de membresía de Motor_Yaw_L y Motor_Yaw_R



Nota. Elaboración propia. Captura tomada desde MatLab

Funciones de Membresía de `Motor_Yaw_R`: Su rango de valores también va de -1000 a 1000, lo que permite aplicar un torque de giro que trabaja de forma coordinada con el motor izquierdo. Usa las mismas siete funciones de membresía que el motor izquierdo, configuradas para que ambas salidas puedan generar giros suaves, equilibrados o más fuertes cuando sea necesario. La simetría entre ambas salidas es importante para lograr un control preciso del rumbo, especialmente en ambientes marinos cambiantes donde se necesita mantener la estabilidad frente a perturbaciones.

Se eligió un controlador difuso tipo Mamdani por su enfoque intuitivo y su capacidad para usar reglas lingüísticas que representan el conocimiento humano. A diferencia de controladores PID o adaptativos, no requiere un modelo exacto del sistema, lo que es útil en entornos dinámicos como el medio acuático. En este contexto, donde el ROV enfrenta perturbaciones constantes, la lógica difusa permite combinar variables como el error y su variación para generar respuestas más eficaces.

Los métodos de inferencia utilizados —prod para las operaciones AND e implicación, probor para OR, y centroide para la desfusificación— fueron seleccionados por su buen desempeño en control continuo. Estos métodos permiten obtener salidas suaves y estables, evitando saturaciones bruscas y facilitando su implementación eficiente en hardware como

FPGA, lo que resulta clave para lograr un control confiable y preciso en tiempo real.

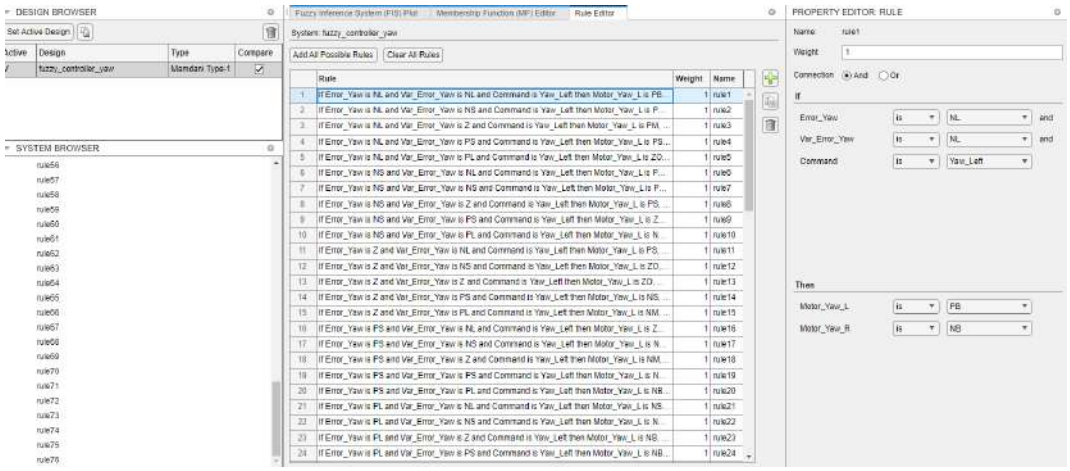
3.5.2. Elaboración de las reglas difusas

Siguiendo el enfoque del controlador difuso tipo Mamdani, se desarrollaron un total de 76 reglas lingüísticas. Estas reglas consideran la interacción entre el error de guiñada (Error_Yaw), su variación (Var_Error_Yaw) y el comando de operación (Command) para determinar las salidas de control hacia los motores izquierdo y derecho. Se priorizó que el control de guiñada solo actúe en los modos correspondientes (como Yaw_Left, Yaw_Right o Default), evitando interferencias durante maniobras como avance o inmersión.

Las reglas mostradas en la figura 71 fueron formuladas de forma simétrica y progresiva, generando respuestas suaves ante errores pequeños y correcciones más enérgicas cuando el error y su variación aumentan. Esta lógica de control permite que el sistema responda de manera proporcional y eficiente, mejorando la estabilidad del ROV en tiempo real sin provocar oscilaciones ni sobrecorrecciones.

Figura 71

Reglas del controlador difuso de Mamdani



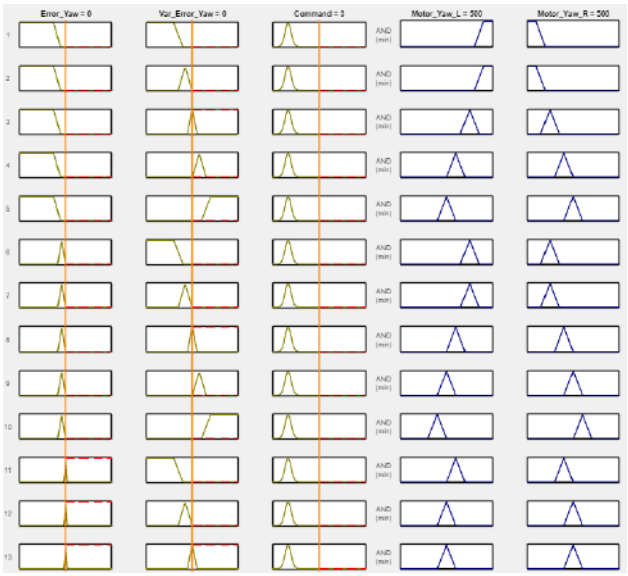
Nota. Elaboración propia. Captura tomada desde MatLab

Con el controlador diseñado visualizamos la intersección de las reglas en la figura 72 y la superficie de control generado en Fuzzy Logic Designer en la figura 73. Además mencionar que para el controlador del ángulo el procedimiento de diseño fue similar

modificando unicamente las reglas para que los motores actúen para compensar pitch.

Figura 72

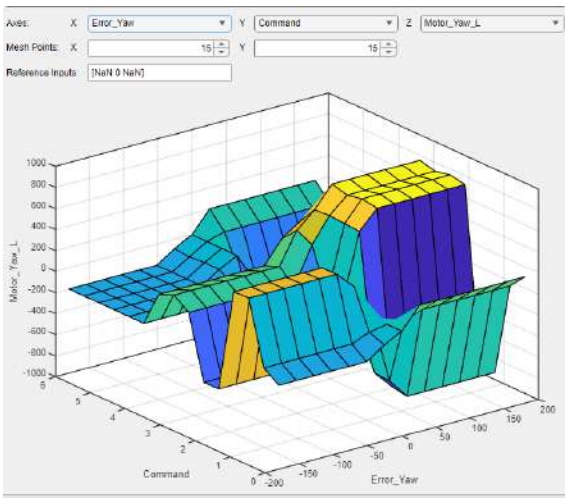
Interferencia de las reglas del controlador difuso



Nota. Elaboración propia. Captura tomada desde MatLab

Figura 73

Superficie de control del Error_ Yaw vs Command



Nota. Elaboración propia. Captura tomada desde MatLab

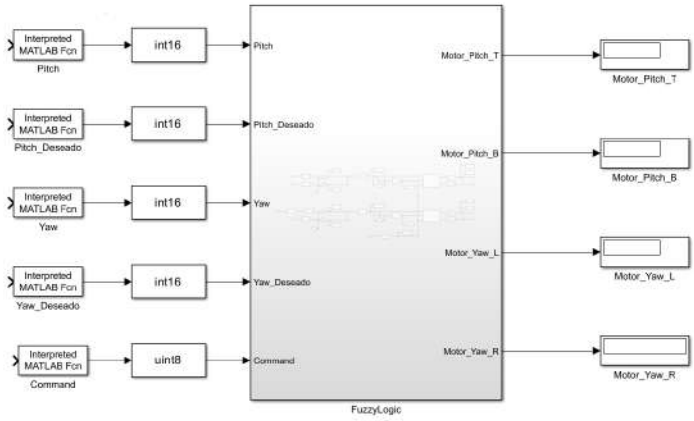
Finalmente, una vez diseñados ambos controladores, se exportaron sus respectivos archivos .fis para su posterior implementación en la FPGA.

3.6. Implementación de controlador difuso en FPGA

Con los sistemas de control difuso ya diseñados, el siguiente paso es implementar la generación de las señales de entrada y salida que interactuarán con los controladores de yaw y pitch en Simulink. Para ello, se diseñó un subdiagrama llamado FuzzyLogic, mostrado en la figura 74, encargado de recibir directamente desde la FPGA los ángulos correspondientes y generar como salida las velocidades de los motores. Actualmente, se utiliza la herramienta HDL Coder de Simulink para generar automáticamente el código VHDL a partir del modelo, lo que facilita el proceso de implementación. Sin embargo, en futuras etapas del proyecto se plantea diseñar e implementar el sistema completo directamente en FPGA, con el objetivo de optimizar el uso de recursos y mejorar el rendimiento del sistema.

Figura 74

Diseño Implementando en Simulink para HDL Coder



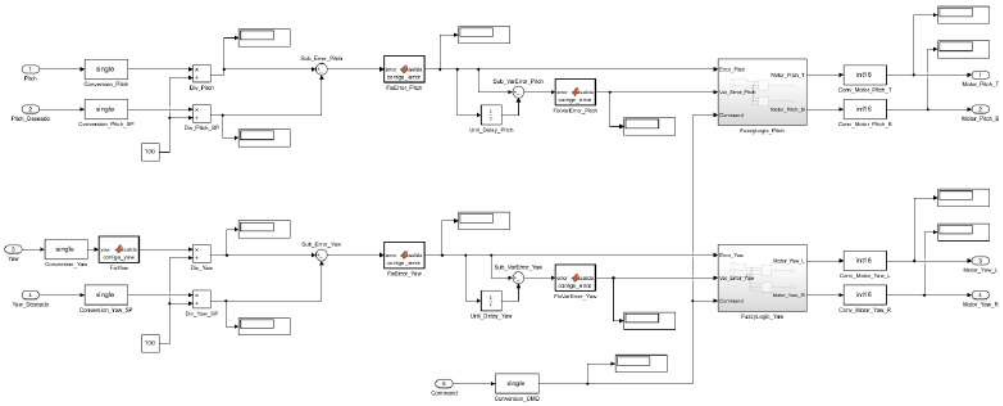
Nota. Elaboración propia. Captura tomada desde Simulink

Internamente este bloque mostrado en la figura 75 contiene funciones de conversión de tipo de dato, algunas operaciones de escalamiento y limitación junto con operaciones para la generación del error y la variación de error. La herramienta FuzzyLogic de MatLab no es compatible con HDL Coder, por ello no se pudo usar directamente este bloque, como solución se utilizó un Lookup Table, con valores previamente calculados mediante otro script de MatLab. Una Lookup Table es una estructura utilizada para reemplazar cálculos

complejos por una simple búsqueda de valores previamente almacenados. En lugar de realizar operaciones matemáticas en tiempo real, la LUT contiene una lista de valores de salida ya calculados para distintas entradas posibles.

Figura 75

Diseño Implementando en Simulink por dentro para HDL Coder

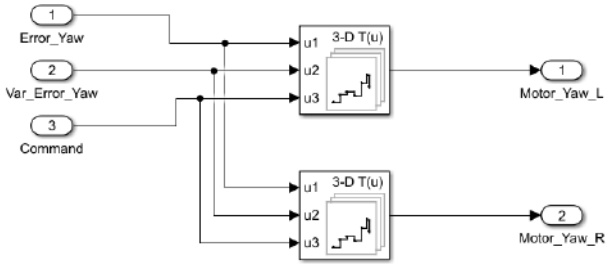


Nota. Elaboración propia. Captura tomada desde Simulink

Se implementó esta solución de forma temporal. Aunque es funcional, consume muchos recursos lógicos en la FPGA, por lo que se usó una precisión reducida al generar la lookup table. A futuro, se planea diseñar el controlador difuso directamente en VHDL usando técnicas más eficientes. La figura 76 muestra el subdiagrama con las tablas precalculadas.

Figura 76

Diagrama del controlador usando LUT



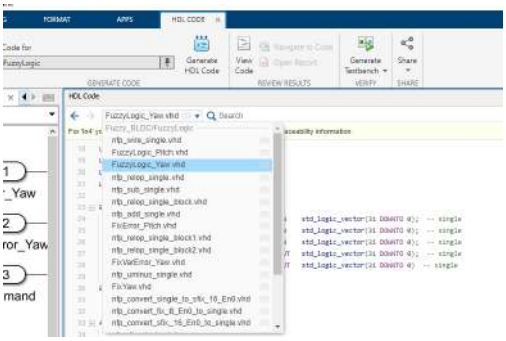
Nota. Elaboración propia. Captura tomada desde Simulink

Una vez finalizado el subdiagrama en Simulink, se generó el código VHDL

utilizando la herramienta HDL Coder. Los resultados se muestran en la figura 77, donde se incluyen las operaciones necesarias para implementar el controlador difuso.

Figura 77

Archivos HDL generados mediante HDL Coder



Nota. Elaboración propia. Captura tomada desde Simulink

Estos códigos fueron implementados junto con los demás módulos en el entorno de Vivado, donde finalmente se obtuvo el diagrama RTL mostrado en la figura 78:

Figura 78

Diseño RTL con el controlador difuso implementado

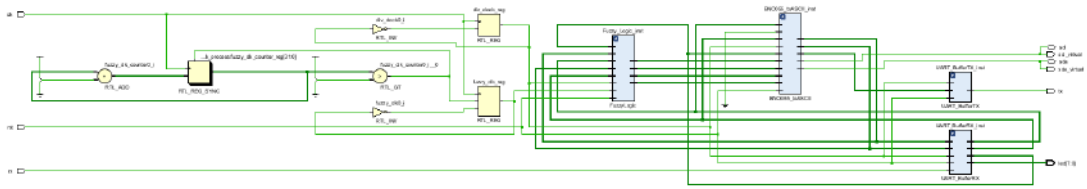
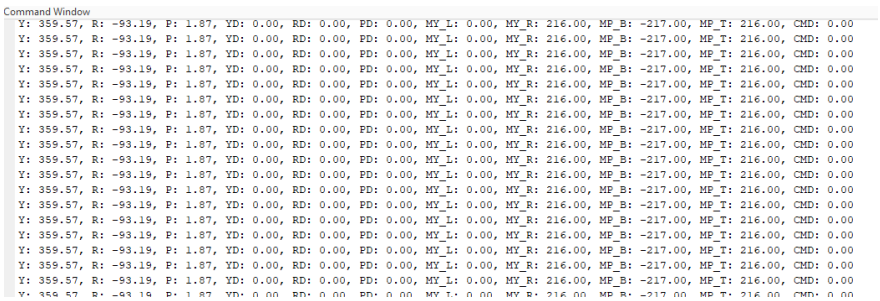


Figura 79

Resultados obtenidos mediante script de MatLab



Referencias

AliExpress. (2025). Controlador ESC sin escobillas 30A para motores BLDC con conector XT60 y BEC 5V/2A [Consultado el 24 de junio de 2025].

Azis, F. A., Aras, M. S. M., Rashid, M. Z. A., Othman, M. N., & Abdullah, S. S. (2012). Problem identification for underwater remotely operated vehicle (ROV): A case study. *Procedia Engineering*, *41*, 554-560.

Dong, J., & Duan, X. (2023). A robust control via a fuzzy system with PID for the ROV. *Sensors*, *23*(2), 821.

Embebidos, I. E.

bibinitperiod. (2021). Manual técnico: Tarjeta Avanxe 7 - Revisión B [Consultado el 24 de junio de 2025].

Embebidos, I. E.

bibinitperiod. (2025). Quiénes somos — INTESC Electrónica & Embebidos [Consultado el 24 de junio de 2025].

GmbH, B. S. (2024). BNO055 Intelligent 9-axis absolute-orientation sensor [Consultado el 24 de junio de 2025].

https://www.bosch-sensortec.com/products/smart-sensor-systems/bno055/

Huang, H. C., Tao, C. W., Chuang, C. C., & Xu, J. J. (2019). FPGA-based mechatronic design and real-time fuzzy control with computational intelligence optimization for omni-Mecanum-wheeled autonomous vehicles. *Electronics*, *8*(11), 1328.

Huang, X., Chen, Y., & Li, Z. (2019). Inspection-Class Remotely Operated Vehicles—A Review [Consultado el 29 de junio de 2025]. *Journal of Marine Science and Engineering*, *7*(1), 13. https://doi.org/10.3390/jmse7010013

IPC. (2019). IPC-2221: Generic Standard on Printed Board Design [Consultado el 24 de junio de 2025]. https://www.ipc.org/TOC/IPC-2221B.pdf

National Instruments. (2012). *Design Patterns: Queued Message Handler* [https://www.ni.com/en-us/innovations/queued-message-handler.html]. National Instruments.

Ogata, K. (2010). *Modern control engineering* (5.ª ed.). Prentice Hall.

Rectifier), I. T. (I. (2019). IR2101(S)/IR2102(S) High- and Low-Side Driver IC – Datasheet [Consultado el 24 de junio de 2025].

Rectifier), I. T. (I. (2021). IRFZ44N Power MOSFET – Datasheet [Consultado el 24 de junio de 2025].

Rectifier, I. (2009). IR2103 - High and Low Side Driver [Accessed: 2025-07-01].

Rehman, S. U., Mustafa, H., & Saleem, H. B. (2021). Prototyping and stabilizing of under-actuated remotely operated vehicle (ROV) using fuzzy PID control algorithm. *Journal of Engineering*.

Sanchez Rosado, R. M. (2016). *Análisis comparativo entre controladores basados en lógica difusa y control PID clásico, aplicados a sistemas de control de velocidad de motores DC sin escobillas* [Tesis de maestría]. Escuela Superior Politécnica del Litoral.

Sensortec, B. (2020). BNO055 Intelligent 9-axis absolute orientation sensor datasheet [Rev. 1.4].

Skaar, S. E., Krovel, O., & Nilssen, R. (2006). Distribution, coil-span and winding factors for PM machines with concentrated windings. *ICEM 2006*, 2-5.

Wang, J., Li, M., Jiang, W., Huang, Y., & Lin, R. (2022). A design of FPGA-based neural network PID controller for motion control system. *Sensors*, *22*(3), 889.

Zadeh, L. A. (1965). Fuzzy sets. *Information and Control*, *8*(3), 338-353.

Anexos

Anexo A: Esquemas del sistema

Anexo B: Código fuente en VHDL

Anexo C: Tablas de resultados