

**Instituto Tecnológico de Costa Rica**

**Área Académica de Ingeniería Mecatrónica**



**Diseño y desarrollo de una estructura de software-hardware para implementar un nuevo autopiloto sobre un AUV (Autonomous Unmanned Vehicle)**

**Informe de Proyecto de Graduación para optar por el título de Ingeniero en Mecatrónica con el grado académico de Licenciatura**

**Anibal Sancho Theoduloz**

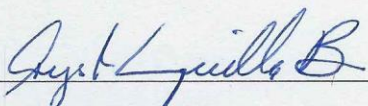
**201112284**


**Cartago, marzo de 2019.**

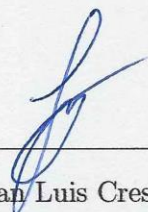
Instituto Tecnológico de Costa Rica  
Área Académica de Ingeniería en Mecatrónica  
Proyecto de Graduación  
Tribunal Evaluador  
Acta de Evaluación

Proyecto de Graduación defendido ante el presente Tribunal Evaluador como requisito para optar por el título de Ingeniero en Mecatrónica con el grado académico de Licenciatura del Instituto Tecnológico de Costa Rica.

Miembros del Tribunal

  
M. Sc. Arys Carrasquilla Batista  
Profesora Lectora

  
M. Sc. Carlos Adrián Salazar García  
Profesor Lector

  
Dr. Ing. Juan Luis Crespo Mariño  
Profesor Asesor

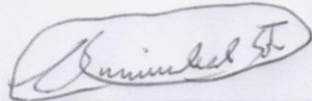
Los Miembros de este Tribunal dan fe de que el presente trabajo de graduación ha sido aprobado y cumple con las normas establecidas por el Área Académica de Ingeniería Mecatrónica.

Cartago, 1 de Marzo de 2019.

Declaro que el presente Proyecto de Graduación ha sido realizado enteramente por mi persona, utilizando y aplicando literatura referente al tema e introduciendo conocimientos propios.

En los casos en que he utilizado bibliografía, he procedido a indicar las fuentes mediante las respectivas citas bibliográficas.

En consecuencia, asumo la responsabilidad total por el trabajo de graduación realizado y por el contenido del correspondiente informe final.



---

Anibal Sancho Theoduloz

Cédula: 1-1497-0760

Cartago, 1 de Marzo de 2019.

## **Resumen**

En este proyecto se presenta parte del trabajo de una línea de investigación con miras en robótica autónoma, específicamente en el desarrollo de submarinos para que sean del tipo AUV. Concretamente se trabajó en la implementación del autopiloto Pixhawk que es del tipo open-hardware y todas las adaptaciones que hubo que hacer para poder llevar a cabo dicha tarea. El Pixhawk, ofrece varias ventajas, como el acceso al código fuente, el respaldo de una comunidad de desarrolladores y el fácil acceso a documentación general (tutoriales, foros, blogs) aparte de la oficial. Como parte de las adaptaciones involucraron la adaptación para el control de los motores y las luces, la incorporación de los sensores como la IMU y el DVL, entre otros.

**Palabras clave:** AUV, ROV, Ardusub, Pixhawk, submarino, ROS.

## **Abstract**

In this project we present part of the work of a research line aimed at autonomous robotics, specifically in the development of AUV (Autonomous Underwater Vehicles). Specifically, we worked on the implementation of the Pixhawk, which is an open-hardware autopilot and all the adaptations needed to develop this task. The Pixhawk, offers several advantages, such as access to the source code, the support of a community of developers and easy access to general documentation (tutorials, forums, blogs) other than the official ones. As part of the adaptations involved the adaptation for the control of the motors and the lights, the incorporation of the sensors like the IMU and the DVL, among others.

**Keywords:** AUV, ROV, Ardusub, Pixhawk, submarine, ROS.

## **Agradecimientos**

Agradezco primeramente a Dios por haberme ayudado a llegar hasta este punto, por ser mi fuente de inspiración en momentos en los que quise dejar todo botado.

Agradezco a mis padres por todo el apoyo que recibí para poder llegar a concluir satisfactoriamente mis estudios.

Agradezco a todos los profesores que de alguna forma u otra ayudaron en este proceso de formación, a los profesores del tribunal evaluador quienes fueron de gran ayuda para lograr el éxito de este proyecto, particularmente hacer mención al profesor Juan Luis Crespo quien aparte de ser mi profesor asesor fue el punto de contacto para lograr realizar este trabajo con el Grupo Integrado de Ingeniería (GII).

Agradezco al GII por abrirme las puertas para poder realizar este proyecto en tan prestigioso grupo de investigación; a todos los que fueron parte de este proceso de aprendizaje y desarrollo, particularmente hacer mención a Moisés Bautista, quien fue el gestor de mi trabajo, por todas las enseñanzas, consejos y ayudas durante el desarrollo de este designio.

Por último, agradecer a todos aquellos compañeros y amigos que durante la carrera fui conociendo, con quienes compartí proyectos, tareas, grupos de estudio, “palmadas”, etc.

# ÍNDICE GENERAL

Índice de figuras.....	III
Índice de tablas .....	V
<b>Capítulo 1. Introducción .....</b>	<b>1</b>
<b>1.1 Entorno del proyecto .....</b>	<b>1</b>
<b>1.2 Definición del problema .....</b>	<b>3</b>
1.2.1 Generalidades .....	3
1.2.2 Síntesis del problema .....	3
<b>1.3 Enfoque de la solución.....</b>	<b>4</b>
<b>Capítulo 2. Objetivos .....</b>	<b>5</b>
<b>2.1 Objetivo general .....</b>	<b>5</b>
<b>2.2 Objetivos específicos .....</b>	<b>5</b>
<b>Capítulo 3. Marco teórico.....</b>	<b>6</b>
<b>3.1 Historia y generalidades de los submarinos.....</b>	<b>6</b>
<b>3.2 Hardware actual .....</b>	<b>8</b>
3.2.1 Descripción de sensores .....	10
3.2.2 Descripción de actuadores .....	12
3.2.3 Otros dispositivos.....	13
<b>3.3 Protocolos de comunicación .....</b>	<b>13</b>
3.3.1 Protocolo de comunicación MAVLink.....	14
3.3.2 Protocolo de comunicación S-Bus .....	15
<b>3.4 ROS .....</b>	<b>16</b>
<b>Capitulo 4. Marco metodológico .....</b>	<b>18</b>
<b>4.1 Caracterización de la arquitectura actual de la planta .....</b>	<b>18</b>
<b>4.2 Diseño de la nueva arquitectura .....</b>	<b>18</b>
<b>4.3 Implementación de la nueva arquitectura .....</b>	<b>19</b>
<b>4.4 Validación del correcto funcionamiento del submarino .....</b>	<b>19</b>
<b>Capitulo 5. Descripción de la solución .....</b>	<b>20</b>

5.1	Requerimientos .....	20
5.2	Propuesta de diseño .....	21
5.3	Caracterización de los motores .....	22
5.4	Luces .....	33
5.5	Circuito inversor para el S-Bus .....	36
5.6	PCB MAX232ACSE+ .....	38
5.7	Placa general contenedor electrónica .....	39
5.8	Cambios realizados al firmware del Pixhawk .....	42
5.8.1	IMU (VectorNav) .....	44
5.8.2	DVL (LinkQuest) .....	45
5.9	Placa general del contenedor de potencia .....	46
<b>Capitulo 6. Validación.....</b>		<b>50</b>
<b>Conclusiones .....</b>		<b>54</b>
<b>Recomendaciones y trabajo futuro .....</b>		<b>56</b>
<b>Estudio económico.....</b>		<b>57</b>
<b>Referencias .....</b>		<b>58</b>
<b>Apéndices .....</b>		<b>60</b>
<b>Apéndice 1. Procedimiento para compilar de cero (build clean) el firmware del Pixhawk.</b>		<b>60</b>
<b>Apéndice 2. Procedimiento para agregar nuevos mensajes de MAVLink .....</b>		<b>61</b>
<b>Apéndice 3. Procedimiento para cambiar la red en la que se conectan la RaspberryPi-GCS</b>		<b>63</b>
.....		

## ÍNDICE DE FIGURAS

Figura 1: Submarino del GII. Fuente: GII [1] .....	2
Figura 2. Clasificación de los submarinos según su sistema de propulsión. a) Seaeye Panther-XT, b) REMO, c) Slocum Glider, d) Seaglider, e) Tuna Robot, f) AQUA. Fuente: Héctor A. Moreno et al. [4].....	8
Figura 3. Diagrama de comunicaciones del submarino. Fuente: GII .....	9
Figura 4. Comparación de sonar de obstáculos con tecnología CHIRP (lado derecho) vs. sistema tradicional (lado izquierdo). Fuente: Tritech International Ltd [7].....	11
Figura 5. Diagrama básico de conexiones del submarino. Fuente: Elaboración propia utilizando draw.io. ....	21
Figura 6. Grados de libertad en un submarino. Fuente: <a href="https://simple.wikipedia.org/wiki/Pitch,_yaw,_and_roll">https://simple.wikipedia.org/wiki/Pitch,_yaw,_and_roll</a> .....	23
Figura 7. Diagrama de cuerpo libre de submarino de referencia. Fuente: Elaboración propia utilizando AutoCAD.....	24
Figura 8. Diagrama de cuerpo libre con la configuración vectorizada del vehículo en cuestión. Cabe destacar que el diagrama de cuerpo libre está hecho siguiendo el sistema europeo de vistas, por tanto, en la vista lateral las fuerzas F5 y F6 son respectivamente hacia arriba (-z) y hacia abajo (+z). Fuente: Elaboración propia utilizando AutoCAD. ....	27
Figura 9: Diagrama de cuerpo libre con la configuración normal de los motores del vehículo en cuestión. Cabe destacar que el diagrama de cuerpo libre está hecho siguiendo el sistema europeo de vistas, por tanto, en la vista lateral las fuerzas F5 y F6 son respectivamente hacia arriba (-z) y hacia abajo (+z). Fuente: elaboración propia utilizando AutoCAD. ....	29
Figura 10. Pruebas con el submarino en el agua. Fuente: Elaboración propia.....	33
Figura 11. Esquemático del circuito de adaptación para la luz. Fuente: Elaboración propia utilizando KiCad. ....	34
Figura 12. PCB del circuito de adaptación de la luz. Fuente: Elaboración propia utilizando KiCad. ....	35
Figura 13. Señales del circuito de control de las luces a diferentes intensidades lumínicas. Siendo la de color amarillo la señal de PWM proveniente del microcontrolador, la de color magenta luego de la etapa del conversor CD-CD y la de color cian la de la salida filtrada. Fuente: Elaboración propia. ....	36



Figura 14. Esquemático del circuito inversor del S-Bus. Fuente: Elaboración propia utilizando KiCad.....	36
Figura 15. PCB del circuito inversor del S-Bus. Fuente: Elaboración propia utilizando KiCad. .	37
Figura 16. Señales del S-Bus antes y después del inversor. Siendo la señal en color amarillo la original proveniente del S-bus y la magenta la señal invertida del S-Bus. Fuente: Elaboración propia. ....	38
Figura 17. Esquemático del conversor MAX232ACSE+ Fuente: Elaboración propia utilizando KiCad.....	38
Figura 18. PCB MAX232ACSE+ Fuente: Elaboración propia utilizando KiCad.....	39
Figura 19. Esquemático de la placa general en contenedor de electrónica. Fuente: Elaboración propia utilizando KiCad.....	41
Figura 20. PCB de la placa general del contenedor de electrónica. Fuente: Elaboración propia. .	42
Figura 21. Comprobación del envío de datos de la IMU. Fuente: Elaboración propia. ....	45
Figura 22. Comprobación del envío de datos del DVL. Fuente: Elaboración propia. ....	46
Figura 23. Esquemático de la placa general del contenedor de potencia. Fuente: Elaboración propia utilizando KiCad .....	48
Figura 24. PCB de la placa general del contenedor de potencia. Fuente: Elaboración propia.....	49
Figura 25. Compartimiento de electrónica del submarino en cuestión. Fuente: Elaboración propia. ....	51
Figura 26. Plataforma del QGroundControl (GCS) conectada al Pixhawk. Fuente: Elaboración propia. ....	52

## ÍNDICE DE TABLAS

Tabla 1. Cuadro resumen de la influencia de los motores del submarino de referencia. Fuente: Elaboración propia. ....	26
Tabla 2. Cuadro resumen de la influencia de los motores del submarino en cuestión con la configuración vectorizada. Fuente: Elaboración propia. ....	29
Tabla 3. Cuadro resumen de la influencia de los motores del submarino en cuestión con la configuración normal. Fuente: Elaboración propia. ....	31
Tabla 4. Rotación de los motores según su movimiento en los diferentes grados de libertad, esto para la configuración vectorizada de los motores. Fuente: Elaboración propia. ....	31
Tabla 5. Control de cambios en el firmware del Pixhawk. Fuente: Elaboración propia. ....	43
Tabla 6. Costes Aproximados. Fuente: Elaboración propia. ....	57

# Capítulo 1. Introducción

## 1.1 Entorno del proyecto

El proyecto que se presenta se realizó en el Grupo Integrado de Ingeniería (GII), el cual es un grupo de investigación multidisciplinario que cuenta con la colaboración externa de empresas y pertenece a la Universidad de la Coruña; se encuentra localizado en el campus de Ferrol, en la provincia de La Coruña, en Galicia, España.

El GII cuenta con varias líneas de investigación entre las cuales se destacan las siguientes: “Ingeniería naval y oceánica” (en la cual se desarrollan proyectos de investigación y desarrollo en tecnologías marítimas y offshore), “Dinámica de fluidos” (en la cual se aplican técnicas avanzadas de dinámica de fluidos tanto experimentalmente como utilizando recursos computacionales para la resolución de problemas en ámbitos de diversa índole como industriales, hidrodinámica naval o sistemas aeroespaciales), “Inteligencia computacional” (es una rama que involucra todo lo relacionado con la inteligencia artificial, la cual se inspira en procesos naturales para la solución de problemas que tradicionalmente son muy complejos de solucionar por los métodos computacionales clásicos), “Ingeniería de organización industrial y transporte” (involucran proyectos que implican el análisis, diseño, planificación control y optimización de procesos industriales considerando aspectos económicos, técnicos y sociales), “Robótica y Cognición” (desde sus orígenes esta ha sido una de las principales líneas de investigación del GII, en el cual se han producido prototipos de robots autónomos para la industria y actualmente está en capacidad para producir estructuras robóticas con capacidades singulares) y por último “Tecnologías avanzadas de sensorización y medida” (esta última implica la suma de conocimientos de las otras líneas de investigación con el fin de aplicar técnicas de medidas con sensores y procesamiento de señales e imágenes).

El proyecto SAILOR [1] es uno de los pilares sobre los que se asienta actualmente el trabajo en vehículos submarinos por parte del GII y del que proviene el vehículo que se empleará en el desarrollo de este trabajo.

Este proyecto se enmarca dentro de la línea de investigación en robótica y cognición, explícitamente en una línea derivada denominada robótica autónoma del GII. El objetivo general

de esta línea es dotar de autonomía a robots que se desplazan en cualquier medio, permitiendo así que realicen tareas sin interacción externa. De forma más concreta, en la actualidad se está trabajando activamente en dotar de autonomía a vehículos submarinos, para que sean capaces de realizar labores como el mantenimiento de estructuras eólicas marinas.

Lo anterior se justifica debido a que en los últimos años ha habido un aumento de estructuras marinas, asociado en gran medida al auge de la eólica marina, está impulsando un fuerte desarrollo en los vehículos submarinos para su uso en la construcción y el mantenimiento de dichas estructuras. Actualmente la gran mayoría de vehículos que se emplean son ROV (Remotely Operated Vehicle). Este tipo de vehículos requiere la presencia permanente de un operador experimentado y generalmente se conectan mediante un cable umbilical a una embarcación de apoyo, donde está alojada la reserva de energía y que se encarga del despliegue y recogida del ROV. La tendencia dominante es aumentar las capacidades autónomas de estos vehículos, de forma que se puedan automatizar tareas para lograr obtener un AUV (Autonomous Unmanned Vehicle), es decir, un vehículo que sea capaz de realizar tareas por sí mismo, sin la interacción del operador.



Figura 1: Submarino del GII. Fuente: GII [1]

## **1.2 Definición del problema**

### **1.2.1 Generalidades**

Como se mencionó anteriormente, la plataforma submarina fue parte de otro proyecto de investigación para el cual se le encargó a otra empresa externa la fabricación del mismo, sin embargo, tiempo antes de la entrega final del submarino trabajando con todo y su respectiva documentación dicha empresa cesó sus operaciones, y aunque ya se tenía el hardware seleccionado el software aún se encontraba en versión de pruebas. Desde ese momento se estuvo trabajando con la finalidad de ponerlo a funcionar con la arquitectura original, luego de ciertos infortunios y fallos de diversa índole se logró hacer pruebas con las cuales se pudo ver el potencial del vehículo subacuático para su uso en el mantenimiento de estructuras marinas.

A pesar de lo anterior también se detectaron varias deficiencias como la falta de ciertos sensores para verificar la integridad del vehículo (por ejemplo, carece de sensores de inundación y debido a esto en una ocasión se tuvieron que cambiar las baterías por infiltración de agua en el respectivo compartimiento). Por otro lado, la arquitectura actual presenta varios inconvenientes para poder lograr lo que se pretende en la línea de investigación, entre ellos se destaca que el sistema de control es de código cerrado, con lo cual dificulta la solución de problemas, su actualización y la mejora de forma sencilla en el futuro; también cabe destacar que dicha estructura actualmente está diseñada para funcionar en modo ROV, ya que hay muchos sensores que actualmente funcionan con el software proporcionado por el fabricante, con lo cual se puede acceder a los datos de forma gráfica pero no se pueden obtener de forma que se puedan procesar para la toma de decisiones autónomas.

### **1.2.2 Síntesis del problema**

Se pretenden mejorar las carencias de la arquitectura de control de la plataforma submarina, sustituyendo el actual autopiloto y toda la electrónica relacionada por una alternativa de código abierto que mejore su potencial de actualización y mejora en el futuro, permitiendo así obtener el máximo partido del hardware de sensorización y actuación con el que cuenta actualmente el submarino.

### **1.3 Enfoque de la solución**

En el GII poseen un submarino completamente funcional, pero se decidió adaptar el hardware a un nuevo controlador/autopiloto (para este caso se utilizará el Pixhawk 2 Autopilot) que le permite una mayor versatilidad ya que este, al ser de código abierto, admite una mayor transparencia en el controlador y deja abierta la posibilidad de mejoras futuras. A continuación, se mencionan las etapas que se siguieron para la implementación de la solución:

- Caracterización de la arquitectura actual de la planta.
- Diseño de la nueva arquitectura.
- Implementación de la nueva arquitectura.
- Validación del correcto funcionamiento del submarino.

## **Capítulo 2. Objetivos**

### **2.1 Objetivo general**

Diseñar una plataforma submarina que sirva de insumo de investigación en tareas autónomas de mantenimiento en estructuras marinas con una arquitectura de código abierto, que permita su ampliación y mejora futura de forma sencilla.

### **2.2 Objetivos específicos**

- 2.2.1. Caracterizar la estructura de hardware/software actual presente en el submarino.
- 2.2.2. Diseñar la nueva estructura de hardware/software para el submarino.
- 2.2.3. Implementar la estructura de hardware/software diseñada a la estructura existente.
- 2.2.4. Validar el funcionamiento del submarino con la nueva estructura implementada por medio de pruebas específicas.

## Capítulo 3. Marco teórico

### 3.1 Historia y generalidades de los submarinos

La Tierra, que es también conocido como el planeta azul, cuenta con dos terceras parte de su superficie cubiertas por agua; cuya profundidad varía mucho dependiendo de su ubicación geográfica, desde unos pocos metros hasta lo más profundo conocido hasta ahora, las fosas Marianas, con una profundidad de 11 034 metros [2]. El hombre siempre ha querido explorar los océanos por diversos motivos, entre los cuales destacan los siguientes campos de aplicación: el estudio de las diferentes variedades de microbios en el océano; estudio de los ecosistemas; exploración y conservación de los océanos; todo referente al desarrollo y puesta en marcha de sistemas de comunicación, transporte de energía y la industria naval; arqueología oceánica; energías renovables marinas; construcciones marinas; entre otras [3].

En el marco de este proyecto se desea dotar de herramientas para el campo de aplicación referente a las construcciones marinas, de manera más específica el mantenimiento de las mismas; esto por medio de un vehículo submarino no tripulado especializado que, entre otras funciones, permita comprobar el estado de dichas estructuras, así como realizar cierto tipo de reparaciones simples o en su defecto detectar e informar de fallas más graves, de forma que se puedan tomar las medidas pertinentes. Debido a lo anterior es que el GII ha dedicado parte de su línea de investigación en robótica autónoma a poder desarrollar dicho vehículo.

A manera introductoria se hablará del surgimiento de estos vehículos submarinos, estos se empiezan a desarrollar en los años 50 del siglo XX, cuando en Francia Dimitri Rebikoff desarrolló el primer ROV submarino denominado POODLE, desde ese entonces han surgido muchos de diferentes tipos, desde los operados remotamente no tripulados (ROV) hasta los utilizados por los grupos navales militares, ya más recientemente se está buscando llegar a la completa autonomía de los mismos, de forma que no se dependa de un operario [4].

Moreno et al. [4] presentan diferentes clasificaciones de submarinos, dependiendo de sus características los clasifican por nivel de autonomía, tipo de misión o tipo de propulsión. Respecto a la clasificación por nivel de autonomía existen tres categorías los ROV que viene del inglés “Remote Operated Vehicle” lo cual implica que el vehículo depende completamente de un operario que lo esté controlando, generalmente posee un “cordón umbilical” a través del cual se da la comunicación entre el vehículo y la estación de control (y en muchos casos es también por donde



se da la alimentación del sumergible); antagónicamente el AUV que viene del inglés “Autonomous Unmanned Vehicle” es un vehículo que se controla por sí mismo, por ende, cuenta con todos los sensores y la lógica necesaria para realizar misiones predefinidas, generalmente, a diferencia de los anteriores, estos vehículos no cuentan con un cordón, por lo que es imprescindible contar con una fuente de alimentación propia (y en algunos casos se cuenta con dispositivos de comunicación inalámbrica como un modem acústico); por último IAUV que viene del inglés “Intervention Autonomous Underwater Vehicle” que son los que requieren de un operario que supervise y realice las tareas de razonamiento necesarias para cumplir la misión, en este tipo de vehículos lo que se busca es llegar a la mayor autonomía posible o que el operario únicamente defina tareas por medio de comandos de alto nivel. Otro criterio de clasificación es por el tipo de misión que van a realizar, ya que pueden ser de inspección o de manipulación, y su diferencia radica en que para el caso de manipulación está diseñado para que pueda hacer reparaciones in situ por ende lleva más sensores y un brazo robótico que contenga las herramientas apropiadas para la aplicación. Por último, en la clasificación por su sistema de propulsión se encuentran los siguientes: impulsores de hélice (el cual consiste en acoplar una hélice a un motor eléctrico y esta al girar produce un fuerza de empuje al desplazar el líquido de un extremo al otro esto debido a la diferencia de presión, ver Figura 2 en “a” se presenta uno con múltiples impulsores y en “b” se presenta configuración con un impulsor), planeador acuático (estos no poseen ninguna hélice puesto que adentran en el agua y se maniobran únicamente cambiando su flotabilidad, ver Figura 2 “c” y “d”), bioinspirado (para este tipo se inspiran en la naturaleza para imitar la fisiología de los animales acuáticos, ver Figura 2 “e” y “f”), por inyección (consiste obtener agua del medio en el que se encuentren, almacenarla y por medio de bombas expulsarlas a presión a través de una boquilla devuelta al medio acuático), impulsor magnetohidrodinámico (supone hacer pasar una corriente por un fluido conductor, como el agua salada, que luego de interactuar con un campo magnético produce un movimiento del fluido) y por tracción en el fondo marino (este es el equivalente a los robots terrestres ya que supone de un artilugio que se desplaza en el fondo marino ya sea por medio de patas, ruedas, orugas, entre otros).

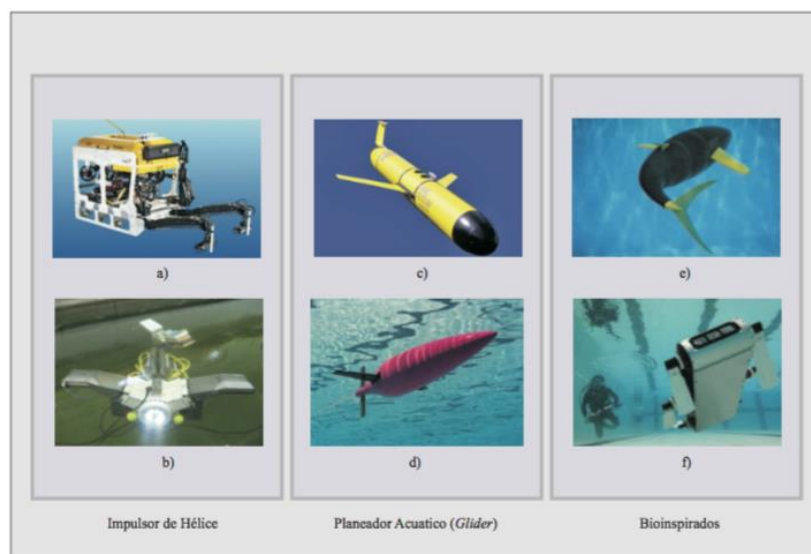


Figura 2. Clasificación de los submarinos según su sistema de propulsión. a) Seaeye Panther-XT, b) REMO, c) Slocum Glider, d) Seaglider, e) Tuna Robot, f) AQUA. Fuente: Héctor A. Moreno et al. [4]

Una vez conocida la gran diversidad de vehículos submarinos en la actualidad, se expondrá a continuación con mayor detalle lo referente al submarino del GII. Por lo que se presentarán los diversos sensores y actuadores presentes en el mismo. A nivel general el vehículo en cuestión está en proceso de desarrollo y aunque todavía no puede utilizarse en modo AUV, presenta el hardware (sensores y actuadores) necesario para implementarse; actualmente solo funciona como ROV, sin embargo, con el sistema de control que tiene actualmente dificulta mucho su actualización para funcionar en modo AUV. En cuanto al tipo de misión entra dentro de la categoría de manipulación, esto es esencial por el tipo de aplicación en el que se desempeñará (mantenimiento de estructuras marinas), por lo que se cuenta con el respectivo brazo robótico, no obstante, dicho brazo actualmente solo se puede controlar de manera remota. En cuanto a su sistema de propulsión, al igual que la mayoría de los vehículos subacuáticos, cuenta con impulsores de hélice.

### 3.2 Hardware actual

A continuación, se presenta el diagrama de comunicaciones actual del submarino, aunque cabe destacar que este diagrama fue desarrollado a partir de pruebas que se le hicieron al mismo esto ya que como se mencionó anteriormente se tenía poca documentación del vehículo, por lo que

hay ciertas conexiones que aún no se tiene certeza del protocolo de comunicación. Esto sin embargo no afectó en la realización del proyecto, debido a que lo que se hizo fue retirar todo lo referente al antiguo autopiloto y al ir implementando los cambios para la integración del nuevo autopiloto se valoró la posibilidad de reutilizar componentes del antiguo sistema, pero principalmente se consultó las hojas de datos de los sensores/actuadores directamente para su respectiva incorporación.

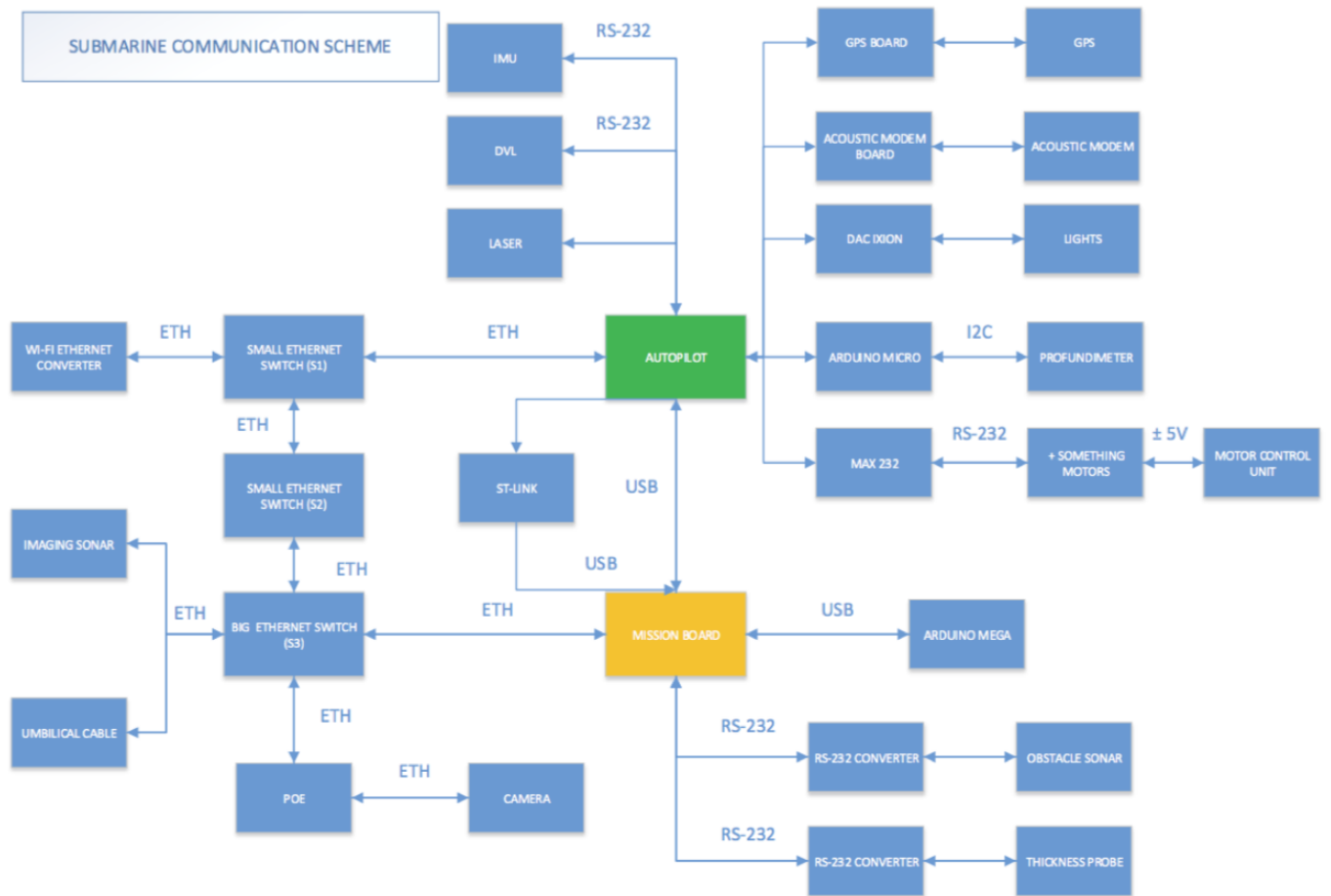


Figura 3. Diagrama de comunicaciones del submarino. Fuente: GII

### 3.2.1 Descripción de sensores

Teniendo como base el diagrama anterior a continuación se hará una descripción de cada uno de los sensores presentes en el submarino:

- IMU (Inertial Measurement Unit): El VectorNav VN-200 que es una Unidad de Medida Inercial que combina acelerómetros, giroscopios y magnetómetros en los 3 ejes, además un sensor de presión barométrico, un receptor de GPS y un pequeño procesador que se encarga de hacer un pre-procesamiento y filtrado de los datos esto con el fin de obtener un Sistema de Navegación Inercial (INS) asistido por GPS que permite aproximar de forma muy precisa la posición la velocidad y la altitud del vehículo. Además, permite aproximar los valores de alabeo (“roll” en inglés), cabeceo (“pitch” en inglés) y guiñada (“yaw” en inglés). [5]
- DVL (Doppler Velocity Log): El NavQuest 600 es un sensor de velocidad por efecto Doppler, con el cual se puede obtener la velocidad de forma rápida, precisa, con rangos de medición significativamente largos y con altitud mínima de operación baja. El efecto Doppler consiste en el cambio de la frecuencia de una señal observada desde un objeto en movimiento, el eco proveniente de las dispersiones aleatorias o de las partículas presentes en el agua lleva la información de las velocidades del instrumento relativas al agua, el eco proveniente de la parte inferior contienen la información de la velocidad relativa al fondo. El sensor lo que hace es transmitir varios pequeños pulsos ultrasónicos a una frecuencia definida y después escucha los ecos tanto del agua como del fondo y a partir de ellos calcula las velocidades a lo largo del sonido emitido. [6]
- Sonar de obstáculos: El Micron Sonar es un pequeño sonar (del inglés SOund Navigation And Ranging, Navegación por sonido y rangos) que cuenta con la tecnología CHIRP (Compressed High Intensity Radar Pulse, Pulso de Radar de Alta Intensidad Comprimido), posee la capacidad para escanear a 360° con un ancho de barrido de 35° en vertical y 3° en horizontal. El principio de funcionamiento radica en producir pulsos ultrasónicos de duración definida y después mostrar los ecos de los objetos que reflejan el sonido al

transductor. Con tecnología CHIRP permite detectar objetos más cercanos entre sí, en comparación con los sistemas tradicionales; esto ya que a diferencia de transmitir los pulsos con una portadora a una misma frecuencia transmite pulsos con diferentes frecuencias definidas a lo largo de la duración del pulso de forma que luego por medio de identificación de patrones se identifica el eco que rebota de los objetos [7].

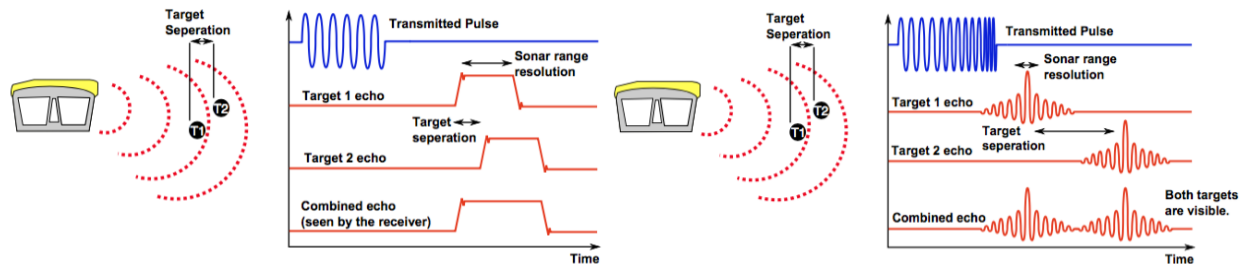


Figura 4. Comparación de sonar de obstáculos con tecnología CHIRP (lado derecho) vs. sistema tradicional (lado izquierdo). Fuente: Trittech International Ltd [7]

- Sonar de imagen: El Gemini 720i es un sonar que ofrece datos casi a tiempo real (con una frecuencia de operación de 720kHz), posee un sensor de velocidad del sonido integrado que ayuda a obtener la imagen más nítida posible (con una resolución de 8mm), ostenta 256 haces que permiten generar imágenes con un ángulo de barrido de 120° y un ancho de barrido de 20°; dichas imágenes permiten detectar objetos/relieves/formas y su respectiva distancia al sensor con un rango que comprende desde 0,2 metros hasta un máximo de 120 metros [8].
- Cámara: El Sony SNC-WR632C es una cámara estilo domo en red para uso en exteriores. Algunas de sus características principales son: estabilización de imágenes, capacidad de adaptación a diferentes condiciones de luz, procesamiento de imágenes para mejorar la visibilidad en las peores condiciones climáticas (lluvia, niebla...), ostenta un grado de protección IP66 e IK10, entre otros [9].
- Sonda de Espesor: El Tritex ndt Multigauge 4000 es una sonda ultrasónica que permite medir espesores de las estructuras sin la necesidad de remover el revestimiento, cuenta con

IPR (del inglés Intelligent Probe Recognition, Reconocimiento de Sondeo Inteligente) que ajusta automáticamente los parámetros del sensor para mejorar su rendimiento y AMVS (del inglés Automatic Measurement Verification System, Sistema de Verificación Automático de Medidas) que asegura que solo muestren las medidas correctas. El principio de funcionamiento se basa en transmitir pulsos ultrasónicos que atraviesan el revestimiento y el metal (la sonda se debió de haber previamente calibrado a la velocidad del sonido del metal que se desea medir) y estos se reflejan al haber un cambio en el material o una pared; los ecos reflejados resuenan a lo largo del metal y un poco en el revestimiento, a partir el tiempo entre los pequeños ecos reflejados se puede obtener el tiempo de los ecos a lo largo del metal que es proporcional al espesor del metal [10].

### **3.2.2 Descripción de actuadores**

Tomando como referencia el diagrama de la Figura 3 ahora se describirán los actuadores presentes en el submarino:

- Luz: SeaLite Lumos 2000 es un foco de luz LED de alto rendimiento (3000 lúmenes), entre sus principales características destacan: la resistencia a los picos de tensión y corriente, al sobrecalentamiento, a las altas vibraciones y golpes [11].
- Propulsores (motores): Para el submarino se cuenta con 2 modelos de motores propulsores (Tecnadyne modelo 300 y 560) de corriente directa sin escobillas especialmente fabricados para pequeños ROV y AUV. Entre sus principales características destacan: el uso de acople magnético para la hélice (de forma que se le coloca un arreglo de imanes y este es accionado por otro arreglo de imanes acoplados al motor que se encuentra dentro del chasis sellado a presión), esto evita cualquier tipo de infiltraciones, además en caso bloqueo la hélice se desacoplaría magnéticamente del rotor evitando daños mecánicos a las partes internas del motor; presenta un motor de altas revoluciones y un tren de engranes planetario de forma que se obtiene una mayor eficiencia; presenta un control de velocidad analógico por medio de una señal de +/- 5 voltios; entre otros. Su principal diferencia radica en la capacidad de empuje ya que el modelo 560 desarrolla una fuerza de empuje de 17 kgf hacia

delante y 10 kgf hacia atrás mientras que el modelo 300 desarrolla una fuerza de empuje de 7,7 kgf hacia delante y 3,2 kgf hacia atrás [12] [13].

### **3.2.3 Otros dispositivos**

- **Modem Acústico:** ATM-900 Acoustic Telemetry Modem es un dispositivo que permite comunicaciones bidireccionales inalámbricas debajo del agua entre un dispositivo local (en este caso la PC de control) y un dispositivo remoto (el submarino, la mini-PC abordo).
- **Pixhawk:** mRo Pixhawk Flight Controller es un microprocesador que fue diseñado originalmente para el manejo de UAV (del inglés Unmanned Aerial Vehicle, Vehículo Aéreo No Tripulado), bajo un proyecto colaborativo denominado Ardupilot, que lo que busca es tener un controlador (Autopiloto) de código abierto para dichos vehículos y sus variantes, solo que con el tiempo se ha ampliado su gama de vehículos soportados bajo la misma plataforma al punto que ya se desarrolló un pequeño submarino ROV básico pero que ha probado tener el potencial para mayores capacidades. Cuenta con varios sensores abordo (giroscopios, acelerómetros, magnetómetros y un barómetro), además tiene varias interfaces de comunicación (UART, CAN, I2C, SPI, Futaba S-Bus, micro-USB externo, RSSI, entre otros) que permiten la expansión a más dispositivos periféricos, también presenta su propio sistema operativo (NuttX OS) dedicado sobre el cual se ejecuta el código que controla al vehículo correspondiente.

## **3.3 Protocolos de comunicación**

A continuación, se presentan los principales protocolos que se utilizan para la comunicación entre los diversos dispositivos periféricos presentes en el submarino, mayoritariamente se utilizarán protocolos de comunicación serial y en algunos casos se utiliza el protocolo de comunicación ethernet.

Ampliando un poco más, los protocolos de comunicación serial que se utilizan varían dependiendo de la aplicación, ya que para ciertos sensores en los que se accede a la información de forma intermitente y no se requiere de altas velocidades de transmisión se utiliza el protocolo

de I2C. Para comunicación entre el Pixhawk y los microcontroladores, la RaspberryPi o ciertos sensores que son de mayores prestaciones se utiliza el protocolo UART/RS-232, estos dos protocolos son prácticamente iguales, de hecho UART se basa en el estándar de comunicación de RS-232, la diferencia es que UART se encarga del envío y recepción de los bits, por lo tanto sus tensiones de operación son a nivel de dispositivos TTL (3,3 o 5 voltios), ya cuando se habla de RS-232 se refiere al nivel de tensión en que se envían los bits, típicamente se suele usar  $\pm 12$  voltios [14]. Lo anterior se debe a que si la comunicación se da en ambientes muy ruidosos o se requiere comunicar por largos trayectos los niveles de señales TTL al ser tan bajas son más vulnerables a que se den pérdidas de datos, debido a esto es que también la comunicación entre contenedores del submarino se da por medio del protocolo RS-232.

Conociendo ya como es que se da la comunicación entre los diversos periféricos, se abordarán los protocolos de comunicación más destacados de la información que se envía, es decir cómo se empaquetan los datos para ser enviados por los medios antes mencionados. Específicamente se explicará de los protocolos de comunicación denominados MAVLink y S-Bus.

### **3.3.1 Protocolo de comunicación MAVLink**

El protocolo Mavlink es un protocolo que nace para que se dé el intercambio de información entre la GCS (del inglés: Ground Control Station, Estación de Control en Tierra) y los UAV. Dicho protocolo está diseñado en forma de librería ligera que contiene los datos necesarios para el intercambio de mensajes; la información de los diferentes mensajes se incluye en ficheros con extensión “.xml”, esto permite que se pueda estructurar y empaquetar desde diferentes lenguajes de programación (como por ejemplo python, C, C++, Java, Swift, WLua, entre otros), de esta manera le da la mayor versatilidad para el desarrollo de soluciones de software. Este protocolo de comunicación facilita su integración a cualquier vehículo o GCS debido a que la definición de sus mensajes se realiza a través de cabeceras en lenguaje C y cuenta con una librería con ciertos mensajes ya previamente definidos y permite la creación de nuevos mensajes personalizados. [15]

La estructura del paquete de datos de Mavlink se basa esencialmente en 3 secciones: una sección de cabecera, una sección del “payload” y una sección de checksum. En la sección de cabecera es en la que se incluye toda la información del mensaje: comienza con un byte de inicio;



le sigue un byte que indica la longitud del mensaje (de 0 a 255); luego tiene un byte que indica el “número de secuencia” (de 0 a 255) que indica el número de paquete si el mensaje es de gran tamaño y requiere dividirse en una secuencia de paquetes, además sirve para ordenarlos en el destino y detecta paquetes perdidos; continúa con un byte que identifica el sistema en caso de que existan varios sistemas en la misma red; le sigue un byte que identifica el componente del sistema en caso de haber varios componentes en el mismo sistema; y por último tiene un número (de 0 a 255) que identifica el mensaje y por ende sería el que indica cómo es que se debe leer el mismo y que significa cada uno de los bytes del paquete. La sección de “Payload” es lo que se denomina como la carga útil del paquete, es donde se encuentra el contenido del mensaje, ya propiamente la información que se desea transmitir, cabe destacar que el máximo payload que se puede enviar es de 255 bytes. Por último, pero no menos importante, el “Checksum” ayuda a verificar que todos los datos estén correctos y que no haya una pérdida o manipulación de los mismos; cabe destacar que el checksum no posee un mecanismo de encriptación de los datos o de autenticación, únicamente verifica la integridad del paquete. [15]

### **3.3.2 Protocolo de comunicación S-Bus**

Este protocolo de comunicación fue desarrollado por Futaba pensando en aplicaciones de control remoto para aficionados. Este se basa en el protocolo de comunicación serial RS232 con la diferencia de que este protocolo está invertido, esto es que mientras que en el serial normal un 0 se mapea a “baja” tensión (0 V o -12V), en el S-Bus se mapea a “alta” tensión (5V o +12V); además tiene la particularidad de que envía los datos a una velocidad (baud rate) no estandarizada de 100 000 baudios, luego también cabe destacar que en la configuración del puerto involucra que por cada byte que se envía se tiene un bit de inicio, un bit de paridad par y dos bits de parada. [16]

Propiamente la estructura del paquete datos del protocolo S-Bus consta de 25 bytes, en los cuales se envía 16 canales de información de 11 bits cada uno, 2 canales digitales y dos banderas (“frame lost” y “failsafe”), organizados de la siguiente manera: 1 byte de cabecera (en HEX 0F); 22 bytes de datos de los diferentes canales (cuyo rango de valores es [192 , 1792] en decimal, siendo en los extremos máxima velocidad en un sentido de giro u otro), empezando por el bit menos significativo del primer canal; le sigue el byte que contiene las banderas y los canales digitales y por último un byte de finalización (en HEX 00). A continuación, se ejemplifica como

es la estructura de los datos del S-Bus, siguiendo la notación canal.bit (siendo el 0 el bit menos significativo del canal) [16]:

Byte 0 (datos): [1.7 1.6 1.5 1.4 1.3 1.2 1.1 1.0]

Byte 1 (datos): [2.4 2.3 2.2 2.1 2.0 1.10 1.9 1.8]

Byte 2 (datos): [3.1 3.0 2.10 2.9 2.8 2.7 2.6 2.5]

Byte 3 (datos): ...

Byte de banderas: [0 0 0 0 failsafe frame\_lost ch18 ch17]

### 3.4 ROS

ROS que viene del inglés “Robotic Operating System”, Sistema Operativo Robótico, es un framework o entrono de trabajo de software libre en el cual se tiene múltiples herramientas, módulos y paquetes con el fin de poder generar una capa de abstracción para configuraciones complejas de software y hardware de robots [17].

ROS surge debido a que típicamente escribir código para el control de robots suele ser una tarea muy compleja ya que constantemente hay un crecimiento y expansión de los sensores y actuadores para dichos dispositivos, además cada robot tiene su propio hardware adaptado especialmente para sus necesidades, lo cual imposibilita un poco la reutilización de código [18].

Existen varios frameworks que se han utilizado tanto en la academia como en la industria para el control de robots, cada uno con sus respectivas ventajas y limitaciones; en Quigley et al. [18] se definen las metas de diseño que se siguieron cuando se programó ROS:

- Peer-to-peer o red entre pares: esto implica que un sistema compilado utilizando ROS se basa en una serie de procesos, con posibilidad de estar en diferentes anfitriones, que se encuentran conectados bajo una topología de red de pares.
- Multilenguaje: cuando se diseñó ROS se tomó en consideración que cada persona tiene un lenguaje de programación preferido, con lo cual se desarrolló para fuese en un lenguaje-neutro; mencionan que soporta C++, Python, Octave y LISP con miras a expandirse a mas lenguajes. Para lograr que fuese realmente multilenguaje se programó de forma que fuese nativo en cada lenguaje, y para la comunicación entre lenguajes se creó una sencilla interfaz de mensajes para la transmisión de información entre los módulos que es común para todos los lenguajes.

- Basado en herramientas: para poder manejar la complejidad de ROS se optó por el diseño de micro-kernel o micro-núcleo en el cual se tiene muchas mini-herramientas que se compilan y ejecutan varios componentes de ROS, en lugar del clásico entorno de desarrollo monolítico. Dichas herramientas pueden ser desde obtener y modificar parámetros, graficar datos provenientes de un mensaje, hasta auto-generar documentación.
- Ligero: típicamente el software que controla robots suele contener muchos drivers y algoritmos que se podrían reutilizar, pero al estar enredado con más código intermedio hace que sea imposible de extraer del contexto para el cual fue originalmente programado. En ROS, en cambio, lo que se busca cada uno de los drivers y algoritmos se programan en librerías aparte, de manera que no se tengan dependencias con ROS. Así, el sistema de compilación de ROS, a su vez compila muchos pequeños módulos de forma que se tiene como muchos pequeños ejecutables que hacen que sea más ligero y además permite la reutilización de código.
- Gratuito y Opensource: lo cual quiere decir que el código fuente de ROS está disponible al público y cuenta con licencia para el desarrollo de proyectos tanto comerciales como no comerciales.

## **Capítulo 4. Marco metodológico**

En esta sección se detalla el procedimiento que se siguió para la ejecución del proyecto. Dicho procedimiento constó de cuatro partes esenciales: primero se caracterizó y estudió la arquitectura actual de la planta, después se procedió al diseño de la nueva arquitectura, luego se dio la implementación de la nueva arquitectura y por último se dio la validación del correcto funcionamiento del submarino una vez implementados los cambios.

### **4.1 Caracterización de la arquitectura actual de la planta**

Para esta etapa se estudió a fondo el funcionamiento actual del submarino, se consultó toda la bibliografía referente al vehículo en cuestión, así como también las hojas de datos de todos los sensores y actuadores, de esta forma se generó una lista de partes que contengan sus respectivas hojas de datos o la caracterización manual en los casos en los que no se encuentren en la bibliografía. Al final de esta etapa también, ya conociendo como es que funciona el submarino y dominando las capacidades del hardware y software con que se cuenta, se procedió a definir los requerimientos que darán solución el problema.

### **4.2 Diseño de la nueva arquitectura**

En esta etapa, se diseñaron todas las adaptaciones de los diferentes sensores y actuadores, además se valoró la posibilidad de utilizar los sensores propios del nuevo autopiloto.

Durante el desarrollo del proyecto no se hicieron transformaciones estructurales al submarino, únicamente se sustituyeron componentes como parte de las modificaciones para la instalación del nuevo autopiloto, las cuales fueron debidamente documentadas y aprobadas por la coordinación del proyecto.

Como parte de las modificaciones, se hicieron adaptaciones a las señales de algunos de los sensores y actuadores, las cuales fueron documentadas por medio de la generación de los respectivos diagramas esquemáticos de los circuitos y los respectivos diseños de los PCB (del inglés “Printed Circuit Board”, Placa de Circuito Impreso).

### **4.3 Implementación de la nueva arquitectura**

Una vez que se tuvo el diseño aprobado con los debidos cambios y actualizaciones del hardware se procedió a la implementación del mismo según los datos de los diagramas correspondientes.

Paralelamente se trabajó en el desarrollo de las adaptaciones al código de programación del controlador, no sin antes haber estudiado las funciones que tiene el autopiloto.

### **4.4 Validación del correcto funcionamiento del submarino**

Una vez que se tuvo el submarino con todas las modificaciones realizadas con éxito, se procedió al diseño y aplicación de pruebas que permitieron evaluar que el mismo cumpliera con los requerimientos para desempeñar las funciones de mantenimiento de estructuras marinas. Se realizaron pruebas que verificaron el funcionamiento de los sensores en ambientes controlados, así como también se verificó que el movimiento (lineal en los ejes “x”, “y” y “z” y rotativo en z) del submarino se dé acorde a las instrucciones que se le den.

## Capítulo 5. Descripción de la solución

En este capítulo se abarcan los detalles para llegar a la solución del problema antes propuesto, siguiendo la metodología descrita anteriormente. Para ello primeramente se establecieron ciertos requisitos, luego se estableció la propuesta de diseño y por último teniendo como base los puntos anteriores se describe en diferentes apartados el trabajo realizado para poder cumplir con los establecido anteriormente.

### 5.1 Requerimientos

A continuación, se presentan los requerimientos planteados para dar solución al problema antes mencionado:

- Se debe tener acceso al código fuente del controlador.
- El diseño del sistema de control debe permitir (aunque no esté implementado todavía) utilizar el submarino en modo AUV.
- El submarino debe detectar posible inundación en cualquiera de sus compartimientos.
- Cualquier modificación del hardware debe caber en el respectivo compartimiento según su función.
- Se debe detectar el sobrecalentamiento en los componentes principales (o de riesgo).
- Se debe adaptar la parte electrónica de todos los sensores y actuadores de forma que funcione con el controlador Pixhawk seleccionado.
- Se debe tener comunicación entre Pixhawk, el computador abordo y el computador en la superficie.
- El diseño debe contemplar la transferencia de datos entre los diferentes contenedores del submarino.
- El diseño debe tener la capacidad de agregar dispositivos periféricos.
- Se debe tener suficiente procesamiento interno para poder analizar los datos provenientes de los sonar (imagen y obstaculos) y las cámaras.
- En caso de algún fallo se debe tener un protocolo de seguridad que evite el mayor daño posible al submarino.

## 5.2 Propuesta de diseño

En la Figura 5, se presenta el diagrama básico de conexiones de la propuesta de diseño para el nuevo sistema de control del submarino.

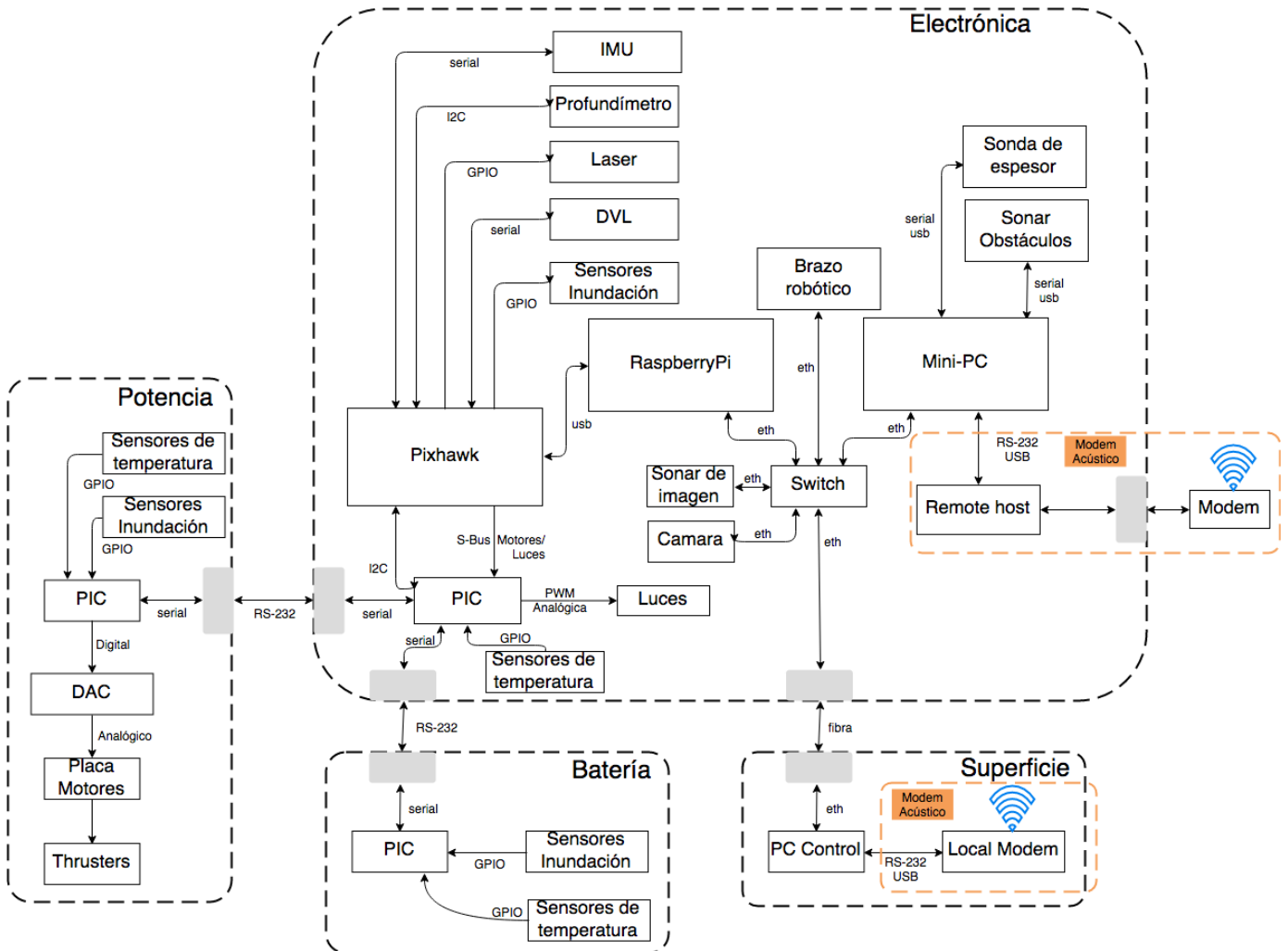


Figura 5. Diagrama básico de conexiones del submarino. Fuente: Elaboración propia utilizando draw.io.

En el diagrama de la Figura 5 se muestra la distribución de los compartimentos que se proponen para el submarino, los cuales han sido nombrados según la función/característica principal en “electrónica”, “potencia” y “batería”.

El contenedor denominado “potencia” hace alusión a que en este es donde se ubica todo lo relacionado a la potencia, de forma que este contiene todos los drivers de los motores, es donde se

transforman a las diferentes tensiones requeridas para alimentación de los diferentes dispositivos (sensores, actuadores, controladores, entre otros).

El compartimiento denominado “batería” como su nombre lo dice es el que contiene la batería, además tiene un BMS (del inglés Battery Management System, Sistema de Gestión de Batería) que es un sistema que permite la carga/descarga óptima de la batería, así como también retorna datos (corrientes y tensiones de cada una de las celdas, temperatura, estado de carga/descarga, entre otros) que muestran la integridad de la misma.

El último compartimiento que es en el cual se basa mayoritariamente este trabajo es el denominado “electrónica” es el que se encargará de toda la adquisición y procesamiento de datos, el control de los diferentes actuadores, la administración de la comunicación entre los diferentes compartimientos y con la superficie (PC control), en general todo lo relacionado al control de navegación; para lo que es todo el control se pretende utilizar el Pixhawk (como control de navegación principal, autopiloto), un microprocesador RaspberryPi (el cual en principio únicamente está pensado utilizarse como interfaz de comunicación entre el control de navegación y el PC de control en la superficie) y un mini-PC abordo (el cual se pretende utilizar entre otras funciones para el manejo de los sensores que requieren de un mayor procesamiento como los sonar). Se propone también integrar microcontroladores que tengan conectados sensores que verifiquen la integridad del submarino en sus diferentes compartimientos; se considerará preliminarmente detección de inundación, temperatura y humedad, aunque se prevé que tenga la posibilidad de implementar nuevos sensores.

En las siguientes secciones se irán describiendo los pasos que se llevaron a cabo para implementar la propuesta de diseño descrita anteriormente.

### **5.3 Caracterización de los motores**

Como parte de las adaptaciones que se deben hacer para la implementación del nuevo autopiloto es la caracterización de los motores, ya que este posee una biblioteca de motores que permite configurar el comportamiento de cada motor, asignándole la contribución en cada uno de los seis grados de libertad de movimiento. En la Figura 6 se puede apreciar los seis grados de movimiento mencionados anteriormente con su respectiva terminología, a saber, alabeo (en inglés



“roll”), cabeceo (en inglés “pitch”), la guiñada (en inglés “yaw”), adelante/atrás (en inglés “forward/back”), izquierda/derecha (en inglés “left/right”), abajo/arriba (en inglés down/up).

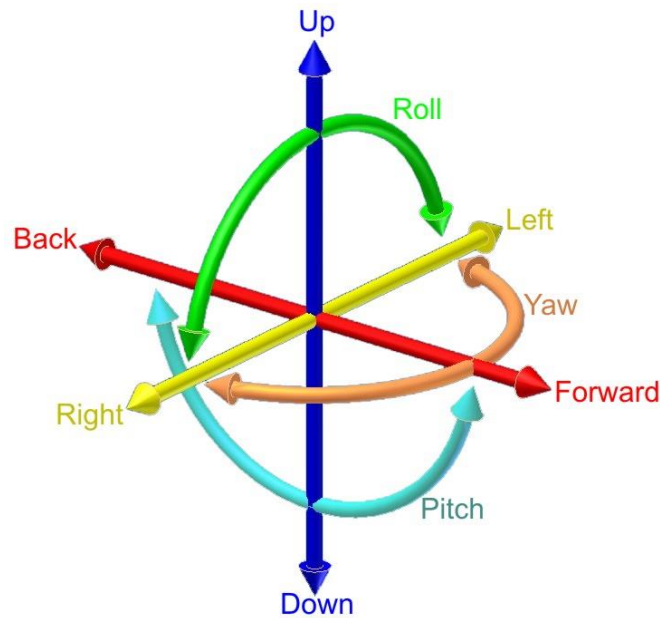


Figura 6. Grados de libertad en un submarino. Fuente: [https://simple.wikipedia.org/wiki/Pitch,\\_yaw,\\_and\\_roll](https://simple.wikipedia.org/wiki/Pitch,_yaw,_and_roll)

Cabe destacar que para el estudio de los movimientos y la fuerzas en submarinos se tiene por convención que el eje “x” positivo corresponde al movimiento hacia adelante, el eje “y” positivo corresponde al movimiento hacia la derecha y el eje “z” positivo corresponde al movimiento hacia abajo.

Teniendo en consideración lo anterior se tomó como referencia la configuración de un pequeño submarino que ya tiene una configuración similar a la del submarino en cuestión. Primeramente, se identificó la dirección de la fuerza de empuje que ejerce la hélice de cada motor (asumiendo para todos los motores el mismo sentido de giro, visto de frente sentido anti-horario); luego, con esta información se desarrolló un diagrama de cuerpo libre (ver Figura 7); por último, se trasladó todas las fuerzas al centro de masa del submarino para ver el efecto de cada motor en el vehículo.

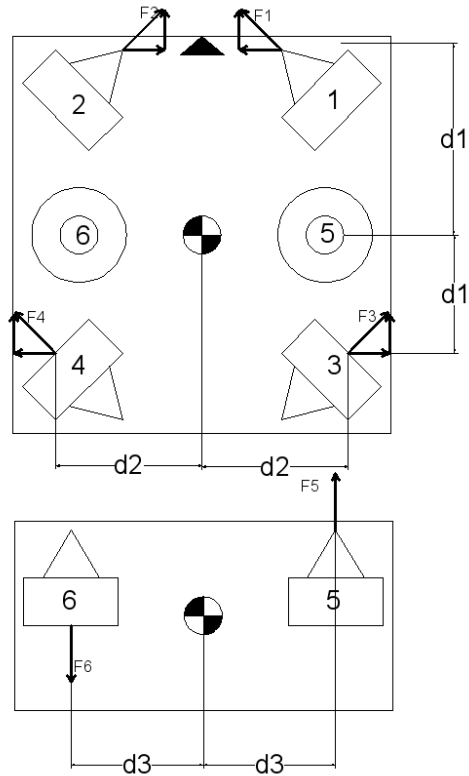


Figura 7. Diagrama de cuerpo libre de submarino de referencia. Fuente: Elaboración propia utilizando AutoCAD.

En la Figura 7 se puede apreciar que la configuración de este submarino es vectorizada, lo cual significa que los motores están girados a 45° por lo tanto se tiene componentes en “x” y “y” de la fuerza de empuje (excepto los motores verticales, el 5 y 6). A continuación, se muestra el resultado en cada motor luego de hacer el traslado de las fuerzas:

Motor1:

- Movimiento en “+x” debido a  $F_{1x}$ .
- Movimiento en “-y” debido a  $F_{1y}$ .
- $Momentos\ en\ Centro\ de\ Masa = M_{F_{1x}} + M_{F_{1y}} = (-d2 \cdot F_{1x} \cdot \text{sen}(90)) + (-d1 \cdot F_{1y} \cdot \text{sen}(90)) \therefore Rotación\ en\ " - z" (yaw)$

Motor2:

- Movimiento en “+x” debido a  $F_{2x}$ .
- Movimiento en “+y” debido a  $F_{2y}$ .

- *Momentos en Centro de Masa* =  $M_{F_{2x}} + M_{F_{2y}} = (d2 \cdot F_{2x} \cdot \text{sen}(90)) + (d1 \cdot F_{2y} \cdot \text{sen}(90))$  ∴ Rotación en " + z" (yaw)

Motor3:

- Movimiento en "+x" debido a  $F_{3x}$ .
- Movimiento en "+y" debido a  $F_{3y}$ .
- *Momentos en Centro de Masa* =  $M_{F_{3x}} + M_{F_{3y}} = (-d2 \cdot F_{3x} \cdot \text{sen}(90)) + (-d1 \cdot F_{3y} \cdot \text{sen}(90))$  ∴ Rotación en " - z" (yaw)

Motor4:

- Movimiento en "+x" debido a  $F_{4x}$ .
- Movimiento en "-y" debido a  $F_{4y}$ .
- *Momentos en Centro de Masa* =  $M_{F_{4x}} + M_{F_{4y}} = (d2 \cdot F_{4x} \cdot \text{sen}(90)) + (d1 \cdot F_{4y} \cdot \text{sen}(90))$  ∴ Rotación en " + z" (yaw)

Motor5:

- Movimiento en "-z" debido a  $F_5$ .
- *Momentos en Centro de Masa* =  $M_{F_5} = (-d3 \cdot F_5 \cdot \text{sen}(90))$   
∴ Rotación en " - x" (roll)

Motor6:

- Movimiento en "+z" debido a  $F_6$ .
- *Momentos en Centro de Masa* =  $M_{F_6} = (-d3 \cdot F_6 \cdot \text{sen}(90))$   
∴ Rotación en " - x" (roll)

A manera de resumen se generó la Tabla 1, la cual se sintetiza todos los cálculos descritos anteriormente y muestra la influencia de cada motor en los seis diferentes grados de libertad del submarino.

Tabla 1. Cuadro resumen de la influencia de los motores del submarino de referencia. Fuente: Elaboración propia.

	Roll (Rotación X)	Pitch (Rotación Y)	Yaw (Rotación Z)	Foward (X)	Lateral (Y)	Down (Z)
Motor 1	0	0	-1	+1	-1	0
Motor 2	0	0	+1	+1	+1	0
Motor 3	0	0	-1	+1	+1	0
Motor 4	0	0	+1	+1	-1	0
Motor 5	-1	0	0	0	0	-1
Motor 6	-1	0	0	0	0	+1

Luego de hacer este análisis se compararon los resultados obtenidos con respecto a los que venían por defecto programados para el funcionamiento del submarino que se tomó como referencia y se comprobó que ambos coincidían, con lo cual se comprueba que el método de caracterización es válido.

Ampliando un poco más, dicha comparación consistió en ver los valores que venían por defecto en el firmware que controla el submarino de referencia y compararlos con los valores que se muestran en la Tabla 1, ya que en ambos casos lo que es de interés es la dirección del movimiento, no así su magnitud, y los signos van de acuerdo con lo establecido en la convención, de forma que si se determinó que la fuerza de empuje de un motor hace que el submarino se mueva hacia delante esta se debe reflejar como un “+1” en el movimiento en X.

Habiendo aclarado lo anterior, la única diferencia que se notó fue con los motores verticales, ya que no seguía la convención y se tomaba el movimiento en “z” positivo hacia arriba, lo cual se justifica si se toma en consideración que en el control remoto si se acciona la palanca para ir hacia abajo (z negativo) se esperaría que el vehículo baje en lugar de subir como lo determina la convención.

Ahora nuevamente se aplicó el mismo procedimiento anterior con las configuraciones propias del vehículo subacuático en cuestión, se analizaron 2 posibles configuraciones una vectorizada y otra normal con los motores alineados. A continuación, se presenta la configuración vectorizada del vehículo, en la Figura 8 ve muestra el diagrama de cuerpo libre de dicha configuración.

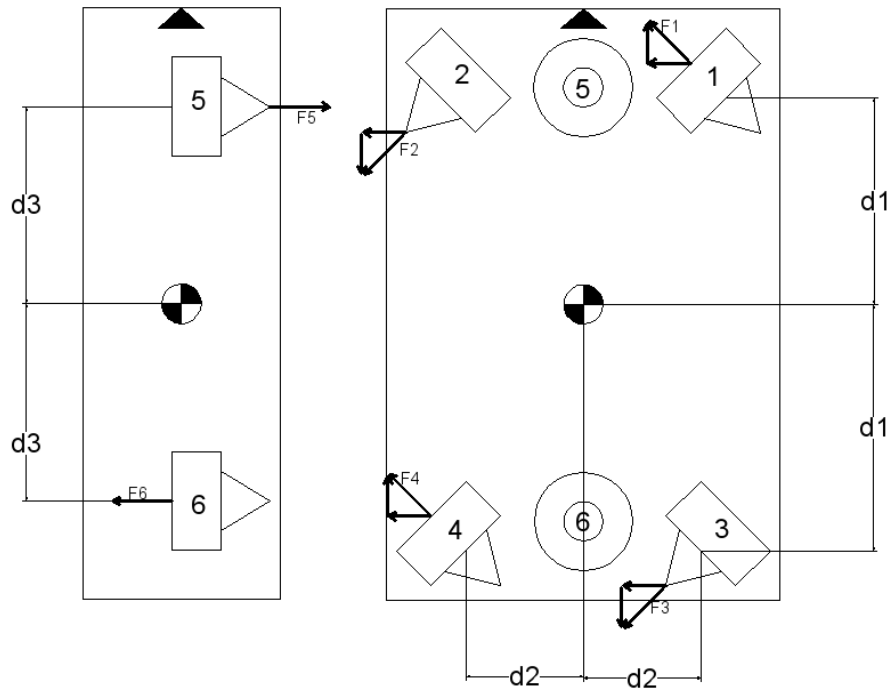


Figura 8. Diagrama de cuerpo libre con la configuración vectorizada del vehículo en cuestión. Cabe destacar que el diagrama de cuerpo libre está hecho siguiendo el sistema europeo de vistas, por tanto, en la vista lateral las fuerzas F5 y F6 son respectivamente hacia arriba (-z) y hacia abajo (+z). Fuente: Elaboración propia utilizando AutoCAD.

#### Motor1:

- Movimiento en “+x” debido a  $F_{1x}$ .
- Movimiento en “-y” debido a  $F_{1y}$ .
- *Momentos en Centro de Masa* =  $M_{F_{1x}} + M_{F_{1y}} = (-d2 \cdot F_{1x} \cdot \text{sen}(90)) + (-d1 \cdot F_{1y} \cdot \text{sen}(90))$  ∴ Rotación en “-z” (yaw)

#### Motor2:

- Movimiento en “-x” debido a  $F_{2x}$ .
- Movimiento en “-y” debido a  $F_{2y}$ .
- *Momentos en Centro de Masa* =  $M_{F_{2x}} + M_{F_{2y}} = (-d2 \cdot F_{2x} \cdot \text{sen}(90)) + (-d1 \cdot F_{2y} \cdot \text{sen}(90))$  ∴ Rotación en “-z” (yaw)

Motor3:

- Movimiento en “-x” debido a  $F_{3x}$ .
- Movimiento en “-y” debido a  $F_{3y}$ .
- *Momentos en Centro de Masa* =  $M_{F_{3x}} + M_{F_{3y}} = (d2 \cdot F_{3x} \cdot \text{sen}(90)) + (d1 \cdot F_{3y} \cdot \text{sen}(90))$  ∴ Rotación en " + z" (yaw)

Motor4:

- Movimiento en “+x” debido a  $F_{4x}$ .
- Movimiento en “-y” debido a  $F_{4y}$ .
- *Momentos en Centro de Masa* =  $M_{F_{4x}} + M_{F_{4y}} = (d2 \cdot F_{4x} \cdot \text{sen}(90)) + (d1 \cdot F_{4y} \cdot \text{sen}(90))$  ∴ Rotación en " + z" (yaw)

Motor5:

- Movimiento en “-z” debido a  $F_5$ .
- *Momentos en Centro de Masa* =  $M_{F_5} = (d3 \cdot F_5 \cdot \text{sen}(90))$   
∴ Rotación en " + y" (pitch)

Motor6:

- Movimiento en “+z” debido a  $F_6$ .
- *Momentos en Centro de Masa* =  $M_{F_6} = (d3 \cdot F_6 \cdot \text{sen}(90))$   
∴ Rotación en " + y" (pitch)

Nuevamente en la Tabla 2 se presenta un cuadro que resume la influencia de los motores del submarino en cuestión, con la configuración vectorizada, en los seis diferentes grados de libertad del submarino.

Tabla 2. Cuadro resumen de la influencia de los motores del submarino en cuestión con la configuración vectorizada. Fuente: Elaboración propia.

	Roll (Rotación X)	Pitch (Rotación Y)	Yaw (Rotación Z)	Foward (X)	Lateral (Y)	Down (Z)
Motor 1	0	0	-1	+1	-1	0
Motor 2	0	0	-1	-1	-1	0
Motor 3	0	0	+1	-1	-1	0
Motor 4	0	0	+1	+1	-1	0
Motor 5	0	+1	0	0	0	-1
Motor 6	0	+1	0	0	0	+1

Por último, se tiene la configuración normal con los motores alineados, para lo cual se tomará como referencia el diagrama de cuerpo libre de la Figura 9.

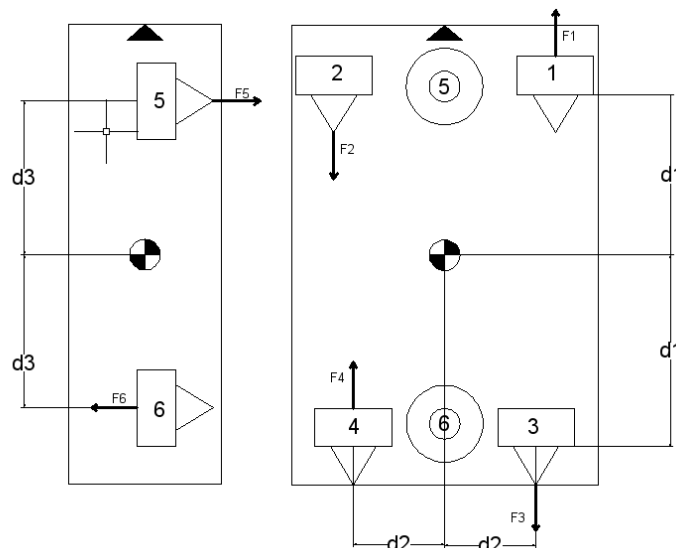


Figura 9: Diagrama de cuerpo libre con la configuración normal de los motores del vehículo en cuestión. Cabe destacar que el diagrama de cuerpo libre está hecho siguiendo el sistema europeo de vistas, por tanto, en la vista lateral las fuerzas F5 y F6 son respectivamente hacia arriba (-z) y hacia abajo (+z). Fuente: elaboración propia utilizando AutoCAD.

Motor1:

- Movimiento en "+x" debido a  $F_1$ .
- *Momentos en Centro de Masa* =  $M_{F_1} = (-d1 \cdot F_1 \cdot \text{sen}(90))$   
 $\therefore$  Rotación en " - z" (yaw)

Motor2:

- Movimiento en “-x” debido a  $F_2$ .
- *Momentos en Centro de Masa* =  $M_{F_2} = (-d1 \cdot F_2 \cdot \text{sen}(90))$   
 $\therefore$  *Rotación en " - z" (yaw)*

Motor3:

- Movimiento en “-x” debido a  $F_3$ .
- *Momento en Centro de Masa* =  $M_{F_3} = (d1 \cdot F_3 \cdot \text{sen}(90))$   
 $\therefore$  *Rotación en " + z" (yaw)*

Motor4:

- Movimiento en “+x” debido a  $F_4$ .
- *Momento en Centro de Masa* =  $M_{F_4} = (d1 \cdot F_4 \cdot \text{sen}(90))$   
 $\therefore$  *Rotación en " + z" (yaw)*

Motor5:

- Movimiento en “-z” debido a  $F_5$ .
- *Momento en Centro de Masa* =  $M_{F_5} = (d2 \cdot F_5 \cdot \text{sen}(90))$   
 $\therefore$  *Rotación en " + y" (pitch)*

Motor6:

- Movimiento en “+z” debido a  $F_6$ .
- *Momentos en Centro de Masa* =  $M_{F_6} = (d2 \cdot F_6 \cdot \text{sen}(90))$   
 $\therefore$  *Rotación en " + y" (pitch)*

Nuevamente, a manera de resumen se generó la Tabla 3, la cual muestra la influencia de los motores del submarino en cuestión, con la configuración normal, en los 6 diferentes grados de libertad del submarino.



Tabla 3. Cuadro resumen de la influencia de los motores del submarino en cuestión con la configuración normal.

Fuente: Elaboración propia.

	Roll (Rotación X)	Pitch (Rotación Y)	Yaw (Rotación Z)	Foward (X)	Lateral (Y)	Down (Z)
Motor 1	0	0	-1	+1	0	0
Motor 2	0	0	-1	-1	0	0
Motor 3	0	0	+1	-1	0	0
Motor 4	0	0	+1	+1	0	0
Motor 5	0	+1	0	0	0	-1
Motor 6	0	+1	0	0	0	+1

Una vez que se implementaron los cambios en el autopiloto, se trabajó colaborativamente para hacer el envío de las señales de control de los motores del contenedor de electrónica al contenedor de potencia y luego hacer el debido procesamiento para poder obtener las señales de control aptas para la placa de motores (señal de -5V a 5V).

Seguidamente se ensamblaron los módulos correspondientes, tanto de la parte de electrónica como de la parte de potencia, para poder realizar los primeros ensayos fuera del agua y así verificar el correcto funcionamiento en los motores. Dichos ensayos consistieron en registrar la dirección de giro de los motores para los movimientos en los diferentes grados de libertad, dichos datos se registraron en la Tabla 4, para la cual se tomó como positivo el sentido de giro de los motores (vistos de frente) cuando estos giraban en sentido anti-horario, esto para ser congruente con la caracterización de los motores y para facilitar la verificación.

Tabla 4. Rotación de los motores según su movimiento en los diferentes grados de libertad, esto para la configuración vectorizada de los motores. Fuente: Elaboración propia.

		Joystick				
		0	1	2	3	
Motor	1	+	-	-	NA	Proa derecha
	2	+	+	-	NA	proa izquierda
	3	+	+	+	NA	popa derecha
	4	+	-	+	NA	popa izquierda
	5	NA	NA	NA	+	proa vertical
	6	NA	NA	NA	-	popa vertical
		+y	+x	yaw -z	+z	
		lateral derecha	adelante	rotación anti-horaria	hacia abajo	

Luego de hacer la respectiva comparación con las fuerzas del diagrama de cuerpo libre del vehículo con la configuración vectorizada (ver Figura 8) se determinó que el sentido de giro correspondía a la caracterización hecha anteriormente, con la única diferencia de que todos los motores tenían el sentido de giro invertido, lo cual indica que cuando se hizo la caracterización para el submarino en cuestión se tomó el sistema referencia al revés, con lo cual únicamente se invirtió el sentido de giro de los motores a través de la plataforma QGroundControl (software de la GCS con el que se conecta el autopiloto Pixhawk) que a su vez hace el cambio en los parámetros del autopiloto.

Por último, una vez que ya se habían hecho las verificaciones del correcto funcionamiento en seco se procedió a hacer la prueba en el agua, sin embargo, antes de realizar esta prueba se verificó la estanqueidad en todos los compartimientos. En la Figura 10 se pueden apreciar imágenes de cuando se hizo las pruebas del submarino en el agua; las cuales consistieron en: primero se empezó a sumergir el submarino lentamente y verificando constantemente que no hubiese indicadores que hagan sospechar que hay infiltración de agua en alguno de los compartimientos (burbujas); seguidamente se procedió a hacer pruebas de movimiento con el submarino aun sostenido por la grúa; por último, se procedió a probar los movimientos del submarino suelto. Como resultado de las pruebas se confirmó que el movimiento del submarino era el adecuado, entre los detalles se notó que los motores mostraron deficiencia en la fuerza para la realización ciertos movimientos (esto se debe a que los motores actualmente se encuentran limitados al 50% de su potencia), luego también en determinado momento se perdió la comunicación (sin embargo esto se debió a que por problemas con la comunicación a través del umbilical se tuvo que recurrir a la comunicación vía wifi, la cual suele ser muy sensible al entrar en contacto con el agua) y el submarino quedo con la última instrucción hasta que volvió a recuperar la comunicación.

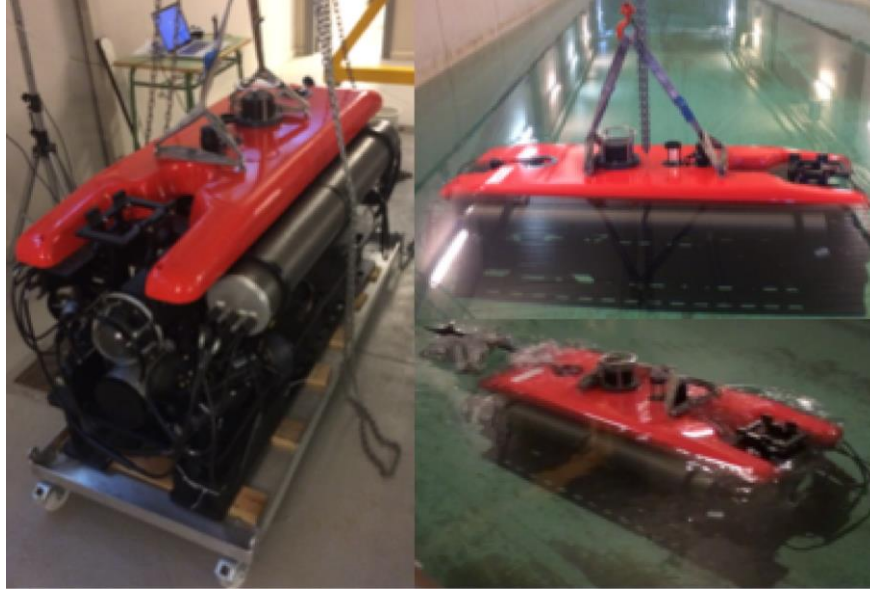


Figura 10. Pruebas con el submarino en el agua. Fuente: Elaboración propia.

#### 5.4 Luces

Para las luces se implementó una etapa de adaptación ya que el Pixhawk cuenta con un código para control de las luces, pero este es un PWM con un ancho de pulso que va desde 1100  $\mu\text{s}$  (apagado) hasta 1900  $\mu\text{s}$  (máximo brillo) y con este no se puede hacer el respectivo control de la luz del submarino en cuestión (requiere de una señal de 0-5V). Por otro lado, en el protocolo de comunicación S-Bus, que es donde se obtienen las señales de control de los motores, cuenta también con un canal que transmite la señal de control de las luces, por lo que se trabajó en conjunto para generar a partir de esa señal a otra de PWM, luego se implementó un conversor DC-DC para elevar la señal del PWM de 3.3V a 5 V, seguidamente se implementó un filtro RC pasabajas y por último se implementó un seguidor de tensión, de esta forma se logró obtener la señal adecuada para el control (ver esquemático del circuito en Figura 11).

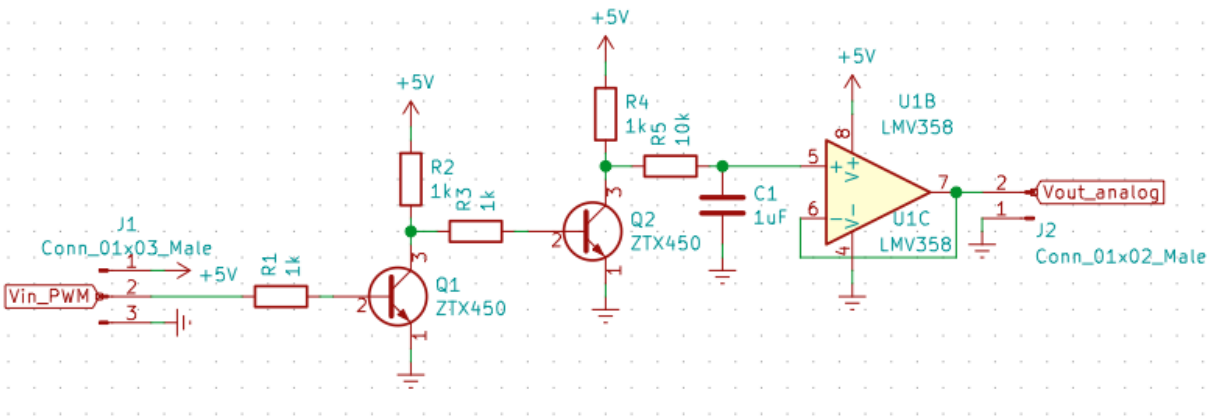


Figura 11. Esquemático del circuito de adaptación para la luz. Fuente: Elaboración propia utilizando KiCad.

Como se puede ver en la primera sección de la Figura 11, el convertor DC-DC se desarrolló por medio de 2 circuitos inversores en cascada, los cuales se diseñaron colocando resistencias de  $1k\Omega$ , esto debido a que de esta forma se asegura la rigidez del circuito (esto al mantener una corriente en el orden de las unidades de miliamperios a lo largo de esta etapa del circuito) y que para lo que se requieren es para protección de los transistores (resistencias R1 y R3), por otro lado R2 y R4 son resistencias de pull-up.

Para el filtro se decidió utilizar una frecuencia de corte que fuera al menos 2 ordenes de magnitud menos que la frecuencia de operación del PWM (100kHz), que tuviese el menor rizado posible, pero también que se pudiese hacer con los componentes disponibles en el lugar de trabajo, con lo cual se llegó a determinar una combinación que dio como resultado la siguiente frecuencia de corte (con un rizado máximo de 0.0125 V en el 50% del ciclo de trabajo del PWM [19]):

$$Frecuencia\ de\ corte = \frac{1}{2 \cdot \pi \cdot R \cdot C}$$

$$Frecuencia\ de\ corte = \frac{1}{2 \cdot \pi \cdot 10\ k\Omega \cdot 1\ \mu F} = 15,9\ Hz$$

En la última sección se implementó un circuito seguidor de tensión con un amplificador operacional, esto porque internamente los focos tienen un driver que, al alimentarse, la señal de control tiene una impedancia conectada a una fuente interna de 5 voltios, con lo cual si se conectaba directamente después del filtro implementado se formaba un divisor de tensión y por ende no se llegaban a apagar las luces. Al implementar esta solución se asegura una alta impedancia después

del filtro y permite opacar la señal interna (del foco) de 5 V, logrando así el control de intensidad lumínica correspondiente.

Seguidamente se procedió a desarrollar un PCB (ver Figura 12) con dicho circuito de forma que se pudiese probar su funcionamiento individual y luego poderlo integrar como un módulo a la tarjeta principal.

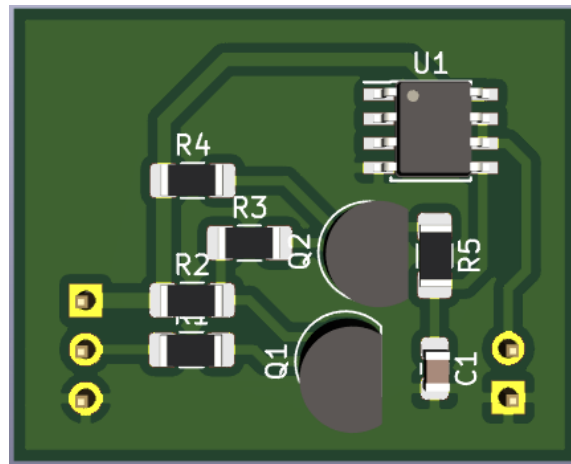


Figura 12. PCB del circuito de adaptación de la luz. Fuente: Elaboración propia utilizando KiCad.

Para validar el correcto funcionamiento de este circuito se utilizó un osciloscopio y se midieron señales en las diferentes etapas del mismo, esto para diferentes intensidades lumínicas. Concretamente en la Figura 13 se puede ver las señales de la salida del PWM del microcontrolador en color amarillo, del PWM luego de la etapa de conversor CD-CD en color magenta y por último la salida rectificadora (después del seguidor de tensión) en color cian; esto para tres diferentes intensidades lumínicas: del lado izquierdo muy baja intensidad, al centro mediana intensidad (aproximadamente el 50%) y del lado derecho completamente encendidas; cabe destacar que el osciloscopio estaba configurado a 1V/división.

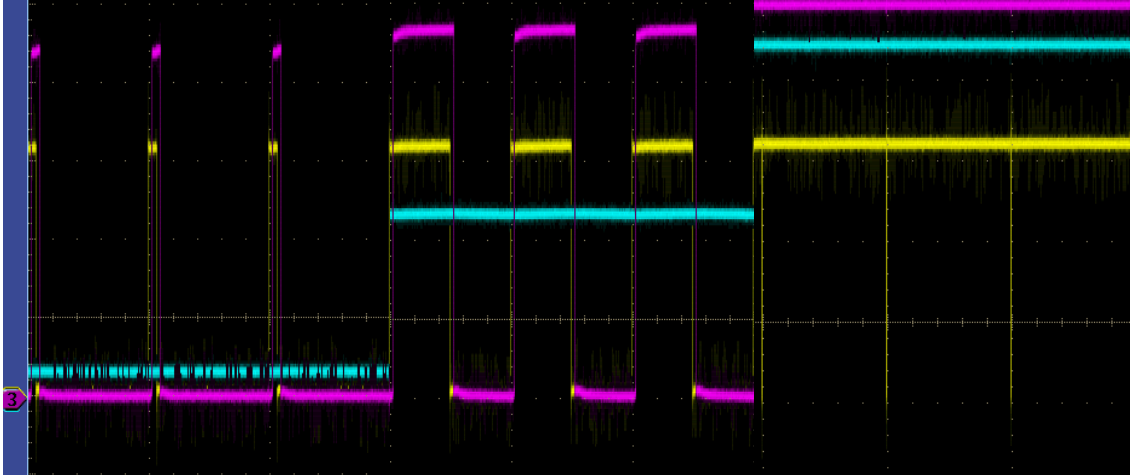


Figura 13. Señales del circuito de control de las luces a diferentes intensidades lumínicas. Siendo la de color amarillo la señal de PWM proveniente del microcontrolador, la de color magenta luego de la etapa del convertor CD-CD y la de color cian la de la salida filtrada. Fuente: Elaboración propia.

## 5.5 Circuito inversor para el S-Bus

Como se mencionó anteriormente, el S-Bus es un protocolo de comunicación serial con ciertas condiciones particulares, una de ellas es que la señal está invertida, por lo que se trabajó colaborativamente en desarrollar un circuito inversor (ver Figura 14) para adaptar dicha señal (antes de ser introducida en el microcontrolador) para luego hacer la lectura e interpretación de dicha señal, y por último ser enviada para al contenedor de potencia para el control de los motores.

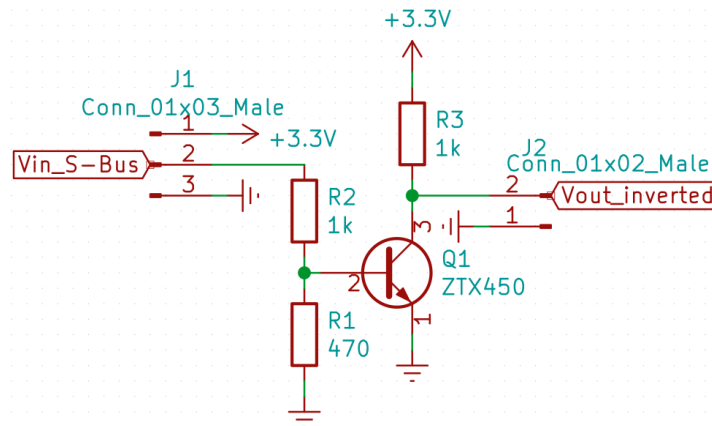


Figura 14. Esquemático del circuito inversor del S-Bus. Fuente: Elaboración propia utilizando KiCad.

Como se puede apreciar en la Figura 14, se tiene un circuito inversor de ganancia unitaria similar al que se usó para el circuito de adaptación de la luz, solo que en este caso se tuvo que hacer en la entrada un divisor de tensión de forma que se asegurara que la tensión de saturación base-emisor no fuese más de 1,1 V, esto debido a que daba desfases en la señal de salida lo cual dificultada en la lectura del microcontrolador. Para el cálculo del divisor de tensión se fijó R2 en 1kΩ y se procedió a buscar una resistencia que cumpliera con lo requerido y se determinó que 470Ω (para R1) era la más adecuada, por otro lado, R3 es una resistencia de pull-up.

$$V_{base-emisor} = \frac{R_1}{R_1+R_2} \times V_{entrada} = \frac{470\Omega}{1000\Omega+470\Omega} \times 3,3V = 1,06V$$

Por último, se procedió a desarrollar circuito impreso (ver Figura 15) que lograra comprobar el funcionamiento individual y luego se pudiese integrar como un módulo a la placa de control general.

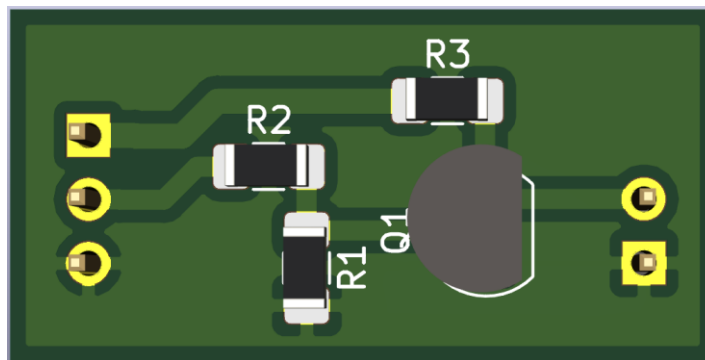


Figura 15. PCB del circuito inversor del S-Bus. Fuente: Elaboración propia utilizando KiCad.

Para validar el correcto funcionamiento de este circuito se procedió a medir con un osciloscopio tanto la señal de entrada como la señal de salida, en la Figura 16 se muestra la señal de entrada original proveniente del S-Bus en color amarillo y la señal de salida invertida en color magenta; cabe destacar que el osciloscopio estaba configurado a 1V/división.

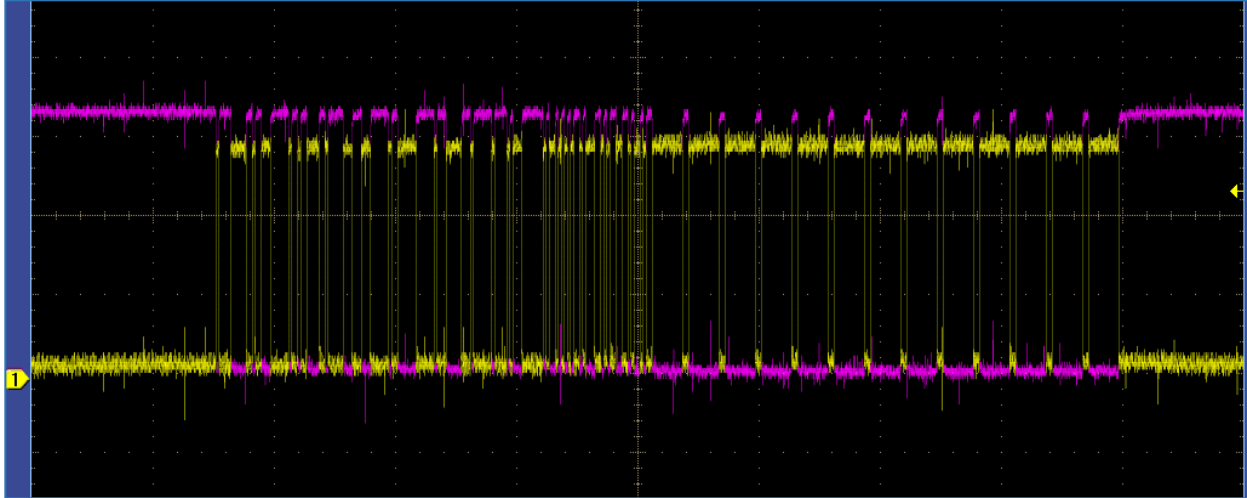


Figura 16. Señales del S-Bus antes y después del inversor. Siendo la señal en color amarillo la original proveniente del S-bus y la magenta la señal invertida del S-Bus. Fuente: Elaboración propia.

### 5.6 PCB MAX232ACSE+

Como se mencionó anteriormente la comunicación entre contenedores se va a realizar por medio del protocolo de comunicación RS-232, por tanto, se diseñó un PCB que contiene al integrado MAX232ACSE+ (ver esquemático en la Figura 17), el cual es un convertor bidireccional de serial TTL (0-5V) a RS-232 ( $\pm 10V$ ) que se utilizó para convertir la señal de comunicación serial (TTL) de los microcontroladores a RS-232, esto para hacer la comunicación entre contenedores.

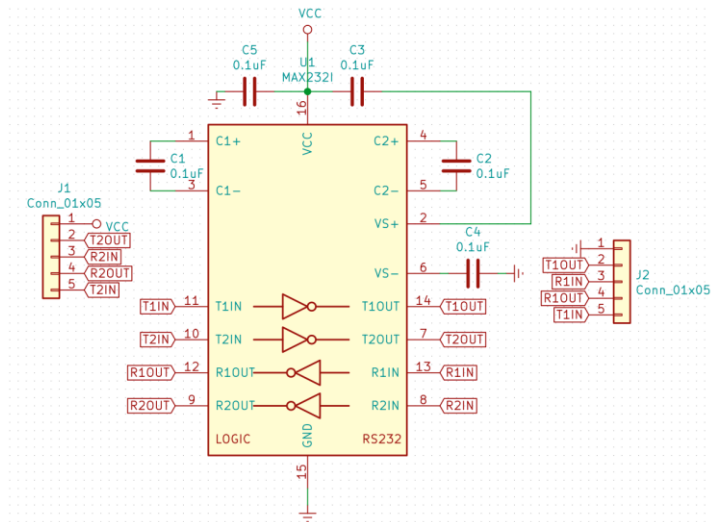


Figura 17. Esquemático del convertor MAX232ACSE+ Fuente: Elaboración propia utilizando KiCad.



En la Figura 17 se puede apreciar el esquemático del PCB diseñado, el cual se realizó siguiendo las recomendaciones del fabricante, por tanto, todo el cableado y selección de componentes se realizó según las especificaciones de la hoja de datos.

Por último, en la Figura 18, se puede apreciar el circuito impreso del convertor, el cual se diseñó, al igual que los anteriores, como un módulo en el cual se pueden hacer pruebas individuales y que luego se pueda integrar a la placa general de control.

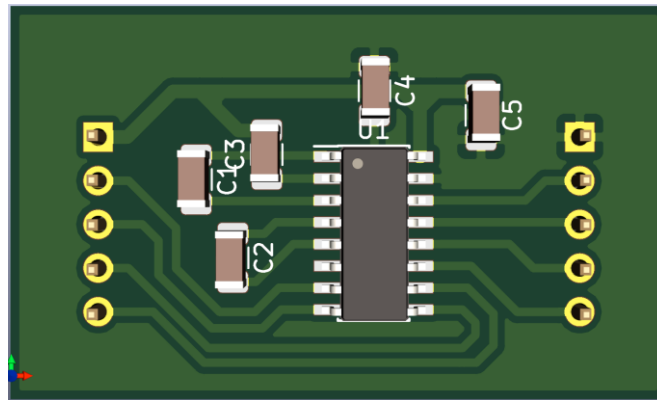


Figura 18. PCB MAX232ACSE+ Fuente: Elaboración propia utilizando KiCad.

## 5.7 Placa general contenedor electrónica

A manera de unificar todo lo referente a los nuevos cambios implementados en el contenedor de electrónica, se desarrolló una placa de circuito impreso en la cual se integran los módulos mencionados anteriormente y el microcontrolador. También se decidió agregar un LED RGB conectado al microcontrolador de forma que sirviese para depurar o simplemente como un indicador. Para determinar el valor de la resistencia se consultó la hoja de datos de fabricante, la cual indica que el valor de corriente en continuo máximo no debe superar los 25mA, con lo cual se decidió tomar como valor máximo 20mA y para una tensión de alimentación de 3,3V se determinó que el valor mínimo debía de ser de 165  $\Omega$ , seguidamente se probó con una resistencia de 330  $\Omega$  y se determinó que en este caso la corriente es de 10mA y al consultar la hoja de datos se obtuvo que el porcentaje de intensidad lumínica relativa es de aproximadamente 80% lo cual es aceptable para esta aplicación.

En la Figura 19 se observa el esquemático de la placa general del contenedor de electrónica, la cual está estructurada de forma que se vea igual que como se vería en el circuito impreso (la ubicación de los módulos y conectores). En los extremos se encuentran los conectores, del lado izquierdo se encuentra primero la alimentación general de la placa y el cable que conecta con el Pixhawk con la información del S-Bus y del lado derecho se encuentra primero el conector con la salida de RS232 que va para el contenedor de potencia y después la señal de control que va para la luz; cabe destacar que para los conectores se quiso estandarizar y se compraron conectores polarizados de la serie SL de Molex con un paso de 2.54mm.

Por último, en la Figura 20 se puede ver el circuito impreso final ya con todos los respectivos módulos que se mencionaron anteriormente, también se colocó ver ahí la IMU de precisión (VectorNav) de forma de que se pudiera aprovechar el espacio que quedó disponible.

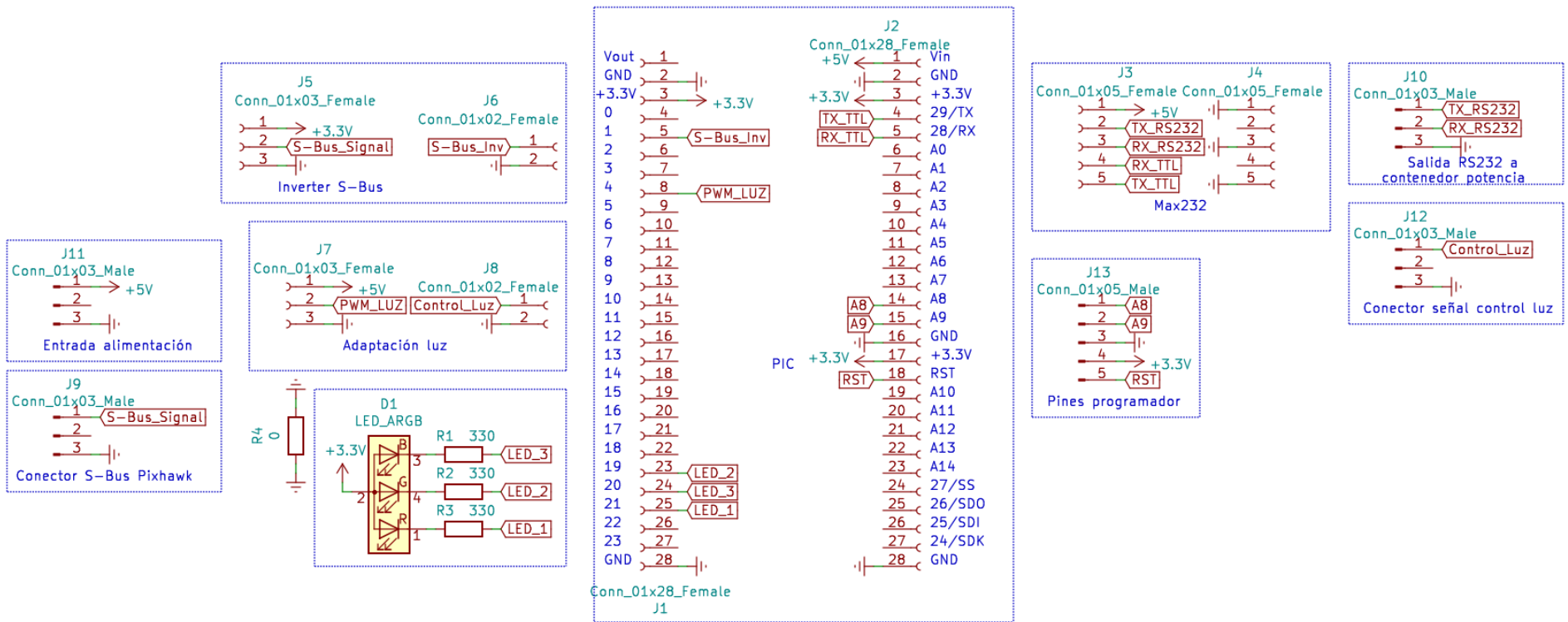


Figura 19. Esquemático de la placa general en contenedor de electrónica. Fuente: Elaboración propia utilizando KiCad.

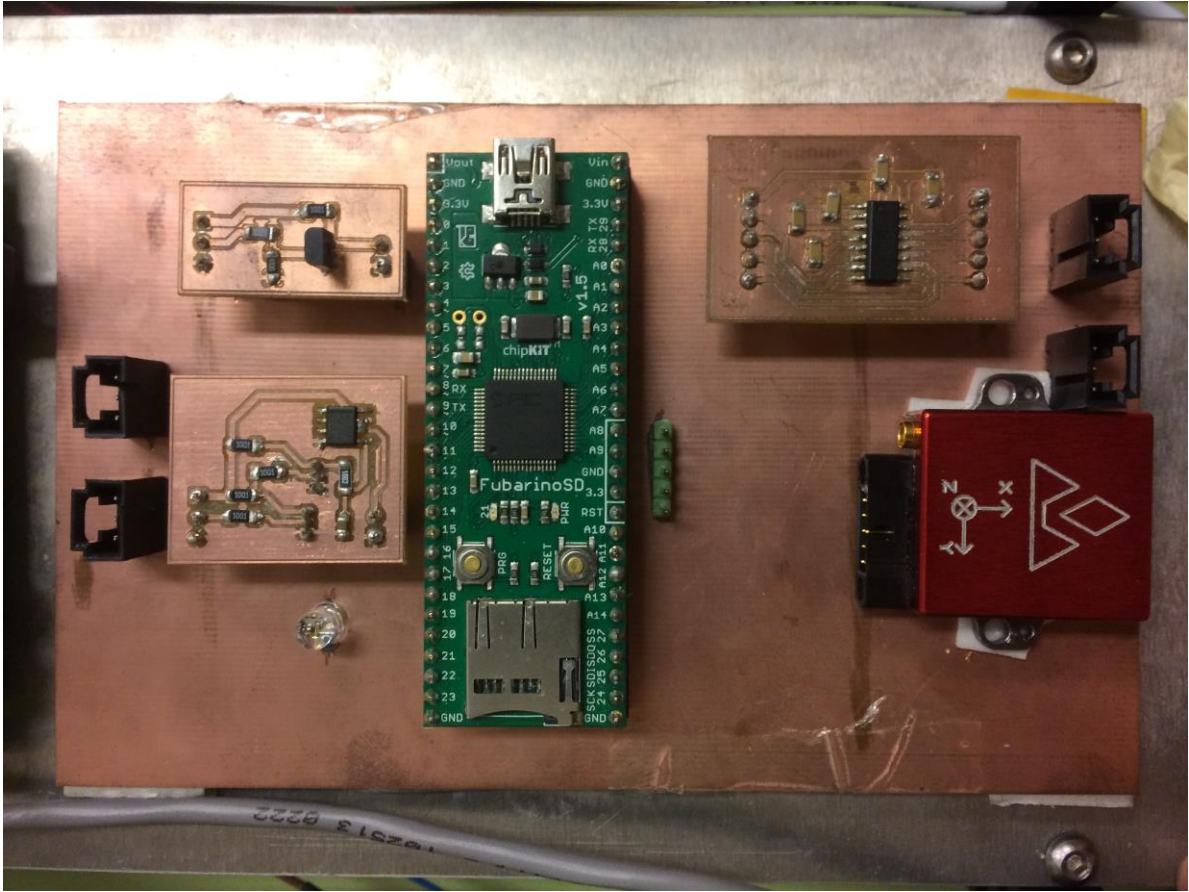


Figura 20. PCB de la placa general del contenedor de electrónica. Fuente: Elaboración propia.

## 5.8 Cambios realizados al firmware del Pixhawk

A manera de resumen se desarrolló la Tabla 5, en la cual se especifican los cambios que se realizaron al firmware del Pixhawk, la cual se estructuró de forma que en la primera columna es la ubicación relativa (a la carpeta ardupilot, que contiene el todo el código fuente del Pixhawk) del archivo modificado, en la segunda columna el nombre del archivo modificado, en la tercera columna las filas en la cuales se hicieron los cambios y por último en la cuarta columna se hace una breve descripción del cambio.

Tabla 5. Control de cambios en el firmware del Pixhawk. Fuente: Elaboración propia.

Ubicación relativa	Nombre	Fila	Descripción
/ArduSub/	GCS_Mavlink.cpp	387-400	Definición función que envía mensaje mavlink temperatures2
/ArduSub/	Sub.h	502-503	Declaración función que envía mensaje mavlink temperatures2
/ArduSub/	APM_Config.h	27,29,32	Habilita para poder utilizar la sección de init, código que se ejecuta a 50Hz y código que se ejecuta a 1Hz respectivamente en el UserCode.cpp
/ArduSub/	UserCode.cpp	N/A	Revisar los comentarios directamente en el archivo.
libraries/AP_Motors/	AP_Motors6DOF.cpp	145-155	Configuración vectorizada de motores para el Submarino Kai (SUB_FRAME_CUSTOM)
libraries/AP_Motors/	AP_Motors6DOF.cpp	157-171	Configuración de motores que esta para depurar, se pensó para ir activando los motores uno a uno (SUB_FRAME_SIMPLEROV_3).
libraries/AP_Motors/	AP_Motors6DOF.cpp	173-189	Configuración que esta para depurar, se pensó para activar todos los motores (SUB_FRAME_SIMPLEROV_4).
libraries/AP_Motors/	AP_Motors6DOF.cpp	190-205	Configuración de motores que esta para depurar, se pensó para ir activando los motores uno a uno (SUB_FRAME_SIMPLEROV_5).
/modules/mavlink/message_definitions/v1.0/	common.xml	3625-3644	Definición de la estructura del mensaje que envía datos de la VectorNav IMU. (id:145 / nombre: VECTORNAV_IMU)
/modules/mavlink/message_definitions/v1.0/	common.xml	3716-3725	Definición de la estructura de un mensaje que se utilizó para hacer pruebas. (ID: 159 / nombre: TEMPERATURES2)
/modules/mavlink/message_definitions/v1.0/	common.xml	3726-3744	Definición de la estructura del mensaje que envía los datos del LinkQuest DVL. (ID: 187 / nombre: LinkQuest_DVL)

A manera de ampliar un poco en el archivo UserCode.cpp es donde se hicieron la mayoría de los cambios y es donde se hace la lectura, decodificación y envío de la información proveniente de la IMU (VectorNav) y del DVL (LinkQuest). A continuación, se amplía el trabajo realizado referente a los sensores antes mencionados.

### 5.8.1 IMU (VectorNav)

A pesar de que el Pixhawk ya cuenta con su propia IMU, se tiene una segunda IMU (VectorNav), la cual tiene mejores prestaciones que la que tiene el Pixhawk, que es la que originalmente tenía el submarino con el sistema anterior, se decidió incorporarla, pero enviando la información por medio de un mensaje de MAVLink, de forma que pueda utilizarse desde el exterior o cuando se llegue a desarrollar el modo AUV se puedan utilizar esos valores para optimizar el control del submarino.

Para realizar lo anterior se programó de forma que primeramente espera el mensaje con la información proveniente de la IMU (coteja el primer y el último carácter del mensaje) y lo almacena en una variable temporal; luego se comprueba el Checksum (el cual se determina haciendo una XOR entre todos los bytes del mensaje) para corroborar que el mensaje sea válido; seguidamente se comprueba la cabecera para identificar el tipo de mensaje y así poderlo decodificar o sino enviar al usuario para su respectiva valoración; por último se obtienen los datos y se envían a través del protocolo de mensajes de MAVLink.

En la Figura 21 se muestran los datos obtenidos de la VectorNav IMU, desplegados desde QGroundControl (es el programa de control de navegación predeterminado para el submarino en la GCS) en la herramienta de inspección de mensajes de MAVLink, con lo cual se comprueba el correcto funcionamiento de la lectura, decodificación y envío de la información de la IMU.

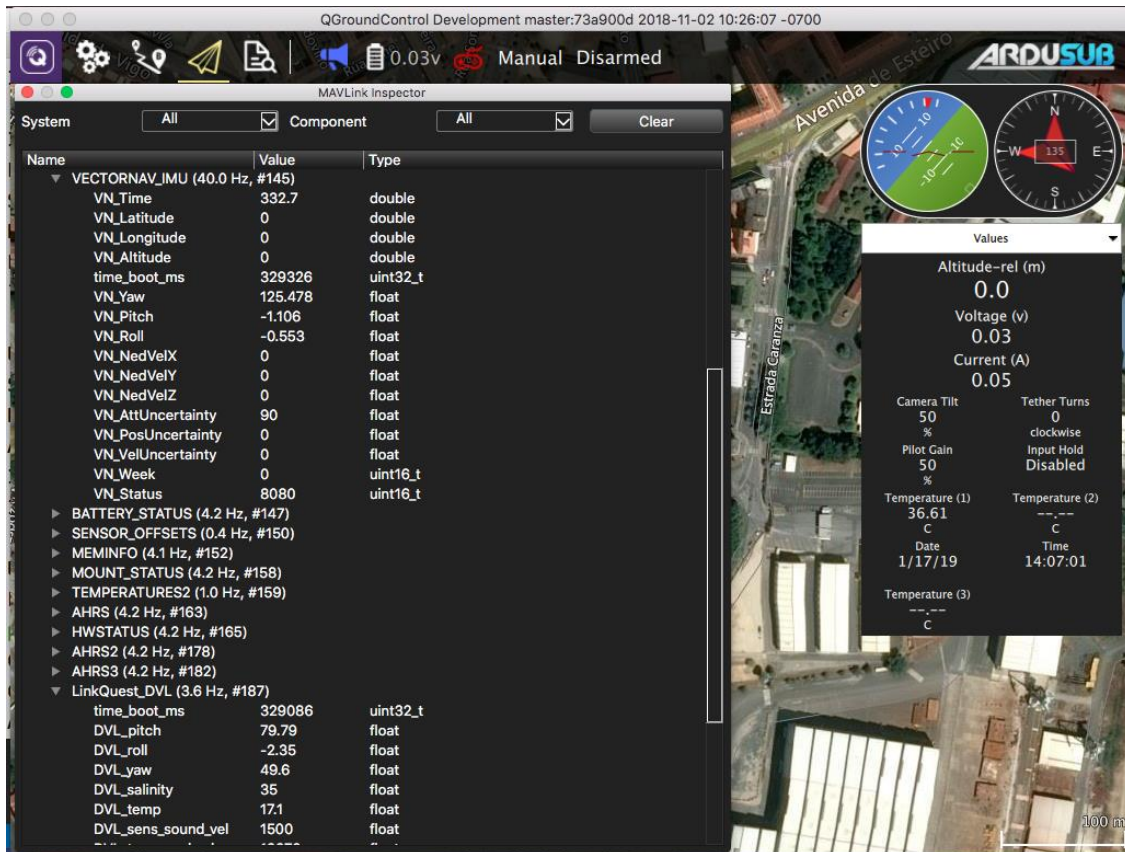


Figura 21. Comprobación del envío de datos de la IMU. Fuente: Elaboración propia.

### 5.8.2 DVL (LinkQuest)

Al igual que la IMU, el DVL se decidió incorporar enviando la información por medio de un mensaje de MAVLink, de forma que estos datos se puedan usar externamente o que sirvan de insumo para cuando se llegue a desarrollar la plataforma AUV.

Antes de desarrollar como fue que se programó hay que destacar la estructura de mensaje proveniente del DVL, ya que esta consta de 5 oraciones que tienen encabezados distintos, pero que tienen en común el primer carácter, además cada oración termina con un salto de línea.

Ya sabiendo lo anterior se programó de forma que primeramente espera por el mensaje proveniente del DVL; seguidamente se comprueban los 2 primeros caracteres de la primera oración (de esta forma se asegura que las oraciones estén en un orden conocido) y luego se verifica que hayan 5 saltos de línea para poder tener el mensaje completo; dado que en este caso no hay Checksum, una vez que se tiene el mensaje completo se procede a hacer una serie de verificaciones



básicas (se verifica el tamaño del mensaje, y el primer carácter de todas las oraciones) para asegurar en cierta forma la confiabilidad del mensaje; una vez aprobadas las verificaciones se decodifica el mensaje, se obtienen los datos y por último se envían a través del protocolo de mensajes de MAVLink.

En la Figura 22 se ve el mensaje con los datos obtenidos del DVL en el inspector de mensajes de MAVLink de QGroundControl, con lo cual se comprueba el funcionamiento de la lectura, decodificación y envío de la información proveniente del DVL.

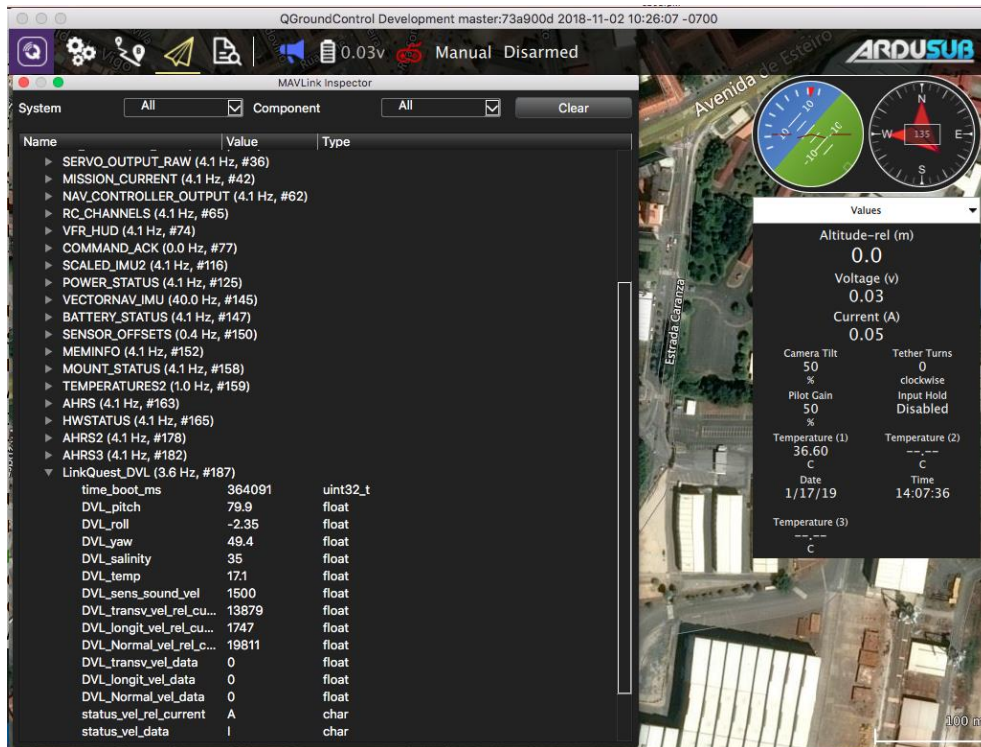


Figura 22. Comprobación del envío de datos del DVL. Fuente: Elaboración propia.

## 5.9 Placa general del contenedor de potencia

Si bien es cierto lo relacionado al contenedor de potencia no es parte esencial de este proyecto se desarrolló una placa que incluía los cambios relacionados al contenedor de potencia, esto ya que era necesario para poder hacer las pruebas correspondientes. En la Figura 23 se puede ver el esquemático de dicha placa, así como los diferentes módulos que se integraron. A continuación, se describirán cada una de las secciones presentes en dicha placa: primeramente, se



cuenta con un módulo Max-232 que es con el cual se convierte la señal RS-232 proveniente del contenedor de electrónica en una señal TTL de forma que pueda ser leído por el microcontrolador; también se tienen 2 DAC conectados en serie que son los que convierten la señal de control de los motores proveniente del contenedor de electrónica a la señal de control que se requiere para la placa de motores (-5V a 5V); por otro lado se tiene un módulo que es un relé que activa la placa de motores; al igual que la placa general del contenedor de electrónica se decidió colocar un LED RGB que sirviera para depurar o como un indicador, el cual se conectó utilizando la misma configuración y las mismas resistencias que en la otra placa; por último, se tiene una sección en la que se previó la conexión de la retroalimentación por parte de los motores para ser enviada devuelta al contenedor de electrónica para su respectivo análisis y procesamiento, pero dado que las señales de retroalimentación son de 0V a 5V y el microcontrolador está limitado a máximo 3.3V, se decidió hacer un divisor de tensión para bajar la señal de máximo 5V a máximo 3V y se buscó con los componentes existentes en el laboratorio una combinación de resistencias que dieran lo más cercano a ese valor y se determinó que  $R_1 = \frac{2}{3} \cdot R_2$  con lo cual luego se llegó a que con la combinación de 3.3k y 4.7k se obtiene un valor de máximo de 2,94V, lo cual ya cumple con lo requerido.

$$V_{salida} = \frac{R_1}{R_1 + R_2} \times V_{entrada} = \frac{4700\Omega}{3300\Omega + 4700\Omega} \times 5V = 2,9375V$$

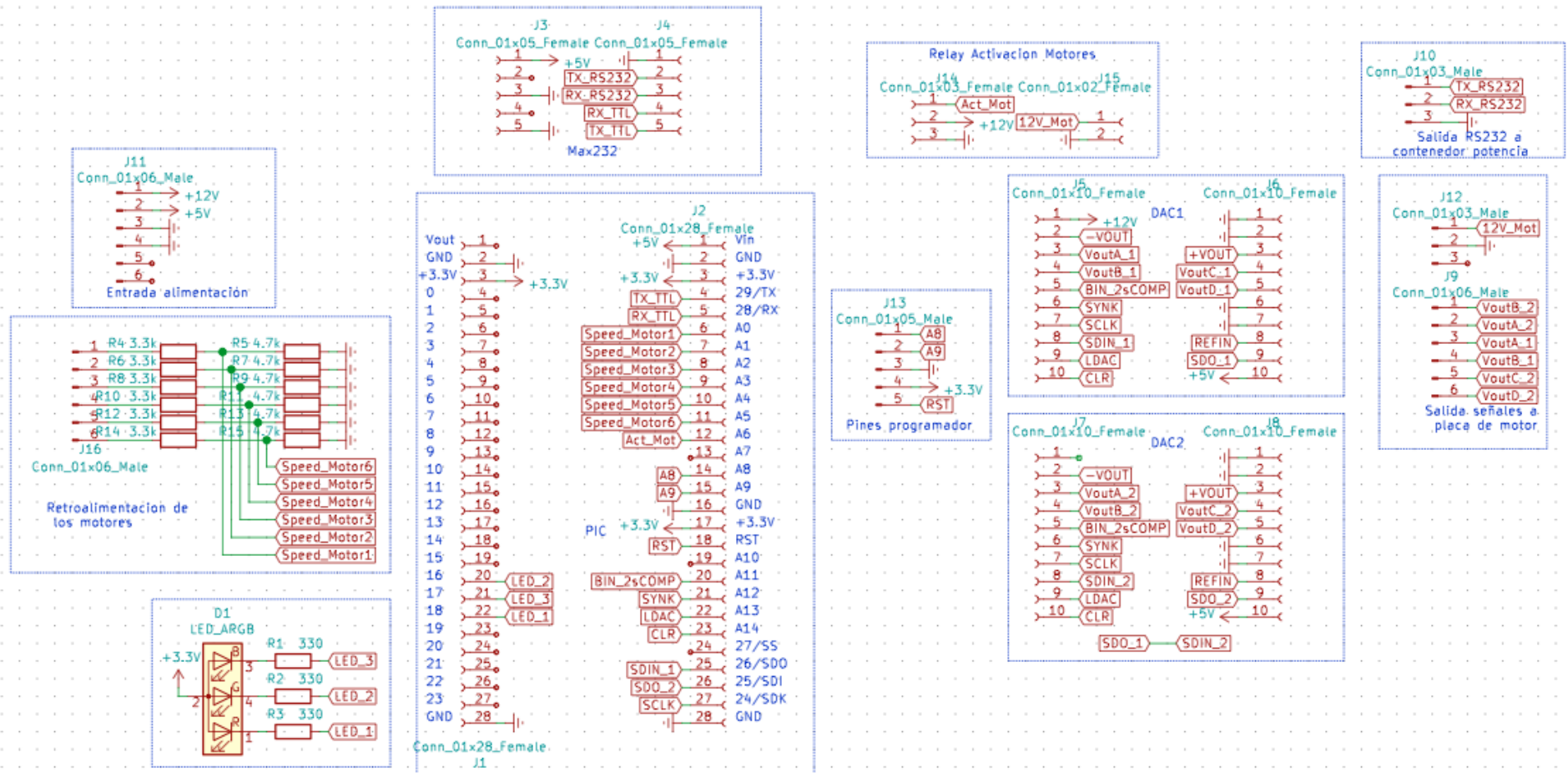


Figura 23. Esquemático de la placa general del contenedor de potencia. Fuente: Elaboración propia utilizando KiCad

Por último, en la Figura 24 se puede ver el circuito impreso final ya con todos los respectivos módulos y las secciones mencionadas anteriormente.

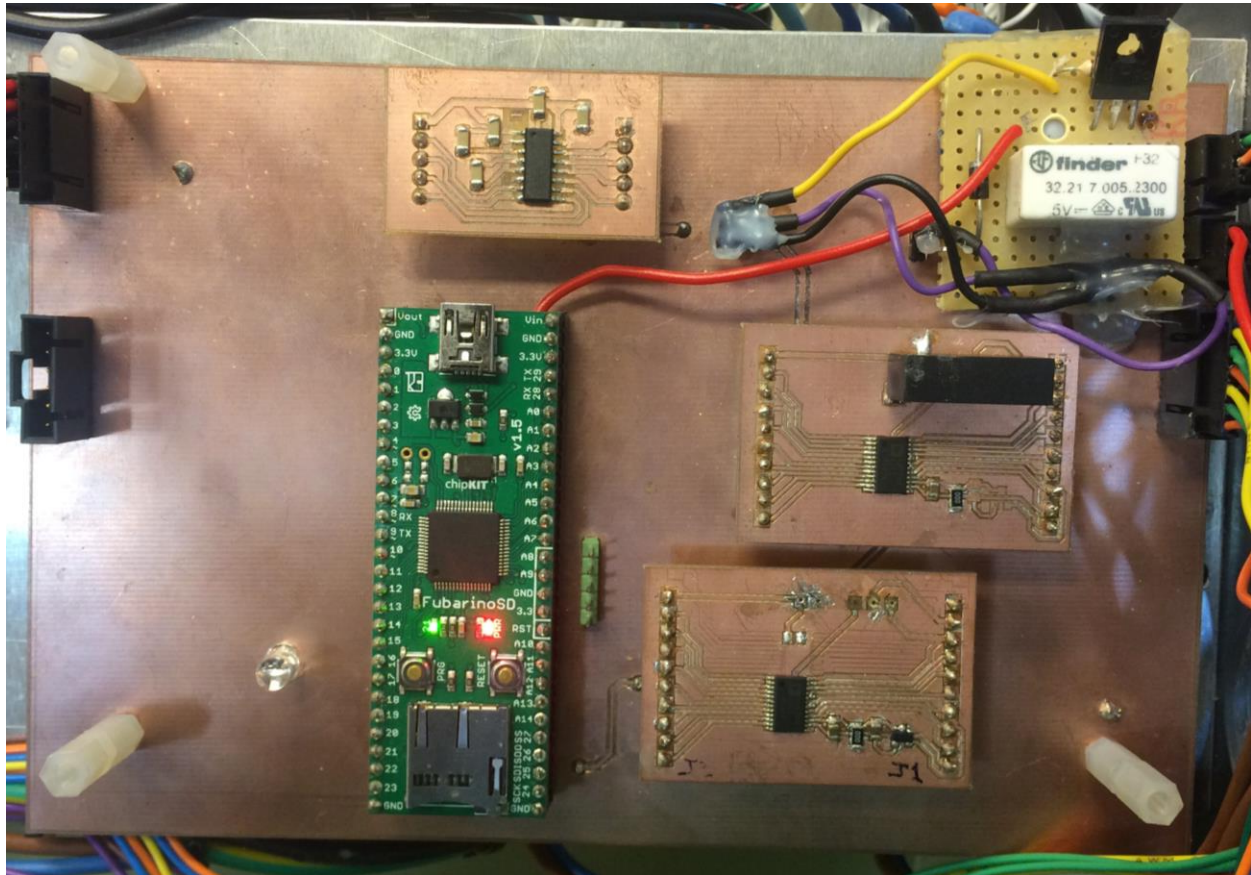


Figura 24. PCB de la placa general del contenedor de potencia. Fuente: Elaboración propia.

## Capítulo 6. Validación

En esta sección se describirá como las tareas realizadas han permitido cumplir con los requerimientos antes establecidos.

- Se debe tener acceso al código fuente del controlador.

Desde el momento en que decidió optar por software/hardware libre ya viene implícito el hecho de tener acceso al código fuente del controlador y el Pixhawk por su naturaleza es un autopiloto del tipo hardware libre colaborativo. Inclusive como parte de las tareas realizadas en este proyecto se modificaron pequeñas secciones de todo el código fuente.

- El diseño del sistema de control debe permitir (aunque no esté implementado todavía) utilizar el submarino en modo AUV.

Esto se cumple puesto que, si se sigue la tendencia de utilizar ROS para los comportamientos autónomos, existe un paquete denominado MAVROS el cual sirve de interfaz entre ROS y los mensajes de MAVLink, con lo cual ya se podría abstraer todos los sensores y otros mensajes de control, inclusive se podrían crear nuevos mensajes (ver Apéndice 2) de forma que el comportamiento AUV se haga desde un alto nivel usando plataformas como ROS.

- El submarino debe detectar posible inundación en cualquiera de sus compartimientos.

Como parte de las funcionalidades que cuenta el Pixhawk es que tiene una entrada digital a través de la cual se puede indicar si hay inundación en el submarino por tanto se consiguieron unos sensores que normalmente están en cero lógico y cuando entran en contacto con el agua cierran el contacto y se emite una señal de uno lógico de forma que activa las alertas del Pixhawk.

- Cualquier modificación del hardware debe caber en el respectivo compartimiento según su función.

Para comprobar este requerimiento se decidió tomar una imagen de compartimiento de electrónica (ver Figura 25) además de tomar las medidas del compartimiento y compararlas con los PCB desarrollados.

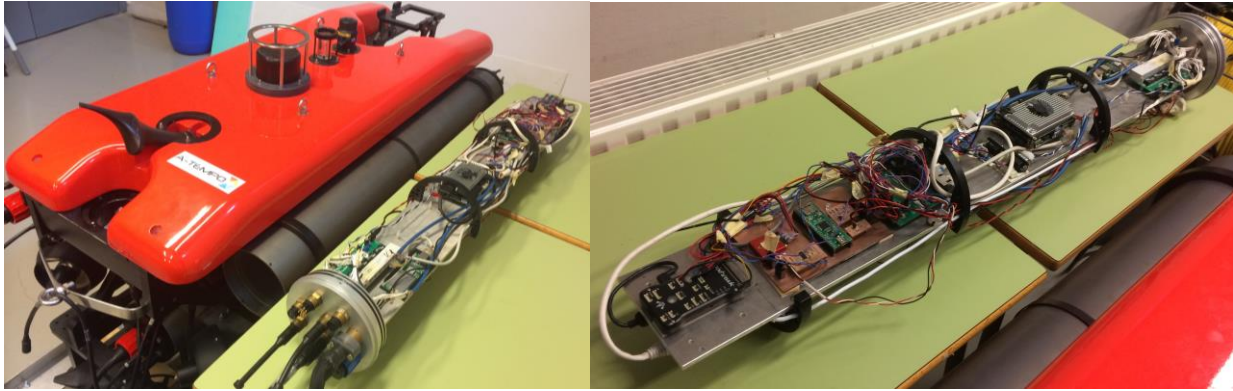


Figura 25. Compartimiento de electrónica del submarino en cuestión. Fuente: Elaboración propia.

Como se puede ver en la Figura 25 los contenedores son con forma cilíndrica la cual tiene un diámetro interno de 15 cm y un largo 1,1m; por otro lado, se tiene una placa que se encuentra en el centro del cilindro que es sobre la cual se ensamblan todos los componentes, sensores, actuadores, controladores y microprocesadores, entre otros; dicha placa tiene dimensiones de 12 cm de ancho por 1m de largo. Luego se tiene que el PCB de la placa general del contenedor de electrónica (ver Figura 20) mide 9,14 cm de ancho por 14,5 cm de largo, con lo cual se comprueba que tiene las dimensiones adecuadas para caber dentro del contenedor correspondiente. Por último, el PCB de la de la placa general de potencia (ver Figura 24) mide 11,4 cm de ancho por 17,2 cm de largo, con lo cual se comprueba que tiene las dimensiones adecuadas para caber dentro del contenedor correspondiente, cabe destacar que todos los contenedores tienen las medidas.

- Se debe detectar el sobrecalentamiento en los componentes principales (o de riesgo).

Dicho requisito no se llegó a cumplir al 100% ya que en este momento el único componente que tiene sensor de temperatura es el autopiloto, esto se debe a que primeramente se tenía la meta de poner a funcionar el submarino con las funcionalidades básicas para ver que su adecuado funcionamiento con el nuevo autopiloto. Una vez llegado a este nivel se pretende agregar lo que se llamó “mantenimiento del hogar” que se refiere a todos los sensores de temperatura y humedad (incluso se pensó en algún momento sensores de corriente en los motores) que sirven principalmente para alertar cuando existe comportamiento anormal que requiera ya sea el retorno a flote para su respectiva revisión o alguna corrección en el momento.



- Se debe adaptar la parte electrónica de todos los sensores y actuadores de forma que funcione con el controlador Pixhawk seleccionado.

Este se requerimiento se logró, pero no se integraron los sensores directamente al control de autopiloto, si no que se envían como un mensaje de MAVLink previendo hacer la manipulación desde alto nivel, utilizando, como se mencionó anteriormente, plataformas como ROS.

- Se debe tener comunicación entre Pixhawk, el computador abordo y el computador en la superficie.

Este requerimiento se cumple ya que la topología que se está utilizando es una red ethernet, con lo cual se tiene comunicación entre todos los dispositivos que se encuentren conectados a la red (incluidos el mini-PC abordo y la GCS), para el caso particular del Pixhawk este se conecta y transfiere datos a través de microprocesador RaspberryPi que si se encuentra conectado a la red. Cabe destacar que como parte de las tareas realizadas implicó cambiar la IP de la red asociada al RaspberryPi, esto porque en el submarino en cuestión ya se tenía una red establecida (ver Apéndice 3). En la Figura 26 se puede ver la conectividad entre la estación en tierra y el autopiloto Pixhawk.

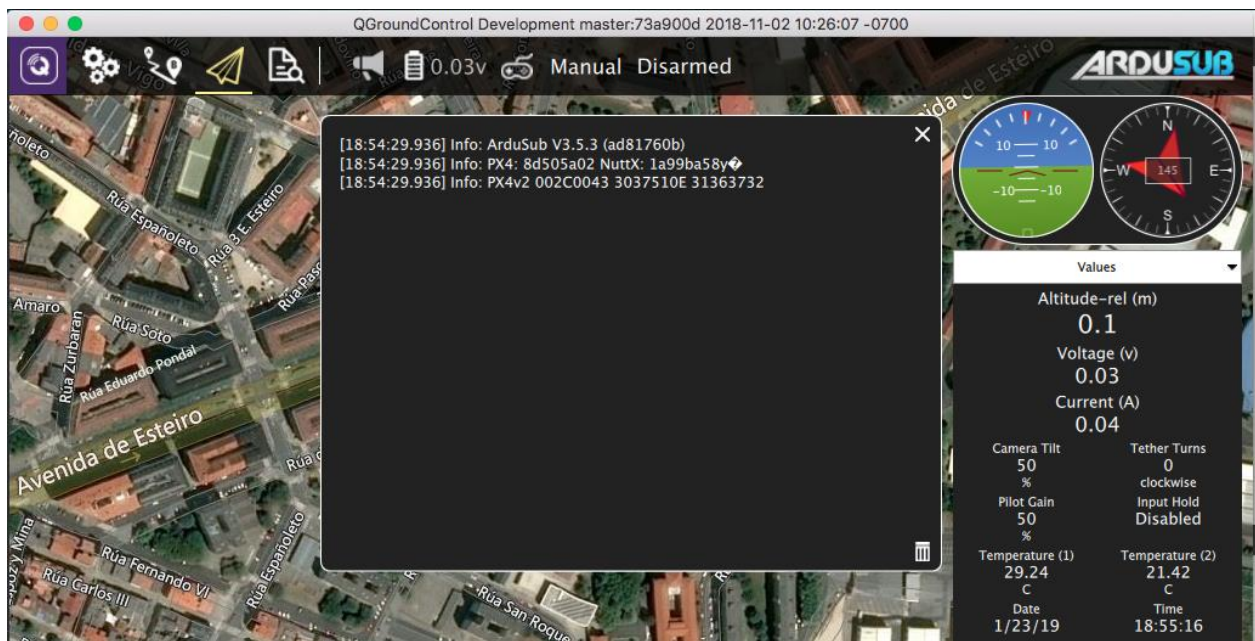


Figura 26. Plataforma del QGroundControl (GCS) conectada al Pixhawk. Fuente: Elaboración propia.

- El diseño debe contemplar la trasferencia de datos entre los diferentes contenedores del submarino.

Este requerimiento se cumple, ya que si bien es cierto actualmente solo están interconectados los contenedores de potencia y electrónica, ya se tiene previsto que todo lo referente a la comunicación entre contenedores se hará por medio del protocolo de comunicación RS-232, y por ende ya se tiene el hardware necesario para lograr la intercomunicación bajo ese protocolo entre los demás contenedores.

- El diseño debe tener la capacidad de agregar dispositivos periféricos.

Con el Pixhawk ya se tiene la prevista para agregar nuevos dispositivos periféricos, sin embargo, si aun así hiciera falta agregar más sensores se pretende que estos se conecten a un microcontrolador que a su vez se conecte con el Pixhawk ya sea por protocolo de comunicación serial o inclusive I2C de forma que transmita la información de los sensores al Pixhawk para su debido procesamiento.

- Se debe tener suficiente procesamiento interno para poder analizar los datos provenientes de los sonar (imagen y obstáculos) y las cámaras.

Actualmente debido a lo que se mencionó anteriormente, de que primeramente se querían hacer las pruebas más básicas con el submarino, para efectos de este proyecto no alcanzo el tiempo para determinar cómo obtener los datos y procesarlos a nivel de línea de comandos (actualmente no se tiene el SDK, del inglés Software Development Kit, Kit de Desarrollo de Software) si no que actualmente lo que se puede hacer es acceder desde ethernet a la dirección correspondiente del sensor por medio de la aplicación grafica que supe el fabricante; por tanto aún no se ha determinado con exactitud cuál mini-PC será el más adecuado para llevar a cabo el procesamiento de los sensores antes mencionados.

- En caso de algún fallo se debe tener un protocolo de seguridad que evite el mayor daño posible al submarino.

Al implementar el autopiloto Pixhawk implícitamente tiene la opción de poder configurar acciones a tomar en caso de que ocurra algún posible fallo. Además, cuando se desarrolló la comunicación entre los microcontroladores se tomó en consideración algunas medidas en caso de algún fallo de comunicación o de información corrupta.

## Conclusiones

Como se mencionó anteriormente este proyecto es parte de una línea de investigación por tanto el trabajo realizado contribuye a una parte importante para seguir la línea rumbo al desarrollo de la completa autonomía del submarino. A continuación, se presentan las conclusiones que lo respaldan.

El Pixhawk ofrece varias ventajas sobre el autopiloto anterior, esto se manifiesta desde el hecho de ser del tipo código abierto, ya que tiene acceso al código fuente, se tiene el respaldo de una comunidad de desarrolladores que puede facilitar la resolución de problemas, también al ser un sistema común en múltiples vehículos facilita el acceso a la documentación general (como tutoriales, foros en línea, blogs, etc.) aparte de los manuales oficiales. También se detectó una desventaja al trabajar con este tipo de autopiloto, esto porque al ser el mismo código para múltiples vehículos, el código fuente involucra múltiples archivos y bibliotecas, al punto que dificulta su edición al no tener claro cuáles son las dependencias y los archivos que afectan directamente al vehículo que se está trabajando.

La caracterización / estudio de la planta fue esencial para determinar la configuración más óptima para la propuesta de diseño, además de servir de insumo importante para la etapa de diseño al conocer el funcionamiento y las capacidades de cada transductor.

Trabajar con el Pixhawk permite una gran versatilidad en la configuración de los motores, esto se debe a que en la forma en la que se programó el código de dicho autopiloto se abstrajo al punto que admite prácticamente cualquier configuración y orientación de los motores, siempre que se conozca la influencia de cada motor en los seis grados de libertad del vehículo en cuestión.

Con el protocolo S-Bus simplificó la tarea del envío de las señales de control de los motores, dado que desde un solo canal de comunicación se pudo obtener todos los valores de dichas señales, luego solo era necesario su empaquetamiento y envío al contenedor de potencia para ser adaptado e introducirse a la placa de control de los motores. Además de lo anterior, también favoreció para el control de intensidad de las luces, ya que dentro del mismo protocolo se configuró para que también se tuviera acceso a la señal de control de las luces.

Con la modulación de los circuitos se facilitó la etapa de desarrollo, ya que permitía probar cada sección por aparte y en caso de algún fallo reemplazar únicamente el módulo implicado;



siempre teniendo en cuenta que cuando ya se tiene un diseño definitivo, se debe incluir todo en una única placa definitiva más compacta.

Con el protocolo de mensajes de MAVLink se facilita la comunicación con la estación de control en tierra y su integración con ROS, esto porque permite agregar nuevos mensajes (aparte de los ya existentes) que puedan estar desplegando información en el panel de la GCS; además admite la obtención de los datos de los sensores para hacer el control en alto nivel utilizando ROS (por medio la herramienta MAVROS), con lo cual se agilizaría el proceso para llegar a la completa autonomía del submarino.

A nivel general proyectos relativos a robótica submarina permiten a los estudiantes ampliar los horizontes y darse cuenta que se tiene la capacidad para el trabajo con este tipo de vehículos que en Costa Rica son poco comunes en cuanto a la investigación y extensión, con lo cual este tipo de intercambios / convenios interuniversitarios son muy valiosos ya que permiten la generación de conocimientos e inclusive pueden servir de insumo para el desarrollo nuevas investigaciones en estas áreas.

## Recomendaciones y trabajo futuro

A continuación, se enumeran las recomendaciones referentes al trabajo realizado:

- Se recomienda la reducción de las placas diseñadas, ya que si bien las que se diseñaron funcionan, no obstante, estas como se estaba en etapa de desarrollo se diseñaron a nivel modular ya que se pretendía probar cada módulo por aparte antes de conectar todo en conjunto, por lo tanto, este tipo de diseños si se dejan como definitivos, estos pueden causar bastantes problemas especialmente en la parte de los contactos.
- Se recomienda verificar y retomar la configuración de las acciones a tomar cuando haya una pérdida de comunicación, ya que como se vio en la prueba del submarino en el agua se vio que cuando había perdida de comunicación el quedaba con la última instrucción.
- Se recomienda hacer una revisión de las tierras, ya que como se puede ver en la Figura 16 la señal del SBus tanto antes como después de pasar por el circuito inversor se ve que es muy ruidosa y eso se podría deber a una mala puesta de las tierras.

Seguidamente se enumerará trabajo futuro que se debe realizar con el fin seguir avanzando con el objetivo de la línea de investigación:

- Conexión e implementación los sensores para la realización de lo que se llamó “mantenimiento del hogar” (sensores de temperatura, sensores de humedad, entre otros).
- Búsqueda del SDK de los sonar (de imagen y obstaculos), y cámara de forma que se pueda obtener y procesar los datos provenientes de dichos sensores.
- Realizar la conexión e implementación de la retroalimentación por parte de los motores.
- Investigar acerca del paquete MAVROS y ver cómo se puede adaptar lo que ya existe en el en MAVLink de forma que se pueda empezar a emigrar a este nuevo tipo de control.
- Realizar la conexión entre los contenedores faltantes (batería-electrónica y batería-potencia.)

## Estudio económico

Como parte del análisis económico se obtuvieron los siguientes costes aproximados (ver Tabla 6), cabe destacar que si bien el rubro de remuneración es bastante elevado esto se debe a que abarca tanto los costes de manutención por 5 meses (hospedaje, alimentación, seguro, etc.) como los costes transporte al proyecto (tiquetes aéreos).

Tabla 6. Costes Aproximados. Fuente: Elaboración propia.

Costes aproximados		
Cantidad	Nombre	Costo
1	Ayudante Técnico ½ tiempo (€515 mensuales)	\$3 000
1	Remuneración (€750 mensuales)	\$4 350
1	Sensores, componentes, conectores y PCBs	\$500
1	Pixhawk 2 Autopilot	\$238
1	RaspberryPi	\$35
	<b>Total</b>	<b>\$8 123</b>

Como parte de la validación económica del trabajo realizado se procedió a hacer una comparativa entre el coste aproximado del submarino (el cual es aproximadamente €700 000 o \$818 140) lo cual implica que el coste del proyecto representa un 1% del coste total del proyecto y entre las principales ventajas que se obtuvo es que ahora se puede modificar el código del nuevo controlador, además por medio del paquete MAVROS se puede lograr un punto de puente entre los mensajes de MAVLink y la plataforma ROS que ha demostrado ser de gran importancia para el desarrollo de robótica compleja.

## Referencias

- [1] F. J. Bellas Bouza, Á. Deibe Díaz, V. Díaz Casás, R. J. Duro Fernández, M. Lema Rodríguez, F. López Peña y M. Míguez González, «Submarino Autónomo para la Inspección de instalaciones Off-shoRe (SAILOR),» [En línea]. Available: <http://www.gii.udc.es/proyectos/detalle/355>. [Último acceso: mayo 2018].
- [2] Duvulgación PLOCAN, «Descripción del oceano profundo,» 2017. [En línea]. Available: <http://divulgacion.plocan.eu/descripcion-del-oceano-profundo/>. [Último acceso: octubre 2018].
- [3] J. M. de la Cruz García, J. Aranda Almansa y J. M. Girón Sierra, «Automática marina: una revisión desde el punto de vista del control,» 2012. [En línea]. Available: <https://polipapers.upv.es/index.php/RIAI/article/download/9590/9531>. [Último acceso: Octubre 2018].
- [4] H. A. Moreno, R. Saltarén, L. Puglisi, I. Carrera, P. Cárdenas y C. Álvarez, «Robótica Submarina: Conceptos, Elementos, Modelado y Control,» 2014. [En línea]. Available: <https://polipapers.upv.es/index.php/RIAI/article/download/9474/9444>. [Último acceso: Octubre 2018].
- [5] VectorNav Technologies, *VN-200 User Manual*, 2017.
- [6] LinkQuest Inc., *User's Guide: NavQuest 300/NavQuest 600/NavQuest 600 Micro Doppler Velocity Log*, 2006.
- [7] Tritech International Ltd, *Micron Sonar: Product Manual*.
- [8] Tritech International Ltd, *Gemini Imaging Sonar: Product Manual*.
- [9] Sony Corporation, *SNC-WR632C Datasheet*, 2014.
- [10] Tritex ndt, «Ultrasonic Thickness Gauge: Multigauge 4000 series,» Octubre 2013. [En línea]. Available: <https://www.tritexndt.com/wp-content/uploads/2015/11/Multigauge-4000-ROV.pdf>. [Último acceso: Octubre 2018].

- [11] DeepSea, «SeaLite® Lumos,» 20 Junio 2018. [En línea]. Available: [http://www.deepsea.com/wp-content/uploads/SeaLiteLumos\\_Specifications.pdf](http://www.deepsea.com/wp-content/uploads/SeaLiteLumos_Specifications.pdf). [Último acceso: Octubre 2018].
- [12] Tecnadyne, «Model 300 DC brushless thruster,» 25 marzo 2014. [En línea]. Available: <http://tecnadyne.com/wp-content/uploads/2017/06/Model-300-Brochure-3.pdf>. [Último acceso: Octubre 2018].
- [13] Tecnadyne, «Model 560 DC brushless thruster,» 25 marzo 2014. [En línea]. Available: <http://tecnadyne.com/wp-content/uploads/2017/06/Model-560-Brochure-1.pdf>. [Último acceso: Octubre 2018].
- [14] C. Salazar-García, «Microcontroladores y protocolos de comunicación Serial. Presentación de clase Microprocesadores y microcontroladores. Área Académica de Mecatrónica. TEC,» 2016.
- [15] E. Santana, «MAVLink: protocolo de comunicación para drones,» [En línea]. Available: <https://www.xdrones.es/mavlink/>. [Último acceso: Octubre 2018].
- [16] M. Faessler, «SBUS Protocol,» 20 Marzo 2018. [En línea]. Available: [https://github.com/uzh-rpg/rpg\\_quadrotor\\_control/wiki/SBUS-Protocol](https://github.com/uzh-rpg/rpg_quadrotor_control/wiki/SBUS-Protocol). [Último acceso: Noviembre 2018].
- [17] S. Balakirsky y Z. Kootbally, «USARSIM/ROS: A combined framework for robotic control and simulation.,» junio 2012. [En línea]. Available: [https://ws680.nist.gov/publication/get\\_pdf.cfm?pub\\_id=910918](https://ws680.nist.gov/publication/get_pdf.cfm?pub_id=910918). [Último acceso: febrero 2019].
- [18] M. Quigley, B. Gerkey, K. Conley, J. Faust, T. Foote, J. Leibs, E. Berger, R. Wheeler y A. Ng, «ROS: an open-source Robot Operating System,» [En línea]. Available: <http://www.robotics.stanford.edu/~ang/papers/icraoss09-ROS.pdf>. [Último acceso: febrero 2019].
- [19] L. Llamas, «Calculadora filtro paso bajo PWM,» [En línea]. Available: <https://www.luisllamas.es/calculadora-filtro-paso-bajo-pwm/>.
- [20] F. Orjales Saavedra, Interviewee, *Información del proyecto submarino*. [Entrevista]. abril 2018.

## Apéndices

A continuación, se presentan ciertos procedimientos que se determinaron a lo largo de la realización de este proyecto, esto miras a facilitar un trabajo futuro. Cabe destacar que (en los casos que corresponda) dichos procedimientos asumen que ya se cuenta con el respectivo entorno de desarrollo instalado.

### **Apéndice 1. Procedimiento para compilar de cero (build clean) el firmware del Pixhawk.**

- Cambiar código y guardar.
- Abrir la Terminal (interfaz de línea de comandos).
- Entrar en la carpeta de ardupilot
- Ejecutar “*./waf distclean*” (borra la compilación anterior)
- Ejecutar “*mkdir -p /Users/anibalsancho/ardupilot/build/px4-v2/modules/PX4Firmware/src/firmware/nuttx/extras*” (crea esa carpeta con el fin de evitar un error de compilación)
- Ejecutar “*./waf configure --board px4-v2*” (configura la versión de la tarjeta a utilizar)
- Ejecutar “*./waf build sub*” (compila el código)

Para cargar el firmware al Pixhawk conectado por USB:

- Ejecutar “*./waf --upload sub*”

Para cargar el firmware remotamente estando conectado a la RaspberryPi (por interfaz web):

- Abrir en cualquier explorador abrir la página [192.168.3.2:2770/system](http://192.168.3.2:2770/system) (se debe estar conectado a la misma red de la RaspberryPi)
- En la sección de Pixhawk "Firmware Update" agregue el archivo de firmware "ardusub.px4" que se encuentra en la siguiente dirección: `/ardupilot/build/px4-v2/bin`

## Apéndice 2. Procedimiento para agregar nuevos mensajes de MAVLink

Para agregar nuevos mensajes de MAVLink se deben hacer cambios tanto en el firmware del Pixhawk como en el programa de control en tierra. También si se tiene conectado a través de la RaspberryPi, también se deben hacer cambios en la misma.

Procedimiento en el Pixhawk:

- Ir a la dirección: `ardupilot/modules/mavlink/message_definitions/v1.0/`
- Cambiar `common.xml` y agregar nuevos mensajes revisando que el ID del mensaje no este repetido y comprobar que se encuentra dentro de la sección en que se definen los mensajes (para ver la estructura que se debe seguir se puede tomar como referencia algún otro mensaje existente).
- Agregar dentro del código una función que envíe el mensaje, típicamente: `mavlink_msg_nombre_nuevo_mensaje_send (chan, dato1, dato2, ...)`
- Recompilar y cargar el nuevo firmware (en caso de dudas ver apéndice 1).

Procedimiento en la GCS (Ground Control Station, por ejemplo: QGroundControl, APM Planner, etc.):

- Buscar en el código fuente la carpeta `message_definitions` (típicamente la dirección relativa es: `/libs/mavlink/include/mavlink/v2.0`); en caso de no encontrarla descargar de [https://github.com/mavlink/c\\_library\\_v2.git](https://github.com/mavlink/c_library_v2.git) y agregar en `common.xml` el mismo mensaje que se agregó en el del pixhawk (CUIDADO, aunque ambos se llaman `common.xml` son distintos).
- Regenerar los headers (las bibliotecas) utilizando la aplicación `mavgenerate.py` que se encuentra en la siguiente dirección `/ardupilot/modules/mavlink/` en la cual pone el archivo `ardupilotmega.xml` (este a su vez incluye el `common.xml` que se modificó, con lo cual deben de estar en la misma carpeta).
- Reemplazar los headers de la carpeta `/libs/mavlink/include/mavlink/v2.0` (dirección relativa) con los generados en el punto anterior (reemplazar únicamente las carpetas/archivos que son iguales).

- Recompilar utilizando la aplicación Qt Creator y seguir utilizando la nueva aplicación generada.

Procedimiento en la RaspberryPi:

- Abrir el archivo `common.xml` en la carpeta `/companion/submodules/mavlink/message_definitions/v1.0`; agregar el nuevo mensaje con el mismo formato e ID que en el Pixhawk y la GCS.
- Reinstalar el pymavlink, ejecutando el `setup.py` (`python setup.py install`) de la carpeta: `/companion/submodules/mavlink/pymavlink`.
- Reiniciar la RaspberryPi (`sudo reboot`).

Notas:

- Para editar archivos desde línea de comandos se puede utilizar "nano".
- Siempre verificar que se haya completado la reinstalación de pymavlink, ya que si se termina antes puede ser un indicativo de que el ID de mensaje que se está utilizando se encuentra repetido.



### Apéndice 3. Procedimiento para cambiar la red en la que se conectan la RaspberryPi-GCS

- En la carpeta principal (/home/pi) cambiar en los archivos "gstreamer.param" y "gstreamer2.param": “!~~udpsink~~ ~~host=192.168.2.1~~ ~~port=5600~~” por “! *udpsink* *host=192.168.xxx.xxx* *port=5600*” donde las "x" representan la nueva red que se desea configurar.
- En la carpeta principal cambiar en el archivo "mavproxy.param": “~~--out~~ ~~udpbcast:192.168.2.255:14550~~” por “--out *udpbcast:192.168.xxx.255:14550*” donde las "x" representan la nueva red que se desea configurar, cabe destacar que en esta sección se está configurando el que sería el broadcast de la red.
- En el archivo /boot/cmdline.txt cambiar: “~~dwc\_otg.lpm\_enable=0~~ ~~console=serial0,115200~~ ~~console=tty1~~ ~~root=PARTUUID=1b4801d2-02~~ ~~rootfstype=ext4~~ ~~elevator=deadline~~ ~~fsck.repair=yes~~ ~~rootwait~~ ~~ip=192.168.2.2~~” por “*dwc\_otg.lpm\_enable=0* *console=serial0,115200* *console=tty1* *root=PARTUUID=1b4801d2-02* *rootfstype=ext4* *elevator=deadline* *fsck.repair=yes* *rootwait* *ip=192.168.xxx.xxx*” donde las "x" representan la nueva IP fija que se le desea asignar a la RaspberryPi.

Notas:

Para el submarino en cuestión quedaron configuradas las siguientes direcciones IP:

- Red: 192.168.3.1
- RaspberryPi: 192.168.3.2
- PC superficie: 192.168.3.10

Nueva dirección de interfaz web de la RaspberryPi (para actualizar firmware Pixhawk):

- <http://192.168.3.2:2770/system>