

---

# Diseño y Construcción de una Computadora de Vuelo para Vehículos Autónomos con Tolerancia a Fallas

Federico Ignacio Nuñez Frau - 98211

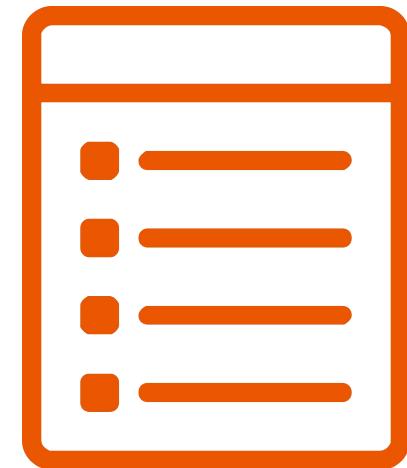
Director: Dr. Ing. Claudio Pose (FIUBA)  
Co-Director: Ing. Leonardo Garberoglio (UTN-FRSN)

06/09/2024

---

# Contenidos

- 1. Introducción y Motivación**
- 2. Diseño y Construcción de la Computadora de Vuelo**
- 3. Sistemas Tolerantes a Fallas**
- 4. Sistema Implementado**
- 5. Resultados**
- 6. Conclusiones y Perspectivas**

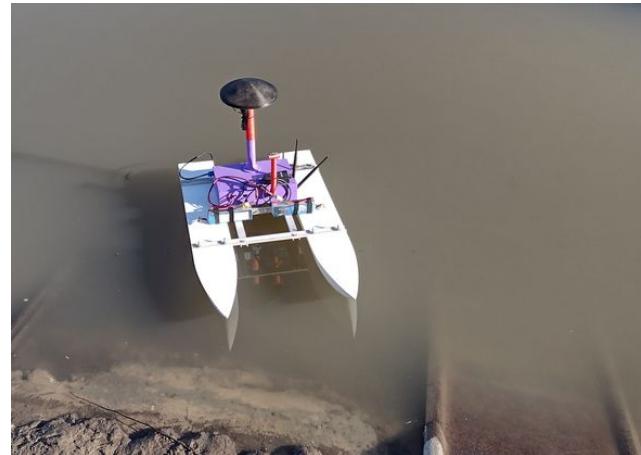


---

# 1. Introducción y Motivación

# Vehículos Autónomos

- Los vehículos autónomos no cuentan con una tripulación ni un piloto a bordo.
- Tienen la capacidad de desplazarse por sí solos.
- Gran variedad de sensores + **computadora central** = sistema de navegación y reconocimiento del entorno.
- También pueden ser comandados de forma remota (**vehículos teleoperados**).



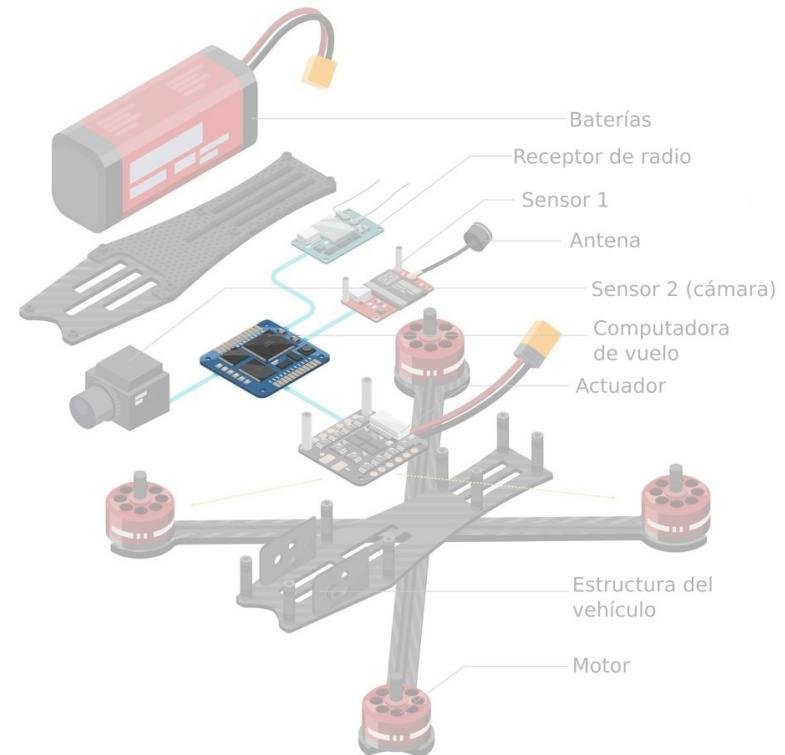
---

# Vehículos Autónomos



# Computadora de Vuelo

- Se componen de varios elementos.
- Todos ellos son susceptibles de manifestar fallas.
- En este trabajo se abordan aspectos relacionados a la computadora de vuelo.



# Tolerancia a Fallas

- Cada vez tienen mayor presencia en **zonas civiles**.
- Teniendo esto en cuenta, la **confiabilidad** es un aspecto que toma mayor relevancia.
- El aspecto más importante en aviones comerciales:  $10^{-9}/\text{h}$  de vuelo.
- Aviones militares:  $10^{-7}/\text{h}$  de vuelo.



- En vehículos aéreos como aviones comerciales y militares, se utilizan **redundancias**.
- También en drones de uso militar.
- Drones militares:  $10^{-5}/\text{h}$  de vuelo.
- ¿Drones civiles/comerciales?

---

# Objetivos

- 1) **Diseñar y construir** una computadora de vuelo para vehículos aéreos no tripulados con la capacidad de tolerar fallas a partir de redundancias.
- 2) **Estudiar** las características de sistemas con tolerancia a fallas aplicados en vehículos aéreos, tanto tripulados como no tripulados.
- 3) **Entender** los requerimientos para implementar un sistema con redundancias.
- 4) **Desarrollar** un firmware que demuestre la capacidad de ser utilizada en un sistema con estas características.

---

## 2. Diseño y Construcción de la Computadora de Vuelo

# Antecedentes

- En el Laboratorio de Automática y Robótica (LAR) se han desarrollado otras computadoras de vuelo anteriormente.

## Primera Versión

- LPC-1769, Cortex M3, 120 MHz, sin unidad de punto flotante.
- Sensores IMU, Barómetro y Magnetómetro fuera de fabricación.
- Dimensiones: 93x93 mm.



V1: 2013

## Segunda Versión

- Mismas prestaciones que la versión 1.
- Se redujeron las dimensiones a 100x60mm.
- Se removió el magnetómetro.



V2: 2015

## Tercera Versión

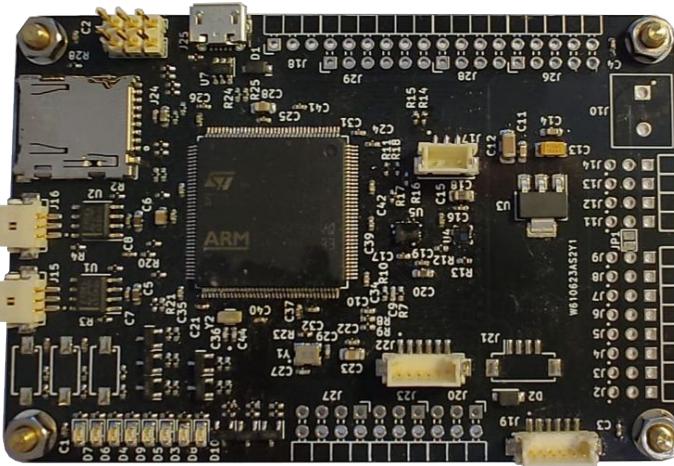
- STM32F722, Cortex M7, 216 MHz, con unidad de punto flotante.
- Se actualizaron los sensores.
- Se removió la fuente comutada.
- Conectores más pequeños.
- Dimensiones: 80x60 mm.
- USB 2.0, memoria uSD.



V3: 2018

V4: 2024

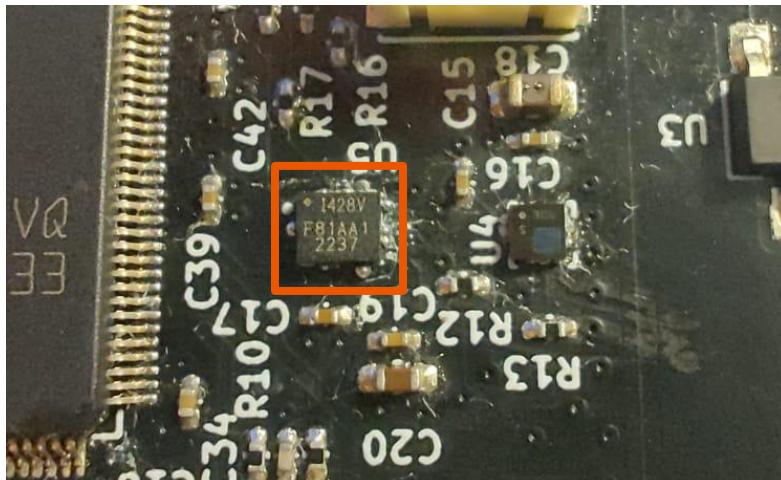
# Microcontrolador



- Microcontrolador seleccionado: STM32F746ZG.
- Mayor cantidad de puertos exteriores UART, SPI e I2C.
- Interfaz CAN x 2 → utilizada para la tolerancia a fallas y redundancias.
- Longevidad de 10 años, hasta enero 2034.
- Es reemplazable por otros de mejores prestaciones, por ejemplo STM32F777ZIT6.

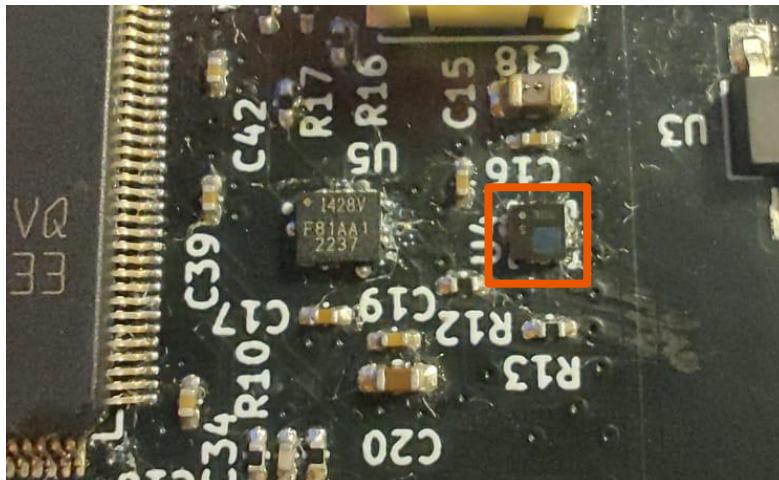
V4: 2024

# Unidad de Medición Inercial (IMU)



- Parámetros de referencia para la selección:
  - Bias vs time (acelerómetros y giróscopos).
  - Noise (giróscopos).
  - Scale factor error (giróscopos).
  - Longevidad.
- Sensor seleccionado: ICM42688p de TDK.
- Menor ruido y error de factor de escala para giróscopos.
- Permite la reutilización de parte del firmware de la versión anterior.

# Barómetro



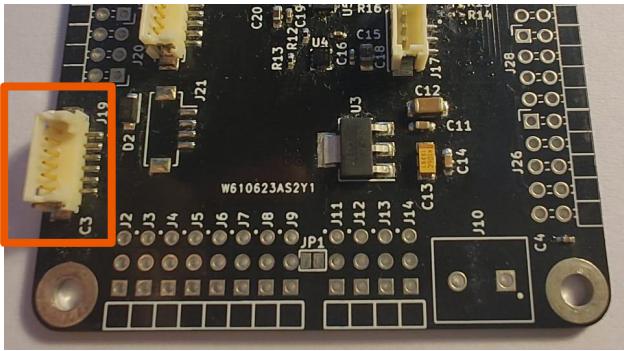
- Se utiliza para medir diferencias de presión atmosférica, lo que permite medir diferencias de altura.
  - Se evaluaron 2 alternativas con prestaciones similares: **ICP-20100** e **ILPS22QSTR**.
  - **Sensor seleccionado:** ILPS22QSTR de ST.
  - Se optó por este último por asegurar una **longevidad de 10 años**, hasta enero 2033.
  - Comunicación: **I2C**

# Magnetómetro

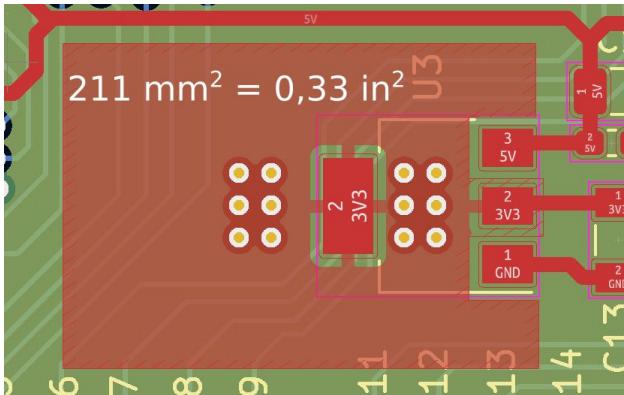


- Se utiliza para medir la dirección y el módulo del campo magnético terrestre:  $H_x$ ,  $H_y$ ,  $H_z$ .
  - Se evaluaron 2 alternativas que superan a las demás: **RM3100** y **MMC5983MA**.
  - **Sensor seleccionado:** MMC5983MA de MEMSIC.
  - Grado Automotriz, AEC-Q100.
  - Solo superado en ruido por **RM3100**.
  - Se descartó por ser mucho más caro y de difícil acceso.

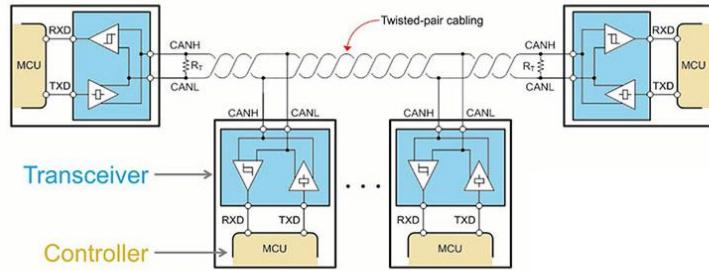
# Alimentación



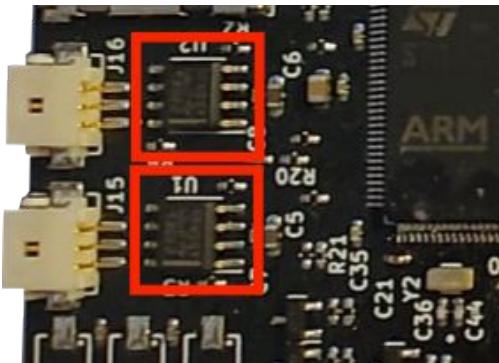
- Regulador lineal: ZLDO1117QG33TA
  - 3,3 V
  - 1 A
  - AEC-Q100.
- Conector estándar para fuentes comerciales.
- Posibilidad de alimentación por USB.
- Consideraciones de disipación: vías y planos en PCB.



# Interfaz de Comunicación CAN



- Bus de comunicaciones utilizado para la tolerancia a fallas.
- CAN: Controller Area Network, desarrollado por Bosch.
- Velocidad máxima de **1 Mbps** (versión High-Speed).
- Hasta **40 m** de longitud y **30 nodos**.



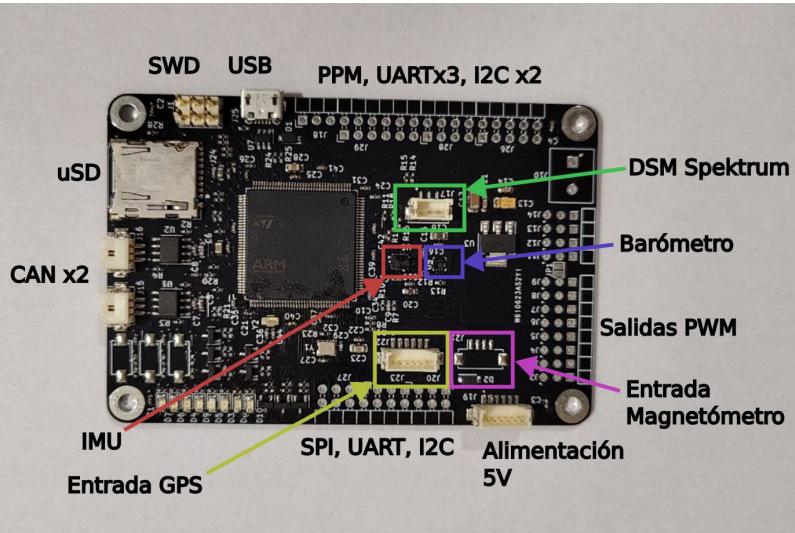
- Transceiver: SN65HVD230 de Texas Instruments.
  - Trabaja con tensiones de 3,3 V → facilidad de uso.
  - Compatible con capa física ISO 11898-2 (High-Speed).

# Interfaz de Comunicación CAN

- CAN también tiene presencia en drones y vehículos autónomos:

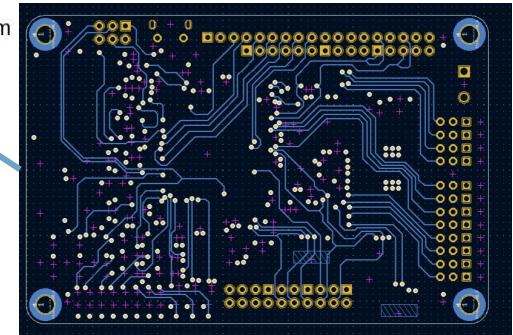
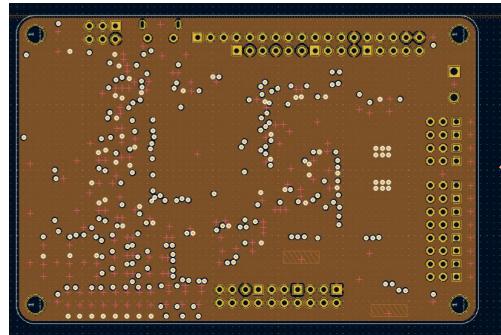
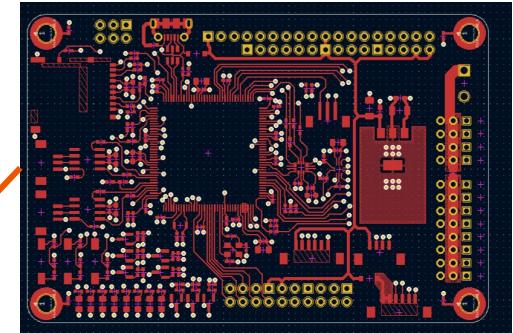
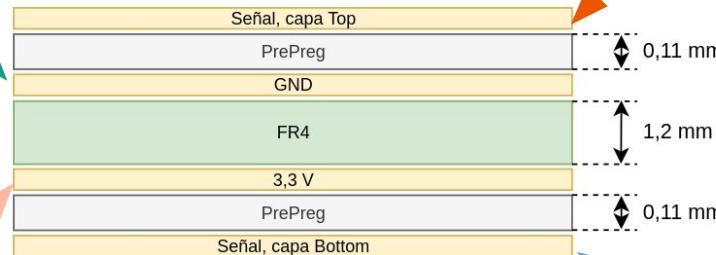
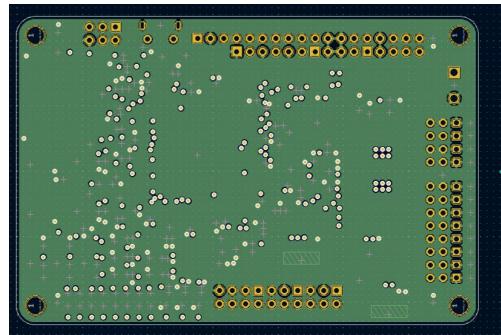
Protocolos de Comunicación de capas superiores	Módulos, actuadores y sensores	Computadoras de vuelo comerciales:
 <b>DroneCAN</b> 84 followers  <a href="http://www.dronecan.org">http://www.dronecan.org</a>	 	

# Otros



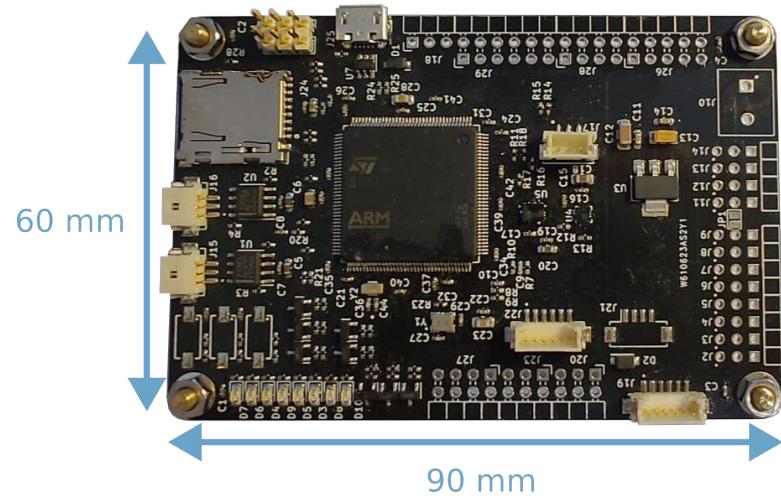
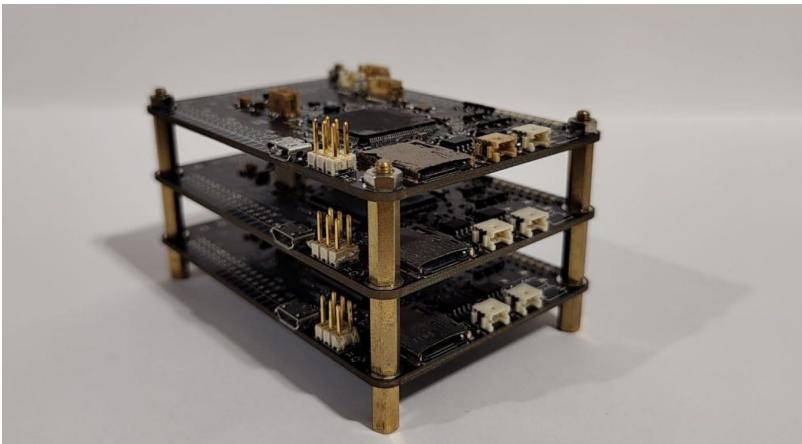
- Conector para receptor **GPS externo**, UART + señal PPS.
- **12 Salidas PWM** para control de motores.
- Entradas de **radio control**: PPM y Spektrum.
- Conector para **programación y debugging**: SWD.
- 8 LEDs indicadores + 2 pulsadores de propósito general.
- Slot para memoria **micro SD**.
- **USB 2.0, full speed**.
- **Conectores por duplicado**:
  - DF-13
  - Tira de pines de 0.1”.

# Características del PCB



# Resumen

- Se fabricaron 3 placas.
- Orificios de montaje métrica 3 en las esquinas.



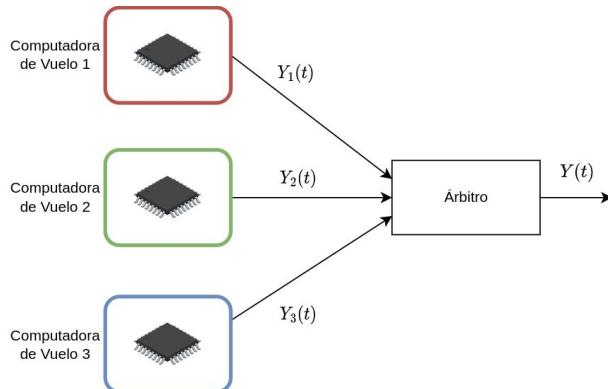
---

## 3. Sistemas Tolerantes a Fallas

# Tolerancia a Fallas

Se acepta que las fallas pueden ocurrir y se incluyen mecanismos para que, a pesar de su ocurrencia, el sistema pueda seguir cumpliendo su función.

- Existen algunas computadoras de vuelo que ofrecen la posibilidad de trabajar con **redundancias**.
- Se comparan resultados de cada réplica para detectar fallas.

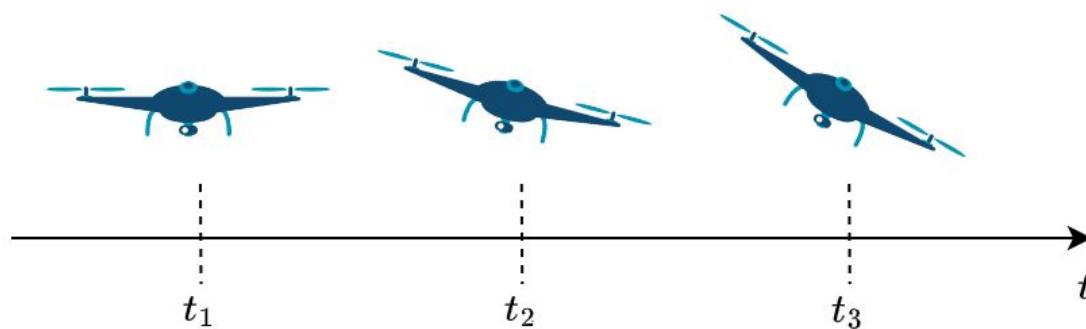


**UAV Navigation**  
grupo osia



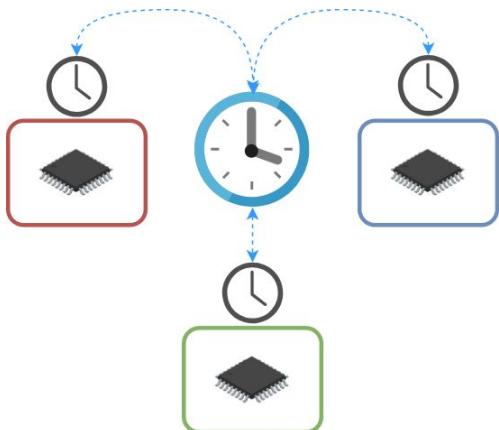
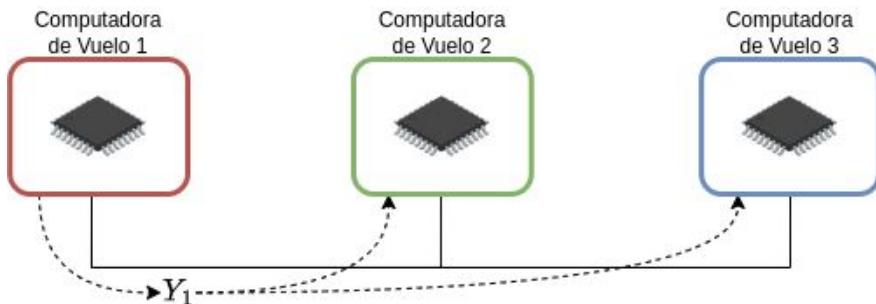
# Problema del Sincronismo

- Periódicamente debe obtenerse una estimación de la pose del vehículo, y de la señal a aplicar sobre los motores.
- Se hacen comparaciones de resultados para detectar fallas.
- Para que las comparaciones tengan sentido, estas **deben hacerse sobre resultados que se correspondan temporalmente**.
- Esto se logra a través de una **sincronización** entre las réplicas.



# Bus de Comunicaciones + Sincronización

- Cada mensaje es recibido por todas las réplicas → **Consenso**.
- Las **colisiones** pueden perjudicar el determinismo.
- Se aprovecha la **sincronización** para ordenar el uso del bus, acceso al medio TDMA.



---

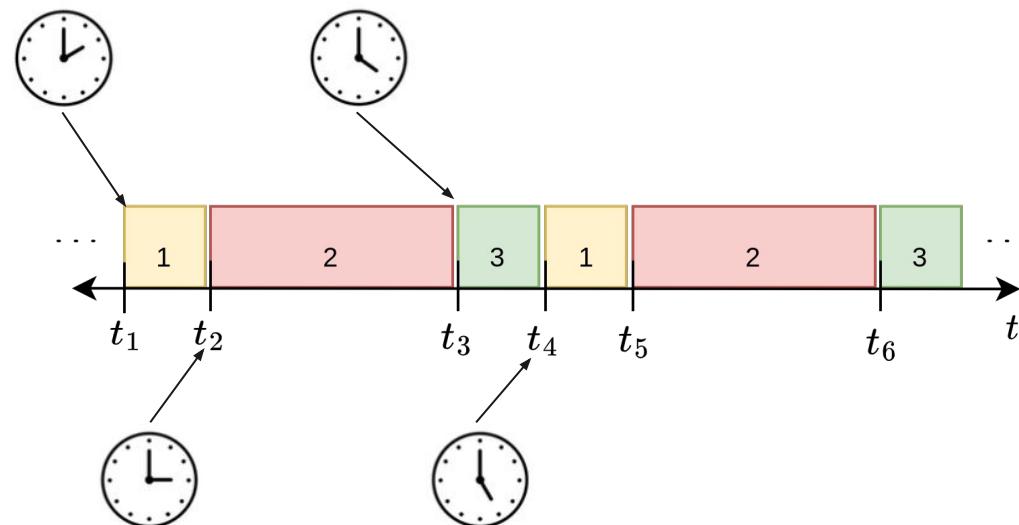
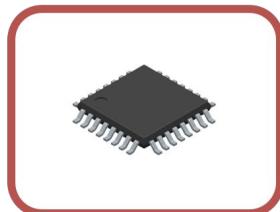
## 4. Sistema Implementado

# Time-Triggered Systems

- El instante de tiempo en el que se ejecuta cada tarea se vuelve parte del diseño del sistema.
- Lo mismo ocurre con el orden y la secuencia de ejecución.

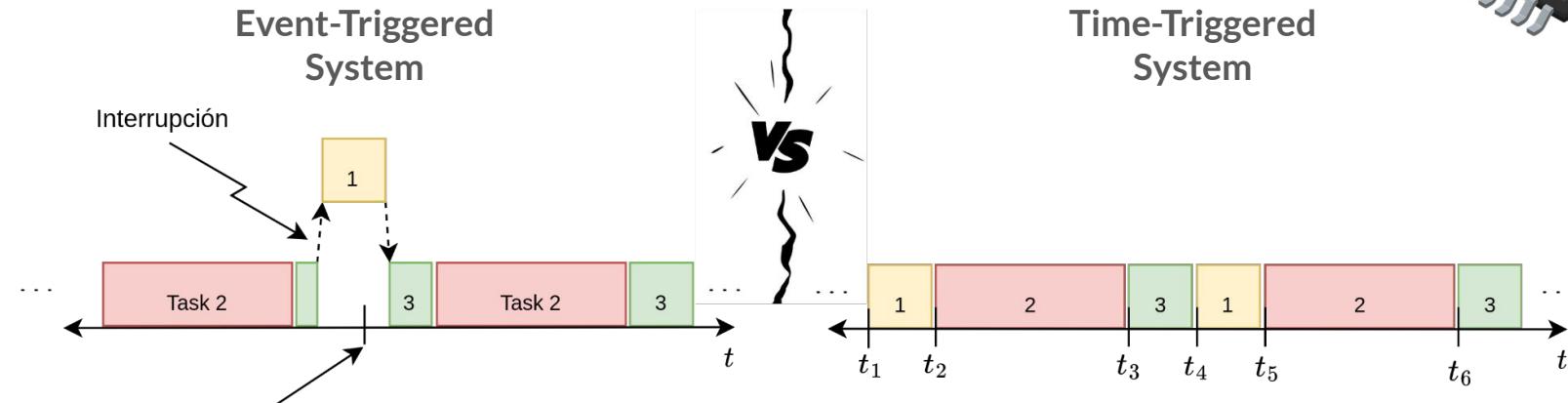
Si tenemos un conjunto de tareas:

- Tarea 1
- Tarea 2
- Tarea 3



# Time-Triggered Systems

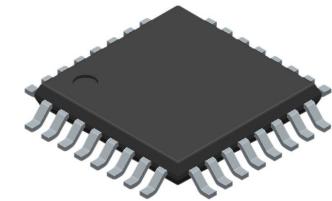
- En contraposición se encuentran los Event-Triggered Systems:



Deadline para aplicar control a motores

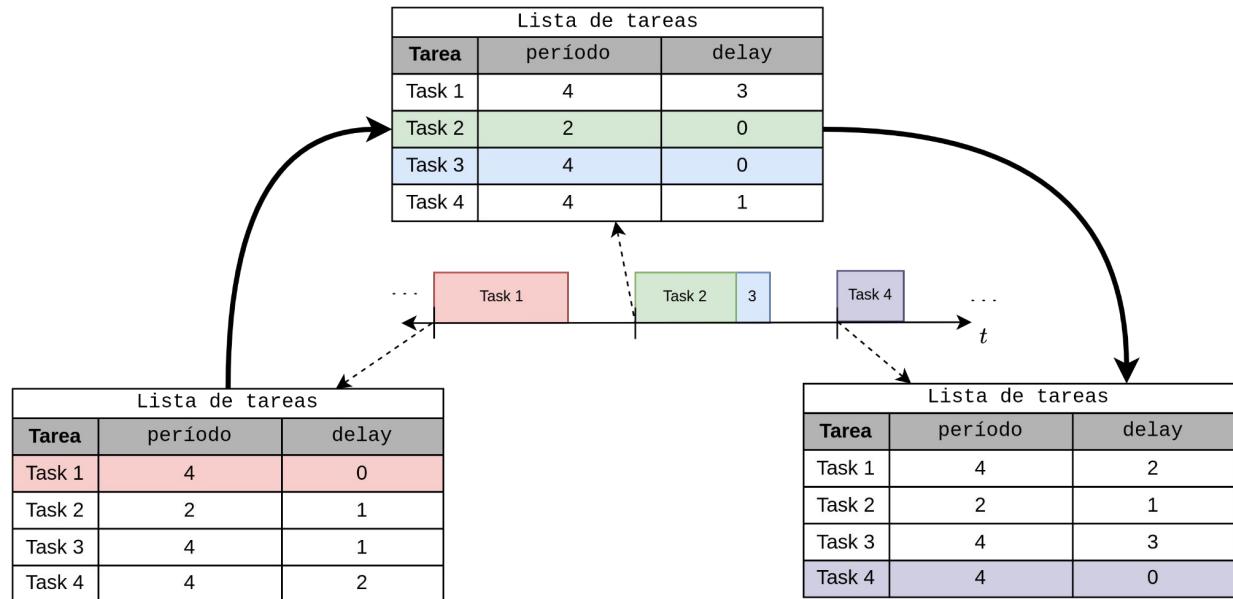
Retraso debido a la interrupción.

- La tarea 1 ha sido rediseñada, teniendo en cuenta sus tiempos de ejecución.
- El comportamiento del sistema se vuelve predecible.



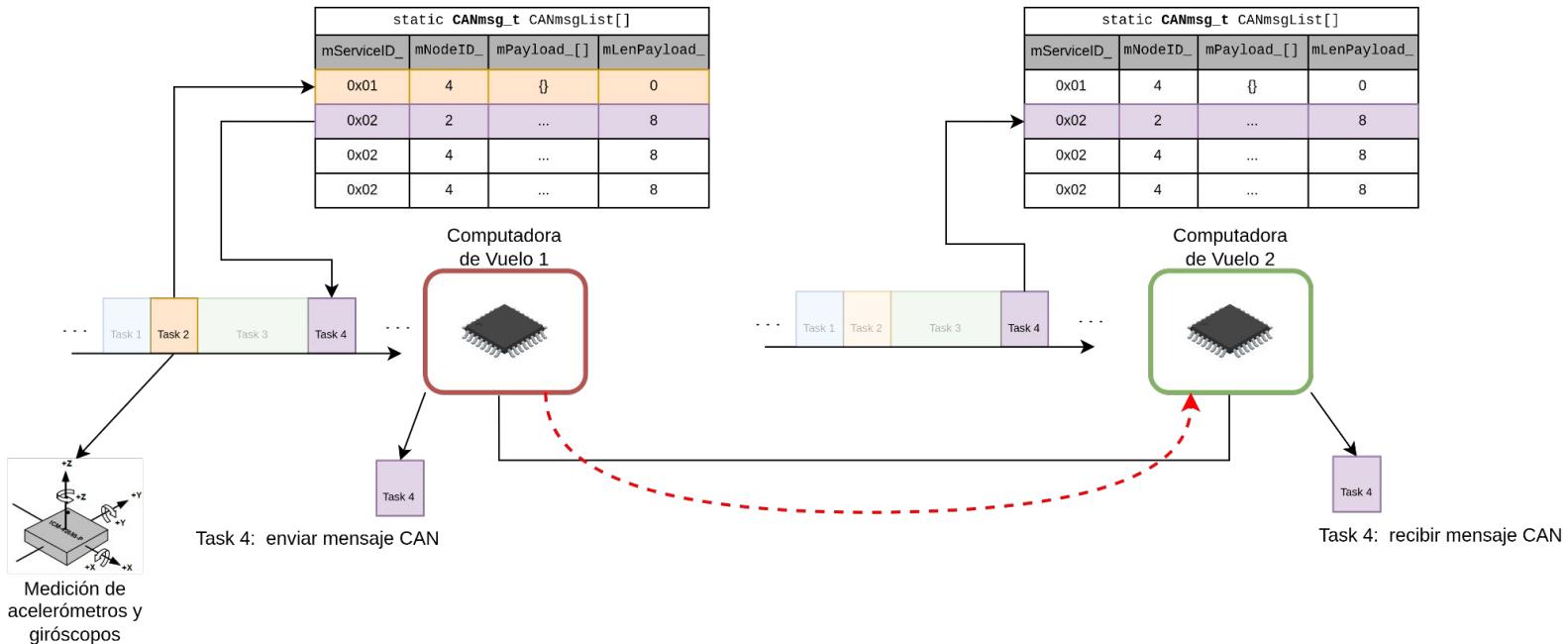
# Scheduler

- 1 timer, 1 lista de tareas y 1 función que selecciona la siguiente tarea a ejecutar.
- En cada **interrupción** del timer se ejecuta la siguiente tarea.
- El scheduler se ejecuta de forma periódica, pero no tiene por qué ser así.



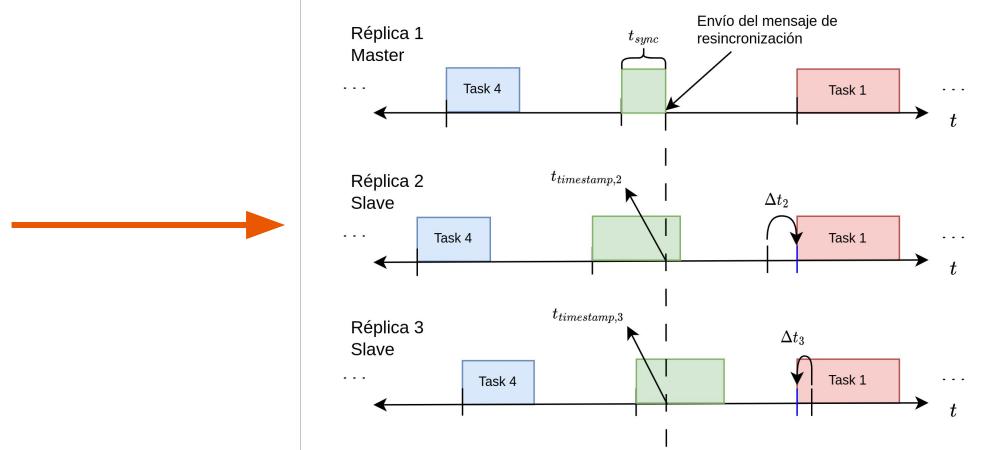
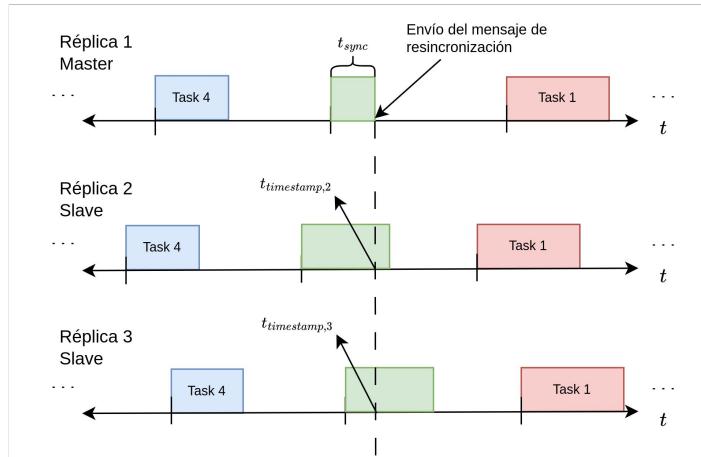
# Comunicación a Través del Bus

- Comunicación entre réplicas y entre tareas:



# Implementación de la Sincronización

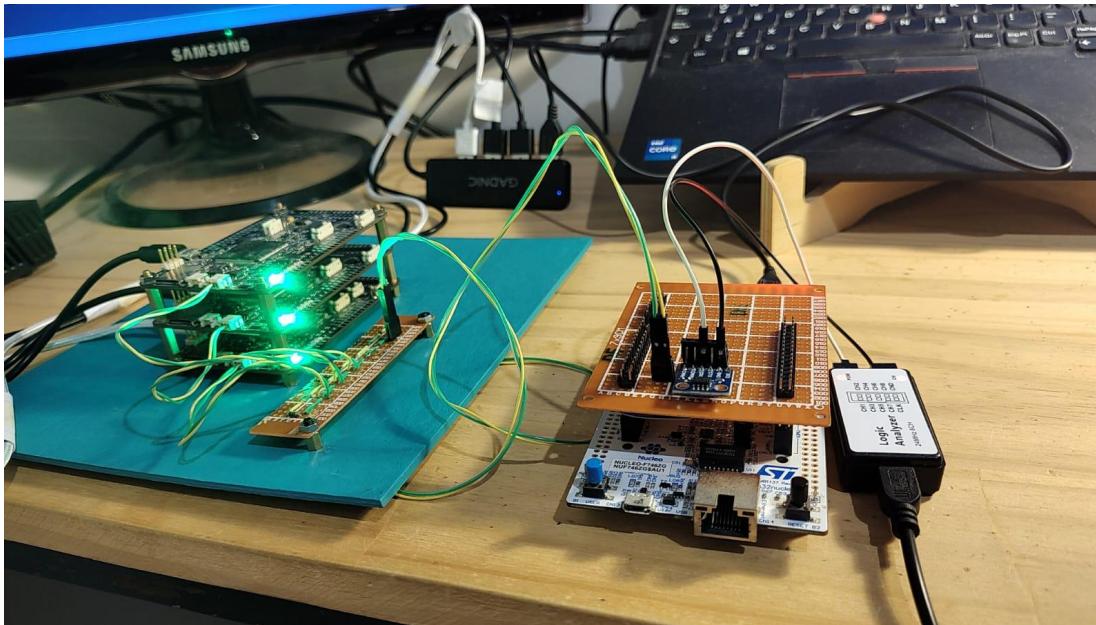
- También se resuelve con una tarea.
- Se utilizó un algoritmo master-slave, pero lo correcto sería usar un algoritmo distribuido.
- Se repite de forma periódica.



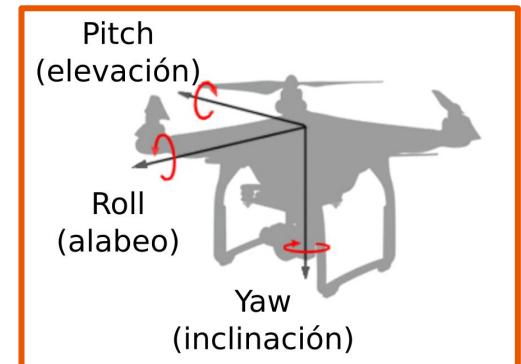
---

# 5. Resultados

# Estimación de la Orientación



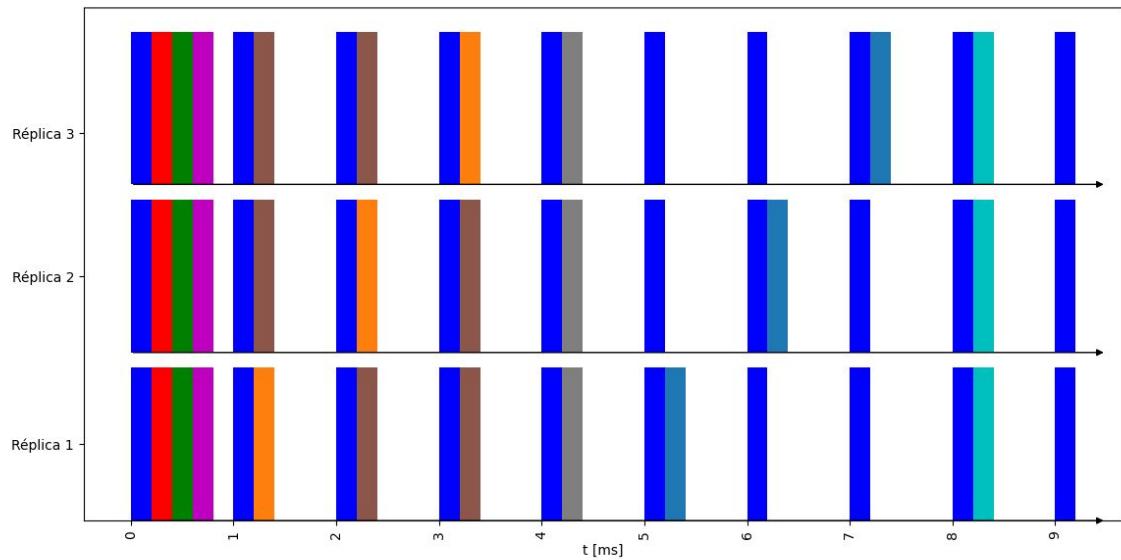
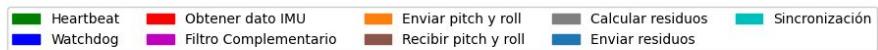
- Utilizando la IMU, se estiman los ángulos de pitch y de roll.
- Se emulan fallas en el sensor de 1 de las 3 réplicas.
- Se calculan las diferencias entre cada par de réplicas: **residuos**.



# Estimación de la Orientación

- Lista de tareas, réplica 1:

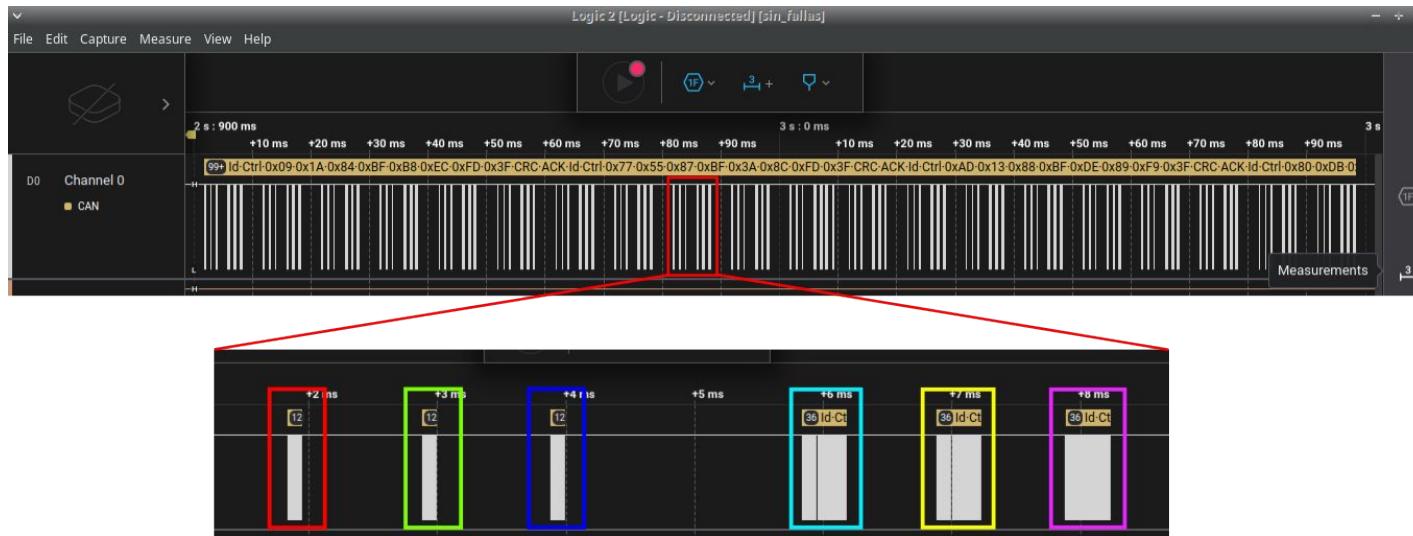
1 Tick = 1 ms		
Tarea	Período [Ticks]	Delay [Ticks]
Heartbeat	1000	0
Watchdog	1	0
Obtener dato IMU	10	0
Filtro Complementario	10	0
Enviar pitch y roll	10	1
Recibir pitch y roll de 2	10	2
Recibir pitch y roll de 3	10	3
Calcular residuos	10	4
Enviar residuos	10	5
Sincronización	1000	8



- Se calcula una nueva orientación cada 10 ms.

# Resultados: Funcionamiento Sincronizado

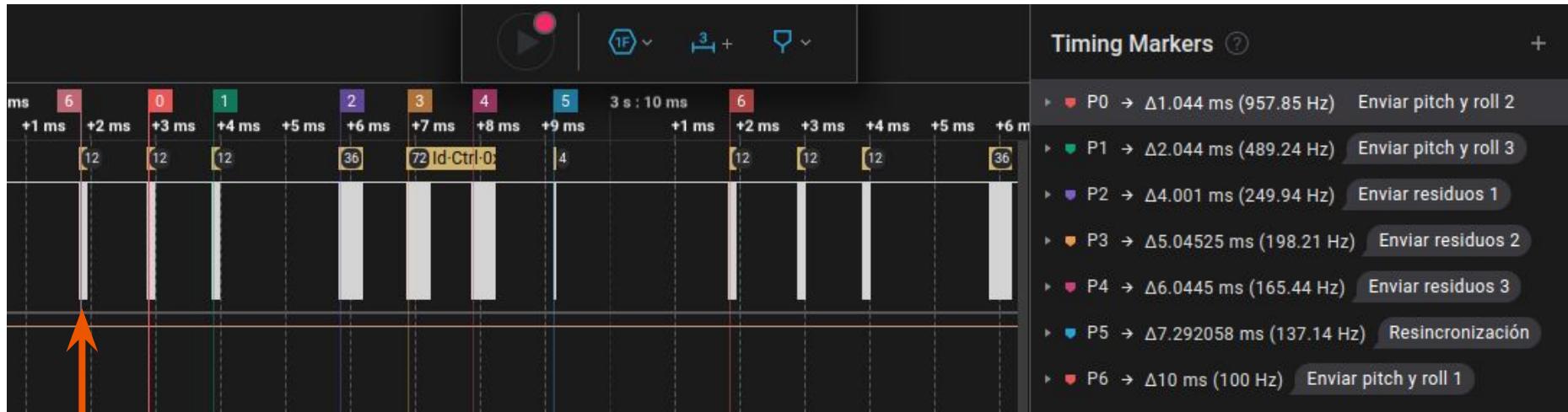
- Se muestran 100 ms con un comportamiento periódico cada 10 ms.



Enviar pitch y roll 1  
Enviar pitch y roll 2  
Enviar pitch y roll 3  
Enviar residuos 1  
Enviar residuos 2  
Enviar residuos 3

# Resultados: Funcionamiento Sincronizado

- Se muestran 100 ms con un comportamiento periódico cada 10 ms.

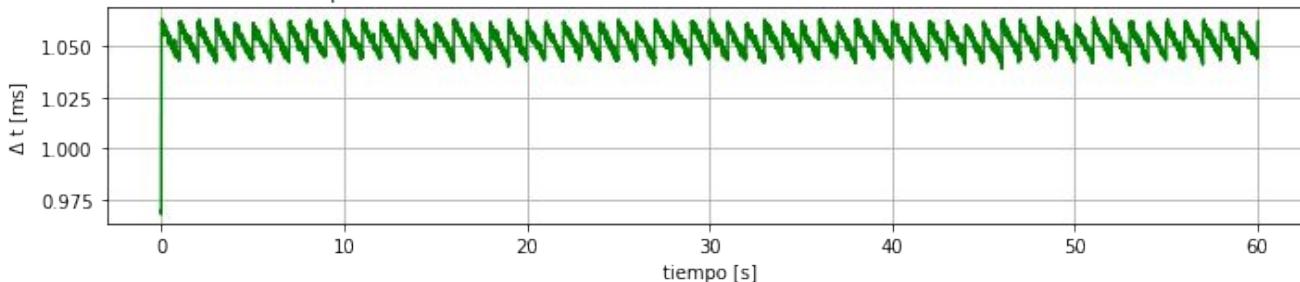


Referencia!

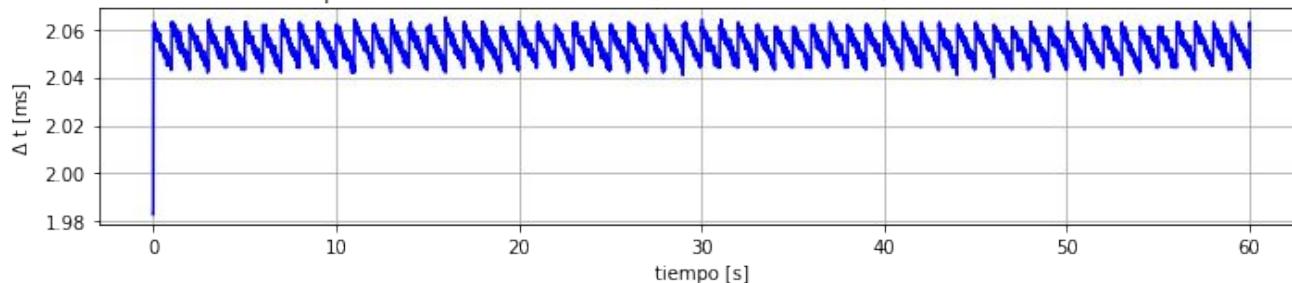
Todas las mediciones son con respecto al mensaje de referencia...

# Resultados: Funcionamiento Sincronizado

Réplica 2: media = 1.05269 ms ; min = 1.03900 ms ; max = 1.06400 ms



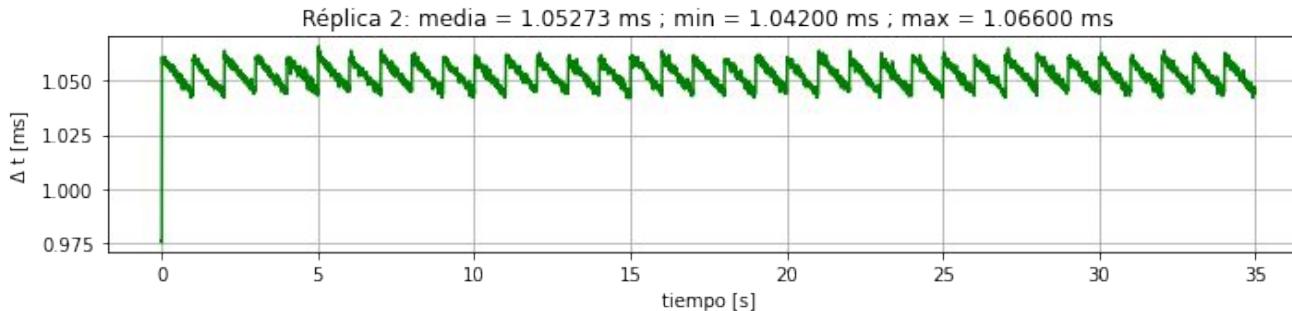
Réplica 3: media = 2.05297 ms ; min = 2.04000 ms ; max = 2.06500 ms



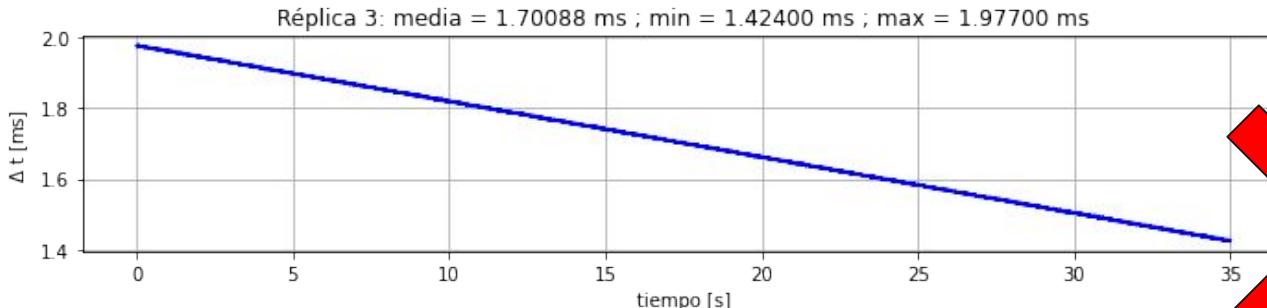
- Se verifica la sincronización durante toda la ejecución.
- Salto inicial, comienza la sincronización.

# Resultados: Funcionamiento Sincronizado

- ¿Qué pasaría si no se ejecutara una sincronización periódica?



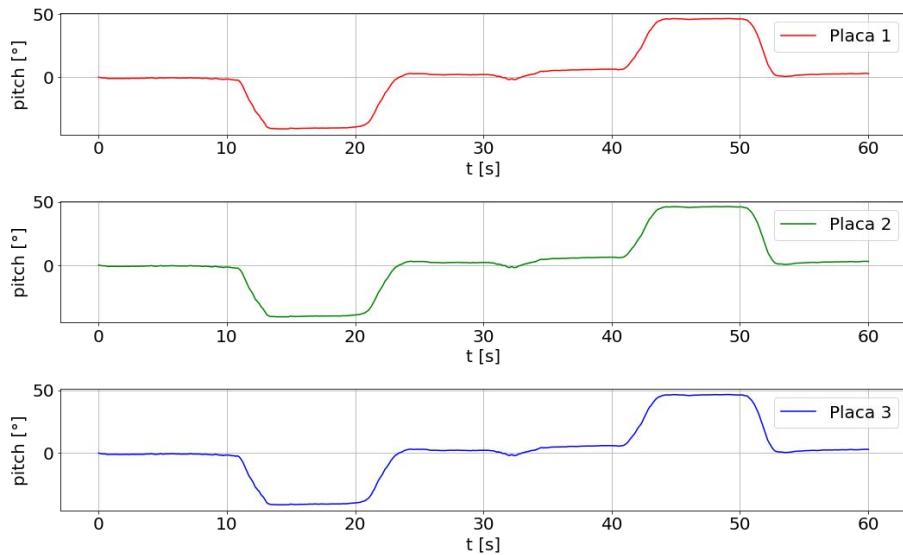
- Se eliminó la tarea de sincronización de la réplica 3.
- Se observa un **desfasaje lineal** de la réplica 3.



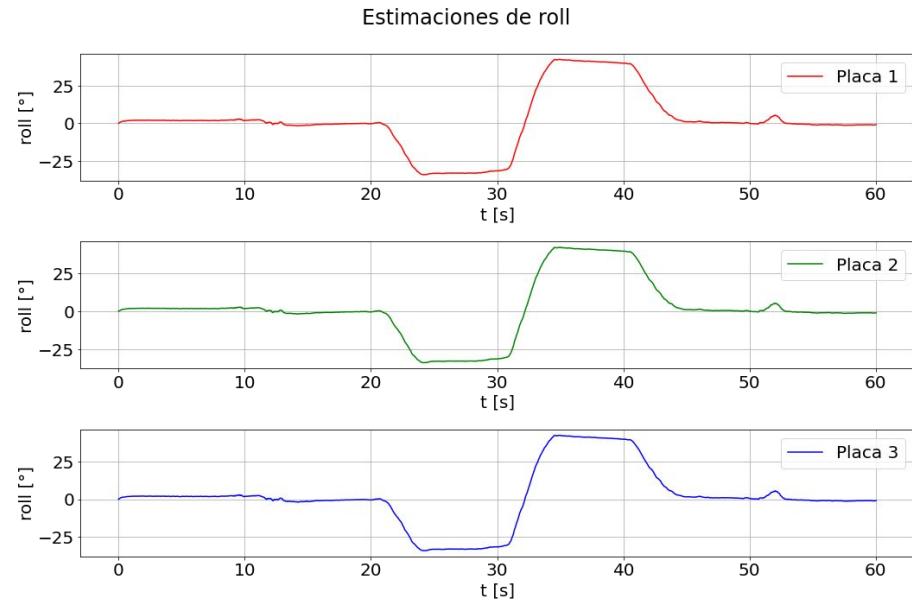
# Resultados: Pitch y Roll

- Resultados de las 3 réplicas:

Estimaciones de pitch



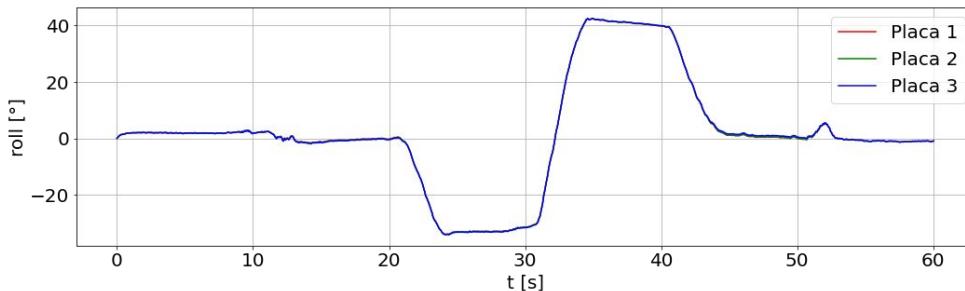
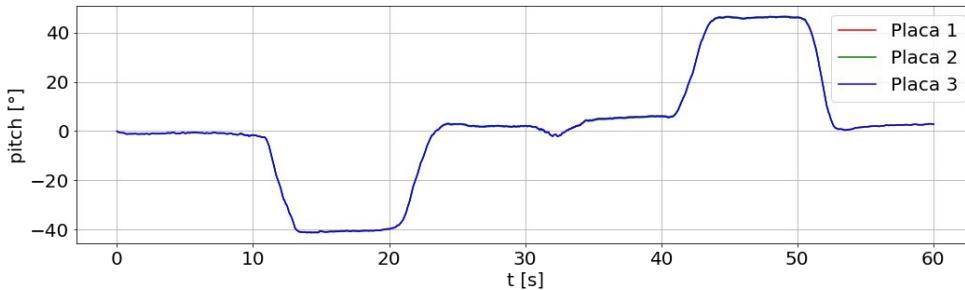
Estimaciones de roll



# Resultados: Pitch y Roll

- Resultados de las 3 réplicas superpuestos:

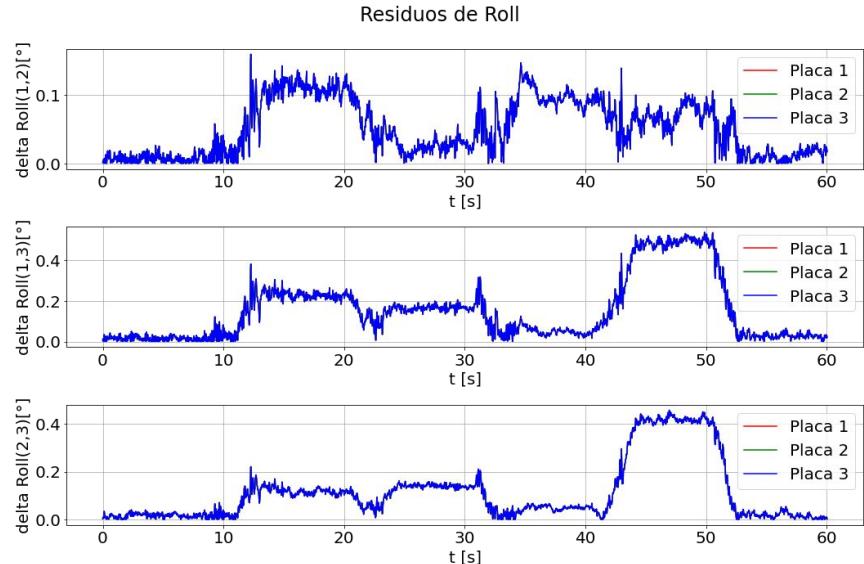
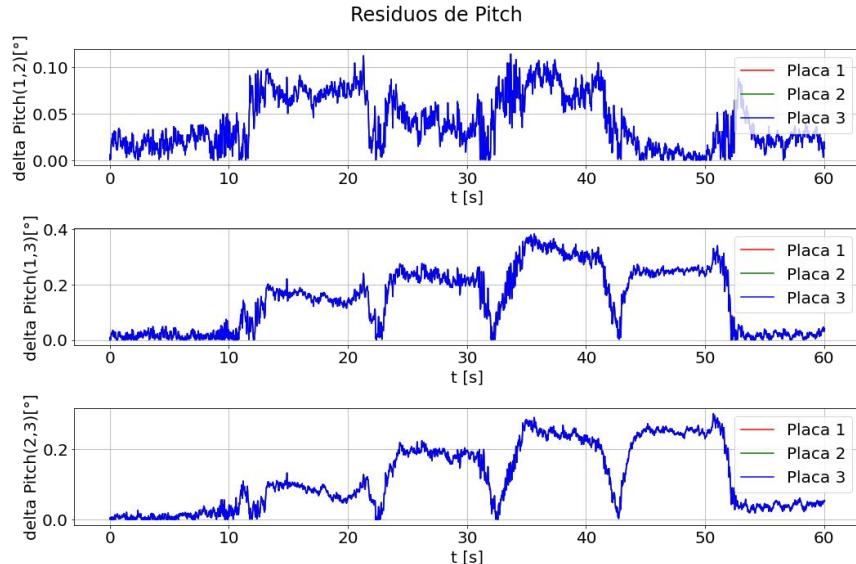
Estimaciones de pitch y roll



- Las curvas coinciden.
- Sin embargo, existen unas leves diferencias.
- Esto es esperable, teniendo en cuenta que los sensores miden magnitudes físicas.
- **Esto se ve reflejado en los residuos.**

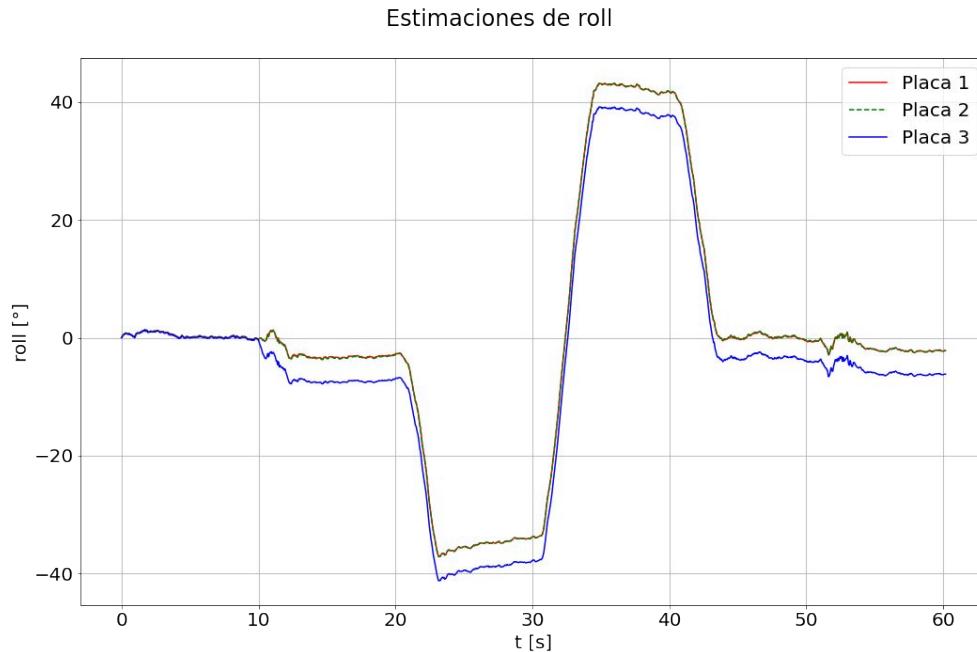
# Resultados: Pitch y Roll

- Residuos calculados por las 3 réplicas superpuestos:



- Las diferencias son inferiores a 1° tanto para pitch como el roll.

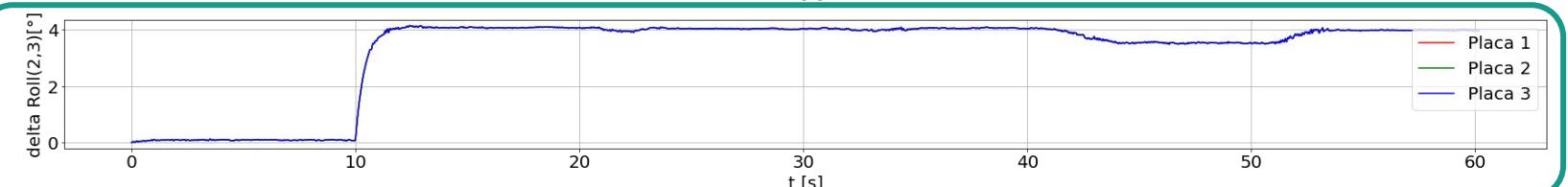
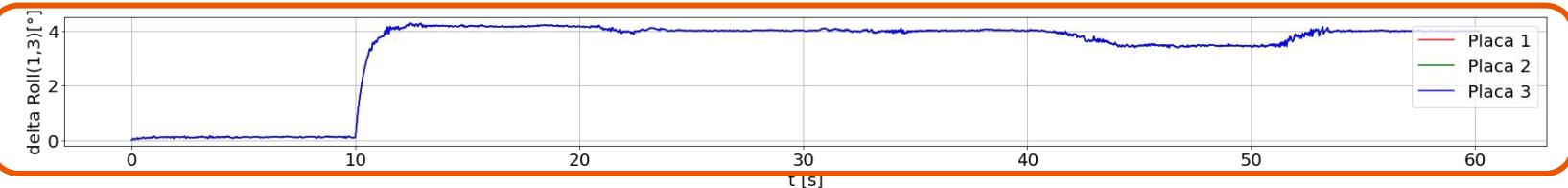
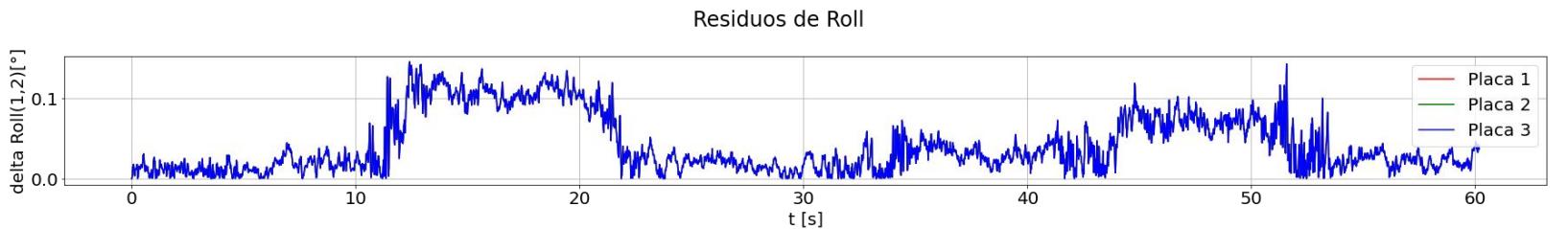
# Sesgo en Giróscopo



- Se suma un sesgo a la medición de giróscopos de la placa 3.
- El sego  $b_x = +10^\circ/\text{s}$ .
- El sesgo se aplica a las mediciones del eje x:  $\omega_x + b_x$ .
- El sesgo se aplica luego de 10 segundos de ejecución

# Sesgo en Giróscopo

- Residuos calculados por las 3 réplicas superpuestos:

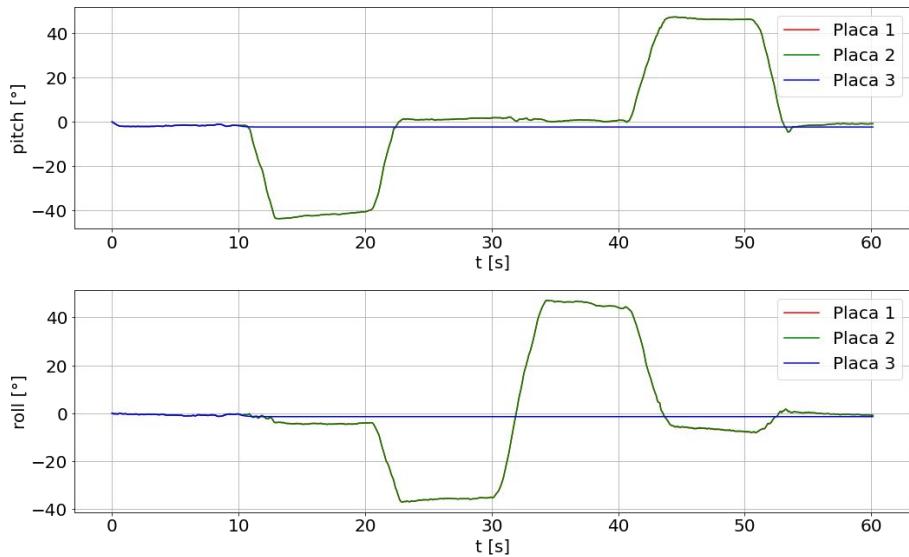


Diferencias entre 1 y 3

Diferencias entre 2 y 3

# Mediciones Invariantes

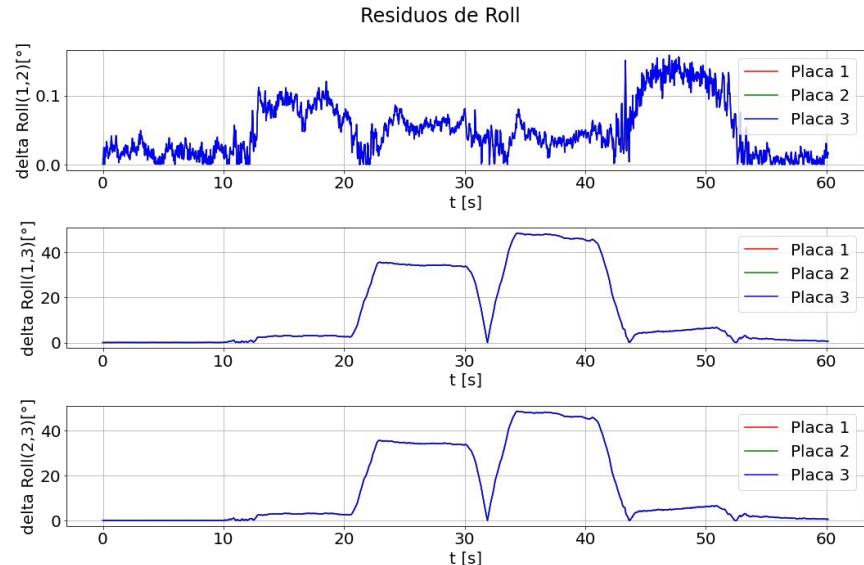
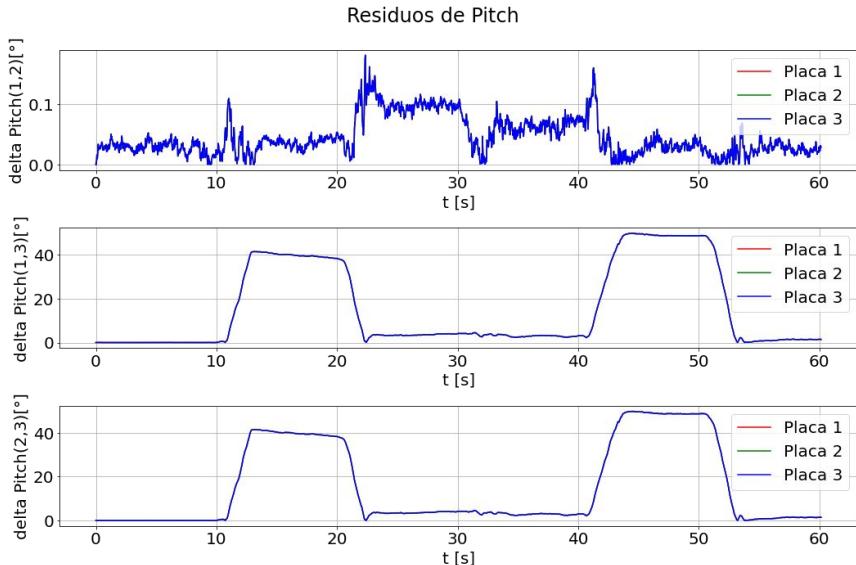
Estimaciones de pitch y roll



- A partir de los 10 segundos, las mediciones de la IMU en la réplica 3 se mantienen constantes.
- En consecuencia, las estimaciones de los ángulos de la réplica 3, también.
- Se muestran las estimaciones de pitch y roll superpuestas.

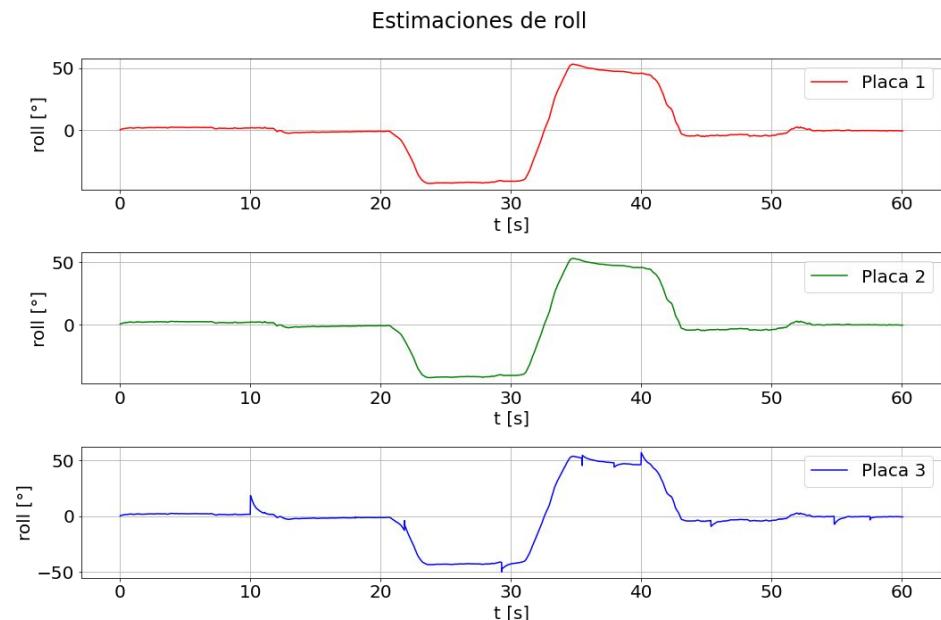
# Mediciones Invariantes

- Residuos calculados por las 3 réplicas superpuestos:



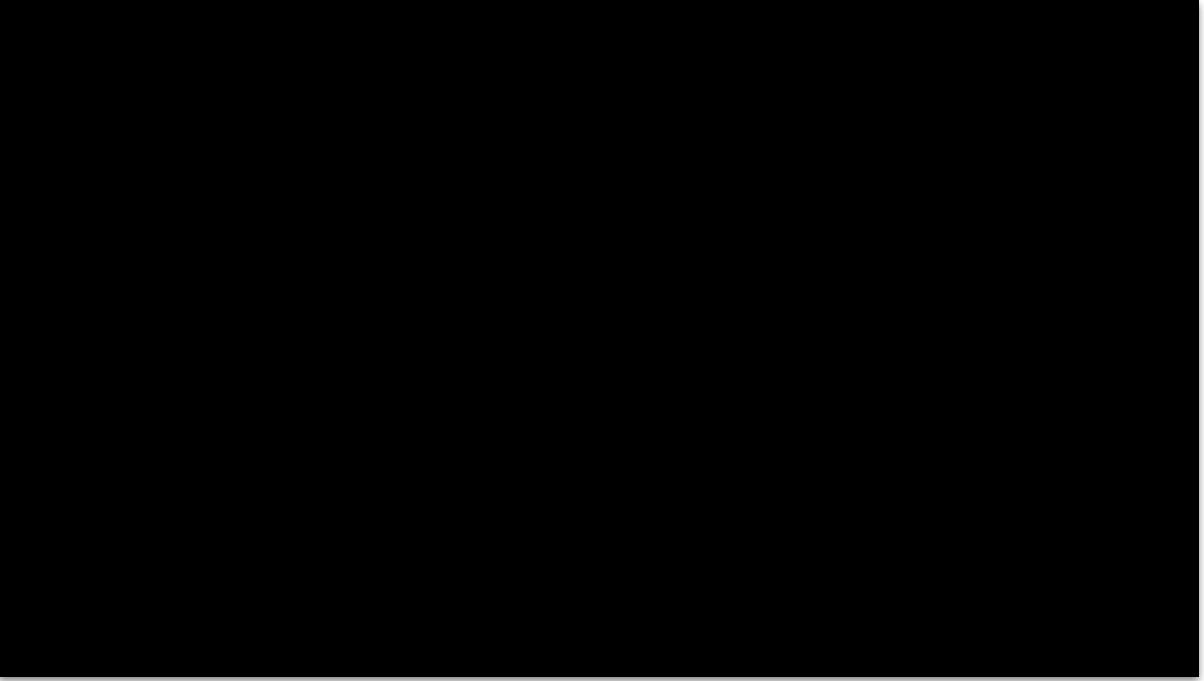
# Mediciones Espúreas en Valores de Giróscopo

- Se suma un sesgo a la medición de giróscopos de la placa 3.
- El sesgo se aplica a las mediciones del eje x:  $\omega_x + b_x$ .
- En este caso, el sesgo se aplica e inmediatamente se remueve, generando un efecto de saltos.
- En cada caso, los valor de amplitud, así como el instante de tiempo en que se aplica el sesgo, se calculan de manera aleatoria.



# Mediciones Espúreas en Valores de Giróscopo

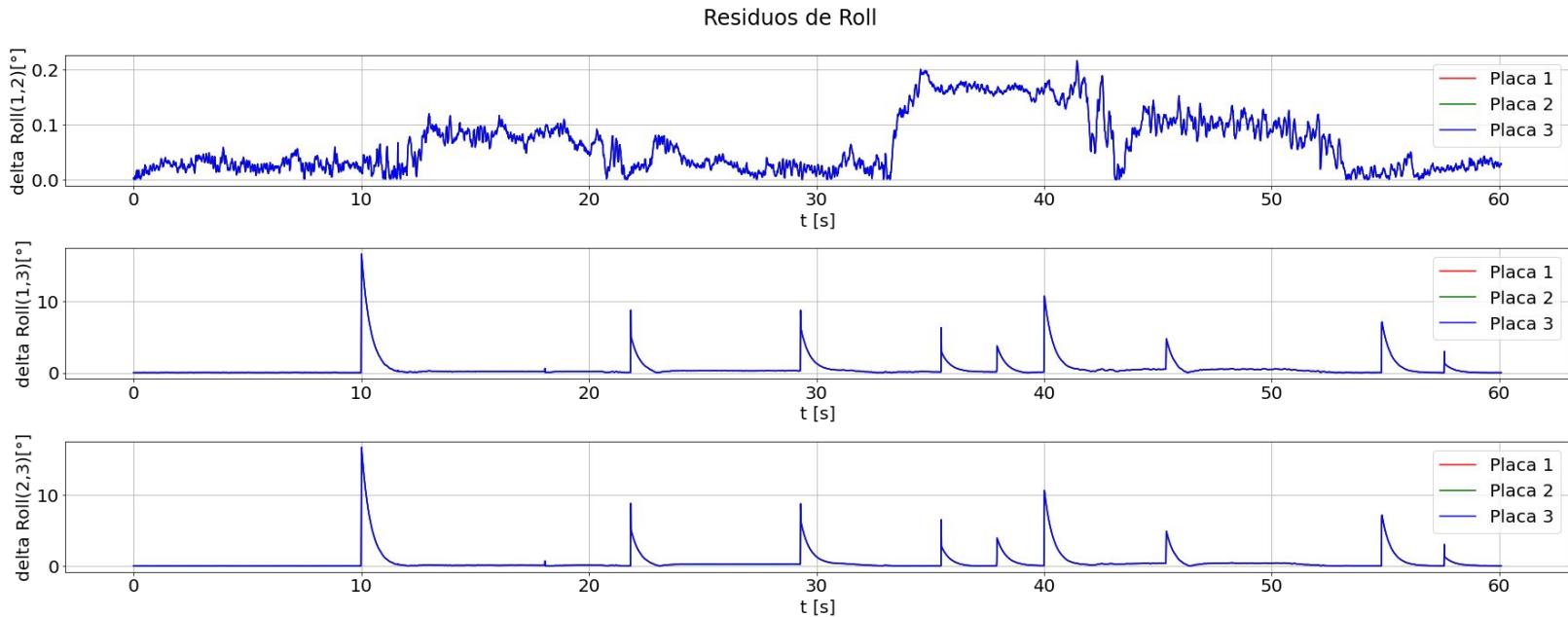
- Residuos calculados por las 3 réplicas:

- 
- Se suma un sesgo a la medición de giróscopos de la placa 3.
  - El sesgo se aplica a las mediciones del eje x:  $\omega_x + b_x$ .
  - En este caso, el sesgo se aplica e inmediatamente se remueve, generando un efecto de saltos.
  - En cada caso, los valor de amplitud, así como el instante de tiempo en que se aplica el sesgo, se calculan de manera aleatoria.



# Mediciones Espúreas en Valores de Giróscopo

- Residuos calculados por las 3 réplicas superpuestos:



---

## 6. Conclusiones y Perspectivas



# Conclusiones

- Se desarrolló una computadora de vuelo con sensores y componentes actualizados. En este trabajo se utilizaron 3 placas, pero se han fabricado otras más. Una de ellas ha sido utilizada en un ASV recientemente.
- Si bien durante la etapa de diseño no fue posible actualizar el microcontrolador por una de las alternativas evaluadas, hoy en día esto sí pudo lograrse gracias a que existen microcontroladores de mejores prestaciones, compatibles pin a pin.
- Se adquirió conocimiento acerca del estado del arte y de las características de sistemas tolerantes a fallas a partir del uso de redundancias.
- Se desarrolló un firmware que implementa una arquitectura Time-Triggered distribuida, demostrando las capacidades de la computadora de vuelo para ser utilizada en sistemas redundantes.



# Perspectivas

- Deben incorporarse todos los demás sensores necesarios para obtener una estimación completa de la pose del vehículo: barómetro, magnetómetro, GPS, etc.
- Debe evaluarse **cómo se comportará el sistema con mayor carga computacional**. Por ejemplo, con la integración de un algoritmo de estimación más complejo.
- Debe evaluarse la **integración de la actuación sobre los motores**. Típicamente para esto se utilizan señales PWM, pero esto obliga a que la actuación sobre estos no sea distribuida.
- Reemplazar el microcontrolador por una alternativa con un **periférico CAN con funcionalidades time-triggered**. Esto simplificaría la tarea de envío y recepción de mensajes en los tiempos predefinidos.

---

# Agradecimientos

- Laboratorio de Automática y Robótica (LAR).
- A la universidad pública.
- A mis amigos, mi familia, compañeros de estudio y de trabajo.



# ¡Muchas gracias!

