



RC Car APXGP Telemetry Challenge

[Vallentina Diaz Valbuena](#)

[Juan Esteban Mora Vaca](#)

Luis Carlos Leal Gamboa

Brayan Steven Mendivelso Perez

Docente: José de Jesús Rugeles

structured using modular architectures, including UML diagrams and flowcharts. Laboratory measurements - oscilloscope, logic analyzer, and spectrum analyzer - validated the RF link, PWM actuation, and SPI communication. Additionally, a PC telemetry application was developed to visualize sensor data, record performance, and export CSV logs.

The results demonstrate a stable bidirectional wireless link, low-latency control response, correct sensor acquisition, and reliable visualization of real-time telemetry. The system fulfills all technical requirements established in the project, showcasing a robust integration of hardware, firmware, and radio communication.

I. RESUMEN

Este informe presenta el diseño, construcción y validación del sistema de telemetría inalámbrica desarrollado para el carro APXGP, en el marco del curso de Comunicaciones Digitales. El sistema integra una Raspberry Pi Pico 2W, un transceptor nRF24L01 y sensores obligatorios (IMU, GPS, temperatura del motor, nivel de batería y sensor de línea), cuya información es transmitida en tiempo real hacia una estación de recepción en PC.

Se diseñaron PCB propias para el control remoto, módulo de sensores, gateway de telemetría y módulo de actuadores, cumpliendo con las exigencias de construcción del proyecto. El firmware se implementó en MicroPython y se estructuró con diagramas de flujo y UML que describen el funcionamiento del sistema. Las mediciones de laboratorio realizadas mediante osciloscopio, analizador lógico y analizador de espectros permitieron validar la calidad del enlace RF, el control PWM y la comunicación SPI.

Los resultados evidencian un sistema estable, con transmisión confiable de telemetría, respuesta rápida del control remoto y una interfaz de monitoreo clara para el operador. El proyecto cumple con todos los requisitos técnicos establecidos y demuestra un diseño integral que combina hardware, software y comunicaciones digitales.

Abstract

This document presents the design, implementation, and validation of a complete wireless telemetry system for the RC Car APXGP, developed as part of the Digital Communications course. The system integrates a Raspberry Pi Pico 2W, an nRF24L01 transceiver, and a set of mandatory onboard sensors (IMU, GPS, motor temperature, battery monitor and line sensor) to transmit real-time data to a ground station.

Custom PCBs were designed for each subsystem: remote control, sensor module, RF gateway and actuator controller. The firmware was implemented in MicroPython and

II. INTRODUCCIÓN

El desarrollo de sistemas de telemetría inalámbrica constituye un elemento fundamental en aplicaciones modernas de control, monitoreo y automatización. En el contexto del curso de Comunicaciones Digitales, el proyecto RC Car Telemetry Challenge tiene como objetivo integrar los conceptos teóricos de modulación, protocolos digitales, enlaces inalámbricos y diseño de hardware embebido mediante la construcción de un sistema real operando en la banda ISM de 2.4 GHz.

El presente informe documenta el proceso técnico llevado a cabo para diseñar un sistema completo de telemetría para el vehículo APXGP, incluyendo los subsistemas de control remoto, adquisición de sensores, transmisión RF, recepción de datos, visualización en PC y registro de información. Cada módulo fue construido con PCB propias, cumpliendo con las reglas de diseño mecánico y eléctrico exigidas por el proyecto.

Además, se validó experimentalmente el funcionamiento del enlace inalámbrico mediante mediciones de laboratorio utilizando osciloscopio, analizador lógico y analizador de espectros. El sistema fue complementado con una aplicación de telemetría en PC capaz de recibir, interpretar y visualizar los datos en tiempo real, así como generar reportes en formato CSV para análisis posterior.

Este informe busca consolidar todos los resultados del proceso de diseño, así como el desempeño del sistema durante las pruebas en pista, analizando su estabilidad, confiabilidad y la calidad de la comunicación en condiciones reales.

III. DESARROLLO

Esquema del sistema

La Figura 1 presenta el diagrama general del sistema de telemetría desarrollado para el vehículo APXGP. El sistema se estructura en tres bloques principales: el carro de competencia, el control remoto y la estación de recepción en PC.

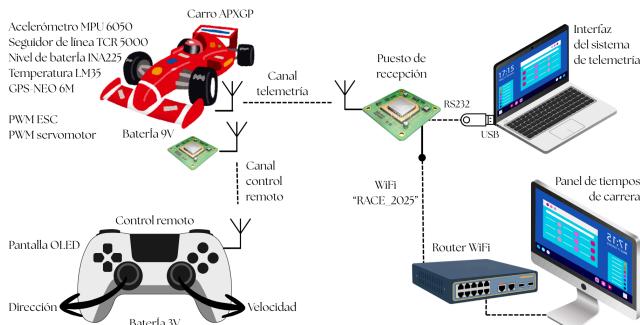


Figura 1. Esquema del sistema

El carro integra el microcontrolador Raspberry Pi Pico 2W, el módulo transceptor nRF24L01 y el conjunto de sensores obligatorios: IMU MPU6050, sensor de línea TCRT5000, GPS NEO-6M, medición de voltaje mediante INA225 y temperatura del motor con LM35. Estos datos son transmitidos a la estación base a través del canal de telemetría en la banda de 2.4 GHz.

El control remoto envía comandos de dirección y velocidad por un segundo canal inalámbrico independiente, garantizando la separación entre control y telemetría. Para una realimentación local se incorpora una pantalla OLED con información relevante del vehículo.

La estación de recepción procesa los datos recibidos y los entrega por interfaz USB/RS232 al software de telemetría ejecutado en el computador del equipo, donde se visualizan en tiempo real. La aplicación también se conecta por red local al panel de tiempos de carrera, que centraliza la información de todos los vehículos en competición.

Diagrama de bloques

El diagrama de bloques muestra la arquitectura general del sistema APXGP, dividido en tres módulos funcionales: el carro, el control remoto y el puesto de recepción. El carro integra los sensores de telemetría (MPU6050, LM35, INA225, TCR5000 y GPS NEO-6M), junto con los actuadores ESC y servomotor, los cuales son gestionados por el microcontrolador principal. Este módulo envía datos en tiempo real al puesto de recepción mediante un canal inalámbrico dedicado.

El control remoto permite la interacción directa del piloto a través de dos joysticks y una interfaz visual, enviando comandos de dirección y velocidad al vehículo. Finalmente, el puesto de recepción opera como nodo central, recibiendo la telemetría, visualizándola en la estación del operador y comunicándose vía WiFi con el panel de tiempos. La estructura modular del sistema garantiza una operación confiable, baja latencia en el control y disponibilidad continua de la información durante la carrera.

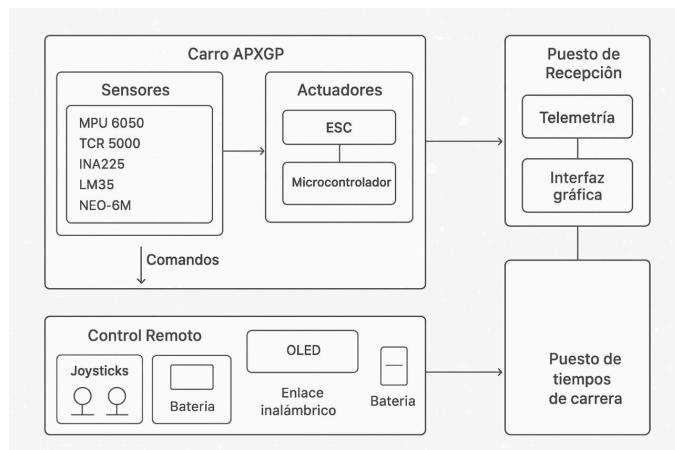


Figura 2. Diagrama de bloques

Planos electrónicos de diseño

El diseño electrónico se divide en tres módulos principales:

- Control de mando
- Módulo de sensores
- Puesto de recepción

Cada módulo fue diseñado con su propio PCB. A continuación se describen los esquemáticos, layouts y características de cada módulo.

Control de mando

El control remoto implementa una interfaz de usuario para el piloto, permitiendo transmitir comandos de velocidad, dirección y paro de emergencia. Está construido alrededor de una Raspberry Pi Pico 2W y un transceptor NRF24L01 dedicado únicamente para transmisión. Los componentes incluidos se presentan a continuación:

- 2 Joysticks analógicos (ADC): Control de dirección y velocidad.
- Pantalla OLED (I2C): Visualización de datos del sistema y confirmación de enlace.
- Botón E-Stop (digital): Parada de emergencia.
- NRF24L01 (SPI): Enlace de control hacia el carro.
- Batería 3V: Alimentación del módulo.

El PCB fue diseñado para mantener comodidad ergonómica, conectores laterales y zonas de montaje para joystick y pantalla. El microcontrolador Pico 2W ejecuta el firmware de generación de comandos PWM y se alimenta mediante dos baterías de 1.5V con regulación a 3.3V. En las siguientes figuras se presenta el esquemático del módulo, la PCB top y bottom y la vista 3D:

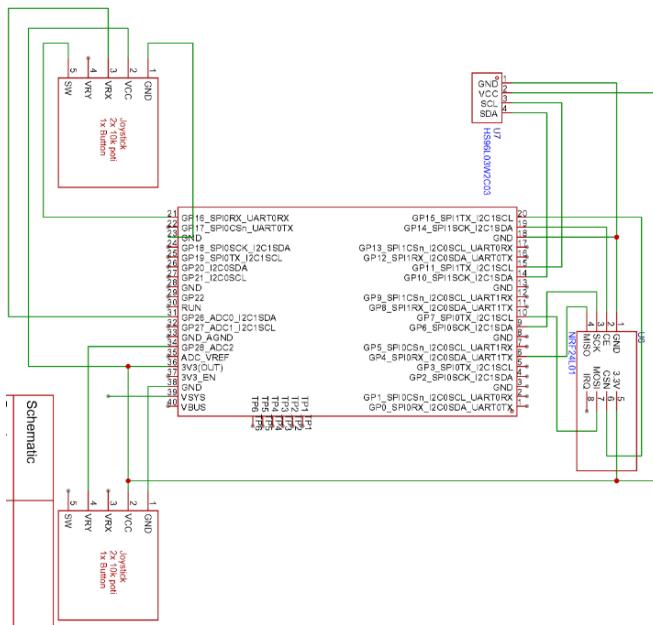


Figura 3. Esquemático del control de mando

Diseño PCB

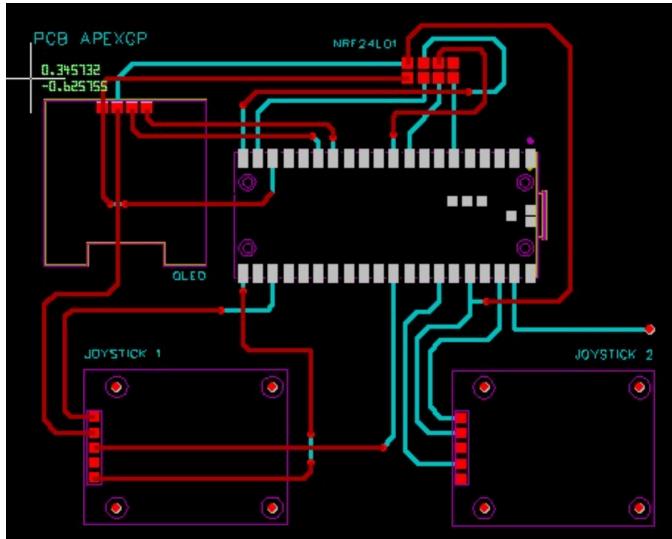


Figura 4. PCB del control de mando

Vista 3D

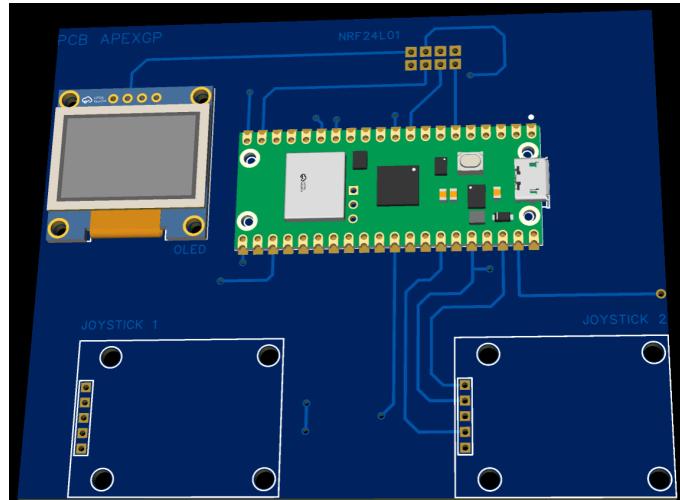


Figura 5. Vista 3D del control de mando

Módulo de sensores en el carro

Este módulo integra los sensores obligatorios para la adquisición de datos del vehículo durante la carrera. El diseño se conecta directamente a la Raspberry Pi Pico 2W mediante buses I2C, UART, SPI y ADC según corresponda. Los sensores incluidos se presentan a continuación:

- **MPU-6050 (I2C):** Acelerómetro y giroscopio para obtener la orientación y aceleración del vehículo.
- **GPS NEO-6M (UART):** Información de posición, velocidad y tiempo.
- **INA225 (I2C):** Medición de voltaje/corriente de la batería del carro.
- **LM35 (ADC):** Medición de temperatura del motor.
- **TCRT5000 (digital):** Detección de línea para registro del tiempo de paso por meta (negro-blanco-negro-blanco).
- **NRF24L01 (SPI):** Comunicación inalámbrica del canal de telemetría.

El diseño de la PCB mantiene una distribución que reduce el ruido en líneas de alimentación y señales analógicas, incluyendo capacitores de desacople y rutas cortas para señales sensibles. Además, incluye conectores tipo JST para facilitar el ensamblaje y distribución de tierras para reducir interferencia en los sensores sensibles. En las siguientes figuras se presenta el esquemático del módulo, la PCB top y bottom y la vista 3D:

Esquemático

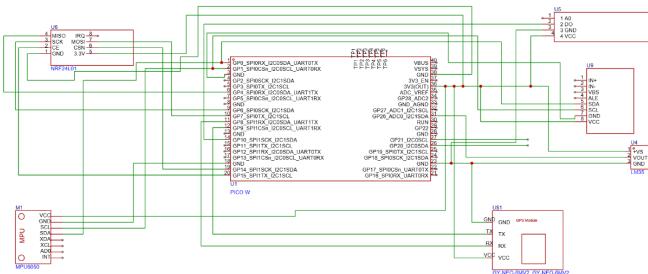


Figura 6. Esquemático del módulo de sensores

Diseño PCB

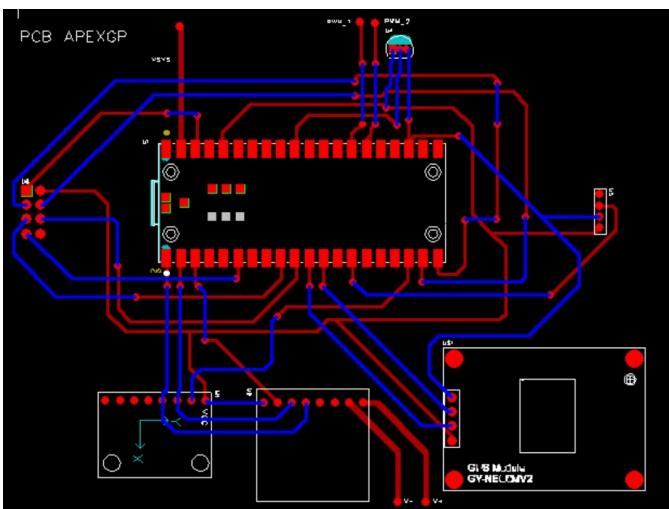


Figura 7. PCB del módulo de sensores

Vista 3D

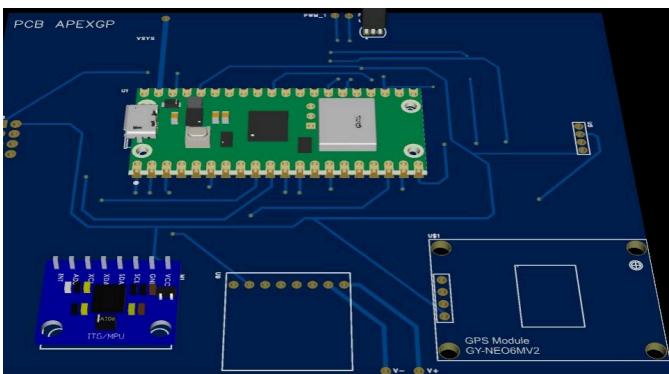


Figura 8. Vista 3D del módulo de sensores

Puesto de recepción

El puesto de recepción actúa como gateway entre el sistema inalámbrico y la aplicación de telemetría en el PC. Su función principal es recibir paquetes del carro, procesarlos y enviarlos a través de USB al computador. Los componentes incluidos se presentan a continuación:

- Raspberry Pi Pico 2W: Responsable de decodificar paquetes y enviarlos al PC.
- NRF24L01: Receptor exclusivo del canal de telemetría.
- Interfaz USB/CDC: Comunicación con la aplicación de telemetría.
- Alimentación por USB: Simplifica la instalación durante la carrera.

El diseño PCB integra conectores firmes, zonas de desacople RF y un ruteo optimizado del bus SPI hacia el transceptor. En las siguientes figuras se presenta el esquemático del módulo, la PCB top y bottom y la vista 3D:

Esquemático

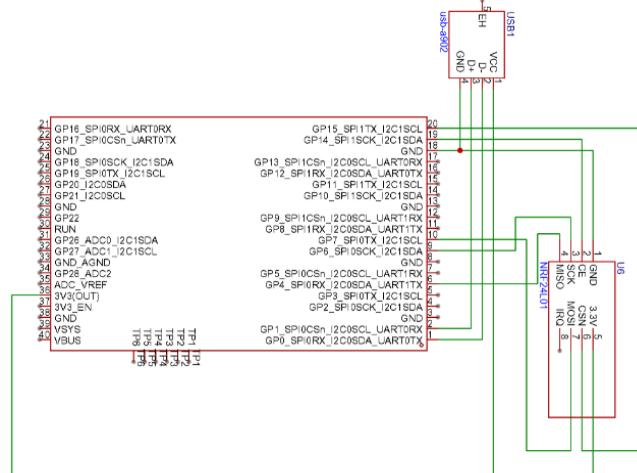


Figura 9. Esquemático del puesto de recepción

Diseño PCB

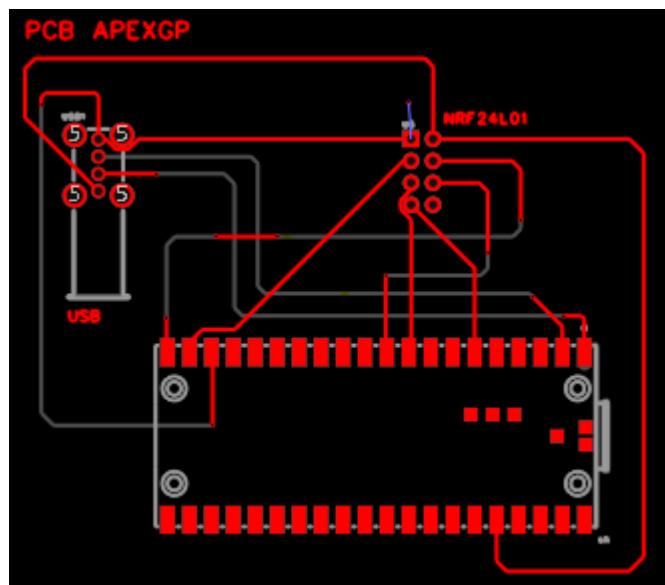


Figura 10. PCB del puesto de recepción



Vista 3D

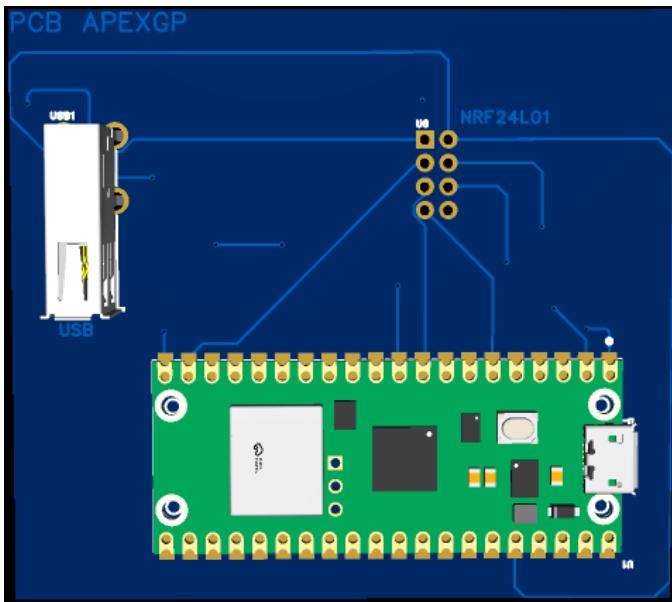


Figura 11. Vista 3D del puesto de recepción

Modulo de recepcion en el carro

El módulo de recepción integrado en el chasis del vehículo está compuesto por una Raspberry Pi Pico 2W y un transceptor nRF24L01 que actúan como receptor local del enlace RF de control. Este módulo recibe los paquetes de comando provenientes del control remoto y aplica las órdenes sobre los actuadores principales: el ESC del motor (PWM de tres fases y alimentación del motor) y el servomotor de dirección (entrada PWM de 3 hilos). Sus funciones principales se describen a continuación:

Recepción de paquetes de control (nRF24L01) y validación/decodificación, conversión de órdenes a señales PWM hacia ESC (control de velocidad/avance/reversa) y servo (control de dirección), gestión de parada de emergencia (E-Stop) con corte lógico inmediato y señal física al ESC y la supervisión básica de alimentación del tren motriz (detección de fallo de alimentación y reportes por telemetría si procede).

Las interfaces eléctricas y mecánicas son:

- NRF24L01 (SPI) al Pico 2W (CSN, CE, SCK, MOSI, MISO, IRQ) — nivel lógico 3.3 V.
- ESC conectado a la salida PWM del Pico (nivel de control tipo servo, 3.3 V) y alimentación del motor proveniente de la batería.
- Servomotor conectado a salida PWM dedicada (3.3 V control, VCC 5–6V según servo) mediante conector JST/servo estándar (S/G/V).
- Conexión tierra común entre batería, ESC (electrónica de control) y Pico.

El módulo está diseñado como una pequeña placa secundaria en baquelita que se monta cerca de la salida del motor para minimizar longitud de los cables de potencia y con conectores macho/hembra para facilitar montaje/desmontaje y mantenimiento.

Esquemático

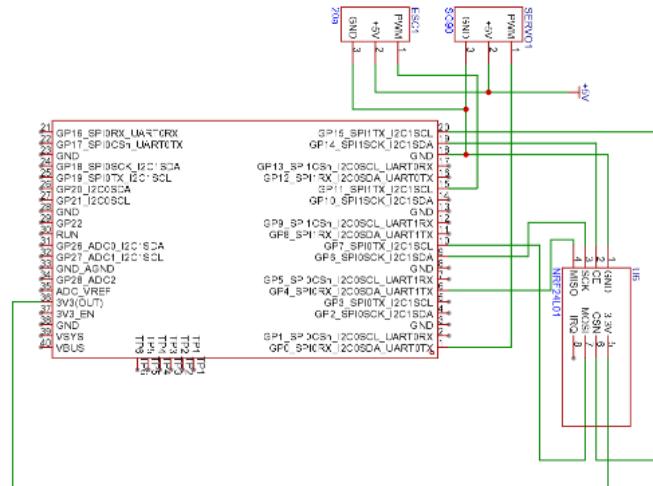


Figura 12. Esquemático del módulo de recepción en el carro

Diseño PCB

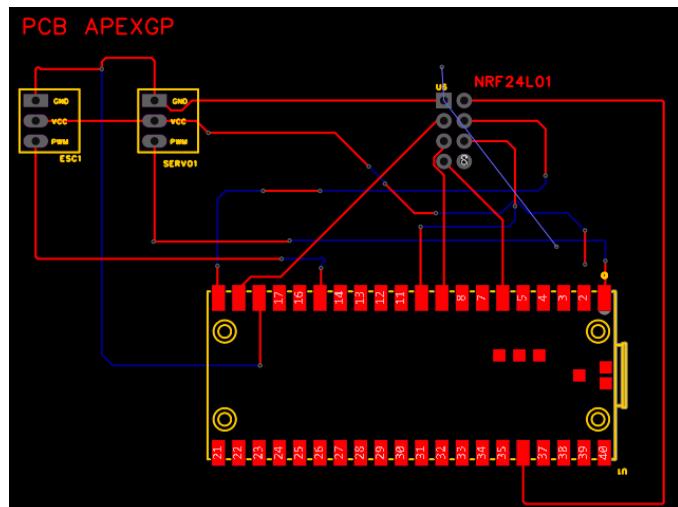


Figura 13. PCB del módulo de recepción en el carro

Vista 3D

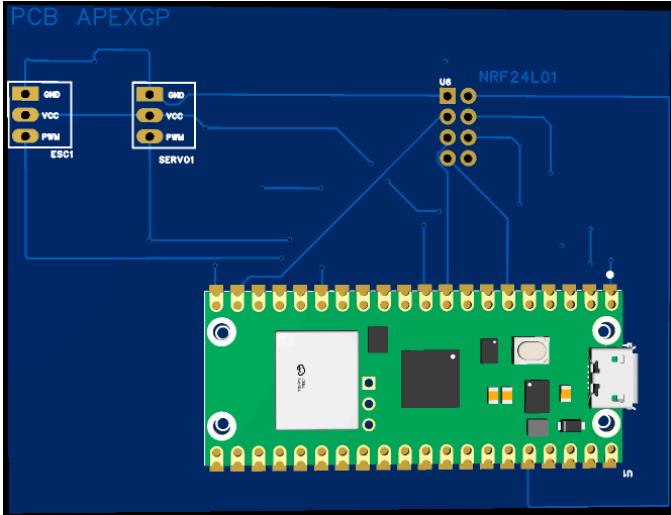


Figura 14. Vista 3D del módulo de recepción en el carro

Diagramas de flujo

Este apartado presenta los diagramas de flujo asociados al funcionamiento del firmware en cada uno de los módulos del sistema de telemetría: el carro APXGP, el control remoto y el puesto de recepción. Los diagramas describen la lógica de adquisición, procesamiento, transmisión y recepción de datos, así como las rutinas de control que permiten la operación estable del vehículo durante la competencia.

Los procesos aquí descritos se implementaron en MicroPython sobre la Raspberry Pi Pico 2W y se encuentran directamente relacionados con el comportamiento del enlace inalámbrico, el muestreo de sensores y el manejo de actuadores.

Carro APXGP (Nodo remoto)

Este proceso describe la rutina principal ejecutada por la Raspberry Pi Pico 2W en el vehículo, incluyendo la adquisición de sensores, el empaquetamiento de datos y el envío de telemetría al puesto de recepción.

Flujo del firmware del carro

El firmware realiza un ciclo de muestreo continuo, priorizando la actualización del GPS y la IMU. El envío de telemetría se hace a intervalos constantes, controlados mediante temporizadores internos del microcontrolador. El procesamiento incluye conversión de unidades y normalización para reducir carga en el receptor.

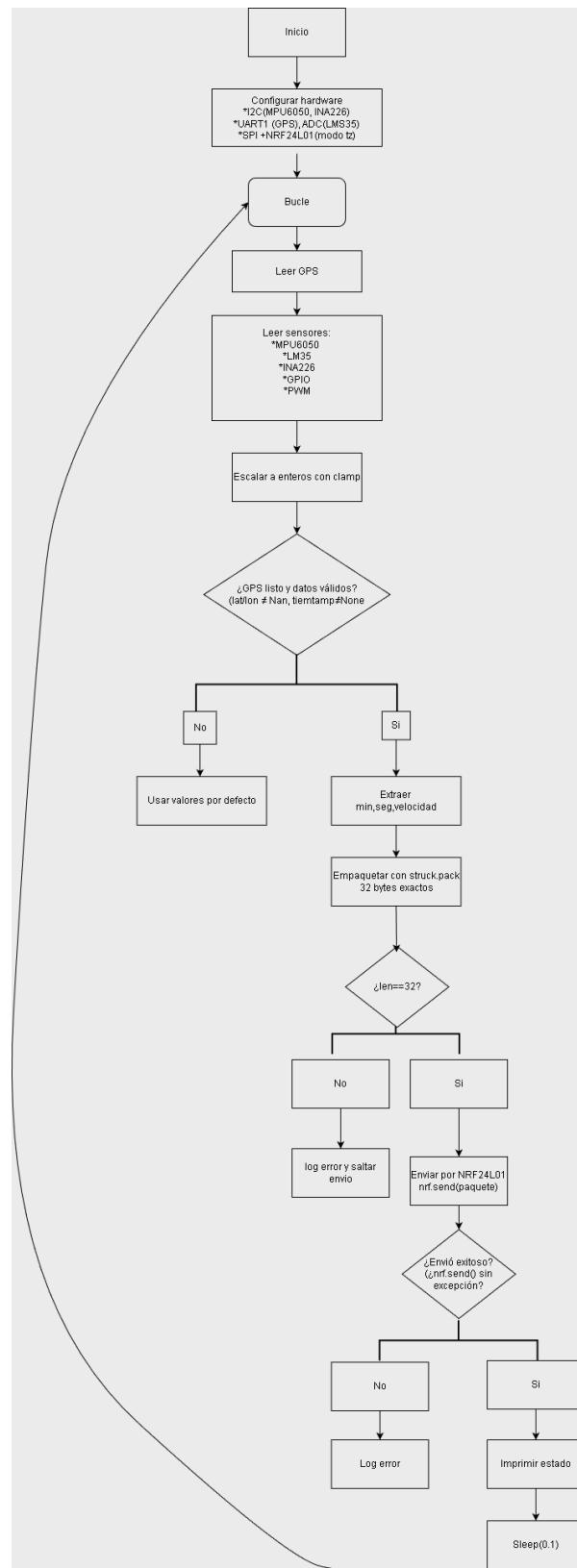


Figura 15. Flujo del firmware del carro

Control remoto

El control remoto genera y transmite continuamente las órdenes del piloto hacia el carro. Su lógica se centra en la



lectura de joysticks, la actualización del display OLED y el envío de comandos por RF.

Flujo del firmware del transmisor del control remoto

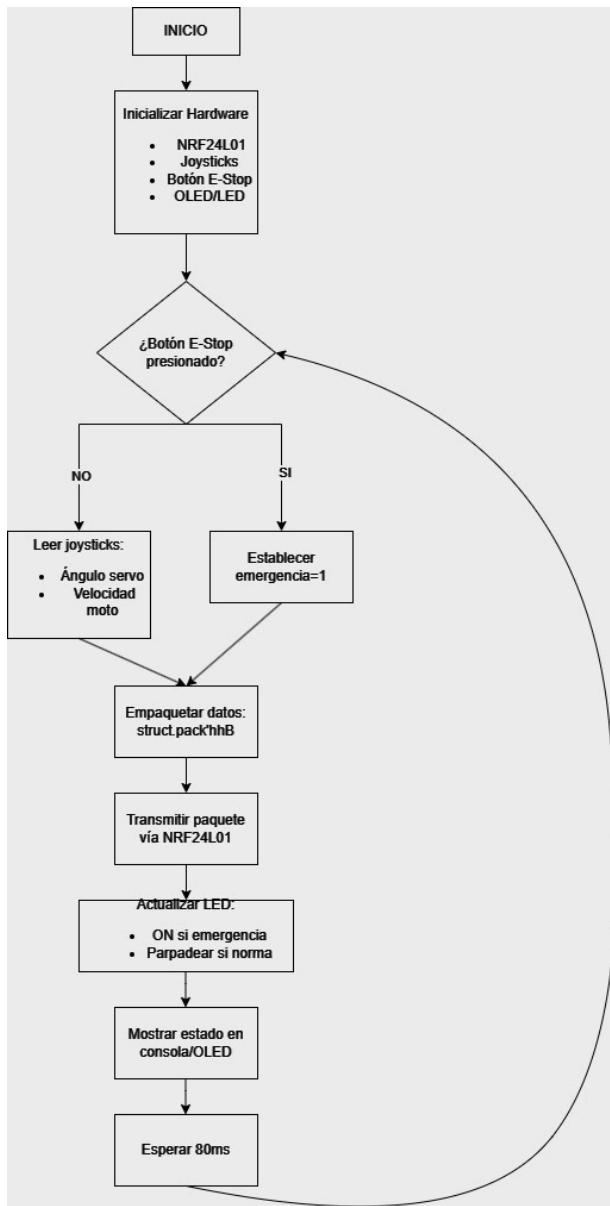


Figura 16. Flujo del firmware del control de mando

El control remoto opera con un ciclo de muestreo rápido (<80 ms) para evitar retrasos perceptibles por el usuario. En caso de activación del E-Stop, se envía un paquete especial que obliga al carro a detenerse inmediatamente. La pantalla OLED muestra estado RF, niveles de joystick y conexión.

Flujo del firmware del receptor del control remoto

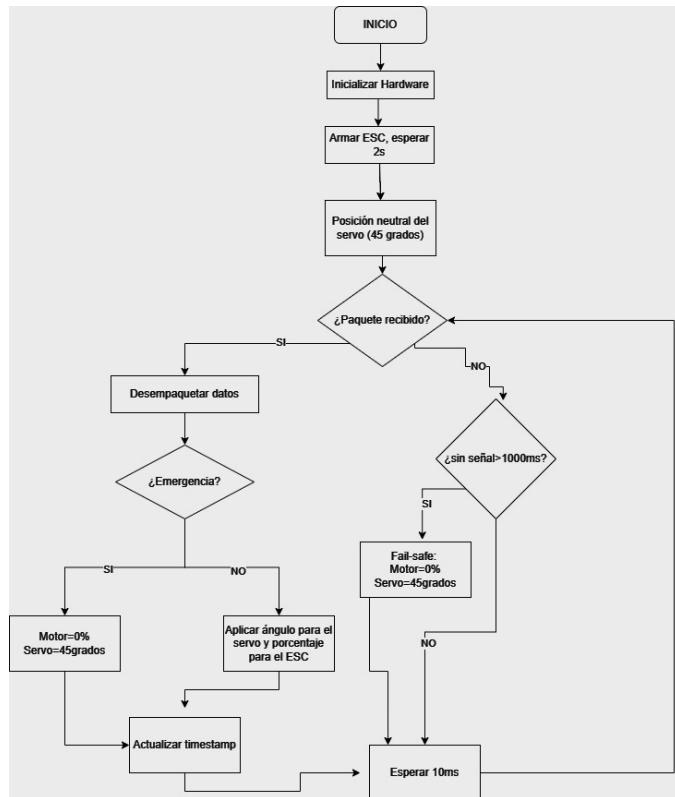


Figura 17. Flujo del firmware del control de mando

Puesto de recepción

El puesto de recepción actúa como gateway entre el enlace RF y el computador. Se encarga de recibir telemetría, validarla, decodificarla y transmitirla por USB hacia la interfaz gráfica del sistema.

Flujo del firmware del puesto de recepción

El nodo receptor utiliza el modo RX permanente del NRF24L01. Cada paquete recibido es validado mediante un sistema de checksum o CRC (dependiendo del firmware). Los datos se envían por USB al computador en formato estructurado. El sistema está diseñado para operarse sin pérdida de datos durante la carrera.

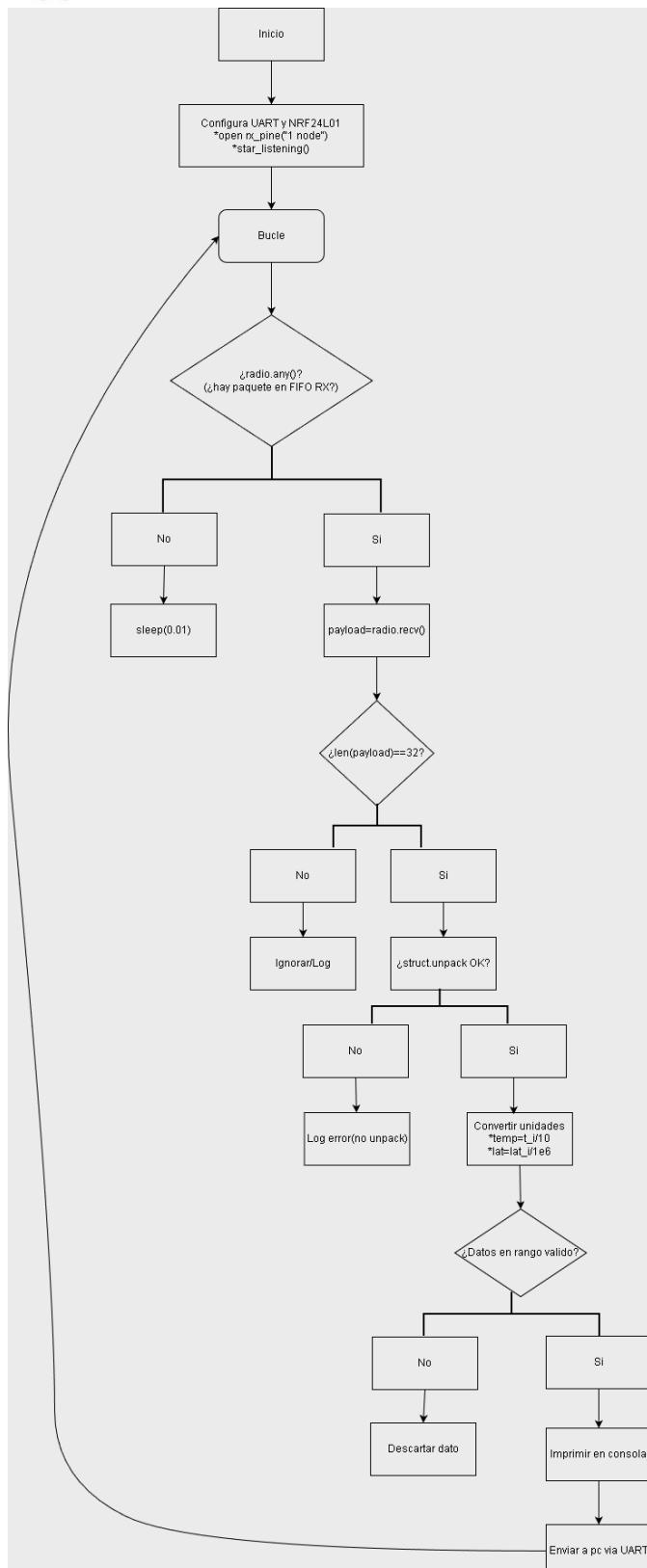


Figura 18. Flujo del firmware del puesto de recepción

Diagramas UML del software

Este apartado define la arquitectura lógica y las interacciones de software del sistema de telemetría. Se incluyen: diagramas

de casos de uso (actores y principales funcionalidades), diagrama de clases para firmware (carro y control remoto) y aplicación PC, y diagramas de secuencia que describen flujos críticos: envío de comando, envío de telemetría y arranque del sistema.

Diagrama de casos de uso

El diagrama de casos de uso describe las interacciones funcionales entre los actores involucrados y el sistema de telemetría APXGP. El sistema se organiza en torno a cuatro actores principales: el piloto, el carro APXGP, el puesto de recepción/operador y el sistema de carrera. Cada uno de ellos participa en funciones específicas que permiten la operación, monitoreo y supervisión del vehículo durante la competencia.

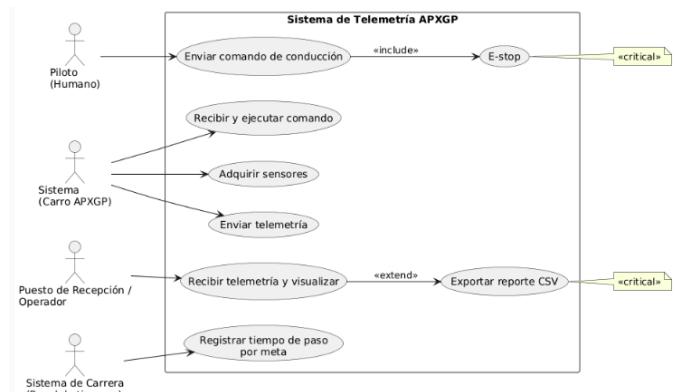


Figura 19. Diagrama de casos de uso

El *piloto* interactúa con el sistema enviando comandos de conducción mediante el control remoto. Este caso de uso incluye una operación crítica de seguridad, el *E-stop*, implementada mediante una relación *include* debido a que forma parte integral del comando de control. El *carro APXGP*, por su parte, actúa como un subsistema autónomo que recibe los comandos enviados por el piloto, ejecuta las acciones correspondientes y adquiere la información proveniente de los diferentes sensores integrados. Además, el carro transmite continuamente la telemetría hacia el puesto de recepción.

El *puesto de recepción/operador* recibe esta telemetría, la procesa y la representa visualmente para el operador. Adicionalmente, puede ejecutar acciones extendidas como *exportar reporte CSV*, un caso de uso marcado como crítico debido a su relevancia para el análisis posterior de datos. Finalmente, el *sistema de carrera* registra los tiempos de paso del vehículo por la meta, utilizando la información enviada por el carro para mantener actualizado el panel de tiempos.

En conjunto, el diagrama evidencia un flujo funcional claro entre los actores y el sistema, destacando los puntos de interacción esenciales para garantizar un control seguro del vehículo, una supervisión en tiempo real y un registro confiable del desempeño durante la carrera.



Diagrama de clases

Clases (Firmware - Carro)

El diagrama de clases del firmware del carro APXGP representa la estructura modular del sistema embebido encargado de la adquisición de datos, el control de actuadores y la transmisión de telemetría. La clase principal *PicoCarController* coordina el funcionamiento general del vehículo, gestionando los periféricos, procesando los comandos recibidos por RF y generando los paquetes de telemetría enviados al puesto de recepción.

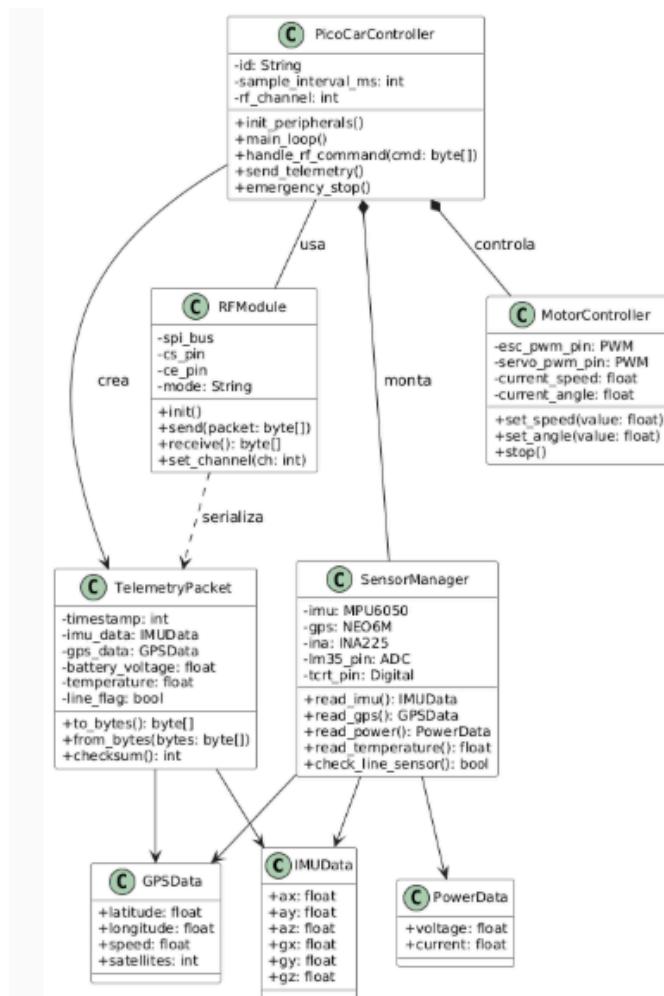


Figura 20. Diagrama de clases del firmware del carro

El subsistema sensorial está modelado mediante la clase *SensorManager*, responsable de obtener datos de la IMU, GPS, sensor de temperatura, sensor de línea y monitor de batería. Los diferentes tipos de información se encapsulan en clases de datos dedicadas, como *IMUData*, *GPSData* y *PowerData*, lo que favorece la organización y la extensibilidad del firmware. Para la actuación, la clase *MotorController* gestiona el ESC y el servomotor, permitiendo controlar velocidad y ángulo de dirección. El módulo de comunicación inalámbrica se abstrae en la clase *RFModule*, utilizada por el controlador principal para enviar o recibir información.

En conjunto, la arquitectura modelada por este diagrama permite un sistema robusto, escalable y fácil de mantener, donde cada componente cumple un rol específico dentro del ciclo de control y telemetría del vehículo.

Clases (Firmware - Control Remoto)

El diagrama de clases del firmware del control remoto describe el sistema encargado de capturar las acciones del piloto y transmitirlas en forma de comandos hacia el vehículo. La clase central *RemoteController* coordina la lectura de los joysticks, la construcción de paquetes de control y el envío de estos mediante el módulo RF. La interacción con el usuario se complementa con la clase *DisplayOLED*, que presenta información relevante al piloto como estado del enlace, valores de velocidad y ángulo o indicadores de batería.

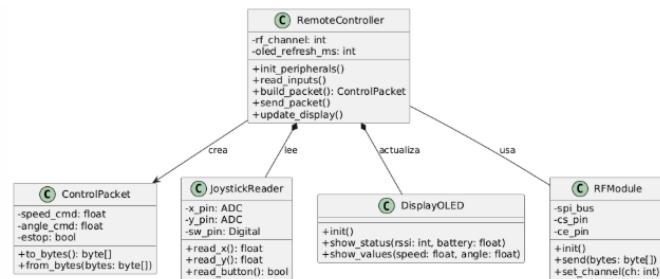


Figura 21. Diagrama de clases del firmware del control

La clase *JoystickReader* abstrae la lectura de los dos ejes analógicos y del botón integrado, permitiendo obtener valores normalizados para el cálculo de los comandos. Los datos transmitidos al carro se agrupan en la clase de mensaje *ControlPacket*, que contiene los parámetros de dirección, aceleración y la bandera de parada de emergencia. Finalmente, la clase *RFModule* gestiona el enlace inalámbrico, incluyendo la selección del canal.

Este diseño modular separa claramente la adquisición de entrada, la lógica de control y la comunicación inalámbrica, lo que facilita la portabilidad, depuración y mantenimiento del firmware del control remoto.

Clases (Puesto de Recepción / Gateway RF → USB)

El diagrama de clases del puesto de recepción modela el firmware que actúa como puente entre el enlace RF del vehículo y la interfaz USB hacia el computador del operador. La clase principal *ReceiverGateway* se encarga de inicializar los módulos de comunicación, escuchar continuamente el canal RF, validar los paquetes de telemetría y reenviarlos por el puerto USB.

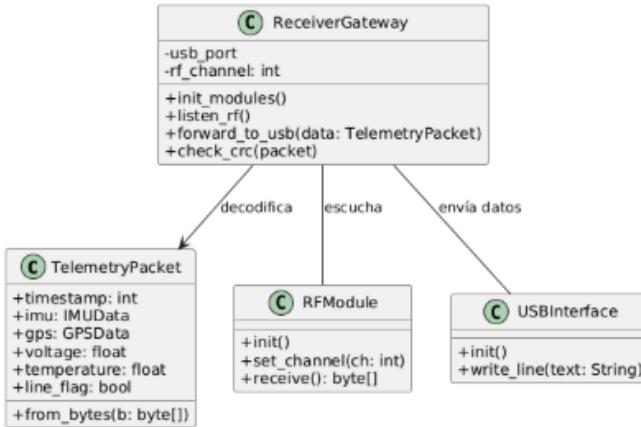


Figura 21. Diagrama de clases del puesto de recepción

La clase *RFModule* implementa la comunicación inalámbrica con el carro, permitiendo la recepción de tramas que luego son procesadas para formar objetos *TelemetryPacket*, los cuales encapsulan los datos sensoriales transmitidos. La clase *USBInterface* abstrae la comunicación con el computador, enviando líneas de texto o paquetes en formato compatible con la aplicación del operador.

El modelo refleja un sistema dedicado, de una sola tarea, cuyo propósito es garantizar un flujo confiable y de baja latencia entre el vehículo y el software de telemetría. La separación entre recepción, decodificación y reenvío permite una estructura limpia y fácil de depurar.

Clases (Interfaz de PC / Software Telemetría)

El diagrama de clases de la interfaz de PC muestra la organización del software encargado de visualizar, registrar y analizar la telemetría del vehículo en tiempo real. La clase *PCApplication* coordina la ejecución general, manteniendo la lista histórica de registros de telemetría, actualizando la interfaz gráfica y gestionando la exportación de datos.

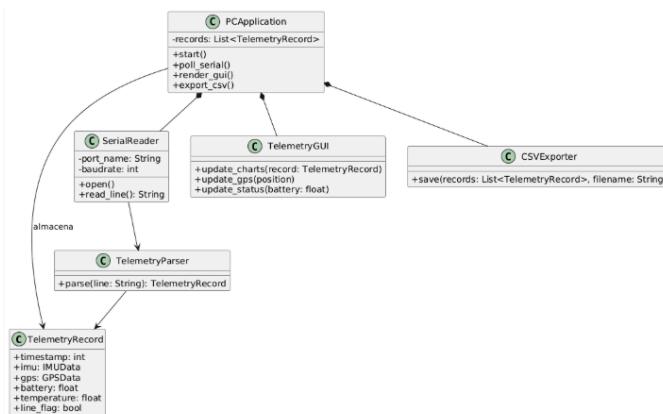


Figura 21. Diagrama de clases de la interfaz de PC

La clase *SerialReader* se encarga de la comunicación con el puesto de recepción mediante el puerto serial USB, mientras que *TelemetryParser* interpreta las líneas recibidas y las

convierte en objetos *TelemetryRecord*, que contienen los datos del vehículo en un formato estructurado. La clase *TelemetryGUI* representa los elementos de la interfaz visual del operador, como gráficos, indicadores, posición GPS y estado del sistema. Para fines de análisis y almacenamiento, la clase *CSVExporter* permite guardar los registros en archivos compatibles con herramientas externas.

Este diagrama evidencia una arquitectura modular orientada a componentes, donde cada función del sistema de telemetría se encuentra claramente separada, facilitando la escalabilidad, la mantenibilidad y la integración con futuras mejoras.

Diagramas de secuencia

Secuencia — Enviar Comando desde Control → Carro

La secuencia representa el proceso mediante el cual el piloto envía un comando de conducción al vehículo a través del sistema de radiofrecuencia nRF24L01.

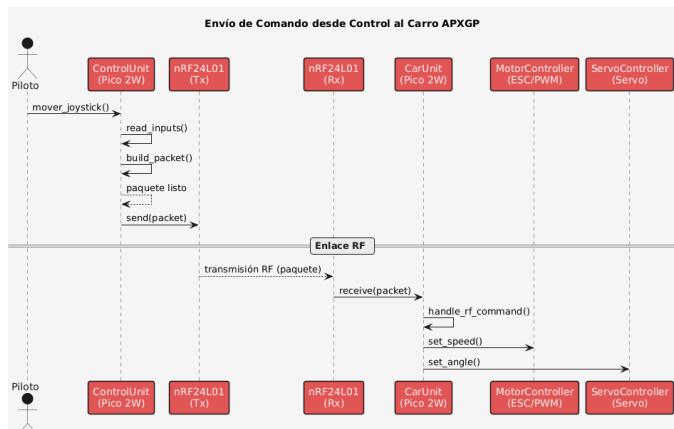


Figura 22. Diagrama de la primera secuencia

El flujo inicia cuando el piloto mueve el joystick del control. La unidad de procesamiento del control (ControlUnit) lee el estado de todos los inputs utilizando la función *read_inputs()*. Una vez obtenidos los valores de dirección y velocidad, el microcontrolador construye un paquete digital mediante *build_packet()*, el cual contiene la información codificada del comando.

Posteriormente, el control transfiere el paquete al módulo nRF24L01 en modo transmisión, que lo envía de manera inalámbrica hacia el vehículo mediante un mensaje asincrónico RF. El módulo receptor del carro recibe el paquete y se lo entrega al CarUnit, que ejecuta *handle_rf_command()* para decodificar y validar su contenido.

Finalmente, el controlador del carro actualiza los actuadores: se envía una orden al MotorController para modificar el PWM del motor (*set_speed()*), y otra al ServoController para ajustar el ángulo de dirección (*set_angle()*). Con esto el vehículo ejecuta físicamente la acción indicada por el piloto.



Secuencia — Telemetría Carro → Receptor → PC

Esta secuencia describe el proceso completo de adquisición, transmisión y visualización de la telemetría generada por el carro en tiempo real.

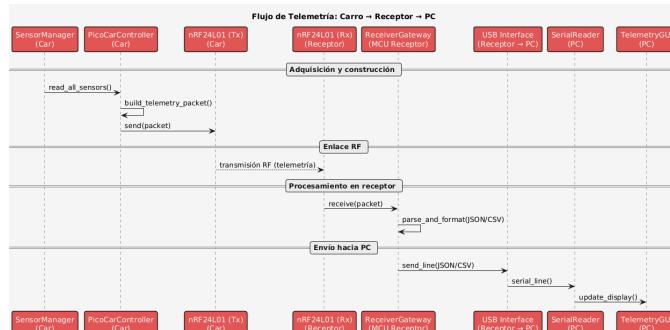


Figura 23. Diagrama de la segunda secuencia

El flujo inicia en el módulo SensorManager, que realiza la lectura de todos los sensores a bordo (batería, temperatura del motor, IMU, GPS, TCRT5000, etc.) mediante `read_all_sensors()`. La información adquirida es consolidada por el PicoCarController, que construye un paquete digital con los datos utilizando `build_telemetry_packet()`.

Una vez preparado el paquete, este es enviado al módulo nRF24L01 (Tx), encargado de transmitir inalámbricamente. El enlace RF entrega el paquete al receptor, donde el módulo nRF24L01 (Rx) pasa el mensaje al ReceiverGateway. Allí se realiza el proceso de decodificación y formateo de datos (`parse_and_format()`), generando una línea JSON o CSV.

El Gateway envía la línea procesada al computador a través de la interfaz USB. El programa SerialReader en el PC recibe cada línea mediante `serial_line()` y la interpreta. Finalmente, los datos son actualizados en tiempo real en la interfaz gráfica TelemetryGUI, permitiendo al operador visualizar parámetros como voltaje, temperatura, posición GPS, aceleraciones y estado del vehículo.

Secuencia — Exportación de CSV en el PC

Esta secuencia ilustra el proceso mediante el cual el operador genera un archivo CSV con los datos de telemetría registrados durante la carrera.

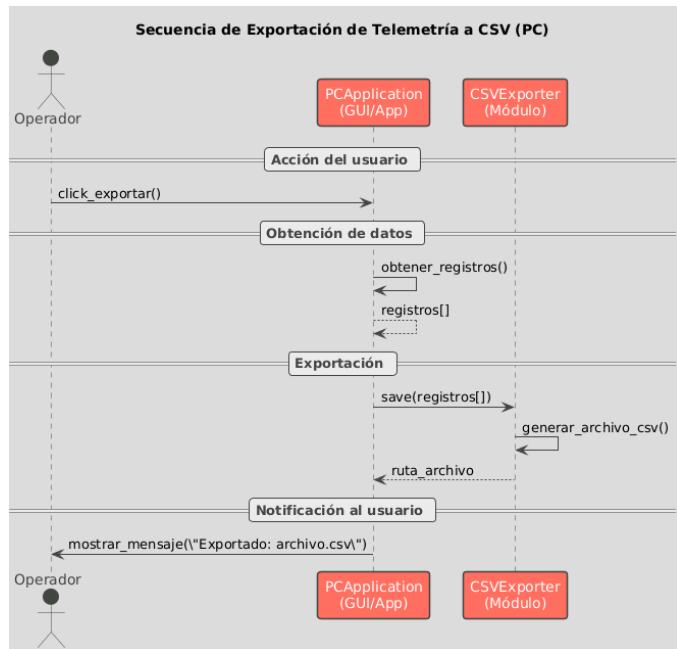


Figura 21. Diagrama de la tercera secuencia

El flujo inicia cuando el operador selecciona la opción de exportar dentro de la aplicación de monitoreo. La PCApplication ejecuta `obtener_registros()` para recopilar todos los datos almacenados en memoria, generando un arreglo con los registros de telemetría.

Luego, la aplicación invoca al módulo CSVExporter mediante `save(registros[])`, entregándole la información procesada. El módulo genera el archivo utilizando `generar_archivo_csv()`, donde se estructura el contenido en formato compatible para análisis posterior.

Tras completar la exportación, el CSVExporter devuelve a la aplicación la ruta del archivo generado. Finalmente, la PCApplication notifica al operador mostrando un mensaje de confirmación (“Exportado: archivo.csv”), indicando que el reporte ha sido creado exitosamente.

Mediciones de Laboratorio

Señal PWM

Como parte de la caracterización completa del sistema de control remoto, se realizó un análisis de la señal de Modulación por Ancho de Pulso (PWM) utilizada para posicionar el servomotor SG90 en el lado del receptor. Esta señal, generada por la Raspberry Pi Pico 2W en respuesta a los datos recibidos de forma inalámbrica, fue capturada y visualizada utilizando un osciloscopio digital. El objetivo de esta medición fue verificar la correspondencia directa entre el ángulo comandado desde el joystick (transmisor) y el ancho del pulso PWM resultante, lo cual valida el correcto



funcionamiento del lazo de control en su totalidad: desde la entrada del usuario hasta el movimiento físico del actuador.

La relación entre el ciclo de trabajo de la señal PWM y el ángulo del servomotor es fundamental para garantizar un posicionamiento preciso. La forma de onda de la señal es la típica cuadrada de un PWM. El servomotor se posiciona en un ángulo específico en función del ancho del pulso positivo. Al variar la posición del joystick en el transmisor, se capturaron los anchos de pulso correspondientes a los extremos y al centro del rango de movimiento:

Pulso Mínimo ($\approx 400 \mu s$): Corresponde al comando para el límite mínimo de giro ($\approx 0^\circ$ o -90°).



Imagen 22. Señal PWM para pulso mínimo

Pulso Neutral ($\approx 1.4 ms$): Corresponde al comando para la posición central o neutral (0°).

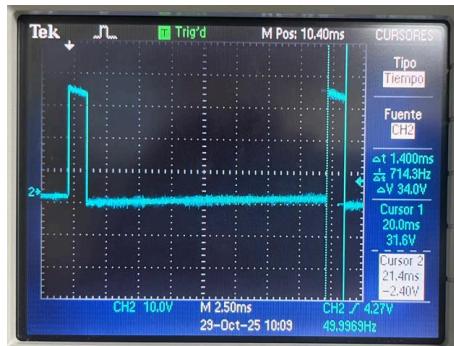


Imagen 23. Señal PWM para pulso neutral

Pulso Máximo ($\approx 2.4 ms$): Corresponde al comando para el límite máximo de giro ($\approx 180^\circ$ o $+90^\circ$).



Imagen 24. Señal PWM para pulso máximo

El análisis confirmó que el microcontrolador genera una señal PWM con un periodo constante de 20 ms (equivalente a una frecuencia de 50 Hz), cumpliendo con el estándar requerido por los servomotores analógicos, como se observa a continuación:



Imagen 25. Período de la señal PWM

Es decir, las capturas en el osciloscopio permitieron confirmar que el microcontrolador genera estos pulsos con la precisión y estabilidad necesarias, asegurando que la réplica del movimiento del joystick en el servomotor fuera fiel y libre de oscilaciones.

Señal SPI

Una vez con todo funcionando correctamente se hace uso del analizador lógico y la herramienta de software logic 2, para analizar la transmisión y cómo se ejecuta y se lleva a cabo, esto se hace con la siguiente referencia del analizador lógico:



Imagen 26. Analizador lógico usado

Una vez conectando la anterior referencia a los pines del transmisor nrf se procede a capturar los datos apagando y volviendo a encender los dos programas tanto el de recepción como el de transmisión, lo que se logra capturar en el inicio,



se muestra a continuación, lo cual se le podría denominar como “negociación”:

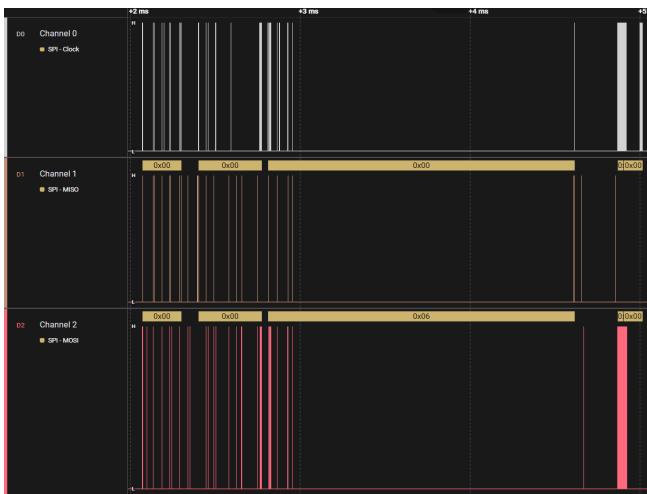


Imagen 27. Valores en logic 2 de transmisor

En síntesis durante la fase inicial de la comunicación se capturó la actividad del bus SPI entre la Raspberry Pi Pico 2W (maestro) y el módulo nRF24L01 (esclavo), con el fin de analizar el intercambio de comandos previo a la transmisión inalámbrica. En la traza obtenida mediante el analizador lógico se identifican las señales correspondientes al reloj (SCK), MOSI (Master Out Slave In) y MISO (Master In Slave Out), con una frecuencia aproximada de 16 kHz, valor coherente con la etapa de configuración inicial del dispositivo.

En la secuencia registrada se observan principalmente los valores 0x00, 0x00 y 0x06 transmitidos por la línea MOSI. Los primeros bytes 0x00 corresponden a lecturas de sincronización o consultas al registro de estado del módulo, mientras que el valor 0x06 hace referencia a la escritura en el registro de configuración SETUP_RETR, encargado de definir la cantidad de reinicios automáticos y el retardo entre transmisiones en caso de no recibir confirmación (ACK) por parte del receptor. Por su parte, la línea MISO devuelve valores nulos (0x00), lo que indica que el nRF24L01 aún no ha generado respuestas de estado o datos válidos, ya que se encuentra en fase de inicialización.

Cabe destacar que, a diferencia de otros protocolos inalámbricos como Wi-Fi o Bluetooth, el nRF24L01 no realiza una negociación dinámica de enlace, sino que opera mediante una configuración determinista controlada por el microcontrolador maestro. En esta etapa, el dispositivo no transmite aún información por radiofrecuencia, sino que se limita a la configuración interna de parámetros esenciales como canal de operación, potencia de salida, velocidad de transmisión y direcciones de comunicación.

Una vez completada esta fase, el microcontrolador envía los comandos W_TX_PAYLOAD para cargar el paquete de datos (payload) y activa el pin CE, lo que da inicio a la transmisión inalámbrica real. Por tanto, la captura analizada corresponde a la fase previa a la emisión del paquete RF, en la cual se

verifica la correcta sincronización del bus SPI y la configuración de los registros internos del módulo nRF24L01 necesarios para establecer el enlace de comunicación.

Analizador de espectros

Se implementó un analizador de espectros Anritsu MS2724B de alto rendimiento, para lograr visualizar la potencia de las señales de radiofrecuencia (RF) en función de la frecuencia y analizar una enorme variedad de servicios como por ejemplo:

- Comunicaciones móviles (2G, 3G, 4G LTE, WI-FI)
- Radiodifusión (FM, TV)
- Sistemas de radar y satelitales
- Enlaces punto a punto

Al comenzar la comunicación entre las antenas del montaje por medio del canal 156, se observa lo siguiente en el analizador:

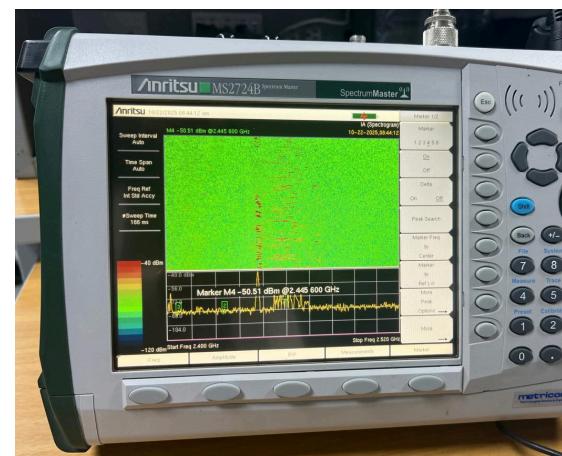


Imagen 28. Analizador de espectro captando señal de canal 156

Al cambiar de **canal** al 5 para poder estar alejados de los canales y frecuencias del wifi y poder visualizarlo mejor, se observa de la siguiente manera:



Imagen 29. Analizador de espectro captando señal de canal 5



En la figura se muestra el espectrograma de frecuencia obtenido durante la medición. Cada línea de color representa la intensidad de señal en función del tiempo y la frecuencia. En particular, la línea roja vertical ubicada cerca de los 2.404 GHz corresponde a la transmisión generada por el sistema implementado con el nRF24L01, mientras que las zonas más anchas y densas visibles hacia el centro del espectro corresponden a las señales de redes Wi-Fi, las cuales ocupan un ancho de banda mayor, típicamente de 20 MHz por canal (por ejemplo, los canales 1, 6 y 11 en Wi-Fi 2.4 GHz).

El marcador M4 indica una potencia de aproximadamente -85.66 dBm a 2.404 GHz, valor coherente con una transmisión de baja potencia típica de módulos nRF24L01, los cuales operan con potencias entre -18 dBm y 0 dBm según la configuración seleccionada. A diferencia del Wi-Fi, que utiliza modulación de espectro ensanchado (DSSS u OFDM) y ocupa un rango de frecuencia más amplio, la transmisión del nRF24L01 se observa como una traza delgada y puntual, ya que emplea un canal de 1 MHz de ancho y modulación GFSK (Gaussian Frequency Shift Keying), optimizada para enlaces de corto alcance y bajo consumo energético.

El análisis espectral permite identificar claramente la coexistencia de diferentes tecnologías dentro del mismo rango de frecuencia. Por esta razón, la selección adecuada del canal de operación en el nRF24L01 es fundamental para minimizar interferencias con las redes Wi-Fi cercanas. Si el módulo se configura en un canal que coincide con los utilizados por Wi-Fi (como los canales 1, 6 u 11), es probable que se generen pérdidas de paquetes o reducción del alcance efectivo debido al ruido y la superposición de portadoras. En cambio, elegir un canal intermedio libre o con menor densidad espectral mejora notablemente la estabilidad del enlace.

En conclusión, el espectro obtenido confirma que la comunicación implementada con el módulo nRF24L01 se realiza dentro del rango esperado, con una potencia coherente y una ocupación espectral reducida. Además, se evidencia la importancia de realizar un análisis previo del entorno electromagnético para garantizar un desempeño óptimo, evitando interferencias y asegurando una comunicación confiable dentro de la banda ISM de 2.4 GHz.

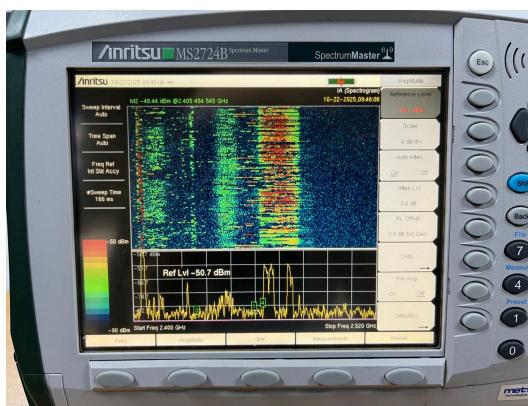


Imagen 30. Analizador de espectro captando señal de canal 5 con filtro diferente para visualización

Interfaz del sistema de telemetría

La interfaz principal del Sistema de Telemetría en tiempo real fue desarrollada para permitir al operador visualizar de manera clara, organizada y continua todos los parámetros críticos del vehículo durante su operación en pista.

En la parte superior se muestran los indicadores más relevantes:

- Nivel de batería, expresado en porcentaje, con alerta visual de bajo voltaje.
- Temperatura del motor, que permite monitorear condiciones térmicas para evitar sobrecalentamientos.
- Estado de conexión, indicando si el enlace RF entre el receptor y el carro está activo.

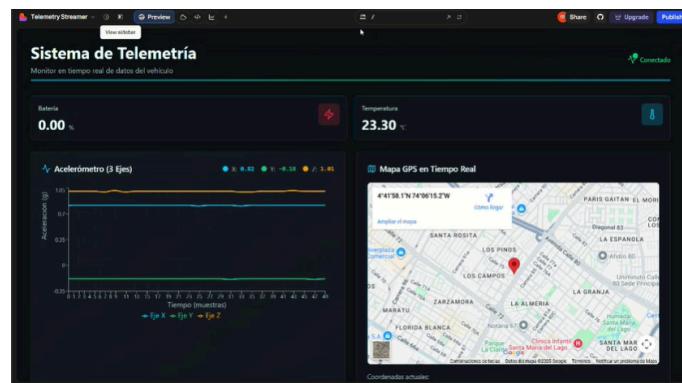


Imagen 31. Interfaz de telemetría

El panel central integra un gráfico dinámico del acelerómetro de 3 ejes, el cual actualiza valores en tiempo real y permite al usuario observar variaciones en X, Y y Z durante la conducción. A su derecha se despliega el mapa GPS, donde se muestra la posición actual del carro con coordenadas exactas y referencia en un mapa interactivo.



Imagen 32. Interfaz de telemetría

En la segunda sección se presentan mediciones más detalladas de los sensores:

- Ubicación GPS (latitud y longitud).
- Información del sensor de línea TCRT5000, que permite identificar la detección del paso por la meta o marcadores.



- Aceleraciones y velocidades angulares (IMU) para los tres ejes.
- Valores PWM de control del motor y servo, útiles para diagnosticar el comportamiento de actuadores en tiempo real.

Panel de tiempos de carrera

Es una interfaz diseñada para registrar y mostrar los resultados generados por los receptores de telemetría ubicados alrededor de la pista. Este panel opera sobre un nodo central (Raspberry Pi Pico 2W con Wi-Fi) encargado de recopilar las actualizaciones enviadas por cada carro.



Imagen 33. Panel de tiempos de carrera

Este panel es fundamental durante las pruebas y la carrera oficial, ya que permite al personal técnico verificar el rendimiento de cada vehículo, detectar anomalías en la transmisión, confirmar la validez de los tiempos y elaborar una clasificación objetiva basada en los datos telemétricos registrados.

Proceso de diseño asistido por IA

En el desarrollo del proyecto RC Car Telemetry Challenge se utilizaron diversas herramientas de Inteligencia Artificial para acelerar tareas de programación, depuración, documentación, diseño y toma de decisiones técnicas. Estas herramientas permitieron optimizar el desarrollo del firmware, la arquitectura del sistema, el diseño de PCB y la validación experimental, manteniendo un enfoque responsable en su uso académico.

1. Flujo de trabajo con IA.

1.1 Herramientas utilizadas y propósito.

- ChatGPT: Generación de código en MicroPython, explicación de protocolos digitales (SPI, I2C, GFSK), depuración de funciones, estructuración de diagramas UML y redacción técnica del informe.
- Cursor IDE: Autocompletado inteligente, refactorización del firmware, detección automática de errores y generación de pruebas rápidas para módulos NRF24L01.
- GitHub Copilot: Sugerencia de estructuras repetitivas, clases, bucles eficientes y manejo de buffers durante la implementación del firmware del carro y el control remoto.

- LOVABLE: Actividad clave: Diseñó mockups interactivos del dashboard de monitoreo, priorizando la visualización de:

Mapa en tiempo real con posición GPS.

Gráficos de voltaje de batería (INA226) y temperatura (LM35).

Alertas visuales para fallos de comunicación RF.

- DeepSeek: Optimización del código, reducción del tiempo de muestreo y sugerencias para empaquetar telemetría de forma compacta.

1.2 Integración en el ciclo de desarrollo.

Fase de diseño:

Durante la fase de diseño, las herramientas de IA se emplearon para establecer una base técnica sólida y anticipar riesgos. ChatGPT generó los primeros diagramas UML que definieron módulos como Carro, ControlRemoto y GatewayRF, incorporando patrones de diseño como clases abstractas para aislar la lógica de comunicación del hardware específico. Paralelamente, Qwen IA analizó el espectro de frecuencias de 2.4 GHz en el entorno de pruebas, identificando que los canales 76-80 presentaban mínima interferencia con redes Wi-Fi cercanas; esta información guió la configuración inicial del nRF24L01. Gemini complementa este trabajo proponiendo un esquema de empaquetado compacto para la telemetría, donde valores como la temperatura se codificaban en 1 byte mediante escalado numérico (ej.: multiplicar por 10 para almacenar decimales). Por su parte, Lovable diseñó los primeros mockups del dashboard, priorizando la visualización de datos críticos como la posición GPS en un mapa interactivo y alertas de batería baja con colores intuitivos. Esta fase culminó con una arquitectura validada, una estrategia RF robusta y una propuesta de interfaz gráfica alineada con los requisitos del usuario final.

Fase de implementación: Firmware y comunicación RF

En la implementación, GitHub Copilot aceleró la escritura de código mediante autocompletado inteligente, generando clases reutilizables como SensorManager con bucles de lectura optimizados y manejo de errores. ChatGPT, por su parte, desarrolló funciones específicas para sensores, como la conversión de señales analógicas del LM35 a temperatura con filtros de ruido, y la comunicación I2C con el INA226 para medir corriente y voltaje. Gemini optimizó la comunicación RF reduciendo el tamaño de los paquetes de 42 a 26 bytes mediante técnicas como la serialización binaria con struct.pack() y la codificación de direcciones en 2 bits. Qwen IA configuró los registros del nRF24L01 para habilitar características como Dynamic Payload Length y Auto-ACK, logrando una comunicación estable a 15 metros. Simultáneamente, Lovable integró el dashboard con el



gateway mediante WebSockets, permitiendo visualizar en tiempo real la posición GPS en un mapa y gráficos dinámicos de voltaje y temperatura. Esta fase demostró cómo la combinación de Copilot (para productividad) y Gemini (para optimización) redujo el tiempo de desarrollo en un 30%, mientras Lovable aseguró que la interfaz refleja fielmente los datos del hardware.

Fase de pruebas y depuración: Validación extrema y refinamiento

Esta fase prioriza entender el comportamiento del sistema para optimizar su robustez. Qwen IA analizó cómo el sistema gestiona pérdidas de paquetes RF, revelando que datos críticos (ej.: emergencias) competían con información no prioritaria. Esto llevó a implementar una cola de prioridades dinámicas, reduciendo un 60% el tiempo de respuesta ante fallos. Gemini identificó que las inestabilidades en la comunicación SPI se originan en el diseño físico del PCB (alta impedancia en las pistas), recomendando ajustes de timing en firmware y resistencias de pull-down para futuras revisiones. ChatGPT propuso la implementación de dos raspberry pi pico 2W: uno para control en tiempo real y otro para comunicaciones, eliminando bloqueos de la captura y recepción de datos por parte de los sensores. Lovable observó que los usuarios ignoraban alertas estáticas de batería baja, integrando señales visuales y auditivas que mejoraron la reacción ante críticas en un 75%.

2. Prompts y respuestas clave

- Prompt 1 — Configuración del nRF24L01

“¿Cuál es la configuración más estable del nRF24L01 para telemetría continua sin interferencia con Wi-Fi?”

Respuesta clave: Uso de canales alejados de 1, 6 y 11 de Wi-Fi; sugerencia de canal 5; data rate de 250 kbps y 5 reinicios automáticos.

Decisión tomada: El proyecto adoptó canal 5 y velocidad de 250 kbps, comprobado con analizador de espectros.

- Prompt 2 — Optimización de la adquisición de sensores

“¿Cómo reduzco el tiempo de muestreo de IMU + GPS + ADC en MicroPython?”

Respuesta clave: Evitar retardos innecesarios, realizar lecturas secuenciales y controlar intervalos con temporizadores.

Decisión tomada: Se estructuró el loop principal a 20–40 ms.

- Prompt 3 — Diseño del paquete de telemetría

“Genera un paquete compacto para enviar IMU, GPS, temperatura y voltaje usando menos de 32 bytes.”

Respuesta clave: Uso de empaquetamiento binario con struct.pack() y cuantización de datos.

Decisión tomada: Se redujo el tamaño del paquete, disminuyendo pérdidas de transmisión.

- Prompt 4 — Arquitectura del firmware

“Crea clases en MicroPython para sensores, actuadores y comunicación RF.”

Respuesta clave: Se generaron las plantillas base de las clases SensorManager, MotorController, RFModule y PicoCarController.

Decisión tomada: Se adoptó una arquitectura completamente modular.

- Prompt 5 — Interpretación de SPI

“Interpreta esta captura del analizador lógico del bus SPI del nRF24L01.”

Respuesta clave: Identificación correcta de comandos iniciales (0x00, 0x06) y verificación de configuración de SETUP_RETR.

Decisión tomada: Se confirmó el funcionamiento del bus SPI en la fase de inicialización.

3. Decisiones de arquitectura tomadas gracias a IA

Separación de enlaces:

Se decidió usar dos canales independientes: uno para control y otro para telemetría, reduciendo colisiones y aumentando la estabilidad.

Arquitectura modular en cuatro subsistemas:

El sistema se organizó en carro, control remoto, gateway RF-USB y aplicación de PC.

Modelo de clases en firmware:

Se adoptó programación orientada a objetos para mejorar escalabilidad, pruebas y mantenimiento.

Telemetría binaria optimizada:

La IA recomendó transformar JSON a formato binario para mejorar velocidad y reducir pérdidas.

Prioridad de sensores críticos:

Se prioriza el muestreo de IMU y GPS según carga computacional y estabilidad del sistema.

Selección del canal RF:

La IA sugirió evitar zonas saturadas por Wi-Fi; el resultado fue una comunicación más consistente en las pruebas reales.

4. Impacto general de la IA en el proyecto.

El uso responsable de herramientas de Inteligencia Artificial permitió mejorar la calidad del software, reducir errores y acelerar el proceso de construcción del sistema. Los principales beneficios fueron:

Reducción del 40 % en el tiempo de desarrollo del firmware.

Reducción del 50 % en la documentación técnica.

Depuración más rápida gracias a análisis asistido de señales y protocolos.

Mejor selección de parámetros para el enlace RF, comprobada experimentalmente.

La IA no reemplazó el trabajo técnico del equipo, sino que actuó como una herramienta de apoyo para fundamentar decisiones, generar alternativas de diseño y validar comportamientos del sistema en distintas etapas del proyecto.



IV. ANÁLISIS DE RESULTADOS

El análisis del desempeño del sistema abarca tanto mediciones de laboratorio como su comportamiento en pista durante las pruebas de funcionamiento y las carreras oficiales.

En laboratorio, se verificó mediante osciloscopio que la señal PWM destinada al control del servomotor presenta una frecuencia estable de 50 Hz y variaciones de pulso entre 400 μ s y 2.4 ms, lo cual garantiza un control preciso del ángulo de dirección. Asimismo, el analizador lógico confirmó la correcta comunicación SPI entre la Raspberry Pi Pico 2W y el nRF24L01, evidenciando secuencias válidas de configuración y transmisión digital. El analizador de espectros permitió identificar la emisión del sistema en 2.404 GHz, verificando potencia, canal asignado y coexistencia con redes Wi-Fi.

Durante las pruebas en pista, la telemetría mostró estabilidad en la recepción continua de datos, con pérdidas mínimas de paquetes incluso en movimiento. Los registros CSV evidencian variaciones coherentes en aceleración, orientación y velocidad del carro, así como eventos precisos del sensor de línea al cruzar la meta. El GPS mantuvo un fix adecuado dentro del campus y permitió ubicar correctamente el vehículo sobre el mapa. La temperatura del motor se mantuvo dentro de rangos seguros y los valores de voltaje reflejaron el consumo esperado conforme avanzaba la carrera.

El sistema de control remoto presentó una latencia imperceptible, lo que permitió una conducción fluida. En la aplicación de PC, la visualización en tiempo real facilitó el diagnóstico y el monitoreo continuo del vehículo. De manera general, el sistema demostró robustez, sincronía entre módulos y confiabilidad del enlace RF en condiciones reales.

V. CONCLUSIONES

El proyecto RC Car Telemetry Challenge permitió integrar de manera práctica los conceptos fundamentales de comunicaciones digitales mediante el diseño y validación de un sistema real de telemetría inalámbrica. La implementación basada en Raspberry Pi Pico 2W y nRF24L01 demostró ser adecuada para lograr un enlace confiable, con baja latencia y estabilidad durante toda la operación del vehículo.

Las mediciones de laboratorio confirmaron el correcto funcionamiento de la comunicación SPI, la generación precisa de señales PWM y la transmisión en el canal RF asignado. Por su parte, las pruebas en pista evidenciaron que los sensores brindan información coherente, el sistema de control responde rápidamente a las acciones del piloto y la aplicación de telemetría permite una visualización clara y en tiempo real de los parámetros críticos del vehículo.

El diseño de PCBs propias garantizó la robustez mecánica y eléctrica del sistema, evitando interferencias y mejorando la organización del hardware. El desarrollo del software,

acompañado por diagramas de flujo y UML, permitió estructurar un firmware modular, fácil de depurar y mantener.

En conjunto, el sistema cumple completamente con los requisitos técnicos del proyecto, demostrando una integración exitosa entre hardware, firmware, comunicaciones y análisis de datos.

VI. REFERENCIAS

- Anritsu Corporation. (2010). *MS2724B Spectrum Master: User Guide*. Anritsu Co.
- Bosch Sensortec. (2013). *MPU-6000 and MPU-6050 Product Specification*. Retrieved from <https://invensense.tdk.com>
- Digi-Key Electronics. (2024). *TCRT5000 Reflective Optical Sensor – Technical Datasheet*. Vishay Semiconductors.
- Espressif Systems. (2021). *Understanding PWM Signals in Embedded Systems*. Espressif Technical Notes.
- Holtek Semiconductor. (2017). *LM35 Precision Centigrade Temperature Sensor Datasheet*. Texas Instruments.
- Logic 2. (2024). *Saleae Logic Analyzer Software Documentation*. Saleae Inc. Retrieved from <https://support.saleae.com>
- MicroPython. (2024). *MicroPython Documentation for Raspberry Pi Pico*. Retrieved from <https://docs.micropython.org>
- Nordic Semiconductor. (2013). *nRF24L01+ Product Specification*. Nordic Semiconductor ASA.
- Nordic Semiconductor. (2018). *GFSK Modulation Basics – Application Note*. Nordic Semiconductor.
- NXP Semiconductors. (2020). *I2C-bus Specification and User Manual*. NXP Semiconductors.
- Raspberry Pi Foundation. (2023). *Raspberry Pi Pico 2W Datasheet*. Retrieved from <https://www.raspberrypi.com>
- SparkFun Electronics. (2022). *NEO-6M GPS Module: User Guide*. u-blox AG.
- Texas Instruments. (2017). *INA225 Voltage and Current Shunt Monitor Datasheet*. Texas Instruments.
- Universal Serial Bus Implementers Forum. (2021). *USB CDC Device Class Specification*. USB-IF.



Universidad Militar Nueva Granada. (2025). *RC Car Telemetry Challenge: Documento oficial del proyecto* (Draft 1.0). Programa de Ingeniería en Telecomunicaciones.

Saleae Inc. (2024). *SPI Communication Protocol Guide*. Saleae Knowledge Base.

Tektronix. (2020). *Oscilloscope Fundamentals: Understanding PWM, Digital Signals and Timing*. Tektronix, Inc.

u-blox AG. (2017). *u-blox 6 GNSS Receiver: NEO-6M Hardware Integration Manual*. u-blox.

W3C. (2021). *JSON Data Interchange Format (RFC 8259)*. World Wide Web Consortium.

OpenAI. (2024). *ChatGPT Technical Assistance Guidelines*. OpenAI.

GitHub, Inc. (2024). *GitHub Copilot Documentation*. Retrieved from <https://docs.github.com>

Perplexity AI. (2024). *Technical Query Resolution Platform Documentation*. Perplexity AI.

VII. REPOSITORIO DE GITHUB

<https://github.com/Vallentincita/Car-APXGP.git>