

Sistema de Telemetría Inalámbrica para Vehículo RC Usando Raspberry Pi Pico 2W y NRF24L01

Angie Ximena Diaz Bejarano
Tomas Alejandro Ramirez Velasquez
Fernando Javier Riaño Rios
est.tomas.ramirez@unimilitar.edu.co
est.fernando.riano@unimilitar.edu.co
est.angie.xdiaz@unimilitar.edu.co
Docente: José De Jesús Rugeles

garantizando robustez, orden y funcionamiento estable durante la competencia.

El sistema final incluyó un panel web de telemetría que permitió visualizar en tiempo real datos del GPS, acelerómetro, giroscopio, temperatura, voltaje de batería, actuadores y sensor de línea, convirtiendo el carro en una plataforma completa de monitoreo. Este informe presenta el desarrollo, integración y resultados obtenidos.

Resumen: El proyecto desarrollado por el equipo “A los Pits” consistió en la construcción completa de un vehículo RC con control inalámbrico y sistema de telemetría integrado usando Raspberry Pi Pico 2W, módulos NRF24L01+, sensores de aceleración, temperatura, GPS, seguidor de línea y lectura de voltaje.

Se realizaron pruebas experimentales iniciales para validar la comunicación RF, señales PWM, protocolos SPI, UART e I2C antes de proceder a la integración mecánica del carro. Posteriormente, se construyeron tres placas principales (TX1, RX1 y TX2) y su estación receptora RX2, todas soldadas en baquelita.

El vehículo fue capaz de enviar información en tiempo real hacia un panel web local, mostrando velocidad, giros, coordenadas GPS, voltaje de batería, temperatura y estado del sensor de línea.

En competencia, el carro demostró estabilidad, rapidez y fiabilidad, logrando el primer lugar tanto en contrarreloj como en la carrera principal. El sistema de telemetría funcionó durante toda la prueba, permitiendo monitoreo continuo del desempeño del vehículo.

INTRODUCCIÓN

El desarrollo de vehículos RC controlados de manera inalámbrica constituye una aplicación directa de sistemas embebidos, comunicaciones digitales y adquisición de datos. En este contexto, el equipo “A los Pits” diseñó e implementó un sistema completo que integra control remoto por radiofrecuencia y telemetría en tiempo real para un vehículo RC basado en la Raspberry Pi Pico 2W.

El proyecto incluyó pruebas experimentales iniciales para validar los subsistemas esenciales: comunicación RF mediante NRF24L01+, generación de PWM para actuadores, lectura de sensores I2C, UART y ADC, además del análisis del espectro RF para seleccionar canales de operación estables.

Posteriormente, el vehículo se modificó mecánicamente para integrar un servomotor SG90, un ESC para el motor de tracción y sensores adicionales. Cada módulo electrónico fue ensamblado primero en protoboard y luego soldado en baquelita,

DESARROLLO EXPERIMENTO

- SISTEMA INALAMBRICO NRF24L01

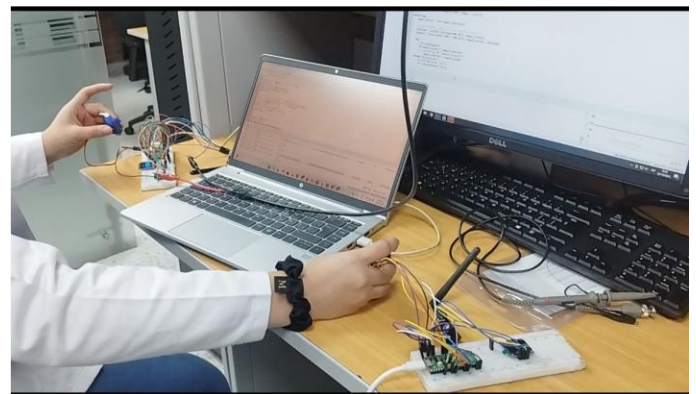


Imagen de lab. 1. configuración inalámbrica para primeras pruebas

Antes de iniciar la modificación del vehículo RC y la construcción de las placas electrónicas, se desarrolló una fase de pruebas experimentales cuyo objetivo principal fue verificar el correcto funcionamiento de los elementos fundamentales del sistema: comunicación inalámbrica, generación de señales PWM, lectura de sensores y estabilidad de los protocolos de comunicación. Estas pruebas garantizaron que la base tecnológica estuviera completamente operativa antes de avanzar a la integración mecánica y electrónica del carro. En este caso se evaluaron los siguientes aspectos: Funcionamiento del enlace RF entre dos Raspberry Pi Pico 2W usando NRF24L01+, señal PWM para el control de servomotor, lectura y estabilidad del bus I2C del sensor MPU6050, y verificación del espectro de transmisión en la banda ISM de 2.4 GHz.

Inicialmente se verifica el enlace inalámbrico con la antena NRF24L01. Se construyó un enlace básico TX-RX utilizando dos Raspberry Pi Pico 2W conectadas a módulos NRF24L01+. El propósito era asegurar:

- Estabilidad del canal de comunicación.
- Envío y recepción de datos sin pérdidas.
- Funcionamiento del protocolo SPI entre la Pico y el módulo RF.

- Tiempo de respuesta adecuado para control en tiempo real.

Durante la prueba:

Se configuró un canal de comunicación inicial.

Se enviaron valores de un joystick simulado para verificar movilidad del servo.

Se trabajó con payloads pequeños para garantizar baja latencia.

El enlace respondió correctamente, sin pérdidas de paquetes y con una latencia imperceptible, demostrando que el nRF24L01+ era apto para el control de movimiento del carro.

Ahora se analiza el PWM del servomotor. El control de dirección del vehículo depende totalmente de la estabilidad del PWM, por lo que se utilizaron un osciloscopio y la Raspberry Pi Pico para verificar:

- Frecuencia del PWM.
- Amplitud de la señal.
- Tiempo de pulso según el ángulo del servo.

Los resultados fueron:

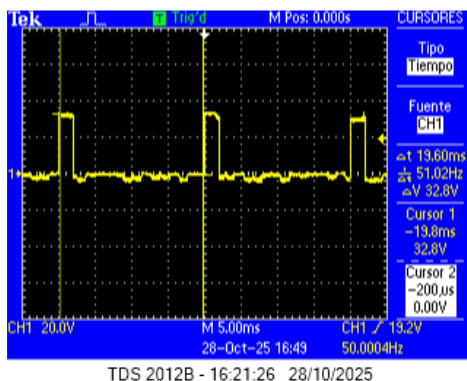


Imagen de lab. 2. pulsos (2,5ms) servomotor desde el osciloscopio

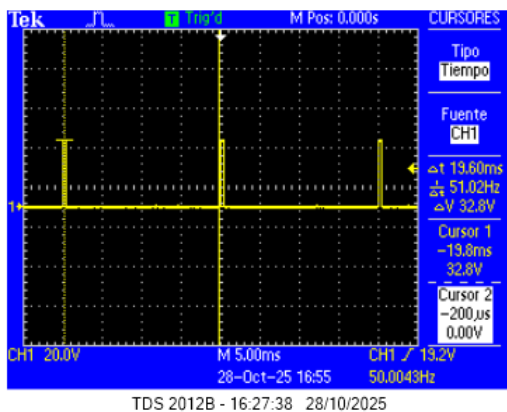


Imagen de lab. 3. segundo pulso visto desde el osciloscopio

- Frecuencia estable entre 50 y 51 Hz.
- Amplitud aproximada de 3.2 V, adecuada para el control del servo.

Anchos de pulso medidos:

- ≈ 0.5 ms \rightarrow servo completamente a la izquierda
- ≈ 1.5 ms \rightarrow posición neutra (centro)
- ≈ 2.5 ms \rightarrow servo completamente a la derecha

No se observaron picos, ruido, ni distorsiones. La señal era limpia y suficientemente estable para la operación continua del servo en el vehículo final.

Por consiguiente, se hace una evaluación del bus I2C con el sensor MPU6050.

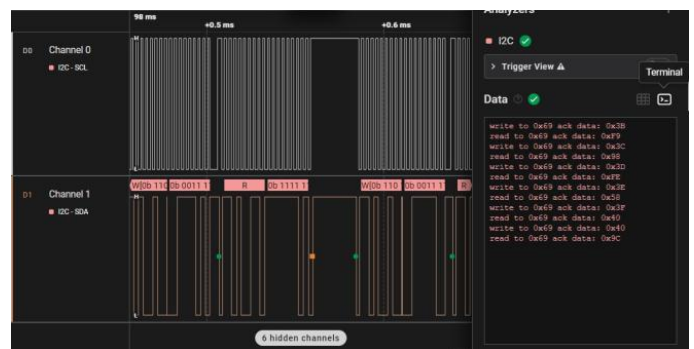


Imagen de lab. 4. captura de trama I2c con analyzer (LOGIC)

Resultados observados:

- Dirección del dispositivo: 0x69.
- Frecuencia del bus: aprox. 400 kHz (modo Fast I²C).
- Todas las operaciones finalizaron con ACK, confirmando integridad en la transmisión.
- Se observaron lecturas correctas de aceleraciones en los registros 0x3B a 0x40.

Esto permitió confirmar que la lectura de la IMU era estable y adecuada para ser transmitida posteriormente en la telemetría del carro.

Se utilizó un analizador de espectros para visualizar la señal transmitida por el módulo nRF24L01+. Esta prueba buscaba garantizar que la frecuencia seleccionada fuese limpia y que el módulo no estuviera generando ruido excesivo o armónicos indeseados.

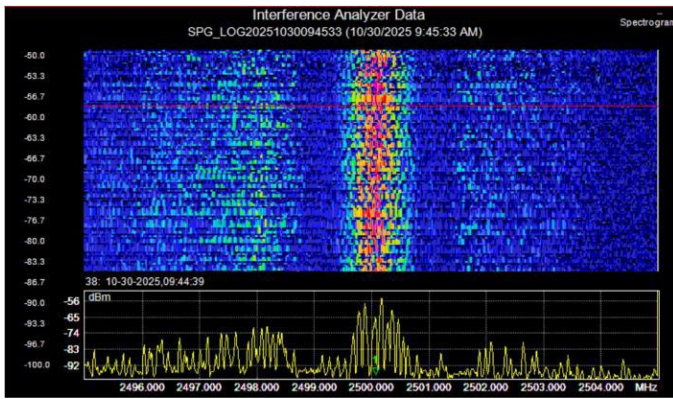


Imagen de lab. 5. Visualización del canal con espectrograma

Los resultados mostraron:

- Señal centrada en la banda ISM (aprox. 2.4 a 2.5 GHz).
- Ancho de canal aproximado de 1 MHz, característico de la modulación GFSK usada por el módulo.
- Señal estable sin fluctuaciones significativas.

Nivel de potencia coherente con la distancia utilizada en pruebas de laboratorio.

Esta prueba permitió posteriormente elegir correctamente los canales definitivos (76 para telemetría y 100 para control).

CONSTRUCCIÓN DEL CARRO

Después de validar el funcionamiento de los elementos electrónicos fundamentales comunicación RF, señales PWM, lectura de sensores y análisis espectral se procedió a adquirir un vehículo a control remoto comercial que sirviera como base mecánica para el proyecto.

El modelo seleccionado fue un carro 4x4 tipo “Max Off-Road Climbing”, de estructura ligera, suspensión independiente y espacio interno suficiente para alojar la electrónica necesaria.



Imagen de lab. 6. configuración del carro inicial

Antes de modificar físicamente el carro, todo el sistema de control fue montado y probado primero en protoboard.

- Esto es fundamental en ingeniería embebida porque:
- Se detectan errores de conexión sin dañar elementos del carro.

- Se verifican niveles de voltaje reales.
- Se comprueba el comportamiento del servo y del ESC antes de fijarlos al chasis.
- Se confirma que los módulos nRF24L01 funcionan con la antena y con la Pico 2W en condiciones reales.

Se procede a intervenir directamente el vehículo. Para ello fue necesario desmontar completamente la carrocería y retirar la electrónica original, dejando expuestos los componentes mecánicos del sistema de dirección y el motor DC del vehículo. Esta etapa permitió adaptar tanto el servomotor encargado del giro como el módulo ESC responsable del control del motor de tracción.

El carro “Guido” (así se apodo) fue desmontado retirando primero la carrocería, luego la cubierta interna y posteriormente los elementos plásticos que sostenían el sistema original de dirección. Una vez expuesta la zona delantera del chasis, se identificó el punto mecánico donde debía integrarse el nuevo servomotor. Se instaló un servomotor SG90, fijándolo a la estructura mediante un hilo cañamo y soportes tipo resorte modificados. Esta intervención requirió:

- Centrar físicamente el brazo del servo,
- Alinear el eje del servo con la barra de dirección del carro,
- Eliminar el juego mecánico del mecanismo original,
- Ajustar los límites de giro para evitar tensiones excesivas en los engranajes.

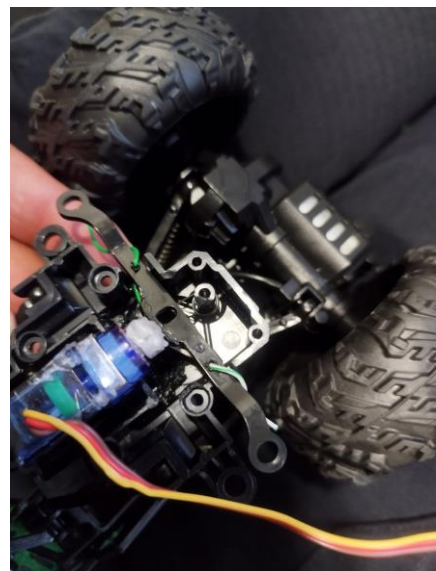


Imagen de lab. 7. Acople del servomotor al carro

Con estas modificaciones, el servomotor quedó vinculado directamente a las ruedas delanteras, garantizando que el control PWM emitido desde la Raspberry Pi Pico 2W se traduzca en un giro preciso y estable.

El motor DC original del carro fue desconectado y preparado para utilizar un ESC

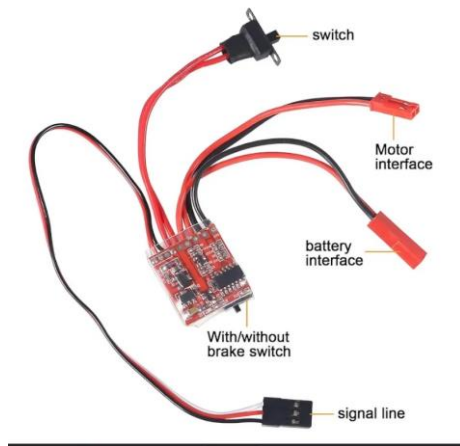


Ilustración 1. ESC

¿Para qué funciona este módulo ESC?

El ESC es un controlador electrónico diseñado para:

- regular la velocidad del motor DC mediante señal PWM,
- proporcionar avance, frenado y reversa,
- manejar corrientes elevadas provenientes de la batería,
- proteger al motor de sobrecargas,
- mantener estable la tracción incluso bajo variaciones de voltaje.

Este módulo posee cuatro interfaces principales:

- Motor interface: salida hacia el motor DC.
- Battery interface: entrada directa de la batería de 7.4V 2.4A.
- Signal line: entrada de la señal PWM proveniente de la Raspberry Pi Pico.
- Switch: interruptor general para encender y apagar la potencia del sistema

Una vez desmontado el vehículo RC y ajustado el sistema de dirección para instalar el servomotor, se procedió a realizar el primer montaje electrónico funcional sobre una protoboard. Este montaje permitió verificar el comportamiento real del servomotor, el motor de tracción y la comunicación inalámbrica antes de soldar los componentes en baquelita e instalarlos

definitivamente en el chasis del carro.

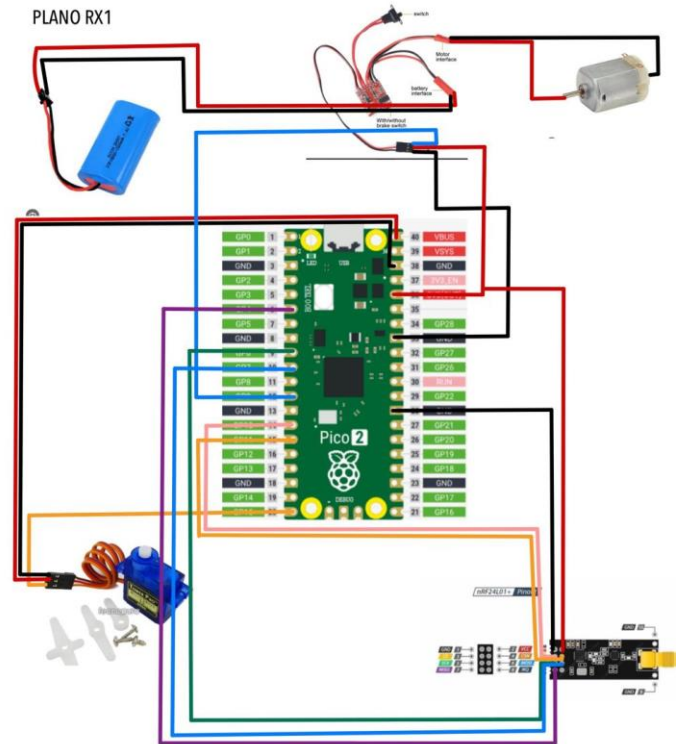


Ilustración 2. conexión a raspberry pico 2

Para esta fase se empleó una Raspberry Pi Pico 2W como microcontrolador principal, ubicada en el centro de la protoboard, desde donde se distribuyeron las señales hacia los actuadores y los módulos de comunicación.

El primer componente conectado fue el servomotor, encargado del giro de las ruedas delanteras. Este elemento es esencial en el proyecto, ya que reemplaza el mecanismo de dirección original del carro por un sistema más preciso y controlado mediante PWM. El servomotor se acopló físicamente a la barra de dirección en la etapa previa, por lo que en esta fase se verificó únicamente su respuesta electrónica. Se comprobó que, al variar el ancho del pulso PWM generado por la Pico, el servo respondía linealmente entre sus límites izquierdo y derecho.

Posteriormente, se integró el módulo ESC (Electronic Speed Controller), el cual permite controlar la velocidad del motor DC del carro. Este módulo recibe señal PWM desde la Raspberry Pi Pico 2W y, en función de su valor, regula la potencia entregada al motor, permitiendo aceleración progresiva, frenado y reversa. El ESC además recibe directamente la alimentación de la batería de 7.4 V – 2.4 A, actuando como un intermediario seguro entre el microcontrolador y el motor, evitando sobrecargas en la Raspberry.

Con ambos actuadores conectados, se añadió el módulo de comunicación nRF24L01, encargado de recibir los comandos provenientes del control remoto. Este módulo se montó también sobre la protoboard y se vinculó por protocolo SPI a la Pico. Para

monitorear su estado se incorporó un LED indicador, que permanece encendido cuando el enlace inalámbrico es estable y parpadea cuando el módulo está buscando conexión o la señal se interrumpe. Esta retroalimentación visual facilitó las pruebas de sincronización entre el transmisor y el receptor. Finalmente, el sistema fue alimentado mediante la batería de 7.4 V, utilizada para el motor y el servo, mientras que la Pico se energizó por separado durante las pruebas iniciales a fin de evitar reinicios por caída de voltaje. Las conexiones se muestran en estas tablas:

Tabla 1. conexión servo

SERVO MOTOR SG90		
COMPONENTE	GPIO	PIN FISICO
VCC	VBUS	40
GND	GND	38
PWM	GPIO 15	20

NRF24L01		
COMPONENTE	GPIO	PIN FISICO
VCC	3V3	36
GND	GND	30
CE	GPIO 10	14
CSN	GPIO 11	15
SCK	GPIO 6	9
MOSI	GPIO 7	10
MISO	GPIO 4	6

Tabla 2. conexión modulo NRF

Después de finalizar el montaje de la placa RX1 en protoboard, se procedió a ensamblar la placa TX1, la cual corresponde al control remoto inalámbrico del vehículo. Esta placa es fundamental, ya que desde ella se generan todos los comandos de dirección y aceleración que el carro recibe a través del módulo nRF24L01.

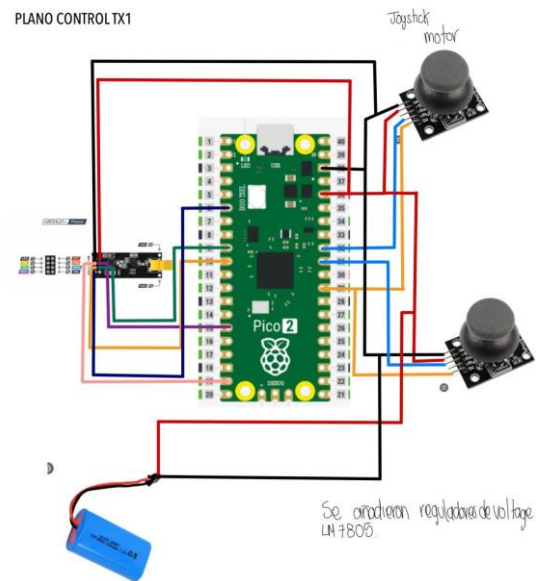


Ilustración 3. segunda conexión de raspberry

El diseño del control remoto incluyó dos joysticks analógicos, un módulo nRF24L01 configurado en transmisión continua, una Raspberry Pi Pico 2W y una batería de 8.4 V con reguladores lineales LM7805 para estabilizar las tensiones de operación. Todo el sistema se ensambló inicialmente en una protoboard para validar el comportamiento antes de fijarlo en baquelita.

La Raspberry Pi Pico 2W se ubicó al centro de la protoboard para facilitar el acceso a los GPIO y distribuir las señales hacia los joysticks y el módulo de comunicación. Esta placa actuó como transmisor principal, encargada de: leer los valores analógicos de los dos joysticks, procesarlos para convertirlos en posiciones normalizadas, empacarlos en una trama compacta, transmitirlos mediante el módulo nRF24L01 en el canal 100, enviar señales estables con baja latencia al receptor montado en el vehículo.

Se utilizaron dos joysticks tipo KY-023, cada uno con dos ejes analógicos (X e Y) y un pulsador central. En este proyecto se emplearon los ejes Y de cada joystick para representar:

Joystick 1: dirección del vehículo (giro izquierda – centro – derecha).

Joystick 2: aceleración y reversa del motor (avance – freno – reversa).

Los joysticks se conectaron a los pines ADC de la Pico, permitiendo leer los voltajes variables derivados del movimiento del mando.

La conexión realizada fue:

VCC del joystick → 5V regulados por LM7805

GND → referencia común

VRx / VRy → GPIO configurados como ADC (según tu plano)

SW (botón) → no usado en esta etapa
El módulo nRF24L01 fue conectado nuevamente utilizando el bus SPI0 alterno de la Raspberry Pi Pico 2W. Este módulo funciona como transmisor (TX1), enviando de manera periódica un paquete de datos que contiene: posición del joystick de dirección, posición del joystick del motor, valores normalizados entre 0 y 255.

Las conexiones realizadas fueron:

VCC → 3.3V (pin 36)

GND → pin 38

CE → GP14

CSN → GP15

SCK → GP6

MOSI → GP7

MISO → GP4

El nRF24L01 fue configurado en potencia máxima y velocidad de 2 Mbps para minimizar la latencia entre movimiento del joystick y respuesta del vehículo.

La placa TX1 se alimentó mediante una batería de 8.4 V (2 celdas Li-ion), lo cual requirió el uso de reguladores de voltaje LM7805 para reducir los 8.4 V a 5 V, suministrando alimentación estable a: los dos joysticks, parte del circuito de referencia, y garantizando aislamiento eléctrico entre los módulos

Imagen de lab. 8 primeras tomas del carro

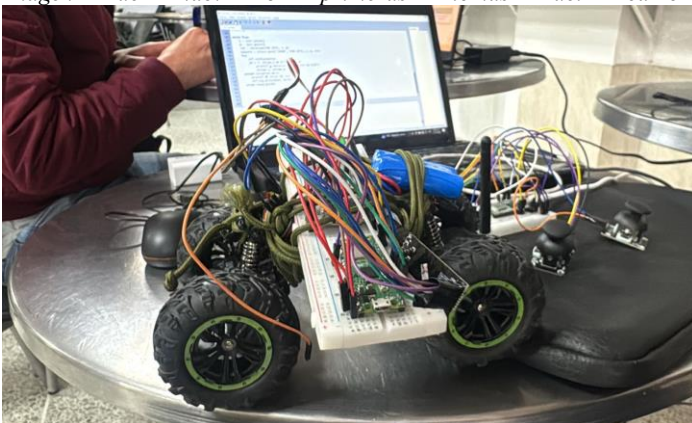
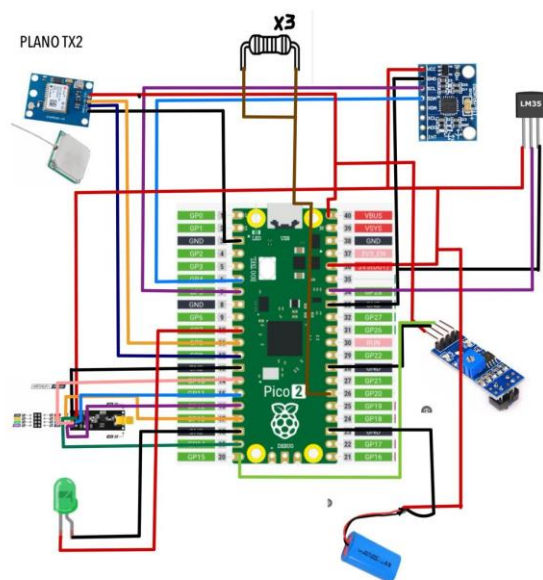


Imagen de lab. 9. carro y prototipo de control remoto

La placa TX2 corresponde al módulo encargado de obtener, procesar y transmitir toda la información de telemetría del vehículo hacia la estación de monitoreo. Esta placa convierte al carro en un sistema inteligente capaz de reportar su posición GPS, aceleración, temperatura, voltaje de batería y frecuencia de giro, entre otras señales internas relevantes para el monitoreo en tiempo real.



TX2 fue ensamblada en una segunda protoboard independiente, pero instalada físicamente dentro del vehículo, trabajando junto al módulo RX1. Su función principal es capturar los datos de los

sensores y enviarlos inalámbricamente mediante un segundo módulo nRF24L01, operando de manera paralela pero independiente del canal usado para el control remoto.

La Raspberry Pi Pico 2W actúa como cerebro del sistema de telemetría. En esta placa se configuraron: entradas analógicas (ADC) para medir temperatura y voltaje, entradas digitales para sensores de pulsos, comunicación I2C para acelerómetro y giroscopio, UART para el módulo GPS, comunicación SPI para el módulo nRF24L01 que transmite la telemetría. Toda la lectura y conversión de señales se realiza localmente en la Pico, que organiza la información en un paquete compacto transmitido periódicamente.

Se integró un GPS NEO-6M, encargado de entregar la ubicación exacta del vehículo (latitud, longitud y altitud) y la velocidad aproximada del desplazamiento. El GPS se conectó usando UART, debido a que transmite datos NMEA a 9600 baudios, compatibles con la Raspberry.

GPS		
CONEXIÓN	PIN FISICO	GPIO
VCC	40	VBUS
GND	3	GND
TX	12	GPIO9
RX	11	GPIO8

El MPU6050 proporciona información de: aceleración en los tres ejes (X, Y, Z), velocidad angular, inclinación del vehículo durante movimiento. Se conectó mediante I2C, usando los pines oficiales de la Pico. Este sensor permite medir vibraciones, impactos y estabilidad del vehículo mientras se desplaza.

MPU		
CONEXIÓN	PIN FISICO	GPIO
VCC	36	3,3V OUT
GND	33	GND
SCL	7	GPIO5
SDA	6	GPIO4

Para medir la temperatura interna del carro, se conectó un sensor LM35, que entrega una salida analógica proporcional a la temperatura en °C. La lectura se usa para monitorear sobrecalentamientos del motor

LM35		
CONEXIÓN	PIN FISICO	GPIO
VCC	36	3,3V OUT
GND	33	GND
OUT	34	GPIO28

Para medir el nivel de la batería Li-ion se usó un módulo

En la zona superior del circuito TX2 se instalaron tres resistencias configuradas como un divisor de tensión, cuya

función es reducir el voltaje de la batería Li-ion (7.4–8.4 V) a un nivel seguro para ser leído por la entrada ADC de la Raspberry Pi Pico 2W (máximo 3.3 V).

Este divisor de tensión permite: monitorear en tiempo real el estado de la batería, evitar descargas profundas, generar alertas o apagar el sistema si el voltaje baja demasiado. Conexiones observadas en el plano:

Entrada del divisor → positivo de la batería.

Salida del divisor → GPIO GP26 (ADC0) de la Pico.

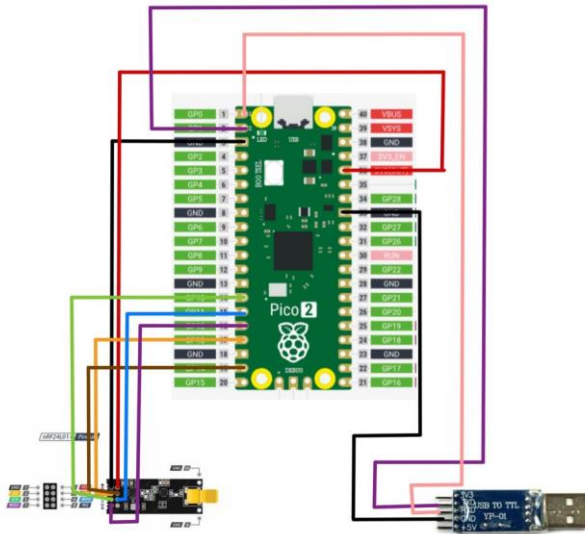
GND del divisor → tierra común.

En la parte derecha del esquemático se encuentra instalado el sensor de línea TCRT5000, encargado de detectar superficies claras u oscuras mediante un emisor IR y un fototransistor integrado. Este sensor se usa específicamente para: detectar cuándo el vehículo pasa sobre la línea negra en el suelo, identificar el paso por meta, enviar esa información a la estación de telemetría, registrar tiempos o eventos según el recorrido.

SENSOR DE LINEA		
CONEXIÓN	PIN FISICO	GPIO
VCC	40	VBUS
GND	28	GND
DO	20	GPIO20

Ahora bien, hablemos finalmente de la placa RX2, esta corresponde a la estación receptora de telemetría. Esta placa es la encargada de recibir, decodificar y retransmitir hacia el computador toda la información enviada por la placa TX2 instalada en el vehículo. Su función es esencial, ya que actúa como un puente entre el sistema inalámbrico del carro y la interfaz gráfica de telemetría desarrollada en la computadora. Para realizar esta retransmisión hacia el computador se utiliza comunicación serial basada en el protocolo RS-232, implementado mediante un módulo USB–TTL que convierte las señales UART de la Raspberry Pi Pico en un puerto COM virtual compatible con los niveles y tramas del estándar RS-232. Gracias a este adaptador, la telemetría puede visualizarse en tiempo real en el software de monitoreo.

PLANO RX2



El nRF24L01 es el componente clave que permite recibir todas las mediciones enviadas por la placa TX2 en el carro.

En RX2 se configuró en modo receptor, utilizando el mismo canal y dirección de comunicación definidos para la telemetría (canal 76).

NRF24L01		
CONEXIÓN	GPIO	PIN FISICO
VCC	3V3	36
GND	GND	3
CE	GPIO 13	17
CSN	GPIO 14	19
SCK	GPIO 10	14
MOSI	GPIO 11	15
MISO	GPIO 12	16

En la parte derecha del plano se encuentra un módulo USB-TTL YP-01, cuya función es permitir que la telemetría recibida pueda visualizarse en: Thonny (consola serial), o una aplicación web local desarrollada para el proyecto.

Conexiones :

VCC → 5 V

GND → GND

TX del módulo USB-TTL → GP1 (RX de la Pico)

RX del módulo USB-TTL → GP0 (TX de la Pico)

Es decir, se configuró UART0 como canal de transmisión hacia el computador. Una vez conectado el USB-TTL al PC, los datos comienzan a recibirse en tiempo real.

RX2 actúa como estación base, recibiendo la telemetría del carro en tiempo real y enviándola al computador en formato de texto. Cada paquete contiene: Temperatura del motor, Aceleración y giros, Lectura del seguidor de línea, Nivel de batería, Coordenadas GPS, Estado del enlace. Otros valores según la programación. Esta información se procesa y se presenta en la interfaz gráfica desarrollada como parte del proyecto.

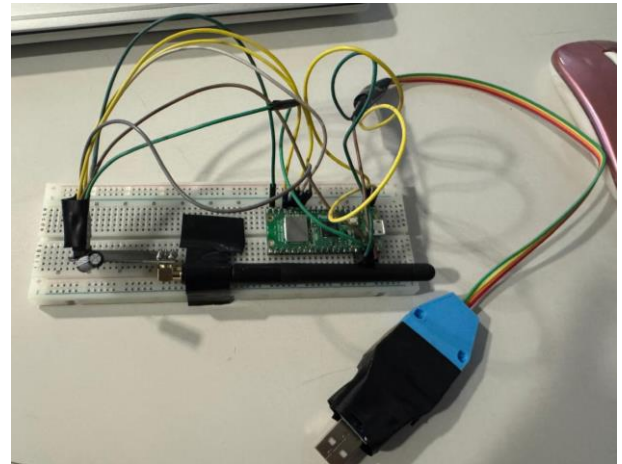


Imagen de lab. 10. Prototipo de receptor de telemetría

Antes de proceder a la etapa de soldadura en baquelas, se realizó una verificación integral del funcionamiento de todos los subsistemas del proyecto con las placas montadas inicialmente sobre protoboard. Esta fase fue fundamental para asegurar que tanto la lógica de control como la telemetría operaran correctamente antes de fijar las conexiones de forma permanente.

Durante estas pruebas se comprobó que:

- ✓ La comunicación inalámbrica entre TX1 y RX1 era estable, permitiendo el control en tiempo real del vehículo.
- ✓ El servomotor de dirección respondía correctamente a los movimientos del joystick, con amplitud calibrada y sin retrasos perceptibles.
- ✓ El motor DC funcionaba adecuadamente a través del módulo ESC, logrando aceleración progresiva y frenado según los comandos enviados desde el control.
- ✓ La placa TX2 transmitía toda la telemetría sin pérdidas, incluyendo datos del GPS, acelerómetro-giroscopio (MPU6050), sensor LM35, divisor de voltaje y el seguidor de línea.
- ✓ El receptor RX2 mostraba en el computador cada uno de los valores recibidos, tanto en Thonny mediante

puerto serial RS-232 virtual como en la interfaz web desarrollada.

- ✓ Los módulos nRF24L01 operaban en sus respectivos canales, diferenciando correctamente el canal de control del canal destinado a telemetría.
- ✓ El enlace completo del sistema (TX1 → RX1 → vehículo → TX2 → RX2 → PC) se encontraba funcional, sin pérdidas de paquetes y con actualización en tiempo real.

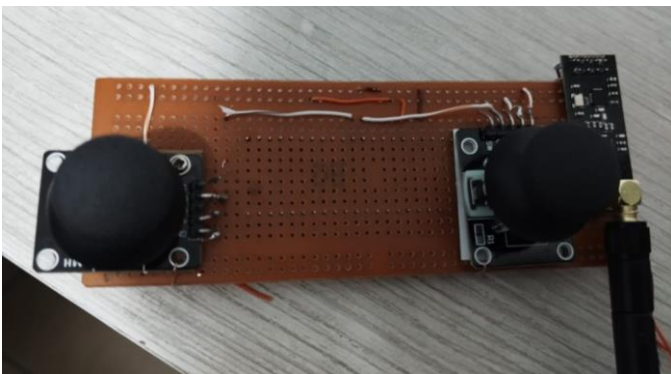
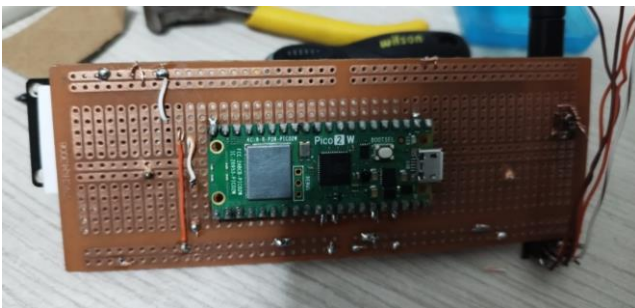
Los sensores instalados entregaban señales estables, verificando su calibración previa, GPS con coordenadas válidas, MPU6050 entregando aceleración y rotación coherentes, LM35 con lectura térmica estable, divisor de voltaje mostrando correctamente el nivel de la batería, seguidor de línea detectando superficies oscuras y claras.

PROCESO DE SOLDADURA

Una vez verificado el correcto funcionamiento de todo el sistema en protoboard, se procedió a la etapa de ensamblaje definitivo mediante soldadura en baquetas universales. Esta fase fue fundamental para garantizar la robustez estructural del sistema, evitar falsos contactos, mejorar la estabilidad eléctrica y permitir la instalación fija de los módulos dentro del vehículo.

Las tres placas principales del proyecto (TX1, RX1 y TX2) fueron soldadas cuidadosamente siguiendo rutas organizadas para alimentación, señales digitales, buses de comunicación y tierra común.

• PLACA TX1- CONTROL REMOTO



Proceso de soldadura:

1. Se fijó la Pico en el centro de la baqueta para facilitar la distribución de señales.
2. Se soldaron los pines de cada joystick, asegurando las rutas hacia los ADC correspondientes.
3. Se realizó el ruteo del bus SPI hacia el nRF24L01 (CE, CSN, MOSI, MISO, SCK y 3.3 V).
4. Se unificaron las tierras en un único punto para evitar ruido en las señales analógicas.
5. Se reforzaron las líneas de alimentación con cable más grueso y termoencogible para evitar fatiga.

• PLACA RX1 , LOS ACTUADORES DEL CARRO

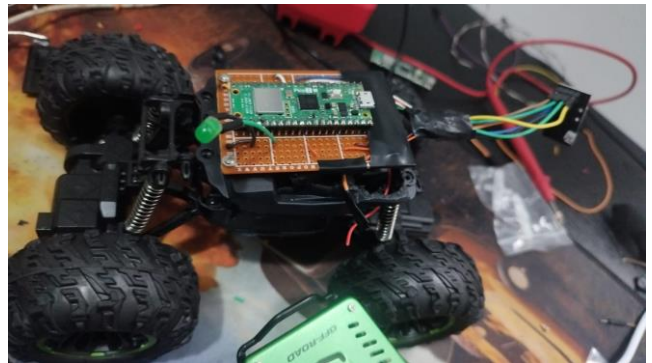
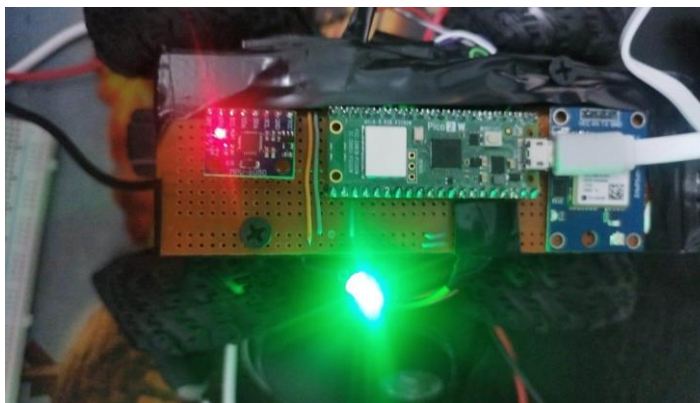


Imagen de lab. 11. Prototipo de carro con la placa Rx

Proceso de soldadura:

1. Se montó la Pico en la baqueta y se soldaron sus pines principales.
2. Se conectaron las señales PWM para el servo y el motor ESC directamente desde la Pico.
3. Se soldó el nRF24L01 con rutas cortas para reducir interferencias en la comunicación SPI.
4. Se instaló la alimentación independiente para el ESC, manteniendo la referencia de tierra unificada.
5. Se reforzaron físicamente las uniones para soportar vibraciones del vehículo.
6. Se probó el movimiento del servo y motor después de soldar cada etapa para evitar errores acumulados.

• PLACA TX2, TELEMETRÍA



Proceso de soldadura:

1. Se soldó la Pico y se distribuyeron las rutas para SPI, I2C, UART, ADC y GND.
2. Se montó el divisor de tensión directamente en la baqueta para asegurar precisión y estabilidad.
3. Se soldó el MPU6050 por I2C (SDA–SCL) con rutas cortas para evitar ruido.
4. El GPS se soldó mediante UART y se fijó físicamente por su peso y tamaño.
5. Se instalaron las conexiones del LM35 y su referencia de tierra estable.
6. El TCRT5000 se soldó a la baqueta y se ubicó físicamente en la parte frontal del carro con cable reforzado.
7. Se soldó el nRF24L01 con su bus SPI y alimentación de 3.3 V.

CODIGOS FUENTES QUE SE USARON PARA FUNCIONAMIENTO

Los códigos completos utilizados en cada una de las placas (TX1, RX1 y TX2), así como las versiones finales del proyecto, pueden consultarse en el repositorio de GitHub en el siguiente enlace:

<https://github.com/bejaranoximena12/PROYECTO-CARRO-RC-upload>

A continuación, se describe de manera general qué hace cada programa y cómo se relaciona con la placa correspondiente.

• TX1

```
1 # ===== TX: MÁXIMA EFICIENCIA CONTROL + UART (no verificado) =====
2 from machine import Pin, SPI, PWM, ADC, UART
3 import utime, struct
4 from nrf24l01 import nRF24L01
5
6 # ---- LED indicador (GP28) ----
7 led = Pin(28, Pin.OUT)
8 led.off()
9 utime_reception = utime_ticks_ms()
10 TIEMPO_SIN_SEÑAL = 2000
11
12 # ---- Servo dirección (GP15) ----
13 servo = PWM(Pin(15))
14 servo.freq(50)
15
16 def mover_servo_instantaneo(angulo):
17     angulo = max(0, min(90, int(angulo)))
18     pulso_us = 500 + (angulo * 2000) // 180
19     servo.duty_ns(int(pulso_us * 1000))
20     return pulso_us
21
22 # ---- Motor ESC (GP9) ----
23 esc = PWM(Pin(9))
24 esc.freq(50)
25
```

La placa TX1 corresponde al control remoto del vehículo y ejecuta el programa encargado de leer los dos joysticks analógicos, convertir esos valores en comandos de dirección y velocidad, empaquetarlos en un formato compacto y enviarlos de forma inalámbrica a la placa RX1 mediante el módulo nRF24L01.

En este código se configuran: La interfaz SPI0 de la Raspberry Pi Pico para comunicarse con el nRF24L01. Una tubería de transmisión con una dirección lógica propia y el canal 100, reservado exclusivamente para el enlace de control del carro. Dos entradas analógicas (ADC), una asociada al joystick de dirección (servo) y otra al joystick de aceleración (motor). Periódicamente, el programa: Lee el valor analógico de cada joystick. Escala el valor del primer joystick a un ángulo de servo (por ejemplo 0–60°). Escala el valor del segundo joystick a un porcentaje de velocidad del motor (0–100%).

Construye un paquete binario que incluye: un byte de sincronismo (SYNC_BYTE), el ángulo del servo, la velocidad del motor, y un checksum calculado a partir de estos campos. Envía el paquete usando el nRF24L01 en modo transmisión. El checksum permite que la placa RX1 verifique la integridad de cada comando recibido antes de aplicarlo a los actuadores. Además, el programa incorpora una lógica básica de robustez: si se detectan varios errores consecutivos en el envío, el radio nRF24L01 se reinicializa automáticamente. De esta manera, el software de TX1 se relaciona directamente con la placa física del control, traduciendo el movimiento de los joysticks en órdenes de dirección y aceleración que viajan vía radio hacia el vehículo.

• RX1

```
1 # ===== RX: MÁXIMA EFICIENCIA CONTROL + UART (no verificado) =====
2 from machine import Pin, SPI, PWM, ADC, UART
3 import utime, struct
4 from nrf24l01 import nRF24L01
5
6 # ---- LED indicador (GP28) ----
7 led = Pin(28, Pin.OUT)
8 led.off()
9 utime_reception = utime_ticks_ms()
10 TIEMPO_SIN_SEÑAL = 2000
11
12 # ---- Servo dirección (GP15) ----
13 servo = PWM(Pin(15))
14 servo.freq(50)
15
16 def mover_servo_instantaneo(angulo):
17     angulo = max(0, min(90, int(angulo)))
18     pulso_us = 500 + (angulo * 2000) // 180
19     servo.duty_ns(int(pulso_us * 1000))
20     return pulso_us
21
22 # ---- Motor ESC (GP9) ----
23 esc = PWM(Pin(9))
24 esc.freq(50)
25
```

La placa RX1 está instalada dentro del vehículo y ejecuta el programa que recibe los comandos enviados por TX1, controla directamente el servomotor de dirección y el módulo ESC del motor DC, y al mismo tiempo genera una telemetría básica que se envía por UART hacia la siguiente placa.

En este programa se configuran: El módulo nRF24L01 en modo receptor, usando el mismo canal 100 y las mismas direcciones lógicas definidas en TX1. Un PWM dedicado al servomotor (control de dirección). Un PWM dedicado al ESC del motor (control de velocidad). Un ADC conectado a un divisor de tensión que permite medir el voltaje de la batería del vehículo. Una interfaz UART0 para enviar telemetría hacia la placa TX2. Un LED indicador de enlace.

El flujo de funcionamiento es el siguiente: La placa RX1 permanece escuchando paquetes en el nRF24L01.

Cada vez que llega un paquete, se desempaqueta y se extraen: el byte de sincronismo, el ángulo deseado para el servo, la velocidad deseada para el motor, y el checksum.

Se verifica el checksum; solo si es correcto se consideran válidos los datos.

Se calcula el ancho de pulso correspondiente para el servomotor (en microsegundos) y se actualiza el PWM de dirección. Se calcula el pulso para el ESC a partir del porcentaje de velocidad y se actualiza el PWM del motor (avance, neutro o frenado). Periódicamente se lee el voltaje de la batería a través del ADC y el divisor de tensión. Se construye una línea de texto de telemetría con el formato: `ángulo,velocidad,pwm_servo,pwm_motor,voltaj` y se envía por UART0 hacia la placa TX2.

Si durante un tiempo definido no se reciben paquetes, se activa un failsafe: el servo se centra y el motor pasa a neutro, evitando movimientos indeseados cuando se pierde la señal. En resumen, el software de RX1 es el responsable de cerrar el lazo entre el control y el carro: recibe por radio lo que envía TX1, lo convierte en movimientos reales de motor y dirección, y a la vez genera una telemetría mínima (ángulo, velocidad y nivel de batería) que se utiliza más adelante en la cadena de monitoreo.

- TX2

```
1 # ===== Placa 3 - Receptor UART de telemetría =====
2 from machine import UART, Pin
3 import utime
4
5 # UART0: RX en GPI (desde GPIO TX de Placa 2)
6 uart = UART(0, baudrate=115200, rx=Pin(1))
7
8 print("Placa 3 lista para recibir telemetría...")
9
10 while True:
11     if uart.any():
12         try:
13             linea = uart.readline()
14             if not linea:
15                 continue
16
17             # Convertir a texto
18             texto = linea.decode().strip()
19
20             except:
21                 continue
22
23             # Esperado: ángulo,velocidad,pwm_servo,pwm_motor,voltaje
24             datos = texto.split(",")
25
```

La placa TX2 (denominada en el código como Placa 3 – Receptor UART de telemetría) cumple el papel de puente de monitoreo intermedio. Su función principal es recibir, vía UART, las líneas de telemetría generadas por la placa RX1 y presentarlas de forma legible, ya sea en consola o como base para otros procesos de registro.

En este programa se configura: La interfaz UART0 en modo recepción, conectada al pin RX que recibe los datos enviados desde RX1.

El funcionamiento es el siguiente: La placa TX2 lee continuamente la UART. Cada vez que llega una línea completa, intenta decodificarla como texto. Se separan los campos usando la coma como delimitador: ángulo del servo, porcentaje de velocidad del motor, pulso del servo en microsegundos, pulso del motor en microsegundos, voltaje de la batería.

Si la línea es válida y tiene los cinco campos esperados, los convierte a números (enteros y flotantes).

Finalmente, presenta la información en un formato compacto del estilo:

Servo: xxx° (xxxx us) | Motor: xxx% (xxxx us) | Batt: x.xx V

Este código permite verificar en tiempo real que la lógica de control que viene desde TX1 y RX1 está funcionando correctamente y que el voltaje de la batería se mantiene dentro de los rangos seguros. Aunque la placa TX2 no interactúa todavía con la interfaz web, su categoría de “receptor de telemetría por UART” la posiciona como un elemento importante dentro de la cadena de software, ya que facilita la depuración y validación del sistema antes de integrar la etapa de telemetría avanzada con WiFi y página web.

SISTEMA DE TELEMETRÍA

El sistema de telemetría constituye la última etapa del proyecto y es el componente encargado de recopilar, transmitir, procesar y visualizar en tiempo real todas las variables físicas y de funcionamiento del vehículo RC. Este sistema integra hardware (placas TX2 y RX2), comunicación inalámbrica vía NRF24L01, comunicación serial (UART) y un módulo web alojado en el computador.

Todo el procesamiento final se muestra en una interfaz web de monitoreo, desarrollada específicamente para visualizar los datos que el carro genera mientras está en movimiento

La placa RX1 genera telemetría básica: ángulo del servo, velocidad del motor, pulsos PWM y voltaje de batería. Esta telemetría se envía a través de UART hacia la placa TX2. La placa TX2 procesa los datos y los transmite por NRF24L01 (canal 76) hacia la estación receptora RX2. La placa RX2 recibe la trama completa de telemetría del vehículo, la decodifica y la reenvía por UART hacia el computador.

El computador ejecuta un script Python que:

abre el puerto serial, interpreta y organiza los datos, alimenta el servidor HTTP en localhost:8080, genera la interfaz visual con todos los indicadores.

El navegador muestra la página con la telemetría en tiempo real.

- Software del Receptor de Telemetría

```
1 # ===== Receptor NRF24L01 =====
2 from machine import UART, Pin, SPI
3 import network
4 import time
5 import struct
6 from nrf24l01 import NRF24L01
7
8 # ===== CONFIGURACIÓN WIFI =====
9 WIFI_SSID = "NOME_WIFI"
10 WIFI_PASS = "PASSWORD_WIFI"
11
12 IP_SERVIDOR = "10.10.10.10"
13 PUERTO = 8080
14 CAR_ID = "CAR_GUI000"
15 TEAM_NAME = "A los pits"
16
17 API_LAP_URL = "http://[IP_SERVIDOR]:[PUERTO]/api/lap"
18
19 CAR_ID = "CAR_GUI000"
20 TEAM_NAME = "A los pits"
21
22 # ===== CONFIGURACIÓN NRF24L01 =====
23 CHANNEL = 76
24 ADDR = b"\xc3\xf0\xf0\xf0\xf0"
25
```

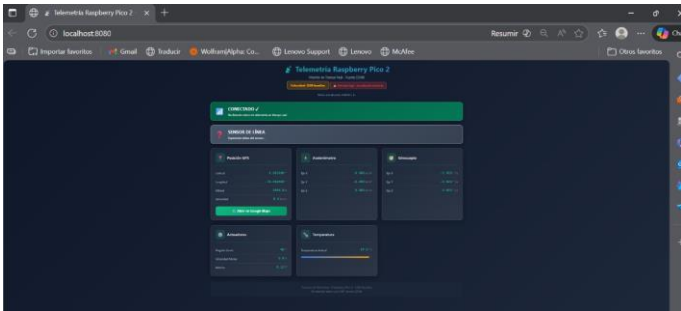
El código permite, inicializar el módulo NRF24L01 en modo RX. Escuchar tramas de 32 bytes, que incluyen: latitud y longitud, altitud, velocidad calculada, aceleraciones (AX, AY, AZ), giroscopio (GX, GY), medición de línea (0: sobre línea / 1: fuera de línea), voltaje de batería, temperatura del motor, valores

PWM de servo y motor. Desempaquetar la trama con struct.unpack("<i12h"). Convertir cada variable a unidades reales. Enviar la telemetría formateada por UART hacia la PC. El programa incluye indicadores LEDs, chequeos de integridad y envío de datos a un formato compatible con el servidor web.

- Software del Servidor Web de Telemetría (PC)

```
1 ***
2 Script Python para PC - Telemetría Raspberry Pico 2
3 Servidor Web Mejorado - BAUD RATE 1200 con Sensor de Línea
4 ***
5 import serial
6 import json
7 import webbrowser
8 import threading
9 from http.server import HTTPServer, BaseHTTPRequestHandler
10 import time
11 import re
12 from datetime import datetime
13
14 # ===== CONFIGURACIÓN =====
15 PUERTO_COM = 'COM6'
16 BAUDRATE = 1200 # Cambiado a 1200 baudios
17 PUERTO_WEB = 8080
18
19 # ===== DATOS GLOBALES =====
20 telemetry_data = {
21     "gps": {"latitude": 0.0, "longitude": 0.0, "altitude": 0.0, "speed": 0.0},
22     "accelerometer": {"x": 0.0, "y": 0.0, "z": 0.0},
23     "gyroscope": {"x": 0.0, "y": 0.0, "z": 0.0},
24     "servo": {"angle": 0.0},
25     "motor": {"speed": 0.0},
26 }
```

El sistema de telemetría del vehículo RC culmina en el computador, donde se ejecuta un script en Python encargado de recibir, procesar y visualizar los datos enviados por las placas instaladas en el carro. Este software se compone de dos partes principales: un módulo de lectura serial y un servidor web que muestra la información en tiempo real.



Pagina web de telemetria

El módulo de lectura serial abre el puerto COM6 a 1200 baudios y recibe de manera continua las tramas enviadas por la placa RX2. Cada línea contiene múltiples valores codificados, por lo que el programa emplea expresiones regulares para extraer y convertir correctamente cada variable. Todo lo recibido se organiza en la estructura telemetry_data, que mantiene actualizados los datos del GPS, acelerómetro, giroscopio, actuadores (servo y motor), voltaje de batería, temperatura del motor y estado del sensor de línea. Además, el sistema registra el estado de conexión y la hora de la última actualización.

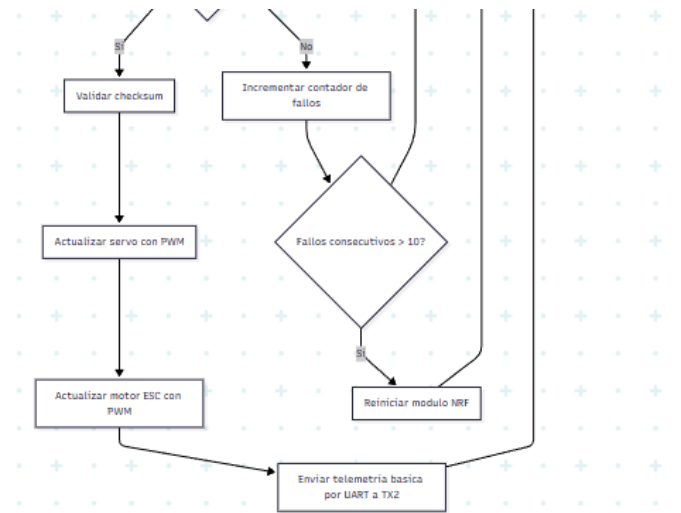
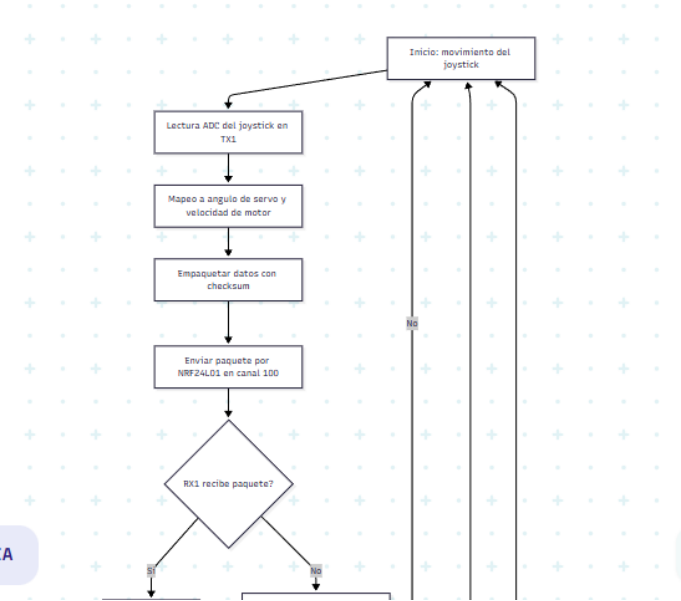
Posteriormente, el mismo script levanta un servidor HTTP local accesible desde <http://localhost:8080>, donde el panel web consulta los datos cada dos segundos a través de dos rutas: /telemetry, que entrega la telemetría en formato JSON, y /state, que informa el estado general del sistema. Esta tasa de actualización es adecuada para el enlace a 1200 baudios, permitiendo una visualización fluida sin saturar el canal.

La interfaz web fue desarrollada en HTML, CSS y JavaScript, con un diseño moderno y responsivo de estilo oscuro. Los datos se presentan en tarjetas independientes que organizan toda la información de manera clara. En la parte superior se incluye un indicador de conexión que confirma si el sistema está recibiendo datos en tiempo real. También se integra una tarjeta exclusiva

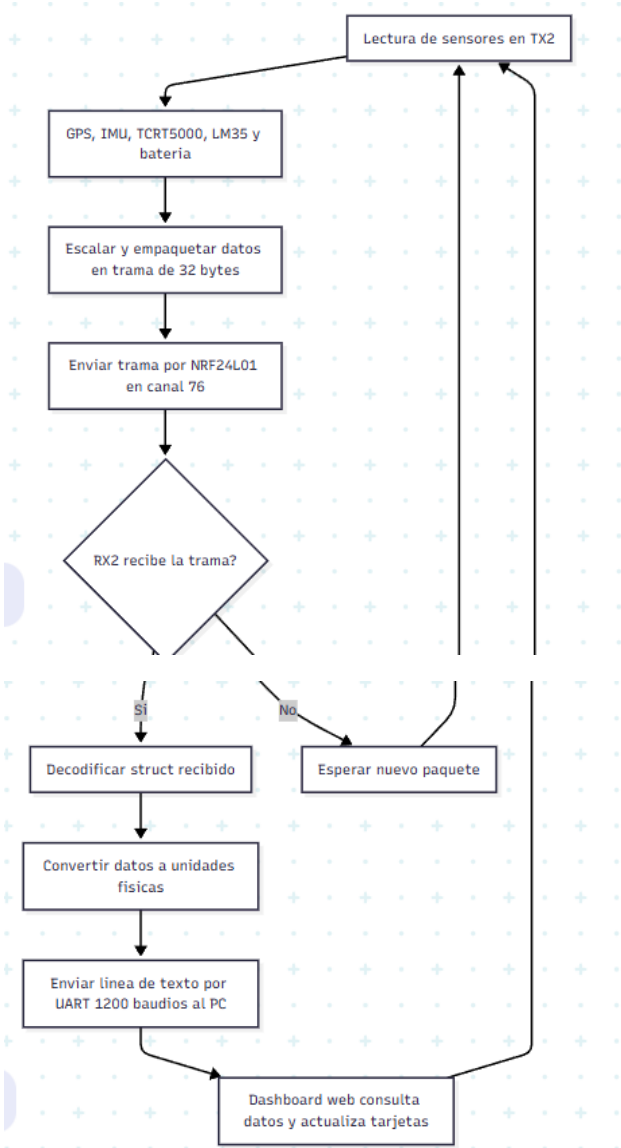
para el sensor de línea (TCRT5000), señalando si el vehículo está sobre la línea o fuera de ella. Adicionalmente, se dispone de tarjetas dedicadas para el GPS con opción de abrir la ubicación en Google Maps, el acelerómetro, el giroscopio, los actuadores del vehículo y el nivel de temperatura. Cada tarjeta actualiza automáticamente sus valores, permitiendo un monitoreo continuo del desempeño del carro durante las pruebas.

DIAGRAMAS

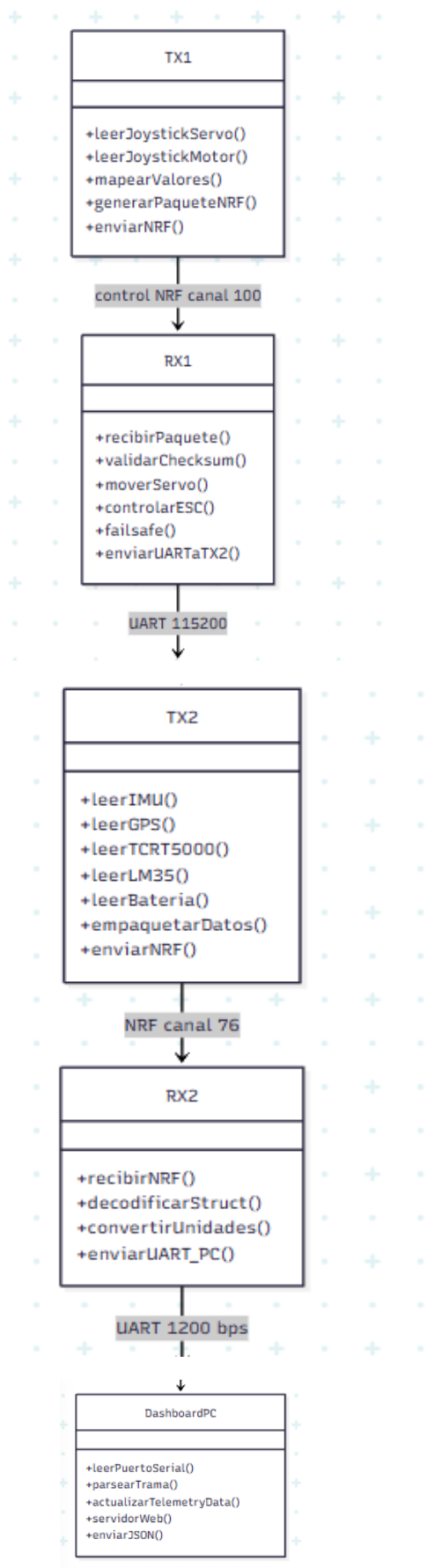
- DIAGRAMA DE FLUJO – Control del vehículo (TX1 → RX1 → actuadores)



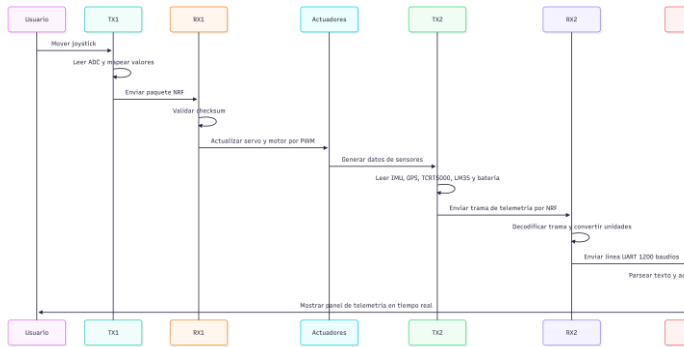
- DIAGRAMA DE FLUJO – Telemetría (TX2 → RX2 → PC)



• DIAGRAMA DE CLASES



• DIAGRAMA DE SECUENCIA

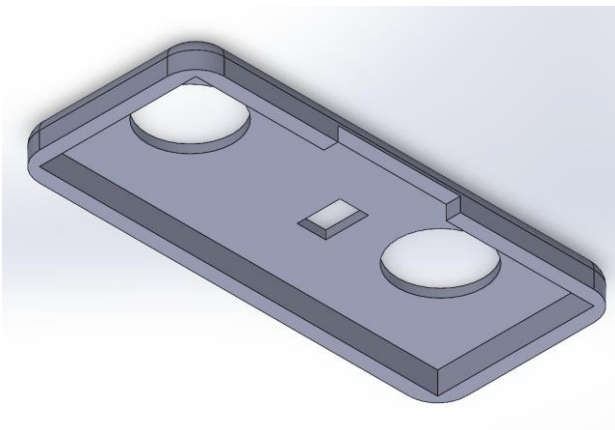


IMPRESIÓN EN 3D

Con el propósito de mejorar la protección, organización y manipulación de los módulos electrónicos utilizados en el proyecto, se diseñaron y fabricaron dos carcasas personalizadas mediante modelado CAD en SolidWorks y tecnología de impresión 3D. Estas cubiertas fueron creadas específicamente para la placa de control TX1 y para la estación receptora de telemetría RX2, ya que son los módulos que requieren mayor ergonomía, movilidad y seguridad física durante las pruebas y operación del vehículo.



- CARCASA PARA EL MÓDULO DE CONTROL TX1



La placa TX1, encargada del manejo del vehículo a través de dos joysticks, la Raspberry Pi Pico y el módulo NRF24L01, necesitaba una estructura que permitiera su uso cómodo como control remoto. Para ello se diseñó una carcasa ergonómica,

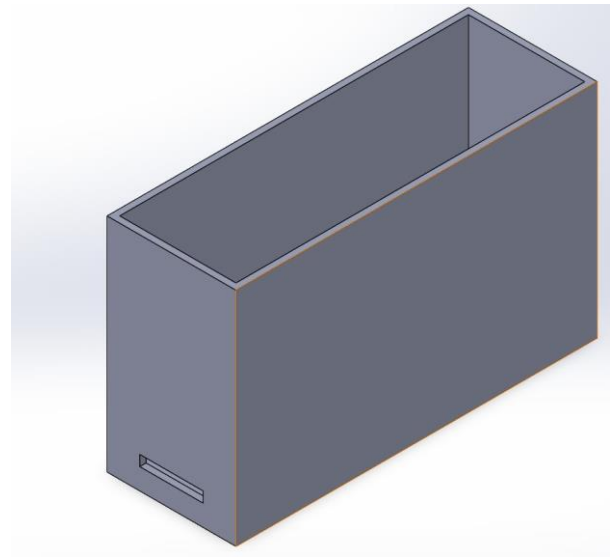
compuesta por dos piezas principales: una base de soporte y una tapa superior.

El diseño incluye:

- Orificios exactos para los dos joysticks, alineados y centrados para garantizar una operación suave.
- Una abertura rectangular en la parte central para el paso del selector o interruptor.
- Paredes internas de sujeción, destinadas a evitar el movimiento de la protoboard y de la Raspberry Pi Pico.
- Bordes redondeados, que mejoran la ergonomía y evitan puntos de presión al sostener el control.
- Ranuras laterales para ventilación y paso de cables, asegurando que la electrónica no quede comprimida.
- Ensamble tipo tapa, de fácil apertura, para mantenimiento, ajustes o conexión del puerto USB.

Gracias a este diseño, el control adquiere un aspecto compacto, resistente y más profesional, facilitando su uso prolongado sin molestias.

- CARCASA PARA LA ESTACIÓN DE TELEMETRÍA RX2



La placa RX2 responsable de recibir vía NRF24L01 los datos de telemetría y transmitirlos por USB hacia el computador también requirió una carcasa dedicada para proteger sus componentes y mantener un orden adecuado durante las pruebas.

La carcasa de RX2 fue diseñada como una bandeja cerrada, con las siguientes características:

- Espacio específico para la Raspberry Pi Pico, alineado con precisión para permitir el acceso directo al puerto micro-USB.
- Ranuras laterales para la conexión del módulo USB–TTL, utilizado en la transmisión hacia la PC.
- Aperturas internas diseñadas para organizar el cableado, evitando tensiones en los pines del módulo NRF24L01.
- Paredes reforzadas para proteger la placa de golpes o manipulación inadecuada.
- Tapa inferior desmontable, lo que permite retirar la placa con facilidad para mantenimiento o reprogramación.

Ambos modelos fueron diseñados en SolidWorks, partiendo de las dimensiones reales de la electrónica para asegurar un ajuste exacto. Las piezas se exportaron en formato STL y se imprimieron en una máquina 3D FDM con parámetros estándar: Material: PLA. Altura de capa: 0.2 mm. Relleno: 20 %. Temperatura del extrusor: 200 °C. Temperatura de cama: 60 °C

ANÁLISIS DE RESULTADOS

Al finalizar la etapa de integración, el vehículo RC del equipo “A los Pits” (nosotros) quedó completamente funcional. El control remoto por radiofrecuencia (TX1–RX1), el sistema de dirección mediante servomotor, la tracción del motor, el sensor de línea, la IMU, el GPS, el sensor de temperatura y toda la telemetría (vía NRF24L01, UART y panel web) operaron correctamente de forma simultánea.



En la jornada de competencia se realizaron dos tipos de pruebas:

Contrarreloj individual:

El vehículo debía completar el circuito en el menor tiempo posible. En esta prueba se logró un tiempo aproximado de 45 segundos, lo que permitió ubicarnos en el primer lugar de la tabla de salida. Este resultado demuestra que la respuesta del control, la tracción y la estabilidad del chasis fueron adecuadas para tomar curvas y rectas sin pérdida de control.

Carrera contra otro vehículo (5 vueltas):

En la prueba principal, el objetivo era completar 5 vueltas al circuito. Nuestro carro cumplió el requerimiento con un tiempo cercano a 2 minutos y 30 segundos, lo que equivale a un promedio de unos 30 segundos por vuelta. Con este registro el equipo “A los Pits” y sus cuatro integrantes resultamos ganadores de la competencia, evidenciando un buen equilibrio entre velocidad, estabilidad y confiabilidad del sistema electrónico.

- COMPORTAMIENTO DE LA TELEMETRÍA

Durante ambas pruebas el servidor web de telemetría se mantuvo activo en el puerto COM6 a 1200 baudios, recibiendo en tiempo real las tramas enviadas por la placa RX2. En el log capturado se observan registros típicos con el siguiente contenido:

- ServoPWM y MotorPWM: pulsos alrededor de 1500 μ s cuando el vehículo se encuentra en posición neutra, y variaciones hacia 500–1900 μ s cuando se ejecutan giros bruscos o cambios de aceleración durante el contrarreloj.
- Voltaje de batería: los valores oscilaron entre 7.2 V y 8.1 V, lo que indica que el divisor resistivo y la lectura ADC trabajan correctamente y que, pese a las exigencias de la carrera, la caída de tensión se mantiene dentro de un rango seguro para el ESC y la electrónica.
- Acelerómetro y giroscopio: en la mayor parte del recorrido se registran aceleraciones cercanas a $\pm 1 \text{ m/s}^2$ sobre los ejes X e Y y alrededor de $+9.8 \text{ m/s}^2$ en Z (gravedad). Sin embargo, en algunos puntos del contrarreloj aparecen picos mucho más altos (por ejemplo, valores de hasta 19.6 m/s^2 y cambios angulares fuertes), que corresponden a frenadas, golpes en la pista o cambios bruscos de dirección.
- GPS: las coordenadas se concentran alrededor de 4.6831° N , 74.0425° O con altitudes entre 2577 m y 2611 m, coherentes con la ubicación del campus. Las variaciones pequeñas de velocidad, entre 0 y aproximadamente 10 km/h, representan los tramos de movimiento y las zonas de frenado dentro del circuito.
- Temperatura de motor: en la prueba de contrarreloj se observaron valores típicos entre 44° C y 50° C , pero también picos elevados (por encima de $90\text{--}100^\circ \text{ C}$

algunos instantes), asociados a aceleraciones prolongadas y posibles errores de lectura puntuales. En la carrera larga la tendencia general muestra que el motor trabaja en un rango alto pero estable, lo que refuerza la importancia del monitoreo térmico para evitar sobrecalentamientos.

- Sensor de línea: en todos los registros el estado de la línea aparece como 0, lo que significa que el sensor TCRT5000 detecta continuamente el color de la pista configurado como “sobre la línea”. Esto indica que, al menos durante las mediciones capturadas, el carro se mantuvo correctamente dentro del trazado.

- ESTABILIDAD DEL ENLACE DE COMUNICACIONES

El enlace de mando entre TX1 y RX1, basado en el módulo NRF24L01, se comportó de forma estable durante la mayor parte del evento, permitiendo el control continuo del servo y el motor sin cortes apreciables. La telemetría desde el vehículo hacia RX2 y posteriormente al PC se mantuvo también operativa, aunque se registró un evento puntual de caída de comunicación durante la competencia, atribuida a saturación del canal 76, posiblemente por interferencias con el otro carro que operaba cerca del canal 70. Este comportamiento confirma que la configuración de potencia y canal del NRF24L01 es adecuada para distancias cortas, pero también deja como recomendación futura explorar otros canales menos congestionados, utilizar velocidades de transmisión más bajas para mejorar la robustez del enlace o implementar una rutina automática de cambio de canal cuando se detecte pérdida prolongada de paquetes.

En conjunto, los resultados demuestran que el diseño electrónico y de software del vehículo “A los Pits” cumplió con los objetivos planteados:

El carro fue capaz de completar el circuito en el menor tiempo del contrarreloj y de ganar la carrera principal de 5 vueltas. La telemetría en tiempo real permitió observar el comportamiento del vehículo (voltaje, temperatura, aceleraciones, velocidad y posición GPS) durante las pruebas, aportando información objetiva para evaluar el desempeño del sistema. A pesar de la limitación de 1200 baudios en el enlace UART hacia el PC, el panel web fue capaz de actualizar los datos sin saturar el canal, manteniendo una frecuencia de refresco adecuada para la aplicación. El incidente de interferencia en el canal del NRF24L01 no impidió completar la prueba, pero resalta la importancia de planear mejor la asignación de canales de radio cuando existan varios vehículos operando en un mismo entorno.

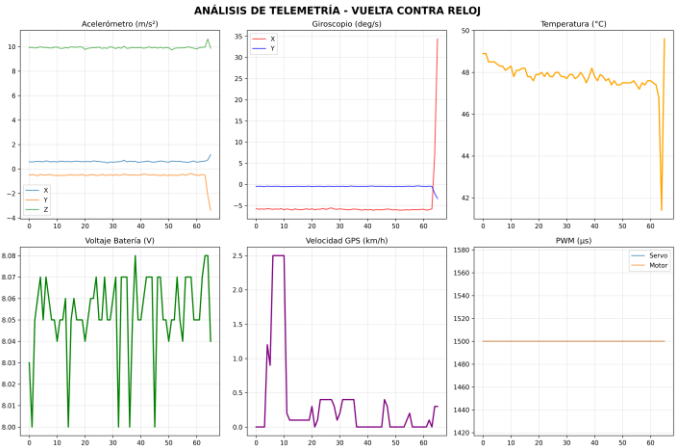


Imagen de resultados 1. Contrarreloj
Tiempo de vuelta= 45 segundos

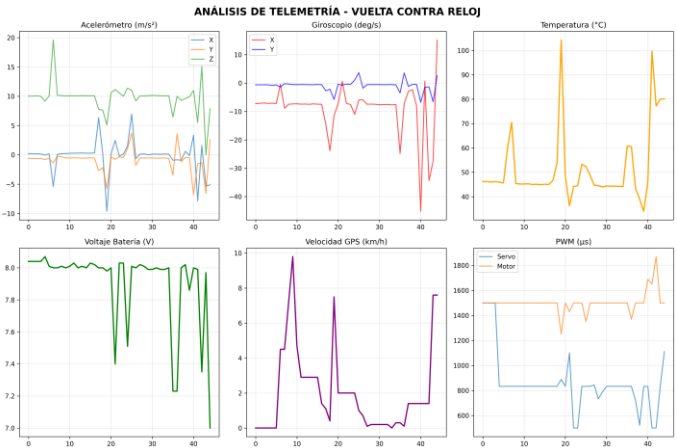


Imagen de resultados 2. Competencia
(Cabe aclarar que durante la competencia con el otro carro la telemetría se perdió en la primera vuelta, ya que había cruce en otro canal cercano usado por el otro competidor, afortunadamente fue nuestro canal de telemetría y su canal para el control del motor y servomotor, lo cual nos fue de ventaja)



GANADORES RACE_2025

CONCLUSIÓN

El proyecto permitió integrar de forma exitosa diversos sistemas de control, comunicación y adquisición de datos en un único vehículo RC totalmente operativo. El equipo “A los Pits”

logró implementar un sistema robusto de mando mediante módulos NRF24L01+ y Raspberry Pi Pico 2W, complementado con un completo módulo de telemetría que registró parámetros esenciales del vehículo durante su funcionamiento.

Las pruebas experimentales previas fueron fundamentales para asegurar la estabilidad del enlace RF, la precisión del control PWM y la correcta lectura de los sensores, lo cual se vio reflejado en el rendimiento del carro durante la competencia. La telemetría demostró ser una herramienta clave para validar el comportamiento del vehículo en tiempo real, permitiendo analizar voltajes, aceleraciones, temperatura del motor, giros y posición GPS.

Finalmente, el desempeño en pista confirmó la confiabilidad del diseño electrónico, mecánico y de software. El carro completó el circuito con tiempos competitivos y ganó la carrera principal, demostrando que la integración de los módulos, la arquitectura del sistema y la programación cumplían plenamente los objetivos planteados. El proyecto deja bases sólidas para futuras mejoras como aumento de la velocidad de muestreo, optimización del canal RF y expansión del análisis de datos en el panel web.

REFERENCIAS

- [1] *Raspberry Pi Pico 2W – Technical Documentation*. Recuperado de <https://www.raspberrypi.com/documentation/>
- [2] *nRF24L01+ 2.4 GHz Radio Transceiver – Product Specification*. Recuperado de <https://infocenter.nordicsemi.com/>
- [3] *MPU-6050: 6-Axis Motion Tracking Device – Register Map and Descriptions*. TDK InvenSense Documentation.
- [4] *NEO-6M GPS Module – Hardware Integration Manual*. u-blox AG.
- [5] *LM35 Precision Centigrade Temperature Sensors – Datasheet*. Recuperado de <https://www.ti.com>
- [6] *Understanding UART/Serial Communication in Embedded Systems*. Nota de aplicación técnica.

REPOSITORIO GIT

- [7] <https://github.com/bejaranoximena12/PROYECTO-CARRO-RC->

LINK DE VIDEO

- [8] <https://youtu.be/ClgzX3KXa7g?si=IpK8gfKYjOauu2ke>