$\mathcal{DIFU}_{100}ci@$
Revista de Difusión Científica, Ingeniería y Tecnologías
Vol. 19, No. 1, Ene - Dic 2025
ISSN:2007-3585
Artículo arbitrado
Publicado: 30 de diciembre de 2025

AT Command-Based Library Development for Microchip RN2903 LoRa Module

Desarrollo de una biblioteca basada en comandos AT para el módulo LoRa RN2903 de Microchip

Pedro Manuel Montoya Lara[*1], **Oscar Osvaldo Ordaz García**[1], **José Ricardo Gómez Rodríguez**[1], **Víktor Iván Rodríguez Abdalá**[1], **Salvador Ibarra Delgado**[1], **Ángel García Durán**[1], **Remberto Sandoval Aréchiga**[1], and **Roberto Solís Robles**[1]

[1] *Universidad Autónoma de Zacatecas (UAZ), Unidad Académica de Ingeniería Eléctrica,*
Posgrado en Ingeniería para la Innovación Tecnológica,
Campus Siglo XXI Carretera Zacatecas - Guadalajara Kilómetro 6, Ejido la Escondida, Zacatecas , Zac., México, 98160.
{pedromontoyalara,oscarordazg,jrgrodri,abdala,sibarra,angel.garcia.duran,rsandoval,rsolis}@uaz.edu.mx

Abstract

This article presents the development of a C library designed to simplify interaction with the Microchip RN2903 LoRa module for Internet of Things applications. The work addresses the complexity of directly managing AT commands, which can hinder developers with limited experience in communication systems. The proposed library provides a structured UART-based interface for configuring parameters and managing data transmission and reception. It also integrates feedback and fault-tolerance mechanisms to ensure reliable operation. Experimental validation was carried out through point-to-point LoRa communication in an urban environment, achieving distances of up to 1.1 km with stable performance.

***Keywords*—** IoT, LoRa, C Library

Resumen

Este artículo presenta el desarrollo de una biblioteca en C para simplificar la interacción con el módulo LoRa RN2903 de Microchip en aplicaciones de Internet de las Cosas. El trabajo surge ante la complejidad de gestionar directamente los comandos AT, lo que dificulta su uso para desarrolladores con experiencia limitada. La biblioteca propuesta proporciona una interfaz estructurada basada en comunicación UART para configurar parámetros y gestionar la transmisión y recepción de datos. Además, incorpora mecanismos de retroalimentación y tolerancia a fallas para asegurar la operación confiable. La validación experimental se realizó mediante comunicación LoRa punto a punto en un entorno urbano, alcanzando hasta 1.1 km con funcionamiento estable.

***Palabras clave*—** IoT, LoRa, Biblioteca en C

I. Introducción

El Internet de las Cosas (IoT, siglas en inglés de Internet of Things) ha mostrado un crecimiento acelerado en los últimos años, las proyecciones indican que esta tendencia se mantendrá en el futuro cercano. El IoT es una red de dispositivos inteligentes que integran sensores, nodos de comunicación, software embebido y otras tecnologías, los cuales son diseñados para la recolección, procesamiento y transmisión de datos hacia la nube. Estos nodos, que pueden contener varios sensores, pueden estar ubicados en áreas remotas, por lo que el envío de información hacia la nube es un reto, para ello, se recomienda el uso de LPWAN (siglas en inglés de Low Power Wide Area Network), la cual es una tecnología diseñada para nodos que deben enviar

[*]Corresponding author

pequeñas cantidades de datos a largas distancias con un consumo de energía bajo [1, 2].

Entre las LPWAN, LoRaWAN (siglas en inglés de Long Range Wide Area Network) se ha destacado como una tecnología entre aplicaciones de corto alcance que usan Wi-Fi y Bluetooth y de alto consumo energético como las redes celulares, ofreciendo una solución de largo alcance, eficiencia energética y de bajo costo para IoT [3, 4].

LoRa (siglas en inglés de Long Range) es la técnica de modulación de espectro expandido que permite transmitir datos a largas distancias con bajo consumo de energía. Sobre esta capa física se construye LoRaWAN, que define el protocolo de red y la arquitectura necesaria para gestionar la comunicación entre dispositivos finales, puertas de enlace (gateways en inglés) y servidores de red y aplicación, permitiendo la integración masiva de dispositivos IoT en entornos urbanos, rurales e industriales [3, 4].

El apoyo de fabricantes como Microchip, han permitido que LoRaWAN sea popular entre desarrolladores con pocos conocimientos de radiofrecuencia, ya que el uso de módulos de solo conectar y usar (plug and play en inglés) propicia un rápido desarrollo de aplicaciones IoT. El RN2903 es un módulo transceptor LoRa que opera en la banda de 915 MHz, su arquitectura integra una pila LoRaWAN completo, lo que permite a los desarrolladores comunicarse con el módulo mediante comandos ASCII (siglas en inglés de American Standard Code for Information Interchange), también conocidos como comandos AT (siglas en inglés de Attention Commands), a través de UART (siglas en inglés de Universal Asynchronous Receiver/Transmitter), sin necesidad de implementar directamente el protocolo LoRaWAN en el microcontrolador [5, 6].

A pesar de la disponibilidad de módulos LoRa como el RN2903, su uso requiere la gestión directa de comandos AT, lo que implica un conocimiento detallado del funcionamiento interno del dispositivo y puede resultar complejo para desarrolladores con experiencia limitada en sistemas de comunicación. Esta situación genera la necesidad de contar con herramientas que abstraigan dicha complejidad. En este contexto, el presente trabajo propone el desarrollo de una biblioteca en lenguaje C que encapsula los comandos AT del módulo RN2903, proporcionando una interfaz estructurada, simple y reutilizable para facilitar su implementación en aplicaciones IoT.

II. Biblioteca RN2903

La biblioteca desarrollada para el módulo RN2903 de Microchip fue escrita en lenguaje C dentro del entorno de desarrollo MPLAB X. Su propósito es ofrecer un conjunto de funciones que facilitan la configuración y el control del radio, integrando de manera estructurada los comandos necesarios para inicializar, ajustar parámetros y gestionar la comunicación LoRa. Con este enfoque, se busca simplificar el trabajo del desarrollador y garantizar una implementación eficiente y flexible en aplicaciones de IoT [7, 8, 9, 10].

Actualmente existen diferentes bibliotecas para radios LoRa, por ejemplo, Semtech ofrece la biblioteca LoRaMAC-Node, escrita en C y orientada a microcontroladores ARM [7]; la biblioteca Arduino LoRa, desarrollada por Sandeep Mistry, permite un manejo básico de módulos basados en el chip SX127x; RadioLib amplía la compatibilidad hacia múltiples dispositivos LoRa, como la serie RFM9X (95, 96, 97, 98), mediante una arquitectura en C++; estas alternativas requieren de la pila de LoRaWAN en el microcontrolador, ya que los radios son configurados a través de los registros de memoria vía SPI [11].

Dado que el RN2903 de Microchip incorpora de manera nativa la pila LoRaWAN en su firmware, no resulta necesario desarrollar una biblioteca adicional que implemente dicho protocolo. Sin embargo, para facilitar el uso del radio y simplificar la interacción con el módulo, sí es conveniente contar con una biblioteca de comandos AT, la cual permite gestionar de forma estructurada las funciones de configuración, control y transmisión, ofreciendo al desarrollador una interfaz más accesible y eficiente. El desarrollo esta biblioteca se basó en la guía de usuario, donde se describen los comandos de la pila LoRaWAN [12].

Los comandos AT son un conjunto de instrucciones basadas en texto que permiten controlar dispositivos electrónicos, como módems o radios, a través de UART. Su funcionamiento se basa en el envío de cadenas de caracteres ASCII desde un microcontrolador o computadora hacia el dispositivo, el cual interpreta dichas órdenes y responde con mensajes de confirmación o los datos solicitados [13, 14].

Es importante destacar que la biblioteca desarrollada opera en modo LoRa punto a punto, utilizando comandos AT del módulo RN2903 en su modalidad “radio”, sin emplear la pila LoRaWAN integrada. Esta decisión permite un mayor control sobre la transmisión y recepción de datos, así como una implementación más simple para aplicaciones específicas que no requieren infraestructura de red LoRaWAN.

III. Descripción de hardware del nodo

La Fig. 1 presenta el diagrama esquemático del nodo IoT desarrollado, el cual está compuesto por una tarjeta PIC-IoT WG de Microchip y el módulo transceptor LoRa RN2903. A diferencia de una arquitectura a nivel de bloques, este diagrama muestra la interconexión eléctrica entre los componentes, destacando principalmente la comunicación mediante la interfaz UART [15, 12, 6].

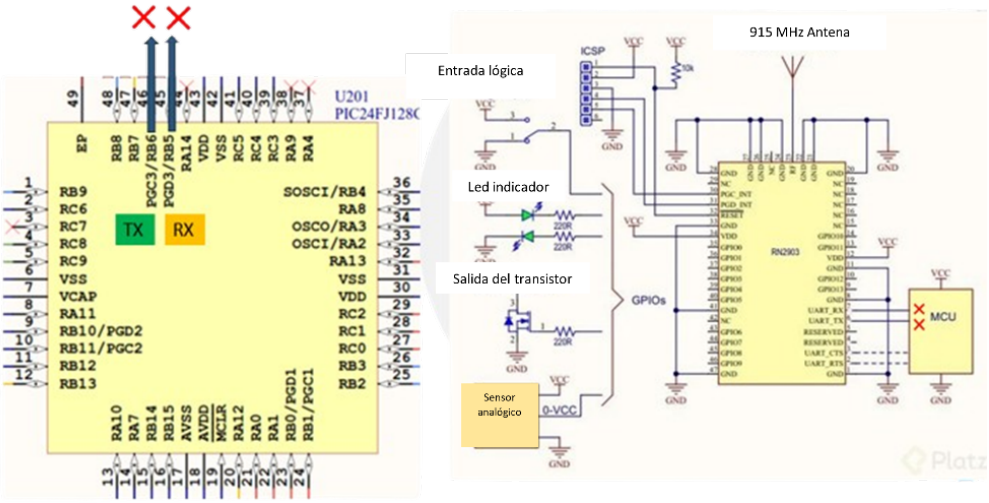


Figura 1: *Diagrama esquemático del nodo IoT basado en PIC-IoT WG y módulo LoRa RN2903*

El microcontrolador PIC24FJ128GA705 actúa como la unidad de procesamiento principal, encargado de ejecutar la biblioteca desarrollada y gestionar la lógica de comunicación. A través de la interfaz UART, el microcontrolador envía comandos AT al módulo RN2903 y recibe las respuestas correspondientes, permitiendo la configuración y operación del enlace LoRa.

El módulo RN2903 integra en su firmware la pila del protocolo LoRaWAN, aunque en este trabajo se emplea únicamente su modo de operación LoRa punto a punto mediante comandos de tipo “radio”. Esta característica permite simplificar la implementación, delegando las funciones de modulación y transmisión al propio módulo. Este opera en la banda de 915 MHz, correspondiente a la región ISM (siglas en inglés de Industrial, Scientific and Medical), la cual es de uso libre en países como Estados Unidos y México, sujeta a regulaciones específicas de potencia y uso del espectro.

Adicionalmente, la tarjeta PIC-IoT WG incorpora un módulo de seguridad ATECC608A y conectividad Wi-Fi mediante el dispositivo ATWINC1510; sin embargo, estos elementos no son utilizados en las pruebas experimentales, ya que el enfoque del presente trabajo se centra exclusivamente en la comunicación LoRa.

Entre las características principales del PIC-IoT WG, se tiene:

- MCU PIC24FJ128GA705: arquitectura de 16 bits, bajo consumo, con integración analógica avanzada.
- ATECC608A CryptoAuthentication: elemento seguro para almacenamiento de claves y autenticación basada en hardware.
- ATWINC1510 Wi-Fi: controlador optimizado para aplicaciones alimentadas por batería, con certificación completa.
- Conectividad en la nube: originalmente orientada a Google Cloud IoT Core (ya descontinuado), pero puede integrarse con Amazon AWS mediante herramientas de provisión de Microchip.
- Diseño compacto y listo para usar: pensado para acelerar prototipos IoT con seguridad y conectividad integradas.

Las características del módulo RN2903, como su bajo consumo energético, la integración de la pila de comunicación y su interfaz UART basada en comandos AT, lo convierten en una solución adecuada para aplicaciones IoT de bajo consumo y largo alcance. Estas propiedades permiten reducir la complejidad en el microcontrolador, ya que las funciones de radiofrecuencia son gestionadas internamente por el módulo. A continuación se listan sus características principales.

- Capa de red integrada: incluye el protocolo LoRaWAN en el firmware.
- Interfaz sencilla: comunicación mediante UART con comandos ASCII.
- Dimensiones compactas: $17.8 \times 26.7 \times 3.34$ mm, con contactos para montaje en PCB.
- Certificaciones: homologado para EE. UU. (FCC, siglas en inglés de Federal Communications Commission), Canadá (IC, siglas en inglés de Industry Canada) y otros países.
- Consumo energético bajo: ideal para dispositivos IoT alimentados por batería.
- Voltaje de operación: 2.1 V a 3.6 V (típico 3.3 V).
- Rango de temperatura: $-40°$C a $+85°$C.
- Modulación: soporta LoRa y FSK.
- Actualización de firmware (DFU siglas en inglés de Device Firmware Upgrade): posible vía UART.
- Identidad única: incluye EEPROM (siglas en inglés de Electrically Erasable Programmable Read-Only

Memory) con dirección EUI-64 (siglas en inglés de Extended Unique Identifier) preprogramada.

IV. Flujo de transmisión y recepción

El funcionamiento de la biblioteca desarrollada se basa en la interacción mediante comandos AT del tipo *radio* entre el microcontrolador PIC24FJ128GA705 y el módulo LoRa RN2903 a través de la interfaz UART. Es importante destacar que el sistema opera en modo LoRa punto a punto, sin emplear la pila LoRaWAN integrada en el módulo, lo que permite un control directo sobre los parámetros de comunicación y simplifica la implementación.

El proceso general de operación se divide en dos modos: transmisión y recepción, ambos estructurados mediante secuencias de configuración, verificación y ejecución. Además, se incorpora un mecanismo de detección de errores que garantiza la robustez de la comunicación, tal como se puede observar en la Fig. 2.

IV.1. Transmisión

El proceso de transmisión inicia con la configuración del módulo RN2903 mediante una serie de comandos AT que permiten establecer los parámetros de comunicación, tal como se observa en la Fig. 2a. Entre los principales comandos utilizados se encuentran:

- `radio set freq 915000000` (configuración de la frecuencia de operación)
- `radio set sf sf7` (factor de esparcimiento)
- `radio set bw 125` (ancho de banda)
- `radio set pwr 14` (potencia de transmisión)

Una vez configurado el módulo, el microcontrolador envía comandos de verificación para asegurar que los parámetros fueron correctamente establecidos. Estas verificaciones se realizan mediante comandos del tipo `radio get`, los cuales permiten consultar los valores actuales del dispositivo.

Posteriormente, se ejecuta el comando de transmisión:

- `radio tx <datos_hex>`

donde los datos a transmitir se codifican en formato hexadecimal. El módulo RN2903 responde con `radio_tx_ok` en caso de una transmisión exitosa o `radio_err` si ocurre un error durante el proceso.

Finalmente, para optimizar el consumo energético, se envía el comando:

- `sys sleep <tiempo>`

el cual coloca al módulo en modo de bajo consumo durante un intervalo definido en milisegundos. Durante este periodo, el microcontrolador puede entrar en estado de reposo o ejecutar otras tareas, mejorando la eficiencia del sistema.

La transmisión queda resumida en los siguientes pasos:

1. Comandos enviados por el microcontrolador al RN2903 para configurarlo en modo LoRa (transmisión).
2. Respuestas de aceptación del RN2903 tras recibir cada comando.
3. Comandos de verificación enviados por el microcontrolador.
4. Respuestas del RN2903 confirmando la correcta configuración.
5. Envío del comando de transmisión en formato hexadecimal.
6. Respuesta del RN2903 indicando si la transmisión fue exitosa.
7. Comando para establecer el modo de sueño por un determinado tiempo en milisegundos.
8. El microcontrolador entra en modo de sueño o realiza otras tareas mientras espera completar la transmisión.

IV.2. Recepción

El proceso de recepción se inicia de manera similar (véase Fig. 2b), mediante la configuración del módulo en modo de recepción utilizando los mismos parámetros de comunicación definidos en el transmisor. Una vez verificada la correcta configuración, se activa la recepción mediante el comando:

- `radio rx 0`

Este comando habilita el modo de escucha continua del módulo RN2903. En caso de que los datos sean recibidos correctamente, el módulo responde con:

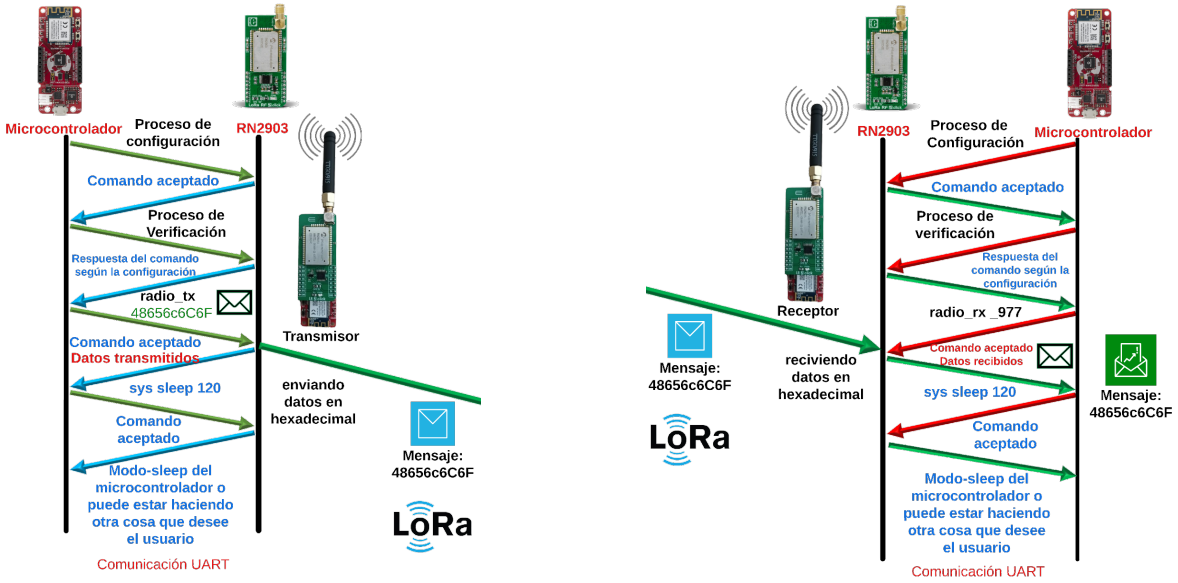
- `radio_rx <datos>`

donde `<datos>` corresponde a la información recibida en formato hexadecimal. Si no se recibe información o ocurre una falla, se devuelve la respuesta `radio_err`.

Al igual que en el proceso de transmisión, se emplea el comando `sys sleep` para reducir el consumo energético durante periodos de inactividad.

Estos pasos se resumen a continuación:

1. Comandos enviados por el microcontrolador para configurar el RN2903 en modo LoRa (recepción).
2. Respuestas del RN2903 tras recibir cada comando.
3. Comandos de verificación enviados por el microcontrolador.
4. Respuestas del RN2903 confirmando la correcta configuración.
5. Envío del comando de recepción junto con la cantidad de símbolos.



(a) *Transmisor* **(b)** *Receptor*

Figura 2: *Funcionamiento de la Biblioteca LoRa*

6. Respuesta del RN2903 confirmando si los datos fueron recibidos exitosamente.
7. Comando para establecer el modo de sueño por un determinado tiempo en milisegundos.

IV.3. Manejo de errores

La biblioteca implementa un mecanismo de tolerancia a fallas que permite detectar problemas en la comunicación UART entre el microcontrolador y el módulo RN2903. Este mecanismo se basa en la verificación de la recepción de una línea completa de datos mediante una bandera de control.

En caso de no recibir respuesta del módulo, se considera que ha ocurrido una falla de comunicación. Ante esta situación, el sistema ejecuta una rutina de recuperación que:

1. Finaliza correctamente la cadena almacenada en el buffer.
2. Genera un mensaje de error para notificar al usuario.
3. Reinicia las variables de control del sistema.
4. Permite reintentar el proceso de comunicación.

Este enfoque garantiza una operación robusta del sistema, permitiendo la detección oportuna de errores y la recuperación automática sin necesidad de reiniciar el microcontrolador.

V. Funcionamiento de la biblioteca RN2903

El funcionamiento de la biblioteca se organiza como un flujo estructurado en tres etapas principales: envío de comandos, recepción de respuestas y validación, lo que garantiza una comunicación confiable con el módulo RN2903. La Fig. 11 describe de manera detallada el funcionamiento de cada una de ellas.

V.1. Etapa 1: Envío de comandos

En esta etapa, el microcontrolador construye una cadena de caracteres que representa el comando AT a ejecutar. Esta cadena es almacenada en un buffer de transmisión y enviada secuencialmente carácter por carácter a través de la interfaz UART. La Fig. 4 muestra el diagrama de flujo de esta etapa.

Este proceso asegura que el módulo RN2903 reciba correctamente la instrucción completa, constituyendo el primer paso en la configuración y control del dispositivo LoRa.

V.2. Etapa 2: Recepción y procesamiento de respuestas

Una vez transmitido el comando, el sistema espera la respuesta del módulo RN2903. La información recibida es almacenada en un buffer de entrada mediante la interfaz UART.

El procesamiento de la respuesta se activa cuando se detecta la recepción completa de una línea de datos. Posteriormente, la cadena recibida es analizada para identificar el estado de la operación solicitada.

Después de procesar la respuesta, las variables de control del buffer son reiniciadas para preparar el sistema

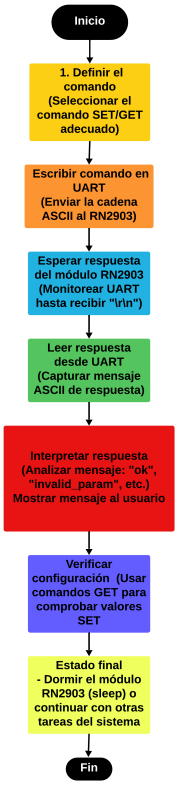


Figura 3: *Diagrama de flujo del procesamiento de comandos en la biblioteca RN2903*

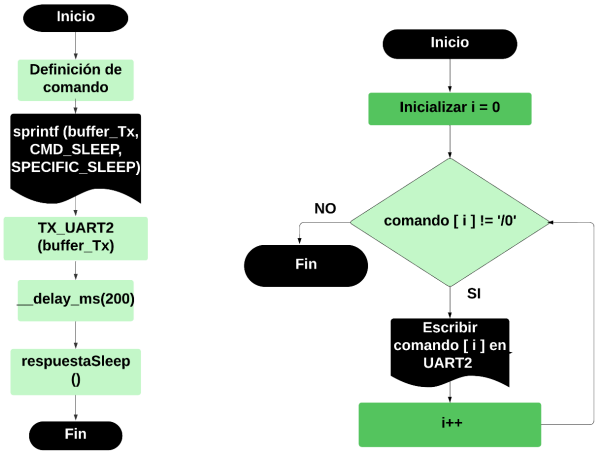


Figura 4: *Diagrama de flujo de la etapa de envío de comandos de la biblioteca RN2903*

para futuras comunicaciones, asegurando la continuidad del flujo de operación, como se ilustra en la Fig. 5.

V.3. Etapa 3: Validación del comando

En esta etapa, la biblioteca evalúa la respuesta proporcionada por el módulo RN2903 para determinar si el

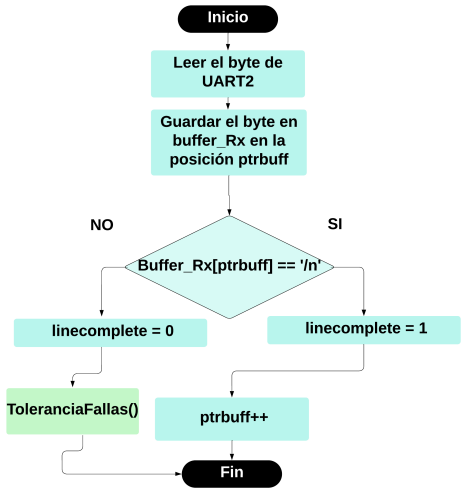


Figura 5: *Diagrama de flujo de la etapa de recepción y procesamiento de respuestas de la biblioteca RN2903*

comando fue ejecutado correctamente. Esta validación se basa en la identificación de patrones dentro de la cadena recibida, como los indicadores `ok` (comando aceptado) o `invalid_param` (parámetro inválido).

La Fig. 6 muestra el diagrama de flujo correspondiente a la etapa de validación de comandos, donde se analiza la respuesta del módulo RN2903 para determinar si la operación fue ejecutada correctamente.

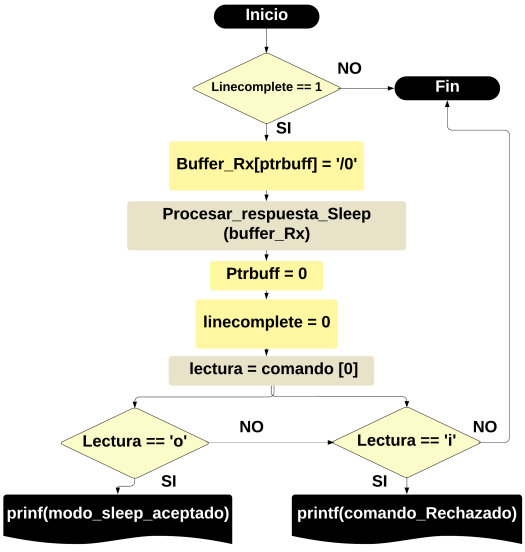


Figura 6: *Diagrama de flujo de la etapa de validación de comandos de la biblioteca RN2903*

El resultado del análisis permite generar un mensaje de retroalimentación para el usuario, indicando el estado de la operación y facilitando la detección de posibles errores de configuración.

V.4. Mecanismo de tolerancia a fallas

Con el objetivo de garantizar la robustez del sistema, se implementa un mecanismo de tolerancia a fallas que permite detectar errores en la comunicación UART entre el microcontrolador y el módulo RN2903.

Cuando no se recibe una respuesta completa dentro del tiempo esperado, el sistema asume la existencia de un fallo de comunicación y ejecuta una rutina de recuperación que incluye:

1. Finalización segura de la cadena en el buffer.
2. Generación de un mensaje de error para el usuario.
3. Reinicio de las variables de control.
4. Preparación para un nuevo intento de comunicación.

Este mecanismo permite detectar fallos de manera oportuna y recuperar la comunicación sin necesidad de reiniciar el sistema, garantizando una operación continua y confiable.

La Fig. 7 muestra el diagrama de flujo del mecanismo de tolerancia a fallas, el cual permite detectar la ausencia de respuesta del módulo RN2903 y ejecutar una rutina de recuperación para mantener la continuidad de la comunicación.

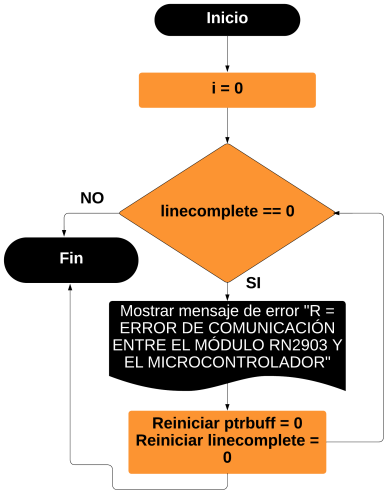


Figura 7: *Diagrama de flujo del mecanismo de tolerancia a fallas en la biblioteca RN2903*

VI. Resultados experimentales

Con el objetivo de validar el funcionamiento de la biblioteca desarrollada, se realizaron diversas pruebas experimentales que evalúan la configuración del módulo RN2903, la detección de fallas en la comunicación UART y el desempeño del enlace LoRa en escenarios reales.

La Tabla 1 muestra los principales parámetros utilizados durante las pruebas experimentales.

Tabla 1: *Parámetros de configuración del dispositivo LoRa*

Parámetro	Valor	Descripción
SPECIFIC_MOD	lora	Modo de modulación utilizado.
SPECIFIC_FREQ	915000000	Frecuencia de operación en Hz.
SPECIFIC_PWR	14	Potencia de transmisión en dBm.
SPECIFIC_SF	7	Factor de esparcimiento
SPECIFIC_CRC	on	Activación de la cabecera CRC.
SPECIFIC_IQI	off	Configuración de Invert IQ.
SPECIFIC_CR	4/7	Tasa de codificación.
SPECIFIC_WDT	2000	Tiempo del watchdog (miliseg).
SPECIFIC_SYNC	12	Valor de sincronización.
SPECIFIC_BW	125	Ancho de banda en kHz.

Los parámetros fueron seleccionados con el objetivo de balancear alcance, consumo energético y robustez del enlace LoRa en un entorno urbano.

VI.1. Pruebas de configuración del módulo RN2903

En primer lugar, se verificó la correcta configuración del módulo LoRa RN2903 mediante el envío de comandos AT desde el microcontrolador. Durante este proceso, se observó que el módulo responde adecuadamente a los comandos de configuración, retornando mensajes de aceptación (`ok`) cuando los parámetros son válidos.

Asimismo, se emplearon comandos de verificación (`radio get`) para comprobar que los valores configurados coincidieran con los parámetros establecidos, garantizando la correcta operación del sistema.

La Fig. 8 muestra un ejemplo de la verificación realizada en la terminal, donde se observan los comandos enviados y las respuestas del módulo RN2903.

```
CONFIGURANDO EL MÓDULO LORA TX
CMD: sys get ver
R = COMANDO_VERSION_GET_SISTEMA_ACEPTADO:

CMD: mac pause
R = COMANDOS_MAC_PAUSE: 4294967245
CMD: radio set mod lora
R = COMANDO_SET_LORA_ACEPTADO
CMD: radio get mod
R = MODO_LORA
```

Figura 8: *Verificación de configuración del módulo RN2903 mediante comandos AT*

VI.2. Pruebas de detección de fallas en la comunicación UART

Se evaluó el mecanismo de tolerancia a fallas implementado en la biblioteca mediante la simulación de errores en la comunicación UART. Para ello, el módulo RN2903 fue desconectado intencionalmente del sistema.

Como resultado, se detectó la ausencia de respuesta del módulo en un tiempo menor a un segundo, generando un mensaje de error en la terminal del usuario y activando una señal visual mediante un LED en la placa de desarrollo. Además, el sistema logró recuperarse automáticamente una vez restablecida la conexión, sin necesidad de reiniciar el microcontrolador.

La Fig. 9 muestra el mensaje de error generado durante la prueba, confirmando la detección de la falla en la comunicación entre el microcontrolador y el módulo RN2903.

```
radio rx 977
R = ERROR DE COMUNICACIÓN ENTRE EL MÓDULO RN2903 Y EL MICROCONTROLADOR
Esperando mensaje...
radio rx 977
R = ERROR DE COMUNICACIÓN ENTRE EL MÓDULO RN2903 Y EL MICROCONTROLADOR
Esperando mensaje...
radio rx 977
R = ERROR DE COMUNICACIÓN ENTRE EL MÓDULO RN2903 Y EL MICROCONTROLADOR
Esperando mensaje...
radio rx 977
R = ERROR DE COMUNICACIÓN ENTRE EL MÓDULO RN2903 Y EL MICROCONTROLADOR
Esperando mensaje...
```

Figura 9: *Mensaje de error de comunicación entre el microcontrolador y el módulo RN2903*

Además, el sistema logró recuperarse automáticamente una vez restablecida la conexión, sin necesidad de reiniciar el microcontrolador, validando la robustez del mecanismo implementado.

VI.3. Pruebas de transmisión y recepción LoRa

Para evaluar el desempeño del enlace LoRa, se configuraron dos nodos basados en la tarjeta PIC-IoT WG, uno como transmisor y otro como receptor, ambos equipados con el módulo RN2903.

Las pruebas se realizaron en un entorno urbano, transmitiendo datos de temperatura obtenidos del sensor MCP9808 integrado en la placa. Se evaluó el sistema durante periodos prolongados de operación continua, obteniendo los resultados que se muestran en la Tabla 2.

La Fig. 10 muestra la primera prueba de transmisión, donde se alcanzó una distancia aproximada de 560 metros con comunicación estable durante 8 horas de operación continua.

Posteriormente, se realizó una segunda prueba en condiciones similares, incrementando la distancia entre los nodos. En esta prueba, se alcanzó una distancia aproximada de 1.1 km, manteniendo comunicación estable durante 12 horas de operación continua.

Tabla 2: *Resultados de las pruebas de transmisión y recepción LoRa*

Parámetro	Enlace A	Enlace B
Distancia	560 m	1.1 km
Duración	8 h	12 h
Paquetes enviados	480	720
Paquetes recibidos	472	705
Tasa de pérdida	1.6 %	2.0 %
Intervalo de envío	60 s	60 s



Figura 10: *Prueba de enlace LoRa A a una distancia de 560 m*

La Fig. 11 muestra la prueba de enlace LoRa B, donde se valida el correcto funcionamiento del sistema a mayor distancia.

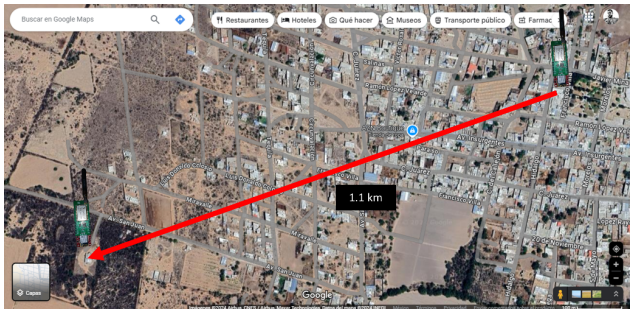


Figura 11: *Prueba experimental de enlace LoRa B (1.1 km) en entorno urbano*

Los resultados obtenidos demuestran que la biblioteca permite establecer comunicación LoRa confiable en condiciones reales, alcanzando distancias superiores a 1 km con operación estable durante largos periodos.

Cabe destacar que el alcance obtenido es menor al máximo teórico reportado por el fabricante; sin embargo, las pruebas se realizaron en un entorno urbano caracterizado por la presencia de edificaciones y condiciones topográficas que introducen atenuación en la señal. Bajo estas condiciones, el desempeño observado es consistente y valida la funcionalidad del sistema.

VII. Conclusiones

En este trabajo se desarrolló una biblioteca en lenguaje C para la gestión del módulo LoRa RN2903 mediante el microcontrolador PIC24FJ128GA705, con el objetivo de simplificar la configuración y operación del dispositivo en aplicaciones de Internet de las Cosas.

La biblioteca propuesta permite abstraer la complejidad asociada al uso de comandos AT, proporcionando funciones estructuradas para la configuración de parámetros de comunicación, así como para la transmisión y recepción de datos. Además, incorpora mecanismos de retroalimentación y detección de errores que mejoran la robustez del sistema.

Los resultados experimentales demostraron el correcto funcionamiento de la biblioteca en escenarios reales, alcanzando distancias de hasta 1.1 km con comunicación estable durante periodos prolongados. Asimismo, se validó la efectividad del mecanismo de tolerancia a fallas, logrando detectar errores de comunicación en tiempos menores a un segundo y recuperarse automáticamente sin reiniciar el sistema.

En conjunto, la biblioteca desarrollada representa una solución eficiente y accesible para la implementación de sistemas IoT basados en LoRa, facilitando su adopción por desarrolladores con conocimientos intermedios en sistemas embebidos.

Como trabajo futuro, se plantea la incorporación de técnicas de optimización energética, la inclusión de métricas adicionales de calidad de enlace y la extensión de la biblioteca para su uso en otras plataformas de microcontroladores.

Referencias

[1] Mario Cruz Vega et al. *Las tecnologías IOT dentro de la industria conectada: Internet of things*. EOI Escuela de Organización Industrial, 2015.

[2] Joan Miquel Solé. «Improving the usability of a LoRa Mesh Library». B.S. thesis. Universitat Politécnica de Catalunya, 2022.

[3] Aloÿs Augustin et al. «A study of LoRa: Long range & low power networks for the internet of things». En: *Sensors* 16.9 (2016), pág. 1466.

[4] ML Liya y D Arjun. «A survey of LPWAN technology in agricultural field». En: *2020 Fourth International Conference on I-SMAC (IoT in Social, Mobile, Analytics and Cloud)(I-SMAC)*. IEEE. 2020, págs. 313-317.

[5] Anne H Ngu et al. «IoT middleware: A survey on issues and enabling technologies». En: *IEEE Internet of Things Journal* 4.1 (2016), págs. 1-20.

[6] Microchip RN2903. *Tomado en línea:* `http://ww1.microchip.com/downloads/en/DeviceDoc/50002390E.pdf`. 2024.

[7] Luis Felipe Patiño Lopez y et al. «Diseño e implementación de un sistema embebido basado en IoT para la gestión del transporte público». 2019.

[8] Matheus Ferreira Bernardo de Souza. «Biblioteca de rede baseada em LoRa: Uma solução independente da plataforma». 2023.

[9] Rajkumar Buyya y Amir Vahid Dastjerdi. *Internet of Things: Principles and paradigms*. Elsevier, 2016.

[10] Alexandru Lavric y Valentin Popa. «Internet of things and LoRa™ low-power wide-area networks: a survey». En: *2017 International Symposium on Signals, Circuits and Systems (ISSCS)*. IEEE. 2017, págs. 1-5.

[11] Joan Miquel Solé et al. «Implementation of a LoRa mesh library». En: *IEEE access* 10 (2022), págs. 113-171.

[12] Documentación-Microchip. *Tomado en la web:* `https://www.microchip.com/en-us/development-tool/ac164164`. 2024.

[13] Microchip Technology Inc. *MPLAB® Code Configurator User's Guide*. Último acceso: 28 de agosto de 2024. 2024. URL: `https://www.microchip.com/mplab/mplab-code-configurator`.

[14] Carlos A Henao y Edison Duque. «Programando microcontroladores PIC en lenguaje C». En: *Scientia et technica* 3.43 (2009).

[15] Guía del usuario PIC-IoT WG. *PIC-IoT WG Hardware User Guide*. 2020.