PROJECT REQUIREMENTS DOCUMENT

Multi-Node Token-Ring Network Over UART

Prepared for learning purposes

Prepared by: Ameed Othman, Embedded Software Developer

10 December 2024

1. INTRODUCTION

1.1 Purpose

The purpose of this document is to clearly define the requirements for implementing a token-ring communication protocol over standard UART lines among multiple embedded boards. The token-ring design ensures deterministic, collision-free data transmission, creating a reliable, multi-node embedded network without introducing additional complex communication hardware.

1.2 Scope

This project focuses on a UART-based token-ring network that operates between three or more development boards. It leverages the Zephyr RTOS environment to provide predictable, real-time behavior. The final deliverable is a working firmware implementation, along with verification results and documentation demonstrating that the network meets the stated performance and reliability targets.

1.3 Definitions, Acronyms, and Abbreviations

- **UART:** Universal Asynchronous Receiver/Transmitter, a serial communication protocol.
- **Token-Ring Network:** A network topology where nodes form a circular data path, and a special "token" is passed to grant transmission rights.
- **CRC:** Cyclic Redundancy Check, used for error detection.
- **RTOS**: Real-Time Operating System, here specifically Zephyr.

1.4 References

- Zephyr RTOS Documentation
- Hardware Datasheets for NUCLEO-F446RE, nRF52840-DK, FRDM-K64F
- UART Protocol Standard

2. OVERALL DESCRIPTION

2.1 Product Perspective

The token-ring network is a lower-level communication infrastructure for distributed embedded systems. Each node acts as both a transmitter and a receiver in a daisy-chain configuration, collectively forming a logical ring. Unlike hub-and-spoke or bus-based topologies, the token-ring approach minimizes collisions by providing a single authorization token that grants one node at a time the right to transmit data.

2.2 Product Features

- **Collision-Free Communication:** Only the token-holding node may transmit, preventing data collisions.
- **Scalability:** Supports an initial set of three boards, with the ability to expand to more nodes.
- **Deterministic Timing:** Offers predictable token rotation intervals, ensuring timely data exchange.
- **Reliable Data Transfer:** Utilizes CRC checks and timeouts to detect and recover from transmission errors.
- **Flexible Data Content:** Transmitted frames may carry sensor readings, configuration commands, and diagnostic information.

2.3 User Characteristics

This network targets embedded developers, test engineers, and integrators who require a low-overhead, wired communication method for multi-node systems. Familiarity with basic microcontroller principles, UART signaling, and Zephyr RTOS concepts is assumed.

2.4 Constraints and Assumptions

- **Hardware Constraints:** Each board must provide at least one UART port accessible at the desired logic level (3.3 V).
- **Common Ground Reference:** All boards share a single ground reference line to ensure proper UART signal interpretation.
- **RTOS Integration:** The project must run under Zephyr RTOS, leveraging its threading and synchronization capabilities.
- **Cable Lengths:** All connections will be short (under 30 cm) during initial development, minimizing signal integrity issues.

3. SYSTEM REQUIREMENTS

3.1 Functional Requirements

- **FR-1:** The system shall form a logical token-ring network by connecting the UART TX of one board to the UART RX of the next until a closed loop is established.
- **FR-2:** The system shall implement a token frame that, when received, grants the recipient node permission to transmit its data frames.
- **FR-3:** Each node shall transmit a defined data frame when holding the token. The data frame shall include a header, payload, and CRC for error detection.
- **FR-4:** Once data transmission is complete, the node shall forward the token to the next node within a maximum predefined interval.
- **FR-5:** The system shall allow for nodes to include configurable parameters, sensor readings, or diagnostic metrics in their data frames.
- **FR-6:** The system shall recover from token loss or corruption by implementing a timeout after which a node attempts to reintroduce a clean token.
- **FR-7:** The network shall handle at least three nodes initially, with the potential to scale up to five nodes without fundamental changes.

3.2 Performance Requirements

- **PR-1:** The token rotation time (the interval required for the token to pass through all nodes) shall be deterministic within a tolerance of ±5%.
- **PR-2:** Each node's data latency (the time from when data is ready to send until it is placed on the network) shall not exceed one full token rotation interval.
- **PR-3:** The system shall operate reliably at a baud rate of at least 115200 bps.

3.3 Reliability and Error Handling

- **RR-1:** Frames shall include CRC or checksum fields to detect transmission errors.
- **RR-2:** On detecting a corrupted frame, a node shall discard the invalid data and wait for the next token.

- **RR-3:** If a node fails to receive a token within a defined timeout period, it shall attempt a token regeneration procedure.
- **RR-4:** A successful error recovery shall restore stable token passing within three token rotation cycles.

3.4 Security and Safety Considerations

- **SR-1:** The UART network is not inherently secure. For this project's scope, no encryption is required, but data integrity through CRC checks is mandatory.
- **SR-2:** The system shall detect and handle unintended disconnections gracefully, maintaining stable operation when connections are restored.

3.5 Integration with Zephyr RTOS

- **ZR-1:** The token handling logic shall run in a dedicated Zephyr thread or task with priority ensuring timely UART message processing.
- **ZR-2:** Use Zephyr synchronization mechanisms (mutexes, semaphores) to ensure thread-safe access to UART buffers.
- **ZR-3:** Leverage Zephyr's logging subsystem for status reporting, including token rotation metrics and error occurrences.

4. VERIFICATION AND VALIDATION

4.1 Test Plan

- **Functional Testing:** Verify that token passing, data frame transmission, and error handling meet the functional requirements by simulating normal operation and introducing controlled errors.
- **Performance Testing:** Measure token rotation times and verify that timing requirements are met under varying node counts and data loads.
- **Stress and Fault Injection Testing:** Introduce corrupted frames and disconnections to ensure that error recovery and token regeneration function as intended.

4.2 Acceptance Criteria

The system is deemed acceptable if it:

- Consistently rotates the token through all nodes under normal conditions.
- Demonstrates stable error recovery and token regeneration.
- Meets defined latency, throughput, and reliability requirements.

5. DOCUMENTATION AND DELIVERABLES

- **Requirements Document:** Finalized and approved prior to design and implementation.
- **Design Document:** Will include architecture diagrams, frame format specifications, and a state machine representation of token passing.
- **Implementation:** Source code integrated into Zephyr RTOS, adhering to coding standards and best practices.
- **Test Reports:** Detailed results from functional, performance, and stress testing.
- **User & Maintenance Manual:** Guides for configuration, integration of new nodes, and long-term maintenance.