Software Requirements Specification

For

Firmware Over-The-Air Update in FreeRTOS

Version 1.0

Prepared By

Jinu Raju

Kiran Thomas George Tharakan

Navaneeth Sunil

Sharan K Lakshman

TKM College Of Engineering

16 March 2023

Table of Contents

4. System Features

5. Other Nonfunctional Requirements

1. Introduction

1.1 Purpose

The purpose of this document is to define the requirements for a reusable library for FreeRTOS with an update portal, based on the reference design from BalenaOS runtime and over-the-air updates.

1.2 Document Conventions

This document is based upon:
- IEEE 830-1998 - IEEE Recommended Practice for Software Requirements Specifications
- IEEE 830-1984 - IEEE Guide for Software Requirements Specifications

1.3 Intended Audience and Reading Suggestions

The intended audience for the SRS document of the project "Firmware Over-The-Update in FreeRTOS" includes project managers, developers, quality assurance teams, testers, and customers/stakeholders.

1.4 Product Scope

This library is intended to provide a solution for low resource hardware modules that do not support POSIX-based systems. It will enable devices to be remotely updated without requiring physical access or manual intervention.

1.5 References

https://www.freertos.org/ota/index.html
https://www.freertos.org/2022/01/delta-over-the-air-updates.html
https://www.balena.io

2. Overall Description

2.1 Product Perspective

The Firmware Over-The-Air (FOTA) update module in FreeRTOS is a software library that enables remote updates of firmware on embedded devices running FreeRTOS operating system. The module will work as an independent service and will be designed to integrate with the existing FreeRTOS architecture.

Tools : ArduinoIDE , ESP32 library for ArduinoIDE, ESP32 devkit ,FreeRTOS-library

2.2 Product Functions

The main function of the FOTA update module in FreeRTOS is to provide a mechanism for updating firmware over the air, allowing embedded devices to receive firmware updates without the need for physical access or manual intervention. The module will also provide the following functions:

-Checking for new firmware updates

-Downloading firmware updates over the air

-Verifying the integrity of downloaded firmware updates

-Applying firmware updates to the device

-Logging update activity for auditing and troubleshooting purposes

2.3 User Classes and Characteristics

The target users of the FOTA update module in FreeRTOS are developers and system integrators working on embedded systems. The users are expected to have experience with embedded system development, including knowledge of C programming and FreeRTOS.

2.4 Operating Environment

The FOTA update module in FreeRTOS is designed to operate in a low-resource environment with limited processing power, memory, and storage. It is intended to be used with embedded devices running FreeRTOS as their operating system. The module will support wireless communication protocols such as Wi-Fi and Bluetooth Low Energy (BLE).

2.5 Design and Implementation Constraints

Hardware Limitations: As the library is designed for low-resource hardware modules, the library must be developed to operate within the constraints of the hardware, such as limited processing power and memory.

Network Connectivity: The library relies on network connectivity for OTA updates, so the hardware module must be connected to a reliable network connection for the updates to be successful.

Security: The update process must be secure and must include encryption of data, certificate validation, and secure storage of sensitive information.
Compatibility: The library must be designed to work with different types of hardware modules and must be compatible with different versions of FreeRTOS.

Time Constraints: OTA updates must be performed quickly and with minimal disruption to the device's operation.

Scalability: The library must be designed to handle a large number of devices, as the update portal may be used by multiple devices at once.

Modularity: The library must be designed with modularity in mind, allowing for easy integration into different systems and configurations.

Testing: The library must be thoroughly tested to ensure its reliability and security. This library was implemented and tested in ArduinoIDE.

Maintenance: The library must be designed for easy maintenance and updates, allowing for bug fixes and new features to be added as needed.

The FOTA update module in FreeRTOS must be designed to work within the resource limitations of embedded devices running FreeRTOS. The module must also be able to handle errors and interruptions during the firmware update process. The module will be developed as a reusable library for FreeRTOS and must be compatible with the existing FreeRTOS architecture.

2.6 User Documentation

The FOTA update module in FreeRTOS will come with comprehensive user documentation, including a user manual and API documentation. The documentation will cover installation, configuration, and usage of the module.

2.7 Assumptions and Dependencies

Assumptions

-The target device has enough storage space to store the updated firmware. Target device has an internet connection to download the update.

-Firmware update files are in specific format compatible with the library. The update is tested and validated before it is deployed.

-FreeRTOS assumes that the target embedded devices have wireless connectivity and support the FreeRTOS operating system.

Dependencies

Native freeRTOS library
Standard C library

3. External Interface Requirements

3.1 User Interfaces

The user interface for the Firmware over the air update module in FreeRTOS will be a web-based portal that allows users to initiate and monitor over-the-air firmware updates. The portal will provide a simple and intuitive interface that can be accessed from any device with a web browser.

3.2 Hardware Interfaces

The Firmware over the air update module in FreeRTOS will interface with the hardware through a bootloader that is responsible for updating the firmware. The bootloader will be designed to work with a variety of hardware platforms, including low resource hardware modules that do not support POSIX based systems.

Supported device types : All embedded devices running FreeRTOS.

3.3 Software Interfaces

The Firmware over the air update module in FreeRTOS will interface with other software components through a set of well-defined APIs. These APIs will provide a standard interface for the kernel to communicate with the firmware update server and other system components and modules. Data items passed between server and the IOT device include binary files containing the updates, login credentials.

Communication protocols : HTTP , FTP , IEEE 802.11 (for wifi)

4. System Features

4.1 Firmware Compatibility:

The OTA update system should be able to identify the specific device model and version and determine the appropriate firmware version that is compatible with the device.

The system may also need to consider any specific hardware or software configurations on the device that could affect compatibility. The system should be able to download and install the correct firmware version automatically, without requiring any user intervention.

4.2 Update Scheduling:

The OTA update system should be able to schedule updates for devices based on a variety of factors, such as device availability, network bandwidth, and user preferences.

For example, the system may schedule updates during off-peak hours when network usage is low and devices are less likely to be in use. The system should be able to prioritize critical updates and ensure that they are installed promptly.

4.3 Network Connectivity:

The OTA update system should be able to establish a reliable network connection with the device, using a secure and encrypted protocol to protect against data breaches or unauthorized access.

The system should be able to monitor the connection throughout the update process and automatically resume the update if the connection is interrupted.

4.4 Error Handling:

The OTA update system should be designed to handle any errors that may occur during the update process. This could include network interruptions, power failures, or other hardware or software issues.

The system should be able to detect these errors, alert the user or administrator, and automatically resume the update process from where it left off once the issue is resolved.

4.5 Verification and Rollback:

After the firmware update is complete, the OTA update system should verify the integrity of the updated firmware to ensure that it is functioning correctly. The system should be able to test key functions and features of the device to confirm that they are working as expected.

If any issues are detected, the system should be able to roll back to the previous firmware version automatically, without requiring any user intervention. This helps ensure that the device continues to function correctly even if there are issues with the updated firmware.

5. Other Nonfunctional Requirements

5.1 Security Requirements

The OTA update process should be secure and reliable to prevent any
unauthorized access or tampering of the device, as this is critical for ensuring the integrity of the device and its data

5.2 Reliability Requirements

The OTA update process should be reliable and robust to minimize the risk
of failures or errors during the update process, as this could lead to downtime,loss of data, and other serious issues.

5.3 Performance Requirements

The system should have low latency and high throughput to ensure timely and efficient updates,as this is essential for minimizing the time required to update devices.

5.4 Compatibility

The system should be compatible with various hardware and software configurations to support a wide range of devices, as this is essential for ensuring that the OTA update process can be used across a diverse set of devices

5.5 Maintainability

The OTA update process should be easy to maintain and update over time, with minimal impact on existing systems and processes, as this is essential for ensuring that the OTA update process can be updated and improved over time as needed.