Software requirements specification for project “STM32-bootloader”

1. Authors

Bortnikov Pavel
Tarasov Artem
Vasilev Pavel
Patrushev Boris

2. Introduction

The main idea of this project is to develop a bootloader and a high-level bootloader utility program for the STM32, where the end user can choose a file with a new version of the program and then send it to the STM32. Then STM32 receives the program and stores it in the flash memory. For security reasons the transferred program is encrypted (it will be decrypted by the bootloader on the STM32 chip). Also the high-level program will be able to upload the bootloader to the device (for the updates to devices that don’t have it yet). In other words, the end user will have to just drag and drop the bootloader/program file to the utility program on their PC, choose the upload options.

3. Glossary

[All the unobvious project- and domain-related terms with their definitions/descriptions comes here.]

bootloader - the program on the STM32 microcontroller that can download or upload data (programs or any other data)
device - an embedded system that has an STM32 chip to run it’s main program
device’s software - the main program on the device that’s supposed to run on it.
flasher - the utility program on the end user’s PC.

4. Actors

[All the actors come here. For each actor it should be defined for its role and the general description of its goals and responsibilities within the given system.]

1. **Name:** End user
   Goal: Updating software/installing the bootloader
   Responsibilities: Literally follow instructions and click the buttons on the PC
2. **Name:** Device developer
   Goal: get the updates to the end user
   Responsibilities: they have to provide the end user with the device’s software (and its updates) and the flasher

5. Functional requirements

5.1. Strategic Use-cases

[Optional. White-level use-cases. This section is useful when there are too many blue-level use-cases and they should be grouped somehow.]

5.1.1. Use-case <UC-S-1>

Name: Software update
Actor: End user, device developer
Goal: To update the device’s software
Trigger condition: Outdated device’s software
Precondition/Context: End user must have a PC and flasher installed on it

Extensions: Flasher installation

Main success scenario:

1. Device developer notifies end users that there is a new version of the device's software
2. End user must download the device's software from the internet
3. Then they open the flasher and choose the downloaded file.
4. Also they need to connect the device to the computer by USB-UART converter.
5. After all they need to click the button "upload" and wait for the device's program to be uploaded to the device.

Alternative scenarios:

5.1.2. Use-case <UC-S-2>

Name: Flasher installation

Actor: End user

Goal: To install the flasher

Trigger condition: They need to update software on the device, but they don't have the flasher on the PC

Precondition/Context: End user must have a PC

Extensions:

Main success scenario:

1. End user opens website with a program
2. Download it
3. Execute the installation program
4. Press ~~F to pay respect~~ repeatedly "Next" button by choosing some options.
5. Click "OK" to exit from installer

Alternative scenarios:

5.1.3. Use-case <UC-S-3>

5.2. Use-cases for <actor1>

[In case the white-level use-cases are defined, here could be one additional level that groups blue-level use-cases by white-level ones, in addition to grouping by actors.]

5.2.1. Use-case <UC-1-1>

[Full UC description]

Actors: *[essential]*

Goals: *[essential, goals for each actor]*

Precondition: *[optional, conditions that must be met within the system for this UC to be performable]*

Trigger condition: *[optional, action(-s) that triggers this UC]*

Extensions: *[optional, other UCs related to this one, i.e. triggered by this one]*

Mains success scenario: *[essential]*

1) <Action1>

2) <Action2>

3) …

Alternative scenario <scenario-name1>: *[optional]*

[Trigger condition for <scenario-name1>]

1) <Action1>

2) <Action2>

3) …

Alternative scenario <scenario-name2>: *[optional]*

…

<Additional section>:

Notes: *[optional, any other useful information about the UC]*

5.2.2. Use-case <UC-1-2>

[Reference-only description, applicable when there are multiple actors for the UC, and the full description provided in other actor's section]

Actors: *[essential]*

The detailed description is provided in <…>

5.3. Use-cases for <actor2>

…

6. System-wide functional requirements

[Optional. System-wide functional requirements that weave with multiple use-cases. Examples: authorization, audit]

7. Non-functional requirements

[All the subsections are optional.]

7.1. Environment

[Environment requirements are limitations for hardware and software usage including supported hardware platforms, networking infrastructure and protocols, programming languages, libraries and external services]

7.2. Performance

[The performance characteristics of the system should be outlined in this section. Examples are response time, throughput, capacity and startup or shutdown times.]

7.3. Reliability

[Reliability includes the product and/or system's ability to keep running under stress and adverse conditions. Specify requirements for reliability acceptance levels, and how they will be measured and evaluated. Suggested topics are availability, frequency of severity of failures and recoverability.]

7.4. Extensibility

[This section indicates requirements that will enhance the extensibility including extension points, compatibility, scalability, configurability]