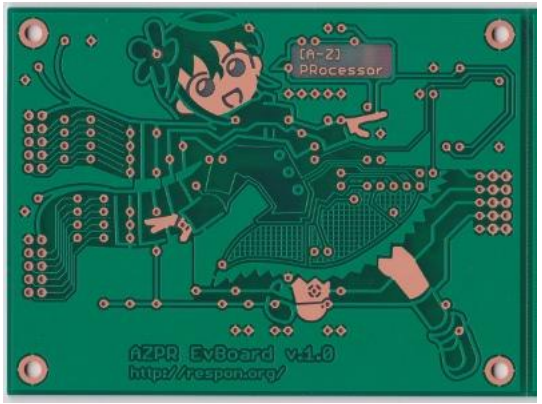
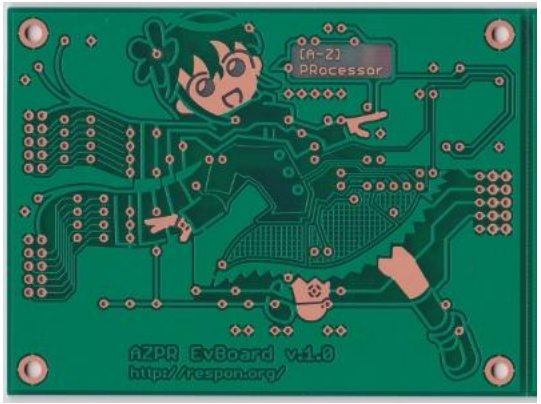
Novosibirsk State University
Department of Information Technology
Computer Science & System Design

STM32 Bootloader



Mentor: Litvinov Vasiliy
Team members:
Tarasov Artem
Vasilev Pavel
Patrushev Boris



What is a bootloader?

Bootloader is software that is responsible for booting a chip. When you upload your code on flash and restart a chip, default STM32 bootloader executes the code from specific address. After that, the chip under is control of your program.

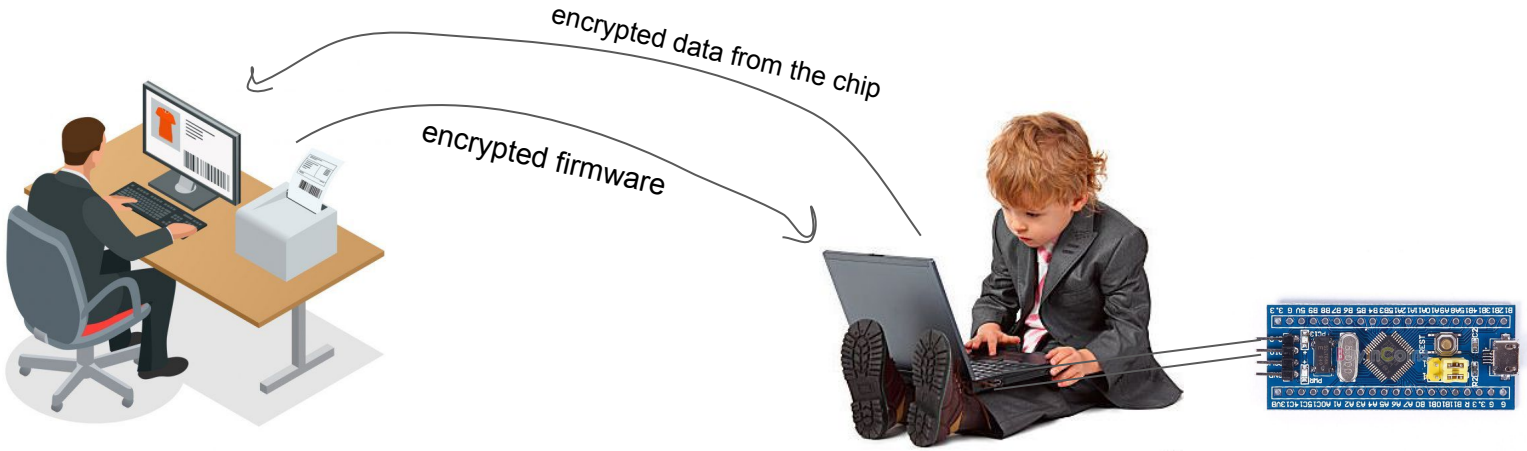
Why do we need a custom bootloader?

- This is hard for user with no knowledge about programming to upload new firmware on the chip
- Sometimes it’s important to keep your code a secret so no one can steal it (intellectual property)

What is the decision?

The decision is to create:

- Custom **bootloader** with encryption and with different communication interface support (UART, SPI, USB and so on)
- A program for PC (called **flasher**) that allows people easily communicate with bootloader.
- Console applications for developers that can help easily upload bootloader on different STM32 chips and make their firmware compatible with bootloader (**bootloader and firmware modifiers**)
- A program for encrypting firmware files (**bootloader file generator**)



Requirements

- Security
- Reliable transmission from user’s computer to STM32 and vice versa
- Small size (our bootloader is about 10-11kB)
- Verification of the uploaded firmware
- Possibility to receive encrypted data from bootloader to developer
- Simple Flasher GUI

Security

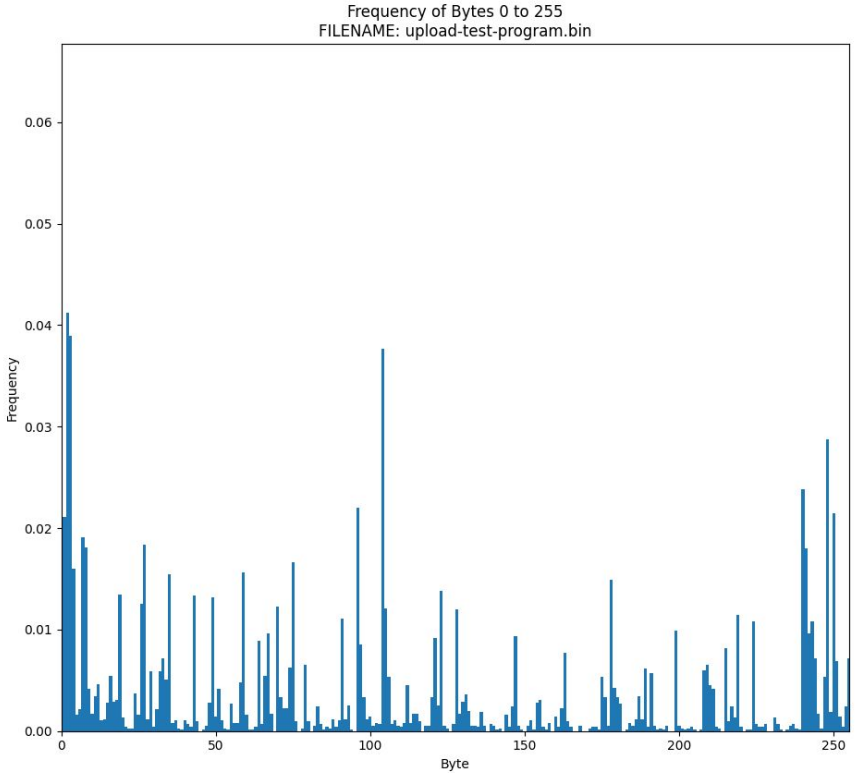
We use AES 256 encryption. This is standard block cipher for symmetric encryption.

When developers create a device with STM32, they put bootloader with embedded secret key and disable reading flash memory from the chip. So now any firmware can be uploaded only via the bootloader.

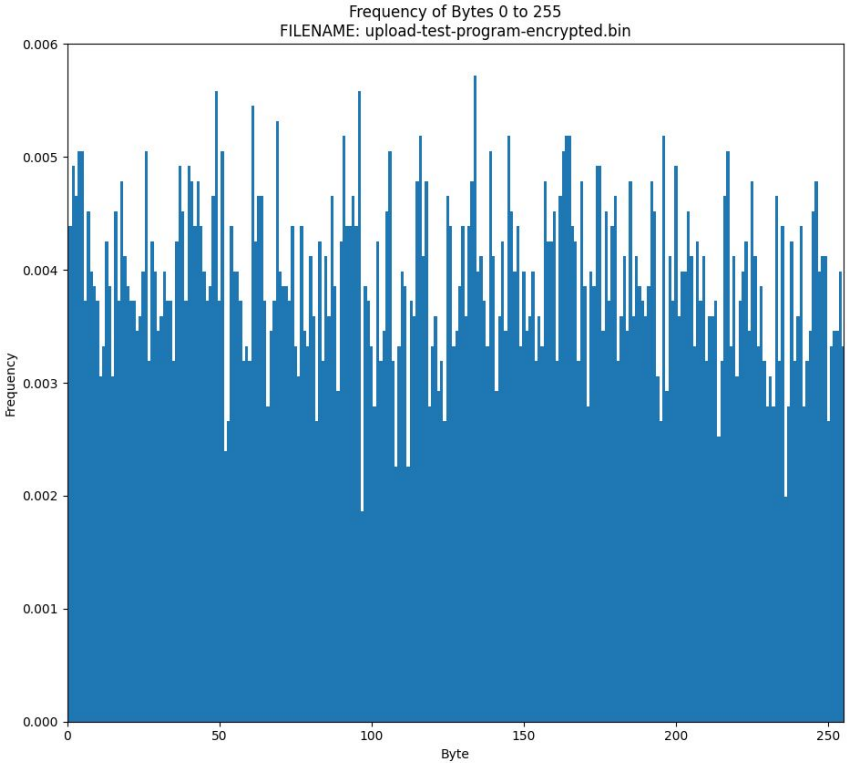
If developers want to upload new code for this device, they encrypt the binary file using **bootloader file generator** and then send it to the user. And then the user uploads new code on STM32 through flasher.

As you can see, only the developer and STM32 chip have access to unencrypted data.

Unencrypted:
Shannon Entropy = 6.542

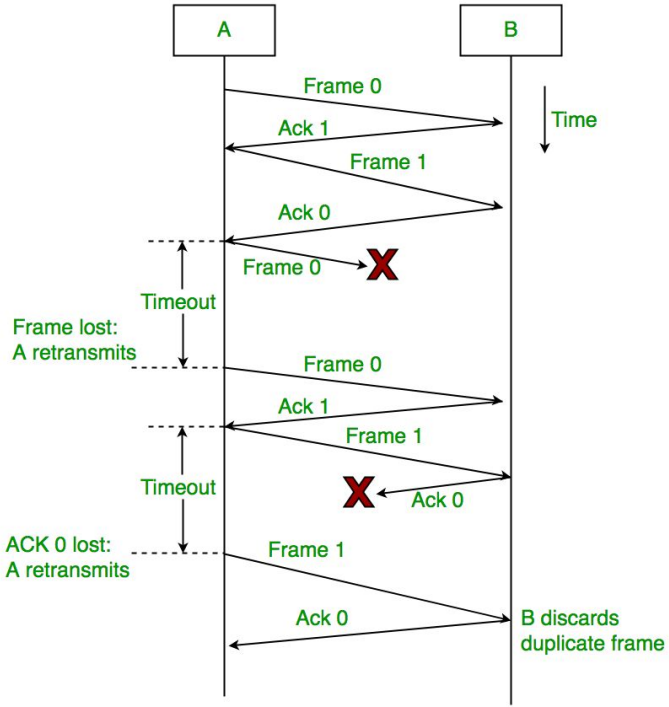


Encrypted:
Shannon Entropy = 7.976



Bootloader and flasher communication

Bootloader and Flasher communicates using STOP and WAIT ARQ protocol to ensure that all data is delivered correctly.



Bootloader and flasher

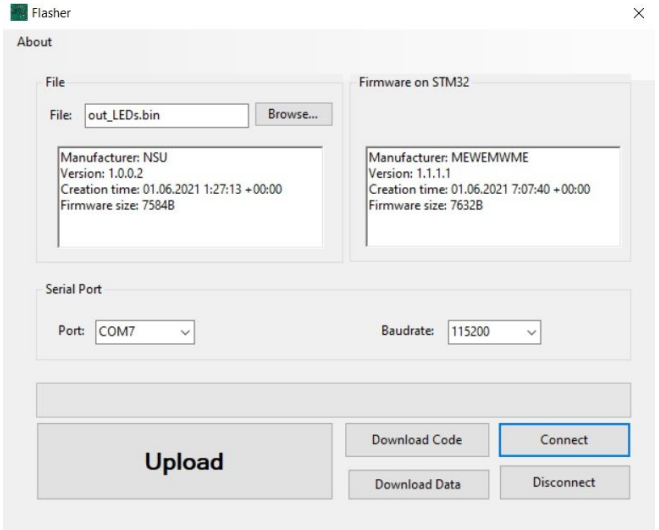
Typical scenario:

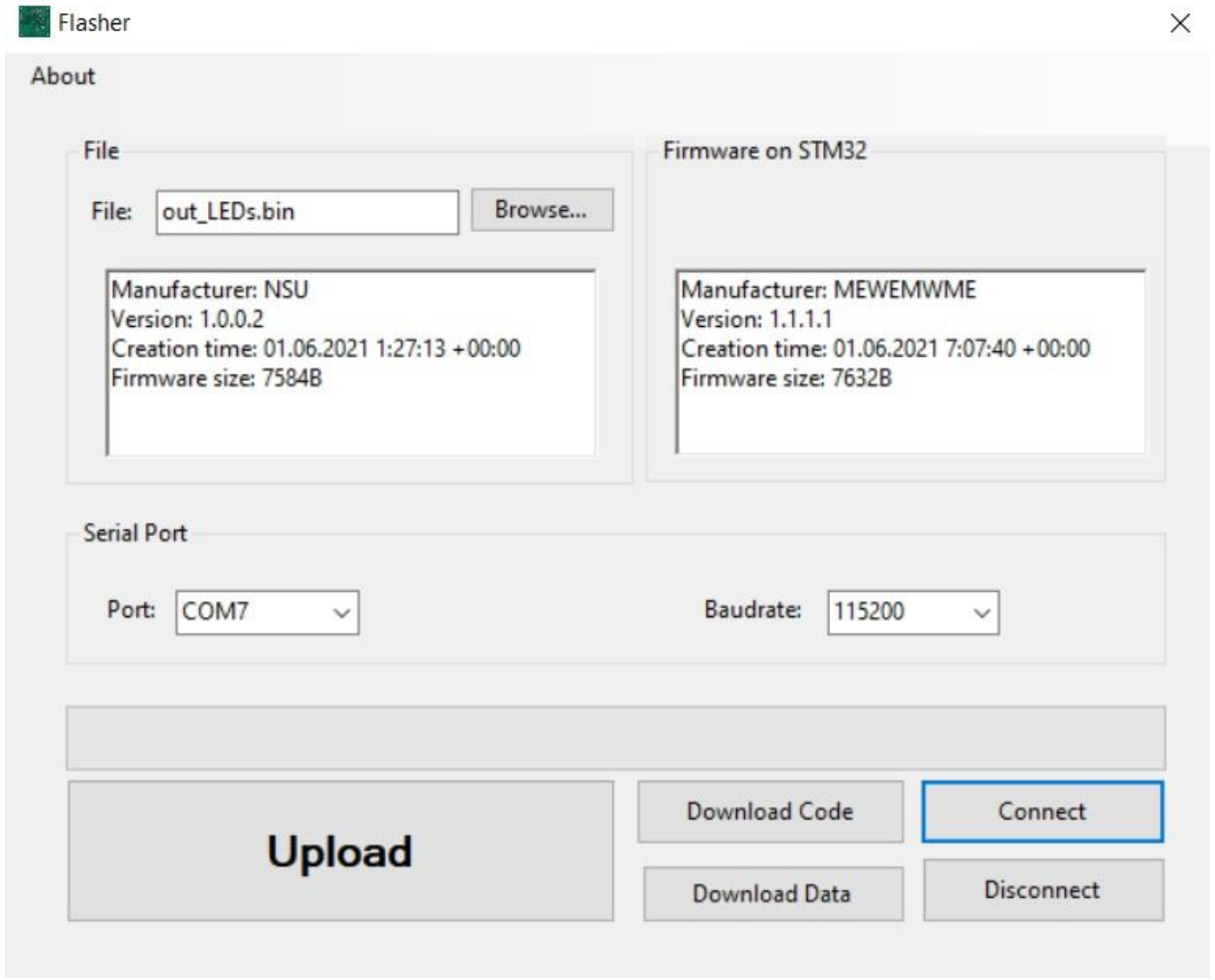
Give me information about current firmware

Current information of the firmware

There's new firmware for you

Ok, got it




Flasher
About
File
File:
out_LEDs.bin
Browse...
Manufacturer: NSU
Version: 1.0.0.2
Creation time: 01.06.2021 1:27:13 +00:00
Firmware size: 7584B
Firmware on STM32
Manufacturer: MEWEMWME
Version: 1.1.1.1
Creation time: 01.06.2021 7:07:40 +00:00
Firmware size: 7632B
Serial Port
Port:
COM7
Baudrate:
115200
Upload
Download Code
Connect
Download Data
Disconnect

Firmware and bootloader modifier

- Firmware modifier changes project files to make firmware compatible with bootloader
- Bootloader modifier helps the developer to build a bootloader for their specific chip and communication interface
- Bootloader file generator creates a file that contains the encrypted executable binary for STM32, metadata and checksums. This file format is used by flasher.

Project member contribution

33% - Artyom Tarasov

33% - Boris Patrushev

33% - Pavel Vasilev

1% - Pavel Bortnikov

Demonstration

Thank for your attention!