

Flasher Secure

Secured Production at
Contract Manufacturers

User Guide & Reference Manual

Document: UM08032
Software Version: 1.12
Revision: b
Date: July 2, 2021



A product of SEGGER Microcontroller GmbH

www.segger.com

Disclaimer

Specifications written in this document are believed to be accurate, but are not guaranteed to be entirely free of error. The information in this manual is subject to change for functional or performance improvements without notice. Please make sure your manual is the latest edition. While the information herein is assumed to be accurate, SEGGER Microcontroller GmbH (SEGGER) assumes no responsibility for any errors or omissions. SEGGER makes and you receive no warranties or conditions, express, implied, statutory or in any communication with you. SEGGER specifically disclaims any implied warranty of merchantability or fitness for a particular purpose.

Copyright notice

You may not extract portions of this manual or modify the PDF file in any way without the prior written permission of SEGGER. The software described in this document is furnished under a license and may only be used or copied in accordance with the terms of such a license.

© 2021 SEGGER Microcontroller GmbH, Monheim am Rhein / Germany

Trademarks

Names mentioned in this manual may be trademarks of their respective companies.

Brand and product names are trademarks or registered trademarks of their respective holders.

Contact address

SEGGER Microcontroller GmbH

Ecolab-Allee 5
D-40789 Monheim am Rhein

Germany

Tel. +49-2173-99312-0
Fax. +49-2173-99312-28
E-mail: support@segger.com
Internet: www.segger.com

Manual versions

This manual describes the current software version. If you find an error in the manual or a problem in the software, please inform us and we will try to assist you as soon as possible. Contact us for further information on topics or functions that are not yet documented.

Print date: July 2, 2021

Software	Revision	Date	By	Description
1.12		210405	AW	Overall improvements
1.12	b	210201	AB	Fixed typos
1.12	a	190906	MF	Minor change server hosting
1.12		190603	MF	Added LogLevel command line argument
1.11	a	180910	MF	Added further error message for better analysis
1.10	b	180910	MF	
1.10	a	180907	MF	Corrected REST API description for manufacturer features
1.00	k	180830	MF	Added REST API description for user, manufacturer and device features, update web interface description
1.00	h	180703	MF	New feature for running as windows service
1.00	g	180618	MF	Bugfixes for Firmware Import and Project Create
1.00	f	180614	MF	Added REST API description for project features
1.00	b	180507	AB/ MF	Minor corrections after review
1.00	a	180504	AB/ MF	Added FlasherSecureServer feature
1.00	0	170401	AB	Initial release.

About this document

Assumptions

This document assumes that you already have a solid knowledge of the following:

- The software tools used for building your application (assembler, linker, C compiler).
- The C programming language.
- The target processor.
- DOS command line.

If you feel that your knowledge of C is not sufficient, we recommend *The C Programming Language* by Kernighan and Richie (ISBN 0--13--1103628), which describes the standard in C programming and, in newer editions, also covers the ANSI C standard.

How to use this manual

This manual explains all the functions and macros that the product offers. It assumes you have a working knowledge of the C language. Knowledge of assembly programming is not required.

Typographic conventions for syntax

This manual uses the following typographic conventions:

Style	Used for
Body	Body text.
Keyword	Text that you enter at the command prompt or that appears on the display (that is system functions, file- or pathnames).
Parameter	Parameters in API functions.
Sample	Sample code in program examples.
Sample comment	Comments in program examples.
Reference	Reference to chapters, sections, tables and figures or other documents.
GUIElement	Buttons, dialog boxes, menu names, menu commands.
Emphasis	Very important sections.

Table of contents

1	Introduction	10
1.1	Flasher Secure overview	11
1.1.1	Features of Flasher Secure	11
1.1.2	Working environment	11
1.2	Specifications	13
1.2.1	Specifications for Flasher Secure	13
1.2.1.1	Flasher Secure download speed	14
1.2.1.2	Supported Target interfaces	14
1.2.2	Specifications for Flasher Secure Server	14
2	The concept of the Flasher Secure	15
2.1	Security based on emSecure	16
2.2	Third party solutions	17
2.2.1	Secure firmware installation (ST)	17
3	Flasher Secure Server	18
3.1	Installation	19
3.1.1	Hardware Requirements	19
3.1.2	Connection Requirements	19
3.1.3	Installation on a Linux system	19
3.1.4	Installation on a Windows system	19
3.1.5	Command line arguments	20
3.1.6	Server Certificates	20
3.2	Projects in the Flasher Secure Server	21
3.3	Web interface	22
3.3.1	Login	22
3.3.2	Main page	23
3.3.3	Projects	24
3.3.3.1	Create a new project	24
3.3.3.2	Project Edit	25
3.3.3.3	Delete Project	26
3.3.4	Users	27
3.3.4.1	User Overview	27
3.3.4.2	Adding a user account	28
3.3.4.3	Editing a user account	29
3.3.4.4	Deleting a user account	30
3.3.4.5	Activating a user account	31
3.3.4.6	Deactivating a user account	32
3.3.5	Firmware	33

3.3.5.1	Firmware Overview	33
3.3.5.2	Firmware upload (emSecure)	34
3.3.5.3	Firmware upload (ST SFI)	35
3.3.5.4	Firmware settings	36
3.3.6	Manufacturers	37
3.3.6.1	Adding a manufacturer account	38
3.3.6.2	Editing a manufacturer account	39
3.3.6.3	Deleting a manufacturer account	40
3.3.6.4	Activating a manufacturer account	41
3.3.6.5	Deactivating a manufacturer account	42
3.3.6.6	Resetting a manufacturer's success rate counter	43
3.3.7	Devices	44
3.3.7.1	Devices overview	44
3.3.7.2	Device list	45
3.3.7.3	Exporting the device list	46
3.3.8	Server status	47
3.3.8.1	Server Access Log	48
3.4	REST interface	49
3.4.1	General	49
3.4.2	REST API	49
3.4.2.1	Creating a project	51
3.4.2.2	Deleting a project	53
3.4.2.3	Activating a project	54
3.4.2.4	Stopping a project	55
3.4.2.5	Finishing a project	56
3.4.2.6	Moving a project to prepare state	57
3.4.2.7	Importing firmware	58
3.4.2.8	Adding a user account	60
3.4.2.9	Activating a user account	61
3.4.2.10	Deactivating a user account	62
3.4.2.11	Deleting a user account	63
3.4.2.12	Editing a user account	64
3.4.2.13	Adding a manufacturer account	65
3.4.2.14	Activating a manufacturer account	66
3.4.2.15	Deactivating a manufacturer account	67
3.4.2.16	Deleting a manufacturer account	68
3.4.2.17	Editing a manufacturer account	69
3.4.2.18	Exporting a device list as CSV	70
3.4.2.19	Exporting a device list as TXT	71
3.4.3	REST API Result Codes	72
4	Flasher Secure unit	80
4.1	General	81
4.2	IP configuration	82
4.2.1	Manual IP configuration	82
4.2.1.1	IP configuraton via J-Link Configurator	82
4.2.1.2	IP configuration via integrated web server	84
4.3	Project Configuration	85
4.3.1	Configuration files	85
4.3.2	Accessing the configuration files	85
4.3.2.1	Access via USB MSD mode	86
4.3.2.2	Access via FTP	86
4.3.2.3	Access via ASCII command protocol	86
4.4	Programming devices	87
4.5	LED status indicators	88
4.6	Web interface	89
4.6.1	Flasher Secure main page	89
4.6.2	Flasher Secure network information page	90
4.6.3	Flasher Secure network configuration	90

4.6.4	Flasher Secure system information page	91
4.6.5	Flasher Secure emulator status page	91
4.6.6	Flasher Secure About page	92
4.7	Additional Information	93
4.7.1	Patch file support	93
4.7.2	PC-based mode	93
5	Remote control	94
5.1	Overview	95
5.2	Handshake control	96
5.3	ASCII command interface	97
5.3.1	Introduction	97
5.3.2	General command and reply message format	97
5.3.3	General usage	97
5.3.4	Settings for ASCII interface via RS232	97
5.3.5	Settings for ASCII interface via Telnet	97
5.3.6	Commands and replies	98
5.3.6.1	Commands to the Flasher	100
5.3.6.1.1	#AUTO	100
5.3.6.1.2	#AUTO NOINFO	100
5.3.6.1.3	#BAUDRATE<Baudrate>	100
5.3.6.2	File I/O commands	102
5.3.6.3	Replies from Flasher	105
6	Device Firmware	107
6.1	emSecure Package	108
6.2	Implementing emSecure	109
6.2.1	Package content:	109
6.2.2	Include directories	109
6.3	Sample Application	110
6.3.1	The main function	113
6.3.2	The _VerifyRSA function	113
6.3.3	The _VerifyECDSA function	113
7	Hardware and Adapters	114
7.1	ARM 20-pin JTAG/SWD Connector	115
7.1.1	Pinout JTAG	115
7.1.2	Pinout SWD	116
7.1.3	Target power supply	117
7.2	Flasher RX 14-pin connector	118
7.2.1	Target power supply	119
7.3	Flasher PPC 14-pin connector	120
7.4	Target board design	121
7.4.1	Pull-up/pull-down resistors	121
7.4.2	RESET, nTRST	121
7.5	Adapters	122
7.5.1	JTAG Isolator	122
7.5.1.1	Pinout	122
7.5.2	J-Link Needle Adapter	123
7.6	How to determine the hardware version	124
8	Support	125
8.1	Contacting support	126
9	Glossary	127
10	Literature and references	130

Chapter 1

Introduction

This chapter provides a short overview about the Flasher Secure.

1.1 Flasher Secure overview

Flasher Secure is a mass production programming system, capable of protecting the vendor's IP regardless of the production site. It provides full control over the programming process at contract manufacturers (CM) and similar environments.

- Authenticated production with full visibility
- Production volume control
- CM administration and setup portal (Flasher Secure Server)
- Ultra fast programming
- Supports Cortex-M, RX, PPC
- Prevents production of counterfeit units
- No overhead in programming time
- Secures production at contract manufacturers

In stand-alone mode, Flasher Secure can be operated via the start/stop button. Flasher Secure connects to a PC or a network using the USB/Ethernet/RS232 interface. It can be controlled by handshake control or by ASCII command protocol. Using the USB connection requires the J-Flash tool (for further details, please refer to the *J-Flash manual* on page 130).

Flasher Secure has a 20-pin connector. Many adapters are available. See chapter *Hardware and Adapters* on page 114.

1.1.1 Features of Flasher Secure

- Three boot modes: PC-based mode, stand-alone mode, MSD mode
- Stand-alone programmer (Once set up, Flasher Secure can be controlled without the use of a PC program)
- No power supply required, powered through USB
- Supports internal and external flash devices
- Approximately 100 MB memory available for target program storage

Flasher model	Supported cores	Supported target interfaces	Flash programming speed (depending on target hardware)
Flasher Secure	Cortex-M/RX/PPC	JTAG, SWD	between 170 and 300 Kbytes/second

1.1.2 Working environment

General

The Flasher can operate from a PC with an appropriate software like J-Flash or in stand-alone mode.

Host System

IBM PC/AT or compatible CPU: 486 (or better) with at least 128MB of RAM, running Microsoft Windows 7, Windows 8, Windows 10. It needs to have a USB, Ethernet or RS232 interface available for communication with Flasher.

Power supply

Flasher Secure: 5V DC, min. 100 mA via USB connector.

Host System for the Flasher Secure Server

- IBM PC/AT or compatible CPU: 486 (or better) running at 1GHz or faster
- At least 8 GB of RAM
- Operating System: Microsoft Windows 7, Windows 8, Windows 10 or Linux
- Ethernet interface available for communication with Flasher Secure.

Installing Flasher PC-software (J-Flash)

The latest version of the J-Flash software, which is part of the J-Link software and documentation package, can always be downloaded from our website:

segger.com/jlink-software.html For more information about using J-Flash please refer to [UM08003_JFlashARM.pdf](#) (J-Flash user guide) which is also available for download on our website.

1.2 Specifications

1.2.1 Specifications for Flasher Secure

General	
Supported OS	Microsoft Windows 7 Microsoft Windows 7 x64 Microsoft Windows 8 Microsoft Windows 8 x64 Microsoft Windows 10 Microsoft Windows 10 x64
Operating Temperature	+5 °C ... +60 °C
Storage Temperature	-20 °C ... +65 °C
Relative Humidity (non-condensing)	<90% rH
Mechanical	
Size (without cables)	121mm x 66mm x 30mm
Weight (without cables)	119g
Available interfaces	
USB Host interface	USB 2.0, full speed
Ethernet Host interface	10/100 MBit
RS232 Host interface	RS232 9-pin
Target interface	JTAG 20-pin (various adapters available)
Target Interface, Electrical	
Power Supply	USB powered, 100mA for Flasher Secure. 500 mA if target is powered by Flasher Secure
Target interface voltage (VIF)	1.2 ... 5V
Target supply voltage	Supply voltage is 5V, max.
Target supply current	max. 400mA
Reset Type	Open drain. Can be pulled low or tristated
Reset low level output voltage (VOL)	$VOL \leq 10\%$ of VIF
For the whole target voltage range ($1.8V \leq VIF \leq 5V$)	
LOW level input voltage (VIL)	$VIL \leq 40\%$ of VIF
HIGH level input voltage (VIH)	$VIH \geq 60\%$ of VIF
For $1.8V \leq VIF \leq 3.6V$	
LOW level output voltage (VOL) with a load of 10 kOhm	$VOL \leq 10\%$ of VIF
HIGH level output voltage (VOH) with a load of 10 kOhm	$VOH \geq 90\%$ of VIF
For $3.6 \leq VIF \leq 5V$	
LOW level output voltage (VOL) with a load of 10 kOhm	$VOL \leq 20\%$ of VIF
HIGH level output voltage (VOH) with a load of 10 kOhm	$VOH \geq 80\%$ of VIF
JTAG Interface, Timing	
Max. JTAG speed	up to 15MHz
Data input rise time (Trdi)	$Trdi \leq 20ns$

Data input fall time (Tfdi)	Tfdi ≤ 20ns
Data output rise time (Trdo)	Trdo ≤ 10ns
Data output fall time (Tfdo)	Tfdo ≤ 10ns
Clock rise time (Trc)	Trc ≤ 10ns
Clock fall time (Tfc)	Tfc ≤ 10ns

1.2.1.1 Flasher Secure download speed

The following table lists the Flasher Secure performance values for writing to memory (RAM) via the JTAG interface:

Hardware	ARM memory download
Flasher Secure	~720 Kbytes/s (15MHz JTAG)

Note

The actual speed depends on various factors, such as JTAG, clock speed, host CPU core etc.

1.2.1.2 Supported Target interfaces

The Flasher Secure supports the following target interfaces:

- JTAG
- SWD
- FINE
- SPD

1.2.2 Specifications for Flasher Secure Server

General	
Supported OS	Microsoft Windows 7 Microsoft Windows 7 x64 Microsoft Windows 8 Microsoft Windows 8 x64 Microsoft Windows 10 Microsoft Windows 10 x64 Linux

Chapter 2

The concept of the Flasher Secure

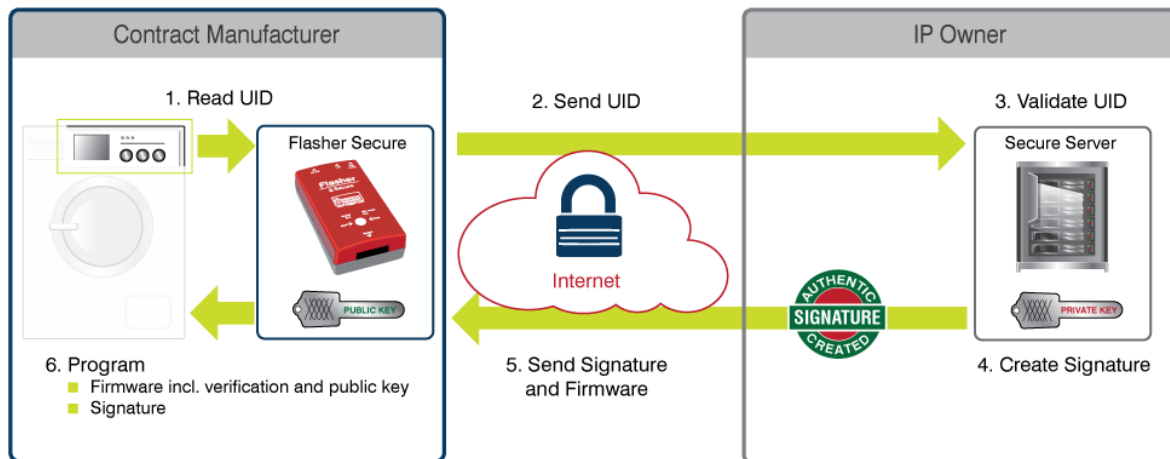
The Flasher Secure is a mass production tool, intended to be used at an external manufacturing company. The goal is to protect the intellectual property against unauthorized copying.

It consists of a server, which runs in a trusted environment. The server should be hosted by the ip owner. The second part is a client in the form of a Flasher Secure device or a PC application. An IP connection via Internet between the server and the client is mandatory.

The following sub-chapters will explain the security concepts of the Flasher Secure solution.

2.1 Security based on emSecure

Most of today's devices provide a "unique identifier" (UID). The UID is factory programmed by the chip vendor and cannot be altered. This characteristic in combination with asymmetric cryptography based on SEGGER's digital signature suite emSecure makes it possible to generate a digital signature unique for each device. The digital signature can be verified by the firmware running on the device, so using the firmware on another device will result in a signature failure.



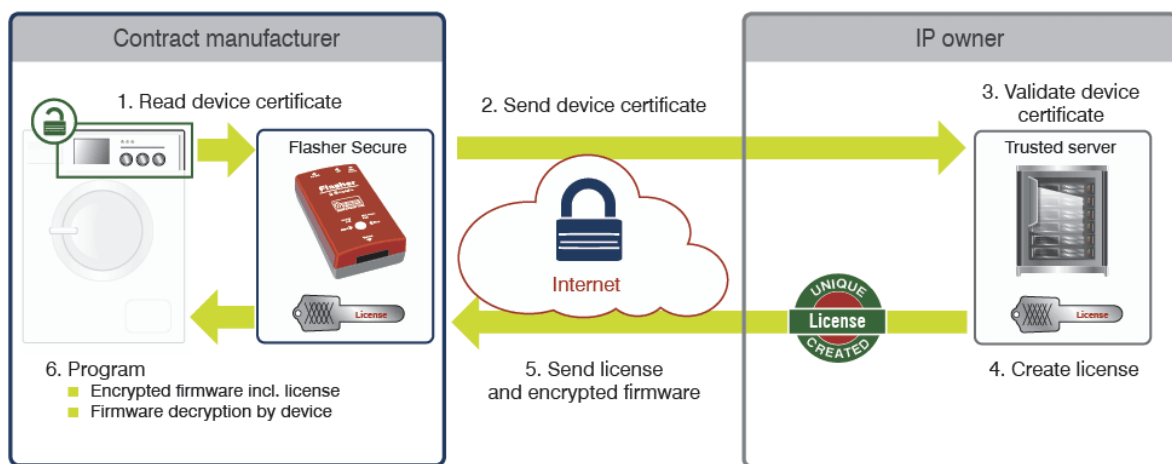
The Flasher Secure server supports emSecure-RSA and emSecure-ECDSA. While emSecure-ECDSA requires slightly less stack RAM compared to emSecure-RSA for verification at the same encryption strength, it significantly performs worse and requires more code space. Therefore, we recommend using emSecure-RSA, but it may make sense to use emSecure-ECDSA, if, for example, you are already using it for other purposes.

2.2 Third party solutions

Besides the emSecure digital signature solution, the Flasher Secure supports proprietary chip vendor specific solutions. These solutions provide end-to-end encryption: The firmware is provided to the target in encrypted form and is decrypted on the target. If you use one of these solutions, there is no need for additional protection via digital signatures over the UID.

2.2.1 Secure firmware installation (ST)

The secure firmware installation process by ST uses a combination of symmetric and asymmetric cryptography. It uses a symmetrically encrypted firmware file, which is decrypted on chip. The chip contains a hidden asymmetric private key and a readable public key. The symmetric key for the decryption of the firmware is encrypted by the server with the chip's public key and sent to the chip together with the encrypted firmware data. The chip's public key additionally is signed by ST, so the server can determine if the public key originates from a real device.



Chapter 3

Flasher Secure Server

The server, as the key component, controls the manufacturing process. It authorizes each produced device, keeps track of the manufacturers and intervenes if they behave in an unusual manner. It also provides automatic firmware updates to the Flasher Secure clients.

The server provides an interface for the Flasher Secure clients and a web interface for administration. Any communication over the network is secured via transport layer security (TLS).

3.1 Installation

The Flasher Secure Server is available for the following operation systems:

- Windows
- Linux

3.1.1 Hardware Requirements

The following hardware is recommended for the Flasher Secure Server:

Hardware part	Required minimum
Cores	1
Core clock	1 GHz
RAM	8 GB
Harddrive	32GB (mainly depends on the number of projects and devices)

3.1.2 Connection Requirements

The Flasher Secure Server requires to be connectable at ports:

- Port 3085 is used for the built-in web server, providing the user interface.
- Port 3110 is used for the Flasher Secure unit to connect to the server.

3.1.3 Installation on a Linux system

The Linux installation requires the following steps:

- Create a user account for the FlasherSecureServer, e.g. FlashSecServ
- Create a directory for your instance of the Flasher Secure Server, e.g. `sudo mkdir /opt/SEGGER/FlasherSecureServer`
- Change the owner of the directory, e.g. `sudo chown FlashSecServUsr /opt/SEGGER/FlasherSecureServer`, the Flasher Secure Server will put all its data into subdirectories of this root directory.
- Copy the shipment `Flasher_Secure_Linux_x64.tar.gz` to the created folder
- Extract the shipment in the folder, e.g. `tar xvzf Flasher_Secure_Linux_x64.tar.gz`
- Create a startup script for your system, e.g. use the sample systemd config file to start the server binary with the correct user account.
- If you have a firewall, ensure that the FlasherSecureServer can be reached at ports 3085 and 3110 via the TCP protocol.
- Reboot your system or start the FlasherSecureServer application. (note: start the Flasher Secure Server from the installation directory and not with a fully qualified path from elsewhere.)

Now everything is ready for the first use.

3.1.4 Installation on a Windows system

The Windows installation requires the following steps:

- Create a directory for your instance of the Flasher Secure Server, e.g. `mkdir C:\SEGGER\FlasherSecureServer`
- Copy the shipment `Flasher_Secure_Windows.zip` to the created folder
- Extract the shipment in the folder
- Create a service for your system that starts the Flasher Secure Server at start-up, e.g. `sc create "FlasherSecureServer" -binpath= "C:\FlasherSecureServer\FlasherSecureServer.exe --service -WorkingDir "C:\FlasherSecureServer"`
Please refer to your Windows documentation for further details.
- If you have a firewall, ensure that the FlasherSecureServer can be reached at the port 3085 and 3110 via the TCP protocol.

- Reboot your system or start the FlasherSecureServer application. (note: start the Flasher Secure Server from the installation directory and not with a fully qualified path from elsewhere, or use the command line argument to set the working directory properly.)

Now everything is ready for the first use.

Note

If you want to run the Flasher Secure Server as a Windows service, you need to pass the command line argument `WorkingDir` to the Flasher Secure Server at start-up. See *FlasherSecureServerCommandLineArguments* on page

3.1.5 Command line arguments

The Flasher Secure Server can be tuned by using some command line arguments.

Argument	Description	Windows	Linux
<code>-WorkingDir</code>	set the working directory	supported	supported
<code>--service</code>	starts the Flasher Secure Server as Windows service executable	supported	not supported
<code>-LogLevel</code>	sets the logging level	supported	supported

Example:

```
FlasherSecureServer -WorkingDir "C:\FlasherSecureServerData\" -LogLevel 7
```

Log Levels

Level	Description
0	no log messages
3	only error messages
5	error and warning messages
7	error, warning and info messages
9	verbose logging

3.1.6 Server Certificates

You may generate the required certificates with OpenSSL. A script file for this task is provided in the CERT sub-directory. The server can be used with signed certificates as well as with self-signed certificates.

The RSA server certificate can be installed on the Flasher Secure Units (SERVER.CRT). If a certificate file is present on a Flasher Secure unit, the unit will only connect to a server using exactly this certificate, effectively preventing man-in-the-middle attacks.

Note

The Flasher Secure units only support RSA. Therefore, the server must be provided at least an RSA certificate. The ECDSA certificate is optional, but can be used for accessing the web interface.

3.2 Projects in the Flasher Secure Server

The projects in the Flasher Secure Server contain the firmware data, programmed devices data and manufacturers related to the project. Programming of devices with the Flasher Secure unit is always related to an active project on the Flasher Secure Server.

The project can have several states during life time. The table shows the possible transitions between the states.

State	Description
Prepare	In this state, the project can be edited, e.g. the firmware can be uploaded or updated or the general setting like total number of devices can be changed.
Active	In this state, Flasher Secure units can program devices.
Finished	In this state, no more devices can be programmed. This is either because the total number of device is reached, or the project was set to the 'Finished' state manually for other reasons.
Deleted	This state is used only internally during the deletion of a project.

3.3 Web interface

3.3.1 Login

The Flasher Secure Server is reachable at `https://[server-ip]:3085`. The startpage is the login page:

The default logins are:

Company	User name	Password
Company	Admin1	Admin1
Company	User1	User1

The Login levels are:

Login Level	Privileges
Administrator	Can read, write and change project and configuration settings.
User	Can only read or view project and configuration settings.

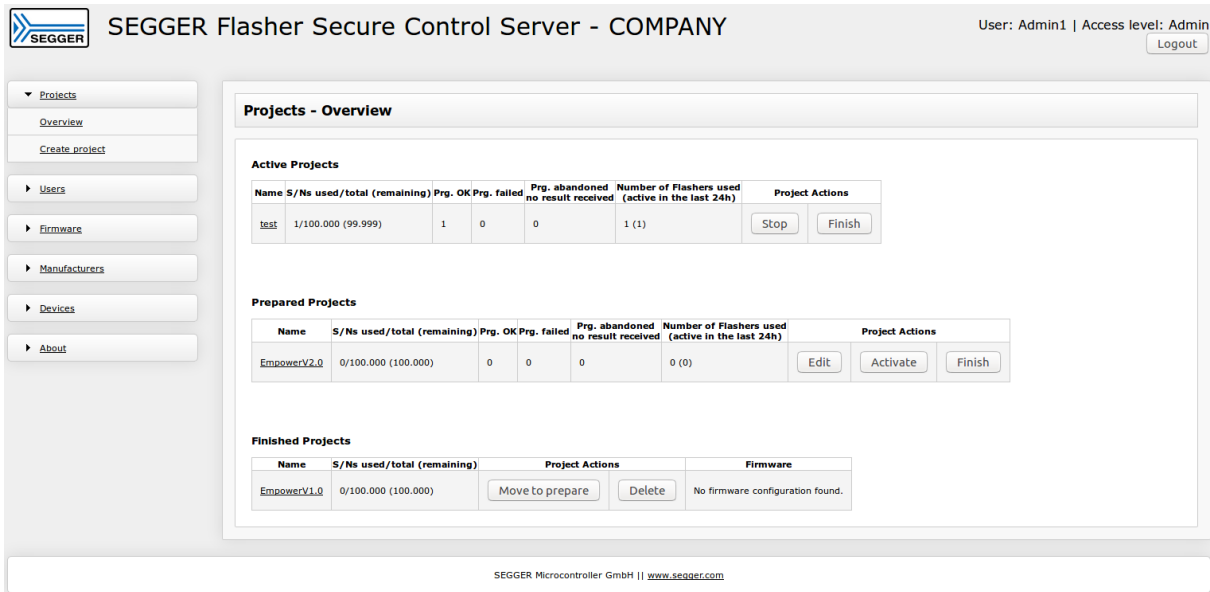
The user administration is described in section *User configuration* on page .

Note

The Flasher Secure Server is delivered with the default company name set to 'Company'. To customize the company name, please rename the `[installation-directory]/DATA/COMPANY` directory to a name of your choice, e.g. `[installation-directory]/DATA/MyCompany`.

3.3.2 Main page

The main page is the project overview, which shows information on the progress of each project.



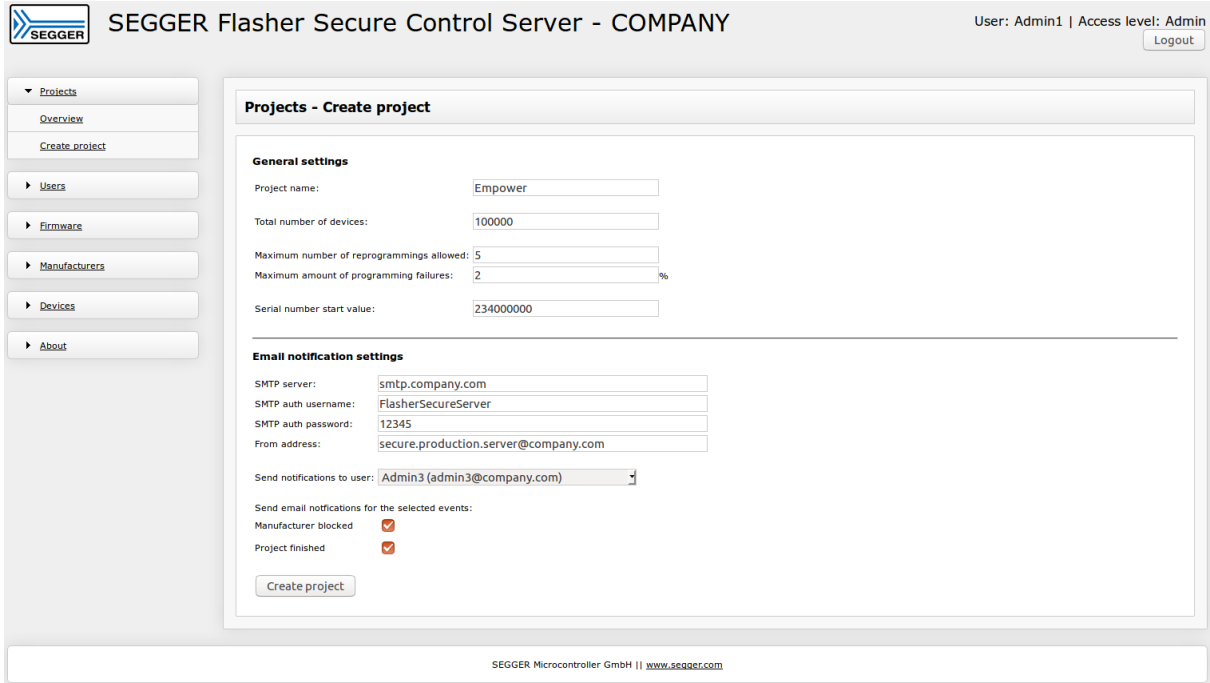
The Project Overview shows the projects in their respective states. The projects can be set to a new state using the buttons in the project action columns.

Button	Action
Active Projects	
Stop	Disables device programming and sets the project to 'Prepare' state.
Finish	Disables device programming and sets the project to 'Finished' state.
Prepared Projects	
Edit	Opens the project settings dialog for changing the project setup.
Activate	Sets the project state to 'Active'. Devices can now be programmed.
Finish	Disables device programming and sets the project to 'Finished' state.
Finished Projects	
Move to prepare	Sets the state to 'Prepare'. The project can now be edited or reactivated for further devices.
Delete	Deletes the project after further confirmation.

3.3.3 Projects

3.3.3.1 Create a new project

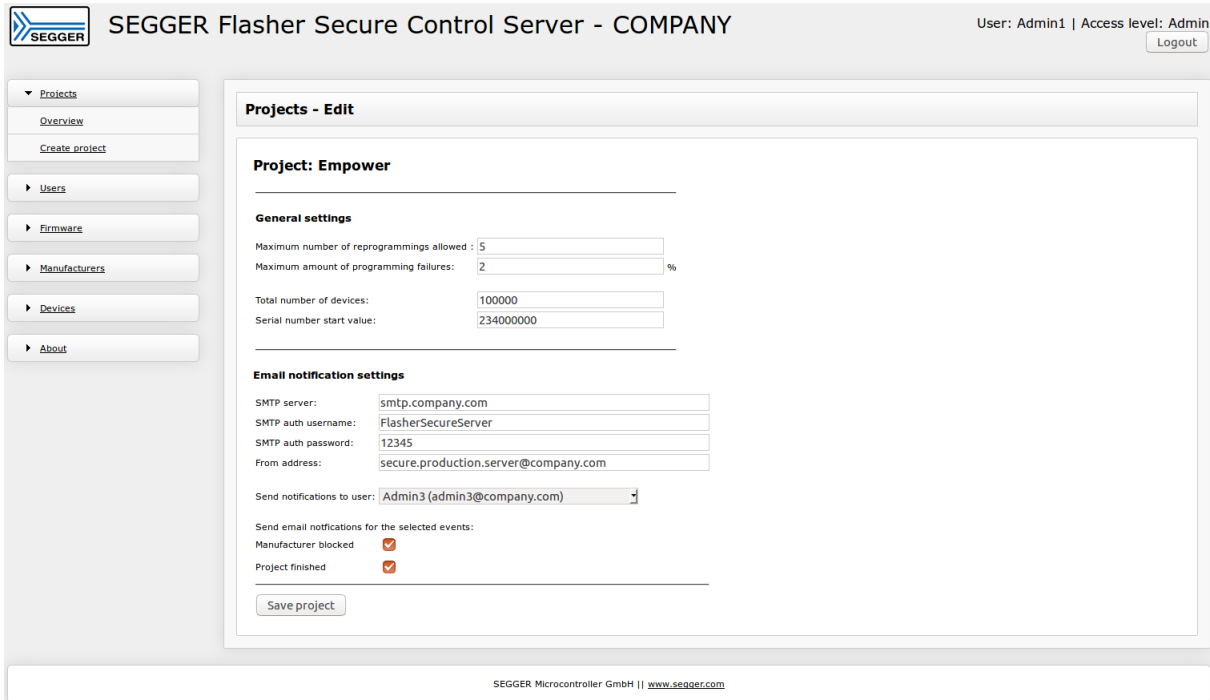
The 'Create project' page creates a new project.



Property	Meaning
Project name	A name for the project. This name is used to identify it in the other menus of the server and on the Flasher Secure unit.
Maximum number of programmings allowed	Defines how often a single device may be programmed.
Maximum amount of programming failures	Maximum rate of failed device programmings, counted over all programming attempts.
Total number of devices	Maximum number of devices which can be programmed.
Serial number start value	First serial number used for programming.
SMTP server	The email server to which Flasher Secure will connect for sending emails.
SMTP auth username	The user name for the email account.
SMTP auth password	The password for the email account.
Send notifications to user	The email will be sent to the chosen user.
Send email notifications for the selected events	
Manufacturer blocked	Email is sent if a 'manufacturer blocked' event was raised.
Project finished	Email is sent if the 'project finished' event was raised.

3.3.3.2 Project Edit

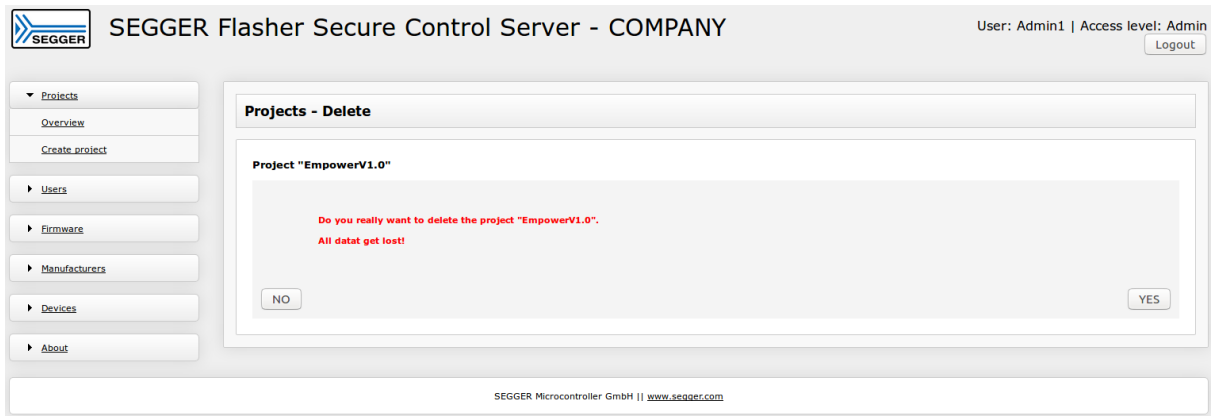
The project settings can be edited on the 'Edit' page. All parameters except the project name can be changed.



Property	Meaning
Maximum number of programmings allowed	Defines how often a single device may be programmed.
Maximum amount of programming failures	Maximum rate of failed device programmings, counted over all programming attempts.
Total number of devices	Maximum number of devices which can be programmed.
Serial number start value	First serial number used for programming.
SMTP server	The email server to which Flasher Secure will connect for sending emails.
SMTP auth username	The user name for the email account.
SMTP auth password	The password for the email account.
Send notifications to user	The email will be sent to the chosen user.
Send email notifications for the selected events	
Manufacturer blocked	Email is sent if a 'manufacturer blocked' event was raised.
Project finished	Email is sent if the 'project finished' event was raised.

3.3.3.3 Delete Project

A project can be deleted if it is in 'Finished' state. The 'Delete' button will start the deletion process. You need to confirm the choice on the following page.



Note

The project deletion will remove all data related to the project from the server databases and the disk. So make sure you have exported a device list if you need the list, e.g. for customer questions, before you delete the project. This process cannot be reverted!

3.3.4 Users

The Flasher Secure Server includes user management and user access management. Users have one of the following two access levels:

- Administrator
- User

While administrators have full access, users only have limited access. The users are not allowed to alter any settings. They have read privileges only.

The Flasher Secure Server knows two types of user accounts:

- Fixed user account
- Configured user account

The fixed user accounts cannot be changed during runtime of the Flasher Secure Server. They are set up using the `CFG_USER.TXT` configuration file, located in the `[installation-directory]/DATA/COMPANY`. These users should only be used for the initial setup process or evaluation.

The 2nd user type can be configured during runtime. These user accounts should be used for the regular operation of the Flasher Secure Server. They can have two states:

- Active
- Inactive

An active user account can be used to login to the Flasher Secure Server and execute the available actions. An inactive account cannot login and therefore cannot do anything with the server. But its data is still available.

Note

The Flasher Secure Server is delivered with two default accounts. We recommend removing them after creating your own accounts.

3.3.4.1 User Overview

The figure below shows the user overview page with a sample user configuration. The buttons in each row will execute the related action.

The screenshot shows the 'Users - Overview' page in the SEGGER Flasher Secure Control Server web interface. The page is titled 'SEGGER Flasher Secure Control Server - COMPANY' and shows the user 'Admin1' with 'Admin' access level. A sidebar on the left contains navigation links for Projects, Users (Overview, Add account), Firmware, Manufacturers, Devices, and About. The main content area is divided into three sections:

- Fixed user accounts:** A table with columns 'Account', 'Access Rights', and 'Email Address'. It lists 'Admin1' (Admin, User, admin1xt@company.com) and 'User1' (User, user1xt@company.com).
- Active user accounts:** A table with columns 'Account', 'Access Rights', 'Email Address', and 'Action'. It lists 'JohannaBest' (Admin, User, johanna.best@company.com) with 'Deactivate' and 'Edit' buttons.
- Inactive user accounts:** A table with columns 'Account', 'Access Rights', 'Email Address', and 'Action'. It lists 'JonDoe' (User, jon.doe@company.com) with 'Activate', 'Edit', and 'Delete' buttons.

The footer of the page reads 'SEGGER Microcontroller GmbH | www.segger.com'.

3.3.4.2 Adding a user account

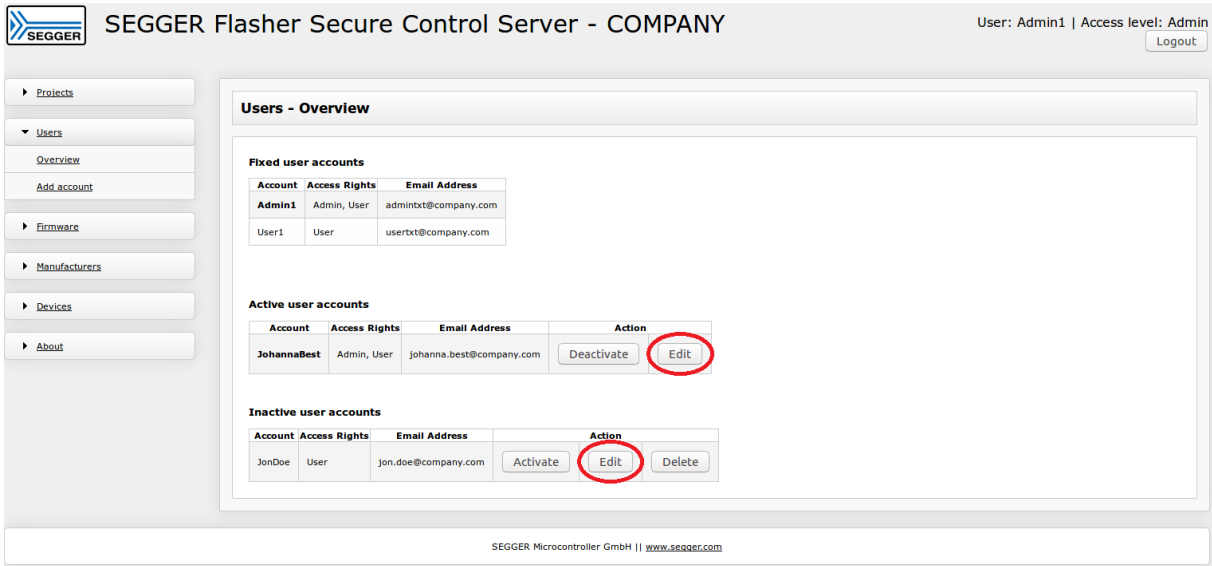
The screenshot shows the web interface for the SEGGER Flasher Secure Control Server. The page title is "SEGGER Flasher Secure Control Server - COMPANY". The user is logged in as "Admin" with an "Access level: Admin". A "Logout" button is visible in the top right corner. On the left side, there is a navigation menu with options: Projects, Users (selected), Overview, Add account, Firmware, Manufacturers, Devices, and About. The main content area is titled "Users - Add account" and contains a form for adding a new user account. The form fields are: Account (MyUser), Password (1234aBcD), and Email address (MyUser@company.com). There are two checkboxes: "User account enabled" (checked) and "User level admin" (unchecked). A "Submit" button is located at the bottom of the form. The footer of the page reads "SEGGER Microcontroller GmbH | www.segger.com".

The following properties can be configured:

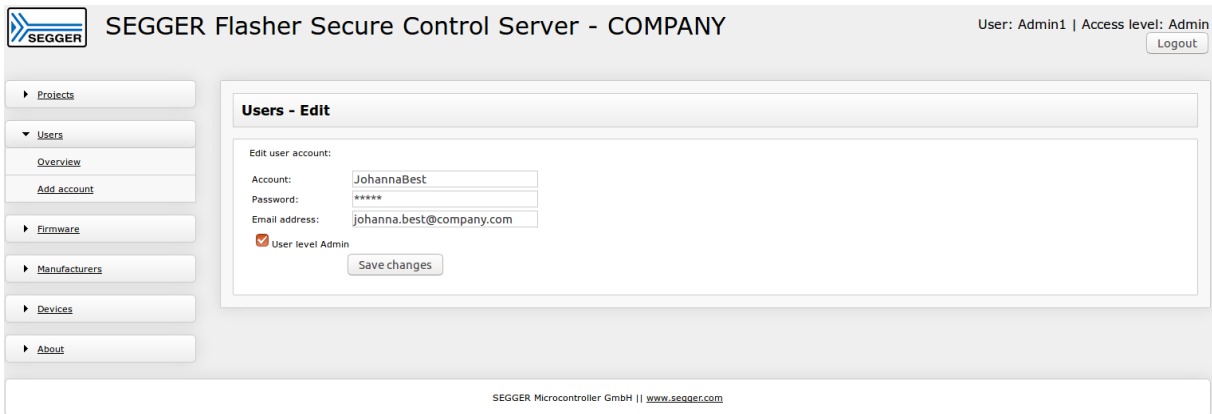
Property	Meaning
Account	User name used for the account.
Password	Password used for login.
Email address	Email address used for notifications.
Login enabled	Login is only enabled if checked.
User level Admin	User has administration access level if checked.

3.3.4.3 Editing a user account

The user account can be changed anytime. Select the user account by pressing the edit button, see figure below.

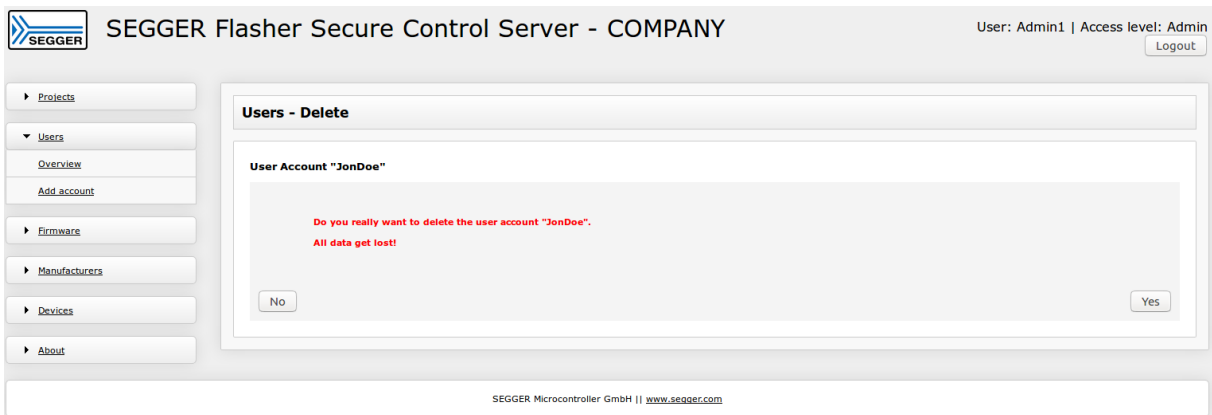
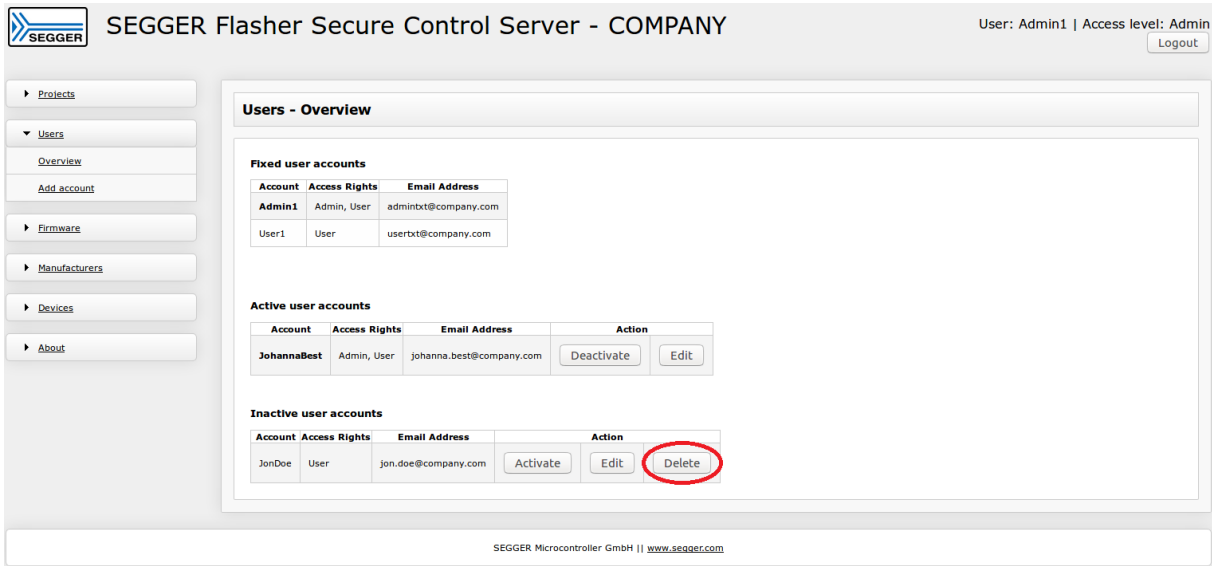


On the user setting page you can change all user settings.



3.3.4.4 Deleting a user account

User accounts can be deleted only if they are inactive. Delete the user account by pressing the delete button and confirm the action with **Yes**.

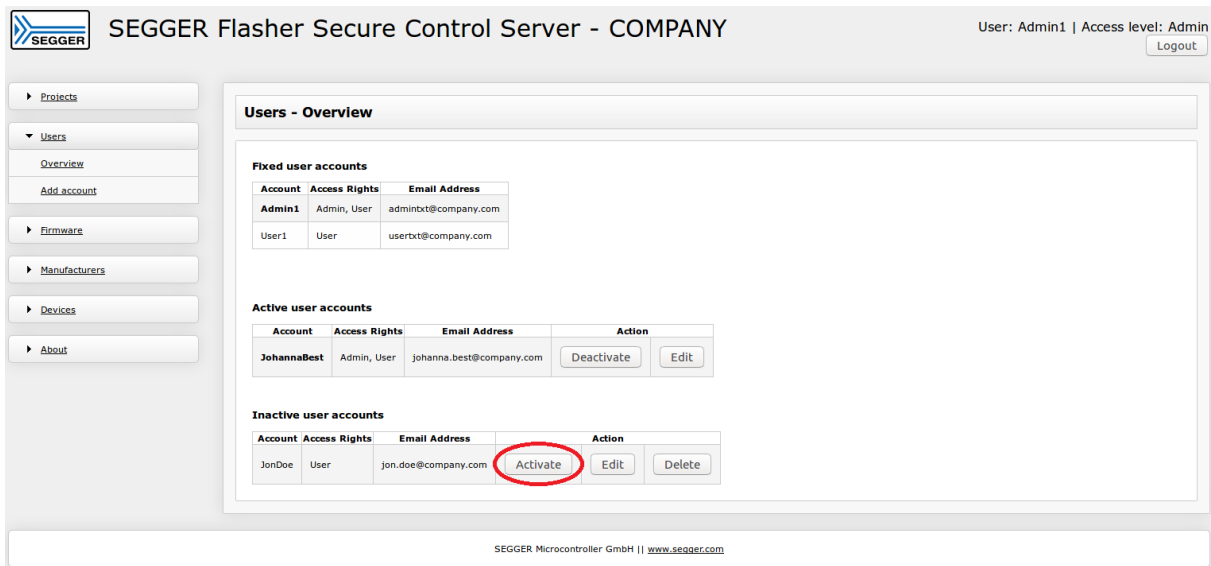


Note

If you only want to deactivate a user account temporarily, you can set the user account to inactive state using the button **DEACTIVATE**.

3.3.4.5 Activating a user account

User accounts are created with the status inactive. This enables you to prepare accounts and activate them when you need them. The activation is done by pressing the button **ACTIVATE**.



SEGGER Flasher Secure Control Server - COMPANY User: Admin1 | Access level: Admin [Logout](#)

Navigation: [Projects](#), [Users](#) (Overview, Add account), [Firmware](#), [Manufacturers](#), [Devices](#), [About](#)

Users - Overview

Fixed user accounts

Account	Access Rights	Email Address
Admin1	Admin, User	admin1xt@company.com
User1	User	usertxt@company.com

Active user accounts

Account	Access Rights	Email Address	Action
JohannaBest	Admin, User	johanna.best@company.com	Deactivate Edit

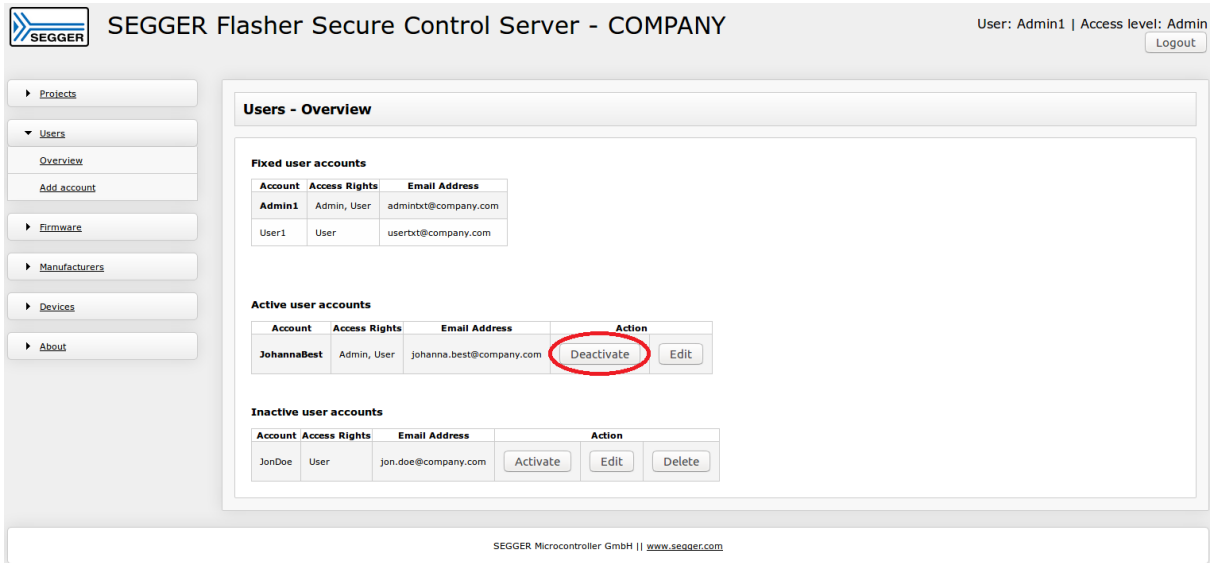
Inactive user accounts

Account	Access Rights	Email Address	Action
JonDoe	User	jon.doe@company.com	Activate Edit Delete

SEGGER Microcontroller GmbH | www.segger.com

3.3.4.6 Deactivating a user account

If you want to block a user account or disable it, you can set it to the inactive state. Therefore you need to press the button `DEACTIVATE`.



SEGGER Flasher Secure Control Server - COMPANY User: Admin1 | Access level: Admin
Logout

Navigation: Projects, Users (Overview, Add account), Firmware, Manufacturers, Devices, About

Users - Overview

Fixed user accounts

Account	Access Rights	Email Address
Admin1	Admin, User	admin1xt@company.com
User1	User	usertxt@company.com

Active user accounts

Account	Access Rights	Email Address	Action
JohannaBest	Admin, User	johanna.best@company.com	Deactivate Edit

Inactive user accounts

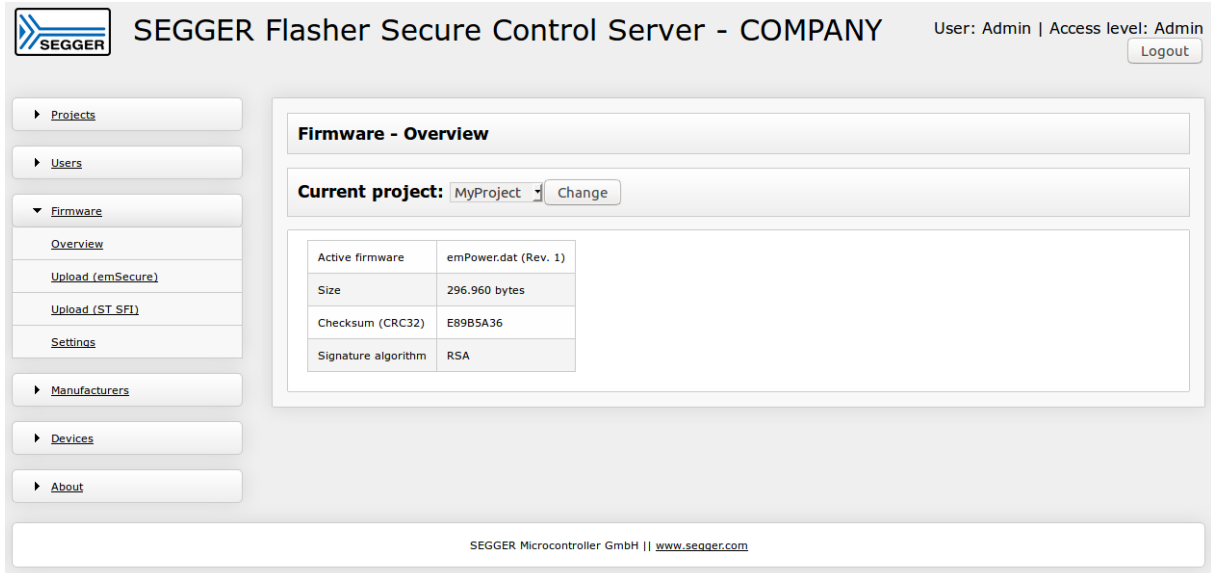
Account	Access Rights	Email Address	Action
JonDoe	User	jon.doe@company.com	Activate Edit Delete

SEGGER Microcontroller GmbH | www.segger.com

3.3.5 Firmware

3.3.5.1 Firmware Overview

The overview page shows the currently used firmware image properties for the selected project.



The screenshot displays the web interface for the SEGGER Flasher Secure Control Server. The header includes the SEGGER logo, the title "SEGGER Flasher Secure Control Server - COMPANY", and user information: "User: Admin | Access level: Admin" with a "Logout" button. A left sidebar contains navigation buttons for "Projects", "Users", "Firmware" (expanded), "Manufacturers", "Devices", and "About". The "Firmware" section is active, showing a sub-menu with "Overview", "Upload (emSecure)", "Upload (ST_SFI)", and "Settings". The main content area is titled "Firmware - Overview" and features a "Current project:" dropdown menu set to "MyProject" with a "Change" button. Below this is a table displaying firmware properties:

Active firmware	emPower.dat (Rev. 1)
Size	296.960 bytes
Checksum (CRC32)	E89B5A36
Signature algorithm	RSA

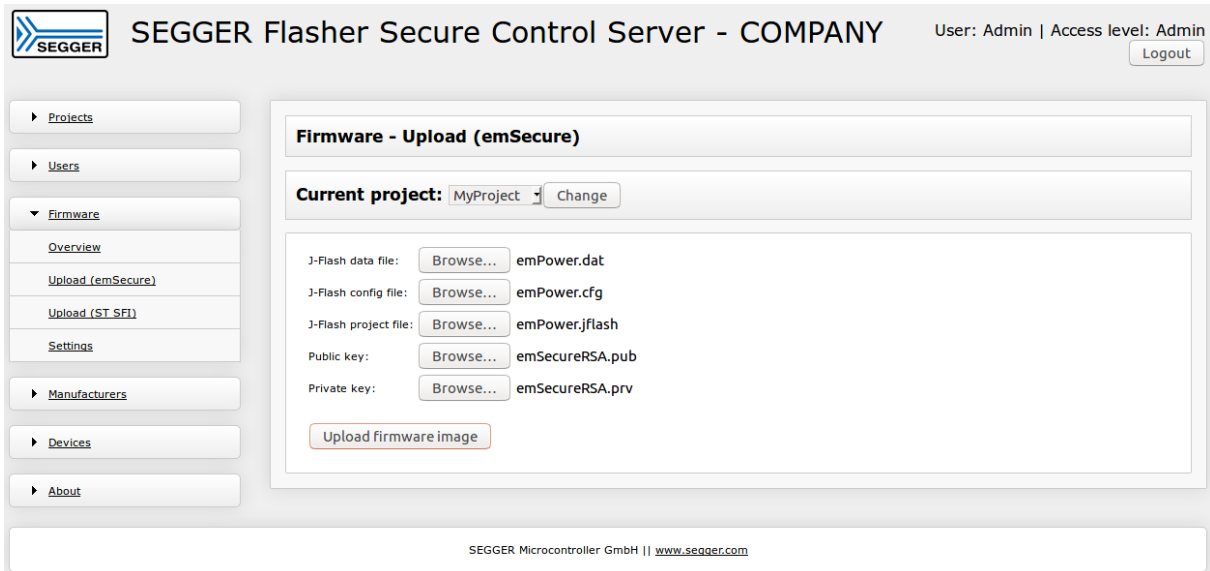
The footer of the interface contains the text: "SEGGER Microcontroller GmbH | www.segger.com".

You can switch between the projects using the drop down list and the change button.

3.3.5.2 Firmware upload (emSecure)

This page provides the upload interface for the firmware image files when using the emSecure programming method. The server uses both the private and the public key file to generate and verify a signature. The key files are created by a key generator included in the RSA or ECDSA package.

The server automatically detects the encryption standard based on the key files.

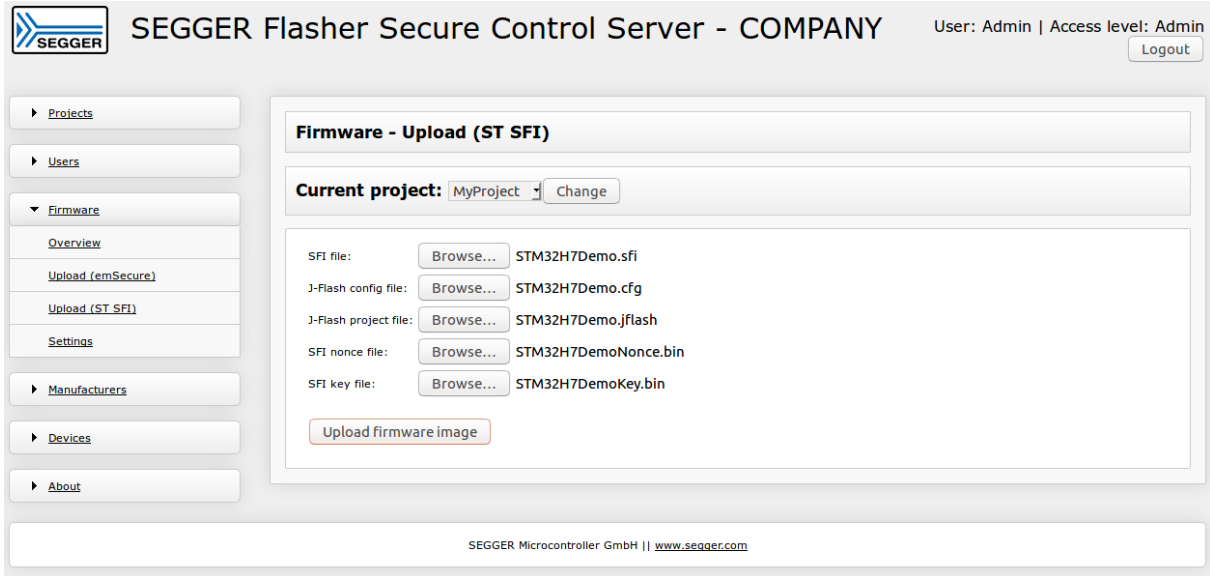


Note

You need to prepare the firmware image with the J-Flash tool for the Flasher Secure outside the Flasher Secure Server. The latest version of the tool is available <https://www.segger.com/downloads/jlink>. For the usage of the J-Flash tool please refer to manual UM08003 (available from the same page) or this link <https://www.segger.com/downloads/jlink/UM08003>.

3.3.5.3 Firmware upload (ST SFI)

This page provides the upload interface for the firmware image files when using ST’s SFI programming method. Instead of a regular data file, the firmware is encapsulated in an encrypted container file (SFI file). The server additionally needs the “nonce file” and the “key file” in order to create the unique license file required to program the target device.



Note

You need to prepare the firmware image with a tool available from ST. Please contact ST for more details and the toolkit for SFI flash programming.

3.3.5.4 Firmware settings

The Flasher Secure needs to know at which address in the target memory the serial number and the signature should be programmed. These addresses need to be specified on this page. If the address `0xFFFFFFFF` is specified, the corresponding data is not programmed into the device.

The screenshot shows the 'SEGGER Flasher Secure Control Server - COMPANY' web interface. The user is logged in as 'Admin' with 'Access level: Admin'. The 'Logout' button is visible in the top right corner. The left sidebar contains navigation options: Projects, Users, Firmware (expanded), Overview, Upload (emSecure), Upload (ST_SFI), Settings, Manufacturers, Devices, and About. The main content area is titled 'Firmware - Settings'. It features a 'Current project:' dropdown menu set to 'MyProject' with a 'Change' button. Below this, there are two input fields: 'Signature address:' with the value '0x0801FF00' and 'Serial number address:' with the value '0x0801FEF8'. A note states: 'Note: A value of 0xFFFFFFFF disables sending a serial number or signature.' A 'Submit' button is located below the input fields. The footer of the page reads 'SEGGER Microcontroller GmbH | www.segger.com'.

Note

The firmware image file needs to contain some dummy data at the target memory range. Otherwise the data cannot be programmed correctly.

3.3.6 Manufacturers

This topic describes how you manage your manufacturer accounts. The overview shows all manufacturers for the selected project. In addition, it also provides an overview about the programming status at each manufacturer.

The screenshot shows the SEGGER Flasher Secure Control Server web interface. The page title is "SEGGER Flasher Secure Control Server - COMPANY". The user is identified as "Admin1" with an "Access level: Admin". A "Logout" button is visible in the top right corner. The left sidebar contains navigation links for "Projects", "Users", "Firmware", "Manufacturers", "Overview", "Devices", and "About". The main content area is titled "Manufacturers - Overview" and displays data for the selected project "testProject".

Project "testProject"

Inactive Manufacturer accounts

Account	Quota	Used	Failed	OK	Success rate	Actions
TestAbC	50.000	1	0	1	100.00% (1/1)	<input type="button" value="Activate"/> <input type="button" value="Edit"/> <input type="button" value="Delete"/> <input type="button" value="Reset rate"/>

Known flashers:

Serial number	Last used	Manufacturer account
844200106	2018-07-03 08:03:04	TestAbC

Project "EmpowerV1.0"

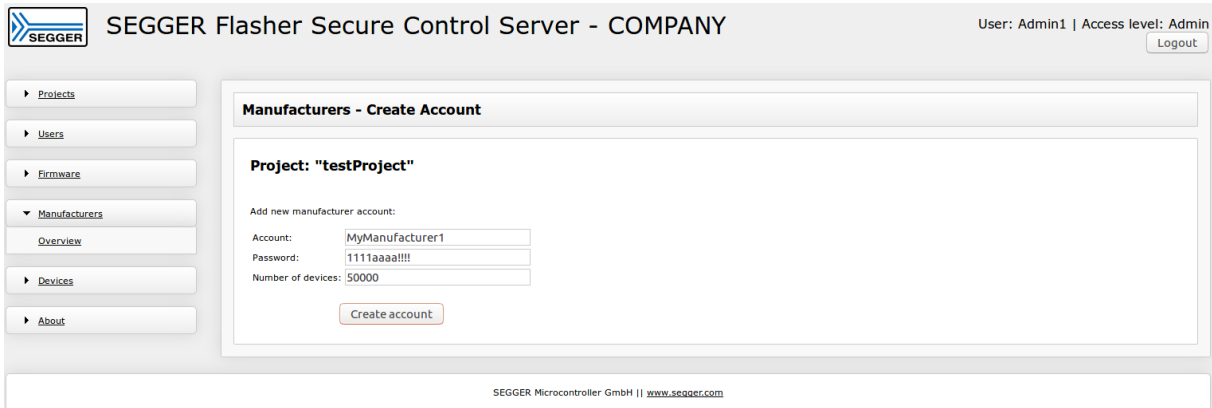
SEGGGER Microcontroller GmbH | www.segger.com

Note

A manufacturer account is always associated with one project. If a manufacturing company shall produce several products, you need to create a separate account for each project.

3.3.6.1 Adding a manufacturer account

This page provides the possibility to add a manufacturer for a project. Choose the project by using the drop down list and change button.



The following properties can be configured:

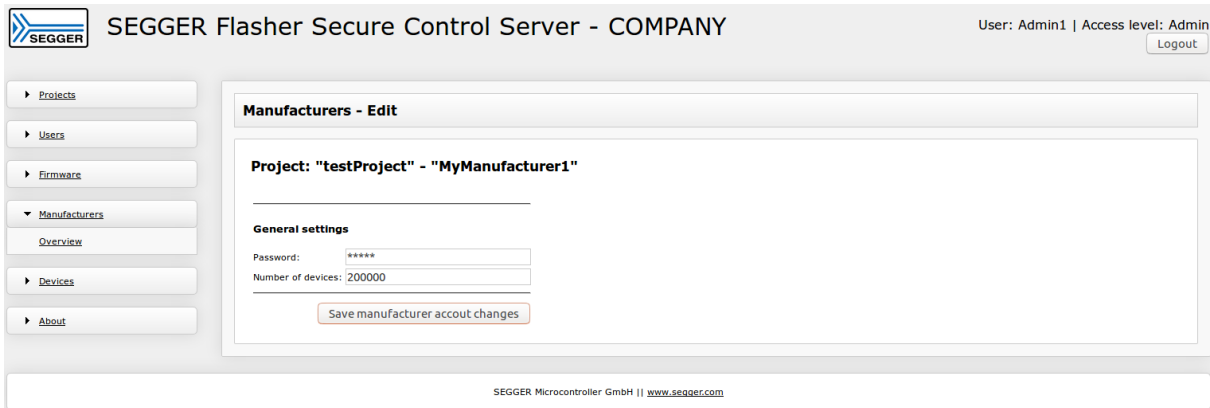
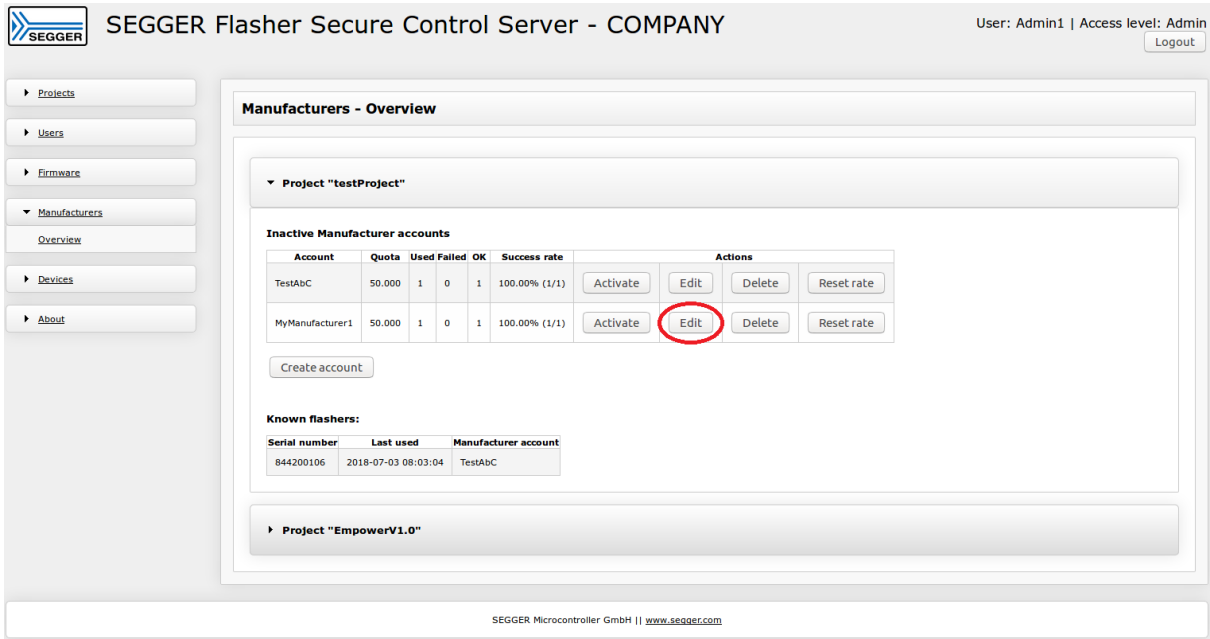
Property	Meaning
Account	User name used for the account or Flasher Secure authentication.
Password	Password used for login or Flasher Secure authentication.
Number of devices	Manufacturers quota on devices.

Note

A manufacturer account is always associated with one project. If a manufacturing company shall produce several products, you need to create a separate account for each project.

3.3.6.2 Editing a manufacturer account

This page allows to edit a manufacturer login. Choose the edit button of the manufacturer you want to edit.



The following properties can be changed:

Property	Meaning
Password	Password used for login or Flasher Secure authentication.
Number of devices	Manufacturers quota on devices.

3.3.6.3 Deleting a manufacturer account

A manufacturer account can be deleted if it is in inactive state. The button `DELETE` will remove the manufacturer account. You need to confirm the action with `Yes` on the opening new page. All data will get lost and cannot be recovered!

SEGGER Flasher Secure Control Server - COMPANY User: Admin1 | Access level: Admin [Logout](#)

Navigation: [Projects](#), [Users](#), [Firmware](#), [Manufacturers](#) (Overview), [Devices](#), [About](#)

Manufacturers - Overview

Project "testProject"

Inactive Manufacturer accounts

Account	Quota	Used	Failed	OK	Success rate	Actions			
TestABC	50.000	1	0	1	100.00% (1/1)	Activate	Edit	Delete	Reset rate
MyManufacturer1	50.000	1	0	1	100.00% (1/1)	Activate	Edit	Delete	Reset rate

[Create account](#)

Known flashers:

Serial number	Last used	Manufacturer account
844200106	2018-07-03 08:03:04	TestABC

Project "EmpowerV1.0"

SEGGER Microcontroller GmbH | www.segger.com

SEGGER Flasher Secure Control Server - COMPANY User: Admin1 | Access level: Admin [Logout](#)

Navigation: [Projects](#), [Users](#), [Firmware](#), [Manufacturers](#) (Overview), [Devices](#), [About](#)

Manufacturers - Delete

Manufacturer account "MyManufacturer1"

Do you really want to delete the manufacturer "MyManufacturer1" for project "testProject".
All data get lost!

[No](#) [Yes](#)

SEGGER Microcontroller GmbH | www.segger.com

Note

If you only want to disable the account temporarily, you can simply deactivate the account.

3.3.6.4 Activating a manufacturer account

A manufacturer account is created with the state set to inactive. This is done with respect to the fact that you may want to set up the project configuration correctly and completely before allowing any manufacturer to start programming devices. Therefore you need to activate a manufacturer account to allow production for that account. The button **ACTIVATE** will set the state to active for the chosen manufacturer account.

SEGGER Flasher Secure Control Server - COMPANY User: Admin1 | Access level: Admin
[Logout](#)

Manufacturers - Overview

Project "testProject"

Inactive Manufacturer accounts

Account	Quota	Used	Failed	OK	Success rate	Actions			
TestABC	50.000	1	0	1	100.00% (1/1)	Activate	Edit	Delete	Reset rate
MyManufacturer1	50.000	1	0	1	100.00% (1/1)	Activate	Edit	Delete	Reset rate

[Create account](#)

Known flashers:

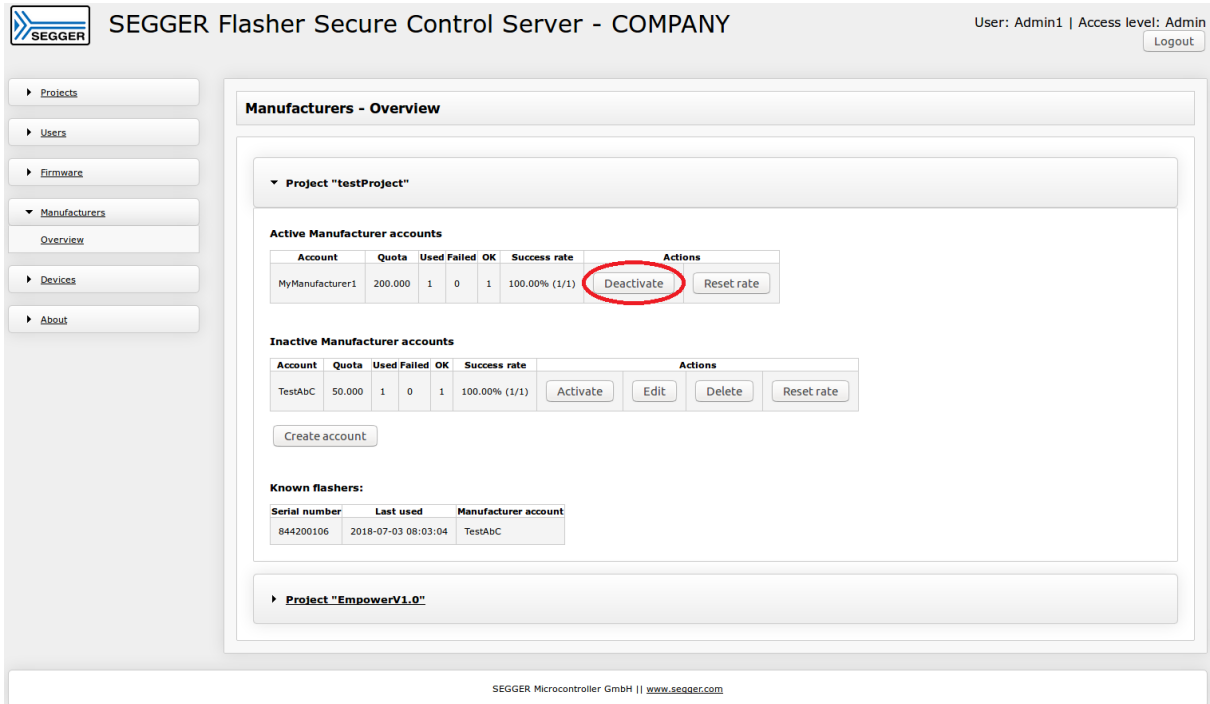
Serial number	Last used	Manufacturer account
844200106	2018-07-03 08:03:04	TestABC

Project "EmpowerV1.0"

SEGGER Microcontroller GmbH | www.segger.com

3.3.6.5 Deactivating a manufacturer account

A manufacturer account can be deactivated e.g. if the current quota has been produced. The button `DEACTIVATE` will set the state to inactive for the chosen manufacturer account and any further login will be blocked.



The screenshot displays the 'SEGGER Flasher Secure Control Server - COMPANY' web interface. The user is logged in as 'Admin1' with 'Admin' access level. The main content area is titled 'Manufacturers - Overview' and is divided into sections for 'Active Manufacturer accounts' and 'Inactive Manufacturer accounts'. The 'Active Manufacturer accounts' table lists 'MyManufacturer1' with a quota of 200,000, 1 used, 0 failed, and 1 OK, resulting in a 100.00% success rate. A red circle highlights the 'Deactivate' button in the 'Actions' column for this account. The 'Inactive Manufacturer accounts' table lists 'TestAbc' with a quota of 50,000, 1 used, 0 failed, and 1 OK, also with a 100.00% success rate. Below this table is a 'Create account' button. The 'Known flashers' section shows a table with columns for 'Serial number', 'Last used', and 'Manufacturer account', containing one entry with serial number 844200106 and last used time 2018-07-03 08:03:04. The footer of the interface reads 'SEGGER Microcontroller GmbH | www.segger.com'.

Account	Quota	Used	Failed	OK	Success rate	Actions
MyManufacturer1	200.000	1	0	1	100.00% (1/1)	Deactivate Reset rate

Account	Quota	Used	Failed	OK	Success rate	Actions
TestAbc	50.000	1	0	1	100.00% (1/1)	Activate Edit Delete Reset rate

Serial number	Last used	Manufacturer account
844200106	2018-07-03 08:03:04	TestAbc

3.3.6.6 Resetting a manufacturer's success rate counter

The Flasher Secure Server monitors the manufacturer accounts and calculates a success rate for the programmed target devices. If a manufacturer success rate falls below the configured rate, the manufacturer account will be blocked. This means that the manufacturer cannot produce any further devices. This shall block a manufacturer who has gone rogue or does anything suspicious.

If a manufacturer is blocked, the button `Reset rate` resets the internal counter for measuring the success rate of the manufacturer and unblocks the manufacturer account.

SEGGGER Flasher Secure Control Server - COMPANY User: Admin1 | Access level: Admin
Logout

Navigation: Projects, Users, Firmware, Manufacturers (Overview), Devices, About

Manufacturers - Overview

Project "testProject"

Inactive Manufacturer accounts

Account	Quota	Used	Failed	OK	Success rate	Actions			
TestABC	50.000	1	0	1	100.00% (1/1)	Activate	Edit	Delete	Reset rate
MyManufacturer1	50.000	1	0	1	100.00% (1/1)	Activate	Edit	Delete	Reset rate

Create account

Known flashers:

Serial number	Last used	Manufacturer account
844200106	2018-07-03 08:03:04	TestABC

Project "EmpowerV1.0"

SEGGGER Microcontroller GmbH | www.segger.com

3.3.7 Devices

3.3.7.1 Devices overview

The device overview summarizes the device lists for a project. See the screenshot for the listed parameters.

SEGGER Flasher Secure Control Server - COMPANY User: Admin1 | Access level: Admin [Logout](#)

Navigation: [Projects](#), [Users](#), [Firmware](#), [Manufacturers](#), **Devices**, [Overview](#), [About](#)

Devices - Overview

▼ Project "testProject"

Devices programmed / total (remaining)	Prg. OK	Prg. failed	Prg. abandoned no result received	Number of Flashers used (active in the last 24h)
2/1,000 (998)	2	0	0	1 (0)

[Show device list](#) [Export device list as CSV-file](#) [Export device list as TXT-file](#)

► Project "EmpowerV1.0"

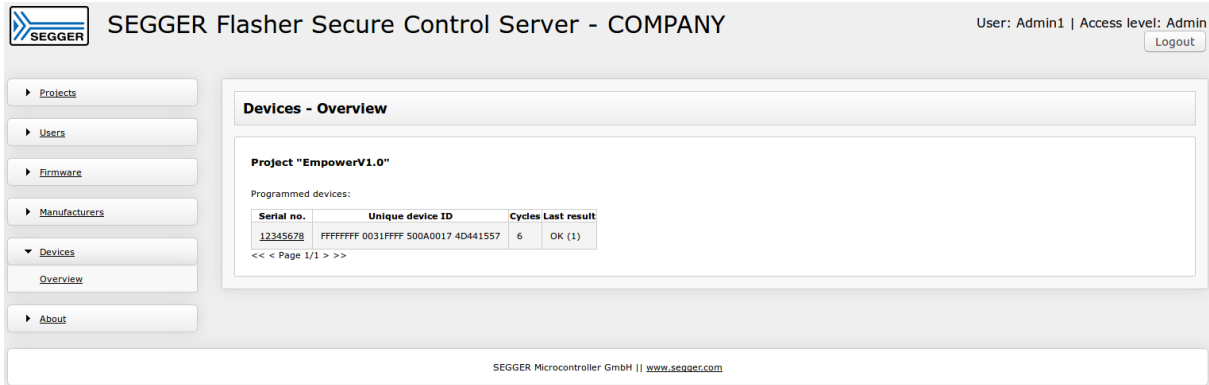
SEGGER Microcontroller GmbH | www.segger.com

3.3.7.2 Device list

You can switch to the detailed view, which will list all devices, using the button `Show device list`.

The list includes the following parameters for each device:

- Running production number
- Serial number
- Unique device ID
- Programming cycles
- Most recent programming result



The screenshot shows the web interface of the SEGGER Flasher Secure Control Server. The header includes the SEGGER logo, the title 'SEGGER Flasher Secure Control Server - COMPANY', and the user information 'User: Admin1 | Access level: Admin' with a 'Logout' button. A left sidebar contains navigation buttons for 'Projects', 'Users', 'Firmware', 'Manufacturers', 'Devices' (expanded to show 'Overview'), and 'About'. The main content area is titled 'Devices - Overview' and displays 'Project "EmpowerV1.0"'. Below this, it shows 'Programmed devices:' followed by a table with the following data:

Serial no.	Unique device ID	Cycles	Last result
12345678	FFFFFFFF 0031FFFF 500A0017 4D441557	6	OK (1)

Below the table is a pagination control: '<< < Page 1/1 > >>'. The footer of the interface reads 'SEGGER Microcontroller GmbH | www.segger.com'.

You can browse through the list using the <<, <, > or >> below the list.

3.3.7.3 Exporting the device list

The device list can be exported. You can choose between an export as:

- Plain text
- CSV (comma separated values) file

The corresponding button will start the export in your browser.

SEGGER Flasher Secure Control Server - COMPANY

User: Admin1 | Access level: Admin

Logout

Devices - Overview

Project "testProject"

Devices programmed / total (remaining)	Prg. OK	Prg. failed	Prg. abandoned no result received	Number of Flashers used (active in the last 24h)
2/1.000 (998)	2	0	0	1 (0)

Show device list Export device list as CSV-file Export device list as TXT-file

Project "EmpowerV1.0"

SEGGER Microcontroller GmbH | www.segger.com

Plain text format example:

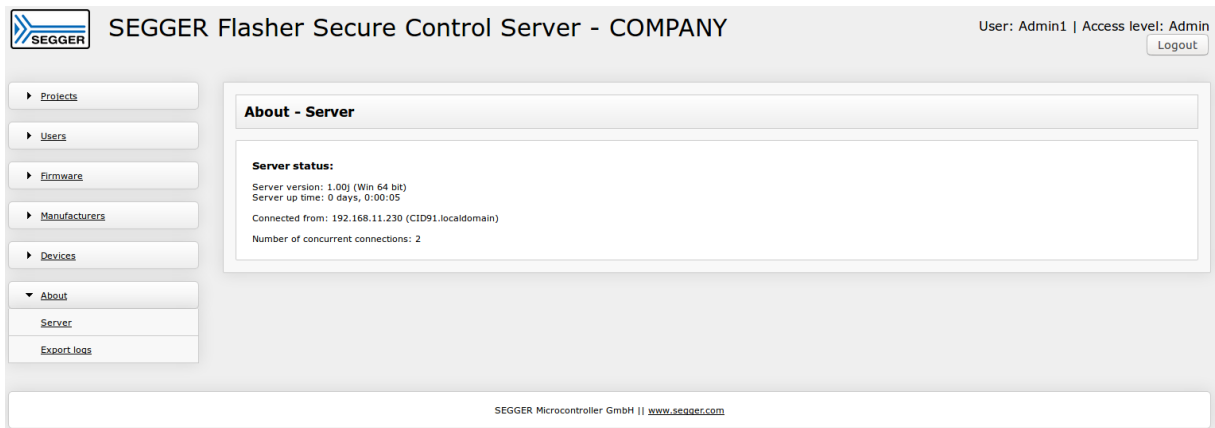
```
2017-03-20 10:23:26 - Programming device 22330000 with unique ID 001B0047
34345109 35353835 00000000 using Flasher 5: OK (1)
2017-03-20 11:13:30 - Programming device 22330001 with unique ID 0039004E
34345109 35353835 00000000 using Flasher 11: OK (1)
```

CSV format example:

```
"Date","Device S/N","Device UID","Flasher S/N","Result"
"2017-03-20 10:23:26","22330000","47001B00095134343538355000000000","5","1"
"2017-03-20 11:13:30","22330001","4E003900095134343538355000000000","11","1"
```

3.3.8 Server status

The server status page provides an overview about the current server status.



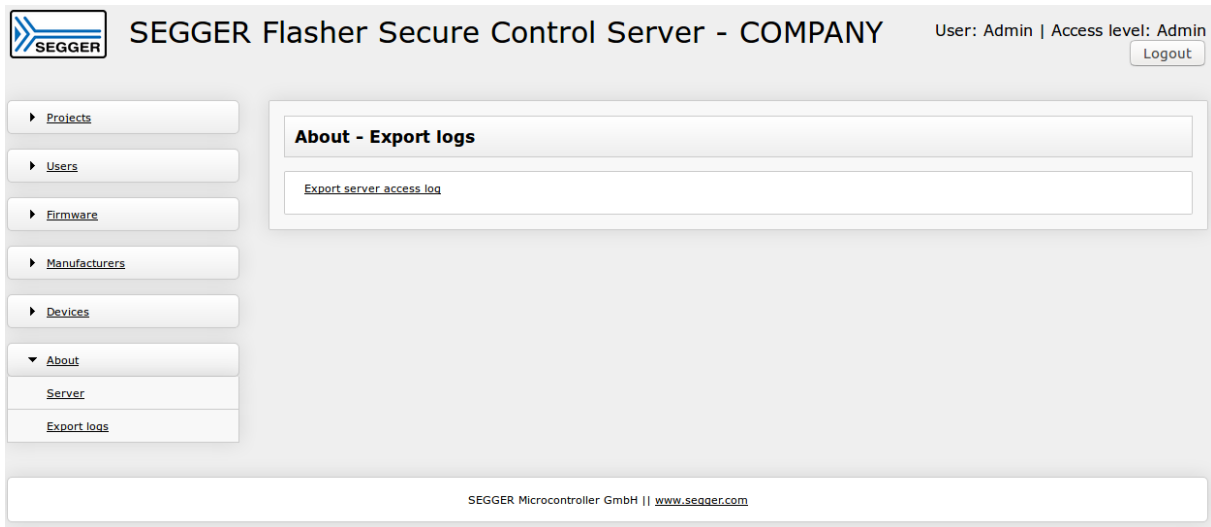
The screenshot displays the web interface for the SEGGER Flasher Secure Control Server. The page title is "SEGGER Flasher Secure Control Server - COMPANY". The user is identified as "Admin1" with an "Access level: Admin". A "Logout" button is visible in the top right corner. The left sidebar contains a navigation menu with the following items: "Projects", "Users", "Firmware", "Manufacturers", "Devices", "About", "Server", and "Export logs". The "About - Server" section is active, showing the following server status information:

```
Server status:
Server version: 1.00j (Win 64 bit)
Server up time: 0 days, 0:00:05
Connected from: 192.168.11.230 (CID91.localdomain)
Number of concurrent connections: 2
```

At the bottom of the page, the footer text reads: "SEGGER Microcontroller GmbH | www.segger.com".

3.3.8.1 Server Access Log

The server's access log, which is also stored on the server as a text file, can be downloaded from the "Export logs" page.



The log file is in the NCSA Common log format. The description of the format can be found here: http://publib.boulder.ibm.com/tividd/td/ITWSA/ITWSA_info45/en_US/HTML/guide/c-logs.html#common

Access log sample:

```
123.45.67.89 - - [2017-01-01 00:00:00] "GET /"  
123.45.67.89 - - [2017-01-01 00:00:00] "GET /style.css"  
123.45.67.89 - - [2017-01-01 00:00:00] "GET /img/SeggerLogo_200x.png"  
123.45.67.89 - - [2017-01-01 00:00:00] "GET /favicon.ico"  
123.45.67.89 - - [2017-01-01 00:00:05] "POST /index.htm"
```


3.4 REST interface

3.4.1 General

The Flasher Secure Server can be remote-controlled using the provided REST API. Most functions are accessible, but not all.

All REST API calls require parameters. The parameters are transferred in plain text format. The parameters are always name/value pairs. Each pair needs to be terminated by a carriage return and line feed, for **example**:

```
parameter=value\r\n
```

The REST API calls require authentication. Therefore, you have to add the parameters COMPANY, USERNAME and PASSWORD for the calls:

```
COMPANY=COMPANY\r\n
USERNAME=Admin1\r\n
PASSWORD=Admin1\r\n
```

Note

The user account names and passwords are case sensitive!

3.4.2 REST API

The table shows the available features.

Project features	
Feature	Description
<i>Create a project</i>	Creates new project
<i>Delete a project</i>	Deletes existing project
<i>Activate a project</i>	Sets the project state to 'active'
<i>Stop a project</i>	Sets the project state to 'prepare'
<i>Finish a project</i>	Sets the project state to 'finished'
<i>Move to prepare</i>	Sets the project state to 'prepare'
Firmware features	
Feature	Description
<i>Firmware import</i>	Imports firmware to a project
User features	
Feature	Description
<i>Add user account</i>	Creates user account
<i>Activate user account</i>	Activates user account
<i>Deactivate user account</i>	Deactivates user account
<i>Delete user account</i>	Deletes user account
<i>Edit user account</i>	Edits user account
Device features	
Feature	Description

<i>Export device list as CSV</i>	Exports the device list as CSV file
<i>Export device list as TXT</i>	Exports the device list as TXT file
Manufacturer features	
Feature	Description
<i>Add manufacturer account</i>	Creates manufacturer account
<i>Activate manufacturer account</i>	Activates manufacturer account
<i>Deactivate manufacturer account</i>	Deactivates manufacturer account
<i>Delete manufacturer account</i>	Deletes manufacturer account
<i>Edit manufacturer account</i>	Edits manufacturer account
<i>Reset manufacturer success rate</i>	Resets manufacturer's success rate

The REST API call will result with a response telling if the requested action was successfully executed or not. The responses are in the following format:

```
Result=xxxx\r\n
Resulttext=OK. Or result text telling something went wrong.\r\n
```

All results with 0 or a positive number mean that the operation was successful. Negative results indicate an error. The result text contains further information on the error cause.

3.4.2.1 Creating a project

This REST API call creates a new project.

URI:

```
[Server IP]:3085/REST/ProjectCreate
```

Parameters:

Parameter		Description
COMPANY	M	The company name
USERNAME	M	The user account name; the user needs to have administration rights
PASSWORD	M	The user password
PROJECT	M	The name of the project to be created
TOTAL_NUMBER_DEVICES	M	The maximum number of devices that can be programmed
REPROGRAMMING	M	The attempts of programming tries for one device
FAILURE_RATE	M	The maximum failure rate in percent for a manufacturer
START_SERIALNUMBER	O	The serial number for the first device
SMTP_SERVER	O	The SMTP server used for email notifications
SMTP_AUTH_USERNAME	O	The SMTP authentication user name
SMTP_AUTH_PASSWORD	O	The SMTP authentication password
SMTP_FROM_EMAIL_ADDRESS	O	The email address from which the notifications are sent
NOTIFICATION_TO_USERNAME	O	The user account which shall receive the email notifications
NOTIFY_MANUFACTURER_BLOCKED	O	Send a notifications if a manufacturer is blocked; set = 1 if yes, set = 0 if no
NOTIFY_PROJECT_FINISHED	O	Send a notifications if a project is finished; set = 1 if yes, set = 0 if no

M: mandatory parameter, O: optional parameter

Example:

```
COMPANY=COMPANY\r\n
USERNAME=Admin1\r\n
PASSWORD=Admin1\r\n
PROJECT=emPowerTest\r\n
TOTAL_NUMBER_DEVICES=100\r\n
REPROGRAMMING=5\r\n
FAILURE_RATE=2\r\n
START_SERIALNUMBER=10010000\r\n
SMTP_SERVER=smtp.company.com\r\n
SMTP_AUTH_USERNAME=FlasherSecureServer\r\n
SMTP_AUTH_PASSWORD=87654321\r\n
SMTP_FROM_EMAIL_ADDRESS=FlasherSecureProduction@company.com\r\n
NOTIFICATION_TO_USERNAME=Admin\r\n
NOTIFY_MANUFACTURER_BLOCKED=1\r\n
NOTIFY_PROJECT_FINISHED=0\r\n
```

Example result:

```
Result=2000\r\n
Resulttext=OK.\r\n
```

3.4.2.2 Deleting a project

This REST API call deletes an existing project.

URI:

```
[Server IP]:3085/REST/ProjectDelete
```

Parameters:

Parameter	Description
COMPANY	The company name
USERNAME	The user account name; the user needs to have administration rights
PASSWORD	The user password
PROJECT	The name of the project to be deleted

Example:

```
COMPANY=COMPANY\r\n
USERNAME=Admin1\r\n
PASSWORD=Admin1\r\n
PROJECT=emPowerTest\r\n
```

Example result:

```
Result=2005\r\n
Resulttext=OK.\r\n
```

3.4.2.3 Activating a project

This REST API call sets the project state to active. In this state, devices can be programmed by the Flasher Secure units.

URI:

```
[Server IP]:3085/REST/ProjectActivate
```

Parameters:

Parameter	Description
COMPANY	The company name
USERNAME	The user account name; the user needs to have administration rights
PASSWORD	The user password
PROJECT	The name of the project to be activated

Example:

```
COMPANY=COMPANY\r\n  
USERNAME=Admin1\r\n  
PASSWORD=Admin1\r\n  
PROJECT=emPowerTest\r\n
```

Example result:

```
Result=2003\r\n  
Resulttext=OK.\r\n
```

3.4.2.4 Stopping a project

This REST API call sets the project state to prepare again if it is active. As a consequence, the Flasher Secure units can no longer program devices.

URI:

```
[Server IP]:3085/REST/ProjectStop
```

Parameters:

Parameter	Description
COMPANY	The company name
USERNAME	The user account name; the user needs to have administration rights
PASSWORD	The user password
PROJECT	The name of the project to be stopped

Example:

```
COMPANY=COMPANY\r\n  
USERNAME=Admin1\r\n  
PASSWORD=Admin1\r\n  
PROJECT=emPowerTest\r\n
```

Example result:

```
Result=2004\r\n  
Resulttext=OK.\r\n
```

3.4.2.5 Finishing a project

This REST API call sets the project state to finish. As a consequence, the Flasher Secure units can no longer program devices.

URI:

```
[Server IP]:3085/REST/ProjectFinish
```

Parameters:

Parameter	Description
COMPANY	The company name
USERNAME	The user account name; the user needs to have administration rights
PASSWORD	The user password
PROJECT	The name of the project to be set to finished state

Example:

```
COMPANY=COMPANY\r\n  
USERNAME=Admin1\r\n  
PASSWORD=Admin1\r\n  
PROJECT=emPowerTest\r\n
```

Example result:

```
Result=2001\r\n  
Resulttext=OK.\r\n
```


3.4.2.6 Moving a project to prepare state

This REST API call sets the project state from finish back to prepare. This way, a project can be reactivated for further programming if a production lot was completed previously.

URI:

```
[Server IP]:3085/REST/ProjectMovePrepare
```

Parameters:

Parameter	Description
COMPANY	The company name
USERNAME	The user account name; the user needs to have administration rights
PASSWORD	The user password
PROJECT	The name of the project to be set to prepare state

Example:

```
COMPANY=COMPANY\r\n
USERNAME=Admin1\r\n
PASSWORD=Admin1\r\n
PROJECT=emPowerTest\r\n
```

Example result:

```
Result=2002\r\n
Resulttext=OK.\r\n
```

3.4.2.7 Importing firmware

This REST API call imports firmware to an existing project.

URI:

```
[Server IP]:3085/REST/FirmwareImportFiles
```

Parameters:

Parameter		Description
COMPANY	M	The company name
USERNAME	M	The user account name, the user needs to have administration rights
PASSWORD	M	The user password
PROJECT	M	The project name
IMPORT_FIRMWARE_DAT_FILE	M	Activates the import of the firmware data file. Set = 1 for yes, set = 0 for no
IMPORT_FIRMWARE_CFG_FILE	M	Activates the import of the firmware configuration file. Set = 1 for yes, set = 0 for no
IMPORT_PROJECT_JFLASH_FILE	M	Activates the import of the J-Flash project file. Set = 1 for yes, set = 0 for no
IMPORT_PUBLIC_KEY_FILE	M	Activates the import of the public key file. Set = 1 for yes, set = 0 for no
IMPORT_PRIVATE_KEY_FILE	M	Activates the import of the public data file. Set = 1 for yes, set = 0 for no
FIRMWARE_DAT_FILE	O	The data file name with extension.
FIRMWARE_CFG_FILE	O	The configuration file name with extension.
PROJECT_JFLASH_FILE	O	The J-Flash project file name with extension.
PUBLIC_KEY_FILE	O	The public key file name with extension.
PRIVATE_KEY_FILE	O	The private key file name with extension.

M: mandatory parameter, O: optional parameter

Example:

```
COMPANY=COMPANY\r\n
USERNAME=Admin1\r\n
PASSWORD=Admin1\r\n
PROJECT=emPowerTest\r\n
TOTAL_NUMBER_DEVICES=100\r\n
REPROGRAMMING=5\r\n
FAILURE_RATE=2\r\n
START_SERIALNUMBER=10010000\r\n
SMTP_SERVER=smtp.company.com\r\n
SMTP_AUTH_USERNAME=FlasherSecureServer\r\n
SMTP_AUTH_PASSWORD=87654321\r\n
SMTP_FROM_EMAIL_ADDRESS=FlasherSecureProduction@company.com\r\n
NOTIFICATION_TO_USERNAME=Admin\r\n
NOTIFY_MANUFACTURER_BLOCKED=1\r\n
NOTIFY_PROJECT_FINISHED=0\r\n
```

Example result:

```
Result=4000\r\n
Resulttext=OK.\r\n
```

3.4.2.8 Adding a user account

This REST API call creates a new user account.

URI:

```
[Server IP]:3085/REST/UserCreate
```

Parameters:

Parameter		Description
COMPANY	M	The company name
USERNAME	M	The user account name; the user needs to have administration rights
PASSWORD	M	The user password
USER_ACCOUNT	M	The new user account name
USER_PASSWORD	M	The password for the new user account
USER_EMAIL_ADDRESS	M	The email address of the new user
USER_ACCESS_LEVEL	M	The access level. Set = 1 for administrator, set = 0 for user.

M: mandatory parameter, O: optional parameter

Example:

```
COMPANY=COMPANY\r\n
USERNAME=Admin\r\n
PASSWORD=Admin\r\n
USER_ACCOUNT=JohannaBest\r\n
USER_PASSWORD=aBcD1234!\r\n
USER_EMAIL_ADDRESS=johanna.best@company.com\r\n
USER_ACCESS_LEVEL=1\r\n
```

Example result:

```
Result=3001\r\n
Resulttext=OK.\r\n
```

3.4.2.9 Activating a user account

This REST API call activates a user account.

URI:

```
[Server IP]:3085/REST/UserActivate
```

Parameters:

Parameter		Description
COMPANY	M	The company name
USERNAME	M	The user account name; the user needs to have administration rights
PASSWORD	M	The user password
USER_ACCOUNT	M	The user account name to be activated

M: mandatory parameter, O: optional parameter

Example:

```
COMPANY=COMPANY\r\n
USERNAME=Admin\r\n
PASSWORD=Admin\r\n
USER_ACCOUNT=JohannaBest\r\n
```

Example result:

```
Result=3002\r\n
Resulttext=OK.\r\n
```

3.4.2.10 Deactivating a user account

This REST API call deactivates a user account.

URI:

```
[Server IP]:3085/REST/UserDeactivate
```

Parameters:

Parameter		Description
COMPANY	M	The company name
USERNAME	M	The user account name; the user needs to have administration rights
PASSWORD	M	The user password
USER_ACCOUNT	M	The user account name to be deactivated

M: mandatory parameter, O: optional parameter

Example:

```
COMPANY=COMPANY\r\n
USERNAME=Admin\r\n
PASSWORD=Admin\r\n
USER_ACCOUNT=JohannaBest\r\n
```

Example result:

```
Result=3003\r\n
Resulttext=OK.\r\n
```

3.4.2.11 Deleting a user account

This REST API call deletes an user account.

URI:

```
[Server IP]:3085/REST/UserDelete
```

Parameters:

Parameter		Description
COMPANY	M	The company name
USERNAME	M	The user account name; the user needs to have administration rights
PASSWORD	M	The user password
USER_ACCOUNT	M	The user account name to be deleted

M: mandatory parameter, O: optional parameter

Example:

```
COMPANY=COMPANY\r\n
USERNAME=Admin\r\n
PASSWORD=Admin\r\n
USER_ACCOUNT=JohannaBest\r\n
```

Example result:

```
Result=3004\r\n
Resulttext=OK.\r\n
```

3.4.2.12 Editing a user account

This REST API call edits an user account.

URI:

```
[Server IP]:3085/REST/UserEdit
```

Parameters:

Parameter		Description
COMPANY	M	The company name
USERNAME	M	The user account name; the user needs to have administration rights
PASSWORD	M	The user password
USER_ACCOUNT	M	The user account name to be edited
USER_PASSWORD	M	The password for the changed user account
USER_EMAIL_ADDRESS	M	The email address of the changed user
USER_ACCESS_LEVEL	M	The access level. Set = 1 for administrator, set = 0 for user.

M: mandatory parameter, O: optional parameter

Example:

```
COMPANY=COMPANY\r\n
USERNAME=Admin\r\n
PASSWORD=Admin\r\n
USER_ACCOUNT=JohannaBest\r\n
USER_PASSWORD=aBcD1234!\r\n
USER_EMAIL_ADDRESS=johanna.best@company.com\r\n
USER_ACCESS_LEVEL=1\r\n
```

Example result:

```
Result=3005\r\n
Resulttext=OK.\r\n
```


3.4.2.13 Adding a manufacturer account

This REST API call adds a manufacturer account.

URI:

```
[Server IP]:3085/REST/ManufacturerCreate
```

Parameters:

Parameter		Description
COMPANY	M	The company name
USERNAME	M	The user account name; the user needs to have administration rights
PASSWORD	M	The user password
PROJECT	M	The project for which the manufacturer account shall be created
MANUFACTURER	M	The manufacturer account name to be added
MANUFACTURER_PASSWORD	M	The password for the user manufacturer
DEVICES	M	The number of devices allowed to be programmed

M: mandatory parameter, O: optional parameter

Example:

```
COMPANY=COMPANY\r\n
USERNAME=Admin\r\n
PASSWORD=Admin\r\n
PROJECT=testProject\r\n
MANUFACTURER=MySuperManufacturer\r\n
MANUFACTURER_PASSWORD=aBcD1234!\r\n
DEVICES=10000\r\n
```

Example result:

```
Result=5000\r\n
Resulttext=OK.\r\n
```

3.4.2.14 Activating a manufacturer account

This REST API call activates a manufacturer account.

URI:

```
[Server IP]:3085/REST/ManufacturerActivate
```

Parameters:

Parameter		Description
COMPANY	M	The company name
USERNAME	M	The user account name; the user needs to have administration rights
PASSWORD	M	The user password
PROJECT	M	The project for which the manufacturer account shall be activated
MANUFACTURER	M	The manufacturer account name to be activated

M: mandatory parameter, O: optional parameter

Example:

```
COMPANY=COMPANY\r\n
USERNAME=Admin\r\n
PASSWORD=Admin\r\n
PROJECT=testProject\r\n
MANUFACTURER=MySuperManufacturer\r\n
```

Example result:

```
Result=5003\r\n
Resulttext=OK.\r\n
```

3.4.2.15 Deactivating a manufacturer account

This REST API call deactivates a manufacturer account.

URI:

```
[Server IP]:3085/REST/ManufacturerDeactivate
```

Parameters:

Parameter		Description
COMPANY	M	The company name
USERNAME	M	The user account name; the user needs to have administration rights
PASSWORD	M	The user password
PROJECT	M	The project for which the manufacturer account shall be deactivated
MANUFACTURER	M	The manufacturer account name to be deactivated

M: mandatory parameter, O: optional parameter

Example:

```
COMPANY=COMPANY\r\n
USERNAME=Admin\r\n
PASSWORD=Admin\r\n
PROJECT=testProject\r\n
MANUFACTURER=MySuperManufacturer\r\n
```

Example result:

```
Result=5004\r\n
Resulttext=OK.\r\n
```

3.4.2.16 Deleting a manufacturer account

This REST API call deletes a manufacturer account.

URI:

```
[Server IP]:3085/REST/ManufacturerDelete
```

Parameters:

Parameter		Description
COMPANY	M	The company name.
USERNAME	M	The user account name; the user needs to have administration rights
PASSWORD	M	The user password
PROJECT	M	The project for which the manufacturer account shall be deleted
MANUFACTURER	M	The manufacturer account name to be deleted

M: mandatory parameter, O: optional parameter

Example:

```
COMPANY=COMPANY\r\n
USERNAME=Admin\r\n
PASSWORD=Admin\r\n
PROJECT=testProject\r\n
MANUFACTURER=MySuperManufacturer\r\n
```

Example result:

```
Result=5005\r\n
Resulttext=OK.\r\n
```

3.4.2.17 Editing a manufacturer account

This REST API call edits a manufacturer account.

URI:

```
[Server IP]:3085/REST/ManufacturerEdit
```

Parameters:

Parameter		Description
COMPANY	M	The company name
USERNAME	M	The user account name; the user needs to have administration rights
PASSWORD	M	The user password
PROJECT	M	The project for which the manufacturer account shall be edited
MANUFACTURER	M	The manufacturer account name to be edited
MANUFACTURER_PASSWORD	M	The password for the manufacturer
DEVICES	M	The number of devices allowed to be programmed

M: mandatory parameter, O: optional parameter

Example:

```
COMPANY=COMPANY\r\n
USERNAME=Admin\r\n
PASSWORD=Admin\r\n
PROJECT=testProject\r\n
MANUFACTURER=MySuperManufacturer\r\n
MANUFACTURER_PASSWORD=aBcD1234!\r\n
DEVICES=10000\r\n
```

Example result:

```
Result=5001\r\n
Resulttext=OK.\r\n
```

3.4.2.18 Exporting a device list as CSV

This REST API call exports the device list as CSV file.

URI:

```
[Server IP]:3085/REST/DeviceExport
```

Parameters:

Parameter		Description
COMPANY	M	The company name
USERNAME	M	The user account name; the user needs to have administration rights
PASSWORD	M	The user password
PROJECT	M	The project for which the device list shall be exported

M: mandatory parameter, O: optional parameter

Example:

```
COMPANY=COMPANY\r\n
USERNAME=Admin\r\n
PASSWORD=Admin\r\n
PROJECT=testProject\r\n
```

Example result:

```
Result=6001\r\n
Resulttext=OK.\r\n
"Date","Device S/N","Device UID","Flasher S/N","Firmware Revision","Result"\r\n
"2017-03-20 10:23:26","22330000","47001B00095134343538353500000000","5","1"\r\n
"2017-03-20 11:13:30","22330001","4E003900095134343538353500000000","11","1"\r\n
```

3.4.2.19 Exporting a device list as TXT

This REST API call exports the device list as TXT file.

URI:

```
[Server IP]:3085/REST/DeviceExport
```

Parameters:

Parameter		Description
COMPANY	M	The company name
USERNAME	M	The user account name; the user needs to have administration rights
PASSWORD	M	The user password
PROJECT	M	The project for which the device list shall be exported

M: mandatory parameter, O: optional parameter

Example:

```
COMPANY=COMPANY\r\n
USERNAME=Admin\r\n
PASSWORD=Admin\r\n
PROJECT=testProject\r\n
```

Example result:

```
Result=6001\r\n
Resulttext=OK.\r\n
2017-03-20 10:23:26 - Programming device 22330000 with unique ID 001B0047
34345109 35353835 00000000 using Flasher 5: OK (1)
2017-03-20 11:13:30 - Programming device 22330001 with unique ID 0039004E
34345109 35353835 00000000 using Flasher 11: OK (1)
```

3.4.3 REST API Result Codes

The table below list all result codes for successful REST API calls.

Code	Meaning
Project actions	
2000	Project created successfully
2001	Project state set to finish successfully
2002	Project state set to prepare successfully
2003	Project state set to active successfully
2004	Project state set to prepare from active (stop) successfully
2005	Project deleted successfully
User account actions	
3001	User account created successfully
3002	User account activated successfully
3003	User account deactivated successfully
3004	User account deleted successfully
3005	User account edited successfully
Firmware actions	
4000	Firmware imported successfully
Manufacturer account actions	
5000	Manufacturer account created successfully
5001	Manufacturer account edited successfully
5002	Manufacturer account success rate reset successfully
5003	Manufacturer account activated successfully
5004	Manufacturer account deactivated successfully
5005	Manufacturer account deleted successfully
Device actions	
6002	Device list exported as CSV file successfully
6003	Device list exported as TXT file successfully

The table list all result codes for **unsuccessful** REST API calls.

Code	Meaning
Project general	
-2000	Unspecified error during project action
Project create	
-2001	Project already exists
-2002	Project name invalid
-2003	Project name is too long
-2004	The email address for notification is invalid
-2005	The email address is too long
-2006	The SMTP server name is invalid
-2007	The SMTP server name is too long
-2008	The SMTP user name is invalid
-2009	The SMTP user name is too long
-2010	The SMTP password is invalid
-2011	the SMTP password is too long
-2012	The user to be notified does not exist
-2013	Unspecified IO error
-2014	Unspecified data error
-2015	The number of total devices is invalid
-2016	The number of reprogramming attempts per device is invalid
-2017	The failure rate is invalid
Finish project	
-2030	Unspecified data error
-2031	Project was not found
-2032	Project not in correct state to be transferred to finish state
Move to prepare state	
-2040	Unspecified data error
-2041	Project was not found
-2042	Project not in correct state to be transferred to prepare state
Activate project	
-2050	Unspecified data error
-2051	Project was not found
-2052	Project not in correct state to be transferred to active state
-2053	Unspecific database error
Stop project	
-2060	Unspecified data error
-2061	Project was not found
-2062	Project not in correct state to be transferred to stop (prepare) state
Delete project	
-2070	Unspecified data error
-2071	Project was not found
-2072	Project not in correct state to be transferred to be deleted
-2073	Unspecified database error

Code	Meaning
-2074	Unspecified IO error

Code	Meaning
User general	
-3000	Unspecified error during user action
User add action	
-3001	User data invalid
-3002	Company name invalid
-3003	User account name invalid
-3004	User account name too long
-3005	User password invalid
-3006	User password too long
-3007	User account already exists
-3008	Email address invalid
-3009	Email address too long
-3010	Access level invalid
-3020	Database error
-3021	Database error
-3022	Database error
User activate action	
-3030	Data invalid
-3032	Company name invalid
-3033	User account name invalid
-3034	User account not found
-3035	User account is in wrong state
-3040	Database error
-3041	Database error
User deactivate action	
-3050	Data invalid
-3052	Company name invalid
-3053	User account name invalid
-3054	User account not found
-3055	User account is in wrong state
-3060	Database error
-3061	Database error
User delete action	
-3070	Data invalid
-3072	Company name invalid
-3073	User account name invalid
-3074	User account not found
-3075	User account is in wrong state
-3080	Database error
-3081	Database error
User edit action	
-3101	Data invalid
-3102	Company name invalid

Code	Meaning
-3103	User account name invalid
-3104	User account name too long
-3105	User password invalid
-3106	User password too long
-3107	User account not found
-3108	Email address invalid
-3109	Email address too long
-3110	Access level invalid
-3120	Database error
-3121	Database error
-3122	Database error

Code	Meaning
Manufacturer general	
-5000	Unspecified error during manufacturer action
Manufacturer add action	
-5001	Manufacturer account already exists
-5002	Manufacturer account name is invalid
-5003	Manufacturer account name is too long
-5004	Manufacturer password is invalid
-5005	Manufacturer password is too long
-5006	Number of devices is invalid
-5007	Number of devices is too long
-5008	Project invalid
-5010	Database error
-5011	Database error
-5012	Database error
-5013	Database error
-5014	Database error
-5015	Project not found
-5016	Data invalid
Manufacturer edit action	
-5020	Manufacturer account is invalid
-5021	Manufacturer account not found
-5022	Manufacturer account name is invalid
-5023	Manufacturer account name is too long
-5024	Manufacturer password is invalid
-5025	Manufacturer password is too long
-5026	Number of devices is invalid
-5027	Number of devices is too long
-5028	Project invalid
-5030	Database error
-5031	Database error
-5032	Database error
-5033	Database error
-5034	Database error
-5035	Data invalid
Manufacturer reset rate action	
-5040	Data invalid
-5041	Manufacturer account is invalid
-5042	Manufacturer account not found
-5043	Project invalid
-5044	Database error
-5045	Database error
-5046	Database error
-5047	Database error

Code	Meaning
Manufacturer deactivate action	
-5050	Data invalid
-5051	Manufacturer account is invalid
-5052	Manufacturer account not found
-5053	Project invalid
-5054	Manufacturer account is in wrong state
-5055	Database error
-5056	Database error
-5057	Database error
Manufacturer activate action	
-5060	Data invalid
-5061	Manufacturer account is invalid
-5062	Manufacturer account not found
-5063	Project invalid
-5064	Manufacturer account is in wrong state
-5065	Database error
-5066	Database error
-5067	Database error
-5068	Database error
Manufacturer delete action	
-5080	Data invalid
-5081	Manufacturer account is invalid
-5082	Manufacturer account not found
-5083	Project invalid
-5084	Manufacturer account account name too long
-5085	Database error
-5086	Project not found

Code	Meaning
Device general	
-6000	Unspecified error during device action
Device export as CSV file action	
-6020	Device data invalid
-6021	Project invalid
-6022	Company name invalid
-6023	Not enough memory
-6030	Database error
Device export as TXT file action	
-6040	Device data invalid
-6041	Project invalid
-6042	Company name invalid
-6043	Not enough memory
-6050	Database error

Chapter 4

Flasher Secure unit

The Flasher Secure unit is based on the SEGGER Flasher unit, so it generally supports the same devices as the Flashers do. However, devices that do not provide a UID or a vendor specific secure programming solution are not supported.

4.1 General

The Flasher Secure unit is the programming unit that is connected to the target device.

Note

The Flasher Secure needs an internet connection or network connection. It needs to be able to reach the Flasher Secure Server, which provides the signature generation service.

4.2 IP configuration

IP configuration is done via DHCP by default. There are different configuration options for setting up a manual configuration:

- J-Link Configurator (refer to *UM08001: J-Link / J-Trace User Guide*, chapter 4 Setup, section 4.5 J-Link Configurator), short summary see here *IP configuraton with J-Link Configurator* on page 82
- J-Link Commander (refer to *UM08001: J-Link / J-Trace User Guide*, chapter 3 J-Link software and documentation package, section 3.2 J-Link Commander)
- Flasher Secure web interface (*Web interface* on page 22).

Note

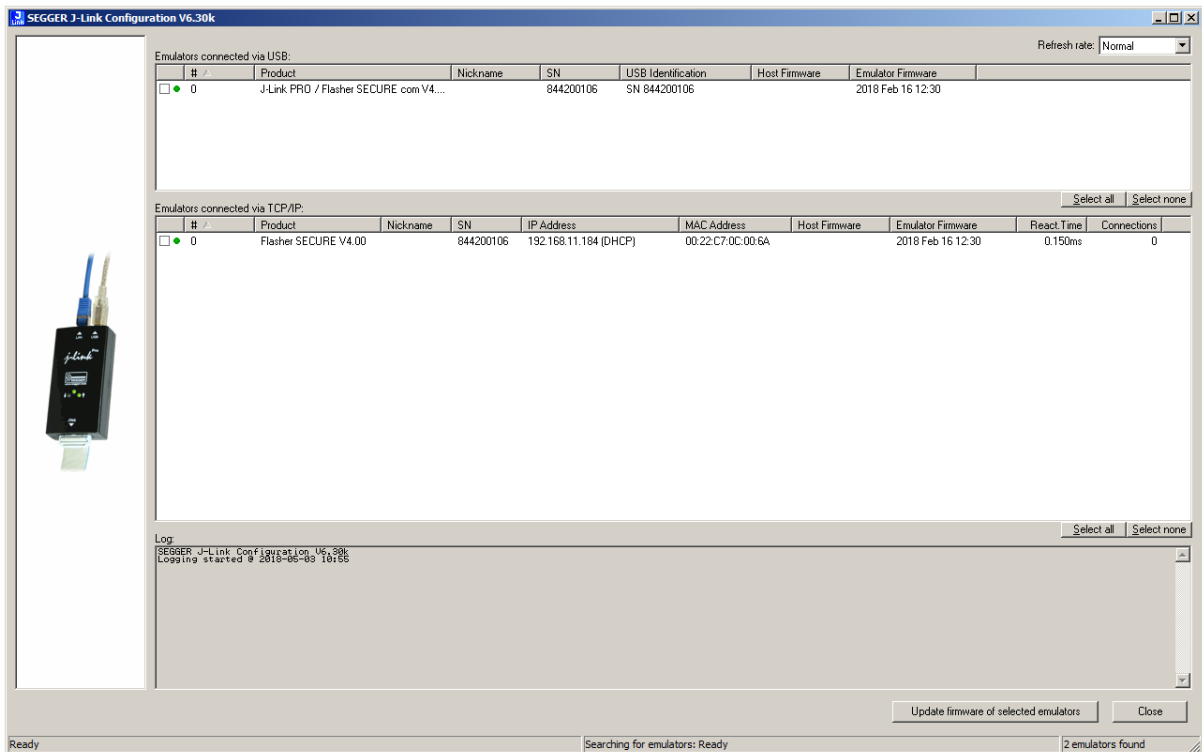
The Flasher Secure unit supports IPv4 only.

4.2.1 Manual IP configuration

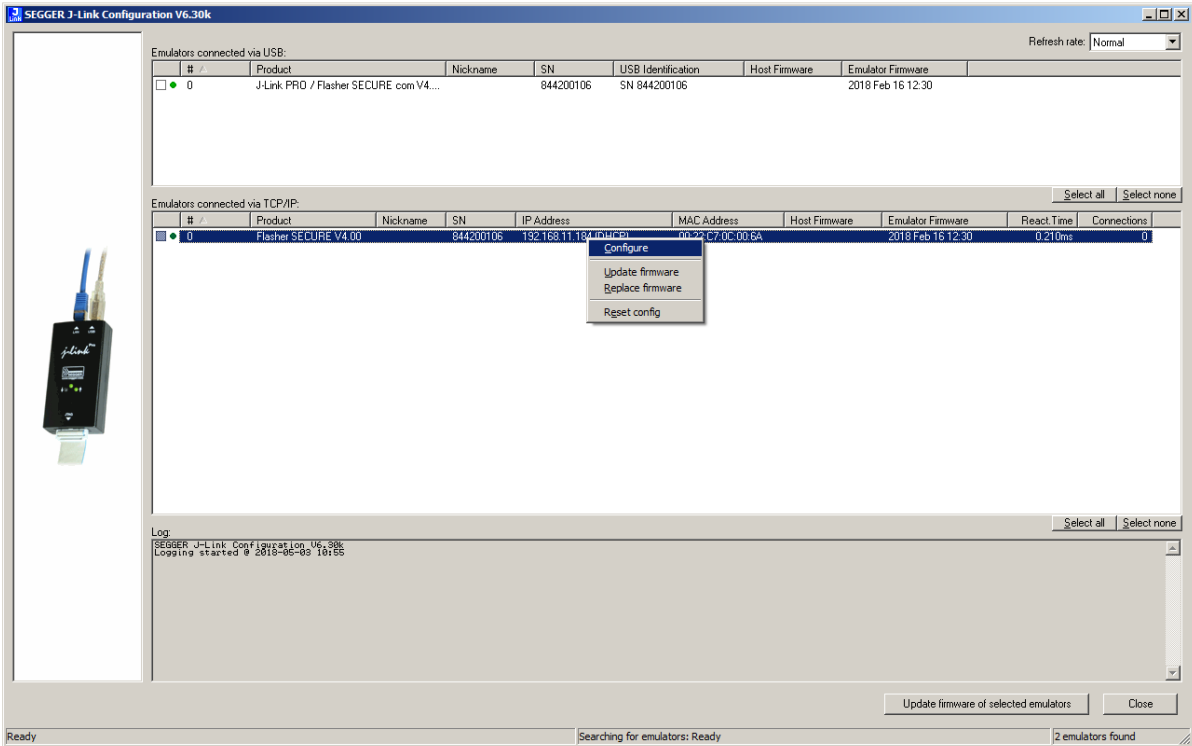
4.2.1.1 IP configuraton via J-Link Configurator

Start the J-Link Configurator and wait until your Flasher Secure was found.

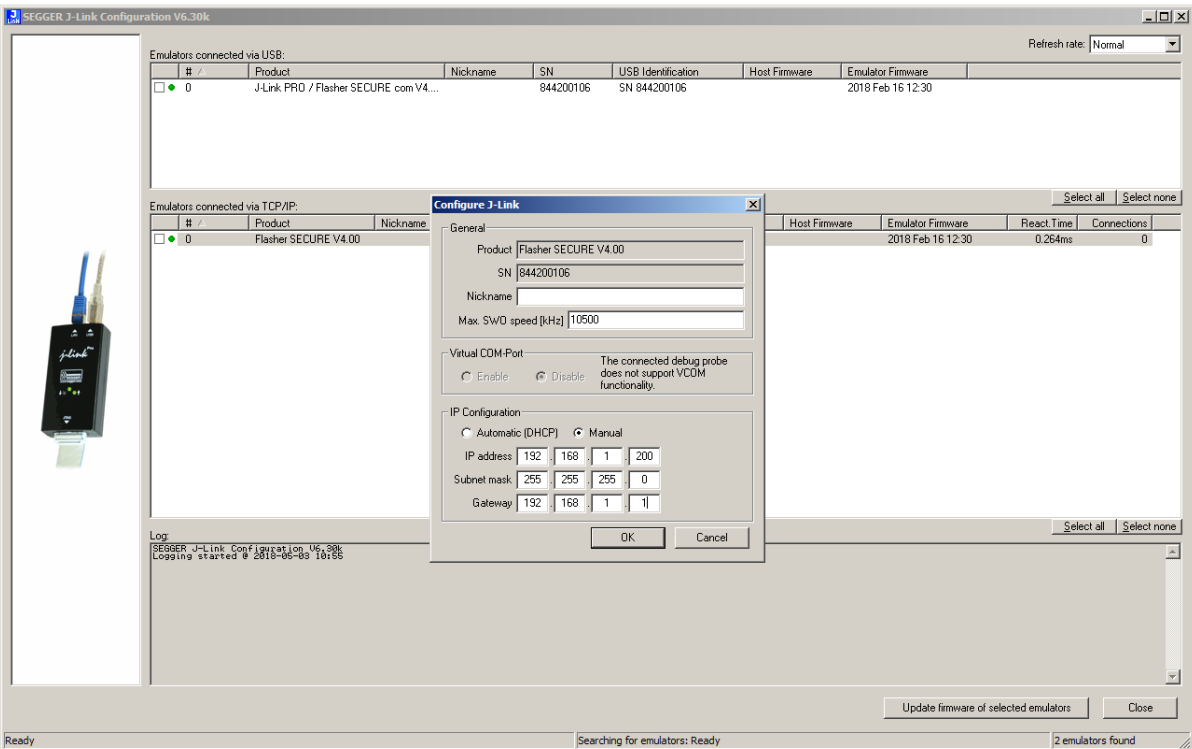
Choose your Flasher Secure in the list, e.g. in the IP connected device list and open the context menu with a right click on the device entry.



Choose the *Configure* entry in the context menu.

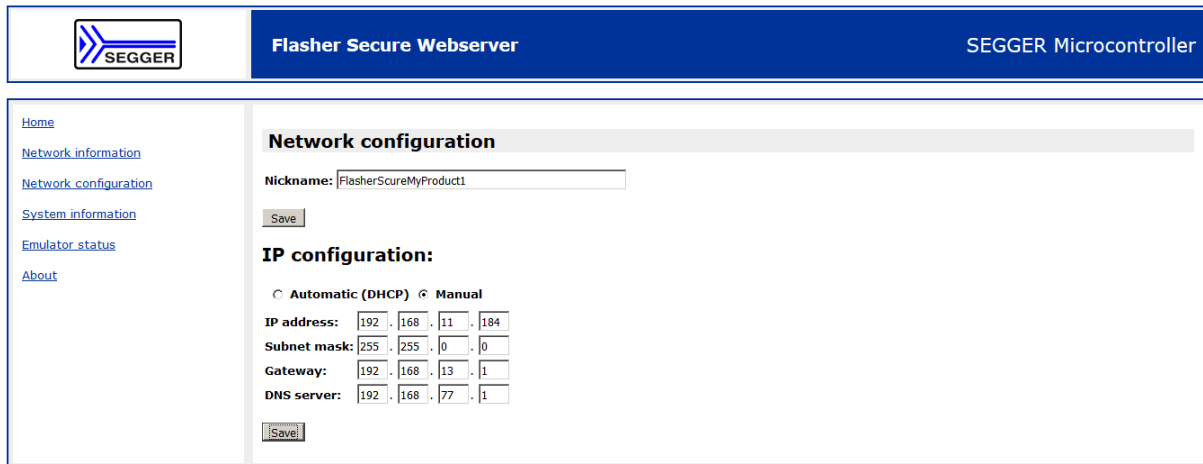


Enter the IP configuration in the pop-up menu and confirm it using the OK button.



4.2.1.2 IP configuration via integrated web server

Open the browser of your choice. Navigate to your Flasher Secure unit, e.g. `http://192.168.1.200`. Choose the *Network configuration* page in the navigation column on the left side.



The screenshot shows the 'Flasher Secure Webserver' interface. At the top, there is a blue header with the SEGGER logo on the left, 'Flasher Secure Webserver' in the center, and 'SEGGER Microcontroller' on the right. Below the header is a navigation menu on the left with links for 'Home', 'Network information', 'Network configuration', 'System information', 'Emulator status', and 'About'. The main content area is titled 'Network configuration' and contains a 'Nickname' field with the value 'FlasherSecureMyProduct1' and a 'Save' button. Below this is the 'IP configuration' section, which has two radio buttons: 'Automatic (DHCP)' (selected) and 'Manual'. Under the 'Manual' option, there are four rows of IP configuration fields: 'IP address' (192, 168, 11, 184), 'Subnet mask' (255, 255, 0, 0), 'Gateway' (192, 168, 13, 1), and 'DNS server' (192, 168, 77, 1). Each field is a small input box. A 'Save' button is located at the bottom of the IP configuration section.

Enter your IP configuration on the page and confirm it by pressing the save button.

Note

The browser may claim the device is no longer present after you changed the IP address. In this case, you need to reenter the correct IP address in the navigation bar of the browser.

4.3 Project Configuration

4.3.1 Configuration files

The Flasher Secure unit has internal and external storage. The internal storage is about 120MB in size and not accessible from the outside. It is used as cache memory for the firmware and configuration files provided by the server. 16MB of external storage are available for the configuration and log files.

The Flasher Secure needs two configuration files, *SECURE.INI* and *SERVER.CRT*.

A typical configuration file *SECURE.INI* contains the following information:

```
[SERVER]
Host = "FlasherSecureServer.Company.com"
Port = "3110"

[LOGIN]
Company = "Company"
Project = "MyProject"
Username = "MyManufacturer1"
Password = "1234abcd!!!!"
```

Entry	Meaning
Server section	defines the server settings
Host	the server host name or IP address
Port	the server port
Login section	the login setup
Company	the company name
Project	the project name
Username	the user name for the Flasher Secure Server login; must be one of the manufacturers logins.
Password	the password for the login

Note

The SERVER section defines the host and the port of the server to connect to. The host may either be a name or an IP address. If it is a name, a DNS server has to be configured in the IP configuration if necessary. The LOGIN section contains the login data corresponding to a manufacturer account on the server.

The other configuration file, *SERVER.CRT*, is the server's TLS certificate. It is used to restrict the Flasher Secure device to only accept connections to servers using the key in the certificate (server identity pinning). Although this file is optional, we recommend to use it. It provides a simple and fool-proof way to avoid man-in-the-middle attacks.

Note

The Flasher Secure supports only RSA certificates for the server validation.

4.3.2 Accessing the configuration files

You have three ways to access the configuration files:

- Using the USB MSD mode
- Using the built-in FTP server
- Using the ASCII command interface via the RS232 connector

4.3.2.1 Access via USB MSD mode

When the Flasher Secure is running in “MSD mode” (**M**ass **S**torage **D**evice), the 16MB external storage is available as a disk drive. To enter MSD mode, power up the device by connecting the USB cable while holding down the Start/Stop button. Keep the button pressed for at least another two seconds after power-up.

4.3.2.2 Access via FTP

The Flasher Secure has a built-in FTP server, meaning you can connect to it using an FTP client of your choice. The FTP server comes with 2 logins.

Login	Password	Privileges
admin	1234	can read, write, upload, download, delete files
anonymous	<i>none</i>	can only read or download files

Note

These logins cannot be changed.

4.3.2.3 Access via ASCII command protocol

The files can also be written via the ASCII command protocol. Refer to chapter *Commands and replies* on page 98 and the included file commands.

4.4 Programming devices

The Flasher Secure is typically used to program the target devices in a production setup. Therefore it will erase, flash and secure the target devices in one programming cycle once the setup is configured.

The Flasher Secure can be controlled in four ways:

- Using the push button on the device
- Using the 3-wire handshake interface integrated in the RS232 connector
- Using the ASCII command protocol via RS323 or Telnet

For all options, the details are described in chapter *Remote Control* on page 94.

A typical programming cycle for a target device may look like this:

Input

```
#AUTO
```

Flasher Secure Response

```
#ACK  
#STATUS:INITIALIZING  
#STATUS:CONNECTING  
#STATUS:UNLOCKING  
#STATUS:ERASING  
#STATUS:PROGRAMMING  
#STATUS:VERIFYING  
#OK (Total 13.993s, Erase 0.483s, Prog 9.183s, Verify 2.514s)
```

4.5 LED status indicators

Progress and result of an operation is indicated by Flasher Secure's LEDs. The behavior is different for PC-based and stand-alone mode. For a definition of the different modes, please refer to *UM08001: J-Link / J-Trace User Guide*, chapter Operating modes.

The following table describes the behavior of the LEDs in stand-alone mode.

#	Status of LED	Meaning
0	GREEN constant	Flasher Secure waits for a start trigger to perform an operation in stand-alone mode.
1	GREEN slow blinking	Flashing operation in progress: <ul style="list-style-type: none"> Erasing (slow blinking on/off time: 80 ms => 6.25 Hz) Programming (slow blinking on/off time: 300ms => ~1.67 Hz) Verifying (slow blinking, on/off time: 100ms => 5 Hz)
2	GREEN: constant RED: off or constant	GREEN constant, RED off: Operation successful. GREEN constant, RED constant: Operation failed Goes back to state #0 automatically, but in case of operation failed, RED remains on until the next time state #1 is entered.

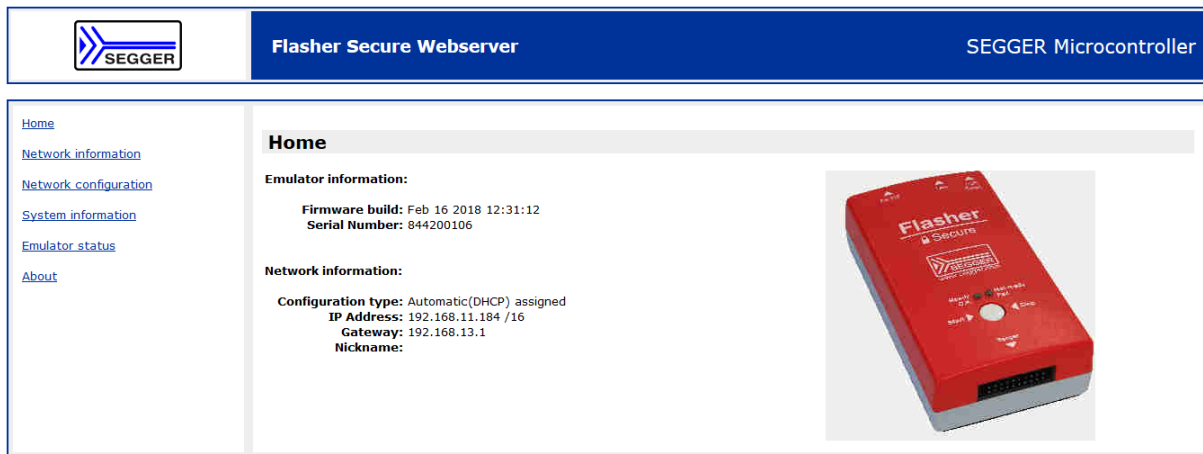
4.6 Web interface

The Flasher Secure has a built-in web interface. This interface shows information of the current status and IP configuration of the unit.

4.6.1 Flasher Secure main page

The Flasher Secure main page shows the following information:

- Firmware version
- Serial number
- Network configuration
- Nickname



Flasher Secure Webserver SEGGER Microcontroller

[Home](#)
[Network information](#)
[Network configuration](#)
[System information](#)
[Emulator status](#)
[About](#)


Home

Emulator information:

Firmware build: Feb 16 2018 12:31:12
Serial Number: 844200106

Network information:

Configuration type: Automatic(DHCP) assigned
IP Address: 192.168.11.184 /16
Gateway: 192.168.13.1
Nickname:



4.6.2 Flasher Secure network information page

The Flasher Secure network information page shows the following information:

- IP configuration
- MAC address
- Nickname
- Memory buffer statistics
- Current network connections

The screenshot shows the Flasher Secure Webserver interface. The top navigation bar includes the SEGGER logo, the title "Flasher Secure Webserver", and "SEGGER Microcontroller". The left sidebar contains links for Home, Network information (highlighted), Network configuration, System information, Emulator status, and About. The main content area is titled "Network information" and contains the following sections:

- Configuration:** Configuration type: Automatic(DHCP) assigned; IP address: 192.168.11.184 /16; MAC address: 00:22:C7:0C:00:6A; Nickname:
- Memory usage:** Small buffers (64 byte): 32 / 32; Big buffers (592 byte): 6 / 6
- Network connections:** Connection info; List of TCP connections

Socket	Local	Peer	State	MTU/MSS	Retrans. delay	Idle time	Local window	Peer window
1	Any:19020	---	Listen	60/0	20190	1705150	0/0	0
2	Any:21	---	Listen	60/0	20190	1704090	0/0	0
3	Any:23	---	Listen	60/0	20190	1704010	0/0	0
4	Any:80	---	Listen	60/0	20190	1692140	0/0	0
15	192.168.11.184:21	192.168.11.145:50050	Established	576/536	2290	4560	1072/1072	64984
23	192.168.11.184:80	192.168.11.145:50062	Established	576/536	3450	0	1072/1072	65392

4.6.3 Flasher Secure network configuration

The Flasher Secure network configuration allows the setup of the network settings:

- IP address
- Subnet mask
- Gateway IP address
- Nickname


The screenshot shows the Flasher Secure Webserver interface for network configuration. The top navigation bar is the same as in the previous screenshot. The left sidebar contains links for Home, Network information, Network configuration (highlighted), System information, Emulator status, and About. The main content area is titled "Network configuration" and contains the following sections:

- Nickname:** FlasherSecureMyProduct1
- Save** button
- IP configuration:**
 - Automatic (DHCP) Manual
 - IP address:** 192 . 168 . 11 . 184
 - Subnet mask:** 255 . 255 . 0 . 0
 - Gateway:** 192 . 168 . 13 . 1
 - DNS server:** 192 . 168 . 77 . 1
 - Save** button

4.6.4 Flasher Secure system information page

The Flasher Secure system information page lists the following items:

- OS statistics
- Task statistics


Flasher Secure Webserver
SEGGER Microcontroller

[Home](#)
[Network information](#)
[Network configuration](#)
[System information](#)
[Emulator status](#)
[About](#)

System information

embOS statistics:

System info

```
Number of tasks      : 7
System time         : 1748650
System stack (size@base) : 512@0x20010B10
```

List of tasks


Id	Priority	Task names	Context switches	Task Stack
0x2000CD28	22	IP Stack	180091	432/1536@0x2000C728
0x20005F4C	17	FTPServer Child	26	1404/2048@0x20010D28
0x2000F210	15	Webserver	1225	288/2048@0x2000EA10
0x2000F25C	14	Webserver Child	3253	3280/3920@0x20012138
0x2000DF1C	10	Terminal UART	1	140/1536@0x2000D91C
0x20002830	9	J-Link Server	6	312/1024@0x20002430
0x20010A88	5	MainTask	873991	4200/5120@0x2000F688

The table above shows some real time information of the RTOS kernel used.
Information about your application can easily be displayed in the same way.

4.6.5 Flasher Secure emulator status page

The Flasher Secure emulator (i.e. Flasher Secure Unit) status page lists the following points:

- Target voltage (VTref)
- Target current consumption
- Used target interface
- Target power (on/off) (supplied by Flasher Secure)
- Reset status


Flasher Secure Webserver
SEGGER Microcontroller

[Home](#)
[Network information](#)
[Network configuration](#)
[System information](#)
[Emulator status](#)
[About](#)

Emulator status

Target voltage: 0.000 V

Current consumption: 0 mA

Target interface: SWD (200 kHz)

System uptime: 00 days 00 hours 29 min. 46 sec.

Target power: Off On

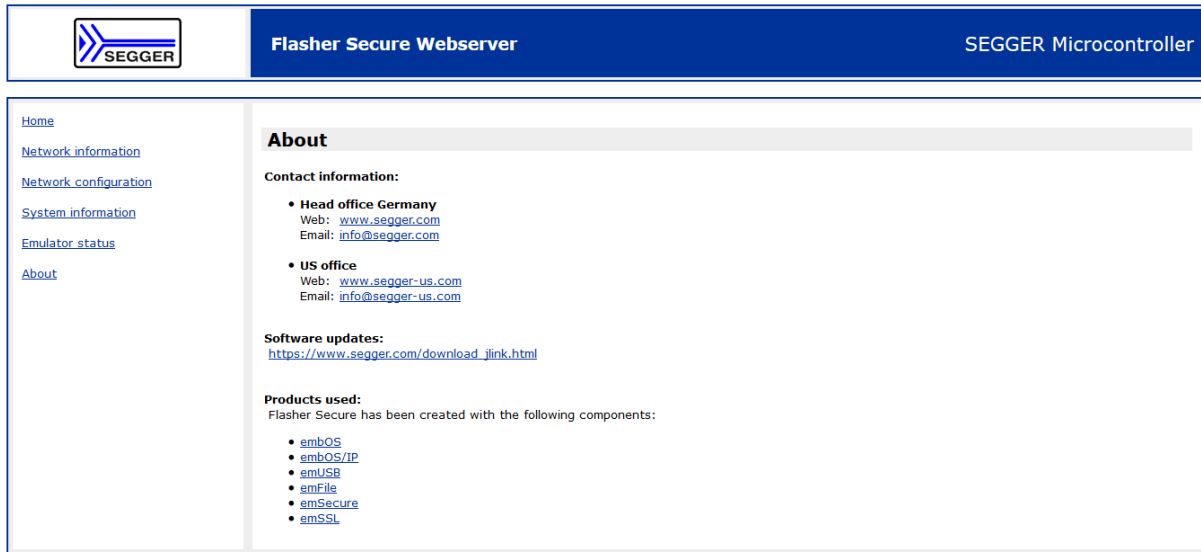
Reset state: Inactive


Flasher Secure User Guide (UM08032)

© 2021 SEGGER Microcontroller GmbH

4.6.6 Flasher Secure About page

The Flasher Secure About page lists links to additional information on the SEGGER web site.



 **Flasher Secure Webserver** SEGGER Microcontroller

[Home](#)
[Network information](#)
[Network configuration](#)
[System information](#)
[Emulator status](#)
[About](#)

About

Contact information:

- **Head office Germany**
Web: www.segger.com
Email: info@segger.com
- **US office**
Web: www.segger-us.com
Email: info@segger-us.com

Software updates:
https://www.segger.com/download_jlink.html

Products used:
Flasher Secure has been created with the following components:

- [smbOS](#)
- [smbOS/IP](#)
- [smUSB](#)
- [smFile](#)
- [smSecure](#)
- [smSSL](#)

4.7 Additional Information

4.7.1 Patch file support

This feature is currently not supported by Flasher Secure.

4.7.2 PC-based mode

The Flasher Secure can operate in PC-based mode. For further details, please have a look into the J-Link Manual (UM08001).

Note

In PC-based mode, the Flasher Secure is able to program target devices using the J-Flash software. The Flasher Secure programming process, including the signature generation, is only available in stand-alone mode programming.

Chapter 5

Remote control

This chapter describes how to control Flasher Secure via the 9-pin serial interface connector or via the integrated Telnet interface.

5.1 Overview

There are 4 ways to control Flasher Secure operation:

- Manual: Programming operation starts when pressing the programming button. The LEDs serve as visible indicators.
- Via Handshake lines: 3 lines on the serial interface are used:
 - 1 line is an input and can be used to start operation,
 - 2 lines are outputs and serve as busy and status signals.
- Terminal communication via RS232
- Terminal communication via Telnet

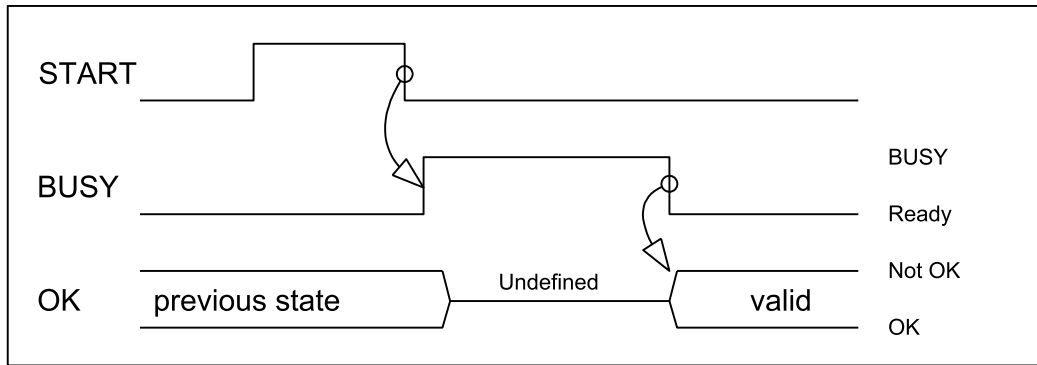
Note

The a.m. ways to control Flasher operation work only if Flasher Secure is in standalone mode. In PC-based or MSD mode they have no effect.

5.2 Handshake control

The Flasher Secure can be remote-controlled by automated testers without the need of a connection to a PC. Therefore the Flasher Secure is equipped with additional hardware control functions, which are connected to the SUBD9 male connector, normally used as RS232 interface to a PC.

The following diagrams show the internal remote control circuitry of the Flasher Secure:



Pin No.	Function	Description
1	START	A positive pulse of any voltage between 5 and 30V with duration of min. 30 ms starts "Auto" function (Clear / Program / Verify) on falling edge of pulse. The behavior of the "Auto" function depends on the project settings, chosen in J-Flash at the Production tab.
4	BUSY	As soon as the "Auto" function is started, BUSY becomes active, which means that transistor is switched OFF.
5	GND	Common Signal ground.
7	OK	This output reflects result of last action. It is valid after BUSY turned back to passive state. The output transistor is switched ON to reflect OK state.

5.3 ASCII command interface

5.3.1 Introduction

Once set up using J-Flash, the Flasher Secure can be driven by any application or just a simple terminal using ASCII commands.

Every known command is acknowledged by the Flasher Secure and then executed. After command execution, the Flasher Secure sends an ASCII reply message.

Note

There are situations where the execution of a known command is rejected with `#NACK:ERRxxx` if the Flasher Secure is currently busy and the received command is not allowed to be sent while the Flasher Secure is busy.

5.3.2 General command and reply message format

- Any ASCII command has to start with the start delimiter `#`.
- Any ASCII command has to end with simple carriage return (`\r`, ASCII code 13).
- Commands can be sent in upper or lower case

5.3.3 General usage

Reply messages must be considered in each case. In general, a new command must not be sent before a reply for the last one has been received.

5.3.4 Settings for ASCII interface via RS232

The Flasher Secure is driven via a RS232 serial port with the following interface settings:

- 9600 baud
- 8 data bits
- no parity
- 1 stop bit

The baud rate can be changed by using the `#BAUDRATE` command. (see `#BAUDRATE` on page 100)

5.3.5 Settings for ASCII interface via Telnet

A client application can connect to Flasher via Telnet on port 23. Find below a screenshot of Flasher which is remote controlled via Telnet:

```

10.0.0.43 - PuTTY
J-Link PRO / Flasher SECURE telnet-shell.
J-Link PRO / Flasher SECURE compiled Jun 23 2021 19:30:32
>#AUTO
#ACK
#STATUS:CONNECTING TO SERVER
#STATUS:LOGIN
#STATUS:INITIALIZING
#STATUS:CONNECTING
#STATUS:ERASING
#STATUS:PROGRAMMING
#STATUS:VERIFYING
#OK (Total 2.369s, Erase 0.081s, Prog 0.326s, Verify 0.140s)

```

5.3.6 Commands and replies

The table below provides an overview about the commands which are supported by the Flasher Secure firmware.

Commands to the Flasher Secure	Meaning
#BAUDRATE<Baudrate>	sets the baud rate of the interface
#AUTO	programs target device
#AUTO NOINFO	programs target device with fewer output messages
#CANCEL	cancels a command
#ERASE	erases the memory
#PROGRAM	programs the memory
#RESULT	requests the result of the most recent action
#SELECT <Filename>	selects the active project
#START	starts the device
#STATUS	requests the current status of the Flasher unit
#VERIFY	verifies the memory content
File I/O commands	
#FCLOSE	closes a file
#FCRC	calculates the CRC for a file
#FDELETE <Filename>	deletes a file
#FOPEN <Filename>	opens a file
#FREAD <Offset>,<NumBytes>	reads a file

Commands to the Flasher Secure	Meaning
#FSIZE	requests the file size
#FWRITE <Offset>, <Num-Bytes>: <Data>	writes a file
#FLIST	lists the files
#MKDIR <Dirname>	creates a directory
Replies from the Flasher	
#ACK	answer command acknowledged
#NACK	answer command not acknowledged
#OK	answer action finished with status OK
#OK: <NumBytes>: <Data>	answer requested bytes
#OK: <Size>	answer requested size
#STATUS:	answer requested status
#ERRxxx	answer action finished with error

Note

Not all commands are supported by all Flasher Types.

5.3.6.1 Commands to the Flasher

5.3.6.1.1 #AUTO

The #AUTO command behaves exactly as the start button or external remote control input. Usually, the following command sequence will be performed when receiving the #AUTO command:

- The Flasher erases the target CPU (if not blank)
- The Flasher programs the target CPU
- The Flasher verifies the target CPU

Depending on the settings chosen in the **Production** tab in J-Flash, this sequence can differ from the one shown above.

Finally, Flasher responds with

- #OK if no error occurred
- #ERRxxx if any error occurred during operation. xxx represents the error code, normally replied to Flasher PC program. The #ERRxxx message may be followed by an additional error text.

During execution of the #AUTO command, the Flasher Secure automatically sends "status" messages via RS232 to reflect the state of execution. Typically during execution of the #AUTO command, the Flasher Secure will reply the following sequence of messages:

```
#ACK
#STATUS:WAITING FOR IP CONFIGURATION
#STATUS:CONNECTING TO SERVER
#STATUS:LOGIN
#STATUS:INITIALIZING
#STATUS:CONNECTING
#STATUS:UNLOCKING
#STATUS:ERASING
#STATUS:PROGRAMMING
#STATUS:VERIFYING
#OK (Total x.xxxx, Erase x.xxxx, Prog x.xxxx, Verify x.xxxx)
```

5.3.6.1.2 #AUTO NOINFO

This command may be used instead of #AUTO, if no status messages from Flasher should be sent during execution. The NOINFO extension is also available for all other commands.

The command ends with #OK or #ERRxxx

5.3.6.1.3 #BAUDRATE<Baudrate>

This command can be sent in order to change the baud rate of the Flasher Secure's RS232 interface used for communication. <Baudrate> is expected in decimal format. The valid range is from 2400 baud to 115200 baud.

If the command succeeds, the Flasher Secure responds with:

```
#ACK
#OK
```

Otherwise, it will respond with one of the following error messages:

```
#ERR255: Invalid parameters
or
#ERR255: Baudrate is not supported
```

Note

After sending the #BAUDRATE command, you will first have to wait until the Flasher Secure responds with the #OK message. It is recommended to wait 5ms before sending

the next command with the new baud rate in order to give the Flasher Secure the time to change the baud rate.

#CANCEL

This command can be sent to abort a running program. It may take a while until the current program is actually canceled.

The Flasher Secure will respond with:

```
#ERR007: CANCELED.
```

#ERASE

This command can be sent to erase all selected target flash sectors.

The Flasher Secure will reply the following sequence of messages:

```
#ACK
#STATUS: INITIALIZING
#STATUS: CONNECTING
#STATUS: UNLOCKING
#STATUS: ERASING
#OK (Total x.xxxx, Erase x.xxxx)
```

#PROGRAM

This command can be used instead of #AUTO to program a target without erasing the target before programming and without performing a final verification.

#RESULT

This command can be sent any time, even during other command execution. Flasher responds with the result of the previously executed command.

#SELECT <Filename>

The #SELECT command is used to select a specific config and data file pair which should be used by the Flasher Secure to program the target. <Filename> specifies the name of file pair without extensions (.CFG and .DAT) on the Flasher Secure which should be selected. The Flasher Secure saves the selected config and data file in the FLASHER.INI file. So this selection is remembered even after power-cycling the Flasher Secure.

This may be very helpful in cases where several config and data files are stored on the Flasher Secure. The user can easily switch between these config and data files without connecting the Flasher Secure to a host.

If the command succeeds, Flasher responds with:

```
#ACK
#OK
```

Below is an example sequence which shows how to use the #SELECT command:

```
#SELECT ATSAM7_1 // ATSAM7_1.CFG and ATSAM7_1.DAT are selected
#ACK
#OK
#AUTO // Start auto programming
#ACK
#STATUS: INITIALIZING
#STATUS: CONNECTING
#STATUS: UNLOCKING
#STATUS: ERASING
#STATUS: PROGRAMMING
#STATUS: VERIFYING
#OK (Total 8.416s, Erase 0.005s, Prog 6.845s, Verify 0.959s)
#SELECT ATSAM7_2 // ATSAM7_2.CFG and ATSAM7_2.DAT are selected
```

```
#ACK
#OK
#AUTO // Start auto programming
#ACK
#STATUS:INITIALIZING
#STATUS:CONNECTING
#STATUS:UNLOCKING
#STATUS:ERASING
#STATUS:PROGRAMMING
#STATUS:VERIFYING
#OK (Total 8.632s, Erase 0.005s, Prog 7.051s, Verify 0.969s)
```

#START

This command can be sent to release Flasher Secure's target interface. All signals from the Flasher Secure to the target will be set into high-Z mode, and the reset pin of the target will be released. This command may be used to start the target application program.

Flasher will reply with the following sequence of messages:

```
#ACK
#STATUS:INITIALIZING
#STATUS:CONNECTING
#OK (Total x.xxxx)
```

#STATUS

This command can be sent any time, even during other command execution. The Flasher Secure responds with its current state. All defined state messages are described under *Replies from Flasher Secure* on page 105.

#VERIFY

This command can be used to verify the target flash content against the data stored in the Flasher Secure.

5.3.6.2 File I/O commands

The ASCII interface of the Flasher Secure also supports file I/O operations.

The following file I/O commands are supported:

#FCLOSE

The #FCLOSE command closes the file on the Flasher Secure that was opened via #FOPEN. After this command has been issued, further file I/O operations except #FDELETE are not allowed until the #FOPEN command is sent again.

A typical sequence when using the #FCLOSE command does look like as follows:

```
#FCLOSE
#ACK
#OK
```

Note

When using the #FCLOSE command, a file has to be open (previously opened by #FOPEN). Otherwise, the Flasher Secure will respond with the following:

```
#ACK
#ERR255:No file opened
```

#FCRC

The #FCRC command calculates a 32-bit CRC of the given file. This CRC can be used to verify file integrity. This command should not be used while a file has been opened via #FOPEN. The CRC will be also be reported by J-Flash when downloading or saving files via J-Flash.

A typical sequence when using the #FCRC command looks as follows:

```
#FCRC flasher.dat
#ACK
#OK:0x75BC855A
```

#FDELETE <Filename>

The #FDELETE command is used to delete a file on the Flasher Secure, where <Filename> specifies the name of the file.

A typical sequence when using the #FDELETE command looks as follows:

```
#FDELETE flasher.dat
#ACK
#OK
```

Note

If deletion of the file fails for example if the file does not exist, Flasher will respond with the following sequence:

```
#ACK
#ERR255:Failed to delete file
```

#FOPEN <Filename>

The #FOPEN command is used to open a file on the Flasher Secure for further file I/O operations. <Filename> specifies the file on the Flasher Secure that should be opened. If <Filename> can not be found on the Flasher Secure, a new one will be created.

A typical sequence using the #FOPEN command looks as follows:

```
#FOPEN flasher.dat
#ACK
#OK
```

Note

Currently, only one file can be open at the same time. If #FOPEN is sent and another file is already open, the Flasher Secure will respond with:

```
#ACK
#ERR255:A file has already been opened
```

#FREAD <Offset>,<NumBytes>

The #FREAD command is used to read data from a file on the Flasher Secure. <Offset> specifies the offset in the file, at which data reading is started. <NumBytes> specifies the number of bytes which should be read.

A typical sequence when using the #FREAD command looks as follows:

```
#FREAD 0,4
#ACK
#OK:04:466c6173
```

If the #FREAD command succeeds, the Flasher Secure will respond with a #OK:<NumBytes>:<Data> reply message. For more information about the Flasher reply messages, please refer to *Replies from Flasher* on page 105.

Note

In order to use the #FREAD command, a file has to be opened via the #FOPEN command. Otherwise the Flasher Secure will respond with the following sequence:

```
#ACK
#ERR255:No file opened
```

#FSIZE

The #FSIZE command is used to get the size of the currently open file on the Flasher Secure.

A typical sequence when using the #FSIZE command looks as follows:

```
#FSIZE
#ACK
#OK:10 // currently open file on Flasher Secure has a size of 16 bytes
```

If the #FSIZE command succeeds, Flasher will respond with a #OK:<Size> reply message. For more information about the Flasher reply messages, please refer to *Replies from Flasher* on page 105.

Note

In order to use the #FREAD command, a file has to be opened via the #FOPEN command. Otherwise, the Flasher Secure will respond with the following sequence:

```
#ACK
#ERR255:No file opened
```

#FWRITE <Offset>,<NumBytes>:<Data>

The #FWRITE command is used to write to a file on the Flasher Secure. <Offset> specifies the offset in the file, at which data writing is started. <NumBytes> specifies the number of bytes that are sent with this command and that are written into the file on the Flasher Secure. <NumBytes> is limited to 512 bytes at a time. This means, if you want to write e.g. 1024 bytes, you have to send the #FWRITE command twice, using an appropriate offset when sending it the second time.

<Offset> and <NumBytes> are expected in hexadecimal format.

```
#FWRITE 0,200:<Data>
#FWRITE 200,200:<Data>
```

The data is expected in hexadecimal format (two hexadecimal characters per byte). The following example illustrates the use of #FWRITE:

```
Data to be sent: Hello !
ASCII values: 0x48, 0x65, 0x6C, 0x6C, 0x6F, 0x20, 0x21
#FWRITE 0,7:48656C6C6F2021
```

Note

In order to use the #FWRITE command a file has to be opened via the #FOPEN command, first. Otherwise Flasher will respond with the following sequence:

```
#ACK
#ERR255:No file opened
```

#FLIST

The #LIST command is used to list all files stored on the Flasher Secure.

A typical sequence using the #FLIST command looks as follows:

```
#FLIST
#ACK
FLASHER.INI Size: 60
SERIAL.TXT Size: 3
FLASHER.LOG Size: 207
FOLDER (DIR)
FOLDER\TEST1.CFG Size: 2048
FOLDER\TEST1.DAT Size: 12288
#OK
```

#MKDIR <Dirname>

The #MKDIR command is used to create a directory on the Flasher Secure. <Dirname> specifies the name of the new directory. <Dirname> may also specify a path to create a sub-directory.

A typical sequence using the #MKDIR command looks as follows:

```
#MKDIR folder
#ACK
#OK
```

Note

If the directory cannot be created because of a bad <Dirname> argument, the Flasher Secure will respond with:

```
#ACK
#ERR255:Failed to create directory
```

5.3.6.3 Replies from Flasher

The reply messages from Flasher follow the same data format as commands. Any reply message starts with ASCII start delimiter #, ends with simple carriage return (ASCII code 13) and is sent in uppercase. In contrast to commands, replies can be followed by a descriptive message, which provides more detailed information about the reply. This description is sent in mixed case. The #OK reply, for example, is such a reply. It is followed by a string containing information about the performance time needed for the operations:

```
#OK (Total 13.993s, Erase 0.483s, Prog 9.183s, Verify 2.514s)
```

The following reply messages from Flasher are defined:

#ACK

Flasher replies with #ACK message on reception of any defined command before the command itself is executed.

#NACK

Flasher replies with #NACK, if an undefined command was received.

#OK

Flasher replies with #OK, if a command other than #STATUS or #RESULT was executed and ended with no error.

#OK:<NumBytes>:<Data>

Flasher replies with #OK:<Len>:<Data> if an #FREAD command was executed. <NumBytes> is the number of bytes that could be read. This value may differ from the number of requested bytes, for example if more bytes than available were requested. <NumBytes> and <Data> are sent in hexadecimal format (for <Data>: two hexadecimal characters per byte).

#OK:<Size>

The Flasher Secure replies with #OK:<Size> if a #FSIZE command has been executed. <Size> is the size (in bytes) of the currently opened file. <Size> is sent in hexadecimal format.

#STATUS:

The Flasher replies with its current state.

The following status messages are currently defined:

Message	Description
#STATUS:READY	Flasher Secure is ready to receive a new command
#STATUS:CONNECTING	Flasher Secure is initiating a connection to the target CPU
#STATUS:INITIALIZING	Flasher Secure is performing self check and internal initialization
#STATUS:UNLOCKING	Flasher Secure is unlocking flash sectors
#STATUS:ERASING	Flasher Secure is erasing the flash of the target device
#STATUS:PROGRAMMING	Flasher Secure is programming the flash of the target device
#STATUS:VERIFYING	Flasher Secure is verifying the programmed flash contents

#ERRxxx

If any command other than #STATUS or #RESULT was terminated with an error, the Flasher Secure cancels the command and replies with an error message instead of #OK message.

Some error codes may be followed by a colon and an additional error text.

For example:

```
#ERR007:CANCELED.
```

The error code numbers are described in the following table:

Message	Description
#ERR007	Flasher received #CANCEL command and has canceled the current operation
#ERR008	Flasher is already busy with execution of previous command
#ERR009	Failed to allocate memory
#ERR010	Failed to open file
#ERR011	Failed to read file
#ERR012	Failed to write file
#ERR013	Failed to delete file
#ERR255	Undefined error occurred (this reply is followed by an error string)

Chapter 6

Device Firmware

The firmware of the target device needs to implement the emSecure software package. The security features of the emSecure software package are used to verify if the flashed device signature is valid or not.

6.1 emSecure Package

The emSecure software package comes with its own detailed documentation. Therefore, only the main topics are described in this manual. For details about the emSecure library, please take a closer look at the UM12002 emSecure RSA or UM 12004 emSecure ECDSA manuals.

6.2 Implementing emSecure

The emSecure software package is part of the Flasher Secure shipment.

6.2.1 Package content:

emSecure is provided in source code and contains everything needed. The following table shows the content of the emSecure package:

Files	Description
Application	Sample applications for bare metal and embOS
Config	Configuration header files
Doc	emSecure documentation
CRYPTO	Shared cryptographic library source code
SECURE	emSecure implementation code
SEGGER	SEGGER software component source code used in emSecure
Sample/Config	Example emSecure configuration
Sample/Keys	Example emSecure key pairs
Windows	Supporting applications in binary and source form

6.2.2 Include directories

Please make sure that the include path contains the following directories (the order of inclusion is of no importance):

- Config
- CRYPTO\Inc
- SECURE\Inc
- SEGGER
- SETUP

Add the source files from the following directories to your project:

- CRYPTO
- SECURE
- SEGGER
- SETUP\Crypto

Note

Always make sure that you have only one version of each file!

It is frequently a major problem when updating to a new version of emSecure-RSA if you have old files included and therefore mix different versions. If you keep emSecure-RSA in the directories as suggested (and only in these), this type of problem cannot occur. When updating to a newer version, you should be able to keep your configuration files and leave them unchanged. For safety reasons, we recommend backing up (or at least renaming) the SECURE directories before updating.

6.3 Sample Application

The Flasher Secure is shipped with a sample application which looks like the listed example below.

Note

The sample application is built based on the embOS OS. emSecure library can be used without an RTOS.

```

/*****
 *
 *          SEGGER Microcontroller GmbH
 *          The Embedded Experts
 *
 *
 *          (c) 1995 - 2021 SEGGER Microcontroller GmbH
 *
 *          www.segger.com      Support: support@segger.com
 *
 *
 *          embOS * Real time operating system for microcontrollers
 *
 *
 *          Please note:
 *
 *          Knowledge of this file may under no circumstances
 *          be used to write a similar product or a real-time
 *          operating system for in-house use.
 *
 *          Thank you for your fairness !
 *
 *****/

-----
File      : FlasherSecureVerifyExample.c
Purpose   : Example code for UID signature check.
----- END-OF-HEADER -----
*/

#include "RTOS.h"
#include "BSP.h"
#include "stm3210xx.h"

#include "SECURE_RSA.h"
#include "SECURE_ECDSA.h"

#include "RSA_PrivateKey.h"
#include "RSA_PublicKey.h"

#include "ECDSA_PrivateKey.h"
#include "ECDSA_PublicKey.h"

int printf(const char *fmt,...);

/*****
 *
 *          Defines, fixed
 *
 *****/
*/

//
// Unique device ID register (96 bits)
//

```

```

typedef struct {
    volatile U32 UID_0;
    volatile U32 UID_1;
    volatile U32 UID_2;
} UID_Type;

#define UID ((UID_Type *)0x1FF80050)

/*****
 *
 *     Defines, configurable
 *
 *****/

// choose the used cryptographic method here!
// #define ECDSA          1 // Use ECDSA
#define RSA              1 // Use RSA

/*****
 *
 *     Static data
 *
 *****/

static OS_STACKPTR int StackHP[128];           /* task stack */
static OS_TASK      TCBHP;                    /* task control block */

static const volatile U8
    __attribute__((section (".otp.SIG"))) _aSignature[256] = { [0 ... 255] = 0xFF };

static const volatile U64 __attribute__((section (".otp.SN")))
    _SerialNo = 0xFFFFFFFFFFFFFFFF;

/*****
 *
 *     Local functions
 *
 *****/

static void _HexDump(const U8 *Data, int NumBytes) {
    int i;
    for (i = 0; i < NumBytes; i++) {
        if (!(i % 16)) {
            printf("\n");
        } else if (!(i % 4)) {
            //printf(" ");
        }
        //printf ("%02X", *Data);
        printf ("0x%.2X, ", *Data);
        Data++;
    }
    printf("\n");
}

#ifdef RSA
static int _VerifyRSA(unsigned char *ID, int NumBytes, int SigBytes) {
    return SECURE_RSA_Verify(&RSAPublicKey, 0, 0, ID, NumBytes, (U8*)_aSignature, SigBytes);
}
#endif
#ifdef ECDSA
static int _VerifyECDSA(unsigned char *ID, int NumBytes, int SigBytes) {
    SECURE_ECDSA_PUBLIC_KEY PublicKey = { _ECDSAPublicKey, &SECURE_ECDSA_CURVE_P192 };
    return SECURE_ECDSA_Verify(&PublicKey, ID, NumBytes, (U8*)_aSignature, SigBytes);
}
#endif
#endif

```

```

/*****
 *
 *      HPTask
 */
static void HPTask(void) {
    while (1) {
        BSP_ToggleLED(0);
        OS_Delay (50);
    }
}

/*****
 *
 *      Global functions
 *
 *****/

/*****
 *
 *      MainTask()
 */
#ifdef __cplusplus
extern "C" { /* Make sure we have C-declarations in C++ programs */
#endif
void MainTask(void);
#ifdef __cplusplus
}
#endif
void MainTask(void) {
    char FW_OK = 0; // check result
    U8  ID[16];     // 16 Bytes UID
    U32 t0;        // time measuring

    t0 = OS_GetTime();
    printf("Unique device ID:\n%.8X %.8X %.8X\n", UID->UID_0, UID->UID_1, UID->UID_2);
    SEGGER_memcpy(&ID[0], (U8*)UID, 12);
    memset(&ID[12], 0, 4); // zero padding

    printf("Combined device ID:");
    _HexDump(ID, sizeof(ID));
    printf("\n");

    printf("Serial number: %lld", _SerialNo);
    printf("\n");
#ifdef RSA
    if (_VerifyRSA(ID, sizeof(ID), 256) > 0) {
        printf("Verified UID is correctly signed... SUCCESS! (%dms)\n", OS_GetTime() - t0);
        FW_OK = 1;
    } else {
        printf("Signed UID did not verify... ERROR! (%dms)\n", OS_GetTime() - t0);
    }
    printf("\nSignature:");
    _HexDump((U8*)_aSignature, 256);
    printf("\n");
#endif
#ifdef ECDSA
    if (_VerifyECDSA(ID, sizeof(ID), 48) > 0) {
        printf("Verified UID is correctly signed... SUCCESS! (%dms)\n", OS_GetTime() - t0);
        FW_OK = 2;
    } else {
        printf("Signed UID did not verify... ERROR! (%dms)\n", OS_GetTime() - t0);
    }
    printf("\nSignature:");

```



```

    _HexDump((U8*)_aSignature, 48);
    printf("End.\n", OS_GetTime() - t0);
#endif
    if (FW_OK) {
        //
        // Firmware main task
        //
        OS_CREATETASK(&TCBHP, "HPTask", HPTask, 150, StackHP);
    }
    OS_TerminateTask(NULL);
}

/***** End of file *****/

```

Features of the example:

- Initializes the hardware required for the example
- Reads the unique ID of the device
- Verifies the signature of the device

6.3.1 The main function

The main function calls the required functions to get the unique ID, verify it and react to the result. Depending on the chosen cryptographic method, the `_VerifyRSA` or the `_VerifyECDSA` function is called. The `FW_OK` variable contains the verification result, and the firmware main functions, here represented by the `HPTask`, are started if it is set to a value greater than 1. Otherwise, the signature check failed and the firmware is not started.

6.3.2 The `_VerifyRSA` function

If the RSA method is chosen as cryptographic method, this function will check the signature of the given UID. It calls the corresponding `emSecure` library function and will report the result.

6.3.3 The `_VerifyECDSA` function

If the ECDSA method is chosen as cryptographic method, this function will check the signature of the given UID. It calls the corresponding `emSecure` library function and will report the result.

Note

The Flasher Secure software package includes either the RSA or the ECDSA cryptographic library.

Chapter 7

Hardware and Adapters

This chapter gives an overview about Flasher Secure specific hardware details, such as the pinouts and available adapters.

7.1 ARM 20-pin JTAG/SWD Connector

Flasher Secure has a JTAG connector compatible with ARM JTAG standard. The JTAG connector is a 20 way Insulation Displacement Connector (IDC) keyed box header (2.54mm male) that mates with IDC sockets mounted on a ribbon cable.

7.1.1 Pinout JTAG

VTref	1 ●	● 2	Vsupply
nTRST	3 ●	● 4	GND
TDI	5 ●	● 6	GND
TMS	7 ●	● 8	GND
TCK	9 ●	● 10	GND
RTCK	11 ●	● 12	GND
TDO	13 ●	● 14	GND
RESET	15 ●	● 16	GND
DBGREQ	17 ●	● 18	GND
V5-Supply	19 ●	● 20	GND

The following table lists the JTAG pinout.

PIN	SIGNAL	TYPE	Description
1	VTref	Input	This is the target reference voltage. It is used to check if the target has power, to create the logic-level reference for the input comparators and to control the output logic levels to the target. It is normally fed from Vdd of the target board and must not have a series resistor.
2	Vsupply	NC	This pin is not connected to Flasher Secure. It is reserved for compatibility with other equipment. Connect to Vdd or leave open in target system.
3	nTRST	Output	JTAG Reset. Output from Flasher Secure to the Reset signal of the target JTAG port. Typically connected to nTRST of the target CPU. This pin is normally pulled HIGH on the target to avoid unintentional resets when there is no connection.
5	TDI	Output	JTAG data input of target CPU. It is recommended that this pin is pulled to a defined state on the target board. Typically connected to TDI of target CPU.
7	TMS	Output	JTAG mode set input of target CPU. This pin should be pulled up on the target. Typically connected to TMS of target CPU.
9	TCK	Output	JTAG clock signal to target CPU. It is recommended that this pin is pulled to a defined state of the target board. Typically connected to TCK of target CPU.
11	RTCK	Input	Return test clock signal from the target. Some targets must synchronize the JTAG inputs to internal clocks. To assist in meeting this requirement, you can use a returned, and retimed, TCK to dynamically control the TCK rate. Flasher Secure supports adaptive clocking, which waits for TCK changes to be echoed correctly before making further changes. Connect to RTCK if available, otherwise to GND.
13	TDO	Input	JTAG data output from target CPU. Typically connected to TDO of target CPU.
15	RESET	I/O	Target CPU reset signal. Typically connected to the RESET pin of the target CPU, which is typically called "nRST", "nRESET" or "RESET".

PIN	SIGNAL	TYPE	Description
17	DBGRQ	NC	This pin is not connected in Flasher Secure. It is reserved for compatibility with other equipment to be used as a debug request signal to the target system. Typically connected to DBGRQ if available, otherwise left open.
19	5V-Target supply	Output	This pin is used to supply power to some eval boards. Typically left open on target hardware.

Pins 4, 6, 8, 10, 12, 14, 16, 18, 20 are GND pins connected to GND in Flasher Secure. They should also be connected to GND in the target system.

7.1.2 Pinout SWD

The 20-pin connector of Flasher Secure is also compatible to ARM's Serial Wire Debug (SWD) interface.

VTref	1 ●	● 2	Vsupply
Not used	3 ●	● 4	GND
Not used	5 ●	● 6	GND
SWDIO	7 ●	● 8	GND
SWCLK	9 ●	● 10	GND
Not used	11 ●	● 12	GND
SWO	13 ●	● 14	GND
RESET	15 ●	● 16	GND
Not used	17 ●	● 18	GND
V5-Supply	19 ●	● 20	GND

The following table lists the SWD pinout.

PIN	SIGNAL	TYPE	Description
1	VTref	Input	This is the target reference voltage. It is used to check if the target has power, to create the logic-level reference for the input comparators and to control the output logic levels to the target. It is normally fed from Vdd of the target board and must not have a series resistor.
2	Vsupply	NC	This pin is not connected in Flasher Secure. It is reserved for compatibility with other equipment. Connect to Vdd or leave open in target system.
3	Not Used	NC	This pin is not used by Flasher Secure. If the device may also be accessed via JTAG, this pin may be connected to nTRST, otherwise leave open.
5	Not used	NC	This pin is not used by Flasher Secure. If the device may also be accessed via JTAG, this pin may be connected to TDI, otherwise leave open.
7	SWDIO	I/O	Single bi-directional data pin.
9	SWCLK	Output	Clock signal to target CPU. It is recommended that this pin is pulled to a defined state of the target board. Typically connected to TCK of target CPU.
11	Not used	NC	This pin is not used by Flasher Secure. This pin is not used by Flasher Secure when operating in SWD mode. If the device may also be accessed via JTAG, this pin may be connected to RTCK, otherwise leave open.
13	SWO	Output	Serial Wire Output trace port. (Optional, not required for SWD communication.)

PIN	SIGNAL	TYPE	Description
15	RESET	I/O	Target CPU reset signal. Typically connected to the RESET pin of the target CPU, which is typically called "nRST", "nRESET" or "RESET".
17	Not used	NC	This pin is not connected in Flasher Secure.
19	5V-Target supply	Output	This pin is used to supply power to some eval boards. Not all J-Links supply power on this pin, only the KS (Kickstart) versions. Typically left open on target hardware.

Pins 4, 6, 8, 10, 12, 14, 16, 18, 20 are GND pins connected to GND in Flasher Secure. They should also be connected to GND in the target system.

7.1.3 Target power supply

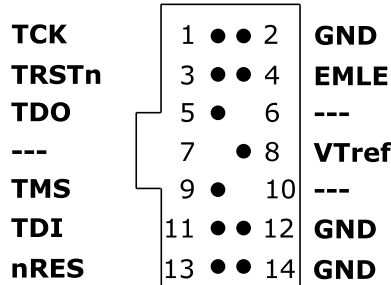
Pin 19 of the connector can be used to supply power to the target hardware. Supply voltage is 5V, max. current is 400mA. The output current is monitored and protected against overload and short-circuit.

Power can be controlled via the J-Link commander. The following commands are available to control power:

Command	Explanation
<code>power on</code>	Switch target power on
<code>power off</code>	Switch target power off
<code>power on perm</code>	Set target power supply default to "on"
<code>power off perm</code>	Set target power supply default to "off"

7.2 Flasher RX 14-pin connector

Flasher Secure itself has a 20-pin JTAG connector mounted but an optional J-Link 14-pin RX adapter is available. This adapter enables Flasher Secure to optionally power the connected target hardware. On the adapter there is a jumper which allows selection between 3.3V and 5V supply target voltage supply. The target is supplied via the VTref connection when the supply option is jumpered.



The following table lists the J-Link RX 14-pin JTAG pinout.

Pin	Signal	Type	Description
1	TCK	Output	JTAG clock signal to target CPU. It is recommended that this pin is pulled to a defined state on the target board. Typically connected to TCK on target CPU.
3	TRSTn	Output	JTAG Reset. Output from Flasher Secure to the Reset signal of the target JTAG port. Typically connected to nTRST of the target CPU. This pin is normally pulled HIGH on the target to avoid unintentional resets when there is no connection.
4	EMLE	Output	Pin for the on-chip emulator enable signal. When the on-chip emulator is used, this pin should be driven high. When not used, it should be driven low. Pulled HIGH to VTref via 1k pull-up resistor on 14-pin adapter.
5	TDO	Input	JTAG data output from target CPU. Typically connected to TDO on target CPU.
6	—	NC	This pin is not connected to Flasher Secure.
7	—	NC	This pin is not connected to Flasher Secure.
8	VTref	Input	This is the target reference voltage. It is used to check if the target has power, to create the logic-level reference for the input comparators and to control the output logic levels to the target. It is normally fed from Vdd of the target board and must not have a series resistor.
9	TMS	Output	JTAG mode set input of target CPU. This pin should be pulled up on the target. Typically connected to TMS on target CPU.
10	—	NC	This pin is not connected to Flasher Secure.
11	TDI	Output	JTAG data input of target CPU. It is recommended that this pin is pulled to a defined state on the target board. Typically connected to TDI on target CPU.
13	nRES	I/O	Target CPU reset signal. Typically connected to the RESET pin of the target CPU, which is typically called "nRST", "nRESET" or "RESET".

- All pins marked NC are not connected to Flasher Secure. Any signal can be applied here; Flasher Secure will simply ignore such a signal.
- Pins 2, 12, 14 are GND pins connected to GND in Flasher Secure. They should also be connected to GND in the target system.

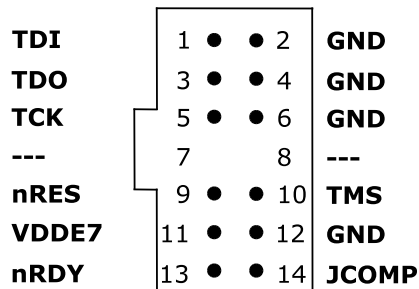
7.2.1 Target power supply

Pin 8 of the 14-pin connector can be used to supply power to the target hardware. Supply voltage is 3.3V / 5V, max. current is 400mA. The output current is monitored and protected against overload and short-circuit. Power can be controlled via the J-Link commander. The following commands are available to control power:

Command	Explanation
<code>power on</code>	Switch target power on
<code>power off</code>	Switch target power off
<code>power on perm</code>	Set target power supply default to "on"
<code>power off perm</code>	Set target power supply default to "off"

7.3 Flasher PPC 14-pin connector

Flasher Secure itself has a 20-pin JTAG connector mounted but an optional 14-pin PPC adapter for PowerPC devices is available.



The following table lists the J-Link PPC 14-pin adapter JTAG pinout.

Pin	Signal	Type	Description
1	TDI	Output	JTAG data input of target CPU. It is recommended that this pin is pulled to a defined state on the target board. Typically connected to TDI on target CPU.
3	TDO	Input	JTAG data output from target CPU. Typically connected to TDO on target CPU.
5	TCK	Output	JTAG clock signal to target CPU. It is recommended that this pin is pulled to a defined state on the target board. Typically connected to TCK on target CPU.
7	—	NC	This pin is not connected to Flasher Secure.
8	—	NC	This pin is not connected to Flasher Secure.
9	nRES	I/O	Target CPU reset signal. Typically connected to the RESET pin of the target CPU, which is typically called "nRST", "nRESET" or "RESET".
10	TMS	Output	JTAG mode set input of target CPU. This pin should be pulled up on the target. Typically connected to TMS on target CPU.
11	VDDE7	Input	This is the target reference voltage. It is used to check if the target has power, to create the logic-level reference for the input comparators and to control the output logic levels to the target. It is normally fed from Vdd of the target board and must not have a series resistor.
13	nRDY	Input	Nexus ready output. Indicates to the development tools that the data is ready to be read from or written to the Nexus read/write access registers.
14	JCOMP	Output	JTAG TAP Controller Enable / JTAG Compliancy (JCOMP). JCOMP is used to enable the TAP controller for communication to the JTAG state machine for boundary scan and for debug access. This pin is set to HIGH by Flasher Secure (in order to enable the JTAG TAP controller on the target device).

- All pins marked NC are not connected to Flasher Secure. Any signal can be applied here; Flasher Secure will simply ignore such a signal.
- Pins 2, 12, 6, 12 are GND pins connected to GND in Flasher Secure. They should also be connected to GND in the target system.

7.4 Target board design

We strongly advise following the recommendations given by the chip manufacturer. These recommendations are normally in line with the recommendations. Please refer to the the appropriate tables depending on the core:

- *Pinout JTAG* on page 115
- *Pinout SWD* on page 116
- *J-Link RX 14-Pin Adapter, J-Link RX FINE 14-Pin Adapter* on page 118
- *J-Link PPC 14-Pin Adapter* on page 120

In case of doubt you should follow the recommendations given by the semiconductor manufacturer.

7.4.1 Pull-up/pull-down resistors

Unless otherwise specified by developer's manual, pull-ups/pull-downs are recommended to be between 2.2 kOhms and 47 kOhms.

7.4.2 RESET, nTRST

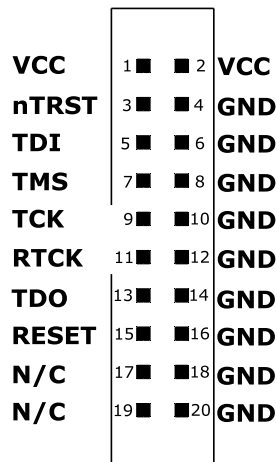
The debug logic is reset independently from the CPU core with nTRST. For the core to operate correctly it is essential that both signals are asserted after power-up.

The advantage of having separate connection to the two reset signals is that it allows the developer performing software debug to setup breakpoints, which are retained by the debug logic even when the core is reset. (For example, at the reset vector address, to allow the code to be single-stepped as soon as it comes out of reset). This can be particularly useful when first trying to bring up a board with a new ASIC.

7.5 Adapters

7.5.1 JTAG Isolator

The JTAG Isolator can be connected between Flasher and JTAG adapter, to provide electrical isolation. This is essential when the development tools are not connected to the same ground as the application. For more information about the JTAG Isolator, please refer to *J-Link JTAG Isolator User Manual (UM08010)* which can be downloaded from our website.



7.5.1.1 Pinout

The following table shows the target-side pinout of the J-Link JTAG Isolator.

Pin	Signal	Type	Description
1	VCC	Output	The target side of the isolator draws power over this pin.
2	VCC	Output	The target side of the isolator draws power over this pin.
3	nTRST	Output	JTAG Reset. Output from Flasher to the Reset signal of the target JTAG port. Typically connected to nTRST of the target CPU. This pin is normally pulled HIGH on the target to avoid unintentional resets when there is no connection.
5	TDI	Output	JTAG data input of target CPU. It is recommended that this pin is pulled to a defined state on the target board. Typically connected to TDI of target CPU.
7	TMS	Output	JTAG mode set input of target CPU. This pin should be pulled up on the target. Typically connected to TMS of target CPU.
9	TCK	Output	JTAG clock signal to target CPU. It is recommended that this pin is pulled to a defined state of the target board. Typically connected to TCK of target CPU.
11	RTCK	Input	Return test clock signal from the target. Some targets must synchronize the JTAG inputs to internal clocks. To assist in meeting this requirement, you can use a returned, and re-timed, TCK to dynamically control the TCK rate.
13	TDO	Input	JTAG data output from target CPU. Typically connected to TDO of target CPU.
15	RESET	I/O	Target CPU reset signal. Typically connected to the RESET pin of the target CPU, which is typically called "nRST", "nRESET" or "RESET".
17	N/C	N/C	This pin is not connected on the target side of the isolator.
19	N/C	N/C	This pin is not connected on the target side of the isolator.

Pins 4, 6, 8, 10, 12, 14, 16, 18, 20 are connected to GND.

7.5.2 J-Link Needle Adapter



Why to choose the J-Link Needle Adapter:

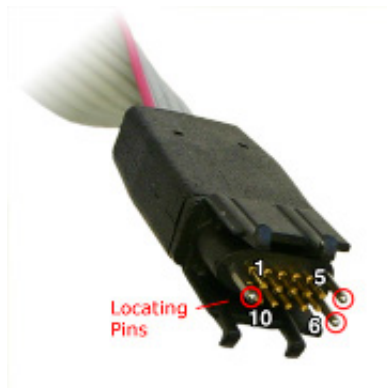
1. No additional connector required on your PCB
2. Very small footprint
3. High reliability spring pins for secure connections
4. Designed with 3 locating pins, so the adapter can not be connected the wrong way
5. No external power supply required! The J-Link Needle Adapter comes with the option to power the target hardware via J-Link.

These features make the J-Link Needle Adapter the perfect solution for production purposes.

The pinout of the J-Link Needle Adapter is based on the pinout of the needle adapter by Tag-Connect. Please note, that both pinouts are not identical since the J-Link Needle Adapter comes with a 5V-supply pin.

As you can see on the image below, the three locating pins ensure, that the adapter cannot be connected to the PCB the wrong way.

Moreover, the two "legs" on each side of the connector guarantee a stable and secure contact between pins and the PCB.



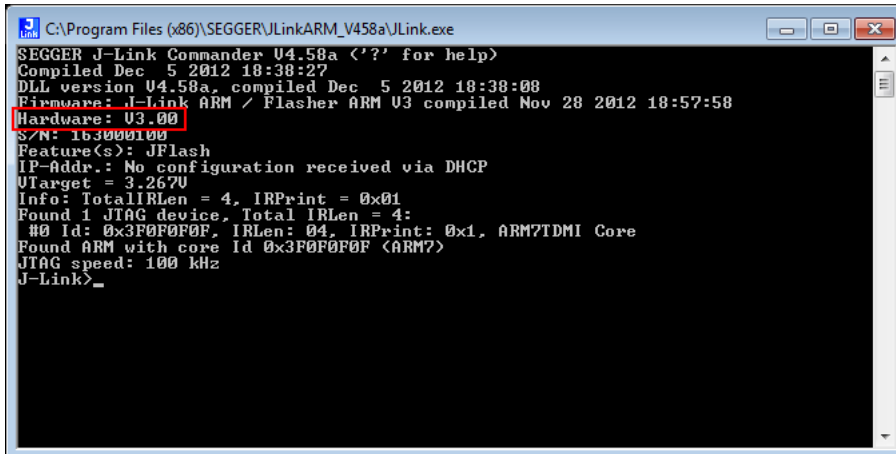
V_{Tref}	1 ● ● 10	nRESET
SWDIO/TMS	2 ● ● 9	TRST
GND	3 ● ● 8	TDI
SWCLK/TCK	4 ● ● 7	RTCK
5V-Supply	5 ● ● 6	SWO/TDO

The J-Link Needle Adapter can be connected to Flasher Secure via the *20-pin 0.1" JTAG to a 10-pin needle connector*.

7.6 How to determine the hardware version

To determine the hardware version of your Flasher, the first step should be to look at the label at the bottom side of the unit. Flasher has the hardware version printed on the back label.

If this is not the case with your Flasher, you can use `JLink.exe` to determine your hardware version (if Flasher is in J-Link mode). As part of the initial message, the hardware version is displayed. For more information about how to ensure that Flasher is in J-Link mode, please refer to *J-Link mode* on page 93.



```
C:\Program Files (x86)\SEGGER\JLinkARM_V458a\JLink.exe
SEGGER J-Link Commander V4.58a <'?' for help>
Compiled Dec  5 2012 18:38:27
DLL version V4.58a, compiled Dec  5 2012 18:38:08
Firmware: J-Link ARM / Flasher ARM V3 compiled Nov 28 2012 18:57:58
Hardware: V3.00
S/N: 163000100
Feature(s): JFlash
IP-Addr.: No configuration received via DHCP
Utarget = 3.2670
Info: TotalIRLen = 4, IRPrint = 0x01
Found 1 JTAG device, Total IRLen = 4:
#0 Id: 0x3F0F0F0F, IRLen: 04, IRPrint: 0x1, ARM7TDMI Core
Found ARM with core Id 0x3F0F0F0F <ARM7>
JTAG speed: 100 kHz
J-Link>_
```

Chapter 8

Support

This chapter contains troubleshooting tips together with solutions for common problems which might occur when using Flasher Secure. There are several steps you can take before contacting support. Performing these steps can solve many problems and often eliminates the need for assistance. This chapter also contains a collection of frequently asked questions (FAQs) with answers.

8.1 Contacting support

Before contacting support, make sure you tried to solve your problem by trying your Flasher Secure with another PC and if possible with another target system to see if it works there. If the device functions correctly, the USB setup on the original machine or your target hardware is the source of the problem, not Flasher Secure.

If you need to contact support, send the following information to support_flasher@segger.com

- A detailed description of the problem
- the serial number
- Information about your target hardware (processor, board, etc.).
- `FLASHER.CFG`, `FLASHER.DAT`, `FLASHER.LOG`, `SERIAL.TXT` file from the Flasher. To get these files, Flasher has to be in MSD mode. For more information about how to boot the unit in MSD mode, please refer to *MSD mode* on page 86. In case of a Flasher Secure some files need to be shared from the Flasher Secure Server.

Flasher Secure is sold directly by SEGGER.

Chapter 9

Glossary

This chapter describes important terms used throughout this manual.

Big-endian

Memory organization where the least significant byte of a word is at a higher address than the most significant byte. See Little-endian.

Cache cleaning

The process of writing dirty data in a cache to main memory.

CM

Contract Manufacturer: A manufacturer that contracts with the customer for components or products.

Coprocessor

An additional processor that is used for certain operations, for example, for floating-point math calculations, signal processing, or memory management.

Dirty data

When referring to a processor data cache, data that has been written to the cache but has not been written to main memory is referred to as dirty data. Only write-back caches can have dirty data because a write-through cache writes data to the cache and to main memory simultaneously. See also cache cleaning.

Halfword

A 16-bit unit of information.

Host

A computer which provides data and other services to another computer. Especially, a computer providing debugging services to a target being debugged.

ICache

Instruction cache.

ID

Identifier.

IEEE 1149.1

The IEEE Standard which defines TAP. Commonly (but incorrectly) referred to as JTAG.

Image

An executable file that has been loaded onto a processor for execution.

Instruction Register

When referring to a TAP controller, a register that controls the operation of the TAP.

IR

See Instruction Register.

JTAG

Joint Test Action Group: The name of the standards group which created the IEEE 1149.1 specification.

Little-endian

Memory organization where the least significant byte of a word is at a lower address than the most significant byte. See also Big-endian.

Memory coherency

A memory is coherent if the value read by a data read or instruction fetch is the value that was most recently written to that location. Obtaining memory coherency is difficult when there are multiple possible physical locations that are involved, such as a system that has main memory, a write buffer, and a cache.

MMU

Memory management unit: Hardware that controls caches and access permissions to blocks of memory, and translates virtual to physical addresses.

MPU

Memory protection unit: Hardware that controls access permissions to blocks of memory. Unlike an MMU, a MPU does not translate virtual addresses to physical addresses.

nTRST

Abbreviation of TAP Reset. The electronic signal that causes the target system TAP controller to be reset. This signal is known as nICERST in some other manuals. See also nSRST.

Open collector

A signal that may be actively driven LOW by one or more drivers, and is otherwise passively pulled HIGH. Also known as a “wired AND” signal.

Processor Core

The part of a microprocessor that reads instructions from memory and executes them, including the instruction fetch unit, arithmetic and logic unit, and the register bank. It excludes optional coprocessors, caches, and the memory management unit.

Remapping

Changing the address of physical memory or devices after the application has started executing. This is typically done to make RAM replace ROM once the initialization has been done.

RESET

Abbreviation of System Reset. The electronic signal which causes the target system other than the TAP controller to be reset. This signal is also known as “nSRST” “nSYSRST”, “nRST”, or “nRESET” in some other manuals. See also nTRST.

RTOS

Real Time Operating System.

TAP

Test Access Port: The port used to access a device’s TAP Controller. Comprises TCK, TMS, TDI, TDO, and nTRST (optional).

TAP Controller

Logic on a device which allows access to some or all of that device for test purposes. The circuit functionality is defined in IEEE1149.1.

Target

The actual processor (real silicon or simulated) on which the application program is running.

TCK

The electronic clock signal which times data on the TAP data lines TMS, TDI, and TDO.

TDI

The electronic signal input to a TAP controller from the data source (upstream). Usually, this is seen connecting the J-Link Interface Unit to the first TAP controller.

TDO

The electronic signal output from a TAP controller to the data sink (downstream). Usually, this is seen connecting the last TAP controller to the J-Link Interface Unit.

TTL

Transistor-transistor logic: A type of logic design in which two bipolar transistors drive the logic output to one or zero. LSI and VLSI logic often used TTL with HIGH logic level approaching +5V and LOW approaching 0V.

UID

Unique Identifier: A number (typically about 128bit) chosen by the chip vendor during production of the chip. This number is unique per device and cannot be changed.

Word

A 32-bit unit of information. Contents are taken as being an unsigned integer unless otherwise stated.

Chapter 10

Literature and references

This chapter lists documents, which we think may be useful to gain a deeper understanding of technical details.

Reference	Title	Comments
[J-Link]	J-Link / J-Trace User Guide	This document describes J-Link and J-Trace. It is publicly available from SEGGER https://www.segger.com/downloads/jlink/UM08001 .
[J-Flash]	J-Flash User Guide	This document describes J-Flash. It is publicly available from SEGGER https://www.segger.com/downloads/jlink/UM08003 .