

STM32CubeProgrammer software description

Introduction

STM32CubeProgrammer (STM32CubeProg) provides an all-in-one software tool to program STM32 devices in any environment: multi-OS, graphical user interface or command line interface, and supports a wide choice of connections (JTAG, SWD, USB, UART, SPI, CAN, I2C), with manual operation or automation through scripting.

This document details the hardware and software environment prerequisites, as well as the available STM32CubeProgrammer software features.





UM2237 Rev 21

1/162

Contents

1	Gettir	ng starte	ed
	1.1	System	requirements
	1.2	Installing	g STM32CubeProgrammer11
		1.2.1	Linux install
		1.2.2	Windows install
		1.2.3	macOS install
		1.2.4	DFU driver
		1.2.5	ST-LINK driver
	1.3	Updater	· 14
		1.3.1	Update steps
		1.3.2	Proxy settings
		1.3.3	Check for updates
	_	_	
2	STM3	2CubeF	Programmer user interface for MCUs
	2.1	Main wi	ndow
		2.1.1	Main menu
		2.1.2	Log panel
		2.1.3	Progress bar
		2.1.4	Target configuration panel 19
	2.2	Memory	v & file edition
		2.2.1	Reading and displaying target memory
		2.2.2	Reading and displaying a file
	2.3	Memory	programming and erasing
		2.3.1	Internal flash memory programming
		2.3.2	External flash memory programming
		2.3.3	Developing customized loaders for external memory
	2.4	Option b	bytes
		2.4.1	MCU unlock (specific for the STM32WL series)
		2.4.2	Debug authentication default configuration (STM32H573/STM32H563/STM32H562 only)
		2.4.3	Debug authentication configuration (STM32H503 only)37
	2.5	Automa	tic mode
	2.6	In applic	cation programming (IAP/USBx)



2.7	Flash th	ne co-processor binary using graphical interface	. 43
	2.7.1	FUS/Stack upgrade	43
	2.7.2	Key provisioning	45
2.8	Serial w	vire viewer (SWV)	47
2.9	Secure	programming interface	. 48
	2.9.1	Introduction	48
	2.9.2	RDP regression with password (STM32U5 series only)	49
	2.9.3	SFI/SFIx	51
	2.9.4	SSP	54
	2.9.5	OBKey provisioning (STM32H573/STM32H563/STM32H562 only) .	56
	2.9.6	Debug authentication (STM32H5 series only)	57
2.10	STM32	CubeProgrammer Script Manager platform for MCUs	. 58
	2.10.1	Introduction for the usage scenarios of Script Manager	58
	2.10.2	Script Manager usage	59
	2.10.3	Loops and conditional statements	64
2.11	DFU IA	P/USBx with custom PID and VID	. 66
2.12	SigFox	™ credentials	. 67
2.13	Registe	r Viewer	. 68
2.14	Hard Fa	ault analyzer	. 69
	2.14.1	Description	69
	2.14.2	Example	71
	2.14.3	Fault analyzer note	71
	2.14.4	Secure Fault analyzer for Cortex-M33	72
2.15	Fill men	nory command	. 73
2.16	Fill men	nory operation	. 75
2.17	Blank c	heck command	. 76
2.18	Blank c	heck operation	77
2.19	Compa	re flash memory with file	. 81
2.20	Compa	rison between two files	. 86
2.21	LiveUpo	date feature	. 91
2 22	Calcula	tor	92
<i>L.LL</i>	Culcula		
STM3	2CubeF	Programmer command line interface (CLI) for MCUs	. 94
3.1	Comma	and line usage	94
3.2	Generic	commands	. 96



3

	3.2.1	Connect command	96
	3.2.2	Erase command	. 104
	3.2.3	Download command	. 104
	3.2.4	Download 32-bit data command	. 105
	3.2.5	Download 64-bit data command	. 105
	3.2.6	Read command	. 106
	3.2.7	Start command	. 107
	3.2.8	Debug commands	. 107
	3.2.9	List command	. 108
	3.2.10	QuietMode command	. 109
	3.2.11	Verbosity command	. 110
	3.2.12	Log command	. 111
	3.2.13	External loader command	. 112
	3.2.14	External loader command with bootloader interface	. 112
	3.2.15	Read unprotect command	. 113
	3.2.16	TZ regression command	. 113
	3.2.17	Option bytes command	. 113
	3.2.18	Safety lib command	. 114
	3.2.19	Secure programming SFI specific commands	. 117
	3.2.20	Secure programming SFIx specific commands	. 117
	3.2.21	HSM related commands	. 119
	3.2.22	STM32WB specific commands	. 120
	3.2.23	Serial wire viewer (SWV) command	. 122
	3.2.24	Specific commands for STM32WL	. 124
	3.2.25	SigFox credential commands	. 127
	3.2.26	Register viewer	. 129
	3.2.27	Hard fault analyzer	. 130
	3.2.28	RDP regression with password	. 131
	3.2.29	GetCertif command	. 132
	3.2.30	Write DBG MCU authentication command	. 132
	3.2.31	OBKey provisioning (STM32H573/STM32H563/STM32H562 only) .	. 133
	3.2.32	Password provisioning (STM32H503 only)	. 133
	3.2.33	Debug authentication commands (STM32H5 series only)	. 134
	3.2.34	Force no debug authentication command	. 136
STM	32Cubel	Programmer user interface for MPUs	137
4.1	Main w	indow	. 137



4

	4.2	Progran	nming windows	38
	4.3	OTP pro	ogramming window	38
		4.3.1	Get OTP structure information 1	39
		4.3.2	Read and display words 1	40
		4.3.3	Edit and fuse words 1	40
		4.3.4	Lock specific/all words 1	41
		4.3.5	Program binary file 1	42
		4.3.6	Save OTP partition	42
5	STM 3	2CubeF	Programmer CLI for MPUs1	44
	5.1	Availabl	e commands for STM32MP1	44
		5.1.1	Connect command 1	44
		5.1.2	GetPhase command 1	45
		5.1.3	Download command1	45
		5.1.4	Flashing service	46
		5.1.5	Start command 1	46
		5.1.6	Read partition command 1	47
		5.1.7	List command	47
		5.1.8	QuietMode command1	48
		5.1.9	Verbosity command1	48
		5.1.10	Log command	48
		5.1.11	OTP programming 1	49
		5.1.12	Programming OTP commands 1	50
		5.1.13	Detach command 1	54
		5.1.14	GetCertif command 1	54
		5.1.15	Write blob command 1	55
	5.2	Secure	programming SSP specific commands	55
6	STM3	2CubeF	Programmer C++ API1	57
7	Revis	ion hist	ory	58



List of tables

Table 1.	Operations supported by Script Manager	59
Table 2.	Document revision history	. 158



List of figures

Figure 1.	Deleting the old driver software	. 13
Figure 2.	STM32 DFU device with DfuSe driver	. 13
Figure 3.	STM32 DFU device with STM32CubeProgrammer driver	. 13
Figure 4.	Proxy settings submenu	. 14
Figure 5.	Proxy settings window	. 15
Figure 6.	Successful connection check	15
Figure 7.	Check for updates submenu	16
Figure 8	Hyperlink button of new version available	16
Figure 9	STM32CubeProgrammer main window	17
Figure 10	Expanded main menu	18
Figure 11	ST-LINK configuration panel	20
Figure 12		22
Figure 13		22
Figure 14		20
Figure 14.		24
Figure 15.		20
Figure 10.		20
Figure 17.		21
Figure 10.		20
Figure 19.		28
Figure 20.		29
Figure 21.		30
Figure 22.	Flash memory programming (external memory).	32
Figure 23.	Flash memory erasing (external memory)	. 32
Figure 24.	Option bytes panel	35
Figure 25.	Unlock chip button	36
Figure 26.	DA default configuration when switching product state to provisioning	37
Figure 27.	Configuration when switching product state to values different from 0x17	37
Figure 28.	Switching product state to provisioning	38
Figure 29.	Switching product state to values different from 0x17	38
Figure 30.	Automatic mode in Erasing & Programming window	. 39
Figure 31.	Automatic mode log traces	40
Figure 32.	Algorithm.	42
Figure 33.	STM32CubeProgrammer in IAP mode	43
Figure 34.	STM32CubeProgrammer API SWD connection	44
Figure 35.	Steps for firmware upgrade	. 44
Figure 36.	Pop-up confirming successful firmware delete	. 45
Figure 37.	Pop-up confirming successful firmware upgrade	. 45
Figure 38.	Download STM32WB patches	45
Figure 39.	Update authentication key	46
Figure 40.	Pop-up requesting to lock authentication key	46
Figure 41.	Store customer key.	47
Figure 42	SWV window.	. 48
Figure 43	RDP regression with password tab.	49
Figure 44	RDP regression with password from level 1	50
Figure 45	Set OFM1 key (LOCK RDP1)	50
Figure 46	Unlock RDP1	50
Figure 47	Disable password	51
Figure 48	Get authentication ID	51
. iguic +0.		



Figure 49.	SFI/SFIx tab	52
Figure 50.	Steps for SFI programming.	52
Figure 51.	SFI parsed info	53
Figure 52.	Open TPC from STM32CubeProgrammer	53
Figure 53.	Display external loader name	53
Figure 54.	HSM-related info panel	54
Figure 55.	SSP PRG user interface	55
Figure 56.	OBKey provisioning	57
Figure 57.	Debug authentication with password	58
Figure 58.	Debug authentication with certificate	58
Figure 59.	Output of Script Manager example 1	62
Figure 60.	Output of Script Manager example 2	63
Figure 61.	Connect via USB DFU panel	66
Figure 62.	Main window after the connection	67
Figure 63.	SigFox credentials	68
Figure 64.	Register Viewer window	68
Figure 65.	Fault Analyzer window	70
Figure 66.	Fault analyzer GUI view when Hard Fault is detected	71
Figure 67.	CCR bits	72
Figure 68.	Example 1	74
Figure 69.	Example 2	74
Figure 70.	Sub-menu displayed from "Read" combo-box	75
Figure 71.	Sub-menu displayed with right click on "Device memory" tab	75
Figure 72.	Sub-menu displayed with right click on the cell of grid	75
Figure 73.	Parameters initialization	76
Figure 74.	Example 1: memory is not blank at address 0x08000014	76
Figure 75.	Example 1: memory is blank	77
Figure 76.	Sub-menu displayed from "Read" combo-box	77
Figure 77.	Sub-menu displayed with right click on "Device memory" tab	78
Figure 78.	Sub-menu displayed with right click on the cell of grid	78
Figure 79.	First address with data	79
Figure 80.	Example 1: memory is blank	80
Figure 81.	Example 2: memory is not blank	81
Figure 82.	Sub-menu displayed from "Read" combo-box	81
Figure 83.	Sub-menu displayed with right click on "Device memory" tab	82
Figure 84.	Sub-menu displayed with right click on the cell of grid	82
Figure 85.	Sub-menu displayed with add tab button	82
Figure 86.	Sub-menu displayed with right click on the opened file tab	83
Figure 87.	Sub-menu displayed from "Download" combo-box displayed in file tab	83
Figure 88.	Data width: 32 bits	84
Figure 89.	Data width: 16 bits	84
Figure 90.	Data width: 8 bits	84
Figure 91.	Data width: 32 bits	85
Figure 92.	Data width: 16 bits	85
Figure 93.	Data width: 8 bits	85
Figure 94.	Before editing the flash memory	86
Figure 95.	After editing the flash memory	86
Figure 96.	Multiple comparisons	86
Figure 97.	Sub-menu displayed from "Read" combo-box in device memory tab	87
Figure 98.	Sub-menu displayed with right click on "Device memory" tab	87
Figure 99.	Sub-menu displayed with right click on the cell of grid	87
Figure 100.	Sub-menu displayed with add tab button	88



Figure 101.	Sub-menu displayed with right click on the opened file tab	. 88
Figure 102.	Sub-menu displayed from "Download" combo-box displayed in file tab	. 88
Figure 103.	Data width: 32 bits	. 89
Figure 104.	Data width: 16 bits	. 89
Figure 105.	Data width: 8 bits	. 90
Figure 106.	Multiple comparisons	. 90
Figure 107.	Live update of data	. 91
Figure 108.	Calculator window	. 92
Figure 109.	STM32CubeProgrammer: available commands	. 95
Figure 110.	Connect operation using RS232	. 98
Figure 111.	Enabling COM DTR pin	. 99
Figure 112.	Connect operation using USB	100
Figure 113.	Connect operation using USB DFU options	101
Figure 114.	Connect operation using SWD debug port	101
Figure 115.	Connect operation using SPI port	102
Figure 116.	Connect operation using CAN port	103
Figure 117.	Connect operation using I2C port	103
Figure 118.	Download operation	105
Figure 119.	Read 32-bit operation	107
Figure 120.	List of available serial ports	109
Figure 121.	Verbosity command	110
Figure 122.	Log command	111
Figure 123.	Log file content	111
Figure 124.	Safety lib command	114
Figure 125.	Flash memory mapping	115
Figure 126.	Flash memory mapping example	116
Figure 127.	SWV command	123
Figure 128.	startswv command	124
Figure 129.	Output of unlockchip command	125
Figure 130.	Disable security	126
Figure 131.	Configure option bytes to their default values	127
Figure 132.	Example of -ssigfoxc command	128
Figure 133.	Example 1 of -wsigfoxc command	128
Figure 134.	Example 2 of -wsigfoxc command	129
Figure 135.	Read core and MCU registers	130
Figure 136.	Fault analyzer CLI view when Hard Fault is detected.	131
Figure 137.	OBKey provisioning example	133
Figure 138.	Discovery log	134
Figure 139.	Debug authentication with password	135
Figure 140.	Debug authentication with certificate	136
Figure 141.	STM32CubeProgrammer main window	137
Figure 142.	TSV programming window	138
Figure 143.	OTP MPU window	139
Figure 144.	Edit denial for locked words	140
Figure 145.	Program Apply confirmation	140
Figure 146.	Lock all words confirmation	141
Figure 147.	All OTP fields are locked	141
Figure 148.	Download binary file	142
Figure 149.	Save OTP partition	142
Figure 150.	Connect operation using RS232	145
Figure 151.	Download operation	146
Figure 152.	TSV file format	146



Figure 153.	Log file content	149
Figure 154.	OTP write command for OTP structure v2	152
Figure 155.	OTP write command for OTP structure v2	153
Figure 156.	Get certificate output file	155
Figure 157.	SSP successfully installed	156



1 Getting started

This section describes the requirements and procedures to install the STM32CubeProgrammer software tool, which supports STM32 32-bit MCUs, based on Arm^{®(a)} Cortex[®]-M processors, and STM32 32-bit MPUs, based on Arm[®] Cortex[®]-A processors.

1.1 System requirements

Supported operating systems and architectures:

- Linux[®] 64-bit
- Windows[®] 7/8/10/11 32-bit and 64-bit
- macOS[®] (minimum version OS X[®] Yosemite)

There is no need to install any Java[™] SE Run Time Environment since version 2.6.0. The STM32CubeProgrammer runs with a bundled JRE available within the downloaded package, and no longer with the one installed on your machine.

Note: The bundled JRE is Liberica 8.0.265.

For macOS software the minimum requirements are:

- Xcode[®] must be installed on macOS computers
- both Xcode[®] and Rosetta[®] must be installed on macOS computers embedding Apple[®] M1 processor

The minimal supported screen resolution is 1024x768.

1.2 Installing STM32CubeProgrammer

This section describes the requirements and the procedure for the software usage. The setup offers also the optional installation of the "STM32 Trusted Package Creator" tool, used to create secure firmware files for secure firmware install and update. For more information, refer to UM2238 "STM32 Trusted Package Creator tool software description", available on *www.st.com*.

1.2.1 Linux install

If you are using a USB port to connect to the STM32 device, install the libusb1.0 package by typing the following command:

sudo apt-get install libusb-1.0.0-dev

To use ST-LINK probe or USB DFU to connect to a target, copy the rules files located under *Driver/rules* folder in */etc/udev/rules.d/* on Ubuntu (*"sudo cp *.* /etc/udev/rules.d"*).

Note: libusb1.0.12 version or higher is required to run STM32CubeProgrammer.



a. Arm is a registered trademark of Arm Limited (or its subsidiaries) in the US and/or elsewhere.



To install the STM32CubeProgrammer tool, download and extract the zip package on your Linux machine from STM32CubeProg-Linux part number on the website, and execute *SetupSTM32CubeProgrammer-vx.y.z.linux*, which guides you through the installation process. In Ubuntu 20 STM32CubeProgrammer, icon is not enabled by default. To enable it, right click on the icon and choose *"Allow launching"*.

1.2.2 Windows install

To install the STM32CubeProgrammer tool, download and extract the zip package from STM32CubeProg-Win-32bits or STM32CubeProg-Win-64bits for, respectively, Windows 32 bits and Windows 64 bits, and execute *SetupSTM32CubeProgrammer-vx.y.z.exe*, which guides you through the installation process.

1.2.3 macOS install

To install the STM32CubeProgrammer tool, download and extract the zip package from STM32CubeProg-Mac part number on the website and execute *SetupSTM32CubeProgrammer-vx.y.z.app*, which guides you through the installation process.

Note: If the installation fails, launch it in CLI mode using the command ./SetupSTM32CubeProgrammerx.y.z.app/Contents/MacOs/SetupSTM32CubeProgrammer-x_y_z_macos.

Make sure you have administrator rights, then double-click *SetupSTM32CubeProgrammer-macos* application file to launch the installation wizard.

In case of error, try this fix:

- 1. Right-Click on SetupSTM32CubeProgrammer-2.12.0
- 2. Select "Show Package Contents"
- 3. Navigate to Contents/MacOs
- 4. Launch SetupSTM32CubeProgrammer-2_12_0_macos

1.2.4 DFU driver

If you are using the STM32 device in USB DFU mode, install the STM32CubeProgrammer DFU driver by running the *"STM32 Bootloader.bat"* file. This driver is provided with the release package, and can be found in the DFU driver folder.

If the DFUSE driver is installed on your machine, first uninstall it, then reboot the machine and run the previously mentioned ".bat" file. Check the "Delete the driver software for this device option" to avoid reinstalling the old driver when, later, a board is plugged in.



	0 0	
Í	Confirm Device Uninstall	
	STM Device in DFU Mode	
	Warning: You are about to uninstall this device from your system.	
	Delete the driver software for this device.	
	OK Cancel	

Figure 1. Deleting the old driver software

Figure 2. STM32 DFU device with DfuSe driver



Figure 3. STM32 DFU device with STM32CubeProgrammer driver



Note:

When using USB DFU interface or ST-LINK interface on a Windows 7 PC, ensure that all the drivers of the USB 3.0 controller drivers are updated. Older versions of the drivers may have bugs that prevent access or cause connection problems with USB devices.



1.2.5 ST-LINK driver

To connect to an STM32 device through a debug interface using ST-LINK/V2, ST-LINKV2-1, ST-LINK-V3, or ST-LINKV3Power, install the ST-LINK driver by running the *"stlink_winusb_install.bat"* file. This driver is provided with the release package, it can be found under the *"Driver/stsw-link009_v3"* folder.

1.3 Updater

STM32CubeProgrammer updater allows users to make automatic updates of the software and its associated packages. The updater is available in all supported operating systems, namely Windows 10/11, Linux, and macOS.

1.3.1 Update steps

- 1. Check the connection and update its settings if needed.
- 2. Check for updates.
- 3. Download the new version.
- 4. Install the downloaded version (the tool restarts once updated).

1.3.2 Proxy settings

The user can manually check the connection by using the "Proxy Settings" window opened with the submenu available in the help button (see *Figure 4*). Three settings are available (see *Figure 5*):

- No proxy
- Use the system parameters
- Use manual configuration of server: add the HTTP proxy name, port, and credentials

Figure 4. Proxy settings submenu





rigure o. rioxy settings window	
Pro Proxy settings —	×
Connection parameters	
Proxy Server Type	
 No Proxy Use System Proxy Parameters Manual Configuration of Proxy Server 	
Manual Configuration of Proxy Server	
HTTP proxy Ips5.gnb.st.com Port 80	
Authentication	
User Login Password	
× Check Connection OK Cance	el

Figure 5. Proxy settings window

The status of connection check is displayed in the "check Connection" button:

- A green icon indicates success (left side of *Figure 6*).
- A red icon indicates that the connection is down (right side of *Figure 6*).

Figure 6. Successful connection check

Proxy settings — X
Connection parameters
Proxy Server Type
No Proxy Use System Proxy Parameters Manual Configuration of Proxy Server
Manual Configuration of Proxy Server
HTTP proxy [lps5.gnb.st.com Port 80
Authentication
User Login Password



1.3.3 **Check for updates**

User can launch the process of update using the Updater window opened with the submenu "Software update" added in the help button.





If there is a new version available, an update button appears in the main menu (Figure 8).

Figure 8. Hyperlink button of new version available								
STM32CubeProgrammer		24	- 0	, c				
STM32	🥺 f 🛛	y	\star	5				
Memory & File edition			Not	connect				

.....

Note:

If the user has already updated the STM32CubeProgrammer, the hyperlink button is no longer displayed at startup.

If a new version is available, the user can make updates through the updater window.

This window displays:

- The current version of the STM32CubeProgrammer .
- The available version in server of STM32CubeProgrammer
- Change log (contains the main changes delivered in the new package)
- License
- Last update (contains the date of the last update, or the message "No previous updates are done")
- The current version of the updater tool
- Refresh button (used to check if there is a new version)
- Close button (used to stop the installation of the new version)

Administrator rights are required to download the new package. Once the update is done, Note: the updater window displays only the new version.



2 STM32CubeProgrammer user interface for MCUs

2.1 Main window

Prg STM	32CubeProgra	immer									- 🗆 X
STM32 Cube	Programmer	r							(19)	f 🖻	y 🔀 🟹
Ξ	Memory	& File ec	liting								Connected
	Device me	emory O	pen file +							ST-LINK	Disconnect
	Address	0x080000	00 💌 Size	0x400	Data width	32-bit 🔻	Find Data	0x	Read 💌	Serial numb	ST-LINK configuration
	Add	ress	0	4	8	С		ASCI	1	Port	
OB	0x080000	000	F216AFB4	226ACC9E	D6665573	035EEBBG	: '~.	ò.1j"sUfÖ¼ë∧.			SWD
	0x080000	010	127421F0	77EB3A25	EB5C49D7	305A8BF	5 ð!t	.%:ëw×I\ëö.ZO		Frequency (4000 🔻
CPU	0x080000	020	F92023EE	BB6EFBAC	100D14C6	01D352D	î#	ù¬ûn≫ÆÞRÓ.		Mode	Normal
	0x080000	030	E3F134C4	977E2E6B	573B5B75	6420E46	7 Ä4ñ	ãk.~.u[;Wgä d		Access port	
SWV	0x080000	040	63B8AFFC	D5AFF944	778FA201	E8606C2	3 ü ⁻ ,	cDù [¯] Õ.⊄.w#l`è			
	0x080000	050	73C33C15	FCE6E21F	E2EE6449	28358102	2 .<Å	s.âæüIdîâ5(Reset mode	Software reset 🛛 🔻
	0x080000	060	9DE06F6E	BF9B3BF3	B4A9FA8A	8902F7A	1 noà	.ó;.¿.ú©´¤÷		Speed	Reliable
	0x080000	070	8CE35653	10196C05	26C41ED2	98FDE710	SVã	1Ò.Ä&.çý.		Shared	
REG	0x080000	080	8C71407F	9D9F263D	8CEF2F8C	FBD4538	3 .@q	.=&/ïSÔû			Disabled:
DEIA	0x080000	090	5434D470	84865B54	8956D849	AA048EC	3 p04	TT[IØV.Ȫ		Debug in Lo External loa	w Power mode 🛛 📈
	0x080000	DAO ·	4C65D31B	F47FECE9	AD8A3D70	064727E	3 .Óe	Léì.ôp=ã'G.		/ Target volta	ge 3.25 V Firmware upgrade
	<								>	Firmware ve	rsion v2356Wi27
	Log					Liv	e Update	Verbosity level	1 2 3		
	15:07:05 : 01 15:07:05 : E 15:07:05 : A	PLOADING C Bank : 0 Address :	0x40022040	A							
	15:07:05 : 8 15:07:05 : 4 15:07:05 : 4	Bank : 0 Address :	2 Bytes 0x01 0x40022060							Board Device	Target information NUCLEO-L552ZE-Q STM3215xx
$\textcircled{0}{0}$	15:07:05 : 01 15:07:05 : 01 15:07:05 : 02 15:07:05 : 4	Size : 16 PLOADING Size : 10 Address :	024 Bytes 0x8000000						~	Type Device ID Revision ID Flash size	MCU 0x472 Rev Z 512 KB
?	6								100% 😣	CPU Bootloader	Cortex-M33 Version

Figure 9. STM32CubeProgrammer main window

The main window is composed of the parts described in the following sections.

2.1.1 Main menu

This menu allows the user to switch between the three main panels of the Memory and file editing, Erasing & programming, and Option bytes tools. The other panels are displayed according to the used device. By clicking on the Hamburger menu (the three-line button) on the top left corner, the menu expands and displays the textual description shown in *Figure 10*.





Figure 10. Expanded main menu

2.1.2 Log panel

Displays errors, warnings, and informational events related to the operations executed by the tool. The verbosity of the displayed messages can be refined using the verbosity ratio buttons above the log text zone. The minimum verbosity level is 1, and the maximum is 3 (all transactions via the selected interface are logged). All displayed messages are time stamped with the format "hh:mm:ss:ms", where "hh" is for hours, "mm" for minutes, "ss" for seconds and "ms" for milliseconds (in three digits).

On the right of the log panel there are two buttons, the first to clean the log, the second to save it to a log file.



2.1.3 Progress bar

The progress bar visualizes the progress of any operation or transaction done by the tool (for example, Read, Write, Erase). You can abort ongoing operations by pressing the "Stop" button in front of the progress bar.

2.1.4 Target configuration panel

This is the first panel to look at before connecting to a target. It allows the user to select the target interface (either the debug interface using ST-LINK debug probe, or the bootloader interface over UART, USB, SPI, CAN or I2C).

The refresh button allows you to check the available interfaces connected to the PC. If this button is pressed while the ST-LINK interface is selected, the tool checks the connected ST-LINK probes, and lists them in the Serial numbers combo box. If the UART interface is selected, it checks the available communication ports of the PC, and lists them in the Port combo box. If the USB interface is selected, it checks the USB devices in DFU mode connected to the PC, and lists them in the Port combo box. Each interface has its own settings, to be set before connection.



ST-LINK settings



Figure 11. ST-LINK configuration panel

- **Serial number**: this field contains the serial numbers of all connected ST-LINK probes. The user can choose one of them, based on its serial number.
- **Port**: ST-LINK probe supports two debug protocols, JTAG and SWD.

Note:

- **Frequency**: the JTAG or SWD clock frequency
- Access port: selects the access port to connect to. Most of the STM32 devices have only one access port, which is Access port 0.

JTAG is not available on all embedded ST-LINK in the STM32 Nucleo or Discovery boards.

- Mode:
 - **Normal**: with "Normal" connection mode, the target is reset then halted. The type of reset is selected using the "Reset Mode" option.
 - Connect under reset: this mode enables connection to the target using a reset vector catch before executing any instructions. This is useful in many cases, for example when the target contains a code that disables the JTAG/SWD pins.
 - Hot plug: enables connection to the target without a halt or reset. This is useful for updating the RAM addresses or the IP registers while the application is running.
 - **Power down**: used to put the target in debug mode, even if the application has not



started since the target power-up. The hardware reset signal must be connected between ST-LINK and the target. This feature can be not fully effective on some boards (MB1360, MB1319, MB1361, MB1355) with STMPS2141 power switch.

- Reset mode:
 - Software system reset: resets all STM32 components except the Debug via the Cortex-M application interrupt and reset control register (AIRCR).
 - Hardware reset: resets the STM32 device via the nRST pin. The RESET pin of the JTAG connector (pin 15) must be connected to the device reset pin.
 - **Core reset**: resets only the Cortex-M via the AIRCR.
- Speed (Cortex-M33 only):
 - **Reliable**: allows the user to connect with a slow mode.
 - **Fast**: allows the user to connect with a fast mode.
- **Shared**: enables shared mode allowing connection of two or more instances of STM32CubeProgrammer or other debugger to the same ST-LINK probe.
- **Debug in low-power mode** (STM32U5/WB/L4 series only): sets the bits in DBGMCU_CR to 1.
- **External loader**: displays the name of the external memory loader selected in the "External loaders" panel accessible from the main menu (Hamburger menu).
- Target voltage: target voltage is measured and displayed.
- **Firmware version**: displays the ST-LINK firmware version. The firmware upgrade button allows you to upgrade the ST-LINK firmware.

UART settings

	🛑 No	ot connected
UART	•	Connect
UAR	T configuratio	on
Port	СОМЗ	- Ø
Baudrate	115200	+
Parity	Even	+
Data bits	8	×
Stop bits	1.0	Ŧ
Flow control	Off	-
RTS	0	Ŧ
DTR	0	-
Read Unprotect (MCL TZEN Regression (MC	ŋ U)	
Tar	get informatio	on
Device Type Device ID Revision ID Flash size CPU Bootloader Versio	n	

Figure 12. UART configuration panel

• **Port**: selects the com port to which the target STM32 is connected. Use the refresh button to recheck the available com port on the PC.

The STM32 must boot in bootloader mode using boot pins and/or the option bits. Check AN2606 "STM32 microcontroller system memory boot mode", available on www.st.com, for more information on the STM32 bootloader.

- **Baudrate**: selects the UART baud rate.
- **Parity**: selects the parity (even, odd, none), must be "even" for all STM32 devices.
- Data bits: must be always 8, only 8-bit data is supported by the STM32.
- **Stop bits**: must be always 1, only 1-bit stop is supported by the STM32.
- Flow control: must be always off.
- **RTS** (request to send): sets the COM RTS pin to either high or low level.
- **DTR** (data terminal ready): sets the COM DTR pin to either high or low level.

Note:



USB settings

	Not connected
USB 🗸	Connect
USB confi	guration
Port No	DFU de 🔻 💋
Serial number	
PID 0xd	f11
VID 0x0	483
Read Unprotect (MCU)	
TZEN REGression (MCU)	formation
Target int Board	formation
Device Type Device ID Revision ID Flash size CPU Bootloader Version	

Figure 13. USB configuration panel

• **Port**: selects the USB devices in DFU mode connected to the PC. You can use the refresh button to recheck the available devices.

Note: The STM32 must boot in bootloader mode using boot pins and/or the option bits. Check AN2606, available on www.st.com, for more information.

Once the correct interface settings are set, click on the "Connect" button to connect to the target interface. If the connection succeeds, it is shown in the indicator above the button, which turns to green.

Once connected, the target information is displayed in the device information section below the settings section, which is then disabled as in *Figure 14*.





Figure 14. Target information panel



SPI settings

	Not connected
ST-LINK	 Connect
ST-LIN	NK configuration
Serial number	00460026 🔻 💋
Port	SPI 👻
Baudrate (kHz)	375 👻
nss	Hard 👻
nsspulse	Pulse 👻
delay	No delay 👻
Direction	Full duplex 👻
External loader _ Target voltage _ Firmware version \	0.03 V V3J9M3B5S1 Firmware upgrade
Targ	et information
Board Device Type Device ID Revision ID Flash size CPU Bootloader Versior	- - - - - - -

Figure 15. SPI configuration panel

- Serial number: this field contains the serial numbers of all connected ST-LINK-V3 probes in case of use of SPI bootloader.
- **Port**: selects the SPI devices connected to the PC. Use the refresh button to recheck the available devices.
- **Baudrate**: selects the SPI baud rate.
- **nss**: slave select software or hardware.
- **nsspulse**: the slave selection signal can operate in a pulse mode, where the master generates pulses on nss output signal between data frames for a duration of one SPI clock period when there is a continuous transfer period.
- Delay: used to insert a delay of several microseconds between data.
- **Direction**: must be always Full-duplex, both data lines are used, and synchronous data flows in both directions.



CAN settings

	Not conne	ected
ST-LINK	▼ Connect	
ST-LIN	NK configuration	
Serial number	00460026 🔻	Ø
Port	CAN	-
Baudrate (kHz)	125	-
Assigned fifo	FIFO 0	-
Filter mode	Mask	-
Filter scale	32 bits	-
Filter bank	0	*
External loader Target voltage Firmware version \	0.03 V V3J9M3B5S1 Firmware u	pgrade
Targ	get information	
Board Device Type Device ID Revision ID Flash size CPU Bootloader Version	n	- - - - -

Figure 16. CAN configuration panel

- **Serial number**: this field contains the serial numbers of all connected ST-LINK-V3 probes in case to use CAN bootloader.
- **Port**: selects the CAN devices connected to the PC. You can use the refresh button to recheck the available devices.
- **Baudrate**: selects the CAN baud rate.
- **Assigned FIFO**: selects the receive FIFO memory to store incoming messages.
- Filter mode: selects the type of the filter, MASK, or LIST.
- Filter scale: selects the width of the filter bank, 16 or 32 bits.
- Filter bank: values between 0 and 13, to choose the filter bank number.



I2C settings



Figure 17. I2C configuration panel

- **Serial number**: this field contains the serial numbers of all connected ST-LINK-V3 probes when using the I2C bootloader.
- **Port**: selects the I2C devices connected to the PC. You can use the refresh button to recheck the available devices.
- **Baudrate**: selects the I2C baud rate.
- Address: adds the address of the slave bootloader in hex format.
- Speed mode: selects the speed mode of the transmission Standard or Fast.
- **Rise time**: chooses values according to Speed mode, 0-1000 (STANDARD), 0-300 (FAST).
- Fall time: chooses values according to Speed mode, 0-300 (STANDARD), 0-400 (FAST).



2.2 Memory & file edition

This panel allows the user to read and display target memory and file contents.

2.2.1 Reading and displaying target memory

Figure 1	8. Device	memory	tab
----------	-----------	--------	-----

M STM	32CubeProgrammer								- a ×
STM:2	çe artistior							🤒 f	🕒 🎽 🛪 🗛
≡	Memory & File editing								Connected
	Device memory Open file +	I							ST-LINK
*	Address 0x08000000 - Size	0x400 Data width	32-bit 💌 Find Data	Dx				Read 💌	ST-LINK configuration
OB	Address	0	4	8	с		ASCII	Save As	VD 👻
	0x08000000	BBBBAAAA	DDDDCCCC	BBBBAAAA	DDDDCCCC	^{aa} »»İİÝÝ ^{aa} »»İİÝÝ		Read all	£000 -
CPU	0x08000010	BBBBAAAA	DDDDCCCC	BBBBAAAA	DDDDCCCC	aa _{>>} tiýýaa _{>>} tiýý		Fill memory	Ctrl+M
	0x08000020	BBBBAAAA	DDDDCCCC	BBBBAAAA	DDDDCCCC	aa _{>>} tiýýaa _{>>} tiýý		Blank check	
swv	0x08000030	BBBBAAAA	DDDDCCCC	BBBBAAAA	DDDDCCCC	aa _{>>} tiýýaa _{>>} tiýý		Compare memory v	vith file
	0x08000040	BBBBAAAA	DDDDCCCC	BBBBAAAA	DDDDCCCC	aa _{>>} ttýýaa _{>>} ttýý		Compare two files	enware reset
$\overline{\mathbb{Q}}$	0x08000050	BBBBAAAA	DDDDCCCC	BBBBAAAA	DDDDCCCC	aa _{>>} IIÝÝaa _{>>} IIÝÝ			Hiable
-	0x08000060	BBBBAAAA	DDDDCCCC	BBBBAAAA	DDDDCCCC	aa _{>>} IIYYaa _{>>} IIYY			Shared Disabled 👻 🚯
	0x08000070	BBBBAAAA	DDDDCCCC	BBBBAAAA	DDDDCCCC	aa»»IIYYaa»»IIYY			Debug in Low Power mode 🛛 🕅 External loader
REG	0x08000080	BBBBAAAA	DDDDCCCC	BBBBAAAA	DDDDCCCC	aa			Target voltage 3.32 V Einnware upgrade
	0x08000090	DDDDAAAA	DDDDCCCC	DDDDAAAA	DDDDCCCC				Terrivale version 1550m303a1
	0x08000000	DDDDAAAA	poppecce	DDDDAAAA	DDDDCCCC				
	0x08000060	DDDDAAAA	DDDDCCCC	DDDDAAAA	DDDDCCCC				
	0x08000000	PPPPAAAA	DDDDCCCC	PPPPAAAA	DDDDCCCC				
	0x08000000	PPPPAAAA	DDDDCCCC	PPPPAAAA	DDDDCCCC	ss1111ss1111			
	0×080000E0	PPPPAAAA	DDDDCCCC	PPPPAAAA	DDDDCCCC	aattooaattoo			
	0x08000100	BBBBAAAA	DDDDCCCC	BEBEAAAA	DDDDCCCC	aa _{nn} ttýýaa _{nn} ttýý			
	0x08000110	BBBBAAAA	DDDDCCCC	REREALAA	DDDDCCCC	aa _{uu} ttýýaa _{uu} ttýý		~	
	100						Live Undete - Markasitu Javal - @	1	
	Log 15/133 / Reset mode : Software reset 15/133 / Dorier D : 0.467 15/133 / Dorier D : 0.467 15/133 / Reset mode : Software reset 15/133 / Reset mode : 10.467 15/133 / Log : 1000 15/133 / Address : 0.6000420 15/133 / Address : 0.6000000 15/133 / Address : 0.6000000 15/133 / Address : 0.6000000 15/133 / Address : 0.60000000 15/133 / Address : 0.6000000 15/133 / Address : 0.6000000 15/133 / Address : 0.6000000 15/133 / Address : 0.6000000 15/133 / Address : 0.60000000 15/133 / Address : 0.60000000 15/133 / Address : 0.60000000 15/133 / Address : 0.600000000 15/133 / Address : 0.60000000000000000000000000000000000	t supported for this device. A					Ure update verbosky even	*	Target information loord STURK-VSHT Device STM22K-MSOCKS DPP-0 K20 DPP-0 Performance Paralise 25648
?								100% 🛞	CPU Cortex-M4 Bootloader Version DxC4

After target connection, the STM32 target memory can be read using this panel. To do this, specify the address and the size of the data to be read, then press the Read button in the top-left corner. Data can be displayed in different formats (8-, 16-, and 32-bit) using the "Data width" combo box.

The user can read all the flash memory using the "Read All" button, save the device memory content in a .bin, .hex, or .srec file using the "Save As..." menu from the tab contextual menu or the action button.

Multiple device memory tabs can be opened to display different locations of the target memory. To do this, click on the "+" tab to display a contextual menu that allows you to add a new "Device memory" tab, or to open a file and display it in a "File" tab.

Figure	19.	Contextual	menu
	-		

	Device memory	Open file	+						
*	Address 0x08000	0000 💌 Si	Open file Open memory tab	idth	32-bit 💌	Find Data	0x	Read 👻	
	Address 0		Compare memory with file	8	C ASCII				
	0x0800000	20000820	compare memory with me	5DD	080015DF		ÝB		
CPU	0x08000010 080015E5		Compare two files	5F1	00000000 åëñ		ñ		
ero	0x08000020 00000000		000000 000000	000 080015F7		F7			

UM2237 Rev 21



2.2.2 Reading and displaying a file

To open and display a file, just click on the "+" and select "Open File" menu, as illustrated in *Figure 19*.

The supported formats are binary files (.bin), ELF files (.elf, .axf, .out), Intel hex files (.hex), and Motorola S-record files (.Srec).

Once the file is opened and parsed, it is displayed in a dedicated tab with its name, as illustrated in *Figure 20*. The file size is displayed in the "Size" field, and the start address of hex, srec, or ELF files, is displayed in the "Address" field (for a binary file it is 0).

STM32CubeProgrammer						6			
M32 CubeProgrammer							🥙 f 🕒) 💆 🛪 🗛	
Memory & File	edition							Not connecte	
Device memory	STM32H743I_EV	'AL.elf × +					ST-LINK	- Connect	
Address 0x800	• 00000	Size 0x400	Data width	n 32-bit	-	Download 👻	ST	-LINK configuration	
Address	0	4	8	с	ASCII	Read	number	52FF7006526654	
0x08000000	20020000	080037A1	08003B5D	08003B5F	;7];;	Save As		SWD	
0x08000010	08003B61	08003B63	08003B65	0000000	a;c;e;	Verify	ency (kH=)	1000	
0x08000020	00000000	00000000	00000000	08003B67	g;	Address 0x800000	ency (kH2)	4000	
0x08000030	08003B69	00000000	08003B6B	08003B6D	i;k;m;	, adress calobood		Normal	
0x08000040	080037F1	080037F1	080037F1	080037F1	ñ7ñ7ñ7		Access port	0	
0x08000050	080037F1	080037F1	080037F1	080037F1	ñ7ñ7ñ7		Reset mode	Hand and a second	
0x08000060	080037F1	080037F1	080037F1	080037F1	ñ7ñ7ñ7			Hardware reset	
0x08000070	080037F1	080037F1	080037F1	080037F1	ñ7ñ7ñ7		Shared	Disabled 👻	
0x08000080	080037F1	080037F1	080037F1	080037F1	ñ7ñ7ñ7ñ7		External loader	M29W128GL_STM32H743I-EV	
0x08000090	080037F1	080037F1	080037F1	080037F1	ñ7ñ7ñ7		Target voltage	2.44 V	
0x080000A0	080037F1	080037F1	080037F1	080037F1	ñ7ñ7ñ7ñ7		Firmware version	V2J32S7	
0x080000B0	080037F1	080037F1	080037F1	080037F1	ñ7ñ7ñ7			Firmware upgrad	
0x080000C0	080037F1	080037F1	080037F1	080037F1	ñ7ñ7ñ7ñ7	~			
Log					Verbosity level	1 2 3			
1/:39:38 : 1hme elapsed during the read operation is: 00:00:00.009 17:39:48 : Read File: C:\binaries\test.bin 17:39:48 : Number of segments: 1 17:39:48 : work of segments: 1 17:39:48 : work of segments: 1 17:39:48 : work of segments: 1 17:39:48 : work of segments: 1 17:39:48 : work of segments: 1 17:39:48 : work of segments: 1 17:39:40 : segment[0]: address= 0x0, size= 0: work of the providet the providet the									
17:40:01 : segn 18:02:42 : Read	ment[0]: addres	s= 0x8000000, si	ze= 0x1080 VAL.elf.hex				Device		
18:02:42 : Numb	per of segments	: 1 s= 0x8003000. si	ze= 0x8C8				Туре		
18:02:57 : Read	File: C:\bina	ries\STM32H743I_	EVAL.elf				Device ID		
18:02:57 : segn	ment[0]: addres	s= 0x8000000, si	ze= 0x43930			U	Flash size		
							СРИ		

Figure 20. File display

The address field can be modified to display the file content starting from an offset. Using the tab contextual menu or the action button, the file can be downloaded using the "Download" button/menu. For a binary file, specify the download address in the "Address" menu. The user can verify if the file is downloaded using the "Verify" menu, and save it in another format (.bin, .hex or .srec).

As for the "Device memory" tab, the user can display the file memory content in different formats (8-, 16-, and 32-bit), using the "Data width" combo box.

2.3 Memory programming and erasing

This panel is dedicated to flash memory programming and erasing operations.



2.3.1 Internal flash memory programming

Pro STM3	2CubeProgrammer						
STM32 Cube	Programmer					🐠 f	🖻 🦻 🛪 🏹
	Erasing & Programming						Connected
	Download	Erase flash m	mory Era	ise external mem	ory	ST-LINK	- Disconnect
.	File path C\Users\\$TM32H7471-EVAL_demo\Binary\STM32Cube_De Browse Browse		Erase se	lected sectors F	ull chip erase	Serial number	ST-LINK configuration 003E004B30375119353
	Start address	Select	Index	Start Address	Size	Port	SWD
ОВ	Skip flash erase before programming		0	0x0800000	128К 🎧	Frequency (kHz)	24000
	Verify programming		1	0x08020000	128K	Mada	24000
	Run after programming		2	0x08040000	128K	Mode	Under reset
	Start Programming		3	0x08060000	128K	Access port	0
	Start Followinning		4	0x08080000	128K	Reset mode	Software reset 🔹
	Automatic Mode		5	0x080A0000	128K	Shared	Disabled 🔹 🕥
	Full chip erase		7	0x080E0000	128K	External loader	MT25TL01G STM32H747I-EVAL
	Download file		8	0x08100000	128K	Target voltage	3.29 V
	Option bytes commands		9	0x08120000	128K	Firmware version	V3J3M2
			10	0x08140000	128K		Firmware upgrade
	Start automatic mode		11	0x08160000	128K		
					>		
	Log		Verbosity	level 💿 1 🤇	2 3		
	108140113 : Device LD : (UKSU) 08:40:13 : UPLOAINCE OPTION BYTES DATA 08:40:13 : Bank : 0x00				^ 4		Device information
(\mathbf{O})	08:40:13 : Size : 308 Bytes 08:40:13 : UPI OADING					Device	STM32H7xx
$\overline{}$	08:40:13 : Size : 1024 Bytes 08:40:13 : Address : 0x8000000					Туре	MCU
	08:40:13 : Read progress: 08:40:13 : Data read successfully					Device ID	0x450
	08:40:13 : Time elapsed during the read operation is: 00:00:00.002				~	Flash size	2 MB
(?)						CPU	Cortex-M7

Figure 21. Flash memory programming and erasing (internal memory)

Memory erasing

Once connected to a target, the memory sectors are displayed in the right-hand panel showing the start address and the size of each sector. To erase one or more sectors, select them in the first column and then click on the "Erase selected sectors" button.

The "Full chip erase" button erases the whole memory.



Memory programming

To program a memory, go through the following steps:

- 1. Click on the browse button and select the file to be programmed. The supported formats are binary files (.bin), ELF files (.elf, .axf, .out), Intel hex files (.hex) and Motorola S-record files (.Srec).
- 2. In case of programming a binary file, the address must be set.
- 3. Select the programming options:
 - Verify after programming: read back the programmed memory and compare it byte per byte with the file.
 - Skip flash memory erase before programming: if checked, the memory is not erased before programming. This option must be checked only when you are sure that the target memory is already erased.
 - Run after programming: start the application just after programming.
- 4. Click on the "Start programming" button to start programming.

The progress bar on the bottom of the window shows the progress of the erase and programming operations.

2.3.2 External flash memory programming

To program an external memory connected to the microcontroller via any of the available interfaces (for example SPI, FMC, FSMC, QSPI, OCTOSPI) you need an external loader.

STM32CubeProgrammer is delivered with external loaders for several STM32 evaluation and discovery boards (refer to the "bin/ExternalLoader" directory). If you need to create a new external loader, see *Section 2.3.3* for more details.

To program an external memory, select one (or more) external loader(s) from the "ExternalLoader" panel, which is (are) used by the tool to read, program, or erase external memories, as shown in *Figure 22*. Once selected, the external loader(s) is (are) used for any memory operation in its (their) memory range.

The "External flash erasing" tab on the right of the "Erasing and Programming" panel displays the memory sectors for each selected loader, and enables sector or full-chip erase, as shown in *Figure 23*.



CubeProgrammer							- 0
grammer							🥸 🖪 🕨 У 🛧 J
ternal loaders							🛑 Not c
vailable external loaders:					Q	Deselect	all ST-LINK • Con
Select	Name	Board	Start Address	Memory Size	Page Size	Туре	ST-LINK configuration
	512W3A_STM3210E-EVAL	STM3210E-EVAL	0x70000000	64M	0x200	NAND_FLASH	No ST-LINK
	IS42S32400F_STM32F469I+DK	STM32F469I+DK	0xC0000000	16M	0x1000000	SRAM	Port SWD
	IS42S32800G_STM32769I-EVAL	STM32769I-EVAL	0xC0000000	32M	0x2000000	SRAM	Frequency (kHz)
	IS61WV102416BLL_STM324x9I-EVAL	STM324x9I-EVAL	0x64000000	2M	0x200000	SRAM	Mode Hot plug
	IS61WV102416BLL_STM324xG-EVAL	STM324xG-EVAL	0x64000000	2M	0x200000	SRAM	Access port
Image: A start of the start	IS61WV1024168LL_STM32769I-EVAL	STM32769I-EVAL	0x68000000	2M	0x200000	SRAM	Reset mode Liandware recet
	IS61WV51216BLL_STM3210E-EVAL	STM3210E-EVAL	0x68000000	1M	0x10000	SRAM	Sneed
	IS66WV51216EBLL_STM32723E-DISCO	STM32723E-DISCO	0x6000000	512K	0x80000	SRAM	Reliable
	IS66WV51216EBLL_STM32F413H-DISCO	STM32F413H-DISCO	0x60000000	512K	0x80000	SRAM	Disabled
	M24LR-A_STM32303C-EVAL	STM32303C-EVAL	0x00000000	8K	0x2000	I2C_EEPROM	External loader 2
	M24LR-A_STM32373C-EVAL	STM32373C-EVAL	0x00000000	8K	0x2000	I2C_EEPROM	Firmware version
	M24LR-A_STM324x9I-EVAL	STM324x9I-EVAL	0x00000000	8K	0x2000	I2C_EEPROM	~
2651 : STM32CubeProgramm	ner API v2.11.0 Windows 448its				Live Lipdate Ver	oosity level 1 2))
							Target information Board Device Type Device ID Revision ID Flash size

Figure 22. Flash memory programming (external memory)



						_	U Con	nect
Download		Erase flash memory Drass	external memory			ST-UNK	• Disco	nnec
File path C\SOFT_DOCS\Test_Files\external_loaders\ftbin	· Browse		Cw60000000	· Erase selected se	tors. Full thip erase		ST-LINK configuration	
Start ad 0x08000000			0x60000000			Serial number	002600383037510	
Skip flash erase before programming		Select	0x90000000	t Address	Size	Post	SWD	
Varity programming				0000000	128K	Frequency (kHz)	24000	
terry programming			1	0w50020000	128K	Mode	Mark and an	
Hun anter programming		U U	2	0x50040000	128K	Access port		=
	Start Program		3	0x50050000	1286			
Automatic Mode			4	0x50080000	1286	Neset model	Software reset	
Full thip erase		H	5	0400040000	1286	Shared	Disabled	
V Download file		- H	7	0+50050000	1286	External loader		
✔ Option bytes commands _ob			8	0x50100000	1286	Target voltage Economic vession		
	Concession of the local division of the loca	- C	9	0x60120000	1286			
Log				Verbosity level	1 02 03			
10:30:67:07:07:07:07:07:07:07:07:07:07:07:07:07					Ĩ €	Roard Onica Type	Target information	15116 571

Г

Т

2.3.3 Developing customized loaders for external memory

Based on the examples available under the *"bin/ExternalLoader*" directory, users can develop their custom loaders for a given external memory. These examples are available for three toolchains: Keil[®] MDK, EWARM, and TrueSTUDIO[®]. The development of custom loaders can be performed using one of these toolchains, keeping the same compiler/linker configurations, as in the examples.

The programming mechanism is the same used by the STM32 ST-LINK utility tool. Any flash loader developed for use with the ST-LINK utility is compatible with the STM32CubeProgrammer tool, and can be used without any modification.

To create a new external memory loader, follow the steps below:

- 1. Update the device information in *StorageInfo* structure in the *Dev_Inf.c* file with the correct information concerning the external memory.
- 2. Rewrite the corresponding functions code in the *Loader_Src.c* file.
- 3. Change the output file name.

Note: Some functions are mandatory and cannot be omitted (see the functions description in the Loader_Src.c file). Linker or scatter files must not be modified.

After building the external loader project, an ELF file is generated. The extension of this file depends upon the used toolchain (.axf for Keil, .out for EWARM, and .elf for TrueSTUDIO or any gcc-based toolchain).

The extension of the ELF file must be changed to ".stldr" and the file must be copied under the "*bin/ExternalLoader*" directory.

Loader_Src.c file

The development of an external loader for a memory, based on a specific IP, requires the following functions:

• Init

Defines the used GPIO pins connecting the external memory to the device, and initializes the clock of the used IPs.

Returns 1 if success, and 0 if failure.

int Init (void)

• Write

Programs a buffer defined by an address in the RAM range.

Returns 1 if success, and 0 if failure.

int Write (uint32_t Address, uint32_t Size, uint8_t* buffer)

• SectorErase

Erases the memory specified sectors.

Returns 1 if success, and 0 if failure.

```
int SectorErase (uint32_t StartAddress, uint32_t EndAddress)
```

Where "**StartAddress**" equals the address of the first sector to be erased and "**EndAddress**" equals the address of the end sector to be erased.

Note: This function is not used in case of an external SRAM loader.



It is imperative to define the functions mentioned above in an external loader. They are used by the tool to erase and program the external memory. For instance, if the user clicks on the program button from the external loader menu, the tool performs the following actions:

- Automatically calls the Init function to initialize the interface (such as QSPI, FMC) and the flash memory
- Calls **SectorErase()** to erase the needed flash memory sectors
- Calls the Write() function to program the memory

In addition to these functions, you can also define the functions below:

Read function

The **Read** function is used to read a specific range of memory, and returns the reading in a buffer in the RAM.

Returns 1 if success, and 0 if failure.

```
int Read (uint32_t Address, uint32_t Size, uint16_t* buffer)
Where "Address" = start address of read operation, "Size" is the size of the read
operation and "buffer" is the pointer to data read.
```

For Quad-/Octo-SPI memories, the memory mapped mode can be defined in the Init function; in that case, the Read function is useless, as data can be read directly from JTAG/SWD interface.

Verify function

Note:

The **verify** function is called when selecting the "verify while programming" mode. This function checks if the programmed memory corresponds to the buffer defined in the RAM. It returns an uint64 defined as follows:

```
Return value = ((checksum<<32) + AddressFirstError)
```

where **AddressFirstError** is the address of the first mismatch, and **checksum** is the checksum value of the programmed buffer.

```
uint64_t Verify (uint32_t FlashAddr, uint32_t RAMBufferAddr,
uint32_t Size)
```

MassErase function

The **MassErase** function erases the full memory.

Returns 1 if success, and 0 if failure.

int MassErase (void)

A checksum function

All the functions described return 1 in case of a successful operation, 0 in case of a fail.

Dev_Inf.c file

The StorageInfo structure defined in this file provides information on the external memory. An example of the type of information defined by this structure is given below:

```
#if defined (__ICCARM__)
    __root struct StorageInfo const StorageInfo = {
#else
    struct StorageInfo const StorageInfo = {
    #endif
        "External_Loader_Name", // Device Name + version number
        MCU_FLASH, // Device Type
        0x08000000, // Device Start Address
```

UM2237 Rev 21



```
0x00100000, // Device Size in Bytes (1MBytes/8Mbits)
0x00004000, // Programming Page Size 16KBytes
0xFF, // Initial Content of Erased Memory
// Specify Size and Address of Sectors (view example below)
0x00000004, 0x00004000, // Sector Num : 4, Sector Size: 16KBytes
0x00000001, 0x00010000, // Sector Num : 1, Sector Size: 64KBytes
0x00000007, 0x00020000, // Sector Num : 7, Sector Size: 128KBytes
0x00000000, 0x00000000,
};
```

2.4 Option bytes

The option bytes panel allows the user to read and display target option bytes grouped by categories. The option bits are displayed in tables with three columns containing the bit name, the bit value, and a description of the impact on the device.

The user can modify the values of these option bytes by updating the value fields, then clicking on the Apply button, which programs and then verifies that the modified option bytes are correctly programmed. The user can click at any time on the Read button, to read and refresh the displayed option bytes.

	Option bytes											
	▼ Read Out Protection											
	Name	Value	Description									
Sob	RDP	AA 🔹	Read protection option byte. The read protection is used to protect the software code stored in Flash memory. AA : Level 0, no protection BB : Level 1, read protection of memories CC : Level 2, chip protection									
	RSS BOR Level User Configuration											
	Boot address Option B											
	▼ PCROP Protection											
	Name	Value	Description									
	PROT_AREA_START1 0xff 0x8001fe0 PROT_AREA_END1 0x0 0x8000000		Flash Bank 1 PCROP start address									
			Flash Bank 1 PCROP End address. Deactivation of PCROP can be done by enbaling DMEP1 bit and changing RDP from level 1 to level 0 while putting									
	DMEP1	\checkmark	Unchecked : Flash Bank 1 PCROP zone is kept when RDP level regression (change from level 1 to 0) occurs Checked : Flash Bank 1 PCROP zone is erased when RDP level regression (change from level 1 to 0) occurs									
	<			;								
				Apply Read								

Figure 24. Option bytes panel

For more details refer to the option bytes section in the programming and reference manuals, available from *www.st.com*.



2.4.1 MCU unlock (specific for the STM32WL series)

The user can unlock the device if bad option bytes are already programmed by clicking on the "Unlock chip" button (available only for STLink connection). After the unlock execution a power cycle is needed.





2.4.2 Debug authentication default configuration (STM32H573/STM32H563/STM32H562 only)

The default configuration is used when programming the product state. It allows the user to provision the configuration after programming the product state to any value. Afterwards, the user can provision its own OBK file.

If the user does not configure the debug authentication (DA) and switches PRODUCT_STATE to provisioned/TZ-closed or closed, it is no longer possible to perform regressions, nor to go back to product state open. All debug features are disabled.

When setting PRODUCT_STATE to 0x17 (provisioning), the user is asked to provision the DA default configuration, or to use its own (see *Figure 26*).




Figure 26. DA default configuration when switching product state to provisioning

When switching PRODUCT_STATE from 0xED (open) to values different from 0x17 (provisioning), the user is asked to pass by the provisioning state first (*Figure 27*).



Figure 27. Configuration when switching product state to values different from 0x17

If the user chooses to provision a default DA configuration, the tool provisions the OBK file under the "bin/ DA_Default_Config" directory. To perform debug authentication, the files under "bin/DA_Default_Config" directory are required.

2.4.3 Debug authentication configuration (STM32H503 only)

If the user does not configure the DA and switches to PRODUCT_STATE provisioned or closed, it is no longer possible to perform regressions, nor to go back to product state open. All debug features are disabled.



UM2237 Rev 21

When setting PRODUCT_STATE to 0x17 (provisioning), the tool checks if there is a password provisioned in OTP. If not, a popup asks the user to set a configuration, to be able to perform a regression later.





When switching PRODUCT_STATE from 0xED (open) to values different from 0x17 (provisioning), the tool checks if there is a password provisioned in OTP. If not, the user is asked to pass by the provisioning state first.





2.5 Automatic mode

This feature, shown in Erasing & Programming window (see *Figure 30*), allows the user to program and configure STM32 devices in loop. Allowed actions:

- Full chip erase: erases all the flash memory
- Download file: activates and sets programming options from Download section:
 - File path
 - Start address
 - Skip erase before programming
 - Verify programming
 - Run after programming
- Option bytes commands: configures the device by setting option bytes command line

Figure 30. Automatic mode in Erasing & Programming window

Prg STN	132CubeProgrammer						— C	X
STM32					(19)		• 🛪 I	$\overline{\mathbf{n}}$
Cube	Fragminer						Cana	ated
	Erasing & Programming						Conne	ctea
	Download	Erase flash m	nemory	Erase external me	mory	ST-LINK	 Disco 	nnect
	File path C:\Users\oueslats\Desktop\apiMulti\test file\data.hex Browse		- Erroro	alacted sectors	ull chip erare	ST-LII	NK configuration	
*			crase s		in chip erase	Serial number	066BFF56	- 0
	Start address	Select	Index	Start Address	Size	Port	SWD	
OR	Skip flash erase before programming		0	0x08000000	16K 🎧	Enanuency (kHz)		
	Verify programming		1	0x08004000	16K	riequency (kiiz)	4000	
			2	0x08008000	16K	Mode	Normal	*
	Kun arter programming		3	0x0800C000	16K	Access port	0	
	Start Programming		4	0x08010000	64K	Reset mode	C (0)	
	Automatic Mode		5	0x08020000	128K		Software rese	
			6	0x08040000	128K	Shared	Disabled	- ()
	- Full chip erase		7	0x08060000	128K	External loader		
	Download file		8	0x08080000	128K	Target voltage		
	Option bytes commands -ob		9	0x080A0000	128K	Firmware version		
			10	0x080C0000	128K			
	Start automatic mode		11	0x080E0000	128K			
	Log	,	Verbosity	r level 💿 1 🤇	2 3			
					4			
						Dev		
							CT (205.42	
B						Device	STM32F42>	xx/F43xxx
						Type		MCU
						Device ID		0x419
6						Flash size		2 MB
÷						СРО	(Cortex-M4

All automatic mode traces are indicated in the Log panel (see *Figure 31*), to show the process evolution and user intervention messages.



Figure 31. Automatic mode log traces



Graphical guide

- Connection to a first target must be established before performing automatic mode to collect connection parameters values associated to all next devices.
- If the Download file is checked, the system takes all Download file options in consideration, otherwise any Download option is performed.
- If the Option bytes commands is checked, the text field is activated, then the user can insert option bytes commands (like CLI commands), and make sure that there are no white spaces at the beginning:
 -ob [OptionByte=value] [OptionByte=value] [OptionByte=value] ...
- Example of Option bytes command: "-ob BOR_LEV=0 nBOOT0=1"
- If the Start automatic mode button is pressed, the system enters in a loop, until a system stop is called.
- While the automatic mode is in execution state, all graphical objects are disabled.
- The user can stop the process at any time by pressing Cancel or Stop automatic mode buttons.



Log messages

- "Starting Automatic Mode..."
 Indicates that the system successfully entered the automatic process.
- "More than one ST-LINK probe detected! Keep only one ST-LINK probe! "

The automatic mode cannot be used if more than one ST-LINK probe is connected to the computer when using JTAG/SWD interfaces. A message is displayed, asking the user to keep only one ST-LINK probe connected to continue using this mode.

• "More than one ST-LINK Bridge detected! Keep only one ST-LINK Bridge!"

The automatic mode cannot be used if more than one ST-LINK bridge is connected to the computer when using bootloader interface SPI/CAN/I²C interfaces. A message is displayed, asking the user to keep only one ST-LINK bridge connected to continue using this mode.

- "More than one ST-LINK USB DFU detected! Keep only one USB DFU!"
 The automatic mode cannot be used if more than one USB DFU is connected to the computer when using USB bootloader interface. A message is displayed, asking the user to keep only one USB DFU connected to continue using this mode.
- "More UART ports detected than last connection!"
 During the first connection the automatic mode calculates the number of the available serial ports, and puts it as a reference, to detect correctly that only one port UART is used for each STM32 device.
- "Please disconnect device and connect the next..."
 If the system finishes the first process, and whatever the result, disconnect the current device to prepare the second device connection.
- "Waiting for device..."
 Once the connection to the previous device is correctly lost, the system keeps searching for a new device.
- "Automatic Mode is stopped."
 Indicates that there is a cancel request, and the system stops the process.





Figure 32. Algorithm



2.6 In application programming (IAP/USBx)

STM32CubeProgrammer supports IAP/USBx only with USB DFU connection mode. When USB connection is chosen and the boot is from flash memory, STM32CubeProgrammer detects the IAP/ USBx like DFU bootloader and after connection an IAP/USBx message appears in the log panel.

Note: Option byte and sector erase are not available with IAP/USBx.

Sample IAPs/USBx are available in CubeFW/CubeAzure on *www.st.com*.

TM32CubeProgrammer	1000	-5 m								□ x
M32 [*] 🖤 ubeProgrammer					(19	f 🕨	y	\star	57
Memory & File	edition							(Conn	ected
	0	1								
Device memory	Open file +	-					USB	*	Disc	connect
Address 0x080	00000 - Size	e 0x400	Data width	32-bit	- Read -			USB config	uration	
							Port	USB	_	- 6
Address	0	4	8	C	ASCII	~	Corial number	_	2	072209620
0x08000000	20001BE0	080033409	08003345	06005547	a4E5G5		Senarnumber			
000000000000000000000000000000000000000	08003349	08003348	0800334D	00000000	13K3M3					
0x08000020	0000000	0000000	00000000	0800334F		_				
0x08000030	08003351	0000000	08003353	08003355	Q3S3U3					
0×08000040	08003425	08003429	0800342D	08003431	%4)4414					
0x08000050	08003435	08003439	08003365	0800343D	5494e3=4					
0x08000060	08003441	08003445	08003449	0800344D	A4E4I4M4					
0x08000070	08003451	08003455	08003459	0800345D	Q4U4Y4]4					
0x08000080	08003461	08003465	08003469	0800346D	a4e4i4m4					
0×08000090	08003359	08003471	08003475	08003479	Y3q4u4y4					
0x080000A0	0800347D	08003481	08003485	08003489	}444					
0x080000B0	0800348D	08003491	08003495	08003499	.444					
0×080000c0	08003490	08003411	08003415	08003449	4 :4 ¥4 ⋒4	~				
Log					Verbosity level 💿 1 💿 2 💿	3				
17:37:06 : STM 17:37:14 : USB 17:37:14 : Manu 17:37:14 : Proc 17:37:14 : STM 17:37:14 : STM 17:37:14 : STM 17:37:14 : IAF 17:37:19 : VPLC 17:37:19 : Kat 17:37:19 : Cat 17:37:19 :	22CubeProgrammer / speed : Full Spe if. ID : STMicroe luct ID : DFU in 1 : 207330863036 version : 0x011a MODING : : 1024 Bytes ess : 0x8000000 progress: : read successful : elapsed during :	API v2.1.2 ed (12MBit/s) lectronics -5 Mode	tion is: 00:00:0	0.005	Ē	\$	Device Type Device ID	Device info	ormation	
					×		Flash size			
						\mathbf{x}	CPU			

Figure 33. STM32CubeProgrammer in IAP mode

2.7 Flash the co-processor binary using graphical interface

2.7.1 FUS/Stack upgrade

- 1. Use STM32CubeProgrammer (version 2.4 or higher), see *Figure 34*
- 2. Access the SWD/bootloader USB interface, see Figure 35
- 3. Delete the current wireless stack, see Figure 36
- 4. Upgrade the FUS version the same way you would download the stack when there is not an updated FUS version
- 5. Download the new FUS
- 6. Download the new wireless stack (a pop-up must appear to ensure successful upgrade), see *Figure 37*

Note: STM32CubeProgrammer (version 2.7 or higher) allows the user to install only new firmware (Stack v1.11.0 or higher). To install the old firmware, use STM32CubeProgrammer v2.6.0.



UM2237 Rev 21

To download WB stacks and FUS from *www.st.com*, press on the logo, as shown in *Figure 38*.

32 DeProgrammer							1 🖸 🖌	* 5
Memory & File	editing							Connected
Device memory	Open file +						ST-LINK	Disconnec
Address 0x08000	0000 👻 Size	0x400	Data width	32-bit 💌 Fin	d Data 0x	Read 🔹	ST-LINI Serial number	K configuration
Address	0	4	8	с	ASCII		Port	CHAID
0x0800000	20000820	0800170D	080015DD	080015DF	ÝB	<u>^</u>		SWD
0x08000010	080015E5	080015EB	080015F1	00000000	åëñ		Frequency (kHz)	4000
0x08000020	00000000	00000000	00000000	080015F7	·····		Mode	Het plug
0x08000030	080015F9	00000000	080015FB	080015FD	ùûý		Access part	
0x08000040	08001739	08001739	08001739	08001739	9999		Heress pore	0
0x08000050	08001739	08001739	08001739	08001739	9999		Reset mode	Software reset
0x08000060	08001739	08001739	08001739	08001739	9999		Speed	Reliable
0x08000070	08001739	08001739	08001739	08001739	9999		Channel	Nenacite
0x08000080	08001739	08001739	08001739	08001739	9999		Shared	Disabled
0x0800090	08001739	08001739	08001739	08001739	9999		Debug in Low Pow	
0×08000040	08001739	08001739	08001739	08001739	a a a a	~	Target voltage	
Log				Live	Update Verbosity level 🔘	1 2 3	Firmware version	
10:16:03 : Size :: 10:16:03 : Bank : 10:16:03 : Address : 10:16:03 : Size : 10:16:03 : UPLOADING : 10:16:03 : Size : 10:16:03 : Address : 10:16:03 : Address : 10:16:03 : Read progr : 10:16:03 : Data reads : 10:16:03 : Time elapse :	96 Bytes : 0x01 : 0x58004080 & Bytes 5 1024 Bytes : 0x8000000 tss: : ccessfully d during the read op	eration is: 00:00:00	.006				Targe Board Device Type Device ID Revision ID	et information STM32WB5MN STM32V G F

Figure 34. STM32CubeProgrammer API SWD connection





• • •	U		
TM STM32CubeProgrammer			- a ×
STREE V			🦥 🖬 🕒 🎽 🛪 🗛
Firmware Upgrade Services			😑 Connected
Firmware Upgrade 🛛 👷	F	US information	ST-UNK • Disconnect
File path C:\Users\bizids\Downloads\stm32wb5x_BLE_HCI_AdvScan_fw.bin	 Browse 	Read FUS infos	ST-LINK configuration
Start address 0x080EB000	Selected file : STACK V1.15.0	FUS State	Serial number 0068FF5548 • Ø
OB First install: No stack delete		DIP Plates	Port SWD V
verify download		POS Status	Mode
Start stack after upgrade		FUS Version	Access part
swv	Message	STACK Version	Reset mode Harviwora rocat
	Firmware delete Success	FUS Operator	Speed Delights
Key Provisioning			Shared Dimbles
Autoendoandon Key :		OK	Debug in Low Power mode
Hie path	Browse update key		Target voltage 3.24 V
[<u>23</u>]	Lock Key	Start Wireless Stack	Firmware version V2357M20
User Key :		Firmware delete	
File path	 Browse Write Key 		
	Simple 👻	Anti-Rollback	
For more information about the customer key storage, refer to ANS185.		Start FUS	
		Lius Lindots Verbruitz Javal 1 2	
T203:90 : Reconnected 1		Dive optime including rotating in the construction of the construc	
12:03:32 : Firmware delete Success 12:03:32 : Download Stack/FUS image at address 0x80EB000			4
1203/32 : Memory Programming 1202/32 : Oranian and particle file stm32wb5x BLE HCL AdvGram for bin			P
120332: File :stm32wb5x_BLE_HCL_AdvScan_fwbin			
12:03:32: Soze : 34:30 KB 12:03:32: Address : 0x080EB000			
12:03:32 : Erasing memory corresponding to segment 0: 12:03:32 : Erasing internal memory sectors [235 243]			
120332 : Download in Progress: 120333 : File download complete			
12:03:33 : Time elapsed during download operation: 00:00:00:903 12:02:23 : Einmunte linearcie encourt studied			
120333 : Application is running. Please Hold on			Target information
(1) 12/03/45 : Reconnected !			Board NUCLEO-W855RG Device 57/432W85x/35xx
1203451 Reconnected 1			Type MCU Device ID 0x495
12:03:45 : Firmware Upgrade Success			Revision ID Rev Y Flash size 1 MB
?		00	CPU Cortex-M4 Bootloader Version 0/05
		04	

Figure 36. Pop-up confirming successful firmware delete





Figure 38. Download STM32WB patches



2.7.2 Key provisioning

STM32CubeProgrammer allows the user to add a customized signature (encrypted and signed by STMicroelectronics) to any image.



UM2237 Rev 21

User authentication

FUS window allows a user authentication key to be stored through the update key button (*Figure* 39).

Once the user authentication key is installed, it can be changed, unless the lock user authentication key button is selected (see *Figure 40*). Once the authentication key is installed, the install or upgrade services must be done with the double signed FUS/Stack, or it is rejected.



Figure 39. Update authentication key



Figure 40. Pop-up requesting to lock authentication key



Customer key storage

STM32CubeProgrammer allows customer keys to be stored in the dedicated FUS flash memory area in binary format (user key types: simple, master, or encrypted), see *Figure 41*.





For more information about the customer key storage, refer to AN5185 "*ST firmware upgrade services for STM32WB Series*". For complete documentation on STM32WBxx products visit the dedicated pages on *www.st.com*.

2.8 Serial wire viewer (SWV)

The serial wire viewer window (see *Figure 42*) displays the printf data sent from the target through SWO, and useful information on the running firmware.

Note: The serial wire viewer is available only through SWD interface.

Before starting to receive SWO data, the user has to specify the exact target System clock frequency (in MHz) to allow the tool to correctly configure the ST-LINK and the target for the correct SWO frequency. The "Stimulus port" combo box allows the user to choose either a given ITM Stimulus port (from port 0 to 31) or receive data simultaneously from all of them.

The user can optionally specify a ".log" file to save the SWV trace log by using the "Browse" button, the default is

"\$USER_HOME/STMicroelectronics/STM32CubeProgrammer/SWV_Log/swv.log".

The user can optionally check the "Activate colors" checkbox to enable colored traces output. This feature requires the original traces to contain the color codes listed below:

- #GRN# for green color
- #RED# for red color
- #ORG# for orange color

Example:

printf("#GRN#This outputs a green message!");



A help window that demonstrates the feature and shows how to use it can be accessed by clicking on the "Info icon" button next to the "Activate colors" checkbox.

Pro STN	132CubeProgrammer	×
STM32 Cube	Programmer 🚳	🖪 🔼 🎽 🔆 🏹
	Serial Wire Viewer	Connected
	Settings	ST-LINK
Ē	System clock (MHz): 32 Stimulus port: All 💌 🗸 Activate colors 🗊	ST-LINK configuration
	SWV Log File Path: Ct\log\swo.log	Serial number 0669FF51 Ø
OB	SWV Frequency: 2000 KHz Status: Stopped Printf data number: 1480	Port SWD -
СРШ	Message Number: 26 Message Number: 27	Frequency (kHz)
	Message Number: 29 Message Number: 29 Stop	Access port
swv	Message Number: 31	Reset mode
-	Message Number: 32 Message Number: 33	Hardware reset
BETA	Message Number: 34 Message Number: 35	Shared Disabled -
	Message Number: 36 Message Number: 37	External loader
	Message Number: 38	Firmware version V2137M26
	Message Number: 59 Message Number: 40	Firmware upgrade
	Message Number: 41 Clear	and the second
		-
	Log Verbosity level 1 2 3	
	10/29/22 : Trace history will be logged in real time in the file: C\log\swo.log	Target information
	10/29/25 : SWV reception stopped	Board P-NUCLEO-WB55
		Device STM32WBxx Type MCU
\bigcirc		Device ID 0x495
<u> </u>		Revision ID Rev Y
(?)	1002	CPU Cortex-M4
	100%	

Figure 42. SWV window

After specifying the SWV configuration, SWV reception can be started or stopped using the "Start" and "Stop" buttons. The SWO data is displayed in the dedicated area, which can be cleared by using the "Clear" button.

The SWV information bar displays useful information on the current SWV transfer, such as the SWO frequency (deduced from the system clock frequency), and the received printf data number (expressed in bytes).

Note: Some SWV bytes can be lost during transfer, due to ST-LINK hardware buffer size limitation.

2.9 Secure programming interface

2.9.1 Introduction

This window facilitates STM32CubeProgrammer CLI commands for secure programming:

- RDP regression with password feature: available for STM32U5 series
- SFI/SFIx feature: available for STM32H7, STM32U5 and STM32L5 series



2.9.2 RDP regression with password (STM32U5 series only)

Some STM32 products offer the possibility to use an optional password-based RDP level regression, including RDP level 2 (*Figure 43*).

- RDP level 1: the OEM1 RDP lock mechanism is active when the OEM1 key is set. It blocks the regression from the RDP level1 (*Figure 45*)
 - To unlock the RDP from level 1 regression, the user must write the OEM1 password, press on "RDP regression" button and then perform the RDP regression from "Option Bytes" interface (*Figure 46*).
 - To remove RDP regression with password from level 1, the user must press on "Disable password" button, as shown in *Figure 47*.
- RDP level 2: provision OEM2KEY to authorize RDP level 2 to level 1 regression: "Set password" button.
 - To unlock the RDP from level 2 regression, the user must write the OEM2 password, press on "RDP regression" button, and then try to connect with STM32CubeProgrammer. If this key matches the OEM2KEY value, the RDP regression to level 1 is launched by hardware.
 - To remove RDP regression with password from level 2, the user must press on "Disable password" button.
- Device authentication ID: Get device identification. Unless the JTAG port is deactivated (OEM2LOCK = 0 and RDP level = 2), a 32-bit device specific quantity can be always read through the JTAG port (see *Figure 48*). The OEM can use this 32-bit information to derive the expected OEM password keys to unlock the device.

Note: Detailed information about this hardware mechanism is available in reference manuals.



Figure 43. RDP regression with password tab



Figure 44. RDP regression with pas	sword from level	
IT STM22CubeProgrammer		
Secure programming		Connected
RDP regression with password SFI/ SFix OBKey Provisioning Debug Authentication SSP	_	ST-LINK V Disconnect
RDP Level 1	Device Authentication ID	ST-LINK configuration
Password: 0x12345678 0xABCDEFAB Set password Unlock RDP1		Port SWD V
CPU The OEM1 lock mechanism is active		Frequency (kHz) 8000
Disable password		Access port
RDP Level 2	ATT 11111	Speed Reliable
Password: 0x12345678 0xABCDEFAB Set password Unlock RDP2	STM32U5	Shared Disabled
The OEM2 lock mechanism is active		External loader Target voltage 3.29 V Firmware version V3J9M3
Disable password		
This interface supports only STM32U5 series!	Read	
Log Live Update	Verbosity level	
1021/40 : Revision ID : Rev B 1021/40 : Debug in Low Power mode enabled.	^ &	
102140: UPLOADING OPTION BYTES DATA 102140: Bank :0:0:0 102140: Bank :0:0:0 102140: JULIADING OPTION BYTES DATA		Target information
102140: Size :36 Bytes 102140: Bank :0x01 102140: Addres :0x020068		Board NUCLEO-057 Device STM32U575/STM32U585 Type MCU
(2) 102140: Size : 8 Bytes 102140: UPLOADING	~	Device ID 0x482 Revision ID Rev B Flash size 2 MB
	100% 🛞	CPU Cortex-M33 Bootloader Version 0x92

regression with password from level 1 Figure 11 PDP

Figure 45. Set OEM1 key (LOCK RDP1)

≡	Secure programming
	RDP regression with password SFI/ SFIx OBKey Provisioning Debug Authentication SSP
	RDP Level 1
OB	Password: 0x12345678 0xABCDEFAB Set password Unlock RDP1
CPU	The OEM1 lock mechanism is active
swv	Disable password
	DDD Laws 10

Figure 46. Unlock RDP1

	Secure proc	gramming					
	RDP regressio	on with password	SFI/ SFIx	OBKey Provisioning	Debug Authentication	n SSP	
.	RDP Level 1					Û	
OB	Password:	0x12345678	0x/	ABCDEFAB	Set password	Unlock RDP1	
CPU	The OEM1 lock	mechanism is	active				
swv						I	Disable password
	0001 14						



_

Figure 47. Disable password

≡	Secure programming
	RDP regression with password SFI/ SFIx OBKey Provisioning Debug Authentication SSP
.	RDP Level 1
OB	Password:
CPU	The OEM1 lock mechanism is not active
swv	Disable password

Figure 48. Get authentication ID



2.9.3 SFI/SFIx

SFI GUI

- 1. Use STM32CubeProgrammer (version 2.11 or higher)
- 2. Access the SWD/bootloader interface
- 3. Open Secure Programming interface, then SFI tab (*Figure 49*)
- 4. Select the license source (from a license file or directly from your connected HSM)
- 5. Select the sfi file, once selected, the sfi parsed info is displayed as in *Figure 51*
- 6. Select the RSSe file (if needed)
- 7. Start sfi sequence (see *Figure 50*)

Note: To open TPC to generate the sfi file, press on the TPC logo, as shown in Figure 52.



UM2237 Rev 21

```
Figure 49. SFI/SFIx tab
```

		Γ			Cabe	
11:101 Anna Anna Anna Anna Anna Anna Anna Ann	ung 113150 Processing Ares 1 113150 Gree Steaman	lan	Markey provide a function of a start of a st	RDP regression with password BFU/STIR	rogrammer Sacura organomina	CubeProgrammar
			rowa			
			SFI/SFix Segments Index 1 2 3			
			Information Type Finnware Finnware Option Bytes			
			n 6380 5370 64			
		The second second second second second second second second second second second second second second second se	Address 0x8100000 0 0x030000 0 0x0 0			
	e uparte Venosity ev	e Undrite Verbruite lau	HSM Former 00 Max counter: HSM Status Varies Type: Type: STM322 Construct Construct Construct			
100%	4		394: 1 Color0000; Lolo; 352 Color000; Lolo; 352 Color000; Lolo; 352 Color000; Lolo; 352 Color000; Lolo; 352 Color00; Lolo; 352 Color00; Color0; Lolo; 352 Color0; Color0; Color0; Color0; Color0; Color0; Color0; Color0; Color0; Color0; Color0; Color0; Color0; Color0; Color0; Color0; Color0; Color0; Color0; Color0; Color0; Color0; Color0; Color0; Color0; Color0; Color0; Color0; Color0; Color0; Color0; Color0; Color0; Color0; Color0; Color0; Color0; Color0; Color0; Color0; Color0; Color0; Color0; Color0; Color0; Color0; Color0;			39
Target Board Dekke Type Dekke ID Revise ID Revise ID Revise ID Robit Star Bootcoder Version			T the control of the	ST-EDIK		FI 🖸 🎐
information NUCLEO-USTSZI-C STM32USTSTSTM32USH MCL Ovelt Ray 2 M Contec-M3			exclusion and a second se	Disconnected	Connected	- 0 ×

Figure 50. Steps for SFI programming

			1
Using License	from file C:\Users\bizids\Downloads\CobraEV_SFIX\u5evallicense.bin	•	Browse
Using Licence	from HSM		
Encrypted firmwar	e 🚱		2
SFI/SFIx Path C	C:\Users\bizids\Downloads\outavecHash_U5.sfi	-	Browse
RSSe	C:\Program Files\STMicroelectronics\STM32Cube\STM32CubeProgrammerv2.11.0-B02\bin\RSSe\U5\enc_si	-	Browse
Ext Loader			



egments			
Index	Туре	Size	Address
1	Firmware	64	0x800a000
2	Firmware	6240	0x8100000
3	Firmware	4048	0xc000000
4	Firmware	16	0xc0fe000
5	Pause	32	0x80f0000
6	Resume	32	0x80f0000
7	External Firmware	272	0×70000000
8	Pause	32	0x80f0020
9	Resume	32	0x80f0020
10	Option	64	0x0

Figure 51. SFI parsed info

Figure 52. Open TPC from STM32CubeProgrammer



SFIx GUI

To perform a successful sfix operation using this graphical interface, perform the same steps described in *SFI GUI*, with two minor modifications:

- 1. Select an sfix (not an sfi).
- 2. Select the external loader via External loaders window. Once done, the name is displayed automatically in the text field below (*Figure 53*).

Then you can start the sfix sequence.

gare eer bieplag external leader hante	Figure 5	53. Dis	play e	external	loader	name
--	----------	---------	--------	----------	--------	------

Ext Loader	512W3A_STM3210E-EVALstldr
S	STM32



HSM related info

This panel contains all the needed information in the sfi process. It allows the user to read the available HSM information when a card is detected, get the license from HSM, get the chip certificate, and read the product ID.



Figure 54. HSM-related info panel

For more details refer to AN5054 "Secure programming using STM32CubeProgrammer", available on www.st.com.

2.9.4 SSP

STM32CubeProgrammer user interface (UI) exports several capabilities that can be used to execute the SSP flow.

To open the SSP window, connect an MPU device via DFU interface, click on security panel, and then choose the SSP tab. The window contains the graphical components needed to perform SSP operations.



Figure 55. SSP PRG user interface

TT STA	132CubxProgrammer				- 🗆 ×
snu 2			<u>(10)</u>	FI 🖸 🎐	* 🗖
≡	Secure programming				Connected
	RDP regression with password SFI/ SFIx SSP			USB	 Disconnect
	License source selection 3	нѕм 1	Slot: 1	USB o	onfiguration
	From file Browse	Firmware ID:	MPU_SSP500020A	Carial avandars	USB1 - Ø
	From HSM	Max counter:	8934	PID	0xdf11
OTP	Input binaries 🚱 👍	HSM Status:	OPERATIONAL_STATE	VID	0x0483
	SSP path erowse	Version:	2	Read Unprotect (N	(CU)
	Tfa-ssp path 💌 Browse	Type:	SSP	TZEN Regression (MCU)
	Start SSP	Ø Refresh			
	5	Get License from H	SM		
		-			
		Device 2			
		Get Product ID	Product ID : -		
		Actual Phase ID: 0	x1		
	Log Live	Update Verbosity lev	el 🖲 1 🦳 2 🦳 3		
	1153:88: Disconnected from device. 1153:04:USB speed (HBI)MBit/s)		4		
	11:53:40 (Manut, ID: 15) Microelectronics 11:53:40 (Product ID: DFU in H5 Mode @Device ID /0x501, @Revision ID /0x1000		e 199		
	115340 C3W = 60 (adducts)5051045656552 115340 DFU protocol: 1.1				
	115340/1806/0 1 115340/1904/e ID 10/0501				
	115348 : Init Communication 115348 : Init Communication Success!				
	113348 : Opening session with slot ID 1 113348 : Succeed to Open session with reader slot ID 1				
	113348 : GetFirmwardehntfiler SuccessIFW ID : MPU_SSP\$00200A 113348 : GetCounter SuccessIFW ID : MPU_SSP\$00200A 113348 : GetCounter SuccessIFW ID : MPU_SSP\$00200A				
	1153/48 : Plugged Card current status is : OPERATIONAL_STATE 1153/48 : GetVention SuccessI Vention value is : 2				
	11:53:48 : GetType Success! Type value is : SSP 11:53:48 : Cloring session with reader dot 10 1			Board	t information
	115348 : Session closed with reader slot D 1			Device Type	STM32MP13xx MPU
	1153x46 : Liosing communication with h5M			Device ID Revision ID	0x501
0				Flash size	 Contex-A7
$\langle \cdot \rangle$			0% 🗵	Bootloader Version	n

The UI is composed of five elements:

- 1. HSM section
- 2. Device section
- 3. Input license
- 4. SSP input binaries
- 5. Verify and Start SSP install

HSM section

This section allows the user to read HSM information when the smart card is detected in the slot selected by the user. With this tab it is also possible to get a license from HSM.

Press "Refresh" button to read and display the related information for the plugged HSM.

Device section

This part allows to get the product ID of the connected device, needed to choose the correct personalization package to be provisioned in the HSM card by the STM32Trusted Package Creator.

If the device is not in a configuration adequate to get the chip certificate, it is mandatory to provide a tfa-ssp file in the "SSP inputs" section. STM32CubeProgrammer will then set the device in the correct state.

The current phase ID is displayed to highlight the device configuration.



Input license

The user must select the source of the license to be used in the SSP flow. Possible sources:

- From file: to select a binary file already generated by the HSM.
- From HSM: the SSP procedure extracts the license directly from the HSM.

SSP input binaries

This part is needed to select the secure input files:

- SSP file: an encrypted SSP image generated by STM32Trusted Package Creator
- tfa-ssp: binary file with .bin or .stm32 extension

To generate a SSP image, launch directly the SSP generation window of STM32Trusted Package Creator tool by clicking on the TPC icon.

Verify and Start SSP install

When the user clicks on "Start SSP" button, STM32CubeProgrammer verifies all mandatory inputs and starts the SSP procedure.

If the operation is successful, an informative popup is displayed, and the device is disconnected.

If an error occurs at any step, the operation stops the flow and displays an error.

Notes

At the end of the flow STM32CubeProgrammer does not make any verification step for the OTP fused words.

If the mentioned tfa-ssp is not the adequate one, the SSP flow fails.

2.9.5 OBKey provisioning (STM32H573/STM32H563/STM32H562 only)

This is a security feature that allows to program OBKey content.

There are two possible cases of OBK provisioning for debug authentication:

- Provisioning with password: before launching it, set TZEN at "0xC3" (disabled), and product state at "0x17" (provisioning).
- Provisioning with certificate: before launching it, set TZEN at "0xB4" (enabled) and product state at "0x17" (provisioning).



Figure 56. OBKey provisioning

Autoritigianime				
une				fi 🕒 🎽 🛧 🗳
Secure programming				Connected
RDP regression with password SFI/ SFI/ OBKey Provisioning Debug Authentication SSP			ST-UNK	 Disconnect
CBKey file path	name	value		ST-LINK configuration
Select File Cs,Program Files,STMicroelectronics;STM32Cube/STM32CubeProgrammer_DA.password/binit, 💌 Browse	Destination	0x0ffd0100	Serial number Port	00004002C3756501620303 •
OBKey Provisioning Description			Frequency (kHz)	8000 -
	Size	0x0000000	Mode	Hat plug
The Os key like is lake to program Oskay content.			Access port	1 *
-OBKey files are generated by STM32 Trusted Package Creator based on XML template. OBKey Provisioned acceptuly CMProgram OBKey Trusted Package Creator based on XML template.	Do Encrypt	00000000	Keset mode	Software reset
ADBK file info: DA gessored binDA Default Config VienCrypto DA Config Ce			Shared	Disabled
1) Destination address: OBKey physical address. 2) Destination address: destey address. execution or not			Debug in Low Pow	er mode
2) Obtain the mean and request output of the second s			External loader	
Orbites for DA Deprovisioning with passived is valid only when TZ-1. CA provisioning with passived is valid only when TZ-6.		Start Provisionin	Target voltage Firmware versio	
		Start Provisionin	Target voltage Firmware versio	
Of Vote for QA Dapprovidency with passed is valid only when TZ-1 DA providency with passed is valid only when TZ-0 Of the topology of topology of the topology of the topology of topology	iate Verbosity level	Start Provisionie	Tanget voltage Firmware versio	
	iate Verbosity level	Start Provisionin	Target voltage Firmware versio	
Preview to AL Approximate an odd only when T2=1. DA proximate with certificate a valid only when T2=4. DA proximate with presence of a valid only when T2=4. So So Co Live type Live type Do Live type	late Verbosity level	Start Provisioning	Target voltage Firmware versio	
Broke top A. Daproscomy with particular is wald only when T2-1. Da prosco	late Verbosity level	Start Provisional	Target voltage Firmware versio	
	late Verbosity level	Start Provisional	Target voltage Firmware venic 3	
Per Venter IV D. D. Approximation with certificate is weld only when T2=1. D. D. Approximation with paramed is valid only when T2=1. D. Approximation with paramed is valid only when T2=1. D. D. D. D. D. D. D. D. D. D. D. D. D. D	late Verbosity level	Start Provision	Target voltage firmmare venic 3 3	
	late Verbosity level	Start Provision	Target voltage Firmware venic 3 3	
Pilete to L Deprocessing with certificate is weld only were T2-1. DE processing with personal is weld on the personal is weld on	late Verbosity level	Start Provision	Tangé vélagé Formare vénic 3	
	iate Verbosity level	Start Provision	Tangé vélagé Tantasé vénik 3	Taget Information
	ate Verbosity level	Start Drevisions	Target voltage Farmane winks	Tagel information noccors with the state
Pieter tol. Deprocineng with predicted is vaid only with TL-1. Deprocineng with the the the the the the the the the t	ate Verbosity level	Start Provisional	Board Board Board Board Board Board Board Board Board Board Board Board	Target Information Notices stress Stress
Pierke tro A. Dappenoom yuth entificate suid only uken T2-1. Dappenoom yuth presend it wild only uken T2-1. Dappenoom yuth presend it wild only uken T2-1. Dappenoom yuth presend it wild only uken T2-1. Coj	late Verbosity level		Bond Bond Bond Bond Bond Bond Bond Bond	Taget Information NocIES State

OBKey file generation is managed by STM32 Trusted Package Creator. For more information, refer to UM2238.

2.9.6 Debug authentication (STM32H5 series only)

This feature allows regression or open debug when the target is provisioned.

To perform debug authentication the user must provide the needed credentials by following these steps:

- Discovery: this operation displays the info about the target. To make sure that provisioning is correctly performed, "Integrity status" field must be checked: it must indicate "0xEAEAEA" as value (displayed in the log). This operation cannot be launched while the target is connected.
- 2. Credentials input: beside discovery related data, debug authentication panel displays at this step a form including the necessary inputs:
 - a) TZEN = 0xC3 (disabled): the form includes only password file path. In this case, only full regression with password is available.
 - b) TZEN = 0xB4 (enabled): the form includes key and certificate path inputs. In this case, the user can select the possible permission.



STIMB2	Sub-Programmer					- e • •	
	Secure programming					Not co	onnected
	RDP regression with password SFV SFIx OBKey Provisioning Debug Authentication SSP				ST-UNK	• Connec	rt
<u> </u>	Password File Path		Discover			ST-UNK configuration	
	Select File C1/Program Files/STMicroelectronics/STM32Cube/STM32CubeProgrammer_DA_password/bin/t 💌 Browse Full Regression	name	value		Serial number	000A002C3756501620303	- 0
OB		Locking Mechanism	Password		Port Fragmency (kHz)	SWD	*
CPU		Soc ID	0x00000000 0x00000000 0x00000000 0x0000	00000	Mode	Hot plug	*
믐		Life Code	CT LIFECUCIE PROVINCIANING		Access port	1	*
SMA		che Cycle	ST_DPECTCLE_PROVISIONING		Reset mode	Software reset	*
1		Device ID	0x484		Speed	Reliable	*
					Shared Detroio Iour Power	Disabled	· 0
		Sti	ep 1: Path selection.		External loader Target voltage	3.29 V 🛌	
		st	ep 2: Execution.		Firmware version	V3J8M3	
		\bigcirc					
	This interface supports only STM32H9						
		Live Up	fate Verbosity level 💌 1 💮 2 🦷	3			
	0907:57 : open_comms : 442 : open : De-asserting target reset		^				
	09/07:57 : open_comms : 488 : open : Communication with the target established successfully 09/07:57 : discovery: target ID			Š.			
	09/07:57 : discovery: SoC (D						
	090757 : discovery: Vendor IDSTMicroelectronics 090757 : discovery: PSA lifecycleST_LIFECYCLE_PROVISIONING						
	09/07:57 : discovery: PSA auth version1.0 09/07:57 : discovery: ST HDPL1 status						
	090757 : discovery: ST HDPL2 status					Target information	
	090757 : discovery: Token Formats				Device		1
\bigcirc	090757 : discovery, cryptosystems				Device ID Revision ID		-
0			v		Flash size CPU		1
Ŷ			0%	\otimes	Bootloader Versio		

Figure 57. Debug authentication with password

Figure 58. Debug authentication with certificate

eProgrammer				a a a a a a a a a a a a a a a a a a a
ure programming				4
P regression with password SFI/ SFix OB	Cey Provisioning Debug Authenti	ion SSP		
y File Path				Discover
			name	value
C:\Cuberrg(V2.13.0\bin\DA_Detault_	Lonng(key_s_iear.pem	Browse	Locking Mechanism	Certificate
rtificate File Path			Soc ID	0xFFFFFFFF 0xFFFFFFF 0xFFFFFFFF 0x000000
ect File C\CubePrmv2.13.0\bimDA_Default (Config\cert leaf chain.b64	Browse Continue		
			Life Cycle	ST_LIFECYCLE_PROVISIONING
missions			Device ID	0x484
Permission	Select			
Ion-Secure Intrusive Debug (Level1)				ep 1: Path selection.
Non-Secure Intrusive Debug (Level2)				
Non-Secure Intrusive Debug (Level2)			¥*	ep 2: Permission selection
ton secure minume bebug (cereis)			0 20	ep 3: Execution.
Secure Intrusive Debug (Level1)	•			
Secure Intrusive Debug (Level2)				
Secure Intrusive Debug (Level3)	•			
Partial Regression				
Full Regression		Evenide		
This interface supports only STM32H51				
19:36 : discovery: PSA auth version			Live up	date verbosity level • 1 • 2
1936 discovery ST HOPL 1 atus	6 5HA256 Tabadada			

Note:

For STM32H503, only a full regression with password is available.

2.10 STM32CubeProgrammer Script Manager platform for MCUs

2.10.1 Introduction for the usage scenarios of Script Manager

The Script Manager platform allows to automate STM32CubeProgrammer CLI commands and adds macros to manipulate data read from STM32 MCU.



2.10.2 Script Manager usage

Create a file with a prg extension, then start writing the command line interface (CLI) supported by all STM32 MCUs and the specific script macros. Once you have finished filling the script, connect the STM32 board and start execution with the *-script* command in CLI mode.

Usage example: STM32_Programmer_CLI -script myScript.prg

The Script Manager can apply mathematical and logical operations (see *Table 1*).

Mathematical	Logical
– + (addition)	– && (logical AND)
 – - (subtraction) 	– (logical OR)
– * (multiplication)	– & (bitwise AND)
– / (division)	– (bitwise OR)
	– ^ (XOR)
	< >> (left and right shift)

Table 1. Operations supported by Script Manager

Using command line interface (CLI): in this script we can use all CLI supported by STM32 MCUs (see *Section 3*).

Using specific Script Manager macros, to analyze, display and modify data, each macro starts with #. Supported macros are described below.

#Write macro:

```
#Write32(Address,data)
#Write16(Address,data)
#Write8(Address,data)
#WriteX(Address,#var)
```

(where X is 8/16/32)

Description: Downloads the specified (32/16/8-bit) data into flash memory starting from a specified address.

#Read macro:

#Read(Address)
#variable=#Read(Address)

Description: Reads 32-bit data memory from a specified address or reads 32-bit data memory from a specified address, and puts it in the used variable.

```
#Display macro:
```

```
#Display("message")
#Display(#errorLevel)
#Display(#variable)
```



UM2237 Rev 21

Description: Displays any message, data, error level and the content of variables already used in the script.

#Delay macro:

#Delay(Time)

Description: Allows user to put the system in standby for a period in (ms).

Calculate macro:

```
#variable=[var1] op [var2]
```

#variable=var1 shift (number of bits to shifted)

Description: Calculates with mathematical and logical operations in script manager.

Disconnection command

--scriptdisconnect

Description: Allows user to disconnect the device and reconnect to another port in the same script.

Note:

Comments in the Script Manager can be added by using "//", as shown in the examples.

Script Manager example 1 (CLI and Script macro), see Figure 59

```
-c port=swd
-e 0 1
#Write32(0x08000000,0xAAAABBBB)
#var0=#Read(0x08000000)
#Display(#var0)
Script Manager example 2, see Figure 60
-c port=swd
#Write32(0x08000000,0xAAAABBBB)
--scriptdisconnect
#Delay(5000)
-c port=COM17
#Write16(0x08000004,0xCCCC)
```

```
Script Manager example 3
-c port=swd
#Display ("Hello World!")
-e 0 1
#Write32(0x08000000,0xAAAABBBB)
#Read(0x08000000)
-r32 0x08000000 0x50
#var0=#Read(0x08000000)
#Display(#errorLevel)
```



```
#Display(#var0)
#Write32(0x08000004,#var0)
#Delay(3000)
#Write16(0x08000008,0xCCCC)
#Read(0x08000004)
#Display(#errorLevel)
#var1=#Read(0x08000008)
#Display(#var1)
#Write8(0x08000010,0xDD)
#Delay(5000)
#var2=#Read(0x08000010)
#Display(#var2)
#var3=(((0xbb*1)+(1-1))/1)
#Display(#var3)
#Write8(0x08000014,#var3)
#var4=((0xbb & 0xaa) | 0xbb )
#Display(#var4)
#var5=((0xbb && 0xaa) || 0xbb )
#Display(#var5)
#var6=(0xbb >>1)
#Display(#var6)
-e 0 1
-w32 0x08000000 0xAAAAAAA
-r32 0x08000000 0x50
```

=== Script Manager BEGIN ===
Operation [1]: -c port=swd
ST-LINK SN : 0668FF565251887067053951 ST-LINK FW : V2J33M25 Board : NUCLEO-F429ZI Voltage : 3.27V SWD freq : 4000 KHz Connect mode: Normal Reset mode : Software reset Device ID : 0x419 Revision ID : Rev 3 Device name : STM32F42xxx/F43xxx Flash size : 2 MBytes Device type : MCU Device CPU : Cortex-M4 BL Version :
Operation [2]: -e 0 1
Erase sector(s)
Existing specified sectors are erased successfully Protected sectors are not erased
Operation [3]: #Write32(0x08000000,0xAAAABBBB)
DOWNLOADING Size : 4 Bytes Address : 0x08000000
Data downloaded successfully
Operation [4]: #var0=#Read(0x08000000)
UPLOADING Size : 4 Bytes Address : 0x8000000 Read progress:
Data read successfully Time elapsed during the read operation is: 00:00:00.001
Operation [5]: #Display(#var0)
#var0 = 0xAAAABBBB Device is disconnected
Script Manager END

Figure 59. Output of Script Manager example 1



Figure 60. Output of Script Manager example 2

```
Script Manager BEGIN
ST-LINK SN : 0668FF565251887067053951
ST LINK FW : V2J33M25
Board : NUCLEO-F429ZI
Voltage
                   : 3.27V
                   : 4000 KHz
SWD freq
Connect mode: Normal
Connect mode: Normal
Reset mode : Software reset
Device ID : 0x419
Revision ID : Rev 3
Device name : STM32F42xxx/F43xxx
Flash size : 2 MBytes
Device type : MCU
Device CPU : Cortex-M4
BL Version : --
 operation [2]: #Write32(0x08000000,0xAAAABBBB)
DOWNLOADING ...
Size : 4 Bytes
: 0x08000000
Erasing internal memory sector 0
Data downloaded successfully
The system go to sleep for 5000 ms.
Port configuration: parity = even, baudrate = 115200, data-bit = 8,
stop-bit = 1.0, flow-control = off
Timeout error occured while waiting for acknowledgement.
 Activating device: OK
Board
Chip ID: 0x419
Chip ID: 0x419
BootLoader protocol version: 3.1
Device name : STM32F42xxx/F43xxx
Flash size : 2 MBytes (default)
Device type : MCU
Revision ID : --
Device CPU : Cortex-M4
 Operation [5]: #Write16(0x08000004,0xCCCC)
DOWNLOADING ...
                : 2 Bytes
: 0x08000004
   Size
   Address
Erasing internal memory sector 0
                                                                                   Data downloaded successfully
  evice is disconnected
```



2.10.3 Loops and conditional statements

The Script Manager supports three macros for looping and conditional statements:

```
#Start
                           #Start
                                                       #Start
While (***)
                           for (***)
                                                       if (***)
                           {
£
                                                       {
Command Line or macros
                            Command Line or macros
                                                        Command Line or macros
}
                           }
                                                       }
#End
                           #End
                                                       else
                                                       {
                                                        Command Line or macros
                                                       }
                                                       #End
```

To use the conditional statements (if, else) and the loopings (While, for), begin with **#Start**, and finish with **#End**.

If-else condition example

```
-c port=swd
-e 0
#x=#Read(0x08000000)
#Start
if (#x > 0x1000)
{
  #Display("Condition 1")
  #Display(#x)
  #Write32(0x08000000,0x1123)
}
else
{
  #Display("Condition 2")
  #Display(#x)
  #Write32(0x08000008,0x1124)
}
#End
For loop
-c port=swd
#ADD=0x08000004
#x=#Read(0x0800004)
```

```
#Start
#Display(#x)
```



```
{
   #x=#Read(#ADD)
   #Display(#x)
}
#End
While loop (example 1)
-c port=swd
-e 0
#Write32(0x08000008,0xCCCCCCC)
#ADD=0x08000000
#x=#Read(#ADD)
#Start
while(#x!=0xCCCCCCCC)
{
    #Display(#x)
    #ADD=([#ADD]+(4))
    #x=#Read(#ADD)
}
#End
While loop (example 2)
-c port=swd
-e 0
#Write32(0x08000000,0xAAAAAAA,0xBBBBBBBB,0xCCCCCCC,0xDDDDDDDD)
#ADD=0x08000000
#x=#Read(#ADD)
#Display(#x)
#Start
while(#x!=0xDDDDDDDD)
{
    #Display(#x)
    #ADD=([#ADD]+(4))
    #x=#Read(#ADD)
}
#End
```

for(#ADD=0x08000000;#ADD<0x0800000C;#ADD=#ADD+4)</pre>



2.11 DFU IAP/USBx with custom PID and VID

STM32CubeProgrammer DFU IAP/USBx supports not only ST product IDs while connecting via DFU IAP.

Before starting the DFU connection using a new product ID, sign your USB driver (for more info visit http://woshub.com).

When USB connection with a new product ID is chosen and the boot is from flash memory, STM32CubeProgrammer detects the IAP/USBx like DFU bootloader and after connection an IAP message appears in the log panel.

To connect via the new USB DFU follow this sequence:

- 1. Modify the default product ID
- 2. Modify the default vendor ID
- 3. Click on refresh button then on the connect button

Note: If user does not enter a PID or VID value STM32CubeProgrammer takes the default PID and VID of ST products (PID = 0xDF11, VID = 0x0483).

Figure 61 shows the steps to connect via the new USB DFU panel, and *Figure 62* the main window of STM32CubeProgrammer after connection.

Prg STM3	2CubeProgrammer		—	
STM32 CubeP	rogrammer	🦻 🗗 🕒	9	: 57
	Option bytes		🛑 N	ot connected
	Read Out Protection	USB	•	Connect
	PCROP Protection BOR Level		JSB configuration	
*	User Configuration	Port	USB1	- 0
OB	Write Protection	Serial number PID	0xA38F	306934663235
СРИ		VID	0x0483	2
swv		Read Unprotect	MCU)	
	Apply Read			
	Log Verbosity level 1 2 3			
	1715471;SN : 306934663235 1715471;RV version : 0.011a 171547:IAP 1715484:UP(JADING		larget informatio	'n
	171548: Stee : 1024 Bytes 171548: Addres: 058000000 171548: Addres: 058000000 171548: Addres: 058000000 171548: Addres: 058000000 171548: Addres: 058000000 171548: Addres: 058000000 171548: Addres: 058000000 171548: Addres: 058000000 171548: Addres: 058000000 171548: Addres: 058000000 171548: Addres: 058000000 171548: Addres: 058000000 171548: Addres: 058000000 171548: Addres: 058000000 171548: Addres: 05800000 171548: Addres: 05800000 171548: Addres: 05800000 171548: Addres: 05800000 171548: Addres: 058000000 171548: Addres: 05800000 171548: Addres: 058000000 171548: Addres: 05800000000000000000000000000000000000	Board Device Type		-
$\textcircled{\begin{tabular}{ c c c c } \hline \hline \hline \hline \hline \hline \hline \hline \hline \hline \hline \hline \hline \hline \hline \hline \hline \hline \hline$	171548 : Time elapsed during the read operation is: 00:00:00:004 171551 : Disconnected from device.	Device ID Revision ID Flash size		-
(?)	100% 🛞	CPU		-

Figure 61. Connect via USB DFU panel



			•							
STM	132CubeProgrammer								-	ЦХ
STM32 Cube	Programmer							🖻 🖪 🕒	⊻ ⊁	: 57
	Memory & File	edition							<u> </u>	onnected
	Device memory	Open file +						USB	Ŧ	Disconnect
	Address 0x80000	000 💌 Size	0x400	Data width	32-bit 👻	ind Data 0x	Read 🔹	U	SB configuration	
	Address	0	4	8	С	ASCII		Port	USB1	- C
	0x0800000	20000FE0	08004D4D	08004CE9	08004CEB	à MMéLëL	â	Serial number		306934663235
	0x08000010	08004CED	08004CEF	08004CF1	00000000	íLïLñL		PID	0xA38F	
CDU	0x08000020	00000000	00000000	00000000	08004CF3	óL		VID		_
CPU	0x08000030	08004CF5	00000000	08004CF7	08004CF9	õL÷LùL		110	0x0483	
SWV	0x08000040	08004D69	08004D6D	08004D71	08004D75	iMmMqMuM		Read Unprotect (I	VCU)	
Jane	0x08000050	08004D79	08004D7D	08004D81	08004D85	уМ}МММ				
	0x08000060	08004D89	08004D8D	08004D91	08004D95	.MMM				
	0x08000070	08004D99	08004D9D	08004DA1	08004DA5	.M,M;M¥M				
	0x08000080	08004DA9	08004DAD	08004DB1	08004DB5	GMΜ±ΜμΜ				
	0x08000090	08004DB9	08004DBD	08004DC1	08004DC5	¹М½МА́МА́М				
	0x080000A0	08004DC9	08004DCD	08004DD1	08004DD5	ÉMÍMÑMŐM	~			
	Log					Verbosity level	1 2 3			
	17:16:36 : Manuf. 10 17:16:36 : Product ID 17:16:36 : SN : 17:16:36 : FW version	: DFU in FS Mode 306934663235 1: 0x011a					^ 4			
	17:16:36 : IAP							T	arget information	<u>ו</u>
A	17:16:37 : OPLOADIN 17:16:37 : Size	: 1024 Bytes						Board		-
	17:16:37 : Address	: 0x8000000						Type		-
	17:16:37 : Read prog 17:16:37 : Data read :	ress: successfully						Device ID		-
	17:16:37 : Time elaps	ed during the read op	peration is: 00:00:0	00.004				Revision ID		-
0	L						~	Flash size		-
\odot							100% 🗙	CPU		-

Figure 62. Main window after the connection

Note: For CLI mode check the Section 3.2.1: Connect command.

2.12 SigFox[™] credentials

As soon as an STM32WL device is connected, the window shown in Figure 63 is displayed.

This window displays the chip certificate, having the size of 136 bytes. The user can save it in binary file and copy the data to the clipboard.

After extracting the chip certificate, a back-end web-service verifies the data and returns two SigFox credentials: binary and header files.

Case 1: Binary-Raw

Use the binary file returned by the back-end web-service. The size of this file must be equal to 48 bytes, it is written at the default address 0x0803E500.

Case 2: Binary KMS

Use the header file returned by the back-end web-service. It is written at the default address 0x0803E500.

Note: To access ST SigFox server using STM32CubeProgrammer, user must click on "Open Sigfox page". A web page opens, the user must manually copy the certificate and then generate the SigFox credentials (binary and header files).



Figure 63. SigFox credentials

Pro STM	132CubeProgramme	r		- 🗆 X
STM32 Cube	Programmer	6	🦻 🖪 🕒	💆 🔀 🟹
	SigFox Crede	ntials		Connected
	Chip certificate		ST-LINK	 Disconnect
	34 39 37 30 31 30 1D 82 F4 8A 9A 1 D0 4D C3 4A E9 D	30 35 07 D7 60 65 98 2A FE 36 29 CA 59 F3 D5 29 98 99 F7 A3 4E C0 8B 15 5F D1 2D D3 C9 2A 9A 02 C0 98 D8 10 FC 2D 28 D9 C9 77 8C 4C BA 38 5B 15 E5 D8 D8 DB 12 46 BA 8C 57 E1 AF4 24 1D DB E3 14 B8 74 8B 14 74 A8 D6 38 B6 74 4E 5A 1	ST	-LINK configuration
	63 CA 76 6B DB A 8C DC 8D 13 97 1	3 80 CF E0 61 F3 01 07 05 DD 6C 74 F6 29 23 17 8F 8D E7 C5 C8 3A 5C 0E 58 58 A3 AB 52	Serial number	50FF6E06726557 👻 💋
OB			Port	SWD
<u> </u>			Frequency (kHz)	4000 -
CPU		Save chip certificate Copy chip certificate	Mode	Hot plug 👻
swv			Access port	0 -
≓	SigFox credentia	l provisioning	Reset mode	Hardware reset 🔹
X	Configuration	Binary-Raw Address 0x0803F800	Shared	Disabled 👻 👔
REG	Binary file	Select binary file	External loader	-
BETA	Header file	Select header file	Target voltage	3.24 V
		Write data	THINWALE VEISION	Firmware upgrade
	Log	Verberity Javel 0 1 0 2 3		
	Log			
			1	arget information
			Board Device	 STM32WLxx
			Туре	мси
(\mathscr{O})			Device ID Revision ID	0x497 Rev 1.1
6			Flash size	256 KB
\bigcirc		100% 🛞	CPU	Cortex-M4

2.13 Register Viewer

STM32CubeProgrammer supports the Register Viewer feature (see *Figure 64*), allowing the user to visualize all the MCU and core registers in real time while running the application. It also allows the modification of MCU registers values or saving them into a log file.

Figure	64. Registe	r Viewer window	

700 STM32	CubeProgrammer				– 🗆 ×
STM32 CubeF	rogrammer			1	F 🕒 🔰 🔆 🖅
	Registers - BETA				Connected
	Device : STM32L152	n: Peripher	Control	Save to file	ST-LINK
I.	Name	Value	Access	Address	ST-LINK configuration
	ACTLR	0x00000000	ReadWrite	© 0xE000E008	Serial number 0667FF3 👻 💋
OB	► ICSR	0x00000000	ReadWrite	© 0xE000ED04	Port
	► VTOR	0x1FF00000	ReadWrite	@ 0xE000ED08	SWD
CPU	DEMCR	0x00000000	ReadOnly	© 0xE000EDFC	Frequency (kHz) 4000
☱	AIRCR	0xFA050000	ReadWrite	@ 0xE000ED0C	Mode Not plug
swv	▶ SCR	0x00000000	ReadWrite	@ 0xE000ED10	morphog
	► CCR	0x00000200	ReadWrite	@ 0xE000ED14	Access port
REG	SHPR1	0x00000000	ReadWrite	@ 0xE000ED18	Reset mode Software reset
	SHPR2	0x0000000	ReadWrite	@ 0xE000ED1C ~	
			Apply	Read 🕕 CPU: RUNNING Reset 💌	Shared Disabled
	Log			Verbosity level 💿 1 💿 2 💿 3	Target voltage 3.27 V
	14:43:49 : Reset mode : Software reset			^ 🛃	Firmware version V2J30M19
	14:43:49 : Device ID : 0x437 14:43:49 : Revision ID : Rev Z				Firmware upgrade
	14:43:49 : UPLOADING OPTION BYTES DATA				
	14:43:49 : Address : 0x40023c1c				
	14:43:49 : Size : 136 Bytes 14:43:49 : Bank : 0x01				
	14:43:49 : Address : 0x1ff80000				Town of information
<u> </u>	14:43:49 : Size : 136 Bytes 14:43:49 : UPLOADING				Point NUCLEO L152PE
	14:43:49 : Size : 1024 Bytes				Device STM32L15xxE/STM32L162xE
-	14:43:49 : Address : 0x8000000 14:43:49 : Read progress:				Type MCU
$(\mathbf{\mathscr{O}})$	14:43:49 : Data read successfully 14:42:49 : Time elanced during the read operation in 00	0000.005			Device ID 0x437
	the read operation is: ou				Revision ID Rev Z
(?)					CPU Cortex-M3
				100% 🛞	

Note: The register viewer is available only through SWD/JTAG interfaces.

Register Viewer has as input a list of files containing the data describing the mapping of the core and STM32 registers ("svd" files).

2.14 Hard Fault analyzer

2.14.1 Description

The STM32CubeProgrammer Fault analyzer feature interprets information extracted from the Cortex-M based device to identify the reasons that caused a fault.

This information is visualized in the Fault analyzer window in GUI mode or in CLI mode. It helps to identify system faults occurring when the CPU is driven into a fault condition by the application software.

Possible detected fault exceptions:

- Hard Fault: default exception, can be triggered by an error during exception processing by Bus Fault, Memory Management Fault, or Usage Fault if their handler cannot be executed.
- Memory Management Fault: detects memory access violations to regions defined in the memory management unit (MPU), such as code execution from a memory region with read/write access only.
- Bus Fault: detects memory access errors on instruction fetch, data read/write, interrupt vector fetch, and register stacking (save/restore) on interrupt (entry/exit).
- Usage Fault: detects execution of undefined instructions, unaligned memory access for load/store multiple. When enabled, divide-by-zero and other unaligned memory accesses are detected.
- Secure Fault: provides information about security related faults for Cortex-M33 based devices.
- Note: Fault analyzer is available only for ST-LINK interfaces.



As shown in *Figure 65*, the Fault Analyzer window has five main sections.

- <complex-block><complex-block>
- Figure 65. Fault Analyzer window

- 1. Hard Faults details: indicates the type of occurred fault, locates the instruction and the called function addresses.
- 2. Bus Faults details: shows the status of bus errors resulting from instruction fetches and data accesses and indicates memory access faults detected during a bus operation. An address should be displayed on the BFAR text field.
- 3. Usage Faults details: contains the status for some instruction execution faults, and for data access.
- 4. Memory Management Faults details: indicates a memory access violation detected by the MPU. If this fault was triggered by a faulty address, access is displayed on the MMFAR text field.
- 5. CPU capture during exception: shows the CPU state when an exception was generated to have an overview for CPU registers and some helpful information.
 - c) NVIC position: indicates the number of the interrupt imposing the error, if it is "-" the interrupt/exception vector has no specific position.
 - d) Execution mode: indicates the operation mode Handler/Thread.
 - e) Stack memory region: indicates the used stack memory during the fault, Main or Process stack.



2.14.2 Example

Develop a simple application that generates a usage fault, set an instruction making a divide by zero (a non-permitted operation) in the main program function.

int a = 4, b = 0, c = 0;

c = a / b;

Open the Fault Analyzer window, press the "Start Analysis" button to start the fault detection algorithm, the reason of the error is displayed.

In this example, it displays "Hard Fault Detected", and the label "divide by zero (DIVBYZERO)" is highlighted with additional informations:

- Faulty instruction address: 0x8000FF0
- Faulty called function address: 0x8000D40, indicates the address calling the faulty instruction
- NVIC position: 0, window watchdog interrupt
- Execution mode: handler
- Stack memory region: main stack

Figure 66. Fault analyzer GUI view when Hard Fault is detected



2.14.3 Fault analyzer note

Fault analyzer can be unable to detect untracked faults not enabled by software.

The configuration and control register (CCR) controls the behavior of the Usage Fault for divide by-zero and unaligned memory accesses, and it is used mainly to control customizable fault exceptions.



31		10	9	8	7	6	5	4	3	2	1	0
	Reserved		STKALIGN	BFHFNMIGN	F	Reserve	d	DIV_0_TRP	UNALIGN_TRP	Reserved	USERSETMPEND	NONBASETHRDENA

Figure 67. CCR bits

The following bits of the CCR control the behavior of the Usage Fault:

- DIV_0_TRP: Enable Usage Fault when the processor executes an SDIV or UDIV instruction with a 0 divider.
 - 0 = do not trap divide by 0; a divide by 0 returns a quotient of 0.
 - 1 = trap divide by 0.
- UNALIGN_TRP: enable usage fault when a memory access to unaligned addresses is performed.
 - 0 = do not trap unaligned half-word and word accesses
 - 1 = trap unaligned half-word and word accesses; an unaligned access generates a usage fault.

Note that unaligned accesses with LDM, STM, LDRD, and STRD instructions always generate a usage fault, even when UNALIGN_TRP is set to 0.

STM32CubeProgrammer enables the required bits at the analysis startup, if no fault is detected an informative popup is displayed to indicate that you must reproduce the scenario and restart the analysis.

2.14.4 Secure Fault analyzer for Cortex-M33

STM32CubeProgrammer provides information about security related faults for Cortex-M33 based devices for both CLI and GUI interfaces.

A new field named "Secure Faults" is added to Fault Analyzer window when connecting a Cortex-M33-based device (such as an MCUs of the STM32L5 series).

The result analysis is based on Secure Fault Status Register (SFSR) settings and a fault is triggered if an error occurs:

- INVEP: this bit is set if a function call from the Non-secure state or exception targets a non-SG instruction in the Secure state. This bit is also set if the target address is a SG instruction, but there is no matching SAU/IDAU region with the NSC flag set.
- INVIS: this bit is set if the integrity signature in an exception stack frame is found to be invalid during the unstacking operation.
- INVER: set to 1 when returning from an exception in the Non-secure state.
- AUVIOL: attempt was made to access parts of the address space that are marked as Secure with NS-Req for the transaction set to Non-secure. This bit is not set if the violation occurred during lazy state preservation.
- INVTRAN: indicates that an exception was raised due to a branch not flagged as being domain crossing causing a transition from Secure to Non-secure memory.


- LSPERR: Indicates that an SAU or IDAU violation occurred during the lazy preservation of floating-point state.
- SFARVALID: this bit is set when the SFAR register contains a valid value.
- LSERR: indicates that an error occurred during lazy state activation or deactivation.
- SFAR: indicates the address value when a secure fault is raised.

2.15 Fill memory command

-fillmemory

Description: This command allows the user to fill memory with a given pattern from the chosen address.

Syntax: -fillmemory <start_address> [size=<value>] [pattern=<value>]
[datawidth=8|16|32]

<start_address>:</start_address>	Start address for write. The address 0x08000000 is used by default.
[size= <value>]:</value>	Size of the data to write.
[pattern= <value>]:</value>	The pattern value to write.
[datawidth=8 16 32]:	Filling data size, can be 8, 16, or 32 bits. The selected value by default is 8 bits.

- Example 1: STM32_Programmer_CLI.exe -c port=swd -fillmemory 0x08000000 size=0x10 pattern=0XAA datawidth=16 (*Figure 68*)
- Example 2: STM32_Programmer_CLI.exe -c port=swd -fillmemory 0x08000000 size= 0x10 pattern=0XCC datawidth=32 (*Figure 69*)



Figure 68. Example 1

Figure 69. Example 2





2.16 Fill memory operation

The user can open the Fill memory window from different sub-menus.

Figure 70	Sub-menu	displayed	from "R	ead" con	ho-box
rigule / u	. Sub-menu	uispiayeu		eau con	100-002

	Memory & File e	dition					
	Device memory	Open file +					
	Address 0x080000	000 🔻 Size	0x400	Data width 32-	bit 👻 Find I	Data Ox	Read 🔻
=	Address	0	4	8	с	ASCII	Save As
OB	0x08000000	00AA00AA	00AA00AA	00AA00AA	00AA00AA	a.a.a.a.a.a.a.	Fill memory
=	0x08000010	FFFFFFF	FFFFFFF	FFFFFFF	FFFFFFF	<u>ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ</u> ÿ	L
CPU	0x08000020	FFFFFFF	FFFFFFF	FFFFFFF	FFFFFFF	<u>ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ</u>	
	0x08000030	FFFFFFF	FFFFFFF	FFFFFFF	FFFFFFF	<u>ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ</u> ÿ	



	Memory & File	edition					
	Device memory	Open file +					
	Save	As Ctrl+S					
	Address C Oper	file Ctrl+O	0	Data width 3	2-bit 🔻 Find I	Data 0x	Read 🔹
	Addre: Close	tab Ctrl+C	4	8	с	ASCII	
OB	0x0800000 Close		AOOAA	00AA00AA	00AA00AA	a.a.a.a.a.a.a.	_
	0x0800001 Close	other tabs	FFFF	FFFFFFF	FFFFFFF	<u>ŸŸŸŸŸŸŸŸŸŸŸŸŸŸŸ</u> Ÿ	
CPU	0x0800002 Optio	on bytes Ctrl+B	FFFF	FFFFFFF	FFFFFFF	ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ	
	0x080000	omony Ctrl+M	FFFF	FFFFFFF	FFFFFFF	ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ	
swv	0x0800004	emory Ctri+M	FFFF	FFFFFFF	FFFFFFF	<u>ŸŸŸŸŸŸŸŸŸŸŸŸŸŸŸ</u> Ÿ	
	0x08000050	FFFFFFF	FFFFFFF	FFFFFFF	FFFFFFF	<u>ŸŸŸŸŸŸŸŸŸŸŸŸŸŸŸ</u> Ÿ	
REG	0x08000060	FFFFFFF	FFFFFFF	FFFFFFF	FFFFFFF	<u>ŸŸŸŸŸŸŸŸŸŸŸŸŸŸŸ</u> Ÿ	
BETA	0×08000070	FFFFFFFF	FEFEFEE	FEFEFEE	FFFFFFF	100000000000000000000000000000000000000	



≡	Memory 8	४ File edi	tion													
	Device men	nory Op	en file	+												
	Address	0x0800000	٣	Size	0x400		Data wid	th 32	-bit 🔹	Find D	ata	0x]	Read	•
	Addre	255	(0	4			8	(ļ	SCII		
OB	0x080000	0 00	04400	AA	00AA00A	A	00AA00	DAA	00AA00	AA	a.a.	^a . ^a . ^a .	a.a.a			Â
	0x0800001	10 F	FFFFF	FF	FFFFFF	F	FFFFF	FFF	FFFFFF	FF	ÿÿÿÿ	yyyyyyy	уууууу	ÿ		
CPU	0x0800002	20 F	FFFFF	FF	FFFFFF	F			FFFFFF	FF	ÿÿÿу	уууууу	уууууу	ÿ		
	0x0800003	30 F	FFFFF	FF	FFFFFF	F	FFFFF	Сору		FF	ÿÿÿÿ	yyyyyyy	уууууу	ÿ		
swv	0x0800004	40 F	FFFFF	FF	FFFFFF	F	FFFFF	Cut		FF	ÿÿÿÿ	ууууууу	уууууу	ÿ		
	0x080000	50 F	FFFFF	FF	FFFFFF	F	FFFFF	Paste		FF	ӰӰӰӰ	ÿÿÿÿÿÿÿ	уууууу	ÿ		
REG	0x080000	50 F	FFFFF	FF	FFFFFF	F	FFFFF	Fill	memory	F	ӰӰӰӰ	ууууууу	уууууу	ÿ		
	0x0800007	70 F	FFFFF	FF	FFFFFF	F	FFFFF			F	ӰӰӰӰ	yyyyyyy	уууууу	ÿ		~

Note:

In addition to sub-menus to display this window, user can open it directly by using the key combination "Ctrl+M".

After clicking on "Fill memory" option, a window is displayed so that the user can initialize the parameters of the operation (see *Figure 73*).



Prg Fill memor	у — 🗆	\times
Start address	0x08000000	
Size (Bytes)	0x10	
Data value	0xAA	
Filling data size	e 16 Bits 32 Bits	
	Fill me	emory

Figure 73. Parameters initialization

2.17 Blank check command

-blankcheck

Description: This command allows the user to verify that the STM32 flash memory is blank. If this is not the case, the first address with data is highlighted in a message.

Syntax: -blankcheck

Examples: STM32_Programmer_CLI.exe -c port=swd –blankcheck

Figure 74.	Example 1:	memory is	not blank at	address	0x08000014
i iguic / .		memory is	not blank at	uuui 033	070000014

ST-LINK SN :	0670FF554949677067035117
ST-LINK FW :	V2J30M20
Board :	STM32H743 -EV
Voltage :	3.23V
SWD freq :	4000 KHz
Connect mode:	Normal
Reset mode :	Software reset
Device ID :	0x450
Revision ID :	Rev Z
Device name :	STM32H7xx
Flash size :	2 MBytes
Device type :	MCU
Device CPU :	Cortex-M7
Flash memory b	blank checking
	100%
Warning: Flash	h memory is not blanck at 0x08000014.



Figure	75.	Exam	ole 1	1:	memory	is	blank
- igaio		EXAMIN					MIMIN

ST-LINK SN	: 0670FF554949677067035117	
ST-LINK FW	: V2J30M20	
Board	: STM32H743 -EV	
Voltage	: 3.22V	
SWD freq	: 4000 KHz	
Connect mod	e: Normal	
Reset mode	: Software reset	
Device ID	: 0x450	
Revision ID	: Rev Z	
Device name	: STM32H7xx	
Flash size	: 2 MBytes	
Device type	: MCU	
Device CPU	: Cortex-M7	
Flash memor	y blank checking	
		100%
Elach momon	, is black	

2.18 Blank check operation

The user can open the Fill memory window from different sub-menus.

	Memory & File e	dition					
	Device memory	Open file +					
.	Address 0x080000	000 🔻 Size	0x400	Data width 32-	bit 👻 Find D	ata Ox	Read 🔻
	Address	0	4	8	с	ASCII	Save As
OB	0x0800000	FAFAFAFA	FAFAFAFA	FAFAFAFA	FAFAFAFA	úúúúúúúúúúúúúúú	Fill memory
≝	0x08000010	FFFFFFF	FFFFFFF	FFFFFFF	FFFFFFF	<u>ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ</u> ÿ	Blank check
CPU	0x08000020	FFFFFFF	FFFFFFF	FFFFFFF	FFFFFFF	<u>ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ</u> ÿ	
	0x08000030	FFFFFFF	FFFFFFF	FFFFFFF	FFFFFFF	~~~~~	

F 1 1 1 1 1 1 1 1 1 1			- 12 1		((D 11)	and the second sec
FIGUID	/h	Siln-monii	dienia	Vod trom	"Road"	compo_poy
Inguic	10.	oup-menu	aispia		ILCUU	



	Memory	& File e	dition						
	Device me	mory)nen file	+					
		Save As	Ctr	I+S					
₽	Address	Open file	e Ctr	I+O	100	Data width	32-bit 👻 Fine	d Data 0x	Read 🛛 👻
	Addr	Close tal	o Ctr	l+C	4	8	с	ASCII	
OB	0x600000				DADADA	DADADADA	DADADADA	00000000000000000	2
	0x600000	Close ot	her tabs		FFFFF	FFFFFFF	FFFFFFF	<u>ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ</u> ÿ	
CPU	0x600000	Option b	ytes Ct	rI+B	FFFFF	FFFFFFF	FFFFFFF	<u>ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ</u>	
	0x600000	Fill mem	orv Ct	rl+M	FFFFF	FFFFFFF	FFFFFFF	<u> </u>	
swv	0x60000	Riank ch	ock Ct	d ± 1	FFFFF	FFFFFFF	FFFFFFF	<u> </u>	
	0x60000	Dialik Ch	eck Cl	II T L	FFFFF	FFFFFFF	FFFFFFF	<u>ŸŸŸŸŸŸŸŸŸŸŸŸŸŸŸ</u> Ÿ	
REG	0x600000	60	FFFFFFF	F	FFFFFFF	FFFFFFF	FFFFFFF	<u> </u>	
DEIA	0x600000	70	FFFFFFF	F	FFFFFFF	FFFFFFF	FFFFFFF	ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ	

Figure 77. Sub-menu displayed with right click on "Device memory" tab



	Memory & File e	edition					
	Device memory	Open file +					
	Address 0x08000	000 💌 Size	0x400	Data width	32-bit 💌 Find D	Ox Ox	Read 🔹
	Address	0	4	8	с	ASCII	
OB	0x08000000	FAFAFAFA	FAFAFAFA	FAFAFAFA	FAFAFAFA	ບໍ່ຜໍ່ຜໍ່ຜໍ່ຜໍ່ຜໍ່ຜໍ່ຜໍ່ຜໍ່ຜໍ່ຜໍ່ຜໍ່ຜໍ່ຜໍ	Â
	0x08000010	FFFFFFF	FFFFFFF	FFFFFFF	FFFFFFF	ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ	
CPU	0x08000020	FFFFFFF	FFFFF Copy	FF	FFFFFFF	ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ	
	0x08000030	FFFFFFF	FFFFFF Cut	FF	FFFFFFF	ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ	
swv	0x08000040	FFFFFFF	FFFFFF Paste	FF	FFFFFFF	<u>ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ</u> ÿ	
	0x08000050	FFFFFFF	FFFFFF Fill	FF	FFFFFFF	<u>ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ</u> ÿ	
REG	0x08000060	FFFFFFF	FFFFF Blank	check	FFFFFFF	<u>ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ</u> ÿ	
L BEIA	0x08000070	FFFFFFF	FFFFF	F	FFFFFFF	ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ	
	0.0000000			eccecce			~

In addition to sub-menus to display this window, user can launch the operation directly by using the key combination Ctrl+L.

After clicking on "Blank check" sub-menu, the process starts to verify that the STM32 flash memory is blank. If the flash memory is not blank, the first address with data is highlighted in a message, as shown in *Figure 79*.

The expected results are shown in figures 80 and 81.



∎	Memory & F	ile editio	on		<u> </u>								
	Device memor	y Open	n file	+									
	Address 0x0	8000000	▼ Si	ize	0x400	Data width	32-bit	*	Find D	ata Ox		Read	*
	Address		0		4	8		c			ASCII		
OB	0x08000000	FFF	FFFFF		FFFFFFF	FFFFFFFF	F	FFFFFF	F	<u> </u>	yyyyy		_
	0x08000010	FFF	FFFFF		FFFFFFF	FFFFFFF	F	FFFFFF	F	<u>ÿÿÿÿÿÿÿÿÿÿÿÿ</u> ÿÿ	ÿÿÿÿ		
CPU	0x08000020	FFF	FFFFF		FFFFFFF	FFFFFFFF	F	FFFFFF	F	<u>ÿÿÿÿÿÿÿÿÿÿÿÿ</u> ÿ	ÿÿÿÿ		
	0x08000030	FFF	FFFFF		FFFFFFF	FFFFFFFF	F	FFFFFF	F	<u> </u>	ÿÿÿÿ		
swv	0x08000040	FFF	FFFFF		FFFFFFF	FFFFFFF	F	FFFFFF	F	<u> </u>	YYYYY		
	0x08000050	FFF	FFFFF		FFFFFFF	FFFFFFF	F	FFFFFF	F	<u> </u>	УУУУУ У		
REG	0x08000060	FFF	FFFFF		FFFFFFF	FFFFFFF	F	FFFFFF	F	<u>ŷŷŷŷŷŷŷŷŷŷŷŷ</u>	yyyyy		
	0x08000070	FFF	FFFFF		FFFFFFF	FFFFFFFF	F	FFFFFF	F	<u>ÿÿÿÿÿÿÿÿÿÿÿÿ</u> ÿ	ÿÿÿÿ		~
	< (>
	Log									Verbosity l	evel 💿 1	2	3
	1121:39 : Come 1121:39 : Reset r 1121:39 : Revice 1121:39 : Revice 1121:40 : UPLOA 1121:40 : Bank 1121:40 : Size 1121:40 : Size 1121:40 : Addre 1121:40 : Size 1121:40 : Read p 1121:40 : Time e 1121:40 : Time e 1121:48 : Flash n	IC mode: Hord node : Hard ID : 0x450 n ID : Rev Z DING OPTIC : 0x00 :ss : 0x55 : 308 By DING : 1024 B tss : 0x80 rogress: ad successf lapsed durir nemory blan	Invig Invig DN BYTES 200201c ytes Bytes 000000 fully ng the rea k checkir	et S DATA : ad opera	 stion is: 00:00:00	.008							
?													~ @
													~ 🛇

Figure 79. First address with data



≡	Memory a	& File ec	lition			•			,				
	Device me	mory C	pen file	+									
	Address	0x080000	• 00	Size	0x400	Data width	32-b	it 🔻	Find D	Data 0x		Read	.
	Addr	ess		0	4	8		c			ASCII		
OB	0x080000	00	FFFFF	FF	FFFFFFF	FFFFFFF		FFFFFF	FF	<u> </u>	ÿÿÿÿ		<u>^</u>
≓	0x080000	10	FFFFF	FF	FFFFFFF	FFFFFFF		FFFFFF	F	<u> </u>	ÿÿÿÿ		
CPU	0x080000	20	FFFFF	FF	FFFFFFF	FEFFFFFF		FFFFFFF	F		ininini Vicini		
	0x080000	30	FFFFF	FF	FFFFFFF	Pro Messag	je					×	
swv	0x080000	40	FFFFF	FF	FFFFFFF								
	0x080000	50	FFFFF	FF	FFFFFFF		Flash n	nemory is	blank.				
REG	0x080000	60	FFFFF	FF	FFFFFFF								
BETA	0x080000	70	FFFFF	FF	FFFFFFF						ОК		~
	< [>
	Log									Verbosity le	vel 💿	1 2	<u>3</u>
	10:43:53 : 0 16:43:53 : 0 16:43:55 : 0 16:43:55 : 0 16:43:55 : 0 16:43:55 : 0 16:43:55 : 0 16:43:55 : 0 16:43:55 : 0 16:43:55 : 0 16:43:55 : 0 16:43:55 : 0 16:43:55 : 0 16:43:55 : 0 16:43:55 : 0 16:43:55 : 0 16:43:55 : 0 16:45 : 0	sec mode :: vvice ID : 0, vvision ID : R PLOADING C lank :: 0 videress : vice : 30 PLOADING vice : 10 laderess : vice : 10 vice	Hardwan (450 ev Z)PTION B (x00 0x52002 08 Bytes 0x80000 c cessfully during th blank ch is blank.	YTES DAT 01c 00 e read op: ecking	A eration is: 00:00:00.00	27							 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4<
?			_				_		_			1	00% 🗵

Figure 80. Example 1: memory is blank



	Memory	& File e	dition			·								
	Device m	emory	Open file	+										
	Address	0x080000	• 00	Size	0x400	Data width	32-	bit 🔻	Find D	ata	0x		Read	•
<u> </u>	Add	iress	C)	4	8		c				ASCII		
OB	0x08000	000	FFFFFF	FF	FFFFFFF	FFFFFFF		FFFFFF	FF	ÿÿÿÿ	, , , , , , , , , , , , , , , , , , ,	, VYYY		<u>^</u>
╘	0x08000	010	FFFFFF	FF	FFFFFFF	FFFFFFF		FFFFFF	FF	ÿÿÿÿ		, VYYY		
CPU	0x08000	020	FFFFF	FF	00DADACC	FFFFFFF		FFFFFF	FF	ÿÿÿÿ	túú.ÿÿÿÿý	, УУУУ		
=	0x08000	030	FFFFFF	FF	FFFFFFF	Freezer	_		-					
swv	0x08000	040	FFFFFF	FF	FFFFFFF	F F Pro War	ming						×	
	0x08000	050	FFFFFF	FF	FFFFFFF	FF	14/-	mina: Elas	h more c	nuis no	t blanck et 0	w00000004		
REG	0x08000	060	FFFFFF	FF	FFFFFFF		vva	rning: Flas	n memo	ry is no	t blanck at u	X08000024.		
BETA	0x08000	070	FFFFF	FF	FFFFFFF	FF							_ [~
	< [OK		>
	Log					L					erbosity le	vel 0 1		3
	16:46:51 : [16:46:51 :] 16:46:51 : [16:46:51 : [16:46:51 :] 16:46:51 : [16:46:51 :] 16:46:51 : [Data downio UPLOADING Size : 4 Address Read progress Data read suc Fime elapsed UPLOADING Size : 1 Address Read progress Data read suc Flash memory Warning: Flas	We are success We are successfully We are successfully We are successfully We are successfully We are an are successfully We are an are successfully We are an are successfully We are an are successfully We are an are successfully We are an are successfully We are an are successfully We are an are successfully We are an are an are an are an are an are an We are an are an are an are an are an are an We are an are an are an are an are an are an We are an are an are an are an are an are an We are an are an are an are an are an are an We are an are an are an are an are an are an are an are an We are an are an are an are an are an are an are an are an are an We are an are an are an are an are an are an are an are an are an We are an are an are an are an are an are an are an are an are an We are an are an are an are an are an are an are an are an are an are an We are an are an are an are an are an are an are an are an are an are an We are an ar	4 read ope 0 <u>read ope</u> cking is not bla	ration is: 00:00:00.0 rration is: 00:00:00.0 nck at 0x08000024.	207								🖏
?						-							1	00% 🗵

Figure 81. Example 2: memory is not blank

2.19 Compare flash memory with file

Description: Compares the MCU device memory content with a binary, hex, srec, elf, out and axf file. The difference is shown in red in the file and in the flash memory panel.

The user can open the comparison window from different sub-menus.

Memo	ry & File e	dition								
Device	memory	Open file	+							ST-LINK
Address	0x08000	• 000	Size	0x400	Data width	32-bit 🔻 Fi	nd Data 0x		Read 👻	Serial numb
4	ddress	0		4	8	с		ASCII	Save As	
0x080	00000	0000FFF	F	FFFFFFF	FFFFFFF	FFFFFFF	ÿÿÿÿÿÿÿÿÿÿÿÿÿ	ÿ	Fill memory	
0×080	00010	FFFFFF	F	FFFFFFF	FFFFFFF	FFFFFFFF	<u> </u>	ÿ	Blank check	
PU 0x080	00020	FFFFFF	F	FFFFFFF	FFFFFFF	FFFFFFF	<u> </u>	ÿ	Compare memor	v with file
0x080	00030	FFFFFF	F	FFFFFFF	FFFFFFF	FFFFFFF	<u> </u>	ÿ		TARCOSSEDIO
WV 0x080	00040	FFFFFF	F	FFFFFFF	FFFFFFF	FFFFFFF	<u> </u>	ÿ		
0x080	00050	FFFFFF	F	FFFFFFF	FFFFFFF	FFFFFFF	<u>ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ</u>	ÿ		Reset mod
EG 0x080	00060	FFFFFF	F	FFFFFFF	FFFFFFF	FFFFFFF	<u> </u>	ÿ		Shared
0x080	00070	FFFFFF	F	FFFFFFF	FFFFFFF	FFFFFFFF	<u> </u>	ÿ		
0x080	00080	FFFFFF	F	FFFFFFF	FFFFFFF	FFFFFFFF	<u> </u>	ÿ	~	External loa
<									>	Target vol

Figure 82. Sub-menu displayed from "Read" combo-box



	Memory	/ & File e	dition							
	Device m	emory (Open file +							
	Address	Save As		Ctrl+S		Data width	a bit	Data a		
*	Address	Open file		Ctrl+O		Data width	2-bit • Find	Data Ox	Read	· · ·
<u></u>	Ad	Close tab		Ctrl+C		8	с	AS	CII	
OB	0x0800				۶F	FFFFFFF	FFFFFFF	ÿÿ··ÿÿÿÿÿÿÿÿÿÿÿÿÿÿ		Â
	0x0800	Close othe	ar tabs		F	FFFFFFF	FFFFFFF	<u>ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ</u> ÿÿ		
CPU	0x0800	Close office			۶F	FFFFFFF	FFFFFFF	<u>ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ</u> ÿ		
	0x0800	Option by	tes	Ctrl+B	F	FFFFFFF	FFFFFFF	<u>ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ</u>		
swv	0x0800	Fill memor	У	Ctrl+M	F	FFFFFFF	FFFFFFF	<u> </u>		
믇	0x0800	Blank cheo	:k	Ctrl+L	F	FFFFFFF	FFFFFFF	<u> </u>		
REG	0x0800	Compare r	memory with file	Ctrl+T	F	FFFFFFF	FFFFFFF	<u> </u>		
- DETA	0x08004				F	FFFFFFF	FFFFFFF	<u>ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ</u> ÿ		
	0x08000	080	FFFFFFF	FFFFF	FF	FFFFFFF	FFFFFFF	<u> </u>		~
	<									

Figure 83. Sub-menu displayed with right click on "Device memory" tab



Figure 84. Sub-menu displayed with right click on the cell of grid

Device memory	Open file +					
Address 0x080	00000 🔻 Size	0x400	Data width 32	·bit • Find	i Data Ox	Read
Address	0	4	8	с	ASCII	
0x0800000	0000FFFF	FFFFFFF		FFFFFFF	ÿÿ··ÿÿÿÿÿÿÿÿÿÿÿÿÿ	
0x08000010	FFFFFFF	FFFFFFF	FFFFFF CODV		<i>ſŸŸŸŸŸŸŸŸ</i> ŸŸ	
U 0x08000020	FFFFFFF	FFFFFFF	FFFFFF Cut		<i>ĬŸŸŸŸŸŸŸŸ</i> ŸŸ	
0x08000030	FFFFFFF	FFFFFFF	FFFFF		<i>ĬŸŸŸŸŸŸŸ</i> ŸŸ	
0x08000040	FFFFFFF	FFFFFFF	FFFFFF Fill .	iemory	<i>ĬŸŸŸŸŸŸŸŸ</i> ŸŸ	
0x08000050	FFFFFFF	FFFFFFF	FFFFF	check	<i>ĬŸŸŸŸŸŸŸ</i> ŸŸ	
0x08000060	FFFFFFF	FFFFFFF	FFFFF	CHECK	th file VYYYYYYY	
0x08000070	FFFFFFF	FFFFFFF	FFFF	re memory wit	ÿÿÿÿÿÿÿÿÿ	
0x08000080	FFFFFFF	FFFFFFF	FEFFFFF	FEFEFFF	~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~	



≡	Memory & File e	dition					
	Device memory	Open file +					
	Address 0x080000	000 🕶 Si 🕻	Dpen file Dpen memory tab	idth 32	-bit 👻 Find I	Data Ox Read	•
	Address	0		th file 8	с	ASCII	
OB	0x08000000	0000FFFF	compare memory wi	FFF	FFFFFFF	ÿÿ··ÿÿÿÿÿÿÿÿÿÿÿÿÿ	^
	0x08000010	FFFFFFF	FFFFFFF	FFFFFFF	FFFFFFF	<u>ŸŸŸŸŸŸŸŸŸŸŸŸŸ</u> ŸŸ	
CPU	0x08000020	FFFFFFF	FFFFFFF	FFFFFFF	FFFFFFF	<u>ŸŸŸŸŸŸŸŸŸŸŸŸŸŸ</u>	
	0x08000030	FFFFFFF	FFFFFFF	FFFFFFF	FFFFFFF	ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ	
swv	0x08000040	FFFFFFF	FFFFFFF	FFFFFFF	FFFFFFF	<u>ŸŸŸŸŸŸŸŸŸŸŸŸŸŸ</u>	
	0x08000050	FFFFFFF	FFFFFFF	FFFFFFF	FFFFFFF	<u>ŸŸŸŸŸŸŸŸŸŸŸŸŸŸ</u> ŸŸ	
REG	0x08000060	FFFFFFF	FFFFFFF	FFFFFFF	FFFFFFF	<u>ŸŸŸŸŸŸŸŸŸŸŸŸŸŸ</u> Ÿ	
BETA	0x08000070	FFFFFFF	FFFFFFF	FFFFFFF	FFFFFFF	<u>ŸŸŸŸŸŸŸŸŸŸŸŸŸŸ</u> Ÿ	
	0x08000080	FFFFFFF	FFFFFFF	FFFFFFF	FFFFFFF	<u>ŸŸŸŸŸŸŸŸŸŸŸŸŸŸ</u>	~
	< [>



Device m	emory H74	Save As	Ctrl+S				
Address	0x8000000	Verify	Ctrl+V	vidth 🔅	32-bit 💌 Find	d Data Ox	Download 🔻
Add	dress	Download	Ctrl+D	8	с	ASCII	
3 0x08000	0000 3	Open file	Ctrl+O	3034	0A324630	:020000040800F2.	
0×08000	010 3	Close tab	Ctrl+C	4130	30303041	:20006000AAAA000	
U 0x08000	0020 4	Address 0x8000000		4646	46464646	0FFFFFFFFFFFFFFF	
0x08000	0030 4	C1		4646	46464646	FFFFFFFFFFFFFFF	
V 0x08000	0040 4	Close tab		4645	46464646	FFFFEEEEFFFFFFF	
0x08000	0050 4	Close other tabs		3646	3030323A	FFFFFFF6A.:200	
G 0x08000	0060 3	Option bytes	Ctrl+B	4646	46464646	08000FFFFFFFFFFF	
0x08000	0070 4	Fill memory	Ctrl+M	4646	46464646	FEEEEFFFFFFFFFFF	
000000	1000 4	Blank check	Ctrl+L	ACAC	ACACACAC		
Log	File : 174	Compare memory with f	ile Ctrl+T		Live U	pdate Verbosity level 🦲	1 2 3
15:06:02 :	Size : 512 Address : 0	Compare two files	Ctrl+F				^ <
15:06:02 :	Erasing memory	corresponding to segment 0:					E E

Figure 86. Sub-menu displayed with right click on the opened file tab

Figure 87. Sub-menu displayed from "Download" combo-box displayed in file tab

≡	Memory & File	edition							Connecte	ed
	Device memory	H743.hex × +					ST-	LINK	 Disconn 	iect
	Address 0x80000	000 👻 Size	0x200	Data width	32-bit 👻 Find	d Data Ox	Download 👻	ST-LIN	K configuration	
	Address	0	4	8	с	ASCII	Read			
OB	0x08000000	3032303A	30303030	30383034	0A324630	:020000040800F2.	Save As		SWD	× .
	0x08000010	3030323A	30303630	41414130	30303041	:20006000AAAA000	Verify	(kHz)	4000	Υ.
CPU	0x08000020	46464630	46464646	46464646	46464646	OFFFFFFFFFFFFFF	Address		Hot plug	-
	0x08000030	46464646	46464646	46464646	46464646	FFFFFFFFFFFFFFF	Address 0x8000000			
swv	0x08000040	46464646	45454546	46464645	46464646	FFFFFEEEEFFFFFFF	Compare memory with f	ile	0	Υ.
	0x08000050	46464646	46464646	0A413646	3030323A	FFFFFFFF6A.:200	Compare two files	-	Hardware reset	
REG	0x08000060	30303830	46464630	46464646	46464646	08000FFFFFFFFFFF	Shar	ed	Disabled	
BETA	0x08000070	45454546	46464645	46464646	46464646	FEEEEFFFFFFFFFFF	0.5			
	000000000	ACACACAC	ACACACAC	ACACACAC	ACACACAC	*********	V Deb			
	Log				Live U	pdate Verbosity level 🔘 1	1 2 3 Exte Targ	rnal loader et voltage	- 3.22 V	

In addition to sub-menus to display this window, the user can launch the operation directly by using the key combination Ctrl+T.

Example 1: Difference between internal flash memory and binary file

Houress Uso	* Si	ze 0x200	Data widt	h 32-bit •	Find Data 0x						Read
	5	Memory device	{ Address range:	[0x08000000 0)	<8000200])				File : F746G.bin	, Size: 512 Bytes	
Address	0	4	8	с	ASCII	0	4	8	с	ASCII	
0x0800000	0000FFFF	FFFFFFFF	FFFFFFFF	FFFFFFFF	yy yyyyyyyyyyyyy	F7E9F9AA	08160655	FFFFFFFF	00000000	*ůé+Uÿÿÿÿ	
0x08000010	FFFFFFF	FFFFFFF	FFFFFFF	FFFFFFF	222222222222222222222222222222222222222	7FFF8000	80007FFF	FF80FFFF	007F0000	ÿ.ÿÿÿ.ÿ	
0x08000020	FFFFFFFF	FFFFFFF	FFFFFFFF	FFFFFFF	3333333333333333333	FF80FFFF	007F0000	FFFEFF00	000100FF	ýý.ýýþýý	
0x08000030	FFFFFFFF	FFFFFFFF	FFFFFFFF	FFFFFFFF	<u> </u>	FFFFFFFF	FFFFFFFF	FFFFFFFF	FFFFFFFF	yyyyyyyyyyyyyyyyy	
0x08000040	FFFFFFFF	FFFFFFFF	FFFFFFFF	FFFFFFFF	9999999999999999999	FFFFFFFF	FFFFFFF	FFFFFFFF	FFFFFFFF	99999999999999999999999	
0x08000050	FFFFFFFF	FFFFFFF	FFFFFFFF	FFFFFFF	עררררררררר	FFFFFFFF	FFFFFFF	FFFFFFF	FFFFFFFF	222222222222222222222222222222222222222	
0x08000060	FFFFFFFF	FFFFFFFF	FFFFFFFF	FFFFFFFF	VYYYYYYY Marning			FFFFFF	FFFFFFFF	<u> </u>	
0x08000070	FFFFFFFF	FFFFFFFF	FFFFFFFF	FFFFFFFF	19999999			FFFFFFF	FFFFFFFF	yyyyyyyyyyyyyyyyy	
0x08000080	FFFFFFFF	FFFFFFFF	FFFFFFFF	FFFFFFFF	yyyyyyy 🔥 Warning: First diffe	ence found at 0x0	8000000.	FFFFFFF	FFFFFFFF	333333333333333333	
0x0800090	FFFFFFFF	FFFFFFFF	FFFFFFFF	FFFFFFFF	99999999		1	FFFFFFF	FFFFFFFF	yyyyyyyyyyyyyyyyy	
0x0800080x0	FFFFFFFF	FFFFFFFF	FFFFFFFF	FFFFFFFF	333333333		OX	FFFFFFF	FFFFFFFF	yyyyyyyyyyyyyyyy	
0x0800080x0	FFFFFFFF	FFFFFFF	FFFFFFF	FFFFFFFF	20000000		-	FFFFFF	FFFFFFF	222222222222222222222222222222222222222	
0x080000C0	FFFFFFFF	FFFFFFFF	FFFFFFFF	FFFFFFFF	<u> </u>	- FFFFFFFF	FFFFFFFF	FFFFFFFF	FFFFFFFF	xxxxxxxxxxxxxxxxxx	

Figure 88. Data width: 32 bits



Device memory	Open	file Co					6G.bin X	+													
ddress 0x0		• Size	0x20	00	Data	a width	16-bit	+	Find Data 0x											Read	•
		м	emory d	levice { A	ddress r	ange: (0:	x080000	00 0x800	10200] }						Fil	e : F7460	i.bin , Sia	te: 512 Bytes			
Address	0	2	4	6	8	A	с	E	ASCII	0	2	- 4	6	8	A	с	E		ASCII		
x08000000	FFFF	F7F9	FFFF	FFFF	FFFF	FFFF	FFFF	FFFF	yyù+yyyyyyyyyyyyy	≏ F9AA	F7E9	0655	0816	FFFF	FFFF	0000	0000	*ùé+Uÿÿÿÿ			
x08000010	FFFF	FFFF	FFFF	FFFF	FFFF	FFFF	FFFF	FFFF	<u> </u>	8000	7FFF	7FFF	8000	FFFF	FF80	0000	007F	ÿ.ÿÿÿ.ÿ			
x08000020	FFFF	FFFF	FFFF	FFFF	FFFF	FFFF	FFFF	FFFF	yyyyyyyyyyyyyyyyy	FFFF	FF80	0000	007F	FF00	FFFE	00FF	0001	ÿÿ.ÿÿþÿÿ			
x08000030	FFFF	FFFF	FFFF	FFFF	FFFF	FFFF	FFFF	FFFF	yyyyyyyyyyyyyyyyy	FFFF	FFFF	FFFF	FFFF	FFFF	FFFF	FFFF	FFFF	<i>уууууууууууууууууу</i> уууу			
x08000040	FFFF	FFFF	FFFF	FFFF	FFFF	FFFF	FFFF	FFFF	yyyyyyyyyyyyyyyyy	FFFF	FFFF	FFFF	FFFF	FFFF	FFFF	FFFF	FFFF	YYYYYYYYYYYYYYYYYY			
x08000050	FFFF	FFFF	FFFF	FFFF	FFFF	FFFF	FFFF	FFFF	<u> </u>	FFFF	FFFF	FFFF	FFFF	FFFF	FFFF	FFFF	FFFF	<i><i>уууууууууууууууу</i>уууу</i>			
x08000060	FFFF	FFFF	FFFF	FFFF	FFFF	FFFF	FFFF	FFFF	yyyyy	1111	1110	1000	erer.	FFFF	FFFF	FFFF	FFFF	>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>			
x08000070	FFFF	FFFF	FFFF	FFFF	FFFF	FFFF	FFFF	FFFF	yyyyy				×	FFFF	FFFF	FFFF	FFFF	YYYYYYYYYYYYYYYYYY			
x08000080	FFFF	FFFF	FFFF	FFFF	FFFF	FFFF	FFFF	FFFF	YYYYY	nce found	**	00000		FFFF	FFFF	FFFF	FFFF	<u> </u>			
x08000090	FFFF	FFFF	FFFF	FFFF	FFFF	FFFF	FFFF	FFFF	yyyyy	ince inverse	er 0.000			FFFF	FFFF	FFFF	FFFF	<u> </u>			
x080000A0	FFFF	FFFF	FFFF	FFFF	FFFF	FFFF	FFFF	FFFF	¥¥¥¥¥¥			_		FFFF	FFFF	FFFF	FFFF	>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>			
×08000060	FFFF	FFFF	FFFF	FFFF	FFFF	FFFF	FFFF	FFFF	YYYYY				OK	FFFF	FFFF	FFFF	FFFF	>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>			
x080000C0	FFFF	FFFF	FFFF	FFFF	FFFF	FFFF	FFFF	FFFF	¥¥¥¥¥¥¥¥¥¥¥¥¥¥¥¥¥¥¥¥		11111	11111	****	FFFF	FFFF	FFFF	FFFF	<u> </u>			



Memory & File	e edi	tion																													
Device memory	Op	en fil	e	Com	pariso	on Di	evice	mem	ory \	ls F7	46G.	x nic	ł																		
Address 0x0			S	ze	0x20	0		Da	ta wi	idth	8-	bit	Ŧ	Fin	d Data Ox																Read 🔹
				Merr	iory d	levic	e (Ac	dres	s ran	ge: ((0x08	0000	00 0	(8000	200] }											ile : F	7460	i.bin ,	Size:	: 512 Bytes	
Address	0	1	2	3	4 5	6	5 7	8	9	A	в	с	D	E F	ASCII	0	1	2	3	4	5	6 7	8	9	A	B	c (E	F	ASCII	
0x0800000	FF	FF	F9	F7 F	FF	FF	FFF	FF	FF	FF	FF	FF F	FFF	F FF	ÿÿù+ÿÿÿÿÿÿÿÿÿÿÿÿ	AA	F9	E9	F7	55 (06 1	.6 08	B FF	FF	FF	FF 0	0 0	0 00	00	≇ùé÷Uÿÿÿÿ	^
0x08000010	FF	FF	FF	FFF	F FF	FFF	F FF	FF	FF	FF	FF	FF F	FFF	F FF	yyyyyyyyyyyyyyyyy	00	80	FF	7F	FF	7F 0	0 80) FF	FF	80	FF 0	0 0	0 7F	00	ÿ.ÿÿÿ.ÿ	
0x08000020	FF	FF	FF	FF F	F F	F FF	F FF	FF	FF	FF	FF	FF F	FFF	F FF	yyyyyyyyyyyyyyyyy	FF	FF	80	FF	00	00 7	'F 00	00	FF	FE	FF F	F 0	0 01	00	ÿÿ.ÿÿþÿÿ	
0x08000030	FF	FF	FF	FF F	F FF	FFF	F FF	FF	FF	FF	FF	FF F	FF	F FF	xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx	FF	FF	FF	FF	FF I	FF F	FFF	FF	FF	FF	FF F	FF	F FF	FF	yyyyyyyyyyyyyyyy	
0x08000040	FF	FF	FF	FF F	F FF	F	F FF	FF	FF	FF	FF	FF F	FF	F FF	<u> </u>	FF	FF	FF	FF	FF I	FF F	FFF	FF	FF	FF	FF F	FF	F FF	FF	<u> </u>	
0x08000050	FF	FF	FF)	FF F	FF	F FF	FFF	FF	FF	FF	FF	FF F	FF	F FF	yyyyyyyyyyyyyy	FF	FF	FF	FF	FF	FF F	FFF	FF	FF	FF	FF F	FF	FFF	FF	<i><i>уууууууууууууууу</i>ууууу</i>	
0x08000060	FF	FF	FF)	FF F	F FF	FFF	F FF	FF	FF	FF	FF	FF F	FF	F FF	ÿÿÿÿ 🌆 Warning							×	FF	FF	FF	FF F	FF	F FF	FF	<u> </u>	
0x08000070	FF	FF	FF	FF F	F FF	FFF	F FF	FF	FF	FF	FF	FF F	FF	F FF	<u> </u>			_					FF	FF	FF	FF F	FF	F FF	FF	<u> </u>	
0x08000080	FF	FF	FF)	FF F	F FF	FF	F FF	FF	FF	FF	FF	FF F	FF	F FF	Warning: First differen	ce fo	und	at 0x	0800	0000.			FF	FF	FF	FF F	FF	F FF	FF	<u> </u>	
0x08000090	FF	FF	FF	FF F	F F	FFF	F FF	FF	FF	FF	FF	FF F	FF	F FF	ÿÿÿj 🚢								FF	FF	FF	FF F	FF	F FF	FF	<u> </u>	
0x080000A0	FF	FF	FF)	FF F	F FF	F FF	F FF	FF	FF	FF	FF	FF F	FF	F FF	<u>9999</u>						OK		FF	FF	FF	FF F	FF	F FF	FF	yyyyyyyyyyyyyyyyy	
0x080000B0	FF	FF	FF	FF F	FF	FFF	FFF	FF	FF	FF	FF	FF F	FF	F FF	ÿÿÿj	_	_	_	_	_			FF	FF	FF	FF F	FF	FFF	FF	<u> </u>	
0x080000C0	FF	FF	FF	FF F	FF	F	FFF	FF	FF	FF	FF	FF F	FF	F FF	xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx	FF	FF	FF	FF	FF I	FF F	FFF	FF	FF	FF	FF F	FF	FFF	FF	yyyyyyyyyyyyyyyy	~
															>																>



UM2237

Example 2: Difference between external flash memory and hex file

Memory & File	e edition										
Device memory	Open file	Comparison Dev	rice memory Vs F	746G.bin Com	barison Device memory Vs 6.hex × 🔸						
Address 0x600	00000 💌 Sia	e 0x100	Data width	32-bit 💌	Find Data 0x	•					Read 💌
	N	emory device (Address range:	0x6000000 0x6	0000100] }			File : 6.hex	, Address range:	[0x6000000 0x60000100]	
Address	0	4	8	с	ASCII	0	4	8	с	ASCII	
0x60000000	DADADADA	DADADADA	DADADADA	DADADADA	000000000000000000000000000000000000000	81080240	81060003	00000080	00000009	G±±	<u>^</u>
0x60000010	FFFFFFF	FFFFFFF	FFFFFFFF	FFFFFFF	yyyyyyyyyyyyyyyy	00000001	00000100	00000004	0000001F		
0x6000020	FFFFFFF	FFFFFFF	FFFFFFFF	FFFFFFF	yyyyyyyyyyyyyyyy	00000011	00000010	01234567	89ABCDEF	gE#.i1«.	
0x6000030	FFFFFFF	FFFFFFF	FFFFFFFF	FFFFFFF	yyyyyyyyyyyyyyyy	01234567	89ABCDEF	00000104	00000001	gE#.i1<	
0x60000040	FFFFFFF	FFFFFFF	FFFFFFFF	FFFFFFFF	yyyyyyyyyyyyyyyyy	00000001	00000105	0000001	0000001		
0x60000050	FFFFFFF	FFFFFFF	FFFFFFFF	FFFFFFF	yyyyyyy <u>yooooooo</u>	00000001	00000001	90000001	00000171	q	
0x6000060	FFFFFFF	FFFFFFF	FFFFFFFF	FFFFFFF	yyyyyyy Pro Warning		×	0000162	0000001	b	
0x6000070	FFFFFFFF	FFFFFFFF	FFFFFFFF	FFFFFFFF	YYYYYYY	an farrad at Def		0000004	00000004		
0x6000080	FFFFFFF	FFFFFFF	FFFFFFF	FFFFFFF	yyyyyyy	ce round at oxeor	00000.	8464753	50595243	SGFXCRYP	
0x6000090	FFFFFFFF	FFFFFFFF	FFFFFFFF	FFFFFFFF	yyyyyyyy			1080003	00000080	128.0±±	
0x600000A0	FFFFFFF	FFFFFFF	FFFFFFFF	FFFFFFF	yyyyyyy		OK	0000100	00000004		
0x6000080	FFFFFFF	FFFFFFFF	FFFFFFFF	FFFFFFFF	<u> </u>	00000011	00000011	8000000	00000000		
0x600000C0	FFFFFFF	FFFFFFFF	FFFFFFFF	FFFFFFFF	yyyyyyyyyyyyyyy	00000000	00000104	00000001	00000000		~

Figure 91. Data width: 32 bits



Device memory	Open	file Co	ompariso	n Device	e memoi	y Vs F74	6G.bin		son Device memory Vs 6.hex × +											
ddress 0x600	00000	▼ Size	0x10	0	Data	width	16-bit	*	ind Data Ox										Rea	ed 💌
		Me	mory de	vice (Ad	ddress ra	inge: (0x	6000000	0 0x600	0100] }					File :	6.hex , A	ddress ra	ange: (0)	60000000 0x60000100]		
Address	0	2	4	6	8	A	с	E	ASCII	0	2	4	6	8	A	с	E		ASCII	
x60000000	DADA	DADA	DADA	DADA	DADA	DADA	DADA	DADA	0000000000000000000000000000000	0240	B10B	0003	B10B	0080	0000	0009	0000	0±±		
x60000010	FFFF	FFFF	FFFF	FFFF	FFFF	FFFF	FFFF	FFFF	yyyyyyyyyyyyyyyy	0001	0000	0100	0000	0004	0000	001F	0000			
c60000020	FFFF	FFFF	FFFF	FFFF	FFFF	FFFF	FFFF	FFFF	yyyyyyyyyyyyyyy	0011	0000	0010	0000	4567	0123	CDEF	89AB	gE#.i1«.		
x6000030	FFFF	FFFF	FFFF	FFFF	FFFF	FFFF	FFFF	FFFF	yyyyyyyyyyyyyyyy	4567	0123	CDEF	89AB	0104	0000	0001	0000	gE#.i1<		
<6000040	FFFF	FFFF	FFFF	FFFF	FFFF	FFFF	FFFF	FFFF	yyyyyyyyyyyyyyyy	0001	0000	0105	0000	0001	0000	0001	0000			
x60000050	FFFF	FFFF	FFFF	FFFF	FFFF	FFFF	FFFF	FFFF	yyyyyyyyyyyyyyyy	0001	0000	0001	0000	0001	0000	0171	0000	q		
x60000060	FFFF	FFFF	FFFF	FFFF	FFFF	FFFF	FFFF	FFFF	YYYYY Warning		-		×	0162	0000	0001	0000	b		
6000070	FFFF	FFFF	FFFF	FFFF	FFFF	FFFF	FFFF	FFFF	303037				~	0004	0000	0004	0000			
x60000080	FFFF	FFFF	FFFF	FFFF	FFFF	FFFF	FFFF	FFFF	VVVVV A Warning: First differen	e found	at 0x600	00000.		4753	5846	5243	5059	SGFXCRYP		
x60000090	FFFF	FFFF	FFFF	FFFF	FFFF	FFFF	FFFF	FFFF	ŷŷŷŷŷ 📥					0003	8108	0080	0000	128.0±±		
x600000A0	FFFF	FFFF	FFFF	FFFF	FFFF	FFFF	FFFF	FFFF	¥¥¥¥¥					0100	0000	0004	0000			
x60000080	FFFF	FFFF	FFFF	FFFF	FFFF	FFFF	FFFF	FFFF	<u> </u>				OK	8000	0000	0000	0000			
(600000C0	FFFF	FFFF	FFFF	FFFF	FFFF	FFFF	FFFF	FFFF	~~~~	0000	0000	0104	0000	0001	0000	0000	0000			

Figure 93. Data width: 8 bits

Memory & File	e edi	tion																																		
Device memory	Op	ien fil	e	Con	npari	son	Devi	ce m	iemo	ry V	ls F7	46G.	bin	Con	npar	isor	Device memory Vs 6.hex × +																			
Address 0x600	00000		S	ze	0x	100			Dat	a wi	idth	8	bit			Find	I Data 0x																		Read	*
				Men	tory	devi	ice (Add	ress	rang	ge: [(0x60	0000	00 0	x600	000	100])									ile : (5.hex	, Ad	dres	s ran	ge: ((x60	0000000 0x60000100]			
Address	0	1	2	3	4	5	6	7	8	9	A	в	с	D	E	F	ASCII	0	1	2	3	4	5	6	7	3 9	A	в	с	D	Ε	F		ASCII		
0x60000000	DA	DA	DA	DA	DA	DA	DA	DA	DA	DA	DA	DA	DA	DA	DA	DA	00000000000000000000000	40	02	0B	81	03	00	DB B	1 8	0 00	00	00	09	00	00	00	0±±			^
0x60000010	FF	FF	FF	FF	FF	FF	FF	FF)	FF	FF	FF	FF	FF	FF	FF	FF	yyyyyyyyyyyyyyy	01	00	00	00	00	01	0 00	0 0	4 00	00	00	1F	00	00	00				
0x6000020	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	yyyyyyyyyyyyyyy	11	00	00	00	10	00	0 00	0 6	7 45	23	01	EF	CD	AB	89	gE#.i1<.			
0x6000030	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	YYYYYYYYYYYYYYYY	67	45	23	01	EF	CD	AB 8	9 0	4 01	00	00	01	00	00	00	gE#.i1<			
0x60000040	FF	FF	FF	FF	FF	FF	FF	FF)	FF	FF	FF	FF	FF	FF	FF	FF	3333333333333333333	01	00	00	00	05	01	0 00	0 0	1 00	00	00	01	00	00	00				
0x6000050	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	yyyyyyyyyyyyyyy	01	00	00	00	01	00	0 00	0 0	1 00	00	00	71	01	00	00	q			
0x60000060	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	VVVV Warning							×	6	2 01	00	00	01	00	00	00	b			
0x60000070	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	2222							~	0	4 00	00	00	04	00	00	00				
0x6000080	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	9999 🔥 Warning: First difference	e fou	und	at Ox	50000	0000			5	3 47	46	58	43	52	59	50	SGFXCRYP			
0x60000090	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	27773			_					0	3 00	08	81	80	00	00	00	128.0±±			
0x600000A0	FF	FF	FF	FF	FF	FF	FE	FF)	FF	FF	FF	FF	FF	FF	FE I	FF	2222					E	0	,	0	0 01	00	00	04	00	00	00				
0x60000080	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	2222						0	·	0	8 00	00	00	00	00	00	00				
0x600000C0	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx	00	00	00	00	04	01	0 00	0 0	1 00	00	00	00	00	00	00				~
()																																				

After launching the comparison between the flash memory and file, and the edit of data in the memory, the user must make an update in the comparison tab using the read button.



Example 3: Update comparison between flash memory and file after editing

ddress 0x60	00000		- 5	ize	0	c100			Da	ta w	idth	8	bit	*	Fin	i Data Ox																Read
				Me	mor	dev	rice	Add	iress	ran	ge: [0x60	0000	00 0x	60000	100] }								1	File :	6.hex	, Ad	dress	ranj	ge: [0)	60000000 0x60000100]	
Address	0	1	2	3	4	5	6	7	8	9	A	8	с	DI	F	ASCII	0	1	2	3	4	5	6	7	8 9	A	8	с	D	E		ASCII
x60000000	DA	DA	DA	DA	DA	DA	DA	DA	DA	DA	DĄ	DA	DA	DA D	A DA	000000000000000000000000000000000000000	40	02	08	81	03	00	08 6	1 8	0 0	0 00	00	09	00	00 0	0 0	
x60000010	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF I	FF	FFF	yyyyyyyyyyyyyyy	01	00	00	00	00	01	00 0	0 0	4 0	0 00	00	1F	00	00 0	0	
x60000020	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	ערררררררר	11	00	00	00	10	00	00 0	0 6	7 4	5 23	01	EF	CD	AS S	9gE#.i1«.	
x6000030	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF I	FF	FFF	<i>YYYYYYYYYYYYYYYYY</i>	67	45	23	01	EF	CD	AB 8	9 0	4 0	1 00	00	01	00	00 0	0 gE#.11«	
x60000040	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	ערררררררר	01	00	00	00	05	01	00 0	0 0	1 0	0 00	00	01	00	00 0	0	
x60000050	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FFF	yyyyyyyyyyyyyyyy	01	00	00	00	01	00	00 0	0 0	1 0	0 00	00	71	01	00 0	0q	
x60000060	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FFF	200000000000000000000000000000000000000	01	00	00	00	00	00	00 0	0 6	2 0	1 00	00	01	00	00 0	0b	
x60000070	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF I	FF	FF	Warning						×	0	0 0	4 0	0 00	00	04	00	00 0	0	
x6000080	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF			-					0	0 5	3 4	7 46	58	43	52	59 5	0SGFXCRYP	
x60000090	FF	FF	FE	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF I	FF	FF	Warning: First difference f	ound	at 0)	6000	0000			6	1 0	3 0	0 08	81	80	00	00 0	0 128.0±±	
x600000A0	FF	FF	FF	FF	FF	FF	FF	FF.	FF	FF.	FF	FF	FF I	FF	FF								0	0 0	0 0	1 00	00	04	00	00 0	0	
x60000080	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF I	FF	FF						0	¢.	0	0 0	8 00	0 00	00	00	00	00 0	0	
x600000C0	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	1			_		_		0	0 0	1 0	0 00	00	00	00	00 0	0	





ouress 0x000	00000	10	1	ue	1		,			0.0	mou		0.0				o cata lox																			News	
				Me	mor	y de	vice	(Ac	idres	is ra	nge:	(0x6	000	0000	0 0x	5000	0100] }									Fi	le:6	i.hes	c, Ad	idre	is ra	nge:	[0x	50000000 0x60000100]			
Address		1	2	3	4	5	6	7	8	9	A	8	c	0	E	F	ASCII	_	0	4	2 3	4	5	6	7	8	9	A	8	c	D	E	F		ASCII		
0x60000000	40	02	A	DA	DA	DA	DA	DA	DA	D	DA	DA	D	- 04	D	A D	\$.000000000000	40	0 0	2 0	8 8	L 03	8 00	0 06	81	80	00	00	00	09	00	00	0 00	0			
0x60000010	FF	FF	FF	FF.	FF	FF	FF	FF	FF	FF	FF	FF	FI	FF	FI	FI	<u> </u>	0	1 0	0 0	0 0	00	0 01	L 00	00	04	00	00	00	1F	00	00	0 00				
0x60000020	FF	FF	FF	FF.	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FI	F	222222222222222222222222222222222222222	1	1 0	0 0	0 0	10	0	00 0	00	67	45	23	01	EF	CD	AB	8	gE#.i1«.			
0x60000030	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FI	FI	yyyyyyyyyyyyyyyyy	6	7 4	5 2	3 0	EF	C	AB	89	04	01	00	00	01	00	00	0 00	gE#.i1«			
0x60000040	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FI	F	yyyyyyyyyyyyyyyyy	0	1 0	0 0	0 0	0 05	0	L 00	00	01	00	00	00	01	00	00	0 00				
0x60000050	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FI	F	<u> </u>	0.	1 0	0 0	0 0	0 01	0	00 0	00	01	00	00	00	71	01	00	0 00	·			
0x60000060	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FE	FF	FF	EF	FI	E	Warning	-					×	10	00	62	01	00	00	01	. 00	00	0 00	b			
0x60000070	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FI	E	and the second se							20	00	04	00	00	00	04	00	00	0				
0x60000080	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FI	FF	FI	: FI	Warning: First difference fou	nd ar	0xi	0000	0002.			20	00	53	47	46	58	43	52	59	5	SGFXCRYP			
0x60000090	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FI	F								26	81	03	00	0 OE	81	. 80	00	00	0 00	128.6			
0x600000A0	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	F	F						~		1 20	00	00	01	00	00	04	00	00	0 0				
0x60000080	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FI	F						1.03	-	20	00	08	00	00	00	00	00	00	0 0				
0x600000C0	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FI	F	xxxxxxxxxxxxxxxxx	0	0 0	0 0	0 0	0.04	0	1 00	00	01	00	00	00	00	00	00	0				

Note:

The user can make multiple comparisons between flash memory and files.

Figure 96. Multiple comparisons

≡	Memory	& File	edition									
	Device m	emory	Open file	Com	parison Devi	ce memory Vs F74	6G.bin	Comp	arison Device	memory Vs ext1.	hex Comparison Device memory Vs ext_comp.bin ×	+
Ŀ	Address	0x0	*	Size	0x30	Data width	32-bit	٠	Find Data	0x		

2.20 Comparison between two files

Description: Compares the content of two different files (binary, hex, srec, elf, out and axf). The difference is colored in red in the grid panel of each file.

This operation does not need a connected board.

The used files can be of different sizes and types.

The user can open the comparison window from different sub-menus.



≡	Memory & File	edition					💛 Connected	
	Device memory	1743.hex +					ST-LINK 👻 Disconnect	
	Address 0x08000	0000 • Size	0x400	Data width	E-bit • Find	Data Ox	Read Still JINK configuration	
	Address	0	4	8	с	ASCII	Save As	8
OB	0x08000000	3032303A	30303030	30383034	0A324630	:020000040800F2.	Fill memory	4
	0x08000010	3030323A	30303630	41414130	30303041	:20006000AAAA000	Blank check 4000	
CPU	0x08000020	46464630	46464646	46464646	46464646	0FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF	Compare memory with file	-
	0x08000030	46464646	46464646	46464646	46464646	FFFFFFFFFFFFFFF	Compare two files	
swv	0x08000040	46464646	45454546	46464645	46464646	FFFFFEEEEFFFFFFF	Compare two mes	
	0x08000050	46464646	46464646	0A413646	3030323A	FFFFFFF6A.:200	Keset mode Hardware reset	
REG	0x08000060	30303830	46464630	46464646	46464646	08000FFFFFFFFFFF	Shared Disabled	
	0x08000070	45454546	46464645	46464646	46464646	FEEEFFFFFFFFFF	Debug in Low Power mode	Z

Figure 97. Sub-menu displayed from "Read" combo-box in device memory tab

Figure 98. Sub-menu displayed with right click on "Device memory" tab





	Memory & File e	dition									
	Device memory	743.hex × +									
	Address 0x80000	00 👻 Size	0x200		Data width 32	e-bit 🔻	Find	Data Ox		Download	•
	Address	0	4	4	8		с		ASCII		
OB	0x08000000	3032303A	303030	30	30383034	0A3240	530	:0200004080	0F2.		^
	0x08000010	3030323A	303036	Conv			\$1	:20006000AA	A000		
CPU	0x08000020	46464630	464646	Cut			16	OFFFFFFFFF	FFFF		
	0x08000030	46464646	464646	Dacto			16	FFFFFFFFFF	FFFF		
swv	0x08000040	46464646	45454!	cill .	amort.		46	FFFFEEEFFF	FFFF		
	0x08000050	46464646	464646	nlash	abaala		3A	FFFFFFFF6A	:200		
REG	0x08000060	30303830	464646	втапк	спеск		46	08000FFFFFF	FFFF		
BETA	0x08000070	45454546	46464	Compar	re memory wit re two files	n tile	16	FEEEEFFFFFF	FFFF		>~



	-	.9			nop.	<u>~</u> ,							
	Memory & File e	edition											
	Device memory H	743.hex ×	+										
<u>.</u>	Address 0x80000	00 - S	Open file Open memory tab		vidth	32-t	oit 🔻	Find	Data	Ox		Download	•
	Address	0		ith filo	8		с				ASCII		
OB	0x08000000	30323034	compare memory w	iui iie	8034		0A32463	0	:020	000004080	0F2.		_
	0x08000010	30303234	Compare two files		130		3030304	1	:200	06000AAA	A000		
CPU	0x08000020	46464630	46464646	464	54646		4646464	6	OFFF	FFFFFFFF	FFFF		
	0x08000030	46464646	46464646	4640	54646		4646464	6	FFFF	FFFFFFFF	FFFF		
swv	0x08000040	46464646	45454546	4640	54645		4646464	6	FFFF	FEEEEFFF	FFFF		
	0x08000050	46464646	46464646	0A4	13646		3030323	A	FFFF	FFFFF6A.	:200		
REG	0x08000060	30303830	46464630	4640	54646		4646464	6	0800	0FFFFFFF	FFFF		
BETA	0x08000070	45454546	46464645	4640	54646		4646464	6	FEEE	EFFFFFFF	FFFF		
	000000000	ACACACAC	ACACACAC	ACA	CACAC		ACACACA	c	erer	errerer	erer		\sim

Figure 100. Sub-menu displayed with add tab button

Figure 101. Sub-menu displayed with right click on the opened file tab

≡	Memory	& File e	editio	on										
	Device m	emory H	1743.1	Save As	Ctrl+S									
I	Address	0x80000	00	Verify	Ctrl+V	dth	32-bit	*	Find D	Data	0x		Download	-
	Add	iress		Download	Ctrl+D	8		с				ASCII		
OB	0x08000	000	30	Open file	Ctrl+O	034	0A	32463	0	:020	000040800F	2.		
	0x08000	010	30	Close tab	Ctrl+C	130	30	30304	1	:200	06000AAAA0	000		
CPU	0x08000	020	464	Address 0x8000000		646	46	46464	6	0FFF	FFFFFFFFF	FF		
	0x08000	030	464			646	46	46464	6	FFFF	FFFFFFFFFF	FF		
swv	0x08000	040	464	Close tab		645	46	46464	6	FFFF	FEEEEFFFFF	FF		
	0x08000	050	464	Close other tabs		646	30	30323	A	FFFF	FFFFF6A.:2	200		
REG	0x08000	060	30	Option bytes	Ctrl+B	646	46	46464	6	0800	OFFFFFFFF	FF		
DETA	0x08000	070	454	Fill memory	Ctrl+M	646	46	46464	6	FEEE	EFFFFFFFFF	FF		
	000000	000		Blank check	Ctrl+L	CAC		ACACA	c					~
	Log			Compare memory with i	ilo Ctrl+T			L	ive Upd	ate	Verbosity leve	el 🔘 1	2	3
	15:06:02 : 15:06:02 : 15:06:02 :	Size : Address	512 I 50x0	Compare two files	Ctrl+F								^	\$
	15:06:02 : 6	trasing mem	nory &	prresponding to segment u:										

Figure 102. Sub-menu displayed from "Download" combo-box displayed in file tab

≡	Memory & File	edition							Connected	d
	Device memory	H743.hex × +						ST-LINK	 Disconnec 	ct
	Address 0x8000	000 👻 Size	0x200	Data width	32-bit 💌	Find Data 0x	Download 👻	ST-LI	NK configuration	
	Address	0	4	8	с	ASCII	Read			
OB	0x08000000	3032303A	30303030	30383034	0A324630) :020000040800F2.	Save As		SWD	× .
	0x08000010	3030323A	30303630	41414130	30303041	:20006000AAAA000	Verify	(kHz)	4000	
CPU	0x08000020	46464630	46464646	46464646	46464646	0FFFFFFFFFFFFFF	Address		Hot plug	
	0x08000030	46464646	46464646	46464646	46464646	5 FFFFFFFFFFFFFFF	Address 0x8000000			
swv	0x08000040	46464646	45454546	46464645	46464646	5 FFFFEEEEFFFFFF	Compare memory wi	th file	0	Υ.
	0x08000050	46464646	46464646	0A413646	30303234	FFFFFFF6A.:200	Compare two files		Hardware reset	
REG	0x08000060	30303830	46464630	46464646	46464646	08000FFFFFFFFFF		Shared	Dirabled	
BETA	0x08000070	45454546	46464645	46464646	46464646	5 FEEEFFFFFFFFFF			Dissipied	•
	008000080	ACACACAC	ACACACAC	ACACACAC	ACACACA		~	Debug in Low Po	wer mode	

Г

In addition to sub-menus to display this window, the user can open it directly by using the key combination Ctrl+F.

Example: Difference between two files of the same type and different sizes

iddress 0x60	000000 💌 Si	ze 0x400	Data widt	h 32-bit 💌	Find Data Ox						Re	ad
		File : 6.hex A	ddress range: [0	x60000000 0x60	000100)			File : 6_9.hex	Address range: [0x6000000 0x6	0000400]	
Address	0	4	8	c	ASCII	Address	0	4	8	c	ASCII	
0x6000000	81080240	81080003	00000080	00000009	G Warning		×	B10B0240	B1060003	00000080	128.0±±	
0x60000010	00000001	00000100	00000004	0000001F			-	00000002	00000100	00000004		
0x6000020	00000011	00000010	01234567	89ABCDEF	Warning: First diff	rrence found at 0x6000010	0.	00000011	00000008	00000000	•••••	
0x6000030	01234567	89ABCDEF	00000104	00000001	gE#.11 ()		_	00000104	00000001	00000000		
0x60000040	00000001	00000105	00000001	0000001			OK	00000001	00000000	00000001		
0x60000050	0000001	00000001	00000001	00000171			_	00000001	00000171	00000001	q	
0x60000060	00000001	00000000	00000162	00000001	h	0,200000340	.00000000	00000162	00000001	00000001	b	
0x60000070	00000000	00000000	00000004	00000004	····· warning		^	00000004	00000004	0000003		
0x6000080	0000003	0000000C	58464753	50595243	Warning: The size	of the two files are differe	oti	58464753	00434150	00383231	SGFXPAC.128.	
0x6000090	00383231	B10B0240	B1060003	00000080	128.0.			00000000	81060240	B1080003		
0x600000A0	0000009	0000002	00000100	00000004				0000009	0000003	00000100		
0x60000080	0000001F	00000011	00000008	00000000			OK	0000001F	00000011	00000004		
0x600000C0	00000000	00000104	00000001	00000000		0.00000130	JOUMDETE	00000104	00000001	00000000	þ0°	
0x60000000	00000105	00000001	00000000	00000001		0x60000160	00000105	00000001	00000000	00000001		
0x60000E0	00000001	00000001	00000171	00000001	q	0x60000170	00000001	00000001	00000171	00000001	q	
0x600000F0	00000000	00000162	0000001	00000001	b	0x60000180	00000000	00000162	00000001	00000001	b	
99 159457 - Ivarning 159457 - Itme etas 159453 - Read File 159453 - Reament 159454 - Reament 159454 - Reament 159454 - Reament 1594554 - Warning	Padoress range (pm psed during the cor- first difference fox is CISOFT_DOCSIN- of segments: 1 (0): address: 0x600 (1): address	00001100 UNB0000 mparison bitween in an 0x60000100, ww folder_V6.hex 00000, size= 0x400 00000, size= 0x400 00000, size= 0x400 00000, size= 0x400	oog is not avanatie les is: 00.0000.001	in (6.bes)						Live Update	Verbosity level 💿 1 🥚	2



Figure 104. Data width: 16 bits





Device memory	Ope	n file	Co							××	•																												
Address 0x600	00000	٣	Size	0	x400				ata	widt	۰ (8-bi	t	¥	Fir	d Da	ita	0x																					Read
				File	: 6.h	ex,	Add	ress	ran	ge: [)×60	000	000 0	0x60	0001	00]	1				File : 6_9.hex .Address range: [0x6000000 0x60000400]																		
Address	0	1 2	3	4	5	6	7	8	5	A	8	c	D	E	F			AS	CII			Address		0	1 2	3	4	5	6	7	8) A	В	с	D	E	F	AS	C11
0x6000000	40	02 06	81	03	00	06	81	8	0	0 0	0	09	00	00	00	Ø.,	.±.,	.±			0×6	000009	0	31 3	2 38	00	40	02	08	1 0	03 0	0 08	81	80	00	00 (00	128.0±±	
0x60000010	01	00 00	00	00	01	00	00	0	0	0 0	0	15	00	00	00						0×6	00000A	0	0 90	0 00	00	02	00	00	0 0	00 0	1 00	00	04	00	00 (00		
0x6000020	11	00 00	00	10	00	00	00	6	4	5 2	0	EF	C	A	89			Warnin	0							×	11	00	00	0 0	08 0	0 00	00	00	00	00 (00		
0x6000030	67	45 23	01	EF	CD	AB	89	0	0	1 0	0	0 01	00	00	00	gE	8.1	_	-								04	01	00	0 0	01 0	0 00	00	00	00	00 (00		
0x60000040	01	00 00	00	05	01	00	00	0	0	0 0	0	01	00	00	00				Warning	: First dif	ference f	lound at 0	hx60000	100.			01	00	00	0 0	00 0	0 00	00	01	00	00 (00		
0x60000050	01	00 00	00	01	00	00	00	0	0	0 0	0	73	03	00	00			- <u> </u>						_			01	00	00	0 7	71 0	1 00	00	01	00	00 (00	q	
0x60000060	01	00 00	00	00	00	00	00	6	0	L 00	0	0 01	. 00	00	00										OK		62	01	00	0 0	01 0	0 00	00	01	00	00 (00	b	
0x60000070	00	00 00	00	00	00	00	00	0	0	0 0	0	04	00	00	00										OK		04	00	00	0 0	04 0	0 00	00	03	00	00 (00		
0x6000080	03	00 00	00	0C	00	00	00	5	4	7 44	51	43	52	59	50			SGFXCR	ΥP		0x6	000011	0	DC 0	0 00	00	53	47	46	8 5	50 4	1 43	00	31	32	38 (00	SGFXPAC.12	8.
0x60000090	31	32 38	00	40	02	08	BJ	0	0	0	B	80	00	00	00	12	8.0	. <u>*</u> *			0×6	000012	0	0 00	0 00	00	00	00	00	0 4	40 O	2 08	81	03	00	06 1	81		. #
0x600000A0	09	00 00	00	02	00	00	00	0	0	1 0	0	04	00	00	00			- Pra Warr	ning							×	9	00	00	0 0	03 0	0 00	00	00	01	00	00		
0x60000080	1F	00 00	00	11	00	00	00	0	0	0 0	0	00	00	00	00				_						_		F	00	00	0 1	11 0	0 00	00	04	00	00 (00		
0x600000C0	00	00 00	00	04	01	00	00	0	0	0 0	0	00	00	00	00			· 🔥	Warni	ng: The si	izes of th	ne two file	is are d	fferer	61		4	01	00	0 0	01 0	0 00	00	00	00	00 (00	þ0°	
0x60000000	05	01 00	00	01	00	00	00	0	0	0 0	0	01	00	00	00			🛎							_		1	00	00	0 0	00 0	0 00	00	01	00	00 (00		
0x600000E0	01	00 00	00	01	00	00	00	73	0	1 01	0	01	. 00	00	00									1	OR	c	1	00	00	0 7	71 0	1 00	00	01	00	00 (00	q	
0x600000F0	00	00 00	00	62	01	00	00	0	0	0 0	0	0 01	00	00	00		b							_			2	01	00	0 0	01 0	0 00	00	01	00	00 (00	b	
Log																					N.C.													Live I	Joda	te	Vert	bosity level 💿 1	() 2
1546:54 : Warning 1546:54 : Time elay 1546:54 : Warning 15:54:04 : Read File	HOORESS Ised dur First diff	ng the	omp ound New	oono vison at Ov folder	bitw 6000	een 0100	lies	s 00	00-0	2.001	un (e	nen;																							.,				
15:54:04 : Number 15:54:04 : segment 15:54:04 : Read File 15:54:04 : segment 15:54:04 : segment 15:54:04 : segment 15:54:04 : segment	Mol Index Targential Society Molece M																																						
15:54:04 : Warning: 15:54:04 : Warning: 15:54:04 : Time elas	The size Address ised dur	s of the range	two f bu600	les ar coto rison	e diff 0.6 bitw	erer 2000	e 800) Nes	is na s: 00	: avi	2.000	in (f	ihex]	٦																										



The user can make multiple comparisons between files.

Figure 106. Multiple comparisons

≡	Memory & File	e edition		
	Device memory	Comparing 6.hex and 6_9.hex	Comparing 200.hex and all_mixed.bin	Comparing 9.hex and 200.hex × +
	Address 0x800	00200 🔻 Size 0x400	Data width 32-bit 👻 Fine	d Data Ox



2.21 LiveUpdate feature

-liveUpdate checkbox

Description: When this feature is used the device memory grid is updated in real time and the modified data are highlighted in pink.

Once the device is connected, the user can check the liveUpdate checkbox, memory data are updated in real time.

STM32 Cube	Programmer					.19	f 🕒	y 🛪 🏹
	Memory & File	e edition						Connected
	Device memory	Open file +					ST-LINK	Disconnect
•	Address 0x200	40000 - Size	0x1000	Data width	32 bit + Find	d Data Ox Read	Serial number	NK configuration
	Address	0	4	8	с	ASCII	Port	066/445 • 0
OB	0x20040000	0000002B	1912F6EF	7BED5384	4099E6AB	+ïöSí{«æ.@		SWO
	0x20040010	FBFB8803	B092FA6B	CBFDB709	E1C5F0B7	ûûkú.°.•ýĔ•ðĂá	Frequency (kHz)	4000 👻
CPU	0x20040020	BCBFB6DC	75C0FF7B	7BFCD6B1	752BE7DD	ܶ¿¼{ÿÀu±Öü{Ýç+u	Mode	Normal
	0x20040030	F78C35D0	704DB9F7	5F6A0541	5645DD97	Ð5.÷÷¹MpA.jÝEV	Access port	
swv	0x20040040	F7A88824	C54C752F	FF570444	4E28D673	\$."÷/uLÅD.WÿsÖ(N	Access bout	0
=	0x20040050	D7FAAE01	06D0AFEC	58342014	42E4159F	.®úxì¯Đ 4XäB	Reset mode	Software reset 🔷
REG	0x20040060	FE751289	1CE2DDF9	BD586240	1C849AEB	uþùÝâ.@bX½ë	Shared	Disabled .
BETA	0x20040070	A7AB803A	4869C32E	BFB34412	13706118	:.«§.ÃiH.D³¿.ap.		
	0x20040080	B6A38F23	OC80FEFB	36FF864E	8557BBFD	#.£¶ûþN.ÿ6ý≫W.	External loader	
	0x20040090	4E7B2245	3DF1FFFE	97DC22F4	98905FD8	E"{Nþÿñ=ô"Ú.Ø	Target voltage	
	0x200400A0	E7F13C0A	94E0D11F	6BDF0E16	0385620A	.<ñç.Ñàßk.b	Firmware version	1 V2J37M26
	0x200400B0	FFABBE93	84EEE5EF	1FF78E82	034E363E	.¾«ÿïåî÷.>6N.		
	0x200400C0	DFF60C52	0340EBD7	5EF792A5	C2A0D3FF	R.öß×ë@.¥.÷^ÿÓ Â		
	20000000					1 1107 1- 0.18	Tar	get information
	Log				V Live U	odate Verbosity level 💿 1 💿 2 💿 3	Board	NUCLEO-L496ZG
•••	11:04:18 : UPLOADI 11:04:18 : Size 11:04:18 : Address 11:04:18 : Read pro	NG : 4096 Bytes : 0x20040000 gress:					Device Type Device ID Revision ID	STM32L496xx/STM32L4A6xx MCU 0x461 Rev 2.0

Figure 107. Live update of data



2.22 Calculator

Description: The Calculator window, created for general purposes, is always available, even if the device is not connected. The user interface has three main goals:

- 1. Number converter
 - Composed by several graphical components, to facilitate the number conversion between decimal, hexadecimal, and binary formats.
 - Use the 32 check boxes, representing a word of 32 bits, to activate or deactivate the relevant bit(s).
 - Use the "Reset" button to reinitialize the number to 0.
 - Any time a bit is changed, the number value is updated.
- 2. Checksum calculation
 - To calculate the checksum value, based on addition algorithm applicable on the file content or on the flash memory.
 - File Checksum: choose your binary file and click on "Calculate" button to display the corresponding result on the log panel.
 - Flash Memory Checksum: calculates the checksum value of a region (based on start address and size of the desired region) once the device is connected.
 - To calculate the full flash memory checksum retrieve the memory size, displayed in the "Flash size" field.
- 3. Memory programming
 - To expose the generic memory edition options
 - Fill memory: see Section 2.16
 - Blank check: see Section 2.18

Figure 108. Calculator window



Number converter and *File checksum* can be used even if there is no device connected. *Flash Memory Checksum, Fill Memory,* and *Blank Check* are applicable only if a device is already connected.



3 STM32CubeProgrammer command line interface (CLI) for MCUs

3.1 Command line usage

The following sections describe how to use the STM32CubeProgrammer from the command line. Available commands are shown in *Figure 109*.

Note: To launch command line interface on macOS, call STM32CubeProgrammer.app/Contents/MacOs/bin/STM32_Programmer_CLI.



Usage : STM32_Programmer_CLI.exe	[command_1] [Arguments_1][[command_2] [Arguments_2]
Seneric commands: -7, -h, -help : -version, -version -1, -list (usb) -g,guietMode -log, -log ({file_Path.log>] : vLevel> tuallable commands for 31 	: Show this help Displays the tool's version List all available communication interfaces UMRT interface USB interface Store the detailed output in log file Path of the log file, STM32Programmer/trace.log Specify perhasity level Specify level, value in (1, 2, 3) H32 MCU
skipErase sl,safelib (file_path) (catar_address) (calce_size) 	: Skip sector erase before programming Add a segment into a firmware file (elf, bin how, sree) consisting computed (DK values trian be modified File path to be modified Filesh memory start address Filesh memory start address Size of data per CKC value Add a binary header and a sbsfu segment to an elf file Hader file path to be modified Hader file path
 -cconnect yopt-(PortName) UART port optional pp (hp-(haudrets)) Tdobadcets): Tdobadcets): Tdobadcets): Ifc=(FlowControl): Incoint=noinit_bit): Incorolol Ifc-%(Frequency): 	Establish connection to the device Interface identifier.ex COMI. /dev/tt980.ushl, JTRG, SUD) Baudrate.ex: 115200. %600.etc. default 115200 Datablish alueuin G. (NOME ODD.EURO) default EUEN Datablish alueuin G. (NOME ODD.EURO) default EUEN Establish alueuin G. (NOME ODD.EURO) default EUEN Establish, value in C(1.1.5, 2), default 1 Flow control Value in COFF.Hardware.Software), default OFF Not supported for STR32MP Set No Init bits.value in (0.1), default 0 Einter URI console node ptional parametore: Fraudrate Company and Company alueuin (2)
<pre>[index=<index>] in=<cerialmunher>]: in=<cerialmunher>]: inode] inode</cerialmunher></cerialmunher></index></pre> inode ino	24000 SUD 21333 with STLINK03 index of the dobug probe Access Fort andex to connect to a default ap 0 Access Fort andex to connect to a default ap 0 Access Fort andex to connect to a default ap 0 Access Fort andex to connect to a default ap 0 default node: NORMAL Beset nodes: SWrst/Wrst/Crst. Default node: SWreset anterps: Baudratc. 1Edge or 2Edge. default 1Edge i analte or 2Edge. default 1Edge i analte or disable (0/1). crc polonon value. Bbit/16bit 21Fnilbuplex/2LRxOnly/1LRx/1LTx Prame Format: Motorola/II Mode: naster/slave
<pre>ide lay=Conl)] ide lay=Conl)] inoinit=noinit=bit]: CAN port optional pan tbm=C+baudrace) idm=Copmed>1 idm=Copmed>1 idm=Copmed>1 ifm=Cformat>1 ifm=Cform</pre>	: Delaoy Nobelaoy, delay of fiew microseconds : Set No Init bits, value in (0,1), default 0 vameters: : Baudrate : 125, 250, 500, 1000 Kbps, default 125 : CAN Tupe : SINNDARD or EXIENDED, default SINNDARD : CAN Tupe : SINNDARD or EXIENDED, default SINNDARD : Frame Roymat: Dilh or HEITENDEL, default MOXMAL : Filer Nobe : MOXM or List, default MOXM : Filer Scale: 16 or 32, default 32 : Filer Scale: 16 or 32, default 32 : Filer Scale: 16 or 32, default 48 : Filer Scale: 10 or 32, default 48 : Filer Scale: 10 or 32, default 48 : Filer Scale: 10 or 32, default 48 : Filer Scale: 10 or 32, default 8 : Filer Scale: 10 or 34 : Filer
<pre>[dnf=(dnfilter)] : [nf=(rtime)] : [ft=(ftime)] : [noinit=noinit_hoit]: -e,erase : [dill] : [(sectorsCodes)] : [((start end])] : -y,yrite</pre>	Inigital noise filter: 6 too ls. default 9 default 6 has time: $\theta = 1000(57)$ default 6 has time: $\theta = 1000(57)$ default 6 so that $\theta = 1000(57)$ default 6 so that $\theta = 1000(57)$ default 6 so that $\theta = 1000(57)$ default 6 has a solution of
-adownload (file_path) -u32(address) (32-hit_data) -vverify -r32 (address) (32-bit_data) -vverify -r32 (cize) -rst	Download the content of a file into device memory File path name to be downloaded: (bin, hex, srec, elf, stn32 or tsv file) Start address of download start address of download S2-bit data to be download values should be separated by space Uerify if the programming operation is achieved successfully head a 32-bit data from device memory Size of the Start set of the separated Rest system
-hardRst : -halt : -step : -coreBeg : ((core_register>)] [core_reg=(value>)] -rread	 Hardwaré reset Hardwaré reset Halte cone Step cone nucleone /ul>
-u,upload <address> <size> <file_path> -el,extload</file_path></size></address>	: Upload the device memory content to a .bin file : Start address of read and upload : Size of memory content to be read : Binary file path : Select a custon external m <u>emory-loader</u>
<pre></pre>	: External memory-loader file path : Run the code at the specified address. : Start address : Remove memory's Read Protection by shifting the RDP Ievel from level i to level 0.
-ob,optionbytes : [displ] : [OptByte=⟨value⟩] :	This command allows the user to manipulate the device 's OptionBytes by displaying or modifying them. This option allows the user to display the whole set On the option byte. Option Byte.

Figure 109. STM32CubeProgrammer: available commands



3.2 Generic commands

This section presents the set of commands supported by all STM32 MCUs.

3.2.1 Connect command

-c, --connect

Description: Establishes the connection to the device. This command allows the host to open the chosen device port (UART/USB/JTAG/SWD/SPI/CAN/I2C).

```
Syntax: -c port=<Portname> [noinit=<noinit_bit>] [options]
```

port= <portname< th=""><th>Interface identifier, ex COMx (for Windows), /dev/ttySx for Linux), usbx for USB interface, SPI, I2C and CAN for, respectively, SPI, I2C and CAN interfaces.</th></portname<>	Interface identifier, ex COMx (for Windows), /dev/ttySx for Linux), usbx for USB interface, SPI, I2C and CAN for, respectively, SPI, I2C and CAN interfaces.
[noinit= <noinit_bit>]</noinit_bit>	Set No Init bits, value in {0, 1}, default 0. Noinit = 1 can be used if a previous connection is active.
ST-LINK options	

[freq= <frequency>]</frequency>	Frequency (in kHz) used in connection. Default value is
	4000 kHz for SWD port, and 9000 kHz for JTAG port

Note: The entered frequency values are rounded to correspond to those supported by ST-LINK probe.

[index= <index>]</index>	Index of the debug probe. Default index value is 0.
[sn= <serialnumber>]</serialnumber>	Serial number of the debug probe. Use this option if you need to connect to a specific ST-LINK probe of which you know the serial number. Do not use this option with Index option in the same connect command.
[mode= <mode>]</mode>	Connection mode. Value in {NORMAL/UR/HOTPLUG}. Default value is NORMAL.
Normal	With "Normal" connection mode, the target is reset, then halted. The type of reset is selected using the "Reset Mode" option.
UR	The "Connect Under Reset" mode enables connection to the target using a reset vector catch before executing any instructions. This is useful in many cases, for example when the target contains a code that disables the JTAG/SWD pins.
HOTPLUG	The "Hot Plug" mode enables connection to the target without a halt or reset. This is useful for updating the RAM addresses or the IP registers while the application is running.
POWERDOWN	Allows to put the target in debug mode, even if the application has not started since the target power up. The hardware reset signal must be connected between ST-Link and the target. This feature might be not fully effective on some boards (MB1360, MB1319, MB1361, MB1355) with STMPS2141 power switch.



[ap= <accessport>]</accessport>	Access port index. Default access port value is 0.
[speed=]	Connection speed. Default is Reliable. Available only for Cortex-M33.
Reliable	Allows the user to connect with a slow mode.
Fast	Allows the user to connect with a fast mode.
[shared]	Enables shared mode allowing connection of two or more instances of STM32CubeProgrammer or other debugger to the same ST-LINK probe.
[tcpport= <port>]</port>	Selects the TCP Port to connect to an ST-Link server. Shared option must be selected. Default value is 7184.
[dlpm / lpm]	Disable/enable the debug in Low power mode (default configuration is enabled for the supported devices (STM32U5/WB/L4 series).
[getAuthID]	Get device identification (only for STM32U5 series): is a 32-bit device specific quantity that can be read though the JTAG port. This 32-bit information is used to derive the expected OEM password keys to unlock this specific device. This command is not applicable when RDP level = 0 (MCU constraint).
Shared mode is supported	d only on Windows.

USB options

The connection under the DFU interface supports two options, namely product and vendor ID (default values PID = 0xDF11, VID = 0x0483).

- SPI options
- [br=<baudrate>] Baudrate (for example 187, 375, 750), default 375

Note: To use SPI on high speed, an infrastructure hardware must be respected to ensure the proper connection on the bus.

[cpha= <cpha_val>]</cpha_val>	1Edge or 2Edge, default 1Edge
[cpol= <cpol_val>]</cpol_val>	Low or high, default low
[crc= <crc_val>]</crc_val>	Enable or disable (0/1), default 0
[crcpol= <crc_pol>]</crc_pol>	CRC polynomial value
[datasize= <size>]</size>	8- or 16-bit, default 8-bit
[direction= <val>]</val>	2LFullDuplex/2LRxOnly/1LRx/1LTx
[firstbit= <val>]</val>	MSB/LSB, default MSB
[frameformat= <val>]</val>	Motorola/TI, default Motorola
[mode= <val>]</val>	Master/slave, default master
[nss= <val>]</val>	Soft/hard, default hard
[nsspulse= <val>]</val>	Pulse/NoPulse, default Pulse
[delay= <val>]</val>	Delay/NoDelay, default Delay



	I2C options	
	[add= <ownadd>]</ownadd>	Slave address: address in hex format
Note:	I2C address option m	ust be always inserted, otherwise the connection is not established.
	[br= <sbaudrate>]</sbaudrate>	Baudrate: 100 or 400 kbps, default 400 kbps.
	[sm= <smode>]</smode>	Speed Mode, STANDARD or FAST, default FAST.
	[am= <addmode>]</addmode>	Address Mode: 7 or 10 bits, default 7.
	[af= <afilter>]</afilter>	Analog filter: ENABLE or DISABLE, default ENABLE.
	[df= <dfilter>]</dfilter>	Digital filter: ENABLE or DISABLE, default DISABLE.
	[dnf= <dnfilter>]</dnfilter>	Digital noise filter: 0 to 15, default 0.
	[rt= <rtime>]</rtime>	Rise time: 0-1000 (STANDARD), 0-300 (FAST), default 0.
	[ft= <ftime>]</ftime>	Fall time: 0-300 (STANDARD), 0-300 (FAST), default 0.
	CAN options	
	[br= <rbaudrate>]</rbaudrate>	Baudrate: 125, 250, default 125.
	[mode= <canmode>]</canmode>	Mode: NORMAL, LOOPBACK, default NORMAL.
Note:	The software must rec	quest the hardware to enter Normal mode to synchronize on the CAN

ote: The software must request the hardware to enter Normal mode to synchronize on the CAN bus and start reception and transmission between the Host and the CAN device. Normal mode is recommended.

[ide= <type>]</type>	Type: STANDARD or EXTENDED, default STANDARD
[rtr= <format>]</format>	Frame format: DATA or REMOTE, default DATA
[fifo= <afifo>]</afifo>	Assigned FIFO: FIFO0 or FIFO1, default FIFO0
[fm= <fmode]< th=""><th>Filter mode: MASK or LIST, default MASK</th></fmode]<>	Filter mode: MASK or LIST, default MASK
[fs= <fscale>]</fscale>	Filter scale: 16 or 32, default 32
[fe= <fenable>]</fenable>	Activation: ENABLE or DISABLE, default ENABLE
[fbn= <fbanknb>]</fbanknb>	Filter bank number: 0 to 13, default 0

Using UART

./STM32_Programmer.sh -c port=/dev/ttyS0 br=115200

The result of this example is shown in *Figure 110*.

Figure 110. Connect operation using RS232

UM2237 Rev 21



STM32CubeProgrammer provides the possibility to configure RTS and DTR pins:

- RTS, used as follows: rts=low
- DTR, used as follows: dtr=high

Example: STM32_Programmer_CLI.exe -c port=COM27 dtr=high (see Figure 111).

Figure 111. Enabling COM DTR pin



Example using USB

./STM32_Programmer.sh -c port=usb1

The result of this example is shown in *Figure 112*.

Figure 112. Connect operation using USB							
establishing connection with the target device							
USB speed Manufacturer ID Product ID Serial number Firmware version Device ID ARFA NAME	: FULL_SPEE : STMicroel : STM32 BO : 326F37603 : 1.1a : 0x0419 SECT_NBR	D(12MBit/s) ectronics OTLOADER 234 ADDRESS	SIZE	түре			
Internal Flash	0000 0001 0002 0003 0004 0005 0006 0007 0008 0007 0010 0011 0012 0013 0014 0015 0014 0015 0014 0015 0016 0017 0018 0017 0018 0019 0018 0019 0020 0021 0022 0021	0×08000000 0×08004000 0×08005000 0×08005000 0×08020000 0×08020000 0×08050000 0×08050000 0×08050000 0×08050000 0×08050000 0×081040000 0×08104000 0×081104000 0×08120000 0×08120000 0×08120000 0×08150000 0×08120000 0×08150000 0×08160000 0×08100000 0×081000000 0×0810000000	0016 KB 0016 KB 0016 KB 0016 KB 0128 KB 0128 KB 0128 KB 0128 KB 0128 KB 0128 KB 0128 KB 0128 KB 0128 KB 0016 KB 0016 KB 0016 KB 0016 KB 0016 KB 0016 KB 0016 KB 0016 KB 0016 KB 0016 KB 0016 KB 0016 KB 0018 KB 0128 KB 0128 KB 0128 KB 0128 KB 0128 KB 0128 KB 0128 KB 0128 KB 0128 KB 0128 KB 0128 KB 0128 KB	REW REW REW REW REW REW REW REW REW REW			
Option Bytes	0000 0001	0x1fffc000 0x1ffec000	0016 B 0016 B	RW RW			
OTP Memory	0000 0001	0x1fff7800 0x1fff7a00	0512 B 0016 B	RW RW			
Device Feature	0000	0×ffff0000	0004 B	RW			

Note: When using a USB interface, all the configuration parameters (for example baud rate, parity, data-bits, frequency, index) are ignored. To connect using a UART interface the port configuration (baudrate, parity, data-bits, stopbits and flow-control) must have a valid combination, depending on the used device.



Example using DFU IAP/USBx options

/STM32_Programmer.sh -c port=usb1 pid=0xA38F vid=0x0438

The result of this example is shown in Figure 113.

Figure 113. Connect operation using USB DFU options



Note:

The default value of product ID and vendor ID are ST products values (PID = 0xDF11, VID = 0x0483).

Example using JTAG/SWD debug port

To connect using port connection mode with ST-LINK probe it is necessary to mention the port name with at least the connect command (for example: -c port=JTAG).

Note: Make sure that the device being used contains a JTAG debug port when trying to connect through the JTAG.

There are other parameters used in connection with JTAG/SWD debug ports that have default values (see the Help menu of the tool for more information about default values).

The example below shows a connection example with an STM32 with device ID 0x415.

Figure 114. Connect operation using SWD debug port

ST-LINK SN :	Ø66BFF574857847167114941
ST-LINK FW :	V2J30M20
Voltage :	3.250
SWD freg :	4000 KHz
Connect mode:	Normal
Reset mode :	Software reset
Device ID :	0×415
Device name :	SIM32L4x1/SIM32L475xx/SIM32L476xx/SIM32L486xx
Device type :	MCU
Device CPU :	Cortex-M4

The corresponding command line for this example is -c port=SWD freq=3900 ap=0



UM2237 Rev 21

In the connect command (-c port=SWD freq=3900 ap=0)

- The <port> parameter is mandatory.
- The index is not mentioned in the command line. The Index parameter takes the default value 0.
- The frequency entered is 3900 kHz, however the connection is established with 4000 kHz. This is due to the fact that ST-LINK probe has fixed values with SWD and JTAG debug ports.
- ST-LINK v2/v2.1
 - SWD (4000, 1800, 950, 480, 240, 125, 100, 50, 25, 15, 5) kHz
 - JTAG (9000, 4500, 2250, 1125, 562, 281, 140) kHz
- ST-LINK v3
 - SWD (24000, 8000, 3300, 1000, 200, 50, 5)
 - JTAG (21333, 16000, 12000, 8000, 1777, 750)

If the value entered does not correspond to any of these values, the next highest one is considered. Default frequency values are:

- SWD: STLinkV2: 4000 kHz, STLinkV3: 24000 kHz
- JTAG: STLinkV2: 9000 kHz, STLinkV3: 21333 kHz
- *Note:* JTAG frequency selection is only supported with ST-LINK firmware versions from V2J23 onward.

To connect to access port 0 the ap parameter is used in this example, so any command used after the connect command is established through the selected access port.

Note: The ST-LINK probe firmware version is shown when connecting to the device. Make sure that you have the latest version of ST-LINK firmware V2J28M17 (STSW-LINK007), available on www.st.com.

Example using SPI

STM32_Programmer_CLI -c port=SPI br=375 cpha=1edge cpol=low

The result of this example is shown in *Figure 115*.

Figure 115. Connect operation using SPI port

ST-LINK FW		U3J1H1
Voltage		0.000
Bridge freq	:	48000 KHz
Baudrate	:	375 KHz
BL version		1.1
Device ID		0×462
Device name		STM32L45×
Device type		MCU
Device CPU		Cortex-M4

Note: Make sure that the device being used supports a SPI bootloader when trying to connect through the SPI.

There are other parameters used in connection with SPI port that have default values, and some others must have specific values (see the help menu of the tool for more information).



Example using CAN

STM32_Programmer_CLI -c port=CAN br=125 fifo=fifo0 fm=mask fs=32 fe=enable fbn=2

The result of this example is shown in Figure 116.

Figure 11	16. Connect	operation	using	CAN	port

SI-LINK FW	: U3J1M1
Voltage	: 0.00U
Bridge Freq	: 48000 KHz
Baudrate	: 125 Kbps
BL version	: 2.0
Device ID	: 0x419
Device name	: STM32F42xxx/F43xxx
Device type	: HCU
Device CPU	: Cortex-M4

Note:

Not all devices implement this feature, make sure the one you are using supports a CAN bootloader.

There are other parameters used in connection with CAN port that have default values and some others must have specific values (see the help menu of the tool for more information).

Example using I2C

STM32_Programmer_CLI -c port=I2C add=0x38 br=400 sm=fast

In the connect command:

- The parameter <add> changes from a device to another, refer to AN2606 to extract the correct one. In this case, the STM32F42xxx has a bootloader address equal to 0x38.
- The baudrate parameter
> depends directly upon the speed mode parameter <sm>, for example, if sm = standard then the baudrate does not support the value 400.

The result of this example is shown in *Figure 117*.

Figure 117. Connect operation using I2C port

ST-LINK FW	:	V3J1M1
Voltage		0.000
Bridge freq	-	192000 KHz
Baudrate		400 KHz
BL version		1.1
Device ID		0×419
Device name		STM32F42xxx/F43xxx
Device type		MCU
Device CPÙ		Cortex-M4

Note: For each I2C connection operation the address parameter is mandatory.

Note: Not all devices implement this feature, make sure that the device supports an I2C bootloader.

There are other parameters used in connection with I2C port that have default values and some others must have specific values (see the help menu of the tool for more information).



UM2237 Rev 21

- Note: For the parallel programming of more than one STM32 device using multiple instances of STM32CubeProgrammer, it is mandatory to add the serial number of each device in the suitable instance, as shown in the following example:
 - "-c port=swd/usb sn=SN1" (instance 1 of STM32CubeProgrammer)
 - "-c port=swd/usb sn=SN2" (instance 2 of STM32CubeProgrammer)
 - *"-c port=swd/usb sn=SN3" (instance 3 of STM32CubeProgrammer)*

3.2.2 Erase command

-e, --erase

Description: According to the given arguments, this command can be used to erase specific sectors or the whole flash memory. This operation can take a second or more to complete, depending on the involved size.

Syntax:

[all]	Erase all sectors. EEPROM area is excluded.
[<sectorscodes>]</sectorscodes>	Erase the sectors identified by codes (for example 0,1,2 to erase sectors 0, 1 and 2). For EEPROM: ed1 & ed2.
[<[start end]>]	Erase the specified sectors starting from start code to end code, for example -e [5 10].

Example

```
./STM32_Programmer.sh --connect port=/dev/ttyS0 -e 2 4
```

This command erases only sectors 2 and 4.

Note: In the case of multiplicity of external loaders, the first selected is the one that will be taken into account during erasing of the external memory.

3.2.3 Download command

-w, --write, -d, --download

Description: Downloads the content of the specified binary file into the memory of the device. The download operation is preceded by the erase operation. A write address is needed to download binary files.

Syntax: -w <file_path> [start_address]

[file_path] Path of the file to be downloaded

[start_address] Start address of download

Example

```
-c port=COM4 -w RefSMI_MDK/All_Flash_0x1234_256K.bin 0x08008000
```

This command programs the binary file "All_Flash_0x1234_256K.bin" at address 0x08008000.

The result of this example is shown in *Figure 118*.

UM2237 Rev 21

Figure 118. Download operation



To verify that the download has been successful, call the verify option (-v or –verify) just after the write command, otherwise the verify option is ignored.

3.2.4 Download 32-bit data command

-w32

Description: Downloads the specified 32-bit data into flash memory starting from a specified address.

```
Syntax: -w32 <start_address> <32_data_bits>
```

<start_address> Start address of download.

<32_data_Bits> 32 data bits to be downloaded. Data must be separated by escape.

Example

./STM32_Programmer.sh -c port=/dev/ttyS0 br=9600 -w32 0x08000000 0x12345678 0xAABBCCFF 0x12AB34CD --verify

Note: This command makes it possible to write the 32 data bits (0x12345678, 0xAABBCCFF, 0x12AB34CD) into the flash memory starting from address 0x08000000.

3.2.5 Download 64-bit data command

-w64

Description: Downloads the specified 64-bit data into a destination address.

Syntax: -w64 <start_address> <64-bit_data>

<start_address> Start address of download. <64_data_Bits> 64-bit data to be downloaded. Data must be separated by escape.

Example:

```
/STM32_Programmer_CLI.exe -c port=swd -w64 0x08000000 0x12345678AABBCCFF
```



3.2.6 Read command

-r, --read, -u, --upload

Description: Reads and uploads the device memory content into a specified binary file starting from a specified address.

Syntax: --upload <start_address> <size> <file_path>

<start_address></start_address>	Start address of read.
<size></size>	Size of memory content to be read.
<file_path></file_path>	Binary file path to upload the memory content.

Example

```
./STM32_Programmer.sh -c port=/dev/ttyS0 br=9600 --upload
0x20007000 2000 "/local/ benayedh/Binaries/read2000.bin"
```

This command makes it possible to read 2000 bytes, starting from address 0x20007000, and uploads the content to a binary file *"/local/benayedh/Binaries/read2000.bin"*

-r32

Description: Read 32-bit data memory.

Syntax: -r32 <start_address> <size>

<start_address></start_address>	Start address of read.
<size></size>	Size of memory content to be read.

Example

./STM32_Programmer.sh -c port=SWD -r32 0x08000000 0x100



Figure 119. Read 32-bit operation

ST-LINK Firmware version : V2J28M17 SWD frequency = 4000K Connection mode: Normal Device ID: 0x450									
000100 10. 07450									
@0x08000000 :		0x20000600	0x08006BA9	0x08005ADD	0x08005ADD				
@0×08000010 :		0x08005AAA	0x08005ADD	0x08005ADD	0×00000000				
@0x08000020 :		0x00000000	0x00000000	0x00000000	0x08005ADD				
@0x08000030 :		0x08005ADD	0x00000000	0x08005AEB	0x080066E3				
@0x08000040 :		0x08005B0D	0x08005B0D	0x08005B0D	0x08005AF9				
@0x08000050 :		0x08005B0D	0x08005B0D	0x08005AF9	0x08005AF9				
@0x08000060 :		0x08005AF9	0x08005AF9	0x08005AF9	0x08003AB9				
@0x08000070 :		0x08003ACB	0x08003ADD	0x08003AF1	0x08003B05				
@0x08000080 :		0x08003B19	0x08003B2D	0x08005B0D	0x08005B0D				
@0x08000090 :		0x08005B0D	0x08005B0D	0x08005BBB	0x08005ABB				
@0x080000A0 :		0x08005AF9	0x08004689	0x08005AF9	0x08005B0D				
@0x080000B0 :		0x08005AF9	0x08005AF9	0x0800469F	0x08005B0D				
@0x080000C0 :		0x08005B0D	0x08005B0D	0x08005B0D	0x08005B0D				
@0x080000D0 :		0x08005B0D	0x080040AB	0x08005AF9	0x08005AF9				
@0x080000E0 :		0x08005AF9	0x08005B0D	0x08005B0D	0x08005AF9				
@0x080000F0 :		0x08005AF9	0x08005AF9	0x08005B0D	0x08005B0D				

The maximum size allowed with the –r32 command is 32 Kbytes.

3.2.7 Start command

-g, --go, -s, --start

Description: This command enables execution of the device memory starting from the specified address.

Syntax: --start [start_address]

[start_address] Start address of application to be executed.

Example

./STM32_Programmer.sh --connect port=/dev/ttyS0 br=9600 --start 0x08000000

This command runs the code specified at 0x08000000.

3.2.8 Debug commands

The following commands are available only with the JTAG/SWD debug port.

-rst

Description: Executes a software system reset;

Syntax: -rst

-hardRst

Description: Generates a hardware reset through the RESET pin in the debug connector. The RESET pin of the JTAG connector (pin 15) must be connected to the device reset pin. Syntax: **-hardRs**t



UM2237 Rev 21

-halt

Description: Halts the core.

Syntax: -halt

-step

Description: Executes one instruction.

Syntax: -step

-score

Description: Displays the Cortex-M core status.

The core status can be one of the following: "Running", "Halted", "Locked up", "Reset", "Locked up" or "Kept under reset"

Syntax: -score

-coreReg

Description: Read/write Cortex-M core registers. The core is halted before a read/write operation.

```
Syntax:-coreReg [<core_register>]
R0/../R15/PC/LR/PSP/MSP/XPSR/APSR/IPSR/EPSR/PRIMASK/BASEPRI/
FAULTMASK/CONTROL
```

[core_reg=<value>]: The value to write in the core register for a write operation. Multiple registers can be handled at once.

Example

-coreReg	This command displays the current values of the core registers.
-coreReg R0 R8	This command displays the current values of R0 and R8.
-coreReg R0=5 R8=10	This command modifies the values of R0 and R8.

3.2.9 List command

-I, -list

Description: This command lists all available UART, DFU and STLink interfaces.

Syntax: -1, --list

Example

./STM32_Programmer.sh --list

The result of this example is shown in *Figure 120*.


Figure 120. List of available serial ports

```
==== DFU Interface
No STM32 device in DFU mode connected
===== STLink Interface =====
 ----- Connected ST-LINK Probes List ------
ST-Link Probe 0 :
   ST-LINK SN : 002200144741500220383733
  ST-LINK FW : V3J8M3
  Access Port Number : 2
===== UART Interface =====
Total number of serial ports available: 2
Port: COM47
Location: \\.\COM47
Description: STMicroelectronics STLink Virtual COM Port
Manufacturer: STMicroelectronics
Port: COM3
Location: \\.\COM3
Description: Intel(R) Active Management Technology - SOL
Manufacturer: Intel
```

3.2.10 QuietMode command

-q, --quietMode

Description: This command disables the progress bar display during download and read commands.

Syntax: -q, --quietMode

Example

```
/STM32_Programmer.sh -c port=/dev/ttyS0 br=115200 --quietMode -w
binaryPath.bin 0x08000000
```



3.2.11 Verbosity command

-vb, --verbosity

Description: This command makes it possible to display more messages, to be more verbose.

Syntax: -vb <level>

<lp><level> : Verbosity level, value in {1, 2, 3} default value vb=1

Example

./STM32_Programmer.sh -c port=/dev/ttyS0 br=115200 -vb 3

The result of this example is shown in *Figure 121*.

```
Figure 121. Verbosity command
```





3.2.12 Log command

-log, --log

Description: This traceability command makes it possible to store the whole traffic (with maximum verbosity level) into a log file.

```
Syntax: -log [filePath.log]
```

[filePath.log] Path of log file, default is \$HOME/.STM32CubeProgrammer/trace.log.

Example

./STM32_Programmer.sh -c port=/dev/ttyS0 br=115200 -log trace.log

The result of this example is shown in *Figure 122*.

```
Figure 122. Log command
```

```
$ ./STM32_Programmer.sh -c port=/dev/ttyS0 br=115200 -log trace.log
Log output file: trace.log
Serial Port /dev/ttyS0 is successfully opened.
Port configuration: parity = none, baudrate = 115200, data-bit = 8,
stop-bit = 1.0, flow-control = off
Activating device: OK
Chip ID: 0x500
BootLoader version: 3.1
```

The log file trace.log contains verbose messages, as shown in *Figure 123*.

Figure 123. Log file content

```
16:41:19:345
Log output file: trace.log
16:41:19:368 Serial Port /dev/ttyS0 is successfully opened.
16:41:19:368 Port configuration: parity = none, baudrate = 115200, data-bit = 8,
                   stop-bit = 1.0, flow-control = off
16:41:19:368 Sending init command:
16:41:19:368 byte 0x7F sent successfully to target
16:41:19:369 Received response from target: 0x79
16:41:19:369 Activating device: OK
16:41:19:369 Sending GetID command and its XOR:
16:41:19:369 byte 0x02 sent successfully to target
16:41:19:369 byte 0xFD sent successfully to target
16:41:19:370 Received response from target: 0x79
16:41:19:370 Received response from target: 0x01050079
16:41:19:370 Chip ID: 0x500
16:41:19:370 Sending Get command and its XOR:
16:41:19:370 byte 0x00 sent successfully to target
16:41:19:370 byte 0xFF sent successfully to target
16:41:19:371 Received response from target: 0x79
16:41:19:371 Received response from target: 0x07
16:41:19:371 Received response from target: 0x07310001020311213179
16:41:19:371 BootLoader version: 3.1
```



UM2237

3.2.13 External loader command

-el

Description: This command allows the path of one or more external memory loaders to be entered, to perform programming, write, erase and read operations with an external memory.

Syntax: -el [externalLoaderFilePath1.stldr] Absolute path of external loader file.

```
-el [externalLoaderFilePath1.stldr]... -el
[externalLoaderFilePath10.stldr] Absolute path of external loader files.
```

Example 1:

```
./STM32_Programmer.sh -c port=swd -w "file.bin" 0x90000000 -v -el
"/local/user/externalLoaderPath.stldr"
```

Example 2:

./STM32_Programmer.sh -c port=swd -e all -el "/local/user/externalLoaderPath.stldr"

Example 3:

```
./STM32_Programmer.sh -c port=swd -w "file.bin" 0x90000000 -v -el
"/local/user/externalLoaderPath1.stldr"
"/local/user/externalLoaderPath2.stldr"
```

Note: This command is only supported with SWD/JTAG ports.

Note: A maximum of ten external loaders can be used.

3.2.14 External loader command with bootloader interface

-elbl

Description: This command allows to provide the path of an external memory loader used to perform programming, write, erase and read operations with an external memory using bootloader interface (only in RSS/RSSe context). This command is used only when performing SFIx process.

Syntax: -elbl [externalLoaderFilePath.stldr] Absolute path of external loader file.

Example 1:

>STM32_Programmer_CLI.exe -c port=usb1 -elbl MX25LM51245G_STM32L552E-EVAL-SFIX-BL.stldr -sfi out.sfix hsm=0 license.bin -rsse RSSe\L5\enc_signed_RSSe_sfi_jtag.bin

Note: This command is only supported with bootloader interface (UART/I2C/SPI/USB).

External loader for SFIx

The external loader for SFIx operation is aligned with the RSSe_SFI_CallNsFunction, as a result, all the functions used inside the external loader must have the same signature of this function.



```
rsse_sfi_ns_call_t
rsse_sfi_ns_call_t description in C coding language :
typedef uint32 t (*rsse_sfi_ns_call t) (void * input param);
```

As a consequence the implementation of these function inside the external loader must be slightly modified to be synchronized with input parameters.

Example of Sector erase function after modification:

```
KeepInCompilation int SectorErase (uint32_t *params)
{
    int result = 0;
    uint32_t BlockAddr;
    uint32_t EraseStartAddress = params[0];
    uint32_t EraseEndAddress = params[1];
```

3.2.15 Read unprotect command

-rdu, --readunprotect

Description: This command removes the memory read protection by changing the RDP level from level 1 to level 0.

Syntax: --readunprotect

Example

./STM32_Programmer.sh -c port=swd -rdu

3.2.16 TZ regression command

-tzenreg, --tzenregression

Description: This command removes TrustZone protection by disabling TZEN from 1 to 0.

Syntax: --tzenregression

Example

./STM32_Programmer.sh -c port=usb1 -tzenreg

Note: This command is only supported for bootloader interface and MCUs with trusted zone.

3.2.17 Option bytes command

-ob, --optionbytes

Description: This command allows the user to manipulate the device option bytes by displaying or modifying them.

Syntax: -ob [disp1] / -ob [OptByte=<value>]

[disp1]: Allows the user to display the whole set of option bytes.



[OptByte=<value>]: Allows the user to program the given option byte.

Example

./STM32_Programmer.sh -c port=swd -ob rdp=0x0 -ob displ

Note: For more information about the device option bytes, refer to the dedicated section in the programming manual and reference manual, both available on <u>www.st.com</u>.

3.2.18 Safety lib command

-sl, --safelib

Description: This command allows a firmware file to be modified by adding a load area (segment) containing the computed CRC values of the user program.

Supported formats: bin, elf, hex and Srec.

```
Syntax: -sl <file_path> <start_address> <end_address> <slice_size>
```

<file_path></file_path>	The file path (bin, elf, hex or Srec)
<start_address></start_address>	Flash memory start address
<end_address></end_address>	Flash memory end address
<slice_size></slice_size>	Size of data per CRC value

Example

STM32_Programmer_CLI.exe -sl TestCRC.axf 0x8000000 0x8010000 0x400 The result is shown in *Figure 124*.

C:\bin>STM32_Programmer_CLI.exe -sl TestCRC.axf 0x8000000 0x8010000 0x400 STM32CubeProgrammer v0.4.0-RC1 Warning: The ELF file will be overwritten CRCs area injected succesfully



The flash program memory is divided into slices, whose size is given as a parameter to the safety lib command as shown in the example above. For each slice a CRC value is computed and placed in the CRC area. The CRC area is placed at the end of the memory, as shown in *Figure 125*.



Figure 125. Flash memory mapping

The address and size of the CRCs area are determined as follows:

CRCs_Area_Size = Flash_Size / Slice_Size * 4 bytes CRCs_Start_Address = Flash_End_Address - CRCs_Area_Size



The CRC values in the CRC area are placed according to the position(s) of the user program in the flash memory, see *Figure 126*.



Figure 126. Flash memory mapping example

The address of a CRCs region inside the CRCs area is calculated as:

OPCs Start Address +	(UserProg_Start_Address – Flash_Start_Address	1 bytes
W - CICS_Start_Address	Slice_Size	4 Dyles



3.2.19 Secure programming SFI specific commands

Secure firmware install (SFI) is a feature supporting secure firmware flashing, available on some STM32 devices. The firmware provider has the possibility to protect its internal firmware against any illegal access, and to control the number of devices that can be programmed.

The protected firmware installation can be performed using different communication channels, such as JTAG/SWD or bootloader interfaces (UART, SPI and USB). For more details refer to AN5054.

-sfi, --sfi

Description: Programs an sfi file

Syntax: -sfi [<protocol=Ptype>] <.sfi file_path> [hsm=0|1]
<lic_path|slot=slotID> [<licMod_path>|slot=slotID]

[<protocol=ptype>]</protocol=ptype>	Protocol type to be used: static/live (only static protocol is supported so far), default: static.
<file_path></file_path>	Path of sfi file to be programmed.
[hsm=0 1]	Sets user option for HSM use value {0 (do not use HSM), 1 (use HSM)}, default: hsm = 0.
<lic_path slot=slotid></lic_path slot=slotid>	Path to the SFI license file (if hsm = 0) or reader slot ID if HSM is used (hsm = 1).
[<licmod_path> slot=slotID]</licmod_path>	List of the integrated SMI license files paths if HSM is not used (hsm = 0), or readers slot IDs list if HSM is used (hsm = 1). Used only in combined case, the list order must correspond to the modules integration order within the SFI file.

-rsse, --rsse

Description: This command allows the user to select the root secure services extension library (RSSe). Mandatory for devices using RSSe to make secure firmware install (SFI). The RSSe binary file can be found in STM32CubeProgrammer bin/RSSe folder.

Syntax: -rsse <file_path>

<file_path> Path of RSSe file

-a, --abort

Description: This command allows the user to clean a not properly finished process. The currently ongoing operation stops and the system returns to idle state.

Syntax: -a

3.2.20 Secure programming SFIx specific commands

Secure firmware install (SFIx) is a feature supporting secure external firmware flashing, available on some STM32 devices with OTFDEC capability. The firmware provider has the



UM2237 Rev 21

possibility to protect its external firmware/data against any illegal access, and to control the number of devices that can be programmed.

The SFIx secure programming can be carried out only with JTAG/SWD interface. For more details refer to AN5054.

-sfi, --sfi

Description: Programs an sfix file

Syntax: -sfi [<protocol=Ptype>] <.sfix file_path> [hsm=0|1]
<lic_path|slot=slotID> [<licMod_path>|slot=slotID]

[<protocol=pt< th=""><td>ype>]</td><td>Protocol type to be used: static/live (only static protocol is supported so far), default: static.</td></protocol=pt<>	ype>]	Protocol type to be used: static/live (only static protocol is supported so far), default: static.
<file_path></file_path>		Path of sfi file to be programmed.
[hsm=0 1]		Sets user option for HSM use value {0 (do not use HSM), 1 (use HSM)}, default: hsm = 0.
<lic_path slo< th=""><td>t=slotID></td><td>Path to the SFI license file (if hsm = 0) or reader slot ID if HSM is used (hsm = 1).</td></lic_path slo<>	t=slotID>	Path to the SFI license file (if hsm = 0) or reader slot ID if HSM is used (hsm = 1).
[<licmod_path< th=""><th>> slot=slotID]</th><th>List of the integrated SMI license file paths if HSM is not used (hsm = 0) or readers slot IDs list if HSM is used (hsm = 1). Used only in combined case, the list order must correspond to modules integration order within the SFI file.</th></licmod_path<>	> slot=slotID]	List of the integrated SMI license file paths if HSM is not used (hsm = 0) or readers slot IDs list if HSM is used (hsm = 1). Used only in combined case, the list order must correspond to modules integration order within the SFI file.
-elblextload	Selects a custom e interfaces	external memory-loader, only for the JTAG/SWD
<file_path></file_path>	External memory-lo	oader file path
-elblextloadbl	Selects a custom e	external memory-loader for the bootloader interface

<file_path> External memory-loader file path

-rsse, --rsse

Description: This command allows the user to select the root secure services extension library (RSSe). Mandatory for devices using RSSe to make secure firmware install (SFI). The RSSe binary file can be found in STM32CubeProgrammer bin/RSSe folder.

Syntax: -rsse <file_path>

<file_path> Path of RSSe file

-a, --abort

Description: This command allows the user to clean a not properly finished process. The ongoing operation stops and the system returns to idle state.

Syntax: -a

Note: The ExternalLoader is different for SFIx use case, as some initializations are already done by RSS, and it is marked with –SFIX at the end of the External FlashLoader name.



3.2.21 HSM related commands

To control the number of devices that can be programmed ST offers a secure firmware flashing service based on HSM (hardware secure module) as a license generation tool to be deployed in the programming house.

Two HSM versions are available:

- HSMv1: static HSM, it allows the user to generate firmware licenses for STM32 secure programming of devices selected in advance.
- HSMv2: dynamic HSM, it is an updated version of the previous one, allows the generation of firmware licenses targeting STM32 secure programming of devices chosen via personalization data at the OEM site.

Before using the HSM, it must be programmed using Trusted Package Creator, this tool can program both versions with some specific input configurations, as detailed in UM2238. For more details refer to AN5054.

-hsmgetinfo

Description: Reads the HSM available information

Syntax: -hsmgetinfo	[slot= <slotid>]</slotid>
[slot= <slotid>]</slotid>	Slot ID of the smart card reader
	Default value: slot = 1 (the PC integrated SC reader)

-hsmgetcounter

Description: Reads the current value of the license counter

Syntax: -hsmgetcounter [slot=<SlotID>]

[slot= <slotid>]</slotid>	Slot ID of the smart card reader
	Default value: slot = 1 (the PC integrated SC reader)

-hsmgetfwid

Description: Reads the Firmware/Module identifier

Syntax: -hsmgetfwid [slot=<SlotID>]

[slot=<SlotID>] Slot ID of the smart card reader Default value: slot = 1 (the PC integrated SC reader)

-hsmgetstatus

Description: Reads the current card life-cycle state

Syntax: -hsmgetstatus [slot=<SlotID>]

[slot= <slotid>]</slotid>	Slot ID of the smart card reader
	Default value: slott = 1 (the PC integrated SC reader)



-hsmgetlicense

Description: Gets a license for the current chip if counter is not null

Syntax: -hsmgetlicense <file_path> [slot=<SlotID>] [protocol=<Ptype>]

<file_path></file_path>	File path into where the received license is stored
[slot= <slotid>]</slotid>	Slot ID of the smart card reader Default value: slot = 1 (the PC integrated SC reader)
[<protocol=ptype>]</protocol=ptype>	Protocol type to be used: static/live Only static protocol is supported so far Default value: static

-hsmgetlicensefromcertifbin, -hsmglfcb

Description: Gets a license for the current certificate binary file if counter is not null. **Syntax:** -hsmglfcb <certif_file_path.bin> <license_file_path.bin> [slot=<SlotID>] [protocol=<Ptype>]

<certif_file_path.bin></certif_file_path.bin>	File path from which the input certificate is read.
<license_file_path.bin></license_file_path.bin>	File path where the received license is stored
[slot= <slotid>]</slotid>	Slot ID of the smart card reader. Default value: slot = 1 (the PC integrated SC reader)

3.2.22 STM32WB specific commands

-antirollback

Description: Perform the antirollback operation

Syntax: -antirollback

-startfus

Description: Start the FUS

Syntax: -startfus

-getuid64

Description: Read the device unique identifier (UID)

Syntax: -getuid64

-fusgetstate

Description: Read the FUS state

Syntax: -fusgetstate

-fusopgetversion

Description: Read the FUS o//perator version



Syntax: -fusgetversion

Note: FUS Operator version is not available via bootloader interfaces.

-fwdelete

Description: Delete the BLE stack firmware

Syntax: -fwdelete

-fwupgrade

Description: Upgrade of BLE stack firmware or FUS firmware.

```
Syntax: -fwupgrade <file_path> <address> [firstinstall=0|1]
[startstack=0|1] [-v]
```

<file_path></file_path>	New firmware image file path
<address></address>	Start address of download
[firstinstall=0 1]	1 for the first installation, otherwise 0 Optional, default value firstinstall=0
[startstack=0 1]	1 to start the stack after the upgrade, otherwise 0 Optional, default value <pre>startstack=1</pre>
[-v]	Verify if the download operation is completed successfully before starting the upgrade

-startwirelessstack

Description: Start the wireless stack

```
Syntax: -startwirelessstack
```

-authkeyupdate

Description: Authentication key update

Syntax: -authkeyupdate <file_path>

<file_path> Authentication key file path. This is the public key generated by STM32TrustedPackageCreator when signing the firmware using -sign command.

-authkeylock

Description: Authentication key lock

Once locked, it is no longer possible to change it using -authkeyupdate command

Syntax: -authkeylock

-wusrkey

For more information about the customer key storage, refer to already cited AN5185.

Syntax: -wusrkey <file_path> <keytype=1|2|3>

<file.path>: customer key in binary format



<keytype=1|2|3>: User key type values: 1 (simple), 2 (master) or 3 (encrypted)

-startwirelessstack

Description: Starts the wireless stack

Syntax: -startwirelessstack

Note: These commands are available only through SWD, USB DFU and UART interfaces.

Note: Under Reset mode is mandatory.

Usage example for SWD interface

- FUS upgrade: STM32_Programmer_CLI.exe -c port=swd mode=UR -ob nSWboot0=0 nboot1=1 nboot0=1 -fwupgrade stm32wb5x_FUS_fw.bin 0x080EC000 firstinstall=1
- Stack install: STM32_Programmer_CLI.exe -c port=swd mode=UR -ob nSWboot0=0 nboot1=1 nboot0=1 -fwupgrade stm32wb5x_BLE_Stack_fw.bin 0x080EC000
- User application install: STM32_Programmer_CLI.exe -c port=swd mode=UR -d UserApplication.bin 0x08000000 -v
- Note: -antirollback command is available starting from FUS v1.2.0.

3.2.23 Serial wire viewer (SWV) command

-SWV

Description: This command allows the user to access the serial wire viewer console mode, which displays the printf data sent from the target through SWO.

In this mode (see *Figure 127*) the user can start and stop the reception of the SWO data by pressing, respectively, the "R" and "S" buttons on the keyboard. The received SWO data are displayed in the console. Pressing the "E" button allows the user to exit the serial wire viewer console mode, and to terminate the reception session.





Figure 127. SWV command

Syntax: swv <freq=<frequency>> <portnumber=0-32> [<file_Path.log>]

<freq=<frequency>></freq=<frequency>	System clock frequency in MHz.
<portnumber=0-31 all></portnumber=0-31 all>	ITM port number, values: 0-31, or "all" for all ports.
[<file_path.log>]</file_path.log>	Path of the SWV log file (optional). If not specified, default is "\$USER_HOME/STMicroelectronics/STM32Programmer /SWV_Log/swv.log".
[-RA]	Option that automatically starts SWV data reception.
Example:	

Example:

STM32_Programmer_CLI.exe -c port=swd -swv freq=32 portnumber=0 C:\Users\ST\swvLog\example.log

Note: The serial wire viewer is available only through SWD interface.

Note:

Some SWV bytes can be lost during transfer due to ST-LINK hardware buffer size limitation.

-startswv

Description: This command allows the user to access the serial wire viewer console mode.

```
Syntax: startswv <freq=<frequency>> <portnumber=0-32>
[<file_Path.log>]
```

```
<freq=<frequency>>
                          System clock frequency in MHz.
```



<portnumber=0-31|all> ITM port number, values: 0-31, or "all" for all ports.

[<file_Path.log>] Path of the SWV log file (optional). If not specified, default is "\$USER_HOME/STMicroelectronics/STM32Programmer /SWV_Log/swv.log"

Example:

STM32_Programmer_CLI.exe -c port=swd -startswv freq=32 portnumber=0
C:\example.log

Figure 128. startswv command

	STM32CubeProgrammer v2.11.0	
ST-LINK SN : ST-LINK FW : Board : Voltage : SWD freq : Connect mode : Reset mode : Device ID : Revision TD :	066FFF535550755187243307 V2J33M25 NUCLEO-F446RE 3.27V 4000 KHz Normal Software reset 0x421 Rev A	
Device name : Device name : Flash size : Device type : Device CPU : BL Version :	STM32F446xx 512 KBytes MCU Cortex-M4	
Press S to Sto Press E to Ex:	op the reception it this mode	
Reception Started		
Hello world		

3.2.24 Specific commands for STM32WL

Before performing the encrypted firmware installation, set the device in its default status, i.e. with security disabled (ESE = 0x0), and all the option bytes at their default values.

STM32CubeProgrammer allows the user to perform these steps using two command lines:



1. **dsecurity**: allows the user to disable security.

Example:

STM32_Programmer_CLI.exe -c port=swd mode=hotplug -dsecurity

2. **setdefaultob**: this command allows the user to configure option bytes to their default values.

Example:

STM32_Programmer_CLI.exe -c port=swd mode=hotplug -setdefaultob

3. **-ob unlockchip**: this command allows the user to unlock the device if bad option bytes are programmed.

Example:

STM32_Programmer_CLI.exe -c port=swd -ob unlockchip

Figure 129. Output of unlockchip command

C:\Windows\System32\cmd.exe	-	• ;	×
STM32CubeProgrammer v2.10.0-004			î
ST-LINK SN : 002F004D3038510534333935 ST-LINK FW : V33542 Board : NUCLEO-NL553C Voltage : 3.27V SHD freq : 12000 KHz Connect mode: Normal Reset mode : Software reset Device ID : 0x497 Revision ID : Rev Z Device me : STM33NLxx Flash size : 256 KBytes Device CPU : Cortex-M4 BL Version : 0xc3			
UPLOADING OPTION BYTES DATA			
Bank : 0x00 Address : 10x58004020			
Size : 96 Bytes			
100%			
Bank : 0x01			
Address : 0x58004080 Size : 8 Bytes			
100%			
0x5800040C : 0x0008000 0x5800408C : 0x45670123 0x58004088 : 0x05F89A8 0x5800400C : 0x08192A38 0x5800400C : 0x08192A38 0x5800400C : 0x4550627F 0x5800400C : 0x4550627F 0x5800400C : 0x4550627F 0x58004014 : 0x08020000 0x58004014 : 0x08020000			
Reconnecting			
Heconnected 1 0x5800408 : 0x45670123 0x5800408 : 0x45670123 0x5800408 : 0x08192A38 0x5800400C : 0x08192A38 0x5800400C : 0x4C50627F 0x5800400C : 0x4FFFFFAA 0x5800402 : 0xFFFFFF00 0x5800402 : 0xFFFFFFF 0x5800403 : 0xFFFFFF00 0x5800403 : 0xFFFFFF00 0x5800403 : 0xFF 0x5800403 : 0x00 0x5800403 : 0x00			
Reconnecting Reconnected ! Warning: Apply Power Off/On to Unlock Chip			
Success to unlock chip			~

Note:

Unlockchip command is available only for STLink connection.



After the execution of these commands, go through a power OFF / power ON sequence. These two commands allow the user to unlock the board in case of inability to change option bytes using the usual method.

Figure 130 and Figure 131 show the results of these command lines.

C:\Windows\System32\cmd.exe	-		×
C:\Program Files\STMicroelectronics\STM32Cube\STM32CubeProgrammer\v2.6.0 \bin>STM32_Programmer_CLI.ex mode=hotplug -dsecurity	e-cpo	ort=swd	Î
STM32CubeProgrammer v2.6.0			
ST-I TNK SN · 002300263038511234333935			
ST-LINK FW : V3J5M2			
Board : STM32WL55C-DK			
Voltage : 3.31V			
SWD freq : 12000 KHz			
Connect mode: Hot Plug			
Reset mode : Software reset			
Device ID : 0x497			
Revision ID : Rev 1.1			
Device name : SIM32WLXX			
Flash Size : 250 KBytes			
Device Cype - nco			
Disabling Security			
Reconnecting			
ST-LINK SN : 002300263038511234333935			
ST-LINK FW : V3J5M2			
Board : STM32WL55C-DK			
Voltage : 3.32V			
SWD freq : 12000 KHz			
Connect mode: Hot Plug			
Reset mode : SortWare reset			
Device ID : 02497			
ST-LINK SN : 002300263038511234333935			
ST-LINK FW : V3J5M2			
Board : STM32WL55C-DK			
Voltage : 3.32V			
SWD freq : 12000 KHz			
Connect mode: Hot Plug			
Reset mode : Software reset			
Device ID : 0X49/			
Revenues and the second s			
Apply Source of /ON to disable the security			
			\sim

Figure 130. Disable security



C:\Windows\System32\cmd.exe	-		×
C:\Program Files\STMicroelectronics\STM32Cube\STM32CubeProgrammer\v2.6.0 ///////////////////////////////////	e-c	port=s	wd
STM32CubeProgrammer v2.6.0			
ST-LINK SN : 002300263038511234333935			
SI-LINK FW : V3J5M2 Board · STM32WIESC-DV			
Voltage : 3.31V			
SWD freq : 12000 KHz			
Connect mode: Hot Plug			
Reset mode : Software reset			
Revision ID: Rev 1.1			
Device name : STM32WLxx			
Flash size : 256 KBytes			
Device type : MCU			
Set default OB for STM32WL			
Reconnecting			
SI-LINK SN : 002300263038511234333935 ST_ITNU FM - V 23580			
Board : STM32WL55C-DK			
Voltage : 3.31V			
SWD freq : 12000 KHz			
Connect mode: Hot Plug Reset mode : Software reset			
Device ID : 6x497			
Revision ID : Rev 1.1			
Reconnected !			
Apply Power UN/OTT to set default OB for SIM32WL			
C:\Program Files\STMicroelectronics\STM32Cube\STM32CubeProgrammer\v2.6.0-A05\bin>			

Figure 131. Configure option bytes to their default values

If the user locks the board and is unable to unlock it with these two commands, there are specific scripts to unlock it. These scripts are under "../bin/STM32WLScripts", they contain a command line using –wdbg option to write directly scripts in the OPTR register.

The folder STM32Scripts contains two files and the Readme.txt:

- 1. "SetRDPLevelCM0.bat" to unlock the board via Cortex M0+
- 2. "SetRDPLevelCM4.bat" to unlock the board via Cortex M4

Note:

: If SFI command finishes with a fail, the STM32WL chip must be set in its default status using the disable security command line (-dsecurity), then the set default option byte command line (-setdefaultob).

3.2.25 SigFox credential commands

These commands are supported only for STM32WL devices.

-ssigfoxc

Description: This command allows to user to save the chip certificate to a binary file.

Syntax: -ssigfoxc <binary_file_path>

Example: STM32_Programmer_CLI.exe -c port=swd -ssigfoxc "/local/user/chip_certif.bin"



ST_LTNK SN		505565067265575458302067
ST-LINK EW		V013777
Boand		4233737
Voltago		
voltage		3.24V
SWD freq		4000 KHz
Connect mode		Normal
Reset mode		Software reset
Device ID		0x497
Revision ID		Rev 1.1
Device name		STM32WLxx
Flash size		256 KBytes
Device type		MCU
Device CPU		Cortex-M4
SigFox cer	ti	ificate File : C:\test\sigfox.bin
Data read successfully		
The Sigfox c	er	tificate file is saved successfully: C:\test\sigfox.bin
0		

Figure 132. Example of -ssigfoxc command

-wsigfoxc

Description: This command allows to user to write the chip certificate at address 0x0803E500

Syntax: -wsigfoxc <binary_file_path> <address> [The address is optional, by default is 0x0803E500]

Example 1: STM32_Programmer_CLI.exe -c port=swd -wsigfoxc "/local/user/sigfox_data.bin"0x0803E500

SigFox credential file : C:\SOFT_DOCS\KmsCreder	ntials\sigfox_data.bin
Memory Programming Opening and parsing file: sigfox_data.bin File : sigfox_data.bin Size : 48 Bytes Address : 0x0803E500	
Erasing memory corresponding to segment 0: Erasing internal memory sector 31 Download in Progress:	100%
File download complete Time elapsed during download operation: 00:00:00	.045
Verifying	
Read progress:	100%
Download verified successfully	

Figure 133. Example 1 of -wsigfoxc command



Example 2: STM32_Programmer_CLI.exe -c port=swd -wsigfoxc "/local/user/sigfox_data.h"

i ige	ite tot. Example z of Wolgi		
SigFox crede	ential file : C:\SOFT_DOCS\KmsC	Credentials\sigfox_data.h	
Memory Program Opening and pa File Size Address	ming arsing file: Sigfox_EmbKey.bin : Sigfox_EmbKey.bin : 592 Bytes : 0x0803E500	1	
Erasing memory Erasing intern Download in Pr	v corresponding to segment 0: nal memory sector 31 rogress:	100%	
File download Time elapsed o	complete during download operation: 00:0	00:00.052	
Verifying			
Read progress:		100%	
Download verit	ied successfully		

Figure 134. Example 2 of -wsigfoxc command

3.2.26 Register viewer

-regdump

Description: Reads and dumps core and MCU registers

```
Syntax: -regdump <file_path.log> [choice=<number>]
```

<file_path.log></file_path.log>	Log file path
[choice= <number>]</number>	Device number from the list of compatible devices (optional). This list is displayed if the command is performed without this optional argument.

Example: STM32_Programmer_CLI.exe -c port=swd -regdump C:\test\STM32F072.log



C:\Program Files (x86)\STMicroelectronics\STM32Cube\STM32CubeProgrammer\v2.7.0 \bin>STM32_Programmer_CLI.exe -c port= swd mode=hotplug -regdump C:\test\STM32F072.log
STN32CubeProgrammer v2.7.0
getDebugInterfaceInfo this->index = 0 ST-LINK SN : 0675FF555354885087101431 ST-LINK FN : V2332M22 Board : NUCLEO-F072RB Voltage : 3.24V SWD freq : 4000 KHz Connect mode: Hot Plug Reset mode : Software reset Device ID : 0x448 Rev1sion ID : Rev Z Device name : STM32F07x Flash Size : 128 KBytes Device type : MCU
You can automatically select a device from this list by adding the parameter "choice=‹device_number>" to the command.
Please select your device number from the list: 1. STM32F0x0 2. STM32F0x1 3. STM32F0x2 4. STM32F0x8 1
Choice: STM32F0x0.svd
Getting the registers information
Read progress: 100%
Registers information saved !

Figure 135. Read core and MCU registers

3.2.27 Hard fault analyzer

To start the analysis (see Section 2.14), use a specific command line.

Syntax: -hf

The output trace contains different kinds of essential information to better understand the reason(s) that caused a particular fault.

An informative message "STM32CubeProgrammer Fault Analyzer" is displayed to indicate that the detection flow has started.

Note: Connection to target must be established before performing Fault Analyzer command.

Example

Using the same example as GUI mode (division by 0).

Command: -c port=swd mode=hotplug -hf

From the command line output, a Green message indicates a "Hard Fault Detected" and "The processor has executed a SDIV or UDIV instruction with a divisor of 0".

Useful informations can be extracted:

- Faulty instruction address: 0x80002E4
- Faulty instruction called by a function located at this address: 0x800022D
- NVIC position: 0, Window watchdog interrupt
- Execution mode: Handler
- Core registers capture



STM32CubeProgrammer Fault Analyzer		
for Bookstone .		
Core Registers :		
n an G near - G - By \$3000000		
and reg of the other of the other of the other of the other of the other		
an a reg 2 0x000000000		
r ap 8 reg 3 0x000000020		
r ap 8 reg 4 8x86888886		
r ap 0 reg 5 0x00000000		
r ap 8 reg 6 8x00000000		
r ap 8 reg 7 8x888888888		
r ap 8 reg 8 8x888888888		
r ap 8 reg 9 8x888888888		
r ap 8 reg 18 8x888888888		
r ap 8 reg 11 8x80000000		
r ap 0 reg 12 0x00000000		
r ap 0 SP 13 0x200003E0		
r ap 0 LR 14 0xFFFFFF9		
r ap 0 PC 15 0x0800032E		
r ap 0 XPSR 16 0x21000003		
r ap 0 RSP - 0x20000310		
r ap 6 PSP - 0x00000000		
r ap 8 ck 28 8X868686868		
Execution Mode : Handlen		
CAECULOT HODE . Handler		
Usage Fault detected in instruction located at 0x08000264		
NVIC position : 0		
DIVBYZER0 : The processor has executed a SDIV or UDIV instruction with a divisor of 0.		
Hard Fault detected :		
Faulty function called at this location 0x0800022D		
Hard Fault State Register information :		
FORCED : fonced Hand Fault.		
exception return information :		
Return to Thread mode, exception return uses non-floating-point		
state from NGD and execution uses MCD after naturn.		
search for the execution uses has after recurn.		

Figure 136. Fault analyzer CLI view when Hard Fault is detected

3.2.28 RDP regression with password

Some STM32 products (for example those of the STM32U5 series) offer the possibility to use an optional password-based RDP level regression, including RDP Level 2.

Detailed information about this hardware mechanism is available in reference manuals. Password lock and unlock CLI commands are:

- lockRDP1

Description: Allows the user to lock the RDP regression from level 1 with a password.

Syntax: - lockRDP1 <Password first 32 bits> <Password second 32 bits> Example:

STM32_Programmer_CLI -c port=swd mode=hotplug -lockRDP1 0x12345678 0xDEADBEEF



- lockRDP2

Description: This command allows the user to lock the RDP regression from level 2 with a password.

Syntax: - lockRDP2 <Password first 32 bits> <Password second 32 bits>
Example:

STM32_Programmer_CLI -c port=swd mode=hotplug - lockRDP2 0x12345678 0xDEADBEEF

- unlockRDP1

Description: This command allows to unlock the RDP regression from level 1 with a password.

Syntax: - unlockRDP1 <Password first 32 bits> <Password second 32 bits>

Example:

STM32_Programmer_CLI -c port=swd mode=hotplug -unlockRDP1 0x12345678 0xDEADBEEF

- unlockRDP2

Description: This command allows the user to unlock the RDP regression from level 2 with a password.

Syntax: - unlockRDP2 <Password first 32 bits> <Password second 32 bits>

Example:

STM32_Programmer_CLI -c port=swd mode=hotplug - unlockRDP2 0x12345678 0xDEADBEEF

- *Note:* After unlocking the RDP, the user must perform an RDP regression, as the listed commands do not include the RDP regression operation.
- Note: To remove RDP regression with password, the user must use the Lock command and a password with value 0xFFFFFFF 0xFFFFFFF, such as **STM32_Programmer_CLI -c** port=swd mode=hotplug -lockRDP1 0xFFFFFFFF 0xFFFFFFFF.

3.2.29 GetCertif command

-gc

Description: This command allows the user to read the chip certificate.

Syntax: -gc certification.bin

3.2.30 Write DBG MCU authentication command

-w32dbgmcu

Description: Downloads the specified 32-bit data into the DBGMCU AUTH HOST register to be able to place a message in the mailbox shared between the device and the host. **Syntax:** -w32dbgmcu <32_data_bits>

UM2237 Rev 21



Example:

-w32dbgmcu 0x12345678

Only STM32H5 devices support this command, use the verbosity to check the message, DBG MCU address, and the verification process for write trace.

Note: After the upcoming reset, the device is able to interpret the message.

3.2.31 OBKey provisioning (STM32H573/STM32H563/STM32H562 only)

-sdp

Description: This is a security feature to program OBKey content. **Syntax:** -sdp [OBKey_File_Path.obk]

[OBKey File Path.obk] Path of OBK file

Example:

/STM32_Programmer_CLI.exe -c port=swd mode=hotplug -sdp "C:\Program Files\STMicroelectronics\STM32Cube\STM32CubeProgrammer_DA_password\bin\DA_ Default_Config\NonCrypto\DA_Config_Certificate.obk"

Figure 137. OBKey provisioning example

SND freq : 8000 KHz
Connect mode: Hot Plug
Reset mode : Software reset
Device ID : 0x484
Revision ID : Rev Z
Device name : STM32H5xx
Flash size : 2 MBytes
Device type : MCU
Device CPU : Cortex-M33
BL Version : 0xE2
Secure Data Provisioning Start. OBK Input file : C:\Program Files\STMicroelectronics\STM32Cube\STM32CubeProgrammer_DA_password\bin\DA_Default_Config\WonCrypto\DA_Config_Certificate.obk
DBKey Provisioned successfully

OBKey file generation is managed by STM32 Trusted Package Creator.

3.2.32 Password provisioning (STM32H503 only)

-pwd

Description: This command provisions the password in OTP, and generates a password.bin file, to be used later for regression.

Syntax: -pwd value=[Password_Value] path=[Password_Path]

Password value	Value that will be programed in OTP
Password path	Location where to save "password.bin" file

Example:

STM32_Programmer_CLI.exe -c port=swd -pwd value=1mc41 path=C:\my_folder

The password size must be between 4 and 16 bytes.

Once the target is successfully provisioned, the "password.bin" file is generated, to be used while performing debug authentication.

Password programming can be executed only once for each target.



3.2.33 Debug authentication commands (STM32H5 series only)

The following commands are available only with the JTAG/SWD debug port.

[Debugauth=<value>]

Discovery: *debugauth=2*

Launches discovery, to display information about the target.

Example:

/STM32_Programmer_CLI.exe -c port=swd debugauth=2

Figure 138. Discovery log

Start Debug Authentication Se	quer	ice			
SDM 0.6.0 Init Sequence					
Open SDM Lib					
open_comms		434 :	: open		Asserting target reset
open_comms		438 :	: open		Writing magic number
open_comms		446 :	: open		De-asserting target reset
open_comms		492 :	: open		Communication with the target established successfully
response_packet_lock					
discovery: target ID			:0x48	4	
discovery: SoC ID			:0x0	0x0	0x0 0x0
discovery: SDA version			:1.0.	3	
discovery: Vendor ID			:STMi	croe	lectronics
discovery: PSA lifecycle			:ST_L	IFEC	YCLE_PROVISIONING
discovery: PSA auth version			:1.0		
discovery: ST HDPL1 status			:0xff	ffff	ff
discovery: ST HDPL2 status			:0xff	ffff	ff
discovery: ST HDPL3 status			:0xff	ffff	ff
discovery: Token Formats			:0x20	9	
discovery: Certificate Format	s		:0x20	1	
discovery: cryptosystems			:Ecds	a-P2	56 SHA256
discovery: ST provisioning in	tegr	ity st	tatus:0xea	eaea	ea
Debug Authentication: Discove	ry S	uccess			

Authentication: *debugauth=1*

Performs debug authentication using credential files.

Credential files (passwords, keys, and certificates) are generated by STM32 Trusted Package Creator.

Syntax: pwd=[password_path.bin] debugauth=1 (authentication with password)

Example:

STM32_Programmer_CLI.exe -c port=swd pwd=C:\password.bin debugauth=1



Figure 139. Debug authentication with password	Figure 139.	Debug	authentication	with	password
--	-------------	-------	----------------	------	----------

Start Debug Authentication Se	quen	ce		
SDM 0.6.0 Init Sequence Open SDM Lib				
open_comms		434	open	Asserting target reset
open_comms		438	open	Writing magic number
open_comms		446	open	De-asserting target reset
open_comms		492	open	Communication with the target established successfully
[00%] discovery command [10%] sending discovery comm [20%] receiving discovery response_packet_lock [40%] loading credentials [50%] sending challenge requ [60%] receiving challenge response_packet_lock [70%] signing token SDMAuthenticate	nand uest	1131	client	Found 3 certificates
response_packet_lock response_packet_lock response_packet_lock [80%] sending respose [90%] receiving status response_packet_lock SDMAuthenticate		1229	client	Authentication successful
[100%] finished authentication	on			
Debug Authentication Success				

Syntax: per=[Permission] key=[Key_Path.pem] cert=[Certificate_Path.b64] debugauth=1 (authentication with certificate)

Permission: the possible values to be chosen by the user are:

- "a": Full regression
- "b": Partial regression.
- "c": Debug secure level 3
- "d": Debug secure level 2
- "e": Debug secure level 1
- "f": Debug Non-secure level 3
- "g": Debug Non-secure level 2
- "h": Debug Non-secure level 1

Example:

```
STM32_Programmer_CLI.exe -c port=swd per=a key=C:\key_3_leaf.pem
cert=C\cert_leaf_chain.b64 debugauth=1
```



Figure 140.	Debug au	thentication	with	certificate
-------------	----------	--------------	------	-------------

Start Debug Authentication Se	quen	ce		
SDM 0.6.0 Init Sequence Open SDM Lib				
open_comms		434 :	open	Asserting target reset
open_comms		438 :	open	Writing magic number
open_comms		446 :	open	De-asserting target reset
open_comms		492 :	open	Communication with the target established successfully
<pre>[00%] discovery command [10%] sending discovery com [20%] receiving discovery response_packet_lock [40%] loading credentials [50%] sending challenge req [60%] receiving challenge response_packet_lock SDMAuthenticate</pre>	mand uest :	1131 :	client	Found 1 certificates
[80%] sending ST password [90%] receiving response response_packet_lock [100%] authentication succes SDMAuthenticate	sful :	1195 :	client	Authentication successful
Debug Authentication Success				

3.2.34 Force no debug authentication command

--force_no_da

Description: This option allows to pass an information to the tool, to force the product state to OB programming, even if the debug authentication is not configured (password not programmed in OTP). In this case, it is no longer possible to perform regression, all debug features are disabled.

This option is available only for STM32H50x devices, it is handled only if there is a request for OB programming.

Syntax: --force_no_da

Prompt a warning message to highlight the case:

- > You are trying to modify the PRODUCT_STATE while OTP are not set,
- > Force No DA option is active!

If this option is not used and you are trying to modify the Product State with OTP not configured, to avoid damages the tool stops the execution and prompts an error message:

> You are trying to modify the PRODUCT_STATE while OTP are not set, data won't be downloaded.

> Please configure your device and try again.





UM2237

4 STM32CubeProgrammer user interface for MPUs

4.1 Main window

Pro STM32	2CubeProgrammer											
STM32 Cube	r 🕥 Programmer									(19)	f 🕨	y 🛧 🏹
≡	Memory & Fi	le editior	ı									Connected
	Device memory	Open fi	ile +								USB	 Disconnect
	Address		• Size			Data width	32-bit 👻		Read	*	U	SB configuration
	Partition	Id	Size	F	Rights						POIL	USBI 👻 🖸
OB	Partition0	0×0	256	KB F	RW						Serial number	001C00303338511934383
	FSBL	0×1	1	MB F	RW							
	Partition2	0x2	1	MB F	RW							
	Partition3	0x3	16	MB F	RW							
	Partition4	0x4	16	MB F	RW						-	
	virtual	0xf1	512	B F	र							
	Log							Verbosity level	1 2	3	-	
	19:22:09 : ST	M32CubePro	grammer API	v2.0.0						Å		
_	19:22:13 : 05 19:22:13 : Ma	nuf. ID :	STMicroelectr	nonics	s) 	0. (0.15.00 @Den.ii	TD /0-0000				D	evice information
	19:22:13 : Pr 19:22:13 : SN	: 001C00	0333851193438	5de @Dev 83330	nce IL) /0x500, @Rev1	sion 10 /0x0000				Device	STM32MPxxx
	19:22:13 : De	vice ID :	0x0500								Type	MPU
(I)											Device ID	0x500
$\underline{}$											Flash size	
(?)											CPU	- Cortex-A7
										\otimes		

Figure 141. STM32CubeProgrammer main window

The main window allows the user to select the interface used to connect to STM32MP1 BootROM, possible interfaces are USB-DFU and UART (programming through stlink interface is not possible with STM32MP1 series). Once connected (using connect button) available partitions are displayed, the user is able to open a TSV file for programming.



4.2 **Programming windows**

					-					1
M STM3	2CubePro	gramm	er							
STM32	rogrami	mer							🐵 f	🕨 y 🔆 🏹
	Memo	ory &	File eo	lition						Connected
	Device	mem	ory Fla	shLayout_sdcard_stm3	2mp157c-ev1-trus	tedTSV ×	+		USB	 Disconnect
	Address	s		~ Size		Data width		Download 👻		USB configuration
	Select	Opt	Id	Name	Туре	IP	Offset	Binary	Port	USB1 🗸 🕤
		-	0x1	fsbl1-boot	Binary	none	0x00000000	tf-a-stm32mp157c-ev1-trusted.stm32	Serial num	ber 001C00303338511934383330
		-	0x3	ssbl-boot	Binary	none	0x00000000	u-boot-stm32mp157c-ev1-trusted.stm32		
		Р	0x4	fsbl1	Binary	mmc0	0x00004400	tf-a-stm32mp157c-ev1-trusted.stm32		
		Р	0x5	fsb12	Binary	mmc0	0x00044400	tf-a-stm32mp157c-ev1-trusted.stm32		
		Р	0×6	ssbl	Binary	mmc0	0x00084400	u-boot-stm32mp157c-ev1-trusted.stm32		
		Р	0x11	bootfs	System	mmc0	0x00284400	<pre>st-image-bootfs-stm32mp-valid-stm32mp1.ext4</pre>		
		Р	0x12	vendorfs	FileSystem	mmc0	0x04284400	st-image-vendorfs-stm32mp-valid-stm32mp1.ext4		
		Ρ	0x13	rootfs	FileSystem	mmc0	0x05284400	st-image-core-stm32mp-valid-stm32mp1.ext4		
		Р	0x14	userfs	FileSystem	mmc0	0x340F0400	st-image-userfs-stm32mp-valid-stm32mp1.ext4		
	-								Ų	
	< [
	Binaries	s path	C:\S	TM32MP1_Binaries				Browse	_	
	Log							Verbosity level 💿 1 💿 2 💿 3		
	19:22	:09 :	STM32C	beProgrammer API v	2.0.0			A	8	
	19:22	13	Manuf.	ID : STMicroelectro ID : DFU in HS Mo	BUMBIT/S) onics de @Device TD /Ox	500. @Revi	sion TD /0x000	0		
3	19:22 19:22	13 1	SN : 00 FW vers	1C0030333851193438 sion : 0x0110	3330]	Device information
W	19:22	:13 :21	Device Read TS	ID : 0x0500 SV File: X:\Wildcat	\binCut2\FlashLay	vout_sdcard	_stm32mp157c-e	v1-trustedTSV	Dèvice	SIM32MPxxx MDI
	13:24	. 2.2 :	wamber	or parciclons: 9					Device ID	0x500
\sim									Flash size	_
(?)									CPU	Cortex-A7

I Igule 142. 13V programming window

To perform TSV files programming the user must perform the following operations:

- Open a TSV file by using "Open file" tab, if TSV file format is correct then TSV content is displayed in the main window. TSV Files are available in STM32MP1 Linux distributions, refer to STM32MP1 wiki for more details.
- Specify binaries path in "Binaries path" text box.
- Select the list of partitions to be programmed in "select" column, by default all partitions are selected.
- Launch download using "Download" button.

For more details concerning flashing operations refer to AN5275, available on *www.st.com*.

4.3 OTP programming window

The OTP window is available exclusively for MPUs. It extracts the OTP partition [ID 0xF2] using u-boot interface to read, display and fuse the OTP registers.

The STM32MP13xx devices have 3072 OTP (one time programmable) bits, which can be read-accessed in 96 words: OTPx (x = 0 to 95). Some OTP words are programmed by STMicroelectronics at factory (product differentiations or keys).



The UI has six main goals:

- 1. Get OTP structure information
- 2. Read and display words in table view format
- 3. Edit and fuse OTP registers
- 4. Lock specific/all words
- 5. Program binary files (frequently for keys fuse)
- 6. Save the OTP partition in output binary file (for debug)

Open the OTP window by pressing the "OTP" button from the main window to start reading the OTP partition, when this is correctly executed, it displays the words in table view.

Pro STM	132CubeProgramm	er						- 🗆 ×
STM32 Cube	Programmer					(f 🕨	y 🔆 🏹
	OTP MPU						<u>.</u>	Connected
	Structure versio	on: 2 Global st	ate: 0x0000000	State:	Secure Open	1	USB	 Disconnect
.	Download file	5		Browse	Start word	D Lock region Program	U Port	SB configuration USB1
ОТР	Name	Word	Value	Status	Lock	Description	Serial number	r 801880023530510438343
لتتقا	CFG0	OTP0	0x00000017	0x30000000		Shadow write sticky lock Shadow read sticky lock	PID	0xdf11
	► CFG1	OTP1	0x00008800	0x10000000		Shadow write sticky lock	VID	0x0483
	CFG2	OTP2	0xD0100000	0x18000000		Sticky programming lock Shadow write sticky lock	Read Unprote	et (MCU)
	► CFG3	OTP3	0x0000000	0x00000000		-	TZEN Regress	ion (MCU)
	CFG4	OTP4	0x00000000	0x08000000	4	Sticky programming lock		
	CFG5	OTP5	0x0000000	0x0000000		-		
	CFG6	OTP6	0x00000000	0x00000000		2+3		
	CFG7	OTP7	0x0000000	0x00000000				
	CFG8	OTP8	0x00000010	0x10000000		Shadow write sticky lock		
	<				_	· · · · · · · · · · · · · · · · · · ·		
	Save OTP par	rtition 6				Lock all Apply Read		
	Log				Live	e Update Verbosity level 💿 1 🔵 2 🔵 3	т	arget information
	12:38:20 : Partit 12:38:20 : Size 12:38:20 : Uploa 12:38:20 : OTP P	tion : UXF2 : 1024 Bytes ding OTP data: artition read successfully) 4	Board Device Type Device ID Revision ID Elath size	 STM32MP13xx MPU 0x501
?						100% 🛞	CPU Bootloader V	Cortex-A7 ersion

Figure 143. OTP MPU window

4.3.1 Get OTP structure information

This section displays an overview of the connected device.

- Structure version
- Global state
- State: Secure open / Secure closed / Invalid

Note: The OTP UI is available only for structure version 2.



4.3.2 Read and display words

After checking of the OTP partition, the tool decodes and shows the words in table view format including five columns:

- Name: OTP word name.
- Word: OTP word ID as a tree component.
- Value: hexadecimal value (red color for locked words) as editable field.
- Status: hexadecimal value.
- Lock: indicate the lock state of the considered word as checkbox component. Checked if the item is locked, otherwise it will be unchecked. The column is disabled if the item is in permanent write lock state.
- Description: decodes the status of the OTP word and displays a brief description for children's items.

Note: Press "Read" button to refresh all table fields or to discard changes.

4.3.3 Edit and fuse words

It is possible to directly edit the "Value" table cell to write a new value (press Enter after each change). The tool verifies the syntax of the input item to respect hexadecimal format and item width, then it checks the locking state of the current item before start fusing.

Name	Value	Status	Lock			Name	Value	Status
Global	0x00000000			RI	Global		0x00000000	
ОТРО	0x00000017	0x30000000		Sł	ОТРО		0x00000017	0x30000000
V OTP1	0x00008800	0x50000000		Sł	► OTP1		0x00008800	0x50000000
qspi_not_default_af	0×1			0:	OTP2		0xD0100012	0x58000000
emmc_if_id	Frror			× P:	• OTP3	Error	0.0000000	0.0000000
sd_if_id				D:	► OTP4	Enor		
no_cpu_pll	OTP1 : [qs	pi_not_default_af] register is loc	ked !	D:	▶ OTP5	X	OTP2 register is locked !	
no_usb_dp_pullup	Edition is r	not permitted for this word !		D:	OTP6		Edition is not permitted for	this word !
uart_instances_disab			ОК	P	► OTP7			ОК
no_data_cache	0/1			b:	· OIF/			

Figure 144. Edit denial for locked words

After modification, press "Apply" button and confirm the operation to start the update and refresh the table view.

Con	Tirmation	
?	This operation cannot be reverted and r device Do you confirm ?	may damage you

Figure 145. Program Apply confirmation



4.3.4 Lock specific/all words

The lock operation allows the user to close the write programming against any modification of the considered OTP word. It is possible to lock several words to lock on one-shot by setting the assigned checkboxes, then clicking on "Apply" to start the operation.

Note: To go faster to the initial OTP lock state, it is recommended to press the "Read" button.

It is possible to lock all words at once by setting the "Lock all" checkbox. A message pops up to emphasize the procedure, which may result in all words being closed and no further changes allowed.



Prg Confirm	nation		×
?	Lock all OTP words is checke Are you sure ?	ed	
		ОК	Cancel

If the operation is successful, the table view is disabled for all "Lock" columns.

Figure 147. All OTP fields are locked

711 STM	a Shūtchdhagannar – D X											
STM 2												
≡	OTP MPU					Connected						
	Structure vers	ion: 2 Global state:	0x0000000	State: Secure Op	en	US8 👻 Disconnect						
	Download file					Browse Start word ID Lock region Program						
					_	Port US81 V 0						
	Name	Word	Value	Status	Lock	ck Description						
ОТР	XK26	OTP58	0xE710EB3C	0x40000000		Permanent write lock						
	XK27	OTP59	0xEC3CE37A	0x40000000		Permanent write lock						
	XK28	OTP60	0x16351203	0x40000000		Permanent write lock 1779/ Revenues MCD						
	XK29	OTP61	0x000000AA	0x40000000		Permanent write lock						
	XK30	OTP62	0x000000AA	0x40000000		Permanent write lock						
	XK31	OTP63	0x0000000B	0x40000000		Permanent write lock						
	XK32	OTP64	0x00000000	0x40000000		Permanent write lock						
	XK33	OTP65	0xBABABABA	0x40000000		Permanent write lock						
	XK34	OTP66	0x0000CACA	0x40000000		Permanent write lock						
	XK35	OTP67	0x0000DADA	0x40000000	\checkmark	Permanent write lock						
	XK36	OTP68	0x0000DEAD	0x40000000	\checkmark	Permanent write lock						
	XK37	OTP69	0x0000ABCD	0x40000000	\checkmark	Permanent write lock						
	XK38	OTP70	0x0000AABB	0x40000000	\checkmark	Permanent write lock						
	XK39	OTP71	0xCACACACA	0x40000000		Permanent write lock						
	XK40	OTP72	0x0ABCDE00	0x40000000		Permanent write lock						
	XK41	OTP73	0x0000FFAA	0x40000000		Permanent write lock						
	XK42	OTP74	0x00000022	0x40000000		Permanent write lock						
	Save OTP p	artition				Lock all 🗸 Apply Read						
	Log					ive Update Verbosity level						
	12:05:21 : Flast 12:05:21 : Flast 12:05:34 : UPU 12:05:34 : Par 12:05:34 : Siz 12:05:34 : OTP	In the second se				Target information Target information Target information Torket To						
(?)						Cortex+A7 100% (x) Bootlaader Version ···						



4.3.5 **Program binary file**

This section contains different graphical components, needed to program a binary file on the OTP registers starting from a word ID, and based on the following steps:

- 1. Choose the adequate binary file (with .bin extension) by clicking on "Browse"
- 2. Mention the start word ID in decimal format (0 to 95)
- 3. Check/Uncheck the "Lock region" checkbox to indicate the operation type (update or write permanent lock)
- 4. Press "Program" button to start the download flow

If the procedure is completed correctly, an informational dialog appears to confirm that the operation is completed.

Note: The input binary is a 32-bit aligned file, the tool adds padding values if the file is not aligned (a warning message is displayed in the log panel).

Structure version: 2	Global state: 0x000	000000 State: Sec	ure Open			
Download file C:\User	rs\oueslats\Desktop\publicK	KeysH Browse S	tart word ID 24	Lock region	Program	
Name	Prg Message	e		×	Description	
Global	0x000(OTP file programming don	e successfully		^	
ОТРО	0x0000			e sticky lock Shadow read s		
► OTP1	0x0000			OK te sticky lock	Permanent writ	
OTP2	0xD0100010	0x58000000		Sticky programming lock	Shadow write	

Figure 148. Download binary file

4.3.6 Save OTP partition

This capability allows the user to save the whole current OTP partition in a binary file (.bin extension), which can be used for future analysis or to archive the current device configuration.

Press on "Save OTP partition" button and choose the desired output name and directory (check permissions).

If the save is completed correctly, an informational dialog appears to confirm that the operation is completed.

Prg Messag	×	
i	Save OTP partition done successfully	
		ОК

Figure 149. Save OTP partition

Note:

The size of the output file must be 1024 bytes.

U-boot program must be installed before launching OTP window, which is necessary to expose the OTP partition.

Words 32 to 95 do not have child fields, can be edited only once, and must be permanently



locked after programming.

Word editing and Lock operation can be performed at the same time, after clicking "Apply".



5 STM32CubeProgrammer CLI for MPUs

5.1 Available commands for STM32MP1

This section details the commands supported on STM32MP1 devices.

5.1.1 Connect command

-c, --connect

Description: Establishes the connection to the device. This command allows the host to open the chosen device port (UART/USB)

```
Syntax: -c port=<Portname> [noinit=<noinit_bit>] [br=<baudrate>]
[P=<Parity>] [db=<data_bits>] [sb=<stop_bits>] [fc=<flowControl>]
```

Interface identifier:			
 – ex COMx (for Windows) 			
 /dev/ttySx (for Linux) 			
 usbx for USB interface 			
Sets No Init bits, value in {0,1}, default 0.			
Noinit = 1 can be used if a previous connection is active (no need to send $0x7F$).			
Baudrate, (for example 9600, 115200), default 115200.			
Parity bit, value in (EVEN, NONE, ODD), default EVEN.			
Data bit, value in (6, 7, 8), default 8.			
Stop bit, value in (1, 1.5, 2), default 1.			
Flow control, value in (OFF, Software, Hardware). Software and Hardware flow controls are not yet supported for STM32MP1 series, default OFF.			

Example

Using UART:

./STM32_Programmer.sh -c port=/dev/ttyS0 p=none The result of this example is shown in *Figure 150*.


Figure 150. Connect operation using RS232

```
STM32CubeProgrammer v1.0.2

Serial Port COM1 is successfully opened.

Port configuration: parity = none, baudrate = 115200, data-bit = 8,

stop-bit = 1.0, flow-control = off

Activating device: OK

Chip ID: 0x500

BootLoader protocol version: 4.0

Device name: STM32MPxxx

Device type: MPU

Device CPU : Cortex_A7
```

- *Note:* When using the USB interface, all the configuration parameters (such as baudrate, parity, data-bits, frequency, index) are ignored.
- *Note:* To connect using UART interface, the port configuration (baudrate, parity, data-bits, stop-bits and flow-control) must have a valid combination.

5.1.2 GetPhase command

-p, --phaseID

Description: This command allows the user to know the next partition ID to be executed.

Syntax: --phaseID

Example

./STM32_Programmer.sh -c port=/dev/ttyS0 p=none br=115200 --phaseID

5.1.3 Download command

-w, --write, -d, --download

Description: Downloads the content of the specified binary file into a specific partition in the flash or SYSRAM memories.

Syntax: -w <file_path> [partitionID]

[file_path] File path to be downloaded (bin, stm32, vfat, jffs2, ubi, ext2/3/4 and img file extensions).

[partition_ID] Partition ID to be downloaded.

Example

./STM32_Programmer.sh -c port=/dev/ttyS0 p=none -d atf.stm32 0x01

This command allows the user to download the atf binary file at Atf partition (partition ID: 0x01).

The result of this example is shown in *Figure 151*.



Figure 151. Download operation

Memory Programming File : atf.s Size : 22521 Partition ID : 0x01	tm32 5 Bytes	
Download in Progress:		100%
File download complete Time elapsed during the	download operation is:	00:00:22.690

Note:

For U-boot with USB interface, to program the nonvolatile memory (NVM) with the loaded partition using download command, the user must execute a start command with the partition ID. Besides, to execute an application loaded in the NVM, the start address. must be specified

Example: Download and manifestation on alternate 0x1

./STM32_Programmer.sh -c port=usb0 -w atf.stm32 0x1 -s 0x01

5.1.4 Flashing service

Description: The embedded flashing service aims to load sequentially the partitions requested by the bootloader. To do this STM32CubeProgrammer needs the TSV file, which contains information about the requested partitions to be loaded.

STM32CubeProgrammer downloads and starts the requested partition ID until the end of operation (phaseID = 0xFE).

```
Syntax: -w < tsv file_path >
```

<tsv file_path> Path of the tsv file to be downloaded.

#Opt	Id	Name	Туре	IP	Offset	Binary
-	0x01	fsbl1-boot	Binary	none	0x0	tf-a-stm32mp157c-dk2-trusted.stm32
-	0x03	ssbl-boot	Binary	none	0x0	u-boot-stm32mp157c-dk2-trusted.stm32
P	0x04	fsbl1	Binary	mmc0	0x00004400	tf-a-stm32mp157c-dk2-trusted.stm32
P	0x05	fsbl2	Binary	mmc0	0x00044400	tf-a-stm32mp157c-dk2-trusted.stm32
P	0x06	ssbl	Binary	mmc0	0x00084400	u-boot-stm32mp157c-dk2-trusted.stm32
P	0x21	bootfs	System	mmc0	0x00284400	st-image-bootfs-openstlinux-weston-extra-stm32mp1.ext4
P	0x22	vendorfs	FileSystem	mmc0	0x04284400	st-image-vendorfs-openstlinux-weston-extra-stm32mp1.ext4
P	0x23	rootfs	FileSystem	mmc0	0x05284400	st-image-weston-openstlinux-weston-extra-stm32mp1.ext4
P	0x24	userfs	FileSystem	mmc0	0x340F0400	<pre>st-image-userfs-openstlinux-weston-extra-stm32mp1.ext4</pre>

Example

./STM32_Programmer.sh -c port=/dev/ttyS0 p=none br=115200 -d Flashlayout.tsv

Note:

While programming the Flashlayout.tsv file, U-boot can spend a long time to start correctly, for this reason configure the timeout value by using the timeout command (-tm <timeout>).

5.1.5 Start command

-g, --go, -s, --start

Description: This command allows executing the device memory starting from the specified address.



Syntax: --start [start_address/Partition_ID]

[start_address] Start address of application to be executed. If not specified with STM32MP and UART interface, last loaded partition is started.

[Partition_ID] This parameter is needed only with STM32MP devices. It specifies the partition ID to be started.

Example

./STM32_Programmer.sh --connect port=/dev/ttyS0 p=none br=115200 --start 0x03

This command allows the user to run the code specified at partition 0x03.

Note: For U-boot with USB interface, to program the NVM with the loaded partition using download command, you need to execute a start command with the partition ID. To execute an application loaded in the NVM, you need to specify the start address.

Example 1: Download and manifestation on alternate 0x1

./STM32_Programmer.sh -c port=usb0 -w atf.stm32 0x01 -s 0x01

Example 2: Execute code at a specific address

./STM32_Programmer.sh -c port=usb0 -s 0xC0000000

5.1.6 Read partition command

-rp, --readPart

Description: Reads and uploads the specified partition content into a specified binary file starting from an offset address. This command is supported only by U-boot.

Syntax: --readPart <partition_ID> [offset_address] <size>
<file_path>

<partition_ID> Partition ID

[offset_address] Offset address of read

<size> Size of memory content to be read

<file_path> Binary file path to upload the memory content

Example:

./STM32_Programmer.sh -c port=/dev/ttyS0 p=none br=115200 -rp 0x01 0x200 0x1000 readPart1.bin

This command allows the user to read 0x1000 bytes from the sebl1 partition at offset address 0x200 and to upload its content to a binary file "readPart1.bin"

5.1.7 List command

-I, -list

Description: This command lists all available communication interfaces UART and USB.

Syntax: -1, --list <interface_name>

<uart/usb>: UART or USB interface



UM2237 Rev 21

Example:

./STM32_Programmer.sh -list uart

5.1.8 QuietMode command

-q, --quietMode

Description: This command disables the progress bar display during Download and Read partition commands.

Syntax: -q, --quietMode

Example:

```
./STM32_Programmer.sh -c port=/dev/ttyS0 p=none br=115200 --quietMode -w
binaryPath.bin 0x01
```

5.1.9 Verbosity command

-vb, --verbosity

Description: This command allows the user to display more messages, to be more verbose.

Syntax: -vb <level>

<level> : Verbosity level, value in {1, 2, 3} default value vb=1

Example:

./STM32_Programmer.sh -c port=/dev/ttyS0 p=none br=115200 -vb 3

5.1.10 Log command

-log, --log

Description: This traceability command allows the user to store the whole traffic (with maximum verbosity level) into log file.

Syntax: -log [filePath.log]

[filePath.log] : path of log file (default is \$HOME/.STM32CubeProgrammer/trace.log)

Example:

./STM32_Programmer.sh -c port=/dev/ttyS0 p=none br=115200 -log trace.log

This command generates a log file "trace.log" containing verbose messages (see an example in *Figure 153*).



Figure 153. Log file content

```
16:41:19:345
Log output file: trace.log
16:41:19:368 Serial Port /dev/ttyS0 is successfully opened.
16:41:19:368 Port configuration: parity = none, baudrate = 115200, data-bit = 8,
                   stop-bit = 1.0, flow-control = off
16:41:19:368 Sending init command:
16:41:19:368 byte 0x7F sent successfully to target
16:41:19:369 Received response from target: 0x79
16:41:19:369 Activating device: OK
16:41:19:369 Sending GetID command and its XOR:
16:41:19:369 byte 0x02 sent successfully to target
16:41:19:369 byte 0xFD sent successfully to target
16:41:19:370 Received response from target: 0x79
16:41:19:370 Received response from target: 0x01050079
16:41:19:370 Chip ID: 0x500
16:41:19:370 Sending Get command and its XOR:
16:41:19:370 byte 0x00 sent successfully to target
16:41:19:370 byte 0xFF sent successfully to target
16:41:19:371 Received response from target: 0x79
16:41:19:371 Received response from target: 0x07
16:41:19:371 Received response from target: 0x07310001020311213179
16:41:19:371 BootLoader version: 3.1
```

5.1.11 OTP programming

Description: These commands allow the user to program the OTP from a host computer. Their functionality (such as downloading or uploading a full OTP image, modifying an OTP value or proprieties) is explained below.

Note: The following commands are not supported in JTAG/SWD debug port connection mode.

- Loading shadow registers values to the tool For load operation, the host requests the OTP partition data and the platform replies with the structure described on https://wiki.st.com/stm32mpu/index.php/STM32CubeProgrammer_OTP_management.
- Writing the modified shadow registers to the target
 This operation is executed by performing the following sequence:
 - a) The user types in the value and the status of each chosen OTP shadow register.
 - b) The tool updates the OTP structure with the newly given OTP shadow registers values and status.
 - c) The tool proceeds with sending the updated structure, with bit0 in the "Write/read conf" field set to 0 ("Write/read conf" is word number 7 in the OTP structure).
 - d) Once the structure is sent, the shadow register values are reloaded to update the OTP structure in the tool.
- Programming the OTP with the modified shadow registers values
 Once the user updates the OTP values and the OTP structure is refreshed, the host sends the OTP structure with bit0 in the "Write/read conf" field (word number 7 in the OTP structure) set to 1.



• Reloading the OTP value to the shadow registers

Once the OTP words are successfully programmed, the host uploads the OTP structure to update the OTP shadow registers. This operation allows the host to verify the status of the last SAFMEM programming via bit4 in the "Status" field.

- BSEC control register programming
 Once the user updates the values of the given BSEC control register (Configuration, Debug configuration, Feature configuration and General lock configuration) the host updates the OTP structure and sends it to the device with bit0 in the "Write/read conf" field set to 0.
- OTP programming CLI

The user is given a set of commands to perform a chosen sequence of operations on the OTP partition. Each one of these commands is described below.

5.1.12 **Programming OTP commands**

STM32CubeProgrammer exports several capabilities that can be used to manage the OTP region via various commands based on the OTP structure version, as detailed below

OTP structure 1

Programming SAFMEM

Description: This command allows the user to program SAFMEM memory by modifying the OTP words.

```
Syntax: -otp program [wordID=(value)] [value=(value)]
[sha_rsl=(value)] [sha_wsl=(value)] [sl=(value)] [pl=(value)]
```

This field contains the shadow register number (between 0 and 95). [wordID=(value)] Value must be written in hexadecimal form. Loads value into the chosen OTP shadow register. [value=(value)] Value must be written in hexadecimal form. [sha_rsl=(value)] Loads value into the corresponding shadow read sticky lock bit. Value can be either 0 or 1. [sha_wsl=(value)] Loads value into the corresponding shadow write sticky lock bit. Value can be either 0 or 1. Loads value into the corresponding programming sticky lock bit. [sl=(value)] Value can be either 0 or 1. Loads value into the corresponding programming permanent lock [pl=(value)] bit. Value can be either 0 or 1.

Example

./STM32_Programmer.sh --connect port=usb1 -otp program wordID=0x00
value=0x3f sl=1 wordID=0x08 value=0x18

Display command

Description: This command allows the user to display all or parts of the OTP structure.



Syntax: -otp displ [upper] [lower] [ctrl]

- [upper] Option to display the loaded upper OTP shadow registers values and status.
- [lower] Loads value into the chosen OTP shadow register. Value must be written in hexadecimal form.
- [ctr1] Option to display the loaded BSEC control registers.

Example

./STM32_Programmer.sh --connect port=usb1 --otp displ

Download file command

Description: To fuse a binary file from a start word ID

Syntax: -otp fwrite {lock} [path.bin] wordid=[index]

{lock}	Optional, to indicate the operation type, update, or permanent lock.
[path,bin]	32-bit aligned file, the tool makes padding values if the file is not aligned (a warning message is displayed).

[index] Value in hex/dec format (from 0 to 95 in decimal).

Note: At the execution time the tool displays the operation to be performed, and prompts a message for confirmation. To skip the prompt confirmation, use the (-y or --yes) command.

OTP structure 2

Programming SAFMEM

Description: This command allows the user to program SAFMEM memory by modifying the OTP words. Able to write up to 96 words in the same command line.

```
Syntax: -otp write {lock} {word=[index] value=[val]...}
```

- **{lock}** Optional, to indicate that a lock has been requested. If lock option is mentioned, all words passed in line are locked.
 - With lock: writes the word, then performs permanent lock.
 - Without lock: updates the word.
- [index] The word index can be written in decimal or hex format.
- [val] The value option accepts hex values.

The tool prints the requests, the user can verify the operation before going on. It will then prompt a confirmation message, the user can press yes/y or no/n to, respectively, continue or stop the write operation.

Example

```
STM32_Programmer_CLI.exe --connect port=usb1 -otp write word=52 value=0xAAAAAAA word=0x50 value=0xBBBBBBBB
```





Uploading (DTP data:
OTP Partiti	ion read successfully
OTP Write o You are try	command: ying to write on OTP partition with the following inputs :
Word	Value
052 080 Lock	ØXAAAAAAAA ØXBBBBBBBB NO
Warning: Th Warning: Do yes The operati	nis operation cannot be reverted and may damage your device. b you confirm ? [yes/no] ion was confirmed

Lock OTP command

Description: This command allows to permanent lock the mentioned words, already written. Up to 96 words can be written in the same command line.

```
Syntax: -otp lock {word=[index]...]
```

[index] The word index can be written in decimal or hex format.

The tool prints the requested modifications, and the user can verify the operation before going ahead (use yes/y or no/n to continue or to stop)

Example

STM32_Programmer_CLI.exe --connect port=usb1 -otp lock word=20 word=0x30

Display command

Description: This command allows the user to display all or parts of the OTP structure.

Syntax: -otp disp1 {word=[index]...}

<pre>{word=[index]}</pre>	Optional, able to display up to 96 specific words in the same command, The index value used to indicate the OTP word ID is in decimal or hex format.
-otp displ	Displays all OTP words (version + Global State + OTP words). Highlights the status word containing a state information (prog lock, read lock, read error, invalid).

Example



```
STM32_Programmer_CLI.exe --connect port=usb1 -otp displ
```

```
Figure 155. OTP write command for OTP structure v2
```

CT VERSI	ON :	csion : 0x0000002					
GLOBAL S	TATE : Va _ S _ H _ E	Value : 0x00000000 Value : 0x00000000 State : Secure Open Hardware Key Set : N Encrypted data : N					
REGISTER	5:						
ID	value	status					
99	0x00000017	0x30000000 _[28] Shadow write sticky lock _[29] Shadow read sticky lock					
81	0x00008800	0x10000000 _[28] Shadow write sticky lock					
02	0xD0100000	0x18000000 [[27] Sticky programming lock [[28] Shadow write sticky lock					
03 04	0×00000000 0×00000000	ex00000000 ex00000000 [[27] Sticky programming lock					
05 06 07	0x00000000 0x00000000 0x00000000	0x00000000 0x00000000 0x000000000 0x000000					

Download file command

Description: to fuse a binary file from a start word ID

Syntax: -otp fwrite {lock} [path.bin] word=[index]

{lock}	Optional, to indicate the operation type, update, or permanent lock.
[path,bin]	32-bit aligned file, the tool makes padding values if the file is not aligned (a warning message is displayed).
[index]	Value in hex/dec format (from 0 to 95 in decimal).

Example

Program a PKH binary file starting from word number 24 STM32_Programmer_CLI.exe --connect port=usb1 -otp fwrite lock /user/home/pkh.bin word=24



OTP File write command: You are trying to program a binary file on OTP partition with the following inputs: ------File name | pkh.bin File size | 32 Bytes Start word ID | 24 Lock | YES

5.1.13 Detach command

Description: This command allows the user to send detach command to USB DFU.

Syntax: -detach

5.1.14 GetCertif command

Description: This command can be used to read the chip certificate and save the data to a binary file. The resulting file is required to obtain the associated device product ID, which can then be used to select the appropriate personalization data for the HSM card before using the SSP procedure.

Syntax: -gc <Output_Path>

This command can be used only if a specific firmware (tfa-ssp) is installed, as it is the basis to retreive the stored certificate. Go through the following steps:

For STM32MP15xx

- STM32_Programmer_CLI -c port=usb1 -d tf-a-ssp-trusted.stm32 0x01 -s
- STM32_Programmer_CLI -c port=usb1 -gc "Certificate.bin"

For STM32MP13xx

- STM32_Programmer_CLI -c port=usb1 -d tf-a-ssp-trusted.stm32 0x01 -s
- STM32_Programmer_CLI -c port=usb1 -detach
- STM32_Programmer_CLI -c port=usb1 -d tf-a-ssp-trusted.stm32 0x01 -s
- STM32_Programmer_CLI -c port=usb1 -gc "Certificate.bin"



Figure 156. Get certificate output file

000000£7	00	01	02	03	04	05	06	07	80	09	0a	0Ъ	0c	0d	0e	0f	
00000000	35	30	30	30	32	30	30	41	13	bb	a 9	2Ь	£3	64	86	ab	5000200A. »@+ódt«
00000010	4Ъ	fa	7£	ь4	31	lc	21	fl	6a	78	de	0a	20	31	9£	2d	Kú['l.!ñjxP. lŸ-
00000020	fd	33	66	91	15	c5	18	2e	49	15	02	ce	1b	5Ъ	3c	41	ý3f`.ÅIÎ.[<a< th=""></a<>
00000030	49	ъз	90	ь7	0a	18	7d	5f	bc	ed	44	29	93	d6	48	ь9	I ³ ·}_%iD) "ÖH ¹
00000040	08	cb	77	39	9d	51	55	08	5e	10	56	7d	75	6c	6a	c2	.Ëw9 QU.^.V}uljÂ
00000050	2Ь	0a	c4	2Ь	54	82	8e	ee	60	3f	22	e8	09	7Ъ	bb	ld	+.Ä+T,Žî`?ªè.{».
00000060	e6	fe	1b	ea	3c	2Ъ	ЗЪ	8a	55	da	c8	77	e6	c7	d6	59	æþ.ê<+;ŠUÚÈwæÇÖY
00000070	89	58	fd	82	73	49	bc	7£	0a	63	8a	e 2	3c	fe	ad	9Ъ	%Xý,sI%(.cŠâ <p-></p->
08000000	d5	41	c7	7d	af	52	d 4	42									ŐAÇ} "RÔB
00000090																	

5.1.15 Write blob command

Description: This command allows the user to send the blob (secrets and license).

Syntax: -wb blob.bin

5.2 Secure programming SSP specific commands

Secure secret provisioning (SSP) is a feature supporting secure secret flashing procedure, available on STM32 MPU devices. STM32MP1 series supports protection mechanisms allowing the user to protect critical operations (such as cryptography algorithms) and critical data (such as secret keys) against unexpected accesses.

This section gives an overview of the STM32 SSP command with its associated tools ecosystem and explains how to use it to protect OEM secrets during the CM product manufacturing stage. For more details refer to AN5054.

STM32CubeProgrammer exports a simple SSP command with some options to perform the SSP programming flow.

-ssp, --ssp

Description: Program an SSP file

```
Syntax:-ssp <ssp_file_path> <ssp-fw-path> <hsm=0|1>
<license_path|slot=slotID>
```

<ssp_file_path></ssp_file_path>	SSP	file path to be programmed, bin or ssp extensions.
<ssp-fw-path></ssp-fw-path>	SSP	signed firmware path.
<hsm=0 1></hsm=0 1>	Set u Defa	ser option for HSM use (do not use / use HSM). ult value: hsm = 0.
<license_path slot=slotid></license_path slot=slotid>	•	Path to the license file (if hsm = 0) Reader slot ID if HSM is used (if hsm = 1)

Example using USB DFU bootloader interface:



All SSP traces are shown on the output console.

STM32_Programmer_CLI.exe -c port=usb1 -ssp "out.ssp" "tf-a-sspstm32mp157f-dk2-trusted.stm32" hsm=1 slot=1

Note:

Figure 157. SSP successfully installed

Requesting Chip Certificate... Get Certificate done successfully requesting license for the current STM32 device Init Communication ... ldm_LoadModule(): loading module "stlibp11_SAM.dll" ... ldm_LoadModule(WIN32): OK loading library "stlibp11_SAM.dll": 0x62000000 ... C_GetFunctionList() returned 0x00000000, g_pFunctionList=0x62062FD8 P11 lib initialization Success! Opening session with solt ID 1... Succeed to Open session with reader solt ID 1 Succeed to generate license for the current STM32 device Closing session with reader slot ID 1... Session closed with reader slot ID 1 Closing communication with HSM... Communication closed with HSM Succeed to get License for Firmware from HSM slot ID 1 Starting Firmware Install operation... Writing blob Blob successfully written Start operation achieved successfully Send detach command Detach command executed SSP file out.ssp Install Operation Success

If there is any faulty input the SSP process is aborted, and an error message is displayed to indicate the root cause of the issue.



6 STM32CubeProgrammer C++ API

In addition to the graphical user interface and to the command line interface STM32CubeProgrammer offers a C++ API that can be used to develop your application and benefit of the wide range of features to program the memories embedded in STM32 microcontrollers, either over the debug interface or the bootloader interface (USB DFU, UART, I²C, SPI and CAN).

For more information about the C++ API, read the help file provided within the STM32CubeProgrammer package under API\doc folder.



7 Revision history

Date	Revision	Changes				
15-Dec-2017	1	Initial release.				
02-Aug-2018	2	Updated: - Section 1.1: System requirements - Section 1.2.3: macOS install - Section 1.2.4: DFU driver Added: - Section 3.2.8: Debug commands - Figure 1: macOS "Allow applications downloaded from:" tab - Figure 2: Deleting the old driver software				
12-Sep-2018	3	Added SPI, CAN and I2C settings on cover page and in <i>Section 2.1.4:</i> <i>Target configuration panel</i> . Updated: - <i>Figure 11: ST-LINK configuration panel</i> - <i>Figure 109: STM32CubeProgrammer: available commands</i> . - <i>Figure 114: Connect operation using SWD debug port</i> Replaced Section 3.2.1: Connect command.				
16-Nov-2018	4	Updated Section 2.1.4: Target configuration panel, Section 2.2.1: Reading and displaying target memory, Section 2.2.2: Reading and displaying a file and Section 2.3.2: External flash memory programming. Updated Figure 9: STM32CubeProgrammer main window, Figure 10: Expanded main menu, Figure 11: ST-LINK configuration panel, Figure 12: UART configuration panel, Figure 13: USB configuration panel, Figure 14: Target information panel, Figure 15: SPI configuration panel, Figure 16: CAN configuration panel, Figure 17: I2C configuration panel, Figure 18: Device memory tab, Figure 20: File display, Figure 21: Flash memory programming and erasing (internal memory) and Figure 22: Flash memory programming (external memory). Minor text edits across the whole document.				
03-Jan-2019	5	Updated Section 1.2.4: DFU driver. Added Section 3.2.19: Secure programming SFI specific commands, Section 3.2.21: HSM related commands and Section 6: STM32CubeProgrammer C++ API. Minor text edits across the whole document.				
04-Mar-2019	6	Updated Introduction and Section 1: Getting started. Updated title of Section 2: STM32CubeProgrammer user interface for MCUs and of Section 3: STM32CubeProgrammer command line interface (CLI) for MCUs. Added Section 2.5: Automatic mode, Section 2.6: STM32WB OTA programming, Section 4: STM32CubeProgrammer user interface for MPUs, Section 5: STM32CubeProgrammer CLI for MPUs and their subsections.				

Table 2. Document revision history	Table 2.	Document	revision	history
------------------------------------	----------	----------	----------	---------



Date	Revision	Changes
19-Apr-2019	7	Updated Section 1.1: System requirements, Section 2.2.2: Reading and displaying a file, Section 2.6.2: OTA update procedure, Section 3.2.19: Secure programming SFI specific commands, Section 3.2.21: HSM related commands and Section 3.2.22: STM32WB specific commands. Updated Figure 21: Flash memory programming and erasing (internal memory).
11-Oct-2019	8	Updated Graphical guide, Section 3.2.19: Secure programming SFI specific commands, Section 3.2.21: HSM related commands and Section 3.2.22: STM32WB specific commands. Added Section 2.6: In application programming (IAP/USBx). Minor text edits across the whole document.
08-Nov-2019	9	Updated Section 1.2.1: Linux install, Section 3.2.22: STM32WB specific commands and Section 5.1.6: Read partition command. Minor text edits across the whole document.
07-Jan-2020	10	Updated Section 1.1: System requirements, Section 1.2.3: macOS install and Section 3.2.19: Secure programming SFI specific commands. Added Section 3.2.16: TZ regression command and Section 3.2.20: Secure programming SFIx specific commands. Removed former Section 5.2.12: Writing to BSEC command. Minor text edits across the whole document.
24-Feb-2020	11	Added Section 2.7: Flash the co-processor binary using graphical interface and its subsections.
23-Jul-2020	12	Added Section 2.8: Serial wire viewer (SWV), Section 3.2.23: Serial wire viewer (SWV) command and Section 5.2: Secure programming SSP specific commands. Updated Section 3.2.1: Connect command and Section 3.2.2: Erase command. Minor text edits across the whole document.
17-Nov-2020	13	Updated Section 1.1: System requirements, Section 1.2.1: Linux install, Section 1.2.2: Windows install, Section 1.2.3: macOS install, Section 2.3.2: External flash memory programming, Section 2.8: Serial wire viewer (SWV), Section 3.2.1: Connect command, Section 3.2.2: Erase command, Section 3.2.13: External loader command, Section 3.2.21: HSM related commands, Section 3.2.20: Secure programming SFIx specific commands, Section 3.2.20: Secure programming SFIx specific commands, Section 3.2.22: STM32WB specific commands and Section 5.1.1: Connect command. Added Section 2.11: DFU IAP/USBx with custom PID and VID, Section 2.12: SigFox™ credentials, Example using DFU IAP/USBx options, Section 3.2.5: Download 64-bit data command, Section 3.2.24: Specific commands for STM32WL and Section 5.2.5: Flashing service via USB serial gadget. Updated Figure 22: Flash memory programming (external memory), Figure 42: SWV window and Figure 66: Available commands for MPUs.
19-Nov-2020	14	Updated Section 5.1.1: Connect command. Removed former Section 5.1: Command line usage and Section 5.2.5: Flashing service via USB serial gadget.

Table 2. Document revision history (continued)



 Bection 1.1: System requirements, Section 1.2.1: Linux install, 2.3: macOS install, Section 2.12: SigFox™ credentials and 2.22: STM32WB specific commands. Ction 2.13: Register Viewer, Section 2.14: Hard Fault analyzer Desections, Section 3.2.26: Register viewer and Section 3.2.27: analyzer. edits across the whole document. Section 2.1.4: Target configuration panel, Section 3.2.1: command, Section 3.2.2: Erase command and Section 3.2.22: B specific commands. Ction 2.15: Fill memory command, Section 2.16: Fill memory Section 2.17: Blank check command, Section 2.18: Blank ration, Section 2.19: Compare flash memory with file,
edits across the whole document. Section 2.1.4: Target configuration panel, Section 3.2.27: amand, Section 3.2.2: Erase command and Section 3.2.2: B specific commands. Section 2.15: Fill memory command, Section 2.16: Fill memory Section 2.17: Blank check command, Section 2.18: Blank ration, Section 2.19: Compare film memory with file,
Section 2.1.4: Target configuration panel, Section 3.2.1: command, Section 3.2.2: Erase command and Section 3.2.22: 3 specific commands. ction 2.15: Fill memory command, Section 2.16: Fill memory Section 2.17: Blank check command, Section 2.18: Blank ration, Section 2.19: Compare flash memory with file,
Section 2.1.4: Target configuration panel, Section 3.2.1: command, Section 3.2.2: Erase command and Section 3.2.22: 3 specific commands. Section 2.15: Fill memory command, Section 2.16: Fill memory Section 2.17: Blank check command, Section 2.18: Blank ration, Section 2.19: Compare flash memory with file,
ction 2.15: Fill memory command, Section 2.16: Fill memory Section 2.17: Blank check command, Section 2.18: Blank ration, Section 2.19: Compare flash memory with file,
20: Comparison between two files, Section 2.21: LiveUpdate discrimination of the discrim
igure 12: UART configuration panel and Figure 111: Enabling
ure 112: Connect operation using USB. edits across the whole document.
ction 2.10: STM32CubeProgrammer Script Manager platform and its subsections. Section 2.1.1: Main menu, Section 2.1.4: Target configuration ction 2.6: In application programming (IAP/USBx), Section 2.7: co-processor binary using graphical interface and its as, Section 2.10: STM32CubeProgrammer Script Manager or MCUs, Section 3.2.1: Connect command and Section 3.2.22: B specific commands. former Section 2.6: STM32WB OTA programming. Figure 9: STM32CubeProgrammer main window, Figure 10: main menu, Figure 11: ST-LINK configuration panel and Target information panel.
edits across the whole document.
ction 1.3: Updater with its subsections, Section 2.4.1: MCU ecific for the STM32WL series), and Section 3.2.29: GetCertif Section 1.1: System requirements, Section 2.1.4: Target ion panel, Section 2.7.2: Key provisioning, Section 3.2.1: ommand, Section 3.2.9: List command, Section 3.2.22: 3 specific commands, and Section 3.2.24: Specific commands WL.

Table 2. Document revision history (continued)



Date	Revision	Changes
29-Jun-2022	19	Added Section 2.9: Secure programming interface, Section 4.3: OTP programming window, and their subsections. Updated Section 2.1.1: Main menu, I2C settings, Section 3.2.1: Connect command, Section 3.2.24: Specific commands for STM32WL, Section 3.2.26: Register viewer, Section 5.1.12: Programming OTP commands, and Section 5.1.14: GetCertif command. Removed former Section 5.1.16: Display command. Updated figures 9 to 17 and 34 to 41. Minor text edits across the whole document.
28-Nov-2022	20	Updated Section 1.2.3: macOS install, Section 2.2.1: Reading and displaying target memory, Section 2.7.1: FUS/Stack upgrade, Section 3.2.4: Download 32-bit data command, Section 3.2.10: QuietMode command, Section 3.2.22: STM32WB specific commands, Section 3.2.23: Serial wire viewer (SWV) command, and Section 3.2.28: RDP regression with password. Updated figures 17 to 19, 35 to 37, 39 to 41, Figure 145: Program Apply confirmation, and Figure 147: All OTP fields are locked. Added Section 2.9.4: SSP, Section 2.10.3: Loops and conditional statements, and their subsections. Minor text edits across the whole document.
24-Feb-2023	21	Updated Section 1.2.5: ST-LINK driver and Section 3.2.23: Serial wire viewer (SWV) command. Added Section 2.4.2: Debug authentication default configuration (STM32H573/STM32H563/STM32H562 only), Section 2.4.3: Debug authentication configuration (STM32H503 only), Section 2.9.5: OBKey provisioning (STM32H573/STM32H563/STM32H562 only), Section 2.9.6: Debug authentication (STM32H5 series only), Section 2.22: Calculator, and sections 3.2.30 to 3.2.34. Updated figures 43 to 47 in Section 2.9.2: RDP regression with password (STM32U5 series only). Minor text edits across the whole document.

Table 2. Document revision history (continued)



IMPORTANT NOTICE – READ CAREFULLY

STMicroelectronics NV and its subsidiaries ("ST") reserve the right to make changes, corrections, enhancements, modifications, and improvements to ST products and/or to this document at any time without notice. Purchasers should obtain the latest relevant information on ST products before placing orders. ST products are sold pursuant to ST's terms and conditions of sale in place at the time of order acknowledgment.

Purchasers are solely responsible for the choice, selection, and use of ST products and ST assumes no liability for application assistance or the design of purchasers' products.

No license, express or implied, to any intellectual property right is granted by ST herein.

Resale of ST products with provisions different from the information set forth herein shall void any warranty granted by ST for such product.

ST and the ST logo are trademarks of ST. For additional information about ST trademarks, refer to www.st.com/trademarks. All other product or service names are the property of their respective owners.

Information in this document supersedes and replaces information previously supplied in any prior versions of this document.

© 2023 STMicroelectronics – All rights reserved

UM2237 Rev 21

