S32 Flash Tool 2.0.1

User Guide

Document Number: S32FTUG
Rev. 1.0, 04/2021

Contents

About S32 Flash Tool

S32 Flash Tool is a solution designed for programming binary files (software images) to an external flash device. The goal of this software is to serve the installation boot of S32 family of devices.

S32 Flash Tool is distributed with the S32 Design Studio installation package. It can also be used with third-party software applications.

There are two ways how you can use the tool:

- Using command line interface
- Using graphical user interface

Architecture

S32 Flash Tool includes two parts – the Host Application running on the computer and the Target Application running on the connected device. The Target Application communicates with the Host Application over the serial interface.

When S32 Flash Tool is started, the Host Application communicates with the device's Boot ROM and downloads the Target Application image to SRAM. When done, the Boot ROM runs the Target Application on the selected CPU core of the device. The Target Application executes the flash device specific Flash Algorithm - a bare-board application that implements general I/O functions for operating with flash memory, and chip-specific functions.

When the Target Application is up and ready, it begins to listen to the Host Application for commands. When a command is received, it is converted to a flash programming instruction. Commands for writing an image are resulted in programming the transferred image to external flash.

The Host Application is terminated when the user closes the command prompt window. The Target Application is terminated when the target system is powered off.

Supported program images

The following image files can be written to external flash memory:

- Image Vector Table (IVT)
- Device Configuration Data (DCD)
- Self-Test Device Configuration Data (ST-DCD)
- Application Boot code

Supported target devices

S32 Flash Tool base installation package supports the following devices:

- S32V234
- S32S247TV

Note: Additional software packages are required to add support for any other specific NXP Arm® based processor families. For details refer to the "Installing S32 Flash Tool" section of the *S32 Flash Tool 2.0.1 Release Notes*.

Supported flash devices

External flash memory devices are memory chips used for data storage that can be electrically erased and reprogrammed. S32 Flash Tool supports SD, eMMC via uSDHC interfaces and external flash devices supported on the target via QSPI peripheral.

The following target applications with the memory device specific algorithms are available in the `<S32FlashTool_install_dir>/flash` folder:

- S32V234: `EMMC.bin`, `SD.bin`, `MX25UM51245G.bin`, `MX25UW51245G.bin`, `S26KL512S2.bin`
- S32S247TV: `MX25UW12A45G_R52.bin`

Using command line interface

Quick Start Guide

To run S32 Flash Tool:

1. Copy the S32 Flash Tool files to your computer. If S32 Flash Tool is installed on your computer, the files are available in S32 Flash Tool installation folder.
2. Connect the target device to your computer and set the device boot mode to Serial.
3. Open the command prompt and switch to the `<S32FlashTool_install_dir>/bin` folder.
4. Run the `S32FlashTool` executable file with the appropriate parameters.

   Find the details in Loading S32 Flash Tool.
5. Use commands to manage external flash memory.

   To learn more about commands and parameters refer to topic Commands.

For user examples refer to topic Using commands.

Commands

Loading S32 Flash Tool

To specify the target device, memory type and frequency (optional) run `S32FlashTool` executable file with the following parameters:

```
S32FlashTool -t [target_file] [-xosc [freq]] -a [filename] -i [uart | can |
 eth] -p [devicename]
```

where:

Option	Description
`-t` [*target_file*]	Specifies the path of the Target Application to be loaded to the device.
`-xosc` [*freq*]	Optional. Specifies frequency of the external generator/quartz. The frequency can be defined with K or M suffix for KHz or MHz, for example, `-xosc 20M`.
`-s`	Optional. Specifies the secure boot mode for selected target Secured Boot image container. **Note:** This prefix is mandatory for the secured chips.[1]
`-a` [*filename*]	Specifies the path of the flash memory specific application (Flash Algorithm) to be run by the Target Application on the device.
`-i` [*uart* | *can* | *eth*]	Specifies the communication interface. Options: • `uart`: Universal Asynchronous Receiver-Transmitter (UART). Use the UART0 module on the target device. • `can`: Controller Area Network (CAN). Use the CAN0 interface on the target device. **Note:** You need to install drivers for used converter before running.

[1] To receive a Secured Boot image container for your device contact your NXP representative (e.g., FAE, customer support) for the Trust Center request.

Option	Description
	• `eth`: The Ethernet connection. For details refer to topic Loading S32 Flash Tool via Ethernet. **Note:** Please consult supported_devices.txt about feature availability for your device.

`-p` [*devicename*]	Specifies the device name.
	• For the UART interface:
	  • In Windows: Specifies the COM port, for example `COM35`.
	  • In Linux: Specifies the device name, for example `/dev/ttyUSB0`. Make sure to have permission to access the device.
	• For the CAN interface include [*devicename, port, serialnumber*], where:
	  • *devicename* specifies the CAN device name. Options:
	  • `ixxat`,
	  • `kvaser`,
	  • `vector` (Windows only).
	  • *port* specifies the port number (optional - if omitted, system uses the default value),
	  • *serialnumber* specifies the CAN device serial number (optional).
	• For the Ethernet interface, specifies the target IP address (format is xxx.xxx.xxx.xxx).

Loading S32 Flash Tool via Ethernet

To support ethernet booting, the target device and PC should be included in one Ethernet network where DHCP and TFTP servers are available.

Note: The option is supported only for some devices. Find option support info in the target device specific supported_devices.txt.

To run S32 Flash Tool for work via Ethernet:

1. Set the device boot mode to Ethernet.

   Note: For appropriate boot mode details refer to the target device Reference Manual.
2. Open the command prompt and switch to the `<S32FlashTool_install_dir>/bin` folder.
3. Create ethernet boot image - run `sbl_gen` executable file with the following parameters:

   ```
   sbl_gen <S32FlashTool_install_dir>/targets/<target_device>Eth.bin
    <path_to_save_images>
   ```

   Note: For the secure boot mode add the -s key.
4. Copy *.img files created on previous step to the TFTP server folder.
5. Power on your device and wait about 10 seconds to finish Bootrom loading process.
6. Load algorithm and work with Flash tool as usual.

Device ID

The `-fid` command:

```
S32FlashTool -t [target_file] -fid -i [uart | can | eth] -p [devicename]
```

where:

Option	Description
`-t` [*target_file*]	Specifies the path of the Target Application to be loaded to the device.
`-i` [*uart* | *can* | *eth*]	Specifies the communication interface. For details refer to topic Loading S32 Flash Tool.
`-p` [*devicename*]	Specifies the device name. For details refer to topic Loading S32 Flash Tool.

Write data to memory

To write a file to memory, use the `-fwrite` command:

```
S32FlashTool -t [target_file] -fwrite -f [filename] -addr [value] [-size
 [value]] -i [uart | can | eth] -p [devicename]
```

where:

Option	Description
`-t` [*target_file*]	Specifies the path of the Target Application to be loaded to the device.
`-f` [*filename*]	Specifies the file that will be written to memory. The file path can be absolute or relative.
`-addr` [*value*]	Specifies the start address in the device's address range. The address can be a decimal or hexadecimal value.
`-size` [*value*]	(Optional) Specifies the size of memory (in bytes) to be written. The size can be a decimal or hexadecimal value. **Note:** In the following cases function takes size of the file as value instead of parameter setting: • value is 0, • value is greater than file size, • option is omitted.
`-i` [*uart* | *can* | *eth*]	Specifies the communication interface. For details refer to topic Loading S32 Flash Tool.
`-p` [*devicename*]	Specifies the device name. For details refer to topic Loading S32 Flash Tool.

Read memory

To read memory, use the `-fread` command:

```
S32FlashTool -t [target_file] -fread -addr [value] -size [value] -i [uart |
 can | eth] -p [devicename] [-b] [-f [filename]]
```

where:

Option	Description
`-t` [*target_file*]	Specifies the path of the Target Application to be loaded to the device.
`-addr` [*value*]	Specifies the start address in the device's address range. The address can be a decimal or hexadecimal value.
`-size` [*value*]	Specifies the size of memory (in bytes) to be read. The size can be a decimal or hexadecimal value.
`-i` [*uart* | *can* | *eth*]	Specifies the communication interface. For details refer to topic Loading S32 Flash Tool.
`-p` [*devicename*]	Specifies the device name. For details refer to topic Loading S32 Flash Tool.

Option	Description
`-b`	(Optional) Outputs data in binary format.
`-f` [*filename*]	(Optional) Outputs data to the specified file. The file path can be absolute or relative.

Erase memory

To erase memory, use the `-ferase` command:

```
S32FlashTool -t [target_file] -ferase -addr [value] [-size [value]] [-f
 [filename]] -i [uart | can | eth] -p [devicename]
```

where:

Option	Description
`-t` [*target_file*]	Specifies the path of the Target Application to be loaded to the device.
`-addr` [*value*]	Specifies the start address in the device's address range. The address can be a decimal or hexadecimal value.
`-size` [*value*]	(Optional) Specifies the size of memory (in bytes) to be erased. The size can be a decimal or hexadecimal value. **Note:** • Size is required for SD and MMC cards only - if option is omitted size value is set to 0. The tool will erase nothing. • For NOR flashes, if size value is 0 (or option is omitted) it specifies the whole sector to be erased. The tool will erase the whole memory block, including the specified start address.
`-f` [*filename*]	(Optional) Specifies the file to take as the size of memory (in bytes) to be erased. The file path can be absolute or relative. **Note:** In the following cases function takes size of the file as value instead of parameter setting: • size option is omitted, • size value is 0, • size value is greater than file size. If size value is less than file size function takes size value.
`-i` [*uart* | *can* | *eth*]	Specifies the communication interface. For details refer to topic Loading S32 Flash Tool.
`-p` [*devicename*]	Specifies the device name. For details refer to topic Loading S32 Flash Tool.

Verify memory

To verify memory with specified file, use the `-fverify` command:

```
S32FlashTool -t [target_file] -fverify -f [filename] -addr [value] [-size
 [value]] -i [uart | can | eth] -p [devicename]
```

where:

Option	Description
`-t` [*target_file*]	Specifies the path of the Target Application to be loaded to the device.

Option	Description
`-f` [*filename*]	Specifies the file that will be verified with flash memory. The file path can be absolute or relative.
`-addr` [*value*]	Specifies the start address in the device's address range. The address can be a decimal or hexadecimal value.
`-size` [*value*]	(Optional) Specifies the size of memory (in bytes) to be verified. The size can be a decimal or hexadecimal value. **Note:** If size is 0 (or option is omitted) it takes size of the file as value.
`-i` [*uart* | *can* | *eth*]	Specifies the communication interface. For details refer to topic Loading S32 Flash Tool.
`-p` [*devicename*]	Specifies the device name. For details refer to topic Loading S32 Flash Tool.

Program memory

To rewrite (erase/write/verify) a new file to memory, use the `-fprogram` command:

```
S32FlashTool -t [target_file] -fprogram -f [filename] -addr [value] [-size
 [value]] [-noverify] -i [uart | can | eth] -p [devicename]
```

where:

Option	Description
`-t` [*target_file*]	Specifies the path of the Target Application to be loaded to the device.
`-f` [*filename*]	Specifies the file that will be written to memory. The file path can be absolute or relative.
`-addr` [*value*]	Specifies the start address in the device's address range. The address can be a decimal or hexadecimal value.
`-size` [*value*]	(Optional) Specifies the size of memory (in bytes) to be erased, written and verified. The size can be a decimal or hexadecimal value. **Note:** In the following cases function takes size of the file as value instead of parameter setting: • size option is omitted, • size value is 0, • size value is greater than file size.
`-noverify`	(Optional) Excludes the verification step.
`-i` [*uart* | *can* | *eth*]	Specifies the communication interface. For details refer to topic Loading S32 Flash Tool.
`-p` [*devicename*]	Specifies the device name. For details refer to topic Loading S32 Flash Tool.

Using commands

Examples of typical tasks to perform.

Working with SD/EMMC card

To run S32 Flash Tool for work with SD/EMMC card at S32V234 device, named COM11, through UART:

1. Download algorithm for SD card:

```
S32FlashTool -t ../targets/S32V234.bin -a ../flash/SD.bin -i uart -p COM11
```

Note: For EMMC replace SD.bin with EMMC.bin algorithm.

2. Get the ID of the device (optional):

```
S32FlashTool -t ../targets/S32V234.bin -fid  -i uart -p COM11
```

3. Program the desired file:
   - Step-by-step programming:
     a. Erase 64 MiB flash in card:

```
S32FlashTool -t ../targets/S32V234.bin -ferase -addr 0x0 -size
 67108864 -i uart -p COM11
```

     or erase the amount of bites for the certain file to be writen later:

```
S32FlashTool -t ../targets/S32V234.bin -ferase -addr 0x0 -f
 C:\Users\...\workspaceS32DS.3.3\hello_world_s32v234\Debug_RAM
\hello_world_blob -i uart -p COM11
```

     b. Read first 16 bytes of memory to verify erased:

```
S32FlashTool -t ../targets/S32V234.bin -fread -addr 0x0 -size 16 -i
 uart -p COM11
```

     c. Flash the desired program:

```
S32FlashTool -t ../targets/S32V234.bin -fwrite -f C:\Users\...
\workspaceS32DS.3.3\hello_world_s32v234\Debug_RAM\hello_world_blob
 -addr 0x0 -i uart -p COM11
```

     d. Verify flashed program (optional):

```
S32FlashTool -t ../targets/S32V234.bin -fverify -f C:\Users\...
\workspaceS32DS.3.3\hello_world_s32v234\Debug_RAM\hello_world_blob
 -addr 0x0 -i uart -p COM11
```

   - One-step programming:
     - with verification:

```
S32FlashTool -t ../targets/S32V234.bin -fprogram -f C:
\Users\...\workspaceS32DS.3.3\hello_world_s32v234\Debug_RAM
\hello_world_blob -addr 0x0 -i uart -p COM11
```

     - without verification:

```
S32FlashTool -t ../targets/S32V234.bin -fprogram -f C:
\Users\...\workspaceS32DS.3.3\hello_world_s32v234\Debug_RAM
\hello_world_blob -addr 0x0 -noverify -i uart -p COM11
```

4. Read first 16 bytes of memory to verify initial first byte of IVT = 0x51 (optional):

```
S32FlashTool -t ../targets/S32V234.bin -fread -addr 0x0 -size 16 -i uart -
p COM11
```

Working with flash

To run S32 Flash Tool for work with flash memory at S32S247TV device, named COM35, through UART:

1. Download algorithm for flash:

```
S32FlashTool -t ../targets/S32S247TV.bin -a ../flash/MX25UW12A45G_R52.bin
 -i uart -p COM35
```

2. Get the ID of the device (optional):

```
S32FlashTool -t ../targets/S32S247TV.bin -fid  -i uart -p COM35
```

3. Program the desired file:
   - Step-by-step programming:
     a. Erase existing data from the first sector:

```
S32FlashTool -t ../targets/S32S247TV.bin -ferase -addr 0x0 -i uart
 -p COM35
```

     b. Read first 16 bytes of memory to verify erased:

```
S32FlashTool -t ../targets/S32S247TV.bin -fread -addr 0x0 -size 16
 -i uart -p COM35
```

     c. Flash desired program:

```
S32FlashTool -t ../targets/S32S247TV.bin -fwrite -f Test.txt -addr
 0x0 -i uart -p COM35
```

     d. Verify flashed program (optional):

```
S32FlashTool -t ../targets/S32S247TV.bin -fverify -f Test.txt -addr
 0x0 -i uart -p COM35
```

   - One-step programming:
     - with verification:

```
S32FlashTool -t ../targets/S32S247TV.bin -fprogram -f Test.txt -
addr 0x0 -i uart -p COM35
```

     - without verification:

```
S32FlashTool -t ../targets/S32S247TV.bin -fprogram -f Test.txt -
addr 0x0 -noverify -i uart -p COM35
```

4. Read first 1000 bytes of memory and write it to the specified file:

```
S32FlashTool -t ../targets/S32S247TV.bin -fread -addr 0x0 -size 1000 -i
 uart -p COM35 -f out.bin
```

Using Graphical user interface

You can use the GUI wrapper of the command-line tool:

1. Connect the target device to your computer and set the device boot mode to Serial.
2. Browse to the `<S32FlashTool_install_dir>/GUI` directory.
3. Run the `s32ft` file.
4. Start with the **Initialization** section:
   - **Target**: select the Target Application to be loaded to the device.

     Note: For Ethernet connection select the `*Eth` target.
   - **Algorithm**: select the flash memory specific application (Flash Algorithm) to be run by the Target Application on the device.
   - **Override XOSC frequensy**: select the checkbox to specify frequency of the external generator/quartz. The frequency can be defined with K or M suffix for KHz or MHz.
   - **Secure boot**: select the checkbox to work with secured chips. Press the **Browse** button to select the required secured boot image container.[2]

     Note: For secured chips only.
5. In the **Communication** section, specify the device name:
   - For the UART interface:
     - On Windows: specify the COM port, for example `COM35`.
     - On Linux: specify the device name, for example `/dev/ttyUSB0`. Make sure to have permission to access the device.

     Note: Use the UART0 module on the target device.
   - For the CAN interface:
     - Select the CAN device name (mandatory),
     - Specify the port number (optional - if omitted, system uses the default value),

       Note: Field accepts numbers only.
     - Specify the CAN device serial number (optional).

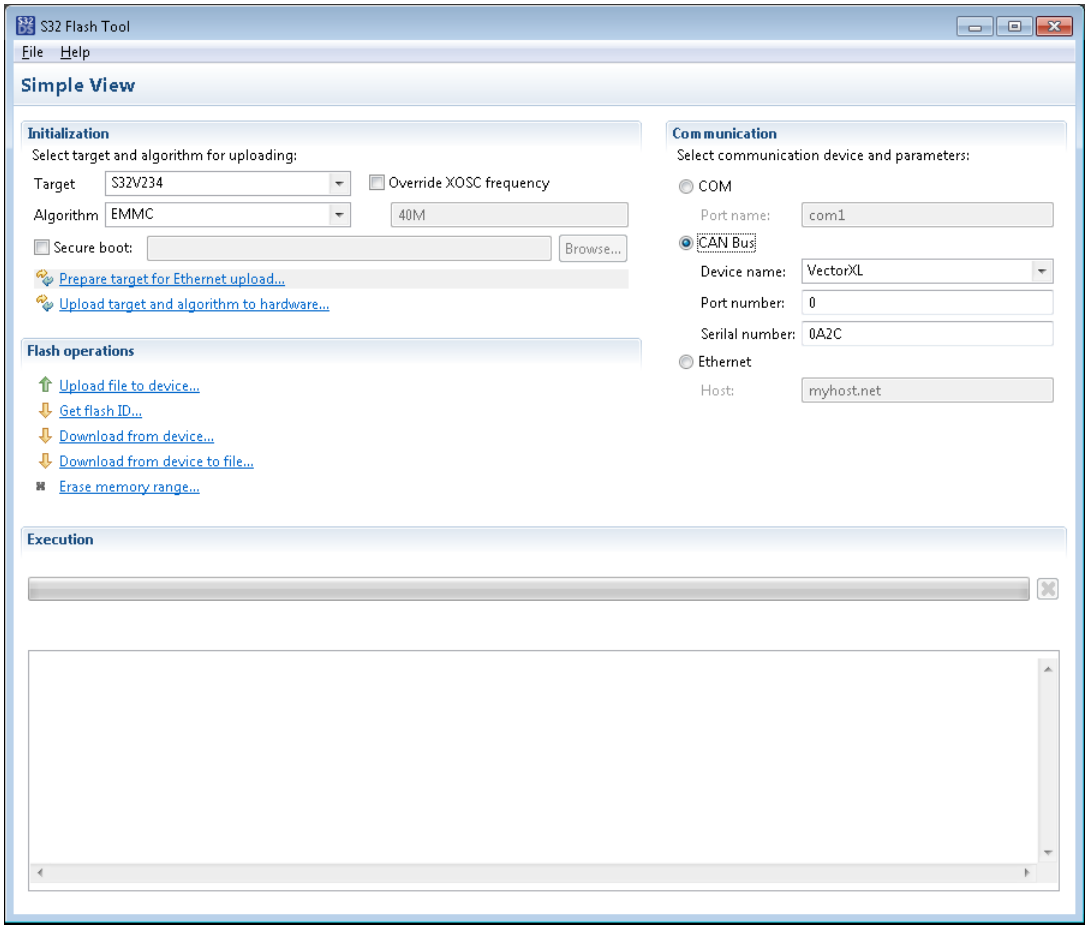
     Note: Use the CAN0 interface on the target device.

     Note: You need to install drivers for used converter before running.
   - For the Ethernet interface, specify the host name or IP address.

     Note: Please consult supported_devices.txt about feature availability for your device.
6. If the Ethernet interface is selected (if not, go to the next step):
   a) Go back to the **Initialization** section and click **Prepare target for Ethernet upload**.
   b) In the **Prepare target for Ethernet upload** dialog box, browse to the TFTP server folder and click **OK**.

   Note: For more details refer to topic Loading S32 Flash Tool via Ethernet.
7. Go back to the **Initialization** section and click **Upload target and algorithm to hardware**.

[2] To receive a Secured Boot image container for your device contact your NXP representative (e.g., FAE, customer support) for the Trust Center request.



8. Use the following options and commands to manage memory:

Table 1: The Flash operations section

Option	Description
Upload file to device	Use this command to write a file to memory: • specify the start address in the device's address range (hex), • check the **Verify** checkbox to enable verification step after writing, • browse to the file that will be written to memory, • click **OK**
Get flash ID	Click to get the ID of the connected device.
Download from device	Use this command to read memory: • specify the start address in the device's address range (hex), • specify the size of memory (in bytes) to be read (hex), • check the **Binary output** checkbox to enable data output in binary format, • click **OK**
Download from device to file	Use this command to read memory and write data to a file: • specify the start address in the device's address range (hex), • specify the size of memory (in bytes) to be read (hex), • check the **Binary output** checkbox to enable data output in binary format, **Note:** In UI outputdata "0" symbol is replaced by "."

Option	Description
	• browse to the file to write data, • click **OK**
Erase memory range	Use this command to erase memory: • specify the start address in the device's address range (hex), • specify the size of memory (in bytes) to be erased (hex), • click **OK**

Troubleshooting

This section contains a table that describes possible solutions to problems that may occur when using S32 Flash Tool. The table contains:

- Symptoms that describe the sign or warning message for the type of problem.
- Possible solutions that describe what you should do to try to solve the problem.

If your problem is not described below, check the list of known issues and workarounds in Release Notes, then refer to the S32DS Public NXP Community space or submit a technical support request.

Table 2:

Symptom	Possible solution
Target image load fails	Check the connection: for details refer to your board documentation (e.g. UART connection on S32V234-EVB2, see Get Started with the S32V234-EVB2 paragraph 1.4)
	Make sure that communication device is enabled and accessible, i.e. related jumpers and switches are set to proper positions (e.g. on S32G-PROCEVB J124 1&3 and 2&4)
	Make sure that boot mode is set to Serial (e.g. on S32V234-EVB2 set J40 = 1&2, J42 = 2&3)
	Make sure that BootROM sends responses for incoming bytes (e.g. sending bytes from any terminal program)
	Make sure that appropriate security level is used for boot (for secured SoC – signed secured image loaded with option “-s”)
	If the problem occurs with a Secured application contact your NXP representative (e.g., FAE, customer support) for the Trust Center request
Algorithm initialization fails	Make sure that memory device is connected and related jumpers are set (e.g. on S32V234-EVB2 set the following: • to enable SD: J48 = 1&2, J49 = 1&2 and insert SD card, • to enable EMMC: J48 = 1&2, J49 = 2&3, • to enable QSPI: J48 = 2&3)
Tool can't open the device	Make sure that proper device is selected and it is not used by another application
Unable to connect to server	Check the IP address assigned to the board and make sure that the correct one is used in the tool
	Make sure that DHCP and TFTP servers are available
	Make sure that ethernet boot images are located on the TFTP server

How to Reach Us:

Home Page:

nxp.com

Web Support:

nxp.com/support

For more information, please visit: http://www.nxp.com

For sales office addresses, please send an email to: salesaddresses@nxp.com

Revision: 1.0, April 2021