Software API compatible with the bridge interface of STLINK-V3 and STLINK-V3PWR

Introduction

This release note is updated periodically to keep abreast of the STLINK-V3 bridge API (STLINK-V3-BRIDGE) evolution, problems, and limitations. Check the product webpage on STMicroelectronics website at *www.st.com* for the latest version. For the latest release summary, refer to Table 1.

Table 1. STLINK-V3 bridge API release summary

Type	Summary
Minor release	STLINK-V3 bridge API v1.3.0 release: • Added support for I^2C repeated start feature • Improved path handling with Unicode® or multibyte character support • Enhanced code quality

Customer support

For more information or help concerning STLINK-V3 bridge API, contact the nearest STMicroelectronics sales office or use the ST community at community.st.com. For a complete list of STMicroelectronics offices and distributors, refer to the *www.st.com* webpage.

Software updates

For software updates and latest documentation, refer to the STLINK-V3-BRIDGE product webpage.

1 General information

1.1 Overview

The bridge API (STLINK-V3-BRIDGE) is a set of source files that allow the development of personal computer applications exercising the STLINK-V3 and STLINK-V3PWR bridge interface of a target board. Refer to the board user manual to check whether it features the STLINK-V3 bridge interface.

The bridge API initializes the microcontroller of the STLINK-V3 subsystem and controls the communication through its interfaces: I^2C, SPI, CAN, and optionally CAN FD. It also allows the configuration of up to four additional signals (GPIOs). Besides, the communication through STLINK-V3 UARTs is controlled by means of the Virtual COM port dedicated USB interfaces.

The development of the embedded application in the target board is facilitated by the use of the STM32Cube MCU Packages. The user must ensure that the parameters, used by the target application to configure the I^2C, SPI, CAN, and optionally CAN FD communication interfaces on the target side, match the STLINK-V3 or STLINK-V3PWR configuration done through the bridge API.

The STLINK-V3-BRIDGE is built for use with the STLINK-V3 and STLINK-V3PWR bridge interface firmware, which runs on an STM32 microcontroller based on the Arm® Cortex®-M processor.

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

1.2 Host PC system requirements

Supported operating systems and architectures

- Windows® 10 and 11, Linux®, and macOS® 64-bit operating systems (x64)

Note: Windows is a trademark of the Microsoft group of companies.

Linux® is a registered trademark of Linus Torvalds.

macOS® is a trademark of Apple Inc., registered in the U.S. and other countries and regions.

Software requirements

STLINK-V3 bridge API relies on the following libraries:

- Windows®: `STLinkUSBDriver.dll`
- Linux®: `STLinkUSBDriver.so`
- macOS®: `STLinkUSBDriver.dylib`

Those libraries are usually provided with the tools supporting ST-LINK. They are also provided in the STSW-LINK007 ST-LINK firmware upgrade package available at *www.st.com*.

1.3 Setup procedure

1. Unzip the software package.
2. Import the source files provided in the `src/sharedStlk` subdirectory into the C++ project.
3. Read the FAQ *How to use Bridge API in a project* in the `STLink_Bridge_API.chm` document and study the files provided in `src/example/bridge/` and `src/project/apiBridgeProject/` as guidelines for integration into an application.
4. On Linux® and macOS®, link the application respectively with `STLinkUSBDriver.so` and `STLinkUSBDriver.dylib`. Note that `libusb` is required at runtime and must be installed separately.
5. On Windows®, make sure to locate `STLinkUSBDriver.dll` so that `STLinkInterface::LoadStlinkLibrary()` finds it at runtime. The proper location depends on the application calling context.

1.4 Licensing

STLINK-V3-BRIDGE is delivered under the *ULTIMATE LIBERTY* software license agreement (SLA0044).

2 STLINK-V3 bridge API v1.3.0 release information

2.1 New features

- Added routine and parameters for the support of the I²C repeated start feature (restart instead of stop between two I²C transactions). The support for the I²C repeated start feature starts from firmware versions v3JxB6 (v3J17M10B6S1) or v4JxB3 (v4J7B3P6).
  - Added the `Brg::WriteReadI2C()` I²C high-level routine for Write-repeated start-Read transaction: I²C writes up to four bytes before repeated start and read.
  - Added the I2cTransPrepT and I2cTransForceT parameters in I²C low-level routines (`Brg::StartReadI2C()`, `Brg::ContReadI2C()`, `Brg::StopReadI2C()`, and `Brg::StartWriteI2C()`, `Brg::ContWriteI2C()`, `Brg::StopWriteI2C()`). Repeated start generation uses I2C_PREP_RESTART and I2C_FORCE_RESTART.
- Improved the application path handling to allow Unicode® or multibyte character set (main changes in `STLinkInterface` class).
- Improved the factorization of interface names (in `STLinkInterface` class).
- Reworked `platform_include.h` to manage the character set indifferently:
  - On Linux®, macOS®, and Windows®
  - Through Visual Studio® or not
  - With MFC or not

2.2 Known problems and limitations

Refer to the `STLink_Bridge_API.chm` file in the delivery package.

STLINK-V3 (firmware version v3JxBxxx) and STLINK-V3PWR (firmware version v4JxBxxx) do not have the same CAN hardware:

- The differences in the support and possible configuration of the CAN filter parameters are detailed in the *Limitations* chapter of the `STLink_Bridge_API.chm` document.
- FDCAN API is not supported by STLINK-V3 (firmware version v3JxBxxx) hardware.
- STLINK-V3PWR (firmware version v4JxBxxx) supports both CAN and FDCAN APIs but they are mutually exclusive and cannot be mixed.
- FDCAN API can be used for both classical CAN or CAN FD communications, while CAN API supports only classical CAN communications.

3 Release information for previous releases

3.1 STLINK-V3 bridge API v1.2.0 release information

3.1.1 New features

- Updated for STLINK-V3PWR support: `Brg::InitFilterCAN()`, CAN limitation (no 16-bit filter support, `Brg::IsCanFilter16Support()`).
- Added routines for configuring the CAN FD connections of STLINK-V3PWR. The COM_FDCAN support starts from firmware version v4JxB2xx.
- Added routines for transferring data through the CAN FD interface between STLINK-V3PWR and a running target.
- Applied a fix to the CAN 32-bit filter configuration (`Brg::FormatFilter32bitCAN()`).
- Updated the CAN `Brg::GetCANBaudratePrescal()` and `Brg::InitCAN()` routines for shared code with CAN FD routines (`Brg::CheckBitTimeClassicCAN()`).
- Updated the `.cpp` and `.h` files in `src/sharedStlk/common/`, using the last version shared with other USB ST-LINK interfaces without impact for USB bridge interface.
- Updated the file tree and improved the `.h` structure, documentation, and project examples:
  - Replaced the `stdafx.h` include by a generic `platform_include.h` to be updated according to user's needs.
  - Clarified the supported SPI communication (`Brg_SpiDirT` description).
  - Moved the `src/` folder to `src/sharedStlk/`.
  - Moved the `example/bridge/` folder to `src/example/bridge/`, moved the `example/bridge/main_example.cpp` file to `src/project/apiBridgeProject/main_example.cpp`.
  - Added Windows® Visual Studio® 2019 project example in `src/project/apiBridgeProject/` using `main_example.cpp`.
  - Updated `main_example.cpp` to add simple GPIO, CAN, CAN FD, and SPI interface tests, and USB bridge ST-LINK communication management.
  - Improved `STLink_Bridge_API.chm` and updated it with STLINK-V3PWR (firmware version v4JxBxxx) and FDCAN API.

3.1.2 Known problems and limitations

Refer to the `STLink_Bridge_API.chm` file in the delivery package.

STLINK-V3 (firmware version v3JxBxxx) and STLINK-V3PWR (firmware version v4JxBxxx) do not have the same CAN hardware:

- The differences in the support and possible configuration of the CAN filter parameters are detailed in the *Limitations* chapter of the `STLink_Bridge_API.chm` document.
- FDCAN API is not supported by STLINK-V3 (firmware version v3JxBxxx) hardware.
- STLINK-V3PWR (firmware version v4JxBxxx) supports both CAN and FDCAN APIs but they are mutually exclusive and cannot be mixed.
- FDCAN API can be used for both classical CAN or CAN FD communications, while CAN API supports only classical CAN communications.

3.2 STLINK-V3 bridge API v1.0.0 release information

3.2.1 New features

- C++ source code implementing the bridge interface of STLINK-V3 on a personal computer
- Routines for configuring the SPI, I^2C, CAN, and GPIO connections of STLINK-V3
- Routines for transferring data through the SPI, I^2C, and CAN interfaces between STLINK-V3 and a running target

3.2.2 Known problems and limitations

Refer to the `STLink_Bridge_API.chm` file in the delivery package.

Revision history

Table 2. **Document revision history**

Date	Revision	Changes
10-May-2019	1	Initial release.
14-Jan-2025	2	Updated for STLINK-V3 bridge API v1.2.0 featuring STLINK-V3PWR and CAN FD support: • Updated the title and cover image • Updated *STLINK-V3 bridge API release summary* and *STLINK-V3 bridge API v1.2.0 release information* • Updated the sections *Overview*, *Host PC system requirements*, and *Setup procedure*
11-Dec-2025	3	Updated for STLINK-V3 bridge API v1.3.0 featuring I^2C repeated start: • Updated STLINK-V3 bridge API release summary and New features

Contents

List of tables