J-Link / J-Trace Getting Started

Document: UM08001
Software Version: 9.10
Revision:
Date: January 14, 2026

A product of SEGGER Microcontroller GmbH

www.segger.com

Table of contents

Chapter 1

Introduction

Thank you for choosing J-Link / J-Trace as your debug probe / trace probe.

This manual presents a quick start guide for J-Link / J-Trace and the associated J-Link Software Pack, supported on Windows, Linux and macOS.

SEGGER's motto is "It simply works!", which also applies to getting started with debug and trace probes and the related software.

To access the complete online J-Link / J-Trace User Guide, please click on the link below.

- *J-Link / J-Trace online User Guide*

Please also consider these helpful online resources:

- *Feature comparison between J-Link and J-Trace*
- *Detailed J-Link model feature comparison*
- *Detailed J-Trace model feature comparison*

Chapter 2

Features

J-Link and J-Trace are packed with features. This chapter provides an overview.

2.1 Ultra-fast download speeds

SEGGER's ultra-fast flash programming algorithms allow for the fastest flash writes in the industry. A high-end J-Link or a J-Trace can flash at speeds reaching 1 Megabyte per second. SEGGER probes aren't limited to internal flash; they can also flash external serial NOR flash memory, either through the target's SPI interface, or through direct SPI communication at up to 50 MHz.

The download speed into RAM is up to 4 MB/s.

- *More on download speed*

2.2 Unlimited breakpoints in flash memory

CPU debug units have a fixed number of hardware breakpoints (for example, 4-6 on Cortex-M devices). SEGGER probes can place an unlimited number of breakpoints in both internal and external flash. As long as the external flash is mapped in memory space, SEGGER probes can set breakpoints to halt the program and enable fast line-by-line execution.

- *More on unlimited flash breakpoints*

2.3 Real-Time Transfer

SEGGER's Real-Time Transfer (RTT) is a technology exclusive to SEGGER probes for interactive user I/O in embedded applications. It combines the advantages of SWO and semihosting at very high performance. With RTT, it is possible to output information from the target microcontroller as well as sending input to an application on the host computer (e.g. SEGGER's SystemView) at a very high speed (up to 3.5 MB/s) without affecting the target's real time behavior. RTT can be used with any J-Link / J-Trace model and any supported target processor that allows background memory access, e.g. Cortex-M, Renesas RX, and certain RISC-V targets.

RTT supports multiple channels in both directions, up to the host and down to the target, which can be used for different purposes and provide the most possible freedom to the user. RTT uses the debug channel for communication, which means no additional hardware or pins are required on the target. The performance of RTT is significantly higher than any other technology used to output data to a host PC. An average line of text can be output in one microsecond or less — basically only the time to do a single `memcopy()`.

- *More on SEGGER RTT*

2.4 Built-in virtual COM port functionality (VCOM)

Most J-Link models come with built-in virtual COM port (VCOM) functionality. This means that in addition to the regular J-Link debug functionality, J-Link will also show up as a COM port in the device manager of the operating system. This VCOM port can be very useful for logging, diagnostics and application control.

- *More on the VCOM port*

2.5 Monitor mode debugging

Monitor mode debugging describes the capability of the CPU to maintain essential functionality while being debugged. This is particularly important when hardware such as a motor has to keep running, or communication links have to stay connected. The default debug mode for most common CPUs is "halt-mode debugging" where the CPU halts on a debug request, causing the user application and, depending on the CPU, also peripherals to stop execution. However, for certain applications and environments it may be necessary that parts of the user application continue execution while the CPU is in debug mode and the

user is debugging a different part of the application. This is where monitor mode debugging provides essential improvement.

- *More on monitor mode debugging*

2.6 Extensive core / architecture support

SEGGER probes support a wide variety of cores / architectures, including 32-bit and 64-bit Arm Cortex-A and Cortex-R, Cortex-M, legacy ARM7/9/11, Renesas RX, 32-bit and 64-bit RISC-V, Cadence Tensilica, Silicon Labs 8051, and Microchip PIC32Mx (MIPS).

- *List of cores supported by J-Link / J-Trace*

2.7 Extensive device support

SEGGER probes support an extensive array of devices, with new ones being added frequently. The list of supported manufacturers, families, devices, and SoCs includes tens of thousands of devices in hundreds of device families.

- *List of devices supported by J-Link*
- *List of devices tested for tracing with J-Trace*
- *List of supported SPI flash devices*

There is also a large variety of debug adapters, trace adapters, and isolators available for J-Link and J-Trace.

- *List of J-Link debug adapters available*
- *List of J-Link isolators available*
- *List of J-Trace trace adapters available (Go to Accessories > Adapters)*
- *List of J-Trace trace isolators available (Go to Accessories > Isolators)*

In addition, there are several trace reference boards available for use with J-Trace.

- *List of J-Trace trace reference boards available*

2.8 Comprehensive software package

The J-Link Software and Documentation Package includes a significant number of tools that make a developer's job easier and extend the capabilities of J-Link / J-Trace. Almost all J-Link tools have multi-platform support and run on Windows, Linux and macOS. All SEGGER probes include free software and firmware updates.

- *Download the J-Link SW and Documentation Package*

2.9 J-Link SDK / J-Link DSK

The **J-Link Software Development Kit (SDK)** allows users to integrate J-Link / J-Trace support into their own applications. This is used in professional IDEs, such as SEGGER's Embedded Studio and others, to allow debugging directly via a J-Link / J-Trace, as well as in customized production utilities. The J-Link SDK is available for Windows, Linux and macOS, as 32-bit and 64-bit versions and can be used with nearly every programming language or solution. The J-Link / J-Trace integration is done via a standard DLL / shared library and provides easy-to-use C-language API functions.

- *More on the J-Link SDK*

The **J-Link Device Support Kit (DSK)** enables silicon vendors and users to add debug and flash programming support for new devices on their own. A new device usually requires a flash loader and (in some cases) also a script defining special connect and reset sequences required by the device. The J-Link DSK comes with the SEGGER flash loader as well as a set of example script files for various devices that require special handling.

- *More on the J-Link DSK*

2.10 Powerful customization options using J-Link script files

In some situations, it is necessary to customize some actions performed by J-Link / J-Trace. In most cases, it is the connection sequence and/or the way in which a reset is performed, since some custom hardware needs some special handling which cannot be integrated into the generic part of the J-Link software. J-Link script files are written in C-like syntax in order to have an easy start to learning how to write J-Link script files. The script file syntax supports most statements (if-else, while, declaration of variables, …) which are allowed in C (but not all of them). Moreover, there are some statements that are script file specific. The script file allows maximum flexibility, so almost any target initialization necessary can be supported.

- *More on J-Link script files*

2.11 Extensive ecosystem

SEGGER probes can be used in a wide array of IDEs and debuggers on multiple platforms.

2.11.1 Ozone

Ozone is SEGGER's own full-featured graphical debugger for embedded applications. It is tailored to support the full extend of the J-Link / J-Trace debug feature set, including full real-time trace support with J-Trace. More information about Ozone can be found later in this document.

2.11.2 Embedded Studio

SEGGER's Embedded Studio is a comprehensive IDE designed specifically for managing, building, testing, and deploying embedded applications. It is optimized to work with J-Link / J-Trace to provide the best development experience. More information about Embedded Studio can be found later in this document.

2.11.3 Other IDEs and tool chains

J-Link / J-Trace can be used with almost any other IDE include Keil, IAR, various Eclipse-based solutions, Renesas e2 Studio, Silicon Labs Simplicity Studio, etc.

- *List of supported IDEs*

2.11.4 GDB based setups

J-Link / J-Trace can be used with GDB-based setups. The GNU Debugger (GDB) is the de facto debugger for development on Linux systems. However, it has now found its way into embedded development (even without Linux running on the target system). GDB provides a standardized interface / API that can be used by an IDE. It also specifies a standardized protocol (GDB remote protocol), which allows GDB to communicate with a GDB Server which knows how to handle the debug probe connected to the target. The J-Link Software and Documentation Package comes with the J-Link GDB Server which allows using J-Link in GDB-based setups.

- *More on J-Link GDB Server*

2.11.5 LLDB

J-Link / J-Trace can also be used with LLDB. Originally, GNU toolchains provided GCC as a compiler and GDB as a debugger. Since Clang's introduction as a compiler, LLDB was introduced (which was essentially a GDB successor). In terms of protocol, it is backward compatible to GDB whilst the API for the IDE is slightly different, so it can also be used with the J-Link GDB Server, too.

Chapter 3

First steps

This chapter describes how to get up and running with J-Link / J-Trace.

3.1 Downloading and installing the J-Link Software and Documentation Pack

The J-Link Software and Documentation Pack is required for proper operation of the J-Link / J-Trace. It can be downloaded from the following web page:

- *J-Link Software and Documentation Pack*

3.1.1 Windows

Download the installer for the x86, x64 or arm64. Double-click on the downloaded file to begin the installation process, and follow the instructions provided.

By default, the J-Link SW Pack will be installed into the `Program Files\SEGGER\JLink_Vxxx` folder. It can also be accessed via the Windows Start menu.

- *Additional information on the Windows installer for the J-Link SW Pack*

3.1.2 MacOS

Download the installer for the x64 or the Apple silicon (or download the universal installer). Double-click on the downloaded file to begin the installation process, and follow the instructions provided. The J-Link SW Pack will be installed into the `Applications/SEGGER/JLink_Vxxx` folder.

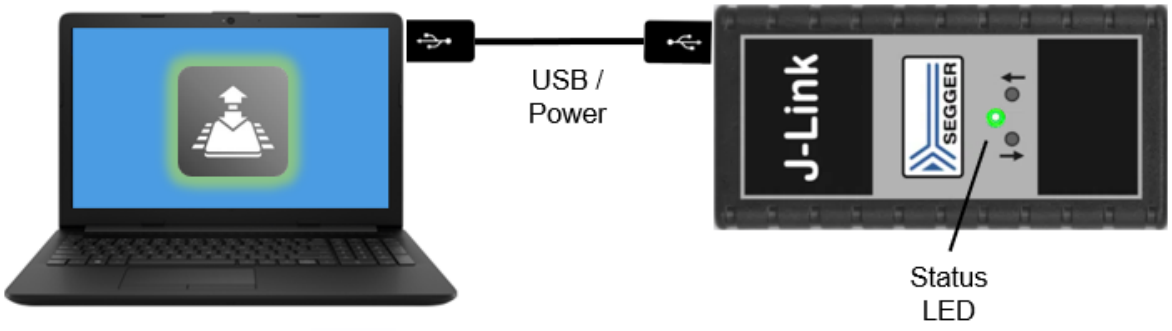
3.1.3 Linux

The Linux version of the J-Link Software Package is available for x86, x64, arm32 & arm64. The package is available as either DEB, RPM or TGZ files. It is entirely self-contained, requiring no external libraries, and as such can be used on every Linux distribution. Download the desired / appropriate file and unpack the downloaded archive to a desired location, such as `/opt/SEGGER/JLink`, or `/usr/local/SEGGER/JLink`.

3.2 Connecting the J-Link / J-Trace to the host computer via USB

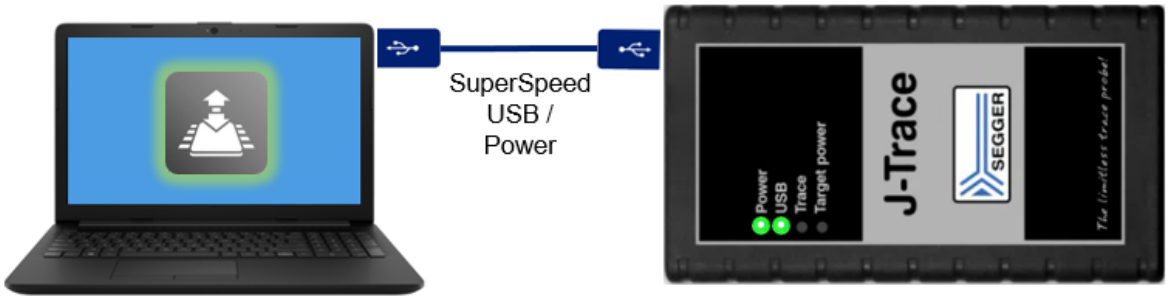
Connect the J-Link / J-Trace to your host computer using the included USB cable. This will also power the probe.

On the J-Link, the green status LED should light up.



> **Note**
>
> Your J-Link may look different depending on the J-Link model you purchased.

On the J-Trace, both the green Power LED and the green USB LED should light up.



> **Note**
>
> If the status LED on the J-Link or the USB LED on the J-Trace continue to flash rapidly, there is a problem with the enumeration of the SEGGER probe.

- *SEGGER Knowledgebase article for troubleshooting this issue*

3.3 Registering the J-Link / J-Trace

It is highly recommended to register the J-Link / J-Trace. Registration is quick, easy, and has the following benefits:

- Verify that your J-Link / J-Trace is an authentic SEGGER product
- Receive up-to-date information on software and product updates
- Gain access to the SEGGER technical support system
- *Knowledgebase article for instructions on how to register*
- *Knowledgebase article for information on technical support for SEGGER probes*

3.4 Verifying the connection to the host computer

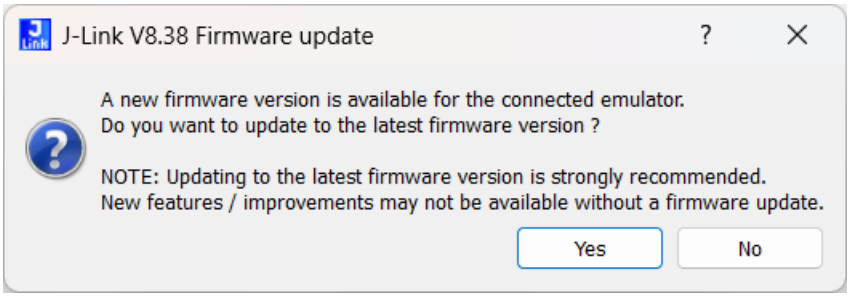
You can use the J-Link Commander utility to verify that the host computer connection to the J-Link / J-Trace is working properly. J-Link Commander is part of the J-Link SW Pack you installed earlier.

To start J-Link Commander under Windows, access the Start menu and select J-Link Commander from the list of available software utilities in the SEGGER - J-Link Vx.xx folder.

When using macOS, navigate to the /Applications/SEGGER/JLink_Vxxx folder and double-click JLinkExe.

Under Linux, open a terminal and navigate to the J-Link SW installation directory using the cd command, e.g. cd /opt/SEGGER/JLink. Then run the J-Link Commander executable: ./JLinkExe.

If J-Link Commander detects that a firmware update is available for the J-Link / J-Trace, it will prompt you as shown below:



Click Yes to allow J-Link Commander to update the firmware.

After J-Link Commander has successfully connected to the J-Link / J-Trace, you will see console output similar to the one below (taken from Windows), providing important information about the connected probe:

```
J-Link Commander V8.3
SEGGER J-Link Commander V8.38 (Compiled May 28 2025 12:43:30)
DLL version V8.38, compiled May 28 2025 12:42:34

Connecting to J-Link via USB...Updating firmware:  J-Link Pro V5-1 compiled Apr  1 2025 10:04:08
Replacing firmware: J-Link Pro V5-1 compiled Feb 20 2025 16:25:57
Waiting for new firmware to boot
New firmware booted successfully
O.K.
Firmware: J-Link Pro V5-1 compiled Apr  1 2025 10:04:08
Hardware version: V5.10
J-Link uptime (since boot): 0d 00h 00m 00s
S/N: 175103366
License(s): RDI, FlashBP, FlashDL, JFlash, GDB
USB speed mode: Full speed (12 MBit/s)
IP-Addr: DHCP (no addr. received yet)
VTref=0.000V


Type "connect" to establish a target connection, '?' for help
J-Link>
```

Note

If J-Link Commander is not able to connect to the SEGGER probe, please refer to this SEGGER Knowledgebase article for troubleshooting this issue:

- *Knowledgebase article for troubleshooting this issue*

3.5 Connecting the J-Link to a target board

3.5.1 J-Link target connector

The J-Link has one target connector as shown in the image below:

The pinout for the 20-pin, 0.1"-pitch debug connector depends on the debug interface used.

For more details on the pinout, please see here:

- *J-Link target connector pinout*

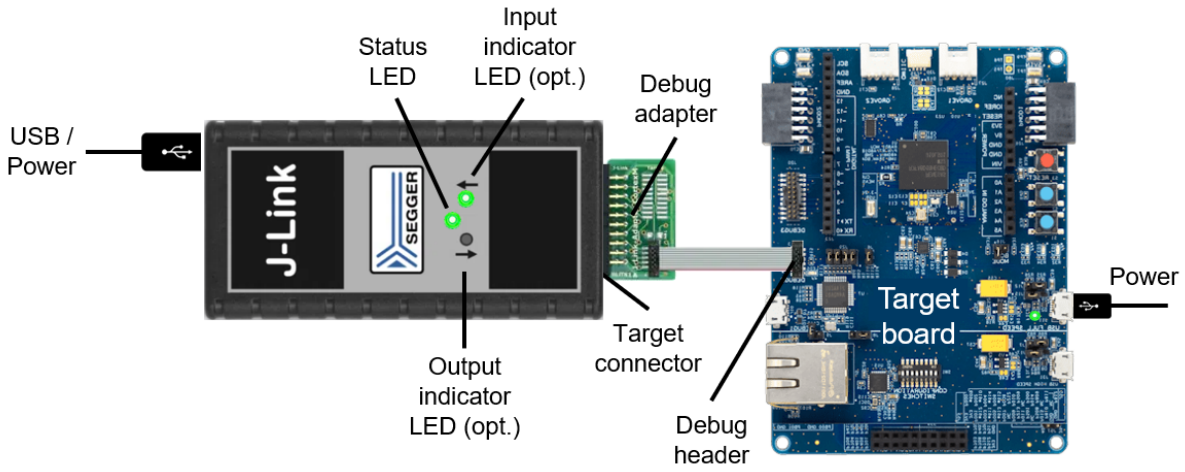
J-Link ships with a matching 20-pin ribbon cable. Whether or not you will need a separate debug adapter depends on the type of debug header on the target board.

You can see the extensive list of debug adapters available from SEGGER here:

- *List of available debug adapters*

3.5.2 Connecting the J-Link to a target board

Below is an example for connecting the J-Link to a target board:



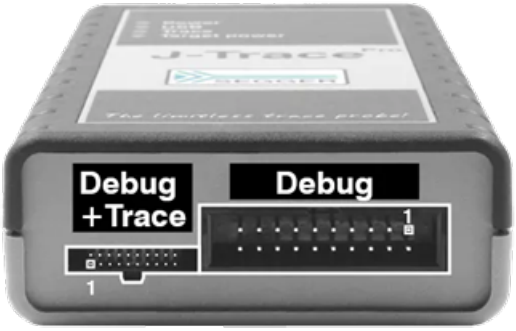
Connect the debug adapter (or the 20-pin ribbon cable, if possible) to the J-Link target connector as well as the debug header on the target board as shown in the image above. Watch out for the correct orientation. The colored wire of the ribbon cable needs to connect to pin 1 of the debug header. Also make sure the connector covers both rows of the debug header.

Then provide power to the target board. The input indicator LED on the J-Link (if so equipped) will light up (as well as typically some power LED on the target board).

3.6 Connecting the J-Trace to a target board

3.6.1 J-Trace target connectors

The J-Trace has two target connectors as shown in the image below:



The 20-pin, 0.05"-pitch debug & trace target connector is used for advanced debugging (instruction trace, code coverage, code profiling, etc.). The pinout of this connector can be found in the SEGGER knowledge base:

- *J-Trace debug & trace connector pinout*

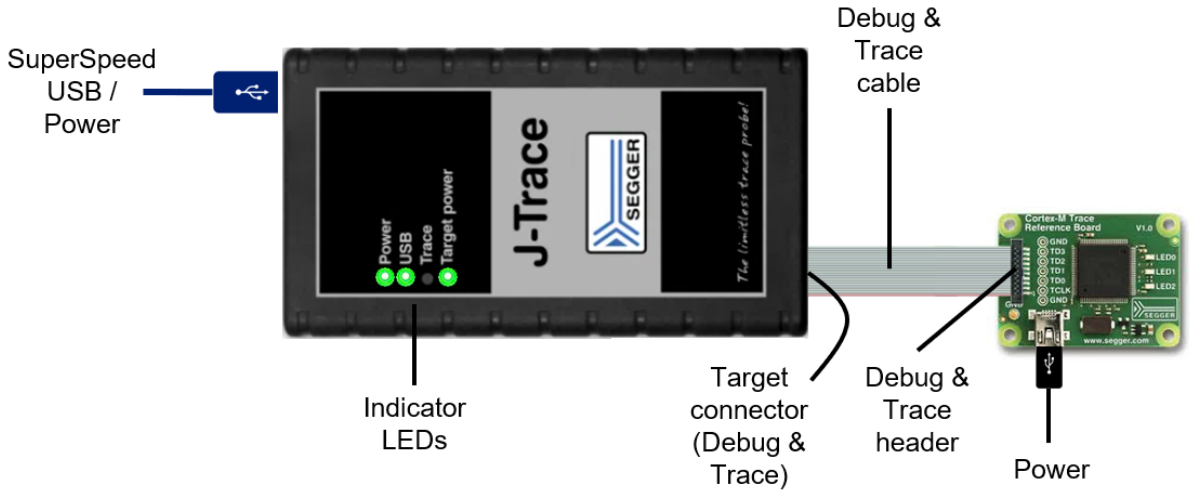
The 20-pin, 0.1"-pitch debug connector is identical to the target connector on the J-Link and is used for basic debugging only. It can be used with the same target debug adapters as the J-Link. Its exact pinout depends on the debug interface used.

- *J-Trace/J-Link debug connector pinout*

Please note that only one of these connectors can be used at any time. Please also note the positions of pin 1 on each of the connectors.

3.6.2 Connecting the J-Trace to a target board

The image below shows the J-Trace connected to the Cortex-M trace reference board (shipped with the J-Trace) for the purpose of using advanced debug features like instruction trace, code profiling, and code coverage:



J-Trace ships with a 20-pin, 0.05"-pitch ribbon cable. In rare cases, depending on the target board, you might need a separate trace adapter.

- *List of trace adapters available from SEGGER (Go to Accessories > Adapters)*

Connect the 20-pin, 0.05"-pitch ribbon cable to the J-Trace target connector as well as the matching debug & trace header on the target board as shown in the image above. Watch

out for the correct orientation. The red wire of the ribbon cable needs to connect to pin 1 of the debug & trace header. Also make sure the connector covers both rows of the header.

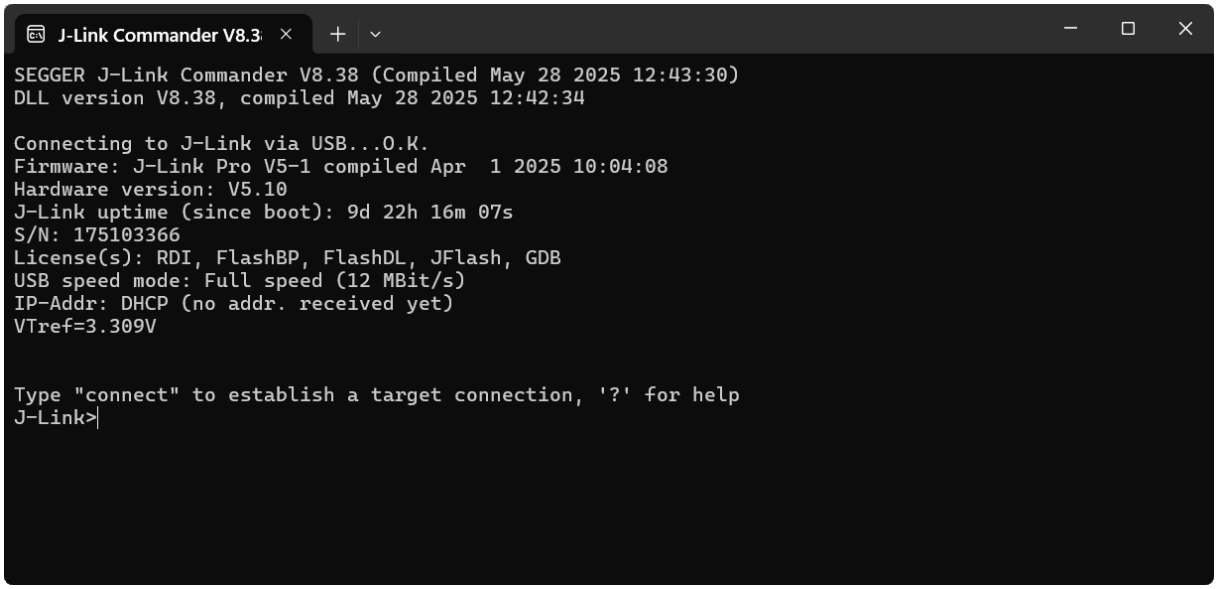
Then provide power to the target board. The target power indicator LED on the J-Trace will light up. Please note that the Cortex-M trace reference board does not have a dedicated power LED.

> **Note**
>
> If your target board doesn't have a 20-pin, 0.05"-pitch debug & trace header, you can still use the J-Trace for standard debugging. In this case, you need to use the 20-pin, 0.1"-pitch debug target connector. Just like with the J-Link, the pinout of this connector depends on the debug interface used.

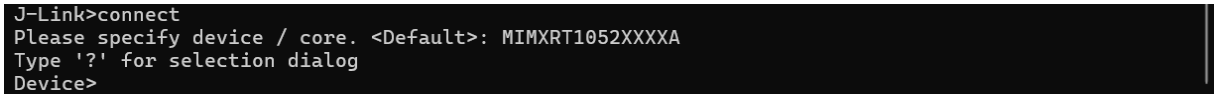
3.7 Verifying the connection to the target device

If your SEGGER probe is connected to both the host computer as well as the target board (as described in the previous sections), and your target board has power, you can verify the connection between the probe and the target device.

Start J-Link Commander as outlined in section *Verifying the connection to the host computer* on page 11.



Then enter `connect` to establish a target connection. If the target device shown as `<Default>` (saved from a previous debug session) matches your target device, you can simply confirm the device with `Enter`. Otherwise, enter `?` to select the target device via the selection dialog.



> **Note**
>
> Sometimes, J-Link / J-Trace already supports devices that are not yet available publicly, which means they are not yet listed in the selection dialog. If you are working

with such a device, you may enter the exact device name that was provided to you (either by SEGGER or by the device manufacturer) at the `Device>` prompt.

Next, specify the target interface you would like to use to connect with the device at the `TIF>` prompt.

```
Specify target interface speed [kHz]. <Default>: 4000 kHz
Speed>10000
```

Note

The maximum target interface speed depends on the J-Link model used.

The SEGGER probe will then establish the connection to the selected target device via the selected interface at the selected speed.

```
Device "MIMXRT1052XXXXA" selected.
Connecting to target via SWD
Found SW-DP with ID 0x0BD11477
DPIDR: 0x0BD11477
CoreSight SoC-400 or earlier
Scanning AP map to find all available APs
AP[1]: Stopped AP scan as end of AP map has been reached
AP[0]: AHB-AP (IDR: 0x04770041, ADDR: 0x00000000)
Iterating through AP map to find AHB-AP to use
AP[0]: Core found
AP[0]: AHB-AP ROM base: 0xE00FD000
CPUID register: 0x411FC271. Implementer code: 0x41 (ARM)
Cache: L1 I/D-cache present
Found Cortex-M7 r1p1, Little endian.
FPUnit: 8 code (BP) slots and 0 literal slots
CoreSight components:
ROMTbl[0] @ E00FD000
[0][0]: E00FE000 CID B105100D PID 000BB4C8 ROM Table
ROMTbl[1] @ E00FE000
[1][0]: E00FF000 CID B105100D PID 000BB4C7 ROM Table
ROMTbl[2] @ E00FF000
[2][0]: E000E000 CID B105E00D PID 000BB00C SCS-M7
[2][1]: E0001000 CID B105E00D PID 000BB002 DWT
[2][2]: E0002000 CID B105E00D PID 000BB00E FPB-M7
[2][3]: E0000000 CID B105E00D PID 000BB001 ITM
[1][1]: E0041000 CID B105900D PID 001BB975 ETM-M7
[1][2]: E0042000 CID B105900D PID 004BB906 CTI
[0][1]: E0040000 CID B105900D PID 000BB9A9 TPIU-M7
[0][2]: E0043000 CID B105F00D PID 001BB101 TSG
I-Cache L1: 32 KB, 512 Sets, 32 Bytes/Line, 2-Way
D-Cache L1: 32 KB, 256 Sets, 32 Bytes/Line, 4-Way
Memory zones:
  Zone: "Default" Description: Default access mode
Cortex-M7 identified.
J-Link>
```

Congratulations, you are up and running now!

Note

If J-Link Commander reports an error, i.e. the SEGGER probe is not able to connect to the target device, please refer to this SEGGER Knowledgebase article for troubleshooting this issue:

- *Knowledgebase article for troubleshooting this issue*

Chapter 4

Working with J-Link / J-Trace

The previous chapter provided you with a good foundation of how to get up and running with the J-Link / J-Trace. To further augment your J-Link / J-Trace experience, SEGGER offers additional software tools to help you develop and test your systems, which are covered in this chapter.

Embedded Studio is the all-in-one solution for managing, building, testing, and deploying embedded applications. It ensures smooth and efficient development, and it offers a wide range of features.

Ozone is a multi-platform debugger and performance analyzer for J-Link and J-Trace.

SystemView is a powerful tool for verification and validation. It records, analyzes, and visualizes captured data in real time; documents tasks, interrupts, and user events; and much more.

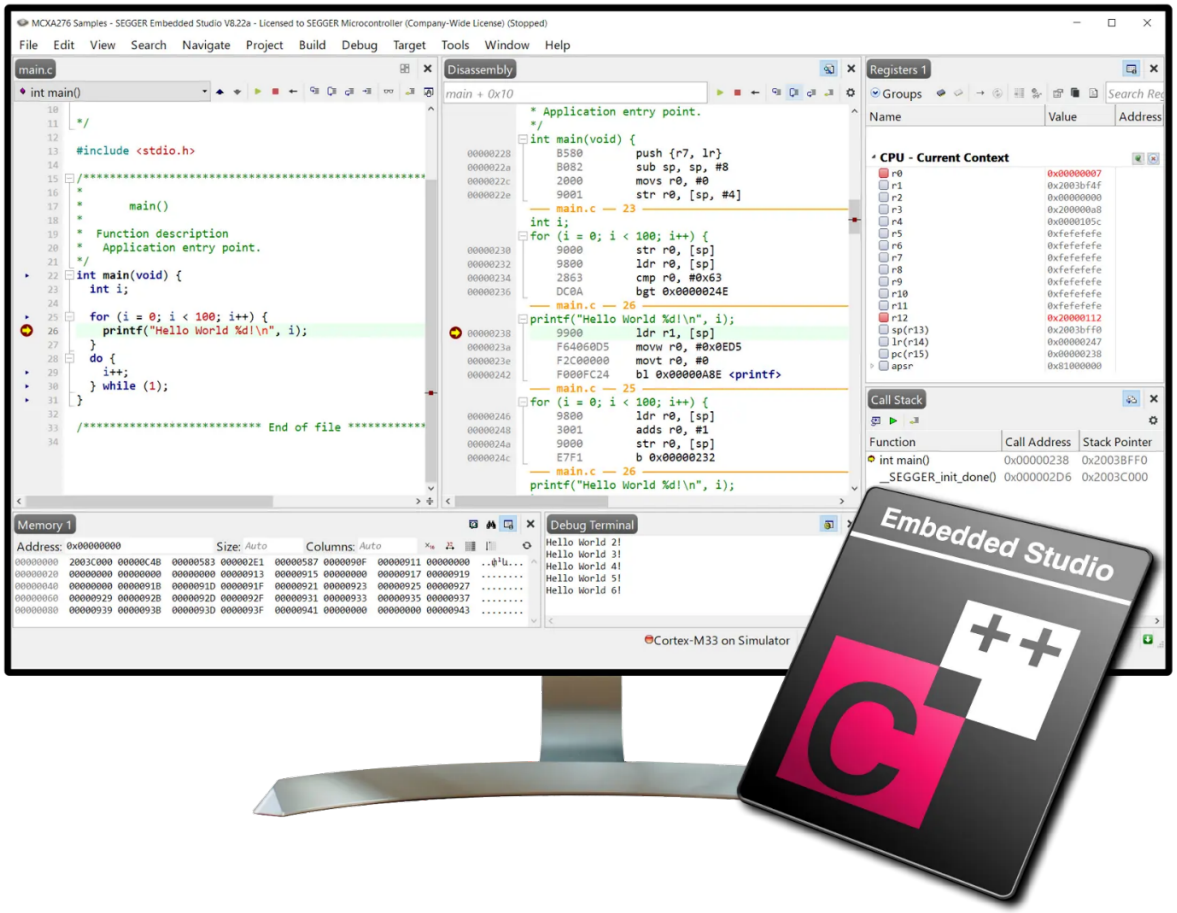
J-Link and J-Trace are also supported by a large number of **3rd-party IDEs and debuggers**, including Visual Studio Code and Eclipse-based solutions.

- *List of supported IDEs*
- *Using J-Link with VS Code*

Furthermore, the **J-Link Software Package** includes a large number of additional useful tools and utilities for working with J-Link. Please see chapter *Software Package Overview* on page 21 for details.

4.1 Embedded Studio IDE

SEGGER's Embedded Studio is a comprehensive integrated development environment (IDE) designed specifically for managing, building, testing, and deploying embedded applications. This means smooth, efficient development operations thanks to a wide range of features. Tailored to meet the needs of developers working with a variety of microcontrollers, Embedded Studio streamlines the entire development process, reducing time-to-market and ensuring reliable, robust performance.



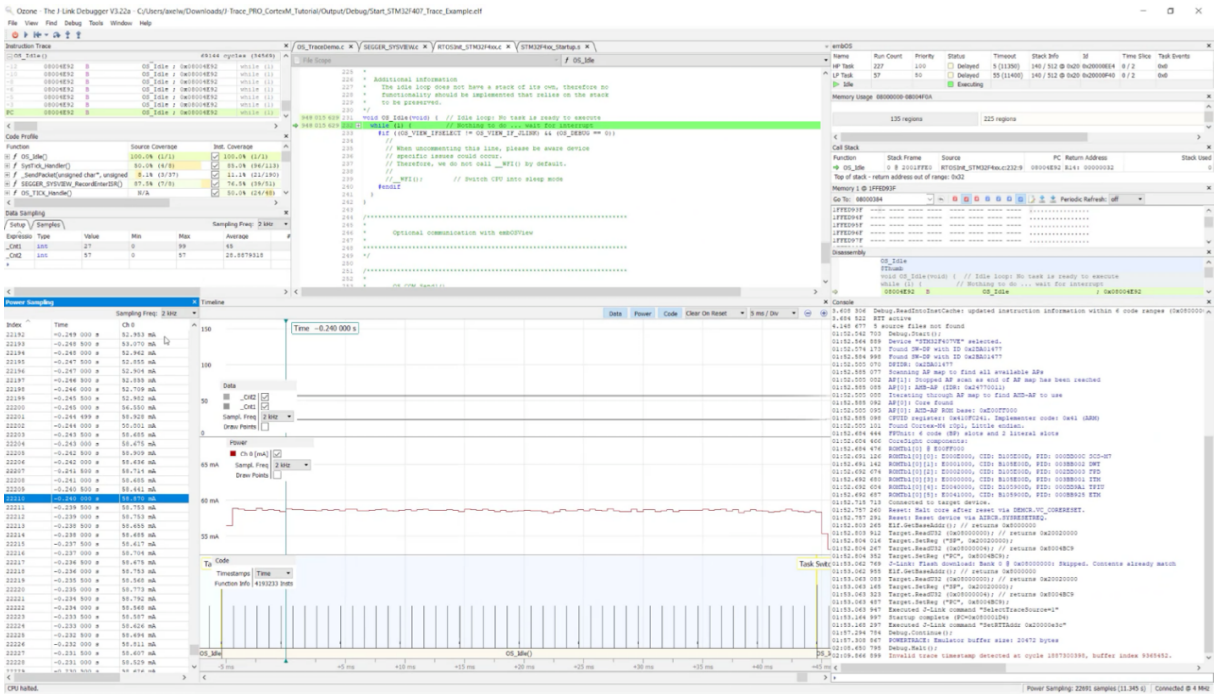
Embedded Studio provides a comprehensive set of tools, including an editor, compiler, debugger, and simulator — all in a single application. Embedded Studio PRO introduces a wide range of libraries and example projects, allowing users to quickly implement common functionalities and accelerate development.

The ability to integrate critical elements streamlines the development process by eliminating the need to switch between multiple tools. With its Visual Studio-like interface, Embedded Studio also offers user-friendliness that enhances developer productivity.

- *Embedded Studio product page*
- *Embedded Studio documentation*

4.2 Ozone

Ozone is a full-featured graphical debugger for embedded applications. With Ozone, it is possible to debug any embedded application on C/C++/Rust source and assembly level. Ozone can load applications built with any tool chain / IDE or debug the target's resident application without any source. Ozone includes all well-known debug controls and information windows and makes use of the best performance of J-Link and J-Trace debug probes. The user interface is designed to be used intuitively and is fully configurable.



Ozone is more than a simple debugger. Its various features, including trace, code profiling and code coverage analysis make it a powerful performance analyzer, enabling you to get full system insight, to track down inefficiencies and bugs, and to make your products even better.

- *Ozone product page*
- *Ozone documentation*

If you have a J-Trace, the Ozone trace tutorial is an excellent starting point:

- *Ozone / J-Trace trace tutorial*

When setting up trace on your own target board, look here for help:

- *Setting up trace on your own target board*

4.3 SystemView

SystemView is a real-time recording and visualization tool designed to analyze and profile the behavior of embedded systems. It offers deep insights into runtime behavior, surpassing the capabilities of traditional debuggers. Ideal for complex systems with multiple threads and interrupts, SystemView helps developers ensure their systems perform as intended, identify inefficiencies, and uncover unintended interactions or resource conflicts. By recording monitor data from the embedded system via the debug interface, SystemView visualizes tasks, interrupts, and software timer execution in detail, documenting their frequency, order, and execution time.

SystemView provides powerful features while being minimally intrusive, which makes it an essential tool for embedded system development. It works with any CPU, RTOS, or bare-metal system, ensuring high flexibility and adaptability to different setups. With this broad compatibility, developers can use SystemView in a wide variety of embedded projects.

- *SystemView product page*
- *SystemView documentation*

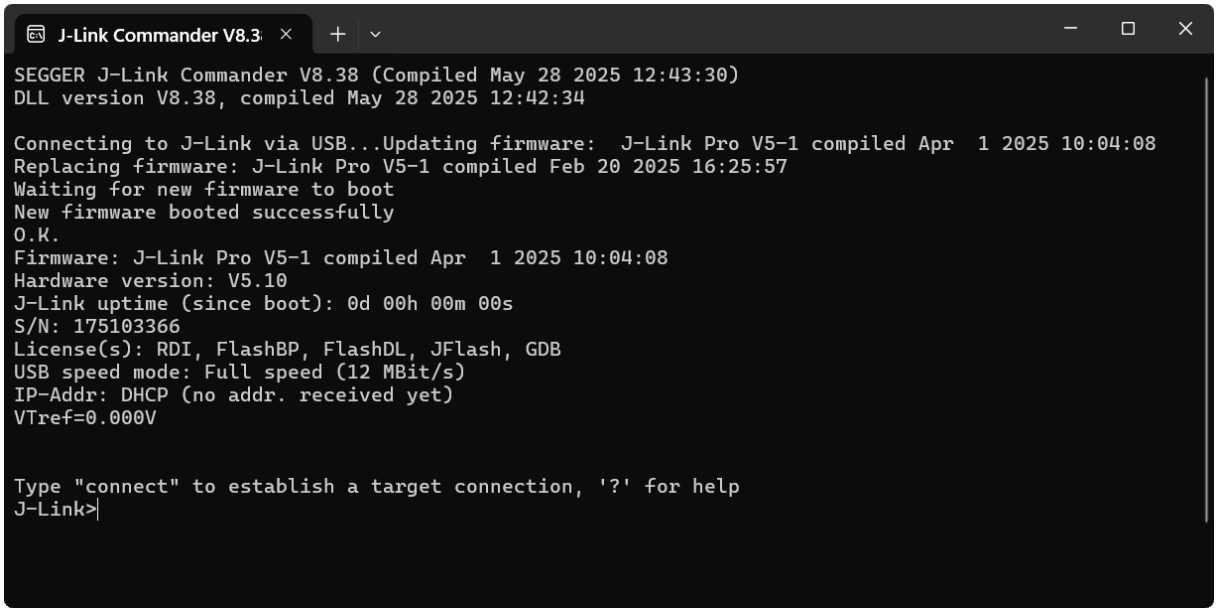
Chapter 5

Software Package Overview

This chapter covers the contents of the J-Link Software Pack, an extensive array of software to fully complement the J-Link / J-Trace.

5.1 J-Link Commander

J-Link Commander (JLink.exe / JLinkExe) is a command-line based utility that can be used for verifying proper functionality of J-Link / J-Trace, as well as for simple analysis of the target system via J-Link / J-Trace. It supports a variety of very useful commands for J-Link / J-Trace configuration, connecting to the target, debugging, flash programming, and more. J-Link Commander is also able to update (or downgrade) the J-Link / J-Trace firmware. J-Link Commander is part of the J-Link Software and Documentation Pack, which is available for download from segger.com.



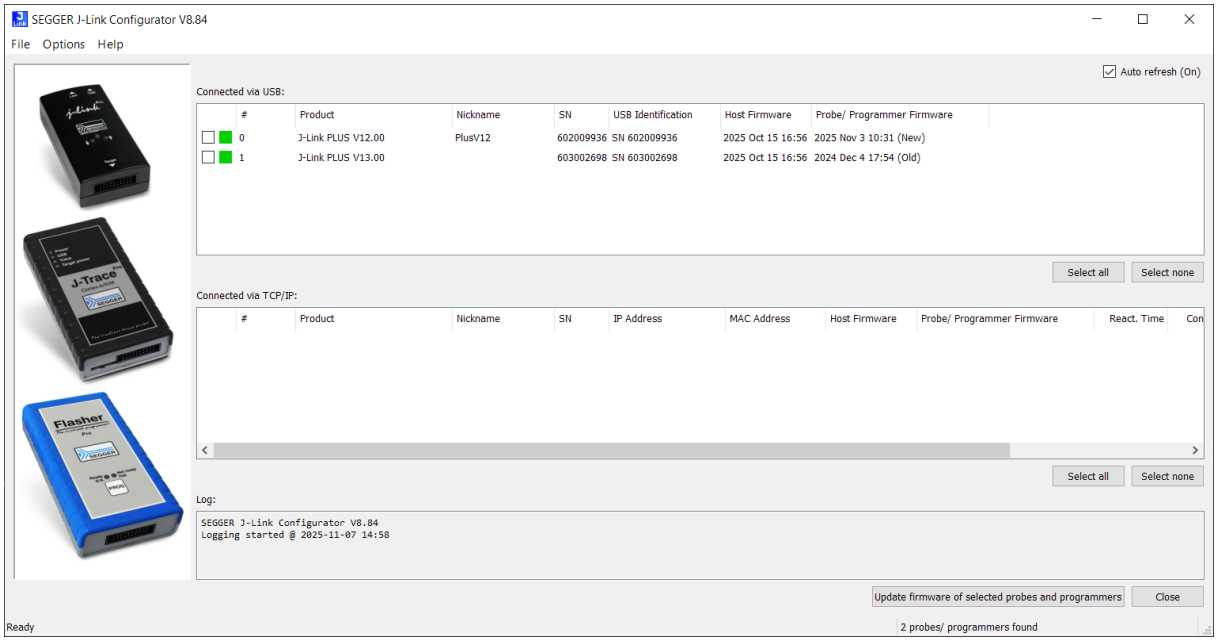
- *J-Link Commander product page*
- *J-Link Commander documentation*
- *J-Link Commander introductory video*

5.2 J-Link Configurator

J-Link Configurator is a multi-platform (Windows / Linux / macOS) GUI application to configure certain features of a SEGGER J-Link or J-Trace. It is part of the J-Link Software and Documentation Pack available from segger.com.

The following settings can be changed via J-Link Configurator:

- Updating the firmware of multiple SEGGER probes connected via USB or TCP/IP
- Configuration of the IP settings (use DHCP, IP address, subnet mask, …) of a J-Trace or a J-Link supporting the Ethernet interface
- Enabling/Disabling VCOM functionality, if supported by the SEGGER probe
- Adding a nickname to a SEGGER probe
- Re-configuring older J-Links to be identified by their real serial number when enumerating on the host computer (required when connecting > 3 older J-Links to one host computer)

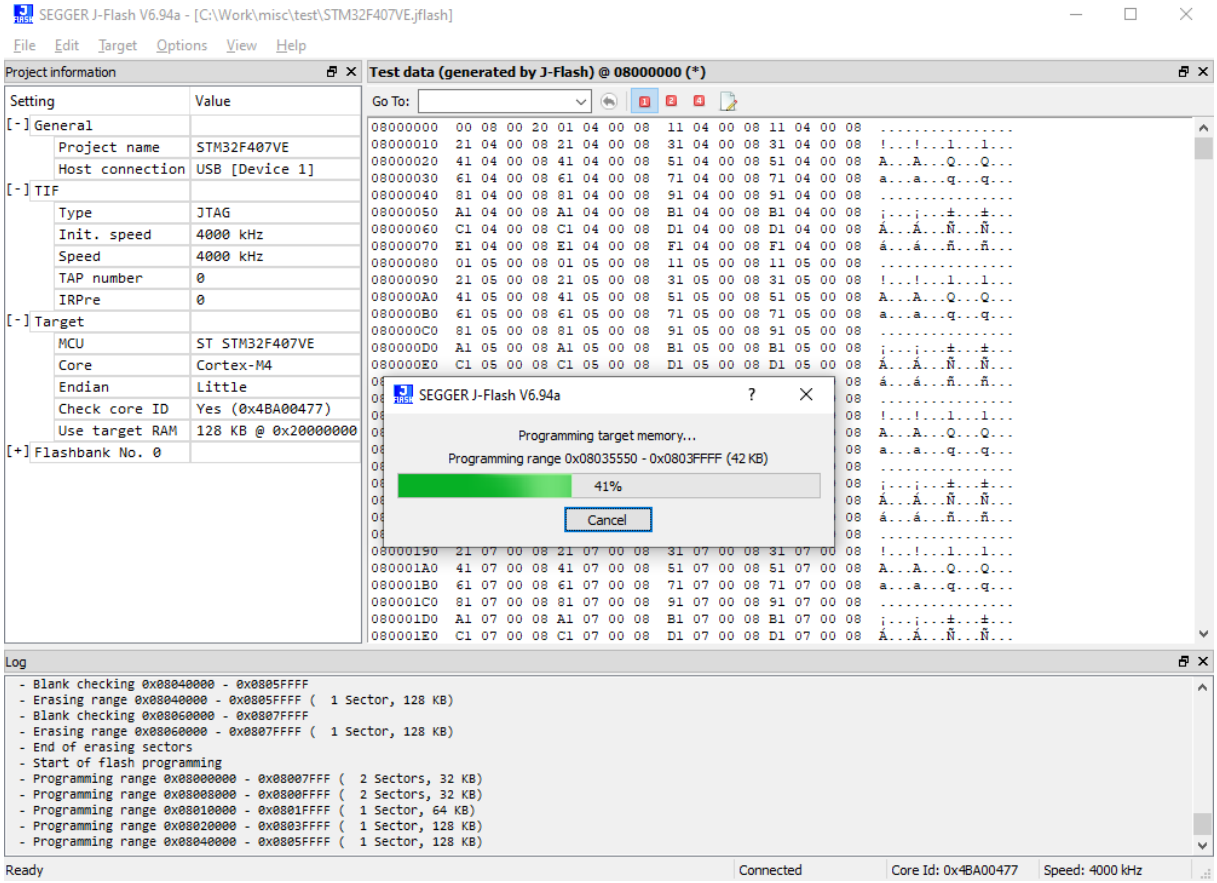


- *J-Link Configurator product page*
- *J-Link Configurator documentation*
- *J-Link Configurator introductory video*

5.3 J-Flash

J-Flash is a multi-platform (Windows, Linux and macOS) application used to program the internal flash of the MCU, as well as external SPI flash connected to the MCU, via a SEGGER J-Link, J-Trace, or Flasher.

J-Flash comes with sample projects that run out-of-the-box for most popular microcontrollers and evaluation boards. J-Flash can be controlled via GUI or via command line, which also makes it possible to use J-Flash for production purposes.

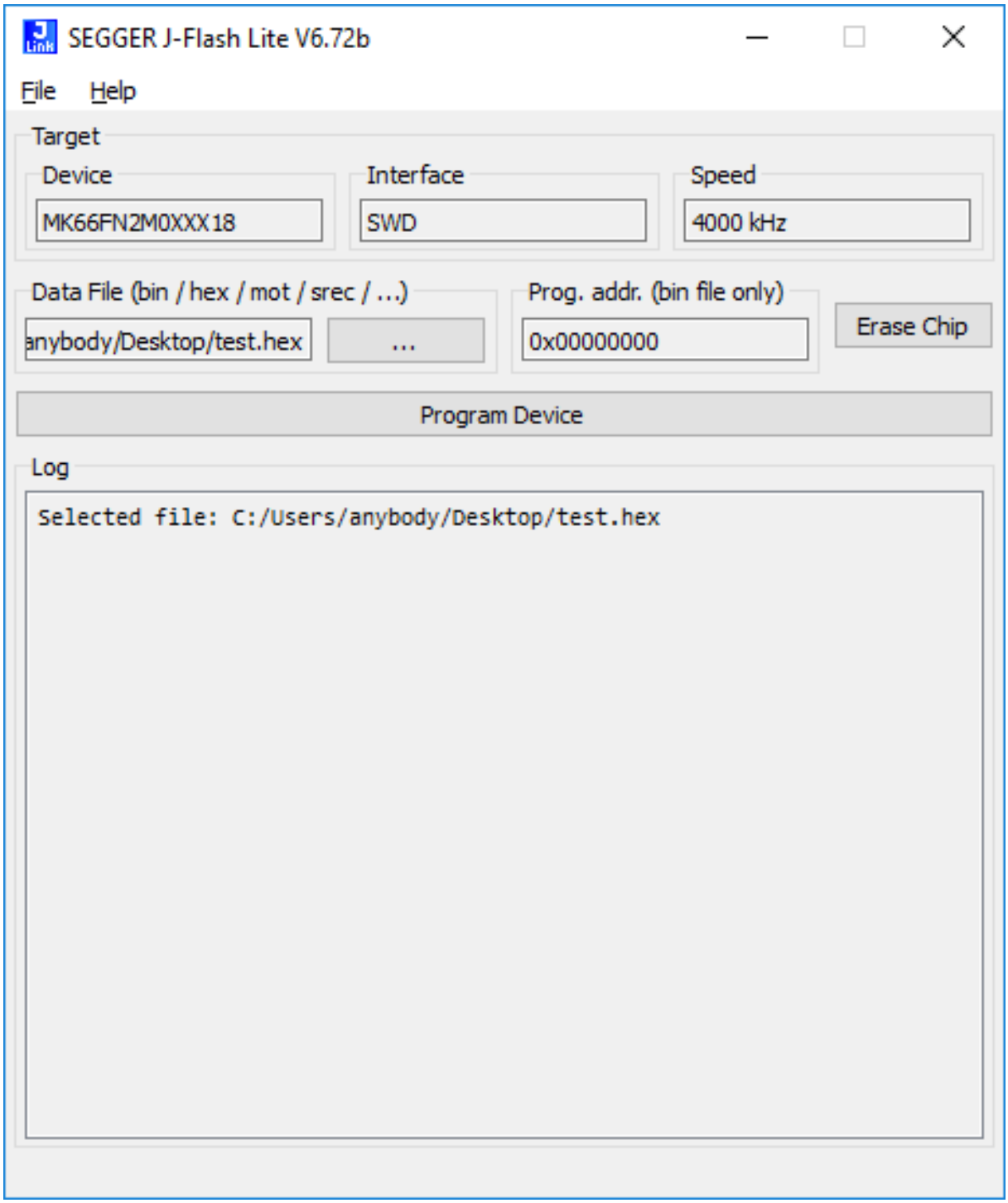


- *J-Flash product page*
- *J-Flash documentation*
- *J-Flash introductory video*

Please note that the use of J-Flash requires a license. This license is included with the purchase of any J-Trace and any J-Link (except for the J-Link Base and J-Link EDU Mini models).

5.4 J-Flash Lite

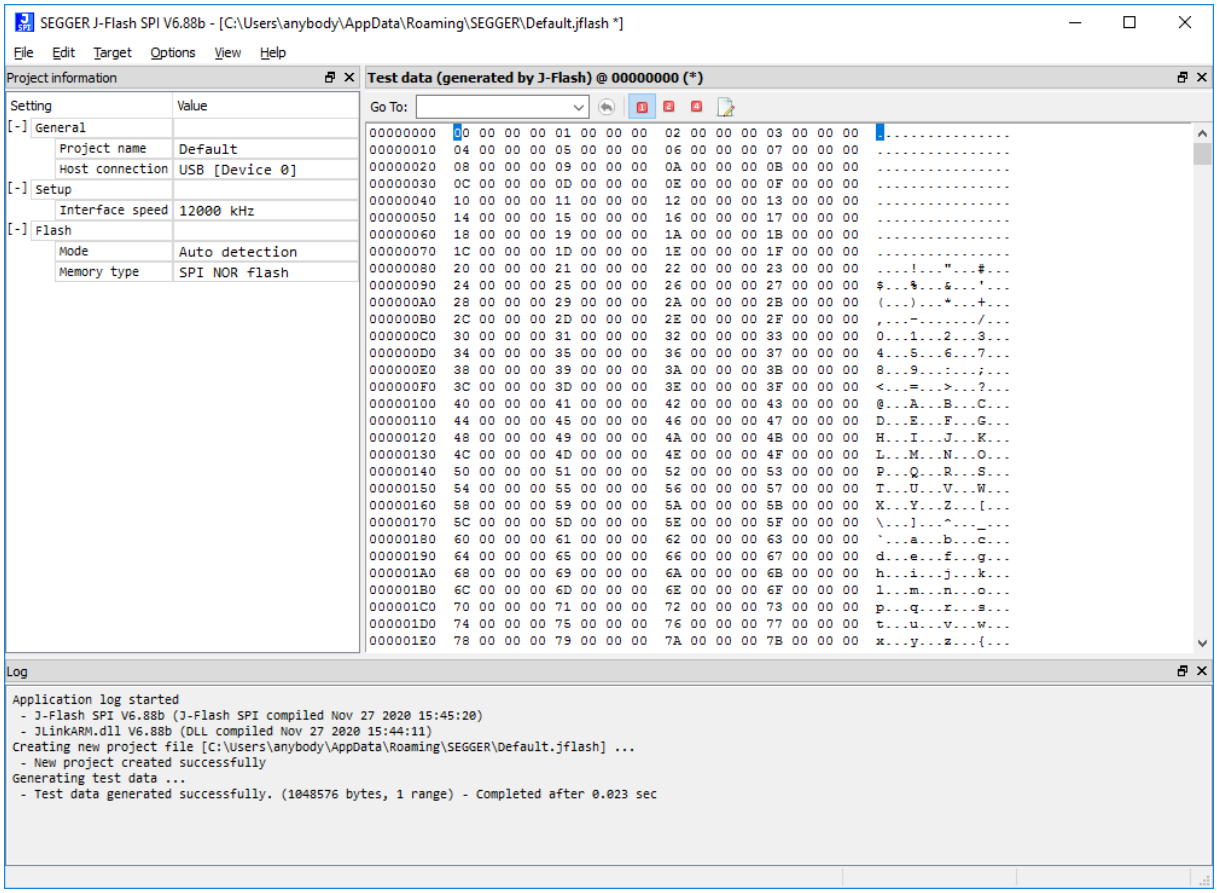
J-Flash Lite is a simple flash programming application to program data images to the flash of a target device. In comparison to J-Flash, J-Flash Lite has a reduced feature set, but does not require a J-Link Plus or higher to operate (it can also be used with J-Link BASE and J-Link EDU Mini).



- *J-Flash Lite documentation*

5.5 J-Flash SPI / J-Flash SPI CL

J-Flash SPI is a flash programming software for Windows, Linux or macOS. It allows direct programming of SPI flashes via a SEGGER J-Link, J-Trace, or Flasher, without going through an MCU (as is the case with J-Flash). J-Flash SPI has an intuitive user interface and makes programming flash devices convenient. J-Flash SPI is able to auto-detect and program all kinds of SPI flashes, even if the MCU they are connected to is not supported by J-Link / J-Trace / Flasher. This is because J-Flash SPI communicates directly with the SPI flash, bypassing all other components of the hardware. Because the SPI flash is programmed in-circuit by J-Flash SPI without any MCU involvement, it programs at the fastest speed possible.



J-Flash SPI CL is a command-line-only version of the J-Flash SPI programming tool. Except for the missing GUI, J-Flash SPI CL is identical to the GUI version. The commands used to configure / control J-Flash SPI CL are exactly the same as for the command line interface of the J-Flash SPI GUI version.

Both J-Flash SPI and J-Flash SPI CL are part of the J-Link Software and Documentation Pack available from segger.com.

- *J-Flash SPI product page*
- *J-Flash SPI documentation*

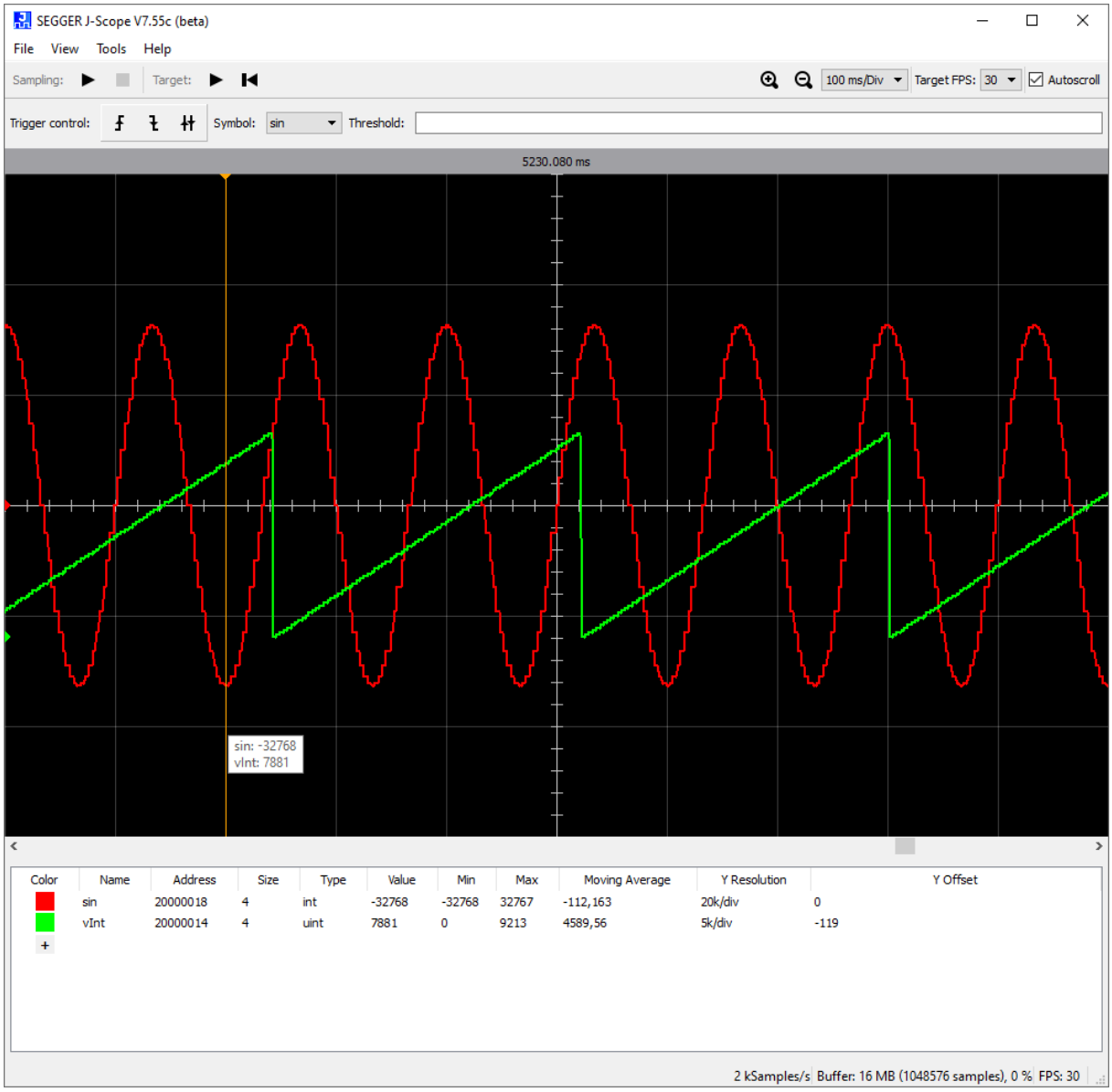
Please note that the use of J-Flash SPI / J-Flash SPI CL requires a license. This license is included with the purchase of any J-Trace and any J-Link (except for the J-Link Base and the J-Link EDU Mini models).

5.6 J-Scope

J-Scope is a software tool to analyze and visualize data on a microcontroller in real-time, while the target is running. J-Scope requires a J-Link / J-Trace as an interface to the target hardware.

Sampling can be done using either SEGGER High-Speed-Sampling (HSS) or SEGGER Real Time Transfer (RTT) technology. Both technologies are available to all MCUs that provide background memory access. SEGGER HSS can be used without any further preparation. SEGGER RTT enables faster sampling speeds, however this requires instrumentation of the target application as described here:

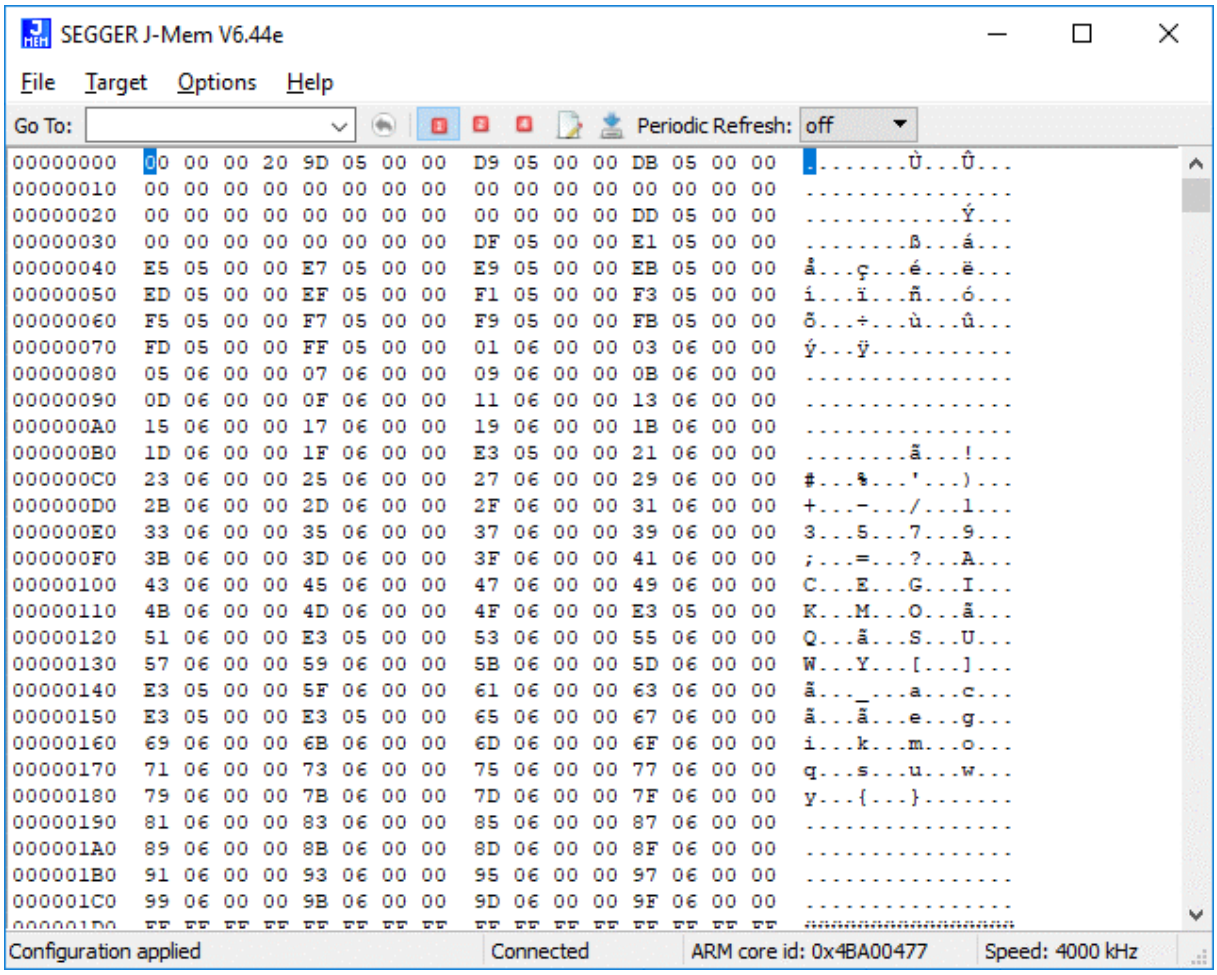
- *Instrumenting an application to use it with J-Scope*



- *J-Scope product page*
- *J-Scope documentation*

5.7 J-Mem

J-Mem is a GUI application for Windows, Linux, and macOS that displays memory contents of microcontrollers. It allows modifications of RAM as well as special function registers while the target is running. Whole regions in RAM can be filled with a desired value. J-Mem can be set up to periodically read the contents to see where data is modified. Memory sections can be downloaded to a binary file for analysis.



- *J-Mem product page*
- *J-Mem documentation*

5.8 J-Run

J-Run is a command line utility for automated tests. It loads an application (elf) file to the device under test, runs it, and captures the application's output.

J-Run supports any ARM and RISC-V device which is supported by J-Link, and elf output from most toolchains. The test application can do standard printf-style debug output to the debugger. J-Run supports handling of RTT and Semihosting. When the test application is finished, it can call the Semihosting exit operation or print the wildcard exit string (by default *STOP*), to exit the J-Run test.

The target output is printed to the terminal. It can be redirected to a file or application for further analysis.

```
C:\Work\JLink\JRun\Test>JRun.exe Target\ST_STM32F407_RTT_JRun_Demo.elf
SEGGER J-Run V1.24 (Compiled Sep 16 2024 10:08:12)
Open application...OK
Connecting to J-Link...OK
Connected to J-Trace PRO V3  compiled Jul  3 2024 16:59:08 (S/N 1223000038).
DLL Version: 7.96j
Set target device to STM32F407IE...OK
Select SWD interface...OK
Set interface speed to 4000 kHz...OK
Reset target...OK
Download 0x08000000 - 0x08000FC9...OK
Set RTT control block at 0x20000000...OK
Start RTT...OK
Start target application (SP 0x20020000, PC 0x08000E9F)... OK
Reading output from target until exit command.
================================================

SEGGER J-Run demo.

Took 8395654 cycles

Target Application exit.
```

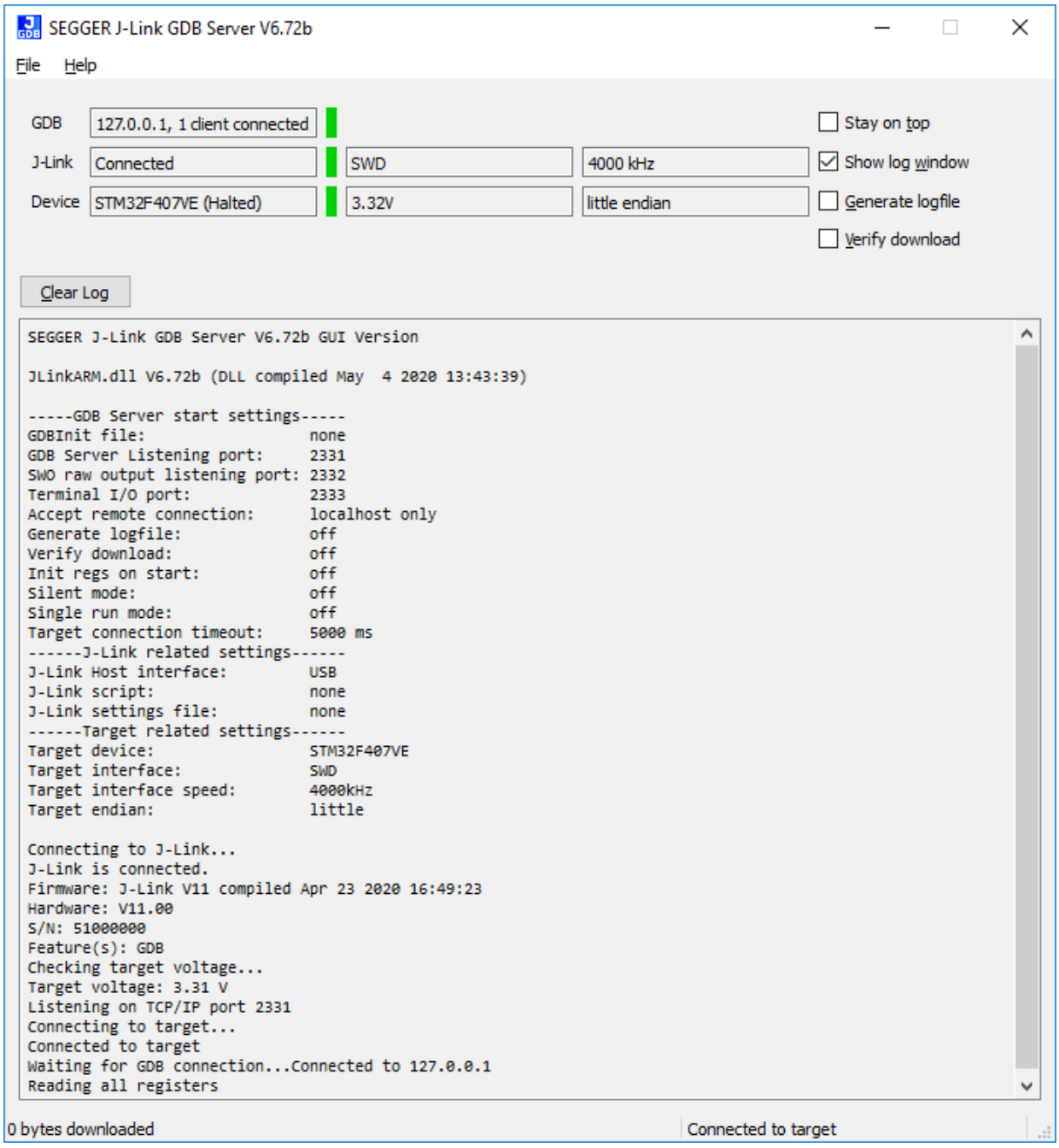
- *J-Run product page*
- *J-Run documentation*
- *J-Run blog article*

5.9 J-Link GDB Server

J-Link GDB Server is a remote server for GDB, making it possible for GDB to connect to and communicate with the target device via J-Link / J-Trace. It is part of the J-Link Software and Documentation Pack, which is available for download from segger.com. J-Link GDB Server and GDB communicate via a TCP/IP connection, using the standard GDB remote protocol. The J-Link GDB Server receives the GDB commands, handles the J-Link / J-Trace communication and replies with the answer to GDB.

With J-Link GDB Server, debugging in ROM and Flash of the target device is possible, and the Unlimited Flash Breakpoints feature is available. The J-Link GDB Server also supports some functionality that is not directly implemented in GDB, but is accessible via monitor commands, sent directly with GDB.

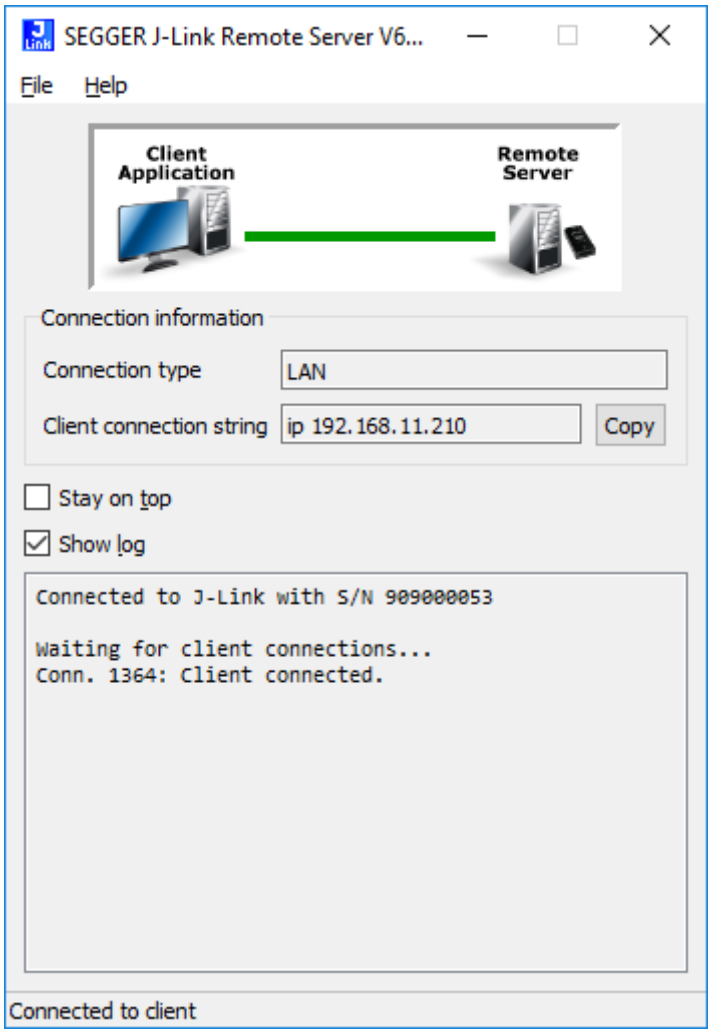
Both the GUI and the CL version of J-Link GDB Server are available for Windows, Linux and macOS.



- *J-Link GDB Server product page*
- *J-Link GDB Server documentation*

5.10 J-Link Remote Server

J-Link Remote Server is a utility (available as command line or GUI application) that makes a J-Link / J-Trace accessible via IP, be it in the local network or from anywhere on the world. The J-Link / J-Trace itself does not need to provide an Ethernet interface, USB is sufficient.

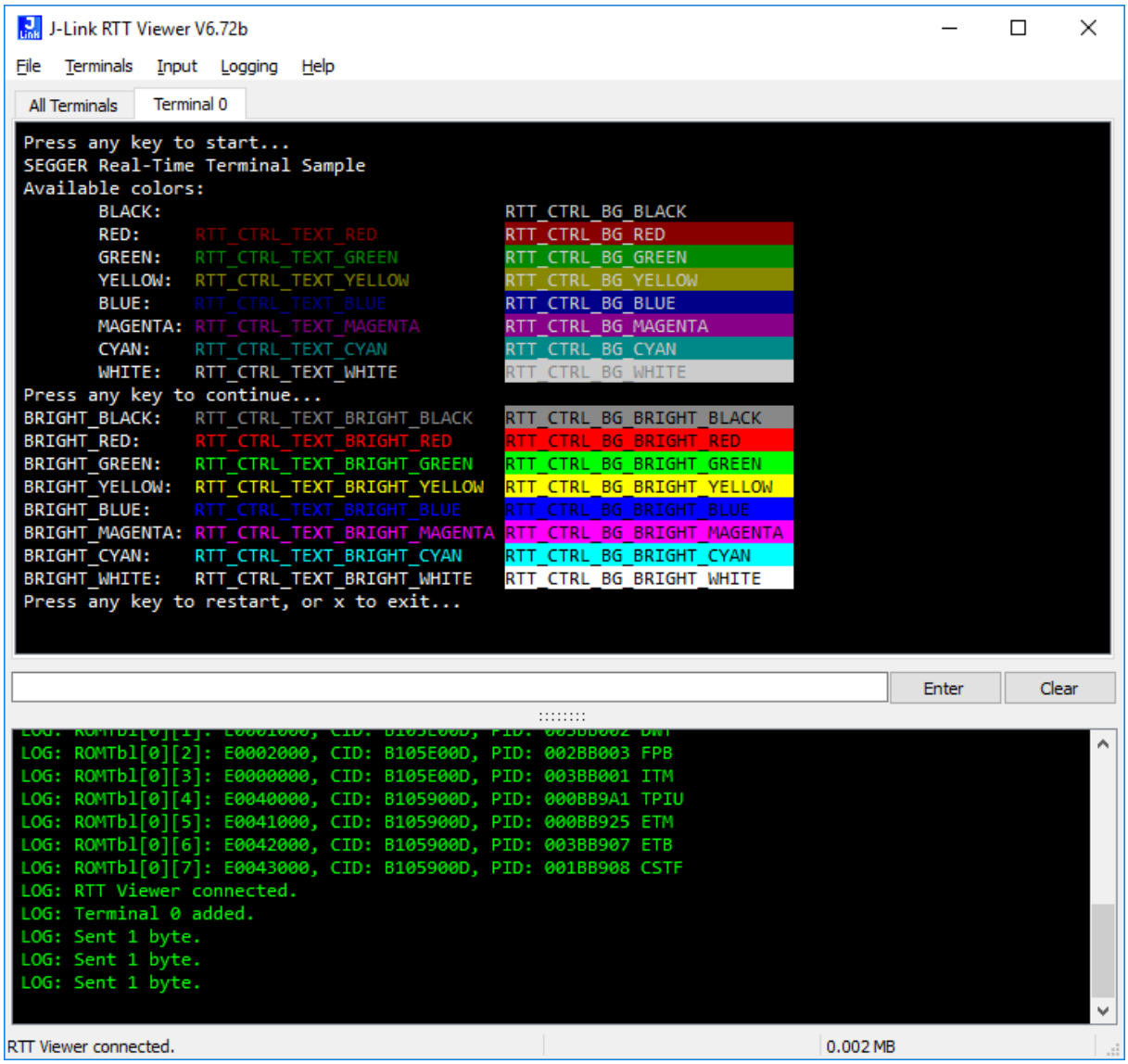


- *J-Link Remote Server product page*
- *J-Link Remote Server documentation*

5.11 J-Link RTT Viewer

J-Link RTT Viewer is a GUI application available for Windows, MacOS and Linux and can be downloaded as part of the J-Link Software and Documentation Pack. It enables you to use all features of SEGGER Real-Time Transfer (RTT) in one application. It supports the following features:

- Displaying terminal output of RTT Channel 0
- Up to 16 virtual Terminals on RTT Channel 0
- Sending text input to RTT Channel 0
- Interpreting text control codes for colored text and controlling the Terminal
- Logging terminal data into a file
- Logging data on RTT Channel 1

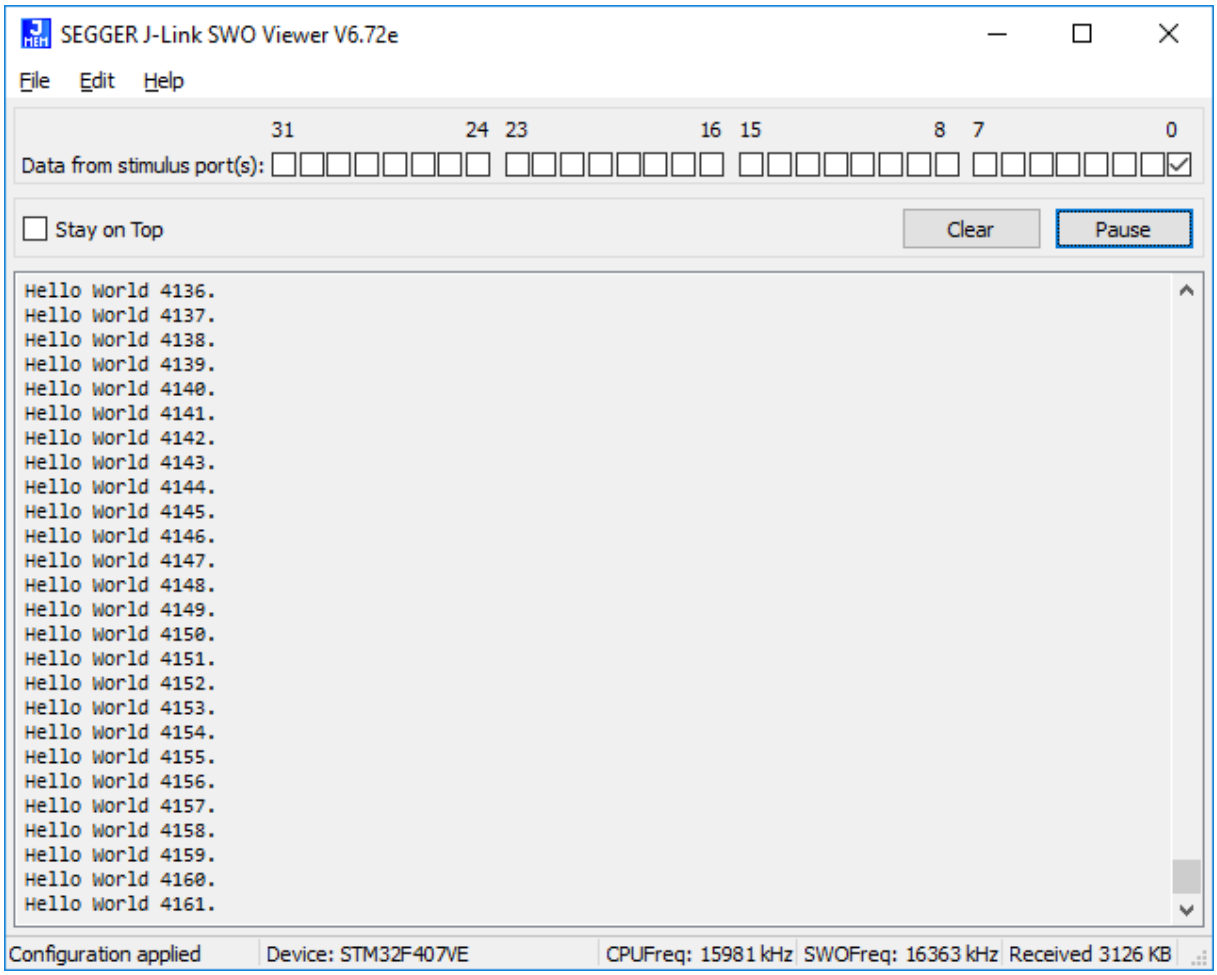


- *J-Link RTT Viewer product page*
- *J-Link RTT Viewer documentation*

5.12 J-Link SWO Viewer / J-Link SWO Viewer CL

J-Link SWO Viewer displays the terminal output of the target using the SWO pin. It is available as GUI and command line version as part of the J-Link Software and Documentation Pack, which is available for download from segger.com. The J-Link SWO Viewer can be used in parallel with a debugger or stand-alone. This is especially useful when using debuggers which do not come with built-in support for SWO, such as most GDB / GDB +Eclipse based debug environments.

J-Link SWO Viewer CL is the command line-only version of J-Link SWO Viewer. All commands available for J-Link SWO Viewer (GUI) can be used with J-Link SWO Viewer CL.

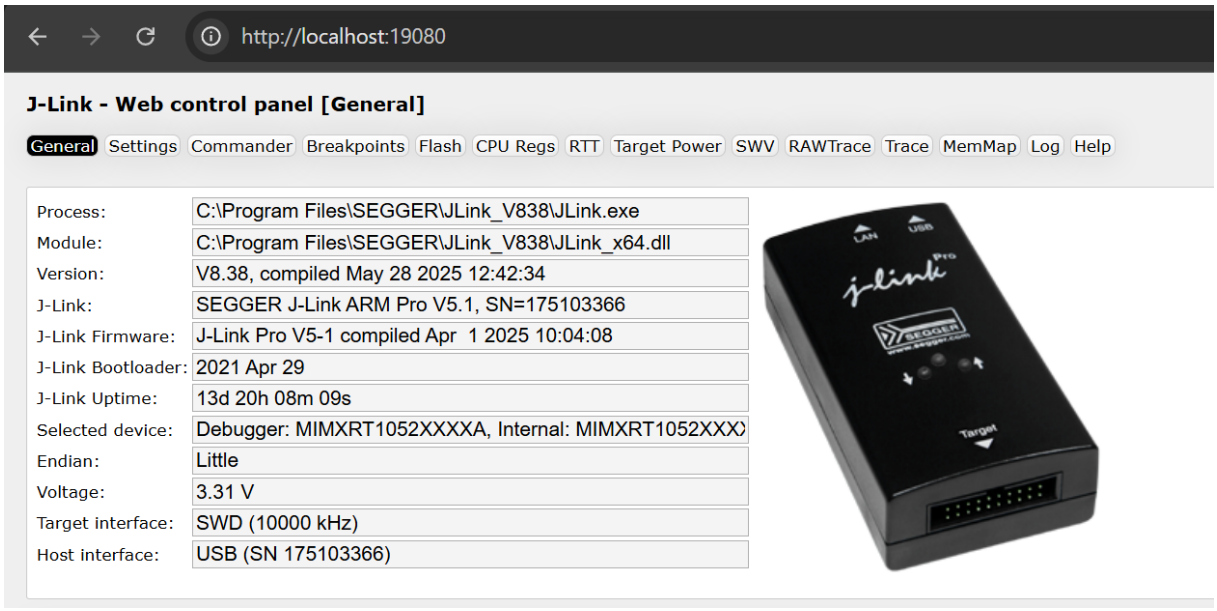


- *J-Link SWO Viewer product page*
- *J-Link SWO Viewer documentation*

5.13 J-Link Web Control Panel

The J-Link Web Control Panel allows you to monitor the J-Link / J-Trace status and the target status information in real-time. It also allows you to configure the use of some J-Link / J-Trace features such as flash download, flash breakpoints, and instruction set simulation. You can access the J-Link web control panel via the J-Link tray icon in the tray icon list (as soon as a debug session is running) or via `http://localhost:19080`.

Note that the port number increases for each new instance of the J-Link DLL you open, i.e. the first instance will use port 19080, the second instance will use 19081, and so on. For this reason, it is highly recommended to launch the J-Link web control panel using the tray icon.

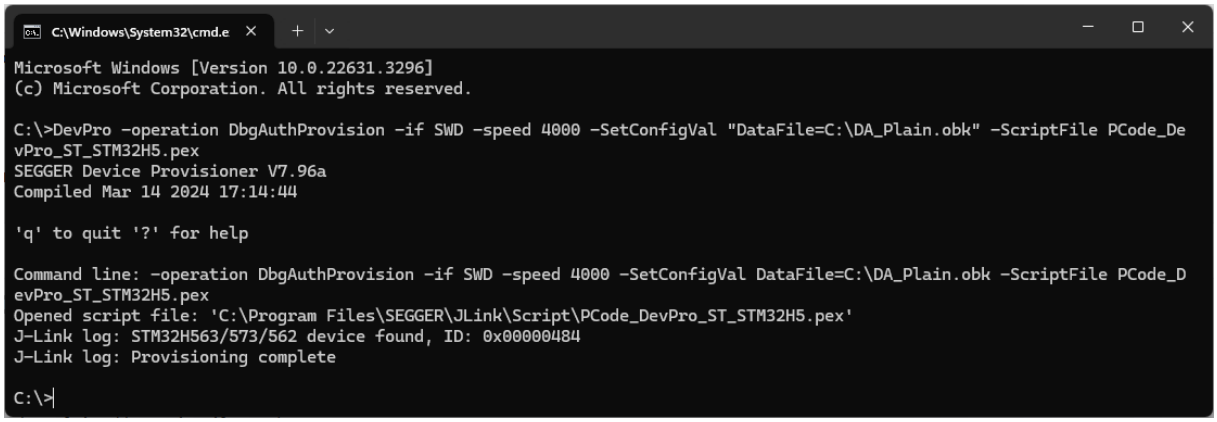


- *J-Link web control panel product page*
- *J-Link web control panel documentation*

5.14 Device Provisioner

Device Provisioner is a command line tool for J-Link / J-Trace (and Flasher), ensuring devices are properly set up and configured for use. The provisioning process includes tasks such as initializing hardware, installing software, configuring settings, and sometimes associating the device with a specific user or network

Created to seamlessly integrate into automation environments, Device Provisioner executes commands from scripts written in C, which can be provided by SEGGER, silicon vendors, or created by users. These scripts can be executed on J-Link and J-Trace probes while connected to a host PC. To protect intellectual property, script files can be distributed in source code or pre-compiled form.



- *Device Provisioner product page*
- *Device Provisioner documentation*

Chapter 6

Summary

We hope this guide provided you with a solid understanding of how to get started and how to work with the J-Link / J-Trace probes from SEGGER. Below are the most commonly accessed pages for further reading.

- *Complete J-Link / J-Trace online User Guide*
- *How to register a SEGGER probe*
- *J-Link SW and Documentation Package*
- *Feature comparison between J-Link and J-Trace*
- *Detailed J-Link model feature comparison*
- *Detailed J-Trace model feature comparison*
- *List of cores supported by J-Link / J-Trace*
- *List of devices supported by J-Link*
- *Embedded Studio IDE product page*
- *Ozone debugger product page*
- *SystemView visualization and analysis tool product page*