The Embedded Experts

SEGGER

- Debug Probes
- Software Tools
- RTOS
- File System
- Compression
- Connectivity
- Security & IoT
- User Interface
- Production

J-Link

J-Link Debug Probes

SEGGER J-Links are the most widely used line of debug probes available today. They have proven their worth for more than 15 years with over 1 Mio units in the field. This popularity stems from the unparalleled performance, extensive feature set, large number of supported CPUs and compatibility with all popular development environments.

Debug smarter and faster with J-Link!

With up to 3 MByte per second download speed to RAM and record-breaking flashloaders, and with the ability to set an unlimited number of breakpoints in flash memory of MCUs, J-Link debug probes are undoubtedly the best choice to optimize your debugging and flash programming experience.

Extensive Device & IDE Support

J-Link debug probes support all popular microcontrollers. For further details please visit our website. All major IDEs include support for the J-Link. The list includes, among others, SEGGER Embedded Studio, IAR, Keil MDK and all GDB-based IDEs.

Direct Download into Flash Memory

J-Link can program the internal flash of almost all popular microcontrollers as well as external CFI compliant flashes. The J-Link flash loaders enable an IDE to download directly into flash memory as if it was RAM. This makes flash download available to any IDE that supports J-Link, without the IDE explicitly supporting any flash download features.

Intelligence in the Firmware

In contrast to other debug probes, J-Link has intelligence for different CPU cores in its firmware. For most debug probes, all debug command sequences are generated on the PC and the debug probe is handled as a "dumb probe".
This does not apply for J-Link: It makes use of intelligence in the firmware which speeds up things massively (easily factor 10 in most cases) and also adds support for corner cases that cannot be supported without such intelligence.
A sample for such a scenario is: Accessing a slowly running CPU at high target interface speeds. J-Links are the most robust probe in these situations.

Software Development Kit (SDK)

For customers who want to build their own applications using J-Link, and for IDE vendors who implement J-Link support for their IDE, SEGGER offers a J-Link SDK which comes with the J-Link DLL, the API documentation and implementation samples. The SDK is available for Windows and Linux.

Cross-Platform Support

The J-Link software package is designed to run on all major platforms: Windows, Linux and macOS.

Supports Single Wire Viewer / Single Wire Output

J-Link fully supports Arm's SWV/SWO feature which is available for most devices supporting the SWD interface. SWO is a single pin output from the core which can be used to transfer terminal data (printf) and also real-time trace data. The latter enables monitoring of variable read and write accesses in compatible processors.

Real Time Transfer

Real Time Transfer (RTT) is SEGGER's technology for interactive user I/O in embedded applications. It combines the advantages of SWO and semihosting at very high performance, with data transfer speed reaching up to 2 MByte per second while retaining the real-time behaviour of the target system.

Unlimited Flash Breakpoints

The Unlimited Flash Breakpoints feature allows the user to set an unlimited number of breakpoints when debugging in flash memory. Without this feature, the number of breakpoints which can be set in flash is limited to the number of hardware breakpoints supported by the debug unit of the CPU. Unlimited Flash Breakpoints works in internal and external flash, even with memory-mapped flashes.

Monitor Mode Debugging

Monitor Mode Debugging enables an embedded system based on a Cortex-M3, M4 or M7 core to maintain essential functionality while being debugged. This offers the possibility to maintain real-time, user-defined functions in selected interrupt services, such as motor control, data acquisition or any application that requires some kind of continuous operation.

Remote Server Debugging in Tunnel Mode

The J-Link Remote Server effortlessly debugs target hardware and applications in remote locations over TCP/IP as if the target was on the developer's desk. Taking this concept to the next level, SEGGER offers a tunnel mode for remote debugging anywhere in the world.
Tunnel mode initiates the connection by sending the serial number or name of the J-Link to the tunnel server. The J-Link DLL is then capable of creating a tunnel connection via the server. For added protection, the connection can be secured with a password.
Support engineers can debug unwieldy hardware at the customer's site without having to travel there, simply by sending a J-Link. Distributed development teams can share early prototypes even in remote locations.

Features

- Supports Arm® Cortex®-M / R / A cores, including the 64-bit cores (ARMv8-A) and Arm® 7 / 9 / 11, Microchip PIC32, Renesas RX, RISC-V and Silicon Labs 8051
- Maximum JTAG speed 15 MHz, J-Link ULTRA+ / PRO: 50 MHz
- Download speed up to 1.5 MB/s (J-Link® / J-Link® PLUS), 3 MB/s (J-Link® ULTRA+ / PRO)
- Power profiling (J-Link® ULTRA+ / PRO)
- Very fast flash loader
- Supported by all popular debuggers
- Support for different debug interfaces: JTAG / SWD / FINE / SPD / ICSP
- Serial Wire Viewer (SWV) with up to 7.5 / 25 MHz supported
- Host interface: USB, Ethernet
- Power over USB
- Support for adaptive clocking
- Multi-core debugging supported
- Wide target voltage range: 1.2V - 5.0V tolerant
- J-Link Remote Server included, which allows using J-Link via TCP/IP networks
- SDK available

J-Trace PRO

J-Trace PRO

J-Trace PRO for Cortex-M and J-Trace PRO Cortex set the standard for trace probes. Enabling continuous streaming trace using Ethernet or USB, the J-Trace PRO models completely remove “trace buffer size” anxiety by eliminating the need for fixed- size trace buffers in the probe.

Streaming Trace: Extended Debugging and Verification

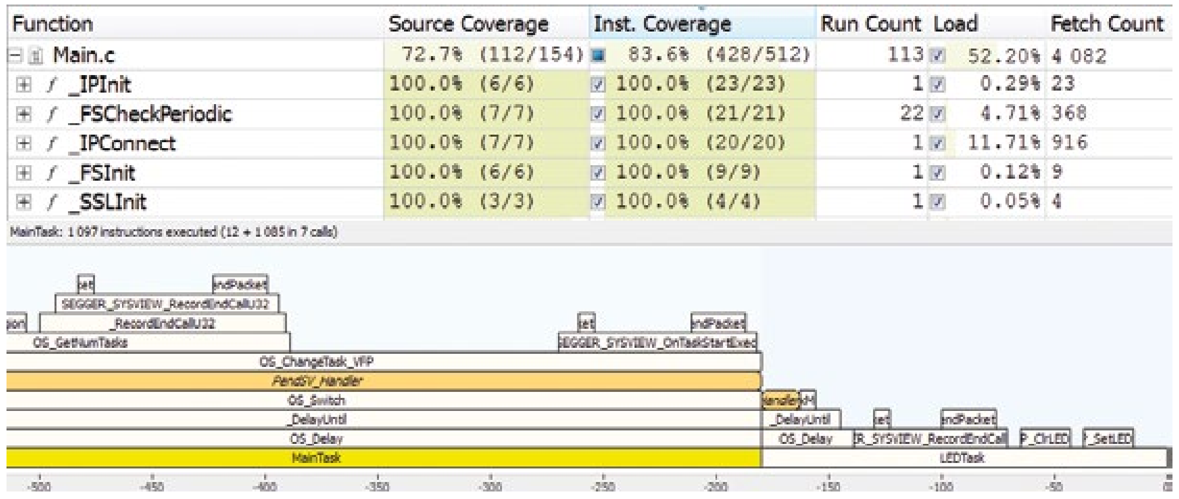
J-Trace PRO captures complete traces over long periods which helps capture infrequent, hard-to-reproduce bugs. This is particularly helpful when the program flow ‘runs off the rails’ and stops in a fault state.

Features

- Live code profiling
- Live code coverage
- Endless streaming trace
- Real-time analysis
- Endless trace buffer
- Gigabit Ethernet
- SuperSpeed USB 3.0
- Supports Cortex-A/R/M
- Cross-platform (Linux, Mac, Windows)
- Best trace experience with free Ozone debugger
- Software Devolpment Kit (SDK) available
- All Arm Cortex features of J-Link PRO included

Maintain Code Quality

J-Trace PRO supports extended trace features, such as code coverage and code profiling. Code coverage shows which parts of the application code have been executed. Code or execution profiling shows which instructions have been executed and how often – so hotspots can be addressed and optimization opportunities can be identified.



Function	Source Coverage	Inst. Coverage	Run Count	Load	Fetch Count
Main.c	72.7% (112/154)	83.6% (428/512)	113	52.20%	4 082
_IPInit	100.0% (6/6)	100.0% (23/23)	1	0.29%	23
_FSCheckPeriodic	100.0% (7/7)	100.0% (21/21)	22	4.71%	368
_IPConnect	100.0% (7/7)	100.0% (20/20)	1	11.71%	916
_FSInit	100.0% (6/6)	100.0% (9/9)	1	0.12%	9
_SSLInit	100.0% (3/3)	100.0% (4/4)	1	0.05%	4

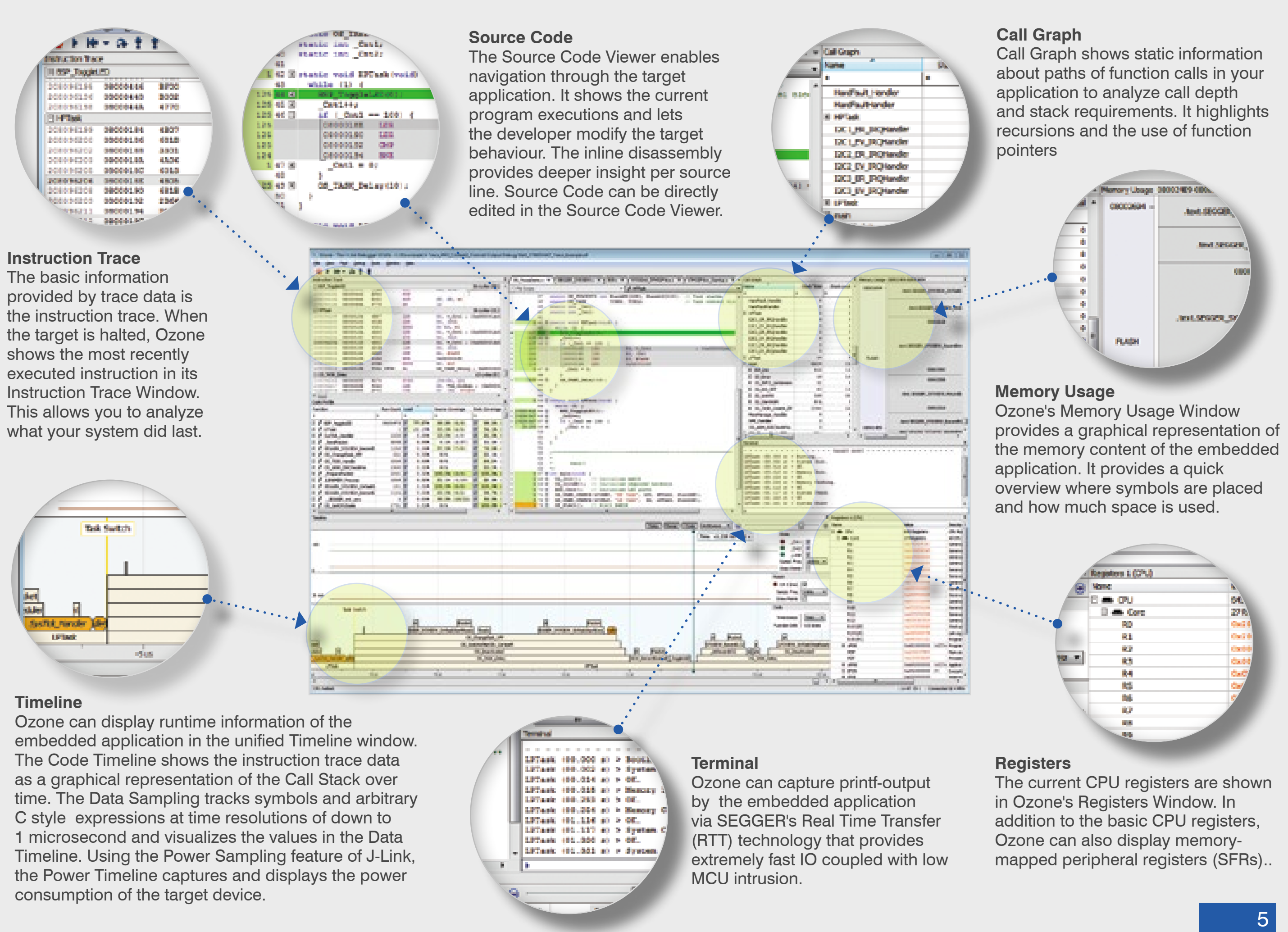
MainTask: 1 097 instructions executed (12 + 1 085 in 7 calls)

Debugging with Ozone

Ozone is a full-featured graphical debugger for embedded applications. With Ozone it is possible to debug any embedded application at C/C++ source and assembly level. Ozone can load applications built with any toolchain/IDE or debug the target's resident application without any source. Ozone includes extensive debug information windows and makes use of the best performance of J-Link debug probes. The user interface is designed to be used intuitively and is fully configurable. All windows can be moved, re-sized and closed. Ozone is available for Windows, macOS and Linux.

Features

- Blazing fast debug and step performance
- Precision stepping at source and assembly level
- Thread-aware debugging (customizable)
- C/C++ code source level debugging
- Assembly instruction debugging
- Direct use of J-Link/J-Trace built-in features
- Extensive debug & processor state windows
- Scriptable project files for test automation
- Live Code Coverage and code profiling
- Power Profiling



Source Code
The Source Code Viewer enables navigation through the target application. It shows the current program executions and lets the developer modify the target behaviour. The inline disassembly provides deeper insight per source line. Source Code can be directly edited in the Source Code Viewer.

Call Graph
Call Graph shows static information about paths of function calls in your application to analyze call depth and stack requirements. It highlights recursions and the use of function pointers

Instruction Trace
The basic information provided by trace data is the instruction trace. When the target is halted, Ozone shows the most recently executed instruction in its Instruction Trace Window. This allows you to analyze what your system did last.

Memory Usage
Ozone's Memory Usage Window provides a graphical representation of the memory content of the embedded application. It provides a quick overview where symbols are placed and how much space is used.

Timeline
Ozone can display runtime information of the embedded application in the unified Timeline window. The Code Timeline shows the instruction trace data as a graphical representation of the Call Stack over time. The Data Sampling tracks symbols and arbitrary C style expressions at time resolutions of down to 1 microsecond and visualizes the values in the Data Timeline. Using the Power Sampling feature of J-Link, the Power Timeline captures and displays the power consumption of the target device.

Terminal
Ozone can capture printf-output by the embedded application via SEGGER's Real Time Transfer (RTT) technology that provides extremely fast IO coupled with low MCU intrusion.

Registers
The current CPU registers are shown in Ozone's Registers Window. In addition to the basic CPU registers, Ozone can also display memory-mapped peripheral registers (SFRs)..

Embedded Studio

Developing with SEGGER Embedded Studio

SEGGER Embedded Studio is a powerful cross-platform C/C++ IDE (Integrated Development Environment) for Arm and RISC-V microcontrollers. It is specifically designed to provide everything needed for professional embedded development: an all-in-one solution providing continuous workflow.

Cross-Platform Support

SEGGER Embedded Studio is available for Windows, macOS and Linux. Its look and feel is similar on all platforms to provide the best experience regardless of the operating system.

Target Support

SEGGER Embedded Studio can be used with Arm®7, Arm®9, the complete Arm® Cortex® microcontroller series and RISC-V microcontrollers.

SEGGER Embedded Studio offers a reduced cost license for those working only with Cortex®-M, and alternatively a license which covers the full range of supported Arm® microcontrollers. There is a separate license for RISC-V microcontrollers.

Powerful Project Manager

An advanced Project Manager is included with SEGGER Embedded Studio which enables simple management of extremely large projects and multi-project solutions.

SEGGER Embedded Studio's Package Manager provides access to Support Packages for various MCUs which can be installed on demand and updated when a new version is available. The Support Packages make starting a new project for new target hardware as simple as clicking a button.

SEGGER Toolchain

Embedded Studio comes with prebuilt LLVM and GCC toolchains. Additionally it includes the SEGGER Compiler, Linker, and Runtime Library, which are designed for use with embedded systems.

SEGGER Linker

The SEGGER Linker makes linking simpler, optimizes application size, and solves common linking problems that usually arise in embedded development.

The SEGGER Linker can optimize size and performance of your application by linking only what is required. It can sort input fragments to optimize memory usage and compress initialized data and code.

Feature-packed Debugger

SEGGER Embedded Studio integrates a feature-packed graphical debugger with enhanced J-Link integration for direct debugging on your target hardware. All of the industry leading J-Link features have been tightly integrated into SEGGER Embedded Studio.

The debugger includes various debugging windows, which make it possible to inspect and manipulate advanced information concerning the running application and its execution, including mixed-mode disassembly, source code, an I/O Terminal for semihosting, SWO, SEGGER's Real-Time Transfer, and a scriptable Threads Window to be used with any (real-time) operating system.

First-Class Editor

The first-class Source Code Editor does not only support user-defined syntax highlighting, automatic code indention and matching bracket highlighting, it also provides a code completion feature for symbols, functions and keywords of your application, as well as configurable code and comment templates to easily match your coding and documentation standards. The Editor is highly integrated into the Project Manager for efficient and advanced search and replace functionality in your files, projects and solutions. The behavior of all features is fully user-configurable.

Features

- Windows, macOS and Linux support
- Powerful Project Manager
- Advanced first-class Editor
- Ultra-fast SEGGER Linker can achieve significant memory savings compared to the GNU Linker
- Package-based Project Generator for all common microcontrollers
- Pre-built C/C++ Compiler, GCC and LLVM included for an immediate start
- Royality free ANSI / ISO C compliant C library for embedded systems
- Embedded Studio offers support for Arm and RISC-V based microcontroller
- Supported debug probes: J-Link/J-Trace, DAP-Link (CMSIS-DAP) as well as 3rd party debug probes via GDB protocol

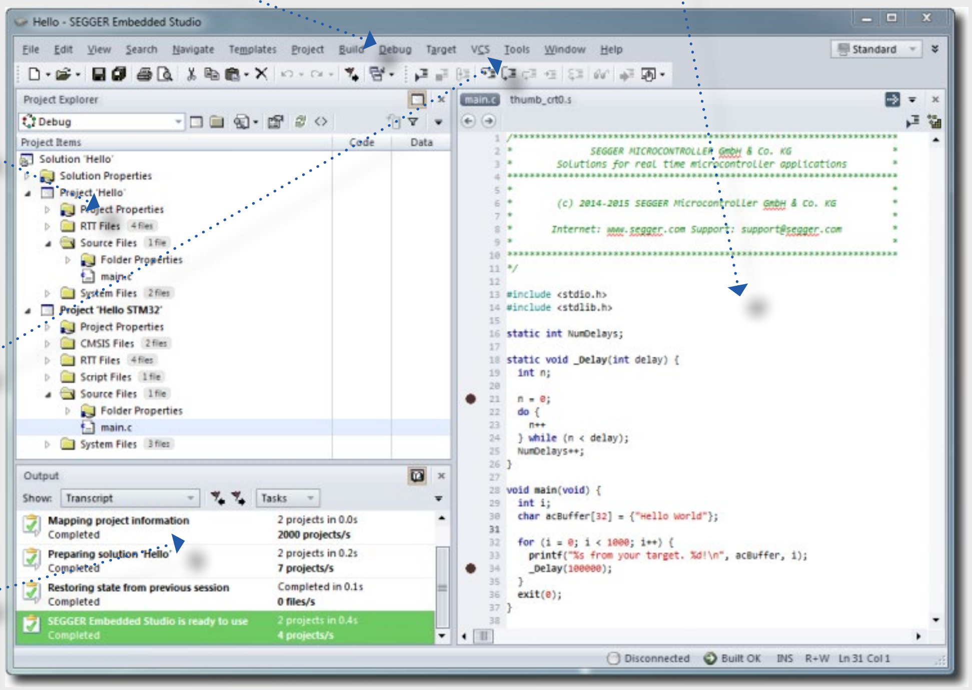
Feature packed debugger
A graphical debugger with J-Link integration and mixed-mode disassembly. The debugger has an I/O Terminal for semihosting, SWO and SEGGER's Real-Time Terminal. The Thread Window provides scriptable RTOS awareness, examples for embOS and Amazon FreeRTOS included.

First-class editor
The code editor supports the language of the source file it is editing, showing code with syntax highlighting, automatic code indention, and matching bracket highlighting. Embedded Studio provides suggestions for code completion based on your application, while you type.

Powerful project manager Capable of managing huge and multi-project solutions easily. Provides access to Support Packages for various Arm MCUs.

Code analysis tools
Embedded Studio has numerous analysis tools to assist the developer. For example: memory usage, code outline, static code analysis, stack usage, symbol browser and source outline. External tools, such as SystemView and MISRA-checker software can be configured.

Toolchains included
Clang/LLVM, GCC and SEGGER compiler. Prebuilt for Arm microcontrollers. Highly-optimized, royalty-free ANSI / ISO compliant standard C/C++ library.



```
/*********************************************************************
*                   SEGGER MICROCONTROLLER GmbH & Co. KG             *
*       Solutions for real time microcontroller applications         *
**********************************************************************
*                                                                    *
*       (c) 2014-2015 SEGGER Microcontroller GmbH & Co. KG           *
*                                                                    *
*       Internet: www.segger.com Support: support@segger.com         *
*                                                                    *
**********************************************************************
*/

#include <stdio.h>
#include <stdlib.h>

static int NumDelays;

static void _Delay(int delay) {
  int n;

  n = 0;
  do {
    n++
  } while (n < delay);
  NumDelays++;
}

void main(void) {
  int i;
  char acBuffer[32] = {"Hello World"};

  for (i = 0; i < 1000; i++) {
    printf("%s from your target. %d!\n", acBuffer, i);
    _Delay(100000);
  }
  exit(0);
}
```

```
Mapping project information                 2 projects in 0.0s
Completed                                   2000 projects/s
Preparing solution 'Hello'                  2 projects in 0.2s
Completed                                   7 projects/s
Restoring state from previous session       Completed in 0.1s
Completed                                   0 files/s
SEGGER Embedded Studio is ready to use      2 projects in 0.4s
Completed                                   4 projects/s
```

Embedded Studio PRO

Start developing right away with Embedded Studio PRO

Embedded Studio PRO is a highly advanced platform combining SEGGER's user-friendly Embedded Studio development environment, along with a selection of our key embedded software components. It is offered as a pure software package or as a bundle with the industry-leading J-Link PLUS debug probe, plus an emPower reference board.

Embedded Studio PRO is the one-stop shop solution that includes everything needed by an embedded engineer: a cutting-edge IDE, a robust software framework including an RTOS, almost all communication and security stacks and a high quality graphical user interface software.

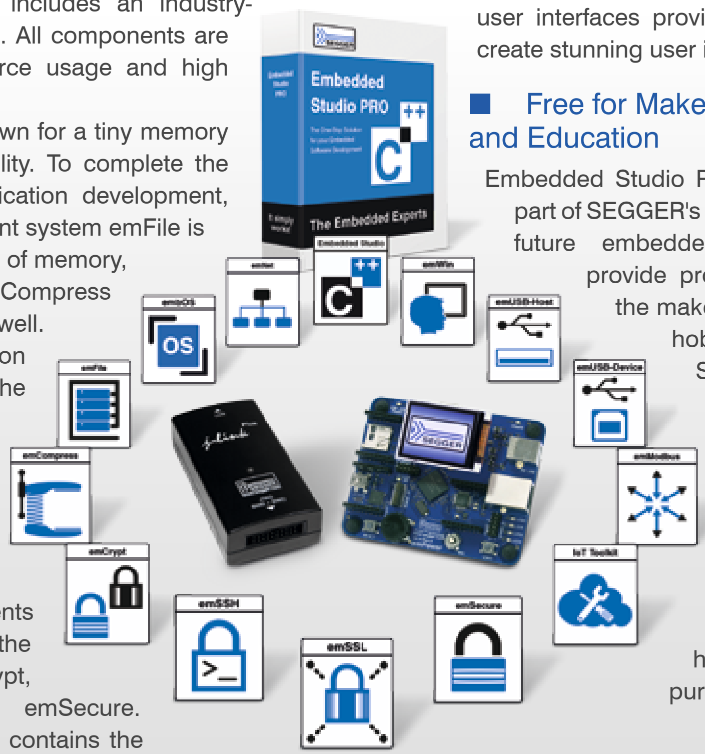
Features

- All-in-one development solution
- Runs out-of-the-box
- Includes more than 50 ready-to-run projects
- Start application development immediately
- State-of-the-art development environment
- Full featured J-Link PLUS Debug probe
- All components developed and supported by SEGGER
- High-performance software with very small footprint
- Royalty-free

Proven and robust embedded software

Embedded Studio PRO includes an industry-proven software package. All components are optimized for low resource usage and high performance.

The RTOS embOS is known for a tiny memory footprint and high reliability. To complete the set of software for application development, the robust file management system emFile is included. For efficient use of memory, data compression with emCompress is part of the package as well. To serve all communication requirements the connectivity solutions emNet and emUSB (device and host) and emModbus are also part of Embedded Studio PRO. To fulfill the security requirements of modern IoT devices, the package includes emCrypt, emSSH, emSSL and emSecure. Last but not least, it also contains the graphics package emWin for high quality user interfaces provides the means to create stunning user interfaces.

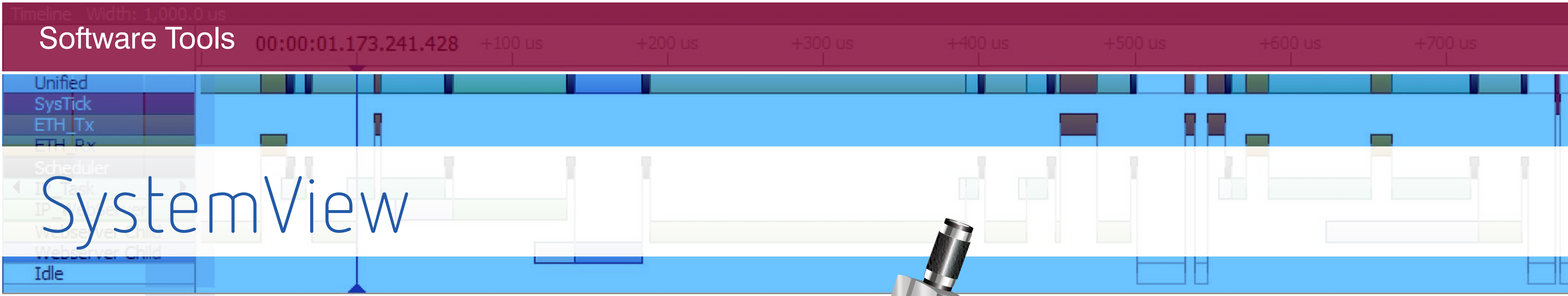


Free for Makers, Hobbyists and Education

Embedded Studio PRO as a whole is part of SEGGER's initiative to educate future embedded engineers. To provide professional tools to the maker, educational and hobby communities, SEGGER created license terms, that enable these communities to use almost all SEGGER tools and software for educational, hobbyist and maker purposes for free.

Components of Embedded Studio PRO package

Component	Description
Integrated Development Environment	
Embedded Studio	Powerful cross platform C/C++ IDE for Arm Cortex®-M microcontrollers, an all-in-one solution aiming at efficiency, stability and a continuous workflow
Software	
embOS	Priority-controlled high-performance real time operating system with zero latency
emFile PRO	Robust file system for embedded applications, high performance library optimized for minimum memory consumption
emCompress	Compression system geared towards embedded systems. Enables even compression on resource-constrained microcontroller based systems
emCrypt	Encryption suite
emSSL	Transport layer security protocol stack for protecting communication
emSSH	Secure shell enabling secure remote administration access and data transfer
emSecure	Authentication software to sign and authorize software and hardware
IoT Toolkit	Toolkit for communication with Cloud based services supporting REST API, JSON
emNet PRO	CPU-independent TCP/IP stack, high-performance library optimized for speed, versatility and a small memory footprint
emUSB-Device PRO	High performance USB device stack specifically designed for embedded systems
emUSB-Host PRO	USB host software stack, implements full USB host functionality, including external hub support
emModbus	Industrial communication stack
emWin PRO	Efficient, processor- and LCD controller-independent GUI for any application that operates with a graphical LCD
Hardware	
J-Link PLUS	Debug probe with USB interface, including J-Flash software and advanced features such as unlimited flash breakpoints, SystemView analysis tool and Monitor Mode
emPower	Reference board designed to enhance software evaluation, prototyping and proof of concept



SystemView

SystemView — Real-Time Analysis and Visualization

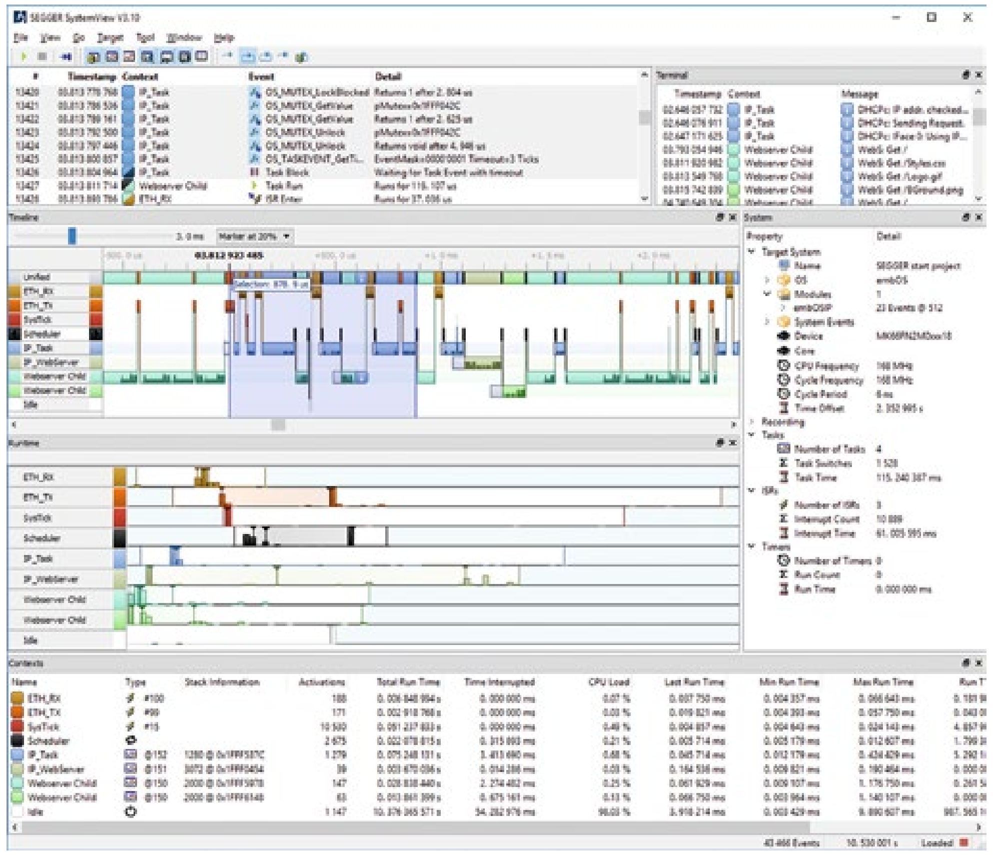
SystemView is a freely available software tool for the visualization of any embedded system. Its ability to provide deep visual analysis revolutionizes the way in which embedded systems are developed and handled. Optimizing a system becomes easier as bottlenecks can easily be found. Users can identify wrong processing orders easily with the visualization provided by SystemView. Any system, with or without RTOS, can be monitored.

SystemView visualizes and analyzes CPU load by tasks and interrupts. Test setups with LED and oscilloscope are a thing of the past. The tool offers cycle-accurate tracing of interrupts and task start/stop as well as task activation and API calls when an RTOS is used. The information gathered and visualized can be used to analyze both bare metal and RTOS-based systems.

SystemView consists of two parts: The PC visualization application SystemViewer, and some code collecting information on the target system. The combined ROM size of RTT and SystemView modules is less than 2 KByte. In a typical system, about 600 Bytes of RAM are sufficient for continuous data recording via J-Link.

Real-Time with J-Link RTT

SystemView has outstanding capabilities when it comes to real-time data collection, enabled by the J-Link Real-Time Transfer (RTT) technology. RTT allows monitoring of events, interrupts etc. in real-time. The data transfer rate of 2MByte/s is exceeding the required bandwidth and thus clearly surpasses the bandwidth of other solutions with a similar purpose. The overhead is less than 1 µs per call (measured for a 200 MHz Cortex-M).



While SystemView works without a J-Link, the debug probe is needed to use the RTT feature. The technology has no additional hardware requirements beyond the standard debug interface. SWO or trace pins are not required. This allows using these extended debug and analysis capabilities on Cortex-M0 based systems as well.

Platform-independent

On systems which are not supported by the RTT technology, the buffer content can be read manually when the system is halted, which allows single-shot recording until the buffer is filled. Single-shot recording can be triggered by the system to be able to control when the recording starts.

SystemView records the data read out from the target and visualizes it in different ways. Recordings can be saved for later documentation and analysis.

Extended Debugging and Analysis

Analyzing the runtime behavior of a system and the interaction among tasks and between tasks and interrupts can be employed for various purposes: For the documentation of new systems, validation of the desired behavior, examination of erratic behavior such as in interrupt handling, or for the analysis and improvement of performance.

SystemView PRO is available, providing advanced filters and unlimited event recording.

Data acquisition via UART and TCP/IP

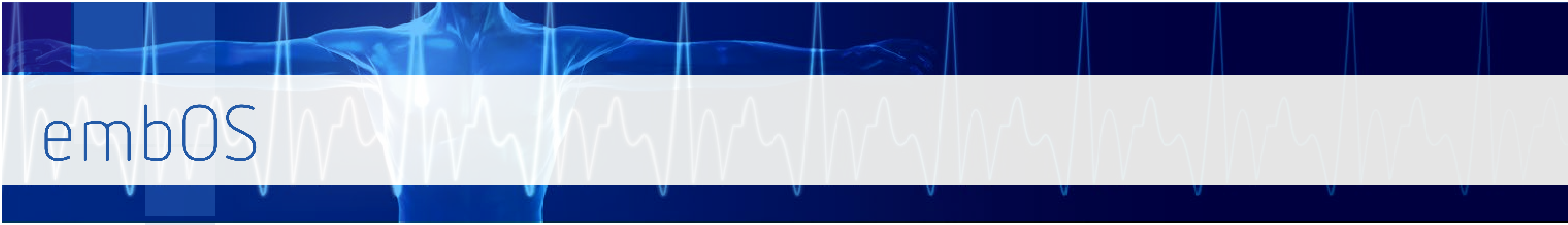
Any system with UART or TCP/IP connection, typically Ethernet, can be monitored and verified. SystemView package includes the target code required for integration as well as example projects.

New Features

The software also adds a variety of new features. New performance markers are useful for performance verification and optimization. A new Runtime window provides information on the runtime distribution of tasks, interrupts and software timers. All windows are updated in real time, with no limit on the sampling time due to streaming data acquisition. The entire data set can be stored for later or remote analysis as well as archival and documentation purposes. SystemView also comes with the files required for many popular RTOS such as embOS, FreeRTOS, uC/OS-II and uC/OS-III, and more.

Features

- Maximum system insight
- Minimally intrusive
- Cycle-accurate profiling
- No additional hardware required
- RTOS tracing
- Interrupt tracing without an RTOS
- Continuous real-time recording and live analysis with J-Link and SEGGER RTT technology
- Works on any CPU
- Data acquisition types: Debug interface, UART and TCP/IP
- Available under SEGGER's friendly licensing policy



embOS

The Real-Time Operating System embOS

For more than 25 years in the embedded market, embOS has been and still is the preferred embedded OS choice for engineers all over the world. It offers incomparable ease-of-use and guarantees 100% deterministic real-time operation for any embedded device. It is highly portable and fully source-compatible on all platforms, making it easy to port applications to different cores.

Developing Applications with embOS

embOS is available in source or object code form. With more than 500 board support packages, embOS runs out-of-the-box on any common hardware. Each board support package comes with all required initialization code and a start project for the IDE of your choice. Run your first embOS application in less than 5 minutes! The BSPs and samples are supplied in source code form. Libraries for all memory models and initialization of the controller in C source are included to tailor the system to any hardware. embOS is free for any non-commercial use like education and evaluation, without any limitations.
For commercial projects, SEGGER offers a versatile embOS licensing policy with no royalties at a reasonable price.

Technical info

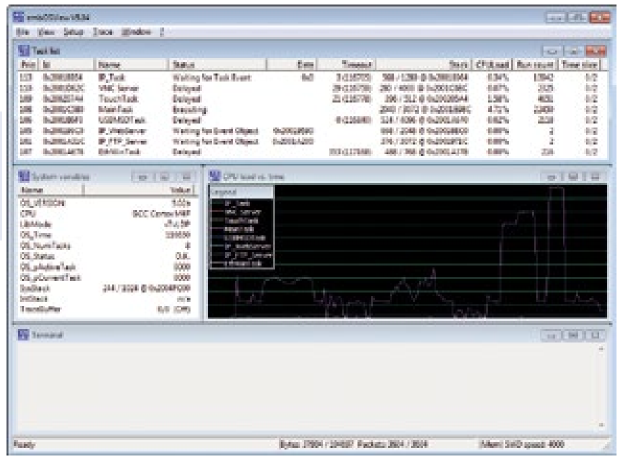
Memory Usage

Kernel size (ROM)	1700 byte *
Kernel RAM usage	67 byte *
Task RAM: Task control block	36 byte *
Task RAM: Minimum stack size	88 byte *

Timing

Context switching time	270 Cycles (1.4 µs with STM32F756 running at 200 MHz)*
Interrupt latency	Zero

* Depends on CPU, and compiler used



(embOSView)

Professional Solution

embOS is suitable for any application, from battery-powered single chip products to high-end systems demanding ultra-high performance. The RTOS embOS, developed by The Embedded Experts, is used in many different target markets such as industrial controls, Internet of Things, networking, consumer electronics, safety critical devices, automotive to medical devices and avionics, making it the market leading RTOS in the embedded industry.

Profiling using SystemView

embOS is fully instrumented for the use with SystemView. SystemView provides a complete picture of all tasks and their behaviour including the usage of all embOS objects, such as mailboxes and timers. In addition, the user gains complete insight into interrupt usage. For more information on SystemView please read the section "SystemView-Real-Time Analysis and Visualization".

Profiling using embOSView

embOSView gives perfect insight into embOS. This profiling tool is included with all embOS shipments. It communicates with embOS running on the target via UART, IP or debug interface and displays all available information of the tasks and major system variables.

Features

- More than 25 years of continuous development
- Available for all popular cores, compilers and development tools
- Deployed in several billion devices in a wide range of application areas
- Certified for functional safety according to IEC 61508 SIL 3 and IEC 62304 Class C
- MISRA-C:2012 compliant
- Powerful and easy-to-use API
- Highest performance with lowest use of memory
- Kernel awareness plugins available
- Zero interrupt latency
- Built for functional safety
- Same code base and API for all embOS editions: embOS-Safe, embOS-MPU and embOS
- Royalty-free
- Unlimited number of OS objects, such as: tasks, semaphores, mailboxes, timers
- Unlimited number of task priorities
- Profiling and trace support
- Low power support including tickless mode
- Seamless watchdog support
- Multi-core support
- Cycle-precise timing routines
- Easy to use board support packages
- Free for non-commercial use

Using an embOS profiling build, embOSView collects and displays precise timing information for all tasks.

Zero-Latency Interrupts

embOS is perfectly suited for hard real-time conditions as it does not block zero-latency interrupts. Instead of disabling interrupts when performing atomic operations, embOS sets the interrupt level of the CPU to a specific threshold. Therefore, all interrupts with interrupt priorities higher than this threshold - the zero-latency interrupts - can still be processed. embOS will never add any latency to zero-latency interrupts. This makes it easy to use time critical interrupt service routines with embOS.

Simulation Environment

A simulation environment running under MS Windows is available. It can be used to write and test the entire application on your PC (all API is 100% identical to your embedded application). This makes debugging and development easy and convenient and saves development time. The simulation is an open environment, which also allows adding C code to simulate the target specific hardware. embOS Simulation comes with a ready-to-go project for Microsoft Visual Studio or MinGW and Eclipse.

Power-Saving Tickless Mode

embOS low-power support reduces the power consumption for e.g. battery powered devices. Instead of having a timer interrupt for each system tick, the timer is reprogrammed with the embOS tickless mode to be able to spend as much time as possible in low-power mode.

Supported processor families

All major processor families are supported!

ALTERA
MM32 MCU
SILICON LABS
ANALOG DEVICES
MIPS
ST
arm
NORDIC
TEXAS INSTRUMENTS
CYPRESS
nuvoTon
TOSHIBA
dialog
NXP
XILINX
GigaDevice
REALTEK
.. and more
HOLTEK
RENESAS
MICROCHIP
RISC-V

embOS-MPU

Enhanced Safety Real-Time Operating System embOS-MPU

embOS-MPU offers memory protection on top of the proven real-time operating system embOS. It significantly enhances both stability and safety of your embedded applications and thereby simplifies any certification process. The operating system and all tasks deemed privileged are memory-protected and isolated from any ill effects of unprivileged tasks. Due to a fully compatible API, existing embOS applications may be adapted to embOS-MPU with minimal effort.
embOS-MPU can be used in any application from battery-powered, single chip products to systems demanding ultra-fast response, flexibility and multiple tasks. Typical fields include, but are not limited to, medical equipment, automation, avionics and other safety-critical applications.

Features

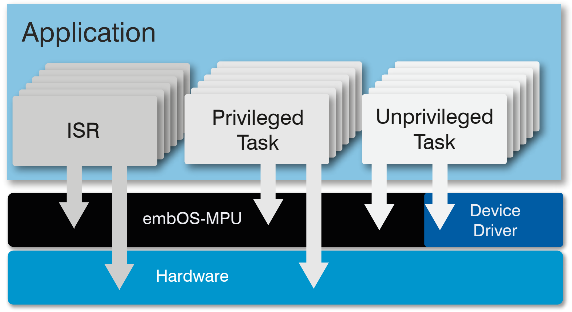
- Suitable for any safety-critical application
- Available for any MCU containing a hardware MPU or MMU
- Unlimited number of privileged und unprivileged tasks possible
- Unprivileged tasks 100% sandboxed
- Simple and straightforward runtime configuration
- Easy to integrate in both new and existing products

What is Memory Protection?

Memory protection is a prevalent mechanism to control memory access rights and is part of most modern processor architectures and operating systems. The main purpose of memory protection is to avert specific tasks from accessing memory that has not been allocated to them, thus preventing possible bugs or even malware contained in one task from affecting the entire system.
In order to achieve this, application tasks that could possibly affect other tasks or the OS itself must be restricted from accessing the whole memory, special function registers and the OS's control structures. For example, tasks that execute third-party code may be considered unsafe and should be restricted accordingly. Such application tasks must not run on the same privilege state as the OS, which runs in fully privileged mode and has access to all memory, peripherals and CPU features. Instead, these tasks must run in an unprivileged state and have restricted access to specific memory locations only.

What is embOS-MPU?

embOS-MPU uses the hardware's memory protection unit as well as additional software mechanisms implemented with embOS-MPU to prevent one task from affecting the entire system. This guarantees that even in case a bug occurs in one task, all other tasks and the operating system itself continue execution.
With embOS-MPU, all privileged tasks have full access to the whole memory. An unprivileged task, however, has specific access permissions to each distinct memory region. In order to access peripherals, additional memory locations and OS control structures, device drivers as well as specific embOS API may be called from within an unprivileged task.



embOS-Safe

Certified
IEC 61508 SIL3,
IEC 62304 Class C

Certified Real-Time Operating System embOS-Safe

TÜV Süd Germany has certified the real-time operating system (RTOS) embOS according to IEC 61508 SIL 3 and IEC 62304 Class C. IEC 61508 is the standard for functional safety and referenced in multiple derived standards in different areas. embOS certifications according to other standards, including ISO 26262, can easily be done.

High-Quality Development Processes

The certification, which was conducted in close cooperation with our partner Embedded Office, confirms the quality of the SEGGER development processes and demonstrates that embOS-Safe is perfectly suited for being the fundamental component of safety products. Trust our professional services and take advantage of all RTOS key features like multi-tasking, comprehensive communication and synchronization services, as well as memory protection. For customers, no effort is required for RTOS certification at all. embOS-Safe comes with a certification kit containing all necessary documents, including the comprehensive embOS safety manual.

Safety tested
Production monitored
TÜV SÜD
Functional Safety

Features

- Enhanced safety for your embedded products
- Simplifies certification and reduces risk
- Certified according to IEC 61508 SIL3 and IEC 62304 Class C by TÜV Süd, Germany
- Same powerful API as embOS and embOS-MPU

Safety for All Product Areas

The RTOS is the most critical component in many safety critical applications. embOS-Safe can be used to isolate safety critical code, ensuring it can operate without interference with other tasks. This simplifies the certification of your application.
embOS-Safe has been designed specifically for the safety sector, including industrial, medical, automotive and home appliance applications.

Consistent, Proven Interface

The Application Programming Interface (API) is unchanged in relation to embOS. Therefore, existing software parts can be (re-)used easily. This helps to use embOS-Safe in existing applications. There is no migration guide required, still SEGGER's knowledgeable and responsive support team is available, if necessary.

One-Stop Solution

The certified RTOS embOS-Safe is available for several popular cores and compilers. With SEGGER's own IDE Embedded Studio, the offering becomes a one-stop shop solution. Naturally, embOS-Safe is fully suited for usage with SEGGER's extensive portfolio of outstanding embedded software, debug probes and production tools, too.

emFile

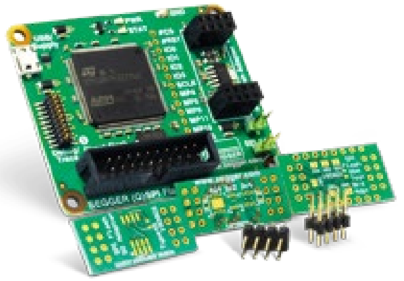
Data Storage with emFile

emFile is a file system for embedded applications that can be used with any type of storage device. It is a versatile high-performance library that was optimized for minimum RAM and ROM consumption and for high speed. The library is written in ANSI C and runs on any CPU.

emFile works with any kind of embedded system and storage device. Ready to use device drivers are provided for the most popular storage devices such as NAND and NOR flash, SD card and eMMC.

NAND and NOR flash

The file system comes with device drivers for NAND flash that support the single-level cell (SLC) as well as the multi-level cell (MLC) technology. Single and multi bit error correction is performed either in the hardware using the ECC engine built into the NAND flash device or in a dedicated NAND flash controller or in the software with the help of SEGGER emLib-ECC. To enable the use of large NAND flash devices, the NAND driver allows the grouping of storage blocks that saves RAM required for their management. NOR flash drivers are provided that can be used with any popular serial and parallel NOR flash device. CFI compliant as well as quad SPI NOR flash devices are supported.



SEGGER (Q)SPI Flash evaluation board

The NAND and NOR drivers make use of static and dynamic wear leveling for the purpose of increasing the lifetime of the storage device. This is realized by keeping track of the number of times a storage block as been erased and by making sure that each storage block is erased for approximately the same number of times. In addition, the NAND and NOR drivers support garbage collection that is performed automatically as required. The garbage collection can also be performed in the application to help minimize the write latencies.

SD card and eMMC

emFile comes with device drivers for accessing SD cards and eMMC devices. SD cards of any type and capacity are supported including SDSC, SDHC and SDXC of any form factors. The same device driver works for all the device types which helps reduce the ROM usage. Efficient data transfer is guaranteed by taking advantage of SD host controllers that can transfer the data via 4 and 8 data lines using DMA. For resource-constrained systems without an SD host controller, a device driver is provided that can exchange the data with an SD card via SPI.

Encryption

The emFile Encryption add-on provides a simple way to encrypt individual files or the storage device as a whole. Encryption can be used with both available file systems - EFS and FAT. All storage device types such as NAND and NOR flash, SD card and eMMC cards are supported. To use encryption, only minor changes to the application program are required in order to select the encryption method and a password for volume or individual files.

Journaling

Journaling is an additional component for emFile which resides above the file system and makes the file system layer fail-safe. File systems without journaling support (for example, FAT) are not fail-safe. Journaling means that a file system logs all changes to a journal before committing them to the main file system. To prevent corruption from unexpected interruptions, caused for example by a power failure, the journaling layer caches every write access to achieve an always consistent state of the file system.

Features

- MS DOS/MS Windows-compatible FAT12, FAT16 and FAT32 support, proprietary EFS file system
- RAID-support (RAID0/RAID1/RAID5) for added performance and safety
- Support for long file names
- Multiple device driver support
- Multiple media support: A device driver allows you to access different media at the same time
- Cache support: Improves the performance of the file system by keeping the most recent used sectors in RAM
- Works with any operating system to accomplish a thread-safe environment
- ANSI C stdio.h-like API for user applications
- Very simple device driver structure: emFile device drivers need only basic functions for reading and writing a block
- Optional NOR flash (EEPROM) driver: Any CFI-compliant NOR flash is supported; Wear-levelling included
- Optional device driver for NAND flash devices: Very high read/write speeds; ECC and wear-levelling included
- An optional device driver for MultiMedia & SD cards using SPI mode or card mode that can be easily integrated
- An optional IDE driver, which is also suitable for CompactFlash, using either True IDE or Memory Mapped mode
- An optional proprietary file system (EFS) with native long file name support
- An optional journaling add-on. It protects the integrity of the file system against unexpected resets
- NAND flash evaluation board available
- MISRA C:2012 compliant source code
- Optional support for Japanese characters in file names
- Optional BigFile add-on that provides support for files larger than 4 Gbytes.

Memory requirements*

Memory requirements depend on the used CPU, compiler, memory model, as well as on various other factors such as configuration switches and selected drivers.

ROM:	app. 9-40 kB
RAM:	app. 2 kB

* Precise values depend on the functionality used. Values are measured on a specific target system and will be different on other systems.

emCompress

Achieve more with highly efficient data compression!

emCompress is a compression system that is able to reduce the storage requirements of data to be embedded into an application. A compressed version of the data is stored in the flash memory of the target system. In the target, a small, fast decompressor can decompress the data on-the-fly, whenever it is required. The decompressor can decompress into RAM or send the output to an application defined function.

More Storage

By compressing the data, a lot more can be stored in flash memory, SPI flash or on the external SD card. This allows a data logger to store more information, or to be less expensive due to the smaller, less expensive storage medium.

More Bandwidth

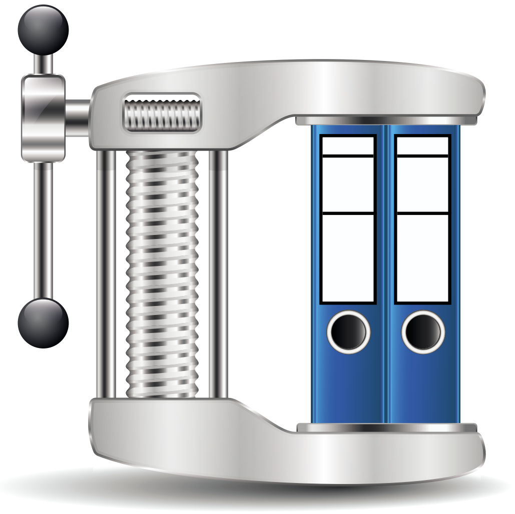
On applications using a low bandwidth link, from local applications using Bluetooth, to space applications such as a Lunar lander sending data to Earth, more information with compression can be transmitted, in a shorter period of time.

Software only Grows in One Direction

With increasing complexity of today's devices, customers expect firmware updates over the lifetime of a device. It's important to be able to replace both firmware images and FPGA configuration bitstreams in the field. Typically, firmware upgrades and other static content grow in size over the lifetime of a device: features are added to software, not taken away, it's just what happens.
This is where emCompress can help. emCompress will compress your data so that it takes much less space on the target device. The decompressors are tiny in ROM, but the benefits of compression means you reclaim more space in your device for the features you're trying to add.
Because emCompress decompressors can be tailored for RAM use, you can decompress static content early in your application and don't devote RAM to decompression buffers.

Compress your Data in Real-Time on any Embedded System!

emCompress-ToGo is a portable compression software, designed primarily for but not limited to embedded systems. It can compress and decompress data with basically no RAM requirements.
The lossless compression is fast and efficient and can be easily done on the embedded system, just like the decompression. In general, there are multiple compression algorithms that allow compression on the host and decompression on target (like emCompress-Embed). There are not many options, if any, to compress easily and efficiently on an embedded system, especially with limited RAM.
emCompress-ToGo uses SEGGER's SMASH (Small Microcontroller Advanced Super High) compression algorithm, specifically designed for these type of applications achieving amazing compression ratios.



Features

- Highly efficient compression
- Small ROM footprint
- No static RAM required,
  configurable decompressor RAM use
- Compressor running on microcontroller based embedded systems
- Wide range of codecs to choose from
- Automatic selection of best codec for each file
- Compression of files and streamed data
- Simple to configure and integrate
- Royalty-free

Decompressing and Processing Data

emCompress-ToGo and emCompress-Flex feature two modes for compression and decompression.
The first works directly on memory. The complete data is compressed or decompressed and stored in a user-provided memory buffer. Although the buffer can be temporary, it requires to have enough free memory for the complete uncompressed data and the workspace. Decompression into memory can for example be useful for dynamic firmware images.
The second mode takes data from a function or sends data to a function, which can easily be used for streaming. Streamed decompression is particularly effective for programming FPGAs or serving web content.

Typical uses of emCompress

Compression has many fields of application. Applications that use static data which is not frequently used and/or has exceptionally high compression rates are the prime target applications for emCompress-Flex and emCompress-Embed. Typical examples are the configuration bitstreams to program FPGA and CPLD devices, permanent files for embedded web server static content, firmware updates and the user interface messages for multiple languages.
With emCompress-ToGo the typical examples are extended to dynamic data – files and streams – that are directed to or generated on embedded systems. This is particularly useful for data loggers and in low bandwidth situations such as Bluetooth devices, avionics or space technology, or satellites.

Edition	emCompress-ToGo	emCompress-Flex	emCompress-Embed
Description	Compress files and streams on host and target with a small and fast algorithm that requires nearly no RAM.	Compress files with standard algorithms on host and decompress on target.	Compress static data to be linked with the application and decompress on target.
On Target Decompression	✓	✓	✓
On Target Compression	✓	☒	☒
Decompress Static Data	✓	✓	✓
Decompress Files&Streams	✓	✓	☒
Available Algorithms	SMASH	LZMA	DEFLATE, Huffman Encoding, LZW, LZSS, LZJU90, RLE-PAR
Use Case	Files and streams directed to or created on embedded systems. Data loggers, Bluetooth devices, avionics and space technology, satellites.	Files compressed on the host and transferred to the application such as firmware updates.	Static data compiled into the application such as FPGA streams.

emNet

Get connected with emNet

emNet is a TCP/IP stack that provides a small memory footprint and high performance for embedded networking solutions. The stack has been optimized for use in real-time, memory-constrained embedded systems. It offers RFC-compliant TCP/IP and a standard socket API. emNet works seamlessly with the embOS operating system. A variety of common and specialized higher level protocols is available as well.

Performance and Resource Usage

The stack has been optimized for both performance and code size. The standard socket interface is complemented by the zero-copy API, which allows reading and writing of data without additional protocol related copy operations.

Multi-Task Support

emNet allows any number of tasks to call API functions concurrently.The stack itself typically uses one task for housekeeping and an interrupt for reception. A second task can be used for reception, allowing minimum interrupt latency. A no task (polling) setup is possible as well.

emNet Structure

emNet is organized in different layers:

Application Layer	DHCP, DNS, FTP, HTTP, …
Transport Layer	TCP, UDP
Network Layer	IPv6, IPv4, ICMP, ARP, …
Link Layer	Ethernet (IEEE 802.3), PPP, WiFi

Easy to Use

The stack comes with a variety of confidence tests and example applications. It runs out-of-the-box. For most microcontrollers, sample configurations are available in the evaluation packages that are available for all popular boards and offer an out-of-the-box experience. All modules can output debug messages and warnings in debug builds. The modules to output this information can be selected at run-time, allowing the developer to focus on the aspect he is analyzing.

Configuration Free

The entire stack can be compiled into a library. Setup is reduced to a minimum, performed at run-time. This, along with a number of sample programs, gets you up and running quickly. Since inter-module dependencies are limited to the parts required for the functionality of the stack, unused parts of the code are automatically excluded by the linker.

emNet Software Products

emNet is offered in a BASE package which includes the most important protocols related to Ethernet communication and the stack itself. Depending on the engineer's needs there are several protocols available as add-ons, as well as the emNet PRO package which includes a device driver and add-ons for extended communication via Internet.

The following protocols are, among others, part of the emNet BASE package:

- ARP (Address Resolution Protocol)
- IP (Internet Protocol)
- ICMP (Internet Control Message Protocol)
- UDP (User Datagram Protocol)
- TCP (Transmission Control Protocol)
- Standard Socket API
- Zero-Copy API
- DNS client
- DHCP client

You can find the full list of additional modules at **www.segger.com**

Features

- Standard sockets interface
- Non-blocking socket support
- High performance
- Small footprint
- Runs "out-of-the-box"
- No configuration required
- Works with any RTOS in a multitasking environment (embOS recommended)
- Zero-copy for ultra-fast performance
- Drivers for most popular microcontrollers and external MACs available
- Tail-tagging support
- Raw socket support
- Unlimited Connections
- IP fragmentation support
- Fully runtime configurable
- Developed from scratch for embedded systems
- PPP/PPPoE available
- Various upper layer protocols available

Additional Products

emWeb Web Server
The emWeb Web Server allows embedded systems to present web pages with dynamically generated content. It has all features typically required by embedded systems: multiple connections, authentication and forms. The RAM usage of the web server has been kept to a minimum by smart buffer handling. The socket interface can be used with any TCP/IP stack.

emNet FTP server and FTP client
The emNet FTP server can use the same file system as the web server. It can be used in r/o or in r/w mode and allows reading and modifying of configuration data or web content. With the FTP client add-on data can be exchanged with any FTP server

emSSL Transport Layer
emSSL is a Transport Layer Security solution which allows secure and private connections with single-chip systems using as little as 7 kBytes of RAM. emSSL works perfectly with emNet. For more details about emSSL, please refer to the product description in section "Safe Data Transport with emSSL".

An excerpt of the available add-ons

emNet CoAP client/server
Client/server data collection service

emNet MQTT Client
IoT Messaging Protocol

emNet SNTP Client
Simple Network Time Protocol

emNet SNMP Client
Simple Network Management Protocol

emNet SMTP Client
Sending emails from your embedded system

emNet Websockets Client/Server
Firewall-friendly communication

Memory requirements

Memory requirements depend on the used CPU, compiler and memory model, as well as on various other factors. A typical ROM size for a system using ARP, IP, ICMP, UDP, TCP and sockets is about 18kB (on typical 32-bit microcontroller with size optimization). Minimum RAM usage is about 6KB for simple applications.

emUSB-Host

Get connected with emUSB-Host

SEGGER's emUSB-Host software stack provides PC-like plug and play peripheral support to embedded systems similar to desktop computers.
Whether a network connection needs to be added or audio communication is required, SEGGER's emUSB-Host stack enables all kinds of connections with standard USB peripherals.

The emUSB-Host software stack supports all transfer modes (control, bulk, interrupt, isochronous). It complies with the USB v1.1 and USB v2.0 specifications. USB pipe management and extended error recovery mechanisms that are required for reliable operation are implemented internally.

emUSB-Host is a USB host stack specifically designed for embedded systems. The software is written in ANSI C and can run on any platform. emUSB-Host can be used with embOS or any other supported RTOS. A variety of target drivers are already available. Support for new platforms can usually be added at no extra cost.
The modular design enables applications to access the USB host programming interface directly, or to use APIs exposed by class drivers.
The stack can handle multiple devices and hubs simultaneously and fully supports hot plugging of devices and hubs. The programming interface supports dynamic device enumeration and identification.

Control Light and Sound

emUSB-Host has recently added support for the audio class, extending the supported peripherals to audio input and output devices. Additionally, emUSB-Host supports the MIDI class, which enables any device running emUSB-Host to serve as a light and music controller.

Audio

CCID

CDC

HID

LAN

MIDI

MSD

MTP

Additional classes

FT232, CP210X, Printer, Vendor (bulk)

Features

- ISO/ANSI-C source code
- Supports USB 1.1 / 2.0
- High-Speed support
- OHCI, EHCI and proprietary drivers
- Extended error recovery during enumeration
- Multiple hub support
- Modem support (3G, 4G)
- LAN device support
- Royalty-free
- MISRA-C:2012 compliant

Gaining Access to the Network

emUSB-Host provides support for LAN via USB. It can use either CDC-ECM or RNDIS for this purpose, making it compatible for almost all devices that can provide their IP connection via a USB device interface.

Storing Data

The Mass Storage Device class supports USB-sticks and external drives, such as CD-ROM or other storage drives. As the market offers many USB-sticks that do not follow the standard closely, emUSB-Host can be configured to accept such devices.

Exchange Media Data

The Media Transfer Protocol (MTP) class supports object-based communication with the device for all types of files. MTP typically is supported by smartphones, digital cameras or other multimedia devices for multimedia content.

Serial Interface Support

emUSB-Host offers all options for serial communication. The offer includes serial communication via the modem class CDC-ACM, or via USB-UART converters such as FTDI's FT232 series of or Silicon Labs CP210x series. Such converters are very common to connect to legacy devices, but there are also numerous devices on the market, which have a USB device interface has been implemented around a serial-USB-converter.

Human Machine Interfaces

The Human Interface Device class (HID) was defined for the handling of devices which are used by humans to control the operation of computer systems. The class enables the use of input devices such as bar code scanners, keyboards, mice, trackballs or touch-controllers.

Printer

A printer or plotter connected to emUSB-Host is automatically configured. The printer class forwards the output data to a connected printer. The actual output has to be generated using printer specific codes.

USB On-The-Go (OTG)

USB On-The-Go (OTG) introduces the dual-role device, meaning a device capable of functioning as either host or peripheral. The On-The-Go-module of emUSB automatically decides whether it has to operate the host or the device side when a new connection is established.
On-The-Go is typically used when only a single USB port is available for host and device use. Digital cameras use the port to provide their data to a host system for further editing or to connect to a printer. Some smartphones and tablets have a single port for device and host functionality as well. USB OTG retains the standard USB host/peripheral model, in which a single host talks to USB peripherals. emUSB OTG offers a simple interface in order to detect the role of the USB OTG controller.

emUSB-Device

Get connected with emUSB-Device

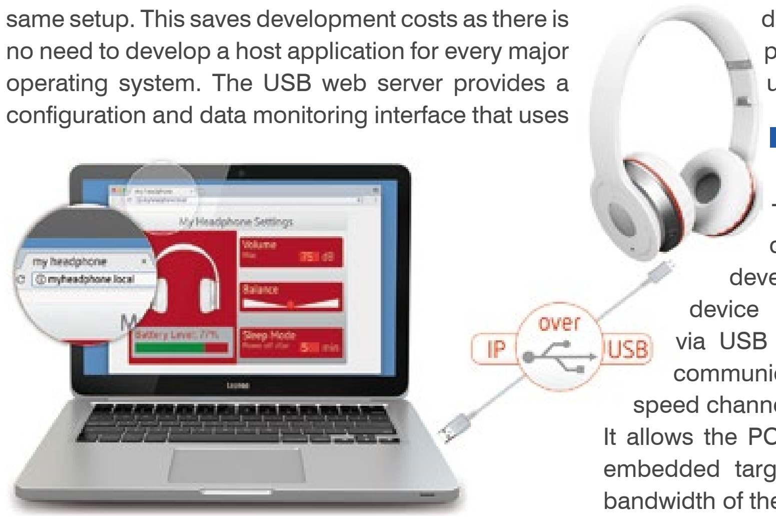
emUSB-Device is a USB device stack specifically designed for embedded systems. The software is written in ANSI C and can run on any platform. emUSB-Device can be used with embOS, any other supported RTOS, or even without an OS. A multitude of target drivers are already available. Support for new platforms can usually be added at no extra charge.

emUSB-Device Components

SEGGER offers a flexible USB device stack structure. The typical emUSB-Device stack package consists of a target driver designed for your target hardware, the emUSB-Device core and the Bulk, Audio, CDC-ACM, CDC-ECM, RNDIS, HID, MSD, MTP or printer component, or for any combination thereof. The different available hardware drivers, the USB class drivers and the Bulk communication component are additional packages, which can be combined and ordered depending on project requirements.

IP-over-USB Component

A USB based web server can be created with little effort using the IP-over-USB component. A higher end device with a real Ethernet connector would use the same setup. This saves development costs as there is no need to develop a host application for every major operating system. The USB web server provides a configuration and data monitoring interface that uses a web browser as a user interface on the host system. The IP-over-USB component automatically detects the nature of the connected host-system and starts the required class (RNDIS, CDC-ECM). The classes started do not require any driver to be installed on the host-system as they are supported natively already. An application example for the IP-over-USB component are low-end stand-alone products that shall be converted into connected devices with the same functionality as other devices on a local network.



SmartMSD Component

The SmartMSD component allows to easily stream files to and from USB devices. Once the USB device is connected to the host, files can be read or written to the application without the need for dedicated storage memory.

It can be used for various types of applications and purposes, with no additional software or drivers necessary on the host side. The SmartMSD software analyzes which operation is performed by the host and passes this to the application layer of the embedded target, which then performs the appropriate action. A simple drag and drop is all it takes to initialize this process, which is supported by a unique active file technology.

Bulk Communication Component

The Bulk component allows developers to quickly and painlessly develop software for an embedded device that requires fast communication via USB (HiSpeed up to 42 MByte/s). The communication is like a single reliable high-speed channel (very similar to a TCP connection). It allows the PC to send and receive data with the embedded target. This permits usage of the full bandwidth of the USB.

Features

- Supports USB 1.1 / 2.0
- Bulk communication component with driver package for Linux, Mac and Windows
- MSD class component
- Smart MSD works without file system, works cross-platform, does not require any driver
- IP-over-USB, works cross-platform, does not require any driver installation
- Audio class component
- MIDI class component
- CDC-ACM class component
- HID class component
- MTP class component
- Printer class component
- Royalty-free
- Windows driver HLK certification
- USB 3.0 Super Speed Support
- MISRA-C:2012 compliant

Audio Component

The Audio Component enables streaming of audio data to and from the USB device. With this class audio data can be recorded or replayed.

MSD (Mass Storage Device) Component

emUSB-Device MSD converts your embedded target device into a USB mass storage device. Your target device can then be plugged into a USB host and be accessed as a disk drive, without the need to develop a driver for the host operating system. This is possible because the mass storage device class is one of the standard device classes, defined by the USB Implementers Forum. Every major operating system on the market supports these device classes out of the box. Since the host already includes drivers for USB mass storage devices, the target device will be recognized as a mass storage device and can be accessed directly.

MTP (Media Transfer Protocol) Component

The MTP class supports object-based communication with the host system for all types of files. MTP is an official extension of the Picture Transfer Protocol designed to allow digital cameras to exchange image files with a computer. MTP extends this by adding support for all types of files. MTP is an alternative to MSD, without some of the latter’s weaknesses. Object based communication gives access to the file system from the host system (PC) and the device at the same time. Using the MTP class also allows selectively exposing content of the file system to the host system, typically a PC. Sudden removal of the USB cable does not endanger the data integrity of the device’s file system.

CDC (Communication Device Class) Component

emUSB-Device CDC-ACM converts the target device into a serial communication device. The host will recognize it as a virtual COM port (USB2COM).

HID Component

The Human Interface Device class (HID) is an abstract USB class protocol defined by the USB Implementers Forum. This protocol was defined for handling devices which are used by humans to control the operation of computer systems. emUSB-Device also allows 'Vendor specific HIDs'. These are HID devices communicating with an application program. The host OS loads the same driver as for any HID and automatically enumerates the device. The application communicates with the particular device using API functions offered by the host. This allows an application program to communicate with the device without the need for loading a custom driver. The HID class is a good choice if ease of use is important and high communication speed is not a requirement.

Printer Component

The USB class protocol for printers was defined for handling output devices like printers and plotters. The emUSB-Device printer receives data from the host and forwards the data to a parser. The printer component provides automatic error handling routines, in case of events like the device running out of paper. The USB class protocol is completely hidden from the developer so he can concentrate on developing the parser.

SEGGER IoT & Security

All-in-One Solution for the Internet of Things

Everybody talks about the Internet of Things (IoT), and we provide everything to build it. Today IoT encompasses a diverse range of smart connected devices and capabilities. This means a broad range of embedded software requirements need to be satisfied when developers are choosing their IoT solution partner. With SEGGER, you can accelerate time to market with security, connectivity, reliability and tools that simply work.

What is an IoT device?

At first there is the functional side of the IoT device. This functional part is essentially an embedded system as we have known for years. IoT devices interact with each other, through a server or via the cloud. To achieve this, the device needs to find its counterpart; this is supported by protocols such as MQTT. The connection itself is handled via IP or UDP based protocols.

Communication increases the exposure of the IoT device, which makes it more important to keep the different aspects of security in mind during development. There are multiple ways to attack an IoT device, therefore it needs secure components such as Transport Layer Security, secure shell, digital signatures and certificates to block out unwanted communication or snooping.

As the environment for the IoT device constantly changes, it is a requirement to establish in-field-upgrade options.

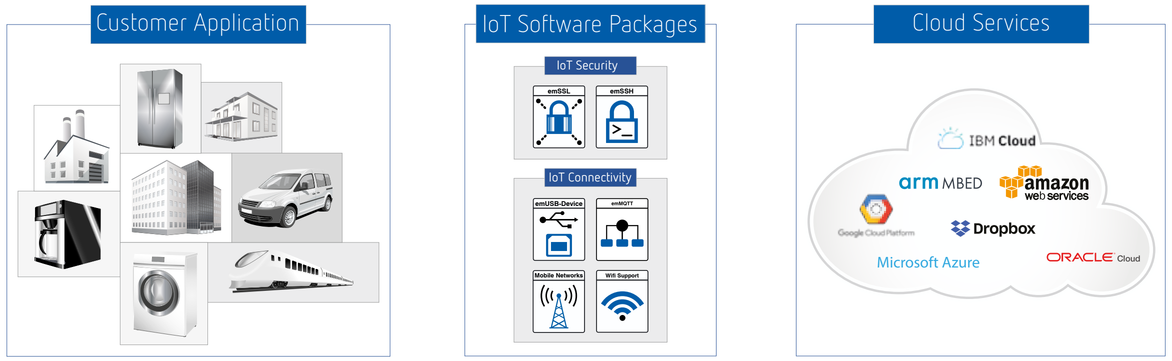
How does SEGGER support the development of IoT devices?

Software IP components from SEGGER such as emSSL, emSSH, emSecure Crypto libraries, HTTP Web server and emNet, to name a few, can be used as foundations for your securely connected IoT device. Our software works on any MCU. We work with all major semiconductor vendors, closely monitoring the market to enable the next generation of designs. Take embOS-MPU RTOS for example. This super-reliable OS works effortlessly on current Cortex-M and including the new generation Arm V8-M architecture.

Connected devices requiring User Interface (UI) or Human Machine Interface (HMI) support can take advantage of SEGGER's popular graphics package emWin. Licensed by many of the top semiconductor companies, its popularity is a testament to its robustness and flexibility, perfect for implementing your custom graphical interface.

We offer a complete end-to-end solution. In addition to our software suites, system developers can also take advantage of SEGGER's Embedded Studio IDE, our SystemView tracing tool and industry leading J-Link debug probe. All products are developed and created by one reliable, established vendor, SEGGER.

Custom IoT solutions can also be developed on demand utilizing SEGGER's embedded engineering know-how and expertise.

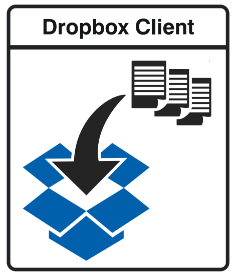


SEGGER Offering for IoT Devices

Connectivity	Safety & Security	Core	Tools
emNet	embOS-MPU	embOS	SystemView
Mobile Networks	emSSH	emUSB-Device	Embedded Studio
emMQTT	emSSL	emUSB-Host	J-Link
Wifi Support	emSecure	emCompress	J-Trace
Dropbox Client	emLoad	emWin	Flasher

SEGGER IoT & Security

Secure Data Exchange for Embedded IoT Devices using Dropbox



IoT devices have different communication needs. The new Dropbox client is ideal for projects which need to manage various amounts of data. This is accomplished by loading or storing files, even large data files, in a secure and reliable way using the public Dropbox API.

Typical Applications

Typical use cases include firmware updates, log files, and any information shared between different IoT devices and/or a central server.

Why Dropbox?

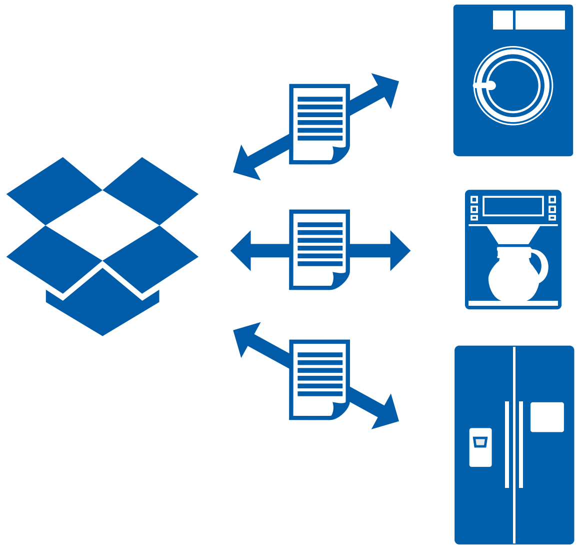
Dropbox is a great platform for this style of connected application, with proven reliability, unlimited capacity and the ability to store data for an unlimited time.
Large data files are easier to share, as they can be stored in the Dropbox and wait there to be picked up by the devices having access to the particular Dropbox. The devices can reduce their own online time, as they can reach the Dropbox at any time and go online whenever it is convenient or most effective.

Requirements

SEGGER's Dropbox client enables secure data exchange, using SSL/TLS, and a standards-compliant TCP/IP stack with a socket interface.
The client requires SEGGER's emSSL transport layer security (TLS) product; integration with SEGGER emNet networking stack enables Dropbox use on small microcontrollers.
A commercial license provides access to the full source code. A PC evaluation package is available which enables evaluation of this solution in just minutes.

Features

- Supports file upload & download
- Supports Dropbox API v1
- Support for Dropbox API v2 available upon request
- Simple to integrate
- Small footprint
- Royalty-free



Safe Data Transport with emSSL

emSSL is a SEGGER software library that enables secure connections across the Internet. emSSL offers both client and server capability. SSL/TLS is a must-have in nearly every application which is connected to the Internet. Products for IoT, smart grid or home automation markets benefit from securing their communication.

Suitable for Single-Chip Systems

The mimized RAM usage enables operation of emSSL on single-chip systems. A secure connection between browser and the web server requires only 7 KB of RAM. This way, even the smallest embedded devices can establish secure connections.

Secured Connections

emSSL offers the possibility to establish a secured connection to any client or server application from your product. It can be used both target independent in native computer applications, and in embedded targets.

The emSSL Package

emSSL is a complete package and comes with everything needed to secure communication. It includes all modules to implement the required functionality to use SSL. They are provided in source code to allow complete control of the code used in the product and create transparency to avoid worries about possible back doors or weakness in code, which cannot be checked in precompiled libraries.

emSSL Makes it Easy

emSSL comes with a simple, yet powerful API to make using emSSL in your product as easy as possible.

It also includes sample applications in binary and source code, which demonstrate how and when emSSL can be used in real life scenarios.

Features

- Complete Software Solution
- Supports Hardware Acceleration
  Performance Increase by 4-32 times
  For more details, please go to:
  https://www.segger.com/emssl-crypto-library.html
- High Performance
  Pure Software AES 2.4 MB/s
- Small Memory Footprint
  Can be scaled down to 7kB of RAM
- Verified against FIPS specifications issued by NIST (FIPS 186-4)

Performance

emSSL is built for high performance with target-independent code. It is completely written in ANSI-C and can be used in any embedded application, as well as in PC applications.

Configurable

emSSL is created for high performance and a low memory footprint. The library can be configured to fit any speed or size requirements. Unused features can be excluded, additional features can easily be added.

Supported Cipher Suites

emSSL includes the most commonly used cipher suites, which allows connection to nearly every TLS-supporting server.

With emSSL, the cipher suites can be added dynamically. When the required cipher suites are known, it is possible to create a minimal size configuration by not linking in unused algorithms. With the included scan suites application it is possible to find out the required cipher suite(s) to connect to a server.

emWin

Graphical User Interface with emWin

emWin is designed to provide an efficient, processor- and display controller-independent graphical user interface (GUI) for any application that operates with a graphical display like EPDs, OLEDs and LCDs. It can be deployed in any environment, if it is a single-task / super loop architecture or any kind of operating system architecture like embOS RTOS. emWin is shipped as C source code. It may be adapted to any physical and virtual display with any display controller and CPU.

Attractive and Useful Display with Reduced Time and Cost

One of the most challenging aspects of many development projects is designing an attractive and useful display. In addition to creating images that look exactly how you want them to appear, the implementation of window techniques, complex drawing routines, different fonts and flicker-free updates are also expected. As part of a complex process which can take months or even years, the developer only has a short time available for the implementation of the display functionality.
emWin, probably the most efficient and comprehensive embedded GUI available, helps developers beat their timelines and reduces development costs. It is written in ANSI C and supports any b/w, gray-scale or color display. Drivers for all popular LCD controllers are available. All types of graphical displays (STN-LCD, TFT, CRT, OLED, Plasma, ...) are supported.

Drivers for Display Controllers

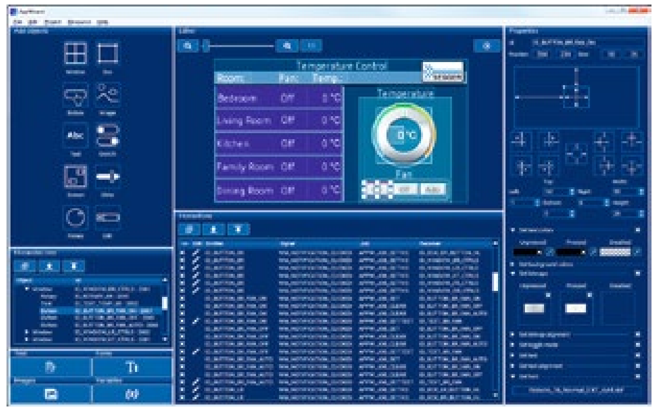
Run-time configurable drivers can be written for all types of displays and display controllers, including monochrome, gray-scale, passive and active color (TFT) displays. Drivers for all popular display controllers already exist.

Bitmap Converter

The Bitmap Converter can convert any bitmap into standard C code or into a binary format which can be located on any media and loaded at run-time. It supports palette conversion for palette based color modes as well as high color, true color or semi transparent images as PNGs. For efficiency, bitmaps may also be saved without palette data and in compressed form.

AppWizard - Advanced GUI Design

A powerful new tool for creating complete, ready-to-run applications for the company's popular emWin embedded graphics library. AppWizard is intuitive to operate and comes with its own built-in resource management capabilities. It facilitates the use of all of emWin's core functions, such as the rendering of animations, language management, widgets, etc.
One of AppWizard's notable features is the 'What You See is What You Get' (WYSIWYG) editor. Through this, engineers can design application interfaces, along with their related interactions and events, then immediately see what these applications will actually look like.
The AppWizard is part of our emWin PRO package.



Temperature Control

Room:	Fan:	Temp:
Bedroom	Off	0 °C
Living Room	Off	0 °C
Kitchen	Off	0 °C
Family Room	Off	0 °C
Dining Room	Off	0 °C

AppWizard workbench

Fonts

A variety of fonts are shipped with the software. The default set of fonts includes quite small fonts up to fairly large fonts, mono spaced and proportional fonts, special digit fonts and framed fonts. Additional fonts can easily be generated from PC fonts using the Font Converter. Monotype and TrueType fonts support is available.

Features

- ISO/ANSI C source code
- Low resource usage
- Alpha-Blending
- Anti-Aliased drawing
- Anti-Aliased fonts
- Multi buffering for flicker-free animation
- Multi-Language-Support with ASCII resource files
- Multi-Layer-Support
- Memory devices for pre-rendering images and shapes
- Free rotation and scaling of images
- Run-time-configurable drivers
- Start/test applications supplied
- Royalty-free
- Any 8/16/32/64-bit CPU; only an ANSI "C" compiler is required!
- Any (monochrome, grayscale or color) LCD with any controller supported(if the right driver is available)
- May work without LCD controller on smaller displays
- Any interface supported using configuration macros
- Display-size configurable
- Interface for hardware acceleration

Font Converter

Font Converter is a tool that makes it easy to add new fonts to emWin. It can convert any installed PC font into a C file that can be compiled and linked with the application or the binary formats ".sif" and ".xbf", which can be loaded during runtime. Simply load a font which is installed on your system into the program, edit its appearance if necessary, and save it. The generated file can then be used by emWin and be shown on the display like any other standard emWin font.

Color Management

emWin features an integrated, very efficient color management system. This system allows conversion of logical colors (RGB format) into physical colors, which can be displayed at run time. As a result, your application does not need to be concerned the available colors, and displays can easily be interchanged.

Virtual Screen Support

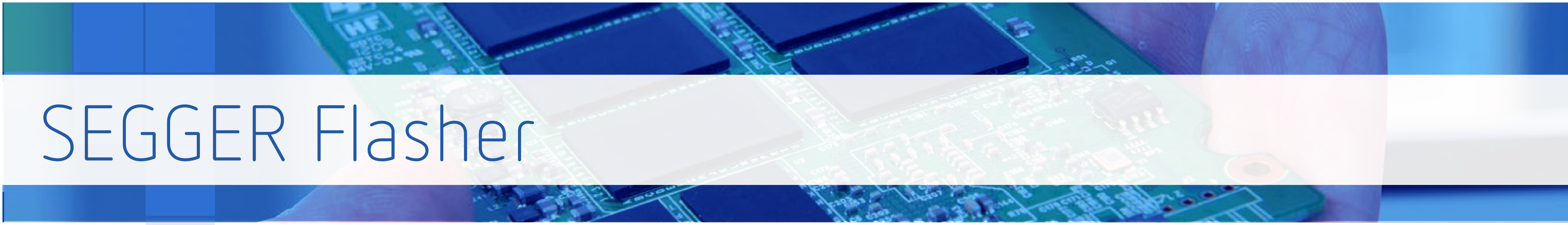
The virtual screen feature supports display areas larger than the physical size of the display. It allows switching between different screens even on slow CPUs.

Window Manager/Widgets

The window manager allows the creation of windows of arbitrary size at any point of the display. It is an optional component, which is fully integrated into the software. Child windows and the exchange of messages between windows and their children/parents are supported.
The window manager allows windows to be transparent and overlapping. Windows can freely be moved and resized. Additionally the window manager allows fading in and out. The window manager performs any necessary clipping. If callback routines are used, it also manages the redrawal of invalidated areas.

Touch Screen Support

emWin supports touch, gesture and multitouch events. The window manager deals with touch messages and widgets such as button objects. It takes only one line of code to create a button or another widget, which then handles touch messages and reacts accordingly. For resistive touch screens support is available as a driver for analog touch panels, which handles the analog input (from an AD-converter), debouncing and calibration of the touch screen is included.



SEGGER Flasher

Fast and Reliable Programming with SEGGER Flasher

The Flasher production tools are a family of in-circuit-programmers with stand-alone programming support. Each Flasher is optimized for high programming speed and engineered for robustness. The hardware and software interfaces allow an easy integration into production environments.

Universal Flash Loader

SEGGER's flash download technology is available for all microcontrollers. The Universal Flash Loader technology enables the adaption to any target system in the market. It is available for Flasher Portable PLUS, Flasher PRO and Flasher ATE.

Simply Press a Button

Using the stand-alone option, service technicians can update devices in the field by simply pressing a button after the device has been set up and loaded with the necessary programming information. With the battery powered Flasher Portable Plus, service technicians do not even need to carry an external power supply to program their targets.

The programming information is stored together with CRC data which has been generated on the host PC. This CRC data is used to verify the integrity of the data stored inside the Flasher and to verify the programming success.

Interfaces for Every Need

In addition to the features like stand-alone operation and RS232 interface, current SEGGER Flasher models support the host interfaces Ethernet and USB to control the Flasher and transfer the programming data into the Flasher. The programmers can also be used in MSD mode to transfer the programming data. Additionally they have internal ftp and web servers.

Proven Programming Solution

Serial number programming and data patching to add additional variable information are supported. The internal memory offers more than enough memory for programming data and configuration files. The programmers operate at 5V via USB interface. For many years the Flasher family of stand-alone In-Circuit-Programmers has proven to be a reliable programming solution.

Multiple Images

Multiple firmware images and configurations can be stored in the internal memory of the Flasher. Which image or configuration is programmed into the target, can be selected via the remote control interface using UART terminal or Telnet via Ethernet.

Authorized Flashing

Authorized Flashing allows the customer to limit the number of flash programming cycles and to protect the Flasher against non-authorized access to the customer's software, thus preventing the production of unwanted quantities by third parties. Authorized Flashing is available for Flasher Arm, Flasher PRO and Flasher Portable PLUS.

Features

- Stand-alone programming
- Verification of data integrity
- Remote Control
- High programming speed
- Easy integration into production environments
- USB interface
- Ethernet interface
- USB-powered
- MSD mode
- Internal FTP server
- Internal Web server
- Serial number programming
- Data patching
- Authorized Flashing

Service Programmer Flasher Portable PLUS

The Flasher Portable PLUS has an LCD to provide full transparency of the programming process and a more flexible user interface. Up to 16 configurations/images can be stored on a Flasher Portabel PLUS to programm different targets just by push button. The currently selected image is shown on the display. The high capacity rechargeable Li-ion-battery makes the user completely independent from external power supplies.

Flasher PRO

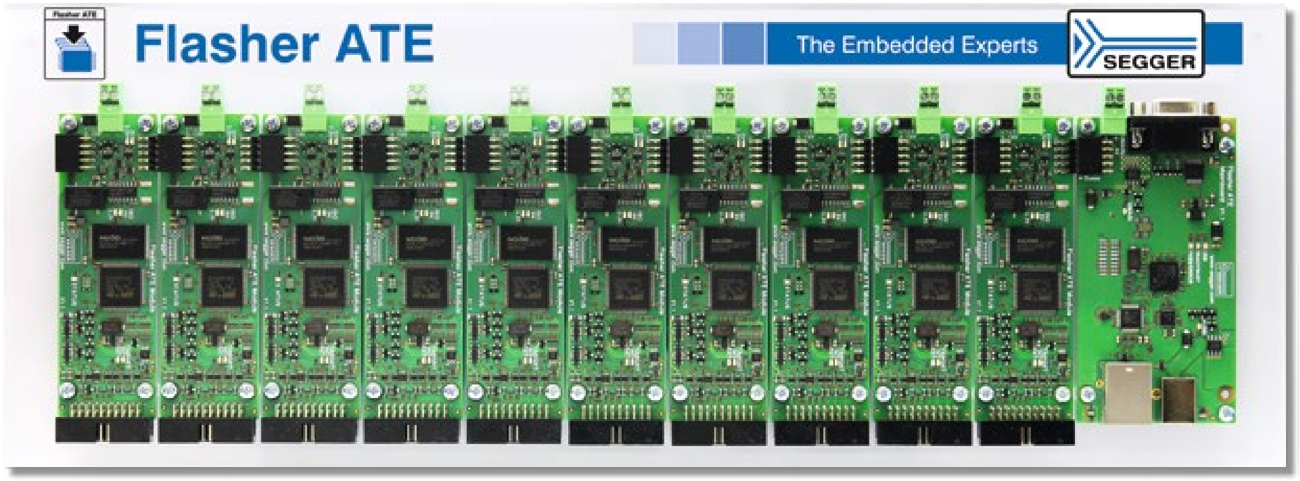
The top model Flasher PRO has all the features required for a mass production programmer. On top of the functionality of the Flasher ARM, it is capable of programming virtually any target system using the Universal Flash Loader. Commonly used target systems are Renesas RL78, Microchip PIC and directly connected SPI Flash or EEPROM memories.

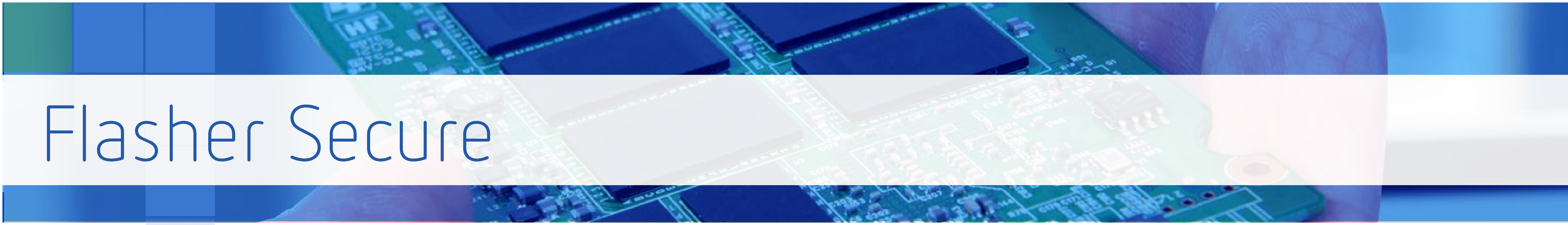
Gang Programmer Flasher ATE

The Flasher ATE is a modular gang programmer specialized for mass production setups. It is built from a mainboard and up to ten programming modules. The number of modules determine the number of PCBs that can be programmed at the same time (gang-programming). Each PCB interface can be configured individually. PCBs with multiple targets on board can have their targets be programmed in parallel.

Flasher ARM

Flasher ARM uses JTAG or SWD as target interface and supports programming of microcontrollers based on ARM cores such as: ARM7/9, Cortex-M0/M0+/M1/M3/M4/M7. Flasher ARM also supports programming of ARM11,





Flasher Secure

Cortex-A5/7/9/12/15/17, R4/R5 based microcontrollers with the help of a PC application. It supports the internal flash memory of microcontrollers as well as external flash memories. CFI-compliant parallel NOR flash memories are recognized automatically and can be programmed directly. QSPI NOR flash memory is also supported on many targets.

Flasher STM8

The Flasher STM8 connects with the target via SWIM interface. The target is optically isolated from the host side. The configuration and operation can be handled with the Flasher software for STM8. Option byte programming is supported.

Flasher 5 Pro

Flasher 5 Pro is the successor of the Flaser 5. It has 64MB of flash memory to store programming data and configuration and has a significantly higher performance. The programmer supports the following devices: Renesas R8C/M16C/R32C/M32C.

Flasher ST7

The Flasher ST7 supports the RS232 interface only and offers 512kB of flash memory to store programming data and configuration. Option byte programming is supported.

Flasher Secure

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

Secured Production at Contract Manufacturers

It is common to employ a contract manufacturer (CM) to mass-produce products. CMs have access to customer IP and large quantities of the components they are contracted to produce. Because of this, it’s essential that customers control both their IP and limit CM production, to prevent theft and to secure revenue. To combat these threats, Flasher Secure uses mutual authentication, authorization and confidentiality to secure your IP and production run. As an IP owner, you have full end-to-end control of your production chain.

Features

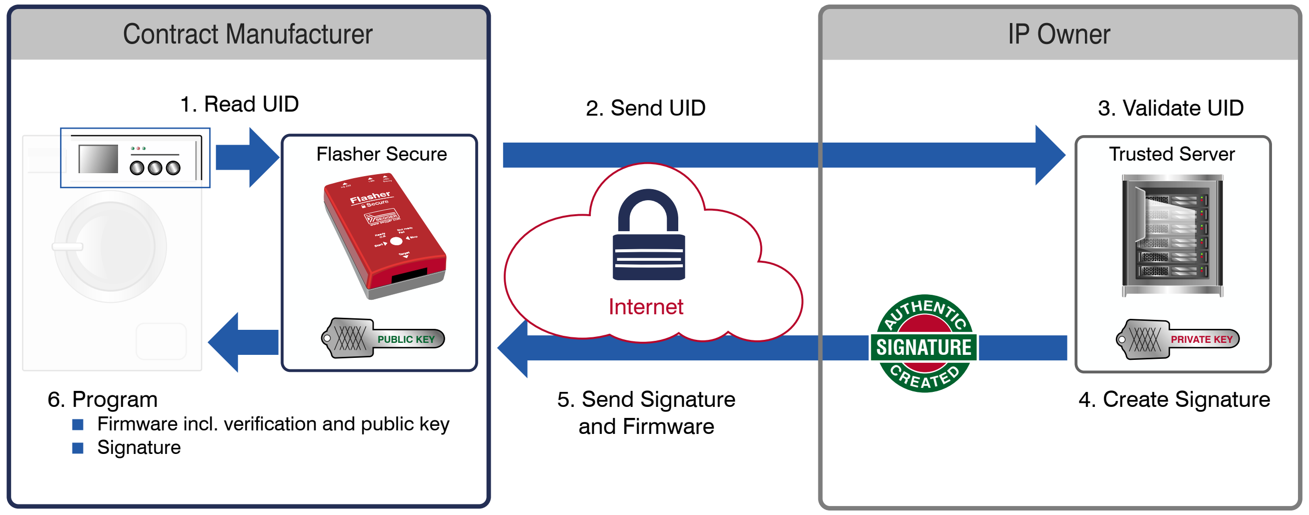
- Authenticated production
- Production volume control
- CM administration and setup portal
- Ultra fast programming
- Supports Cortex-M, RX, PPC
- Prevents production of counterfeit units
- No overhead in programming time
- List of programmed devices
- Remote maintenance

End-to-End Security

Flasher Secure supports vendor-specific trusted firmware features to ensure end-to-end encryption, authentication and confidentiality covering the whole process including the ‘last mile’.

Secured Firmware

Modern products carry a huge amount of intellectual property (IP). From the IP owners point of view most of the IP is located inside the firmware. IP owners want to protect their IP. The Flasher Secure system uses authentication algorithms to make sure, that only authorized boot loaders and firmware are used in the system. If one component is not genuine, the device will stop working. Simply copying the firmware and/or bootloader from one device to another is not possible anymore.



About SEGGER

Who we are

We at SEGGER Microcontroller are active in the industry of Embedded Systems.

SEGGER has over twenty-five years of experience in Embedded Systems, producing state-of-the-art RTOS and Software Libraries, and offering a full set of hardware tools (for development and production) and software tools.

Our motto is: "It simply works". This reflects in our products which are professional and easy-to-use in a complex, highly-technical, industry. Embedded Systems are a growing part of everyday life and we at SEGGER are passionate about being a part of it.

What we offer

SEGGER provides an RTOS plus communication and security software, so developers get a head start, benefiting from SEGGER's decades of experience in the industry. SEGGER's professional software libraries and tools for Embedded System development are designed for simple usage and are optimized for the requirements imposed by resource-constrained embedded systems.

The company also supports the entire development process, with affordable, high quality, flexible and easy-to-use tools.

The J-Link debug probes, with Ozone the accompanying debugger and performance analyzer, and Embedded Studio the powerful cross platform C/C++ IDE, are available to support developers in creating their own embedded systems.

Why SEGGER?

In short, SEGGER has a full set of tools for embedded systems, offers support through the entire development process, and has decades of experience as the Embedded Experts.

In addition, SEGGER software is not covered by an open-source or required-attribution license and can be integrated in any commercial or proprietary product, without the obligation to disclose the combined source.

Finally, SEGGER offers stability in an often volatile industry making SEGGER a very reliable partner for long-term relationships.

SEGGER Microcontroller GmbH
Ecolab-Allee 5
40789 Monheim am Rhein
Germany

www.segger.com
Tel: +49-2173-99312-0
info@segger.com

SEGGER Microcontroller Systems LLC
Boston area
101 Suffolk Lane
Gardner, MA 01440, USA

Tel.: +1-978-874-0299
us-east@segger.com

Silicon Valley
Milpitas, CA 95035, USA

Tel.: +1-408-767-4068
us-west@segger.com