VECTOR

MICROSAR Bootloader – Legacy Option

Product Catalogue

vector.com

Contents

1. Flash Bootloader Overview 4
1.1 Flash Bootloader System Architecture 4
1.2 Functions 4
1.3 Configuration 4
1.4 Scope of Delivery 5
1.5 Additional OEM and Third Party Modules 5
1.6 Multiprocessor Systems 5
1.7 64-bit Microcontroller Support 5

2. Modules and Add-Ons 6
2.1 Required Bus Systems and Specifications 6
2.2 Bootloader Adressing Mode 6
2.3 XCP Programming Within the Flash Bootloader 6
2.4 Bus Support Add-On CAN FD 6
2.5 CAN Bus Add-On J1939 6
2.6 Memory Support 6
2.6.1 Internal Data Flash Driver 6
2.6.2 External Flash Driver 6
2.7 NVM Data Handling 7
2.7.1 Integration Ea/Eep 7
2.7.2 Integration Fee 7
2.7.3 Integration Third-Party Fee Infineon 7
2.8 Security / Crypto 7
2.8.1 Security Measures 7
2.8.2 Secure Boot 8
2.8.3 Project Specific Security Access 8

3. Further Add-Ons and Options 9
3.1 Flash Bootloader Updater 9
3.2 Software Activation Manager 9
3.3 Bootmanager 9
3.4 Diagnostic Gateway Functionality 10
3.5 Downloadable Flash Driver 10

4. Multiprocessor 11
4.1 Multiprocessor Memory Driver 11
4.1.1 ISP Emulation 11
4.1.2 IPC Interface 11

©2026 Vector Informatik GmbH. Any distribution or copying prohibited.

4.2 Multiprocessor Communication Interface 12

5. Data Processing 13

5.1 Data Decompression 14

5.1.1 Data Decompression LZ77/LZSS 14

5.1.2 Data Decompression LZMA 14

5.2 Delta Update 14

5.3 Pipelined Programming 14

5.4 Pipelined Verification 15

6. Bootloader Integration Packages 16

7. vFlash Programming Tool 17

V4.0 [2026-07-20]

©2026 Vector Informatik GmbH. Any distribution or copying prohibited.

1. Flash Bootloader Overview

1.1 Flash Bootloader System Architecture

The Flash Bootloader embedded implementation is based on a modular architecture consisting of Bootloader Application Layer (for adapting the bootloader to the project specific requirements), OEM Download Manager (containing the bootloader runtime environment, state handling and diagnostic implementation), hardware-dependent parts (like the communication and memory drivers) and add-ons (for download performance optimizations and security extensions).

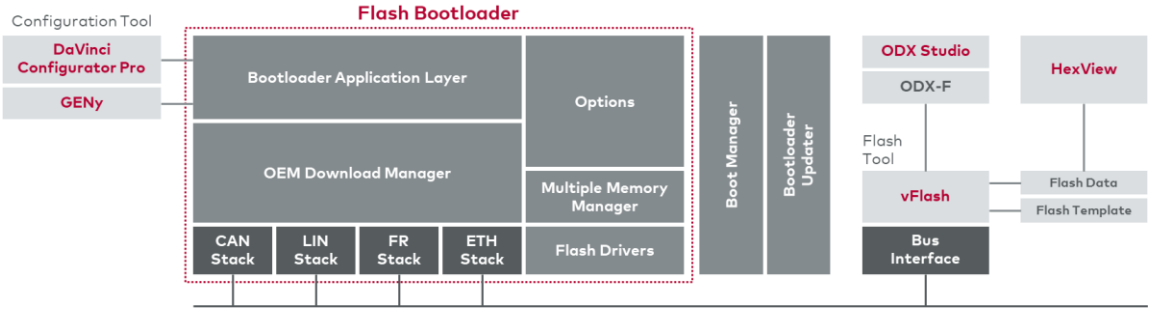


FIGURE 1: General Flash Bootloader System Architecture

In case the Flash Bootloader shall be updatable or in case Secure Boot is ordered, the Flash Bootloader is complemented by the Boot Manager component that performs sanity checks and starting of the according boot target.

In addition of the static source code of the Flash Bootloader, the bus-specific configuration, the security configuration, and the flash memory configuration are generated by the configuration tool.

The Flash Bootloader is delivered as a preconfigured product for a certain OEM and integrated on a specific microcontroller derivative. To reflect project and ECU-specific requirements, the configuration, bootloader application layer, startup code and linker command files need to be adapted.

1.2 Functions

The Flash Bootloader is stored in a dedicated ECU memory area and is started as the first software instance in the boot phase after a reset. It then checks whether a flash reprogramming request or valid application software exists. If the ECU shall be reprogrammed, the Flash Bootloader starts the reprogramming sequence. In case a Boot Manager is used, the Boot Manager takes the role to start either a valid application or start the Flash Bootloader to perform a software update.

Depending on the OEM requirements, a Flash Driver is dynamically downloaded by the Flash Tool into the ECU's RAM to perform the low-level flash memory access like erasing and writing portions of the flash. After the optional Flash Driver download, the Flash Bootloader erases the memory area that needs to be updated and performs the download. Following the data transfer, the Flash Bootloader performs the verification of the downloaded data – either based on CRC or digital signature.

In case the flash download sequence is interrupted, the Flash Bootloader ensures that reprogramming of the ECU is possible at any time.

1.3 Configuration

The project specific configuration of the Flash Bootloader is done either via the configuration tool GENy or DaVinci Configurator Pro. Please refer to the separate product information for additional information.

©2026 Vector Informatik GmbH. Any distribution or copying prohibited.

1.4 Scope of Delivery

The Flash Bootloader delivery includes:
> Bootloader as configurable C source code
> Flash driver for a specific hardware platform
> GENy or DaVinci Configurator Pro generator components
> OEM- specific vFlash Template file to resemble the OEM's download sequence
> HexView for preparing flash data and containers during development
> Technical reference documentation

Please note, that the licenses for the DaVinci Configurator Pro tool is not part of the delivery, and a license needs to be obtained for the configuration of the Flash Bootloader. Also, for the vFlash Programming Tool a separate license is needed.

The Bootloader is delivered in two parts. The first delivery comprises the software package of the Bootloader software. It does not contain a demonstration project of the Flash Bootloader. In a second delivery the demonstration project is provided as a Runtime Integration Service. It features example code that varies based on the Flash Bootloader content, showcasing application updates, FBL updates, memory and NvM data management, data processing, security measures and further optional OEM and Third-Party Modules.

1.5 Additional OEM and Third Party Modules

Vector offers further engineering support for the integration of additional OEM modules and third-party modules into the Flash Bootloader. The customer must provide these modules to Vector to ensure the proper integration of the modules. Additional charges may apply for this work.

1.6 Multiprocessor Systems

The Flash Bootloader can handle multiprocessor systems, where information about the project is exchanged and recorded in an Open Item List (OIL), e.g., interprocessor communication (IPC), synchronization at startup, switching between AP and Flash Bootloader, content of the configuration and about user callbacks (especially version changes and validation check).
Timely and mutual clarification of requirements is a prerequisite for the adherence of costs and delivery times. Especially changes in an advanced project phase can cause additional costs.

1.7 64-bit Microcontroller Support

Due to limitations given by underlying specifications (ASAM, ISO, AUTOSAR, etc.) not all address-based services can be used on 64-bit controllers without limitations. A typical limitation can be that only the lower half of the 64-bit address range can be accessed.

©2026 Vector Informatik GmbH. Any distribution or copying prohibited.

2. Modules and Add-Ons

The Vector Flash Bootloader solution offers a variety of additional functions to make software update more effective, secure, and comfortable.

2.1 Required Bus Systems and Specifications

The Vector Flash Bootloader supports all common bus systems used in a modern vehicle like CAN, CAN-FD, LIN, FlexRay and Ethernet.

Many OEMs define their own download specification. The Vector Bootloader supports a huge amount of these download specifications. In case there are no OEM specific requirements, Vector also offers an OEM-independent Flash Bootloader which is based on the ISO14229-1 download sequence. Please refer to supported OEMs for an up-to-date list of supported OEMs.

2.2 Bootloader Adressing Mode

For CAN-based communication, the Vector Bootloader provides the addressing modes Normal, Normal fixed and Extended.

2.3 XCP Programming Within the Flash Bootloader

This add-on extends the Flash Bootloader by XCP download support for development of OEM agnostic end of line programming. XCP download is supported in addition to the OEM diagnostic download protocol.

Since the software update via XCP does not consider any OEM-specific security extensions or validation mechanisms, the XCP programming add-on must be disabled for series production.

2.4 Bus Support Add-On CAN FD

This add-on enables the Flash Bootloader to support CAN with flexible data rates (FD). The Flash Bootloader supports both CAN-FD Modes, Mode 1 (CAN communication with flexible data rate) and Mode 2 (support of up to 64-byte payload).

2.5 CAN Bus Add-On J1939

The extension adds support for J1939 baud rate detection and J1939 network management (Nm) to the FBL.

2.6 Memory Support

By default, the Vector Flash Bootloader supports reprogramming of the internal code flash. For special use cases, additional memory drivers can be provided for reprogramming of internal data flash or external memory devices, e.g. serial flash devices.

2.6.1 Internal Data Flash Driver

This add-on is an extension to support software download into an internal data flash memory. It includes the embedded implementation of a hardware-specific data flash driver as well as the according configuration options.

2.6.2 External Flash Driver

By means of this add-on, the Flash Bootloader supports software download into external flash memory devices. Vector can provide flash drivers for parallel and serial flash devices. In case of serial flash devices, this add-on also includes the necessary low-level communication driver, e.g., a SPI driver.

©2026 Vector Informatik GmbH. Any distribution or copying prohibited.

2.7 NVM Data Handling

For some OEMs, the Flash Bootloader needs access to non-volatile memory to store logistic data like programming date, tester serial number, etc. and to persist status information. As an add-on, Vector can integrate a module into the bootloader to access non-volatile memory (NVM). If no NVM integration is requested, an interface to integrate your own non-volatile memory handling module will be provided.

The NVM data handling solution for your ECU is either based on EEPROM memory or Flash emulated EEPROM memory. Typically, the ECUs NVM solution is defined by the application and is to be integrated to the Flash Bootloader.

2.7.1 Integration Ea/Eep

The add-on Integration Ea/Eep contains the project-specific integration of the MICROSAR module Ea (EEPROM Abstraction) into the Flash Bootloader.
The delivery includes a demo integration of the Ea/Eep but not the Ea/Eep basic software modules.

2.7.2 Integration Fee

The add-on Integration Fee contains the project-specific integration of the MICROSAR module Fee (Flash EEPROM Emu-lation) into the Flash Bootloader.
The delivery includes a demo integration of the Fee but not the Fee basic software modules.

2.7.3 Integration Third-Party Fee Infineon

This add-on covers the integration of the third-party Fee from Infineon into the Flash Bootloader.

2.8 Security / Crypto

The security/crypto add-ons address the following topics:
> Authorization of diagnostic tester by symmetric/asymmetric or OEM-specific cryptographic algorithms
> Download Source Authentication using digital signature verification based on symmetric or asymmetric cryptographic algorithms.
> IP-protection by data encryption
> Data consistency by cyclic redundancy checks
> Integrity checks by secure/authenticated boot

2.8.1 Security Measures

Several security mechanisms are available, addressing different levels of security. They can provide protection of the software being downloaded and the download itself. To ensure that a download has been error free cyclic redundancy checking (CRC) is included. The Flashbootloader can perform a CRC check to verify the integrity of the written data, without the use of cryptographic material (formerly: Security Class DDD). By default, the standard CRC-32 algorithm is supported. Other CRC implementations can be offered on request.

To verify that the download has been error free and that the downloaded software originates from a legitimate source the following options are available:

Download Authentication Symmetric
Support of symmetric digital signature within the Flash Bootloader is added (formerly: Security Class C). The signature is based either on a HMAC or CMAC. The security component calculates the hash of the downloaded data and performs a signature verification to validate and check the authenticity of the downloaded data. A tool for creating test signatures during the development phase is part of this option.

Download Authentication Asymmetric
Support of asymmetric digital signature within the Flash Bootloader is added (formerly: Security Class CCC). The signature is based on an asymmetric cryptographic algorithm, e.g. RSA or ECC in combination with a non-reversible standard hash function and a public key. The security component calculates the hash of the downloaded data and performs a signature verification to validate and check the authenticity of the downloaded data. A tool for creating test signatures during the development phase is part of this option.

©2026 Vector Informatik GmbH. Any distribution or copying prohibited.

Download Authentication Asymmetric (PQC)
This option comprises the standard asymmetric cryptographic algorithms (e.g. RSA or ECC) mentioned above. In addition, post-quantum cryptography schemes (PQC) are provided to allow hardening the software download against attacks utilizing a quantum computer.

Download Encryption
This line item is an extension of the Flash Bootloader to support encrypted downloads (formerly Security Class AAA). The encrypted data is decrypted by the Flash Bootloader on the fly before the data is written to target memory. The data is decrypted by the AES-128 algorithm in CBC mode. The initialization vector can be part of the download as the first data block or is fixed to a constant. Data alignment to multiples of the AES block length must be done according to PKCS#5 or ISO7816-4. A development tool for data encryption is included.

2.8.2 Secure Boot

With the help of Secure Boot, the integrity of the application software (SW) and boot loader will be checked before startup. The check will be done either with hardware support or without hardware support, depending on chosen option.
Secure Boot needs to be based on some kind of hardware trust anchor (HTA) which can be a HSM or Read-only startup partition in combination with secure storage of symmetric key, if applicable.

Security Boot HW veHsm
Hardware-based Secure Boot using the Vector hardware security module (veHsm).
This option includes an interface for using HSM for check of Integrity and the integration of the Vector HSM solution (veHSM). The veHsm basic software module itself is not included and offered separately. The Flash Bootloader and the MICROSAR veHsm must be compatible with each other. If an existing veHsm delivery shall be used, please contact Vector in advance so that it can be determined if both software versions are compatible with each other. It may be required that veHsm needs to be updated to a compatible version.

Secure Boot HW SHE
Hardware-based Secure Boot using the Vector Secure hardware extension (She).
This option includes Interfaces for using Vector crypto driver for check of Integrity. and an AUTOSAR 4.3 compliant crypto driver.

Secure Boot 3rd Party
An interface for the integration of an AUTOSAR 4.3 compliant crypto driver is provided. A driver from alternative HTA vendor must be selected and integrated by you. The chosen driver needs to be based on SHE1.1 or EVITA HSM compliant device. If combined with Secure Boot (HW) variant, interfaces for using 3rd party crypto driver for check of integrity are provided.

Secure Boot (SW)
An interface allowing the usage of provided SW based security library is provided (CSM interface). This allows for both asymmetric (typically) and symmetric algorithms to be used.

2.8.3 Project Specific Security Access

This add-on covers the implementation of a project specific seed/key algorithm as specified in the questionnaire if there is no standard algorithm defined by the OEM. It includes the implementation of the seed/key algorithm in the Flash Bootloader and a DLL which can be used in vFlash. The specification of the seed/key algorithm must be provided if this add-on is ordered. Otherwise, the bootloader will include an interface to implement an ECU specific seed/key algorithm.

The source code of the Flash Bootloader implementation and a Visual Studio project with the source code of the DLL will be provided with the delivery.

©2026 Vector Informatik GmbH. Any distribution or copying prohibited.

3. Further Add-Ons and Options

Vector offers other build units besides the Flash Bootloader.

3.1 Flash Bootloader Updater

With the Flash Bootloader Updater an existing Flash Bootloader can be replaced. The Flash Bootloader Updater is downloaded just like a normal application and once started, replaces the existing Flash Bootloader by the new version. Depending on the hardware capabilities, this update process is not reset-safe. If a power failure or a reset occurs during the update process, the ECU cannot be reprogrammed anymore via the standard download process. A reset-safe implementation of the Flash Bootloader Updater process on all hardware platforms is only possible in combination with the Bootmanager add-on.

The figure below shows a conceptual overview of the update process:

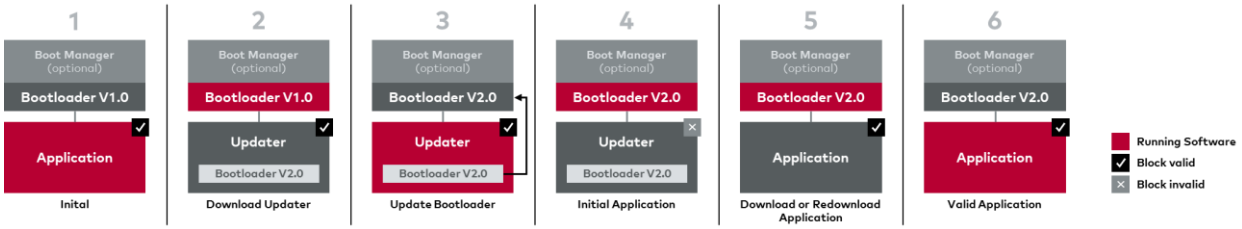


FIGURE 2: Overview of the Flash Bootloader Update Process

3.2 Software Activation Manager

The Software Activation Manager extends the Flash Bootloader by the functionality to perform a version switch to a software that was previously downloaded in the application context or in Flash Bootloader context.

It provides an A/B software version switch in case the corresponding microcontroller provides hardware support for this like internal shadow flash memory, memory mapping unit (MMU) or external dual bank memory.

Copying a background software update from an external to an internal flash to activate the software update is also supported. Please note that an additional flash driver might be needed to access external flash devices.

The add-on MultiPartitionProgramming is used for programming in active partition and rollback in the flash bootloader context.

3.3 Bootmanager

The Bootmanager is a dedicated basic software module to control the boot process to either start the application or the Flash Bootloader. The main objective of the Bootmanager is to provide routines to decide if application software or bootloader should be started. Additionally, it can provide a reset-safe implementation for the Flash Bootloader update process and can be used to speed up the ECU startup time for secure boot.

The Bootmanager add-on contains a framework of callback-functions, according to configuration options and a sample imple-men¬tation for consistency/validation checks. If the Bootmanager is used for the reset-safe bootloader update use-case or the secure boot use case, the Bootmanager is compiled and linked as a separate project and resides in a dedicated flash sector that cannot be reprogrammed by the Flash Bootloader.

©2026 Vector Informatik GmbH. Any distribution or copying prohibited.

3.4 Diagnostic Gateway Functionality

The gateway feature extends the Flash Bootloader by routing capabilities. It uses the PduR module to route functional diagnostic requests to sub-nets so that these sub-nets are kept in their current state (diagnostic session). The gateway feature is normally used while the gateway itself is updated.

3.5 Downloadable Flash Driver

The handling of downloadable flash drivers is supported by the MemAcc module, starting with MICROSSAR Release 35 and by the vMemAccM legacy module. This feature has to be supported by the flash driver as well. This needs to be checked with the driver vendor.

©2026 Vector Informatik GmbH. Any distribution or copying prohibited.

4. Multiprocessor

To support reprogramming of multiple microcontrollers, where only one microcontroller is directly connected to the vehicle communication network, the Multiprocessor add-on provides a solution to support such hardware configurations. Multiprocessor systems are characterized by hiding additional (slave/sub) microcontrollers from the diagnostic tester, i.e. the master/main microcontroller needs to transpose the diagnostic requests into an inter-processor communication protocol (IPC). The slave/sub microcontrollers are mapped as dedicated virtual address areas into the address space of the master/main microcontroller.

There are two solutions for to support multiple controller ECUs:
> Multiprocessor Memory Driver
> Multiprocessor Communication Interface

4.1 Multiprocessor Memory Driver

The Multiprocessor Memory Driver supports the following hardware configurations:



FIGURE 3: Single Update Interface, Single Slave/Sub Microcontroller



FIGURE 4: Single Update Interface, Multiple Slave/Sub Microcontroller

With the Multiprocessor Memory Driver all data processing operations like decryption, decompression, etc. are handled by the master/main microcontroller. Per default, integrity or authenticity checks are also handled by the master/main microcontroller but can be also implemented on the slave/sub microcontroller.

Within the Multiprocessor Memory Driver approach, there are two variants on how to reprogram additional microcontrollers:
> Emulation of an in-system reprogramming protocol (ISP) on master side
> Interfacing the slave microcontroller by means of a serial communication interface using an IPC

4.1.1 ISP Emulation

Since there is no software needed to implement the reprogramming sequence on the slave/sub microcontroller, this solution can be also applied for slave/sub microcontrollers with very limited RAM/ROM resources.

There are some preconditions to be considered, in case the slave/sub microcontroller shall be reprogrammed by an in-system reprogramming protocol:
> The master/main microcontroller needs to be capable to control the reset input of the slave/sub microcontroller
> All relevant I/O lines for the ISP need to be connected between the microcontrollers according to the technical documentation of the microcontroller silicon vendor

The ISP Emulation approach is available only for selected slave/sub microcontrollers. Please contact Vector regarding the availability for a specific microcontroller.

4.1.2 IPC Interface

For the IPC Interface based solution, the master/main microcontroller Flash Bootloader is extended by additional software layers for the inter-processor communication which connect to the HIS memory driver API.

©2026 Vector Informatik GmbH. Any distribution or copying prohibited.

The delivery for this add-on contains the high-level software layers for the master/main microcontroller:
> Memory Layer (McmpMem) which implements the lower layer for the HIS memory driver interface
> Session Layer (McmpSl) for timeout monitoring
> Transport Protocol Layer (McmpTp)

The add-on also contains template components for the hardware specific IPC layers:
> Data Link Layer (McmpDl) which interfaces the Transport Protocol Layer with the Low-Level Driver (LLD)

In case the multiprocessor communication interface is based on CAN, ETH, FR or LIN, Vector can also provide the Data Link Layer as well as the Low-Level Driver. Please ask Vector for the availability for any other multiprocessor communication interfaces like SPI, UART, etc.

The following illustration shows the layered architecture for the multiprocessor specific components:

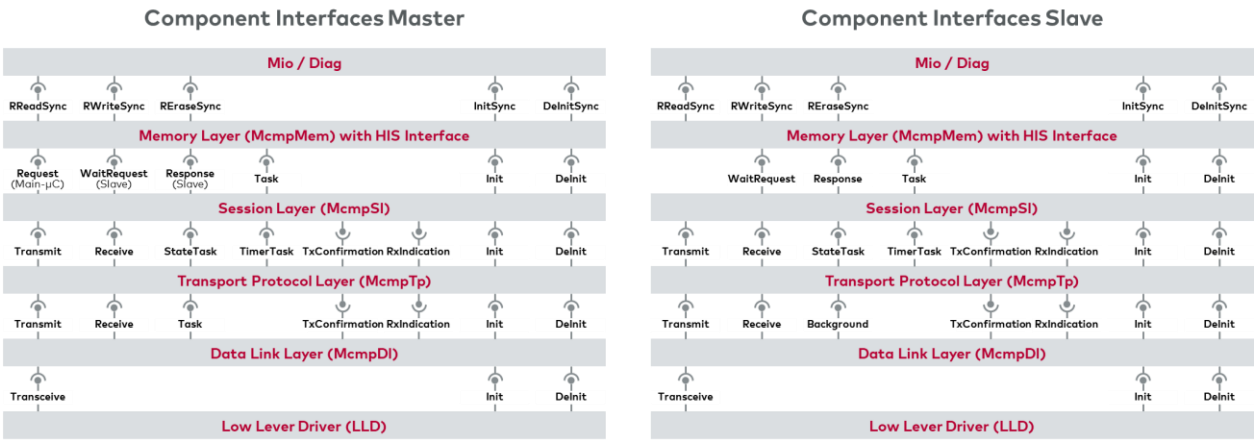


FIGURE 5: Multiprocessor Memory Driver Software Layers for Master and Slave Microcontroller

With the Multiprocessor Memory Driver Slave add-on, also the high-level IPC components for the slave microcontroller are delivered. Depending on availability, Vector can also provide a flash driver for the slave/sub microcontroller part as well as the Data Link Layer and the Low-Level Driver. In case no Data Link Layer/Low-Lever Driver is available, template components for the hardware specific IPC layers are delivered.

4.2 Multiprocessor Communication Interface

The Multiprocessor Communication Interface contains the extension of the Flash Bootloader for interfacing one (or more) microcontroller by means of an inter processor communication protocol (IPC). The Multiprocessor Communication Interface extension supports the following hardware configurations:



FIGURE 6: Single Update Interface, Single Slave/Sub Microcontroller

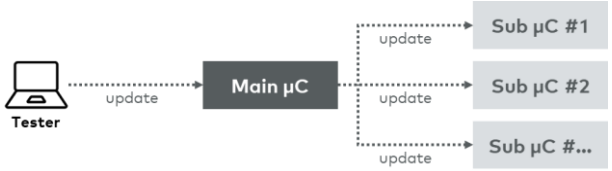


FIGURE 7: Single Update Interface, Multiple Slave/Sub Microcontroller

©2026 Vector Informatik GmbH. Any distribution or copying prohibited.



FIGURE 8: Single Update Interface, Cascaded Slave/Sub Microcontroller

The Multiprocessor Communication Interface add-on provides a high-level interface for the master/main micro¬controller to access the raw data that is transferred by the diagnostic tester. Therefore, all data processing operations like decryption and decompression must be implemented on the slave/sub microcontroller as well. The interface provides the following APIs:

> Erase indication to trigger erasing memory areas
> Block/segment start/end indications for consistency checking and update verification
> Data transfer indication for forwarding data to the slave/sub microcontroller
> Verification indication for triggering the verification of the downloaded data

The figure below shows a sequence diagram of the relevant API calls:

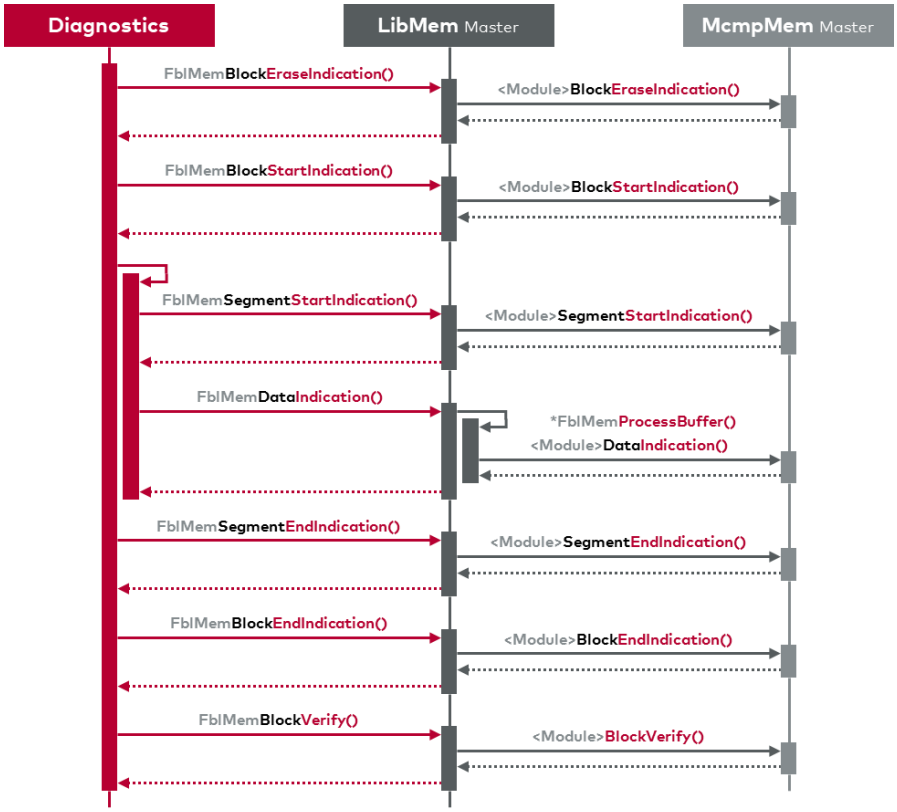


FIGURE 9: Call Sequence of Multiprocessor Communication Interface APIs

5. Data Processing

The Data Processing add-ons provide download optimizations by either reducing the data transfer time using data compression algorithms or delta download or by parallelizing data processing by means of Pipelined Programming and Pipelined Verification.

Some of the Data Processing add-ons can be also combined to provide the best download performance, e.g. Pipelined Programming, Pipelined Verification and Data Decompression. Please note that these add-ons have a significant impact on the RAM resource requirements and might not be applicable for all microcontrollers. Please ask Vector for resource estimations.

©2026 Vector Informatik GmbH. Any distribution or copying prohibited.

The download performance improvement of these add-ons highly depends on the type of download data, the used bus system and the performance of the microcontroller.

5.1 Data Decompression

5.1.1 Data Decompression LZ77/LZSS

The add-ons Data Decompression LZ77 and LZSS extends the Flash Bootloader to support the download of compressed data. The data compression method is based on a LZ77/LZSS compression algorithm and is optimized for low runtime requirements and small footprint. The LZSS method is an optimized variant based on LZ77.

This add-on includes the embedded implementation of the data decompression algorithm for the Flash Bootloader as well as a PC-based data compression tool.

5.1.2 Data Decompression LZMA

This add-on extends the Flash Bootloader to support the download of compressed data. The data compression method is based on a LZMA compression algorithm. Please note that the LZMA decompression is mainly targeted for high-performance 32-bit microcontrollers with appropriate RAM resources.

This add-on includes the embedded implementation of the data decompression algorithm for the Flash Bootloader as well as a PC-based data compression tool.

5.2 Delta Update

Instead of downloading and programming the complete target image, only the differences of two software/data set versions are transmitted to the Flash Bootloader. Depending on the extent of the change record, this may significantly reduce download times.

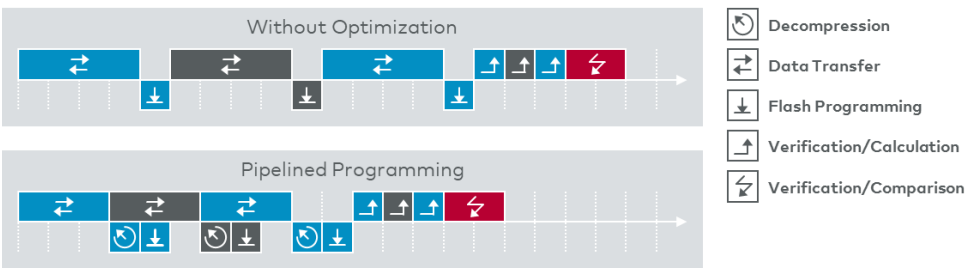
The Vector Flash Bootloader supports integration of the Vector delta library, that effectively extends the Bootloader to support the download of delta images. Two approaches to apply the delta information are supported:

Monolithic
The delta image is downloaded into a temporary memory region (typically a reserved area of the internal program flash). After the download is completed, the information in the delta image is applied to the source image in memory in a single processing step.

Stream-based
The information in the delta image is applied to the source image in memory, while the delta image is downloaded. A development tool for creating delta images is part of the Vector Delta Download solution. To achieve a significant download time reduction, the delta images need to be compressed, e.g. with a LZMA compression algorithm. As the compression module is not part of the Vector Delta Download solution it needs to be ordered separately.

5.3 Pipelined Programming

The add-on Pipelined Programming extends the Flash Bootloader to program and receive data in parallel. The received data is written into a RAM buffer and a positive response is sent immediately after the complete reception of the diagnostic service. Afterwards, the data is written into the flash memory while the next data can be received. This provides a virtual parallelization of the flash programming and reception.



©2026 Vector Informatik GmbH. Any distribution or copying prohibited.

FIGURE 10: Pipelined Programming

5.4 Pipelined Verification

This add-on covers the performance optimization for verification of the programmed flash data in parallel to the download. The verification is performed section by section on the written data.

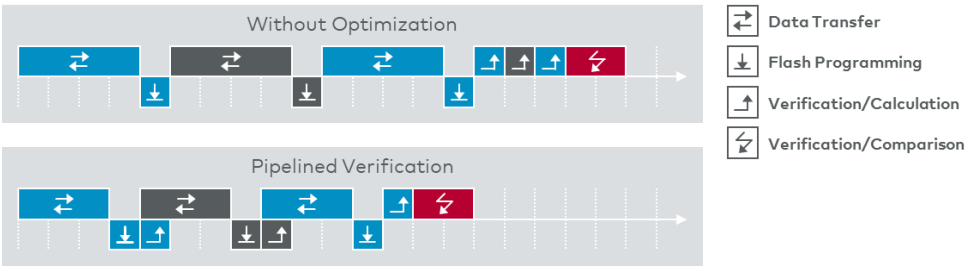


FIGURE 11: Pipelined Verification

©2026 Vector Informatik GmbH. Any distribution or copying prohibited.

6. Bootloader Integration Packages

Package	Text
Integration CAN Transceiver	This add-on covers the integration of a MICROSAR 4 CAN transceiver in the Flash Bootloader. A MICROSAR4 CAN transceiver driver is a precondition for this option.
Integration SBC	This add-on contains the integration of a MICROSAR4 System Basis Chip (SBC) driver supporting different peripheral devices such as transceivers and watchdogs. A MICROSAR4 SBC driver is a precondition for this option.
Integration EthSwtDrvExt	This add-on covers the integration of a MICROSAR 4 Ethernet switch driver in the Flash Bootloader. A MICROSAR4 Ethernet switch driver is a precondition for this option.
Integration Wdg	This add-on contains the integration of a MICROSAR4 watchdog (Wdg) driver. A MICROSAR4 Wdg driver is a precondition for this option.
Customer Hardware	This add-on contains the integration and tests of the Flash Bootloader on customer hardware (e.g. A-sample) instead of an evaluation board.
Identity Manager	The Identity Manager covers the support of multiple diagnostic connections aka “postbuild selectable”. It enables the support of multiple diagnostic connections that can be selected in the startup phase of the Flash Bootloader.
Third Party MCAL Integration FBL	The Third Party MCAL Integration FBL covers the integration of necessary third party MCAL modules provided by the customer like Fls, Spi, Cry. Preconditions for the MCAL integration are the
Flash Bootloader License Extension Brand	The Flash Bootloader License Extension Brand covers the right to use the purchased Flash Bootloader which was licensed for one brand also for a further brand. A precondition of the license extension is an agreement of the initially targeted OEM that the software is allowed to be used for a further brand. The customer must provide evidence of this agreement to Vectors in advance. This option does not include a delivery.
Flash Bootloader License Extension OEM	The Flash Bootloader License Extension OEM covers the right to use the purchased Flash Bootloader which was licensed for one OEM also for a further OEM. A precondition of the license extension is an agreement of the initially targeted OEM that the software is allowed to be used for a further OEM. The customer must provide evidence of this agreement to Vectors in advance. This is not required for Vector SLP. This option does not include a delivery.
FBL Adaptation to OEM Spec	On project basis, the Flash Bootloader is adjusted to an OEM specification. Adjustments of the vFlash template are also included. This option does not include the implementation of a security access algorithm.
Additional Version Management	This add-on covers the additional version tracking of an existing Flash Bootloader delivery. A new license tracking number is created with the set of the corresponding component versions. Based on this, new deliveries can be performed without affecting the issue reporting of the previous Flash Bootloader delivery. Additional Version Management covers the following services:

©2026 Vector Informatik GmbH. Any distribution or copying prohibited.

7. vFlash Programming Tool

vFlash is a very easy-to-use and extensible tool for reprogramming ECUs and enables efficient flashing in the lab, at programming stations, on the lab vehicle or in the vehicle. Thanks to its plug-in concept, vFlash can be used for all projects and already supports more than 140 different flash specifications from over 80 vehicle manufacturers. vFlash works seamlessly with the Vector Flash Bootloader: All Vector Flash Bootloaders come with a corresponding vFlash template that mirrors the OEM-specific download sequence. vFlash is aimed at all users at automotive OEMs and suppliers whose tasks include (re)programming ECUs.

> Software download via CAN, CAN-FD, LIN, FlexRay or Ethernet (DoIP or SoAd)
> Easily share a vFlash project and all relevant data with others – in a single package (.vflashpack)
> High data transfer rate
> Support of different protocols and flash sequences/flash specifications by plugin concept (vFlash Templates)
> Direct “native” programming of data in Intel hex, Motorola-S and binary format or alternatively programming based on container formats like ODX-F and OEM-specific formats
> Download of compressed and encrypted data
> Interactive download via the GUI as well as automated flashing over a programming interface (C, C# API)
> Validation of the flash bootloader in the ECU in combination with CANoe.DiVa (via CAN, FlexRay, DoIP, LIN)
> Simultaneous flashing of multiple ECUs, each with an individual communication channel
> Remote flashing of ECUs from any location with vFlash Remote

©2026 Vector Informatik GmbH. Any distribution or copying prohibited.

MORE INFORMATION
Visit our website for:

> News
> Products
> Demo software
> Support
> Training and workshops
> Contact addresses

vector.com

©2026 Vector Informatik GmbH. Any distribution or copying prohibited.