Bootloader Validation Strategies

Version 1.0
2012-03-21
Application Note AN-ISC-8-1143

Author(s)	Achim Strobelt

Restrictions	Customer confidential - Vector decides

Abstract	This application note describes application validation strategies used in Vector Bootloaders.

Table of Contents

1.0 Introduction

Today, most automotive ECUs support software download to be able to update the application software or data at any time. The Flash Bootloader from Vector provides a convenient way to fulfill this task according to the specification of the respective car manufacturer. While the process of downloading new software into the ECU is greatly simplified by the Bootloader, the ECU supplier has to apply an algorithm, which guarantees integrity and authenticity of the downloaded software/data.

1.1 Scope

Checking downloaded software/data usually takes some time to reveal the validity state. This is why the validity state is determined by the Bootloader directly after the download and stored within a special memory region called “Validation Area”. There are various kinds of methods for handling the validity information, the most common ones are setting a flag in EEPROM or storing a special pattern in flash memory. During start-up of the ECU, the Bootloader checks the validity information to decide whether to start the application software or to stay in the Bootloader. This approach allows short start-up times, which is a common requirement for automotive ECUs.

This application note will mainly focus on the mechanisms, which can be used to perform the actual validation check. A method which can be used to store the validity information in the flash memory is described in chapter 4.3 Mask/Pattern Mechanism.

2.0 Validation Basics

ECU software often includes several modules (e.g. the actual application and a calibration module). At least one of these modules is mandatory, and modules can be optional to add functionality or data for special use cases.

To determine the validity of an application, several preconditions have to be fulfilled:

- All mandatory modules have to be present.
- All downloaded modules (mandatory and optional) have to be downloaded correctly.
- The modules have to be compatible.

The validation of an ECU's software can be seen as a two-step validation. Depending on the Bootloader implementation, it is performed in one or two steps.

2.1 Module Validation

Module validation should ensure that the downloaded module has been downloaded correctly. Additionally, a check if the downloaded module is permitted to run on the ECU can be performed.

The module validation is performed after a module has been downloaded, e.g. with a check routine request for each logical block or during a request transfer exit request. The validation result should be stored in non-volatile memory to avoid executing the complete validation sequence every time the validation state has to be checked.

2.1.1 Download Integrity

Before a downloaded module is used by the ECU, the Bootloader has to ensure that the file has been downloaded correctly. This includes checking if all data has been written to flash correctly and that the data was transferred correctly to the ECU.

Data written to flash is read back by the flash driver immediately after writing and compared to the diagnostic buffer. This comparison ensures that the intended data was flashed.

If a checksum is available in the downloaded data or was sent by a special service, this checksum can be used to cover the complete data transfer from tester to the ECU.

Note
If a CRC32 checksum is used to check the downloaded data, but no signature check is done, the module validation is called security class "DDD" or security class "None".

2.1.2 Download Authenticity

Some ECUs require that only software or data from a legitimate source is used on the ECU (e.g. in security relevant ECUs or as tuning protection). The authentication is normally done by calculation of a cryptographic signature over the downloaded data. The signature can be either supplied by the OEM or the ECU supplier.

Note
Depending on the OEM, several security classes for data authentication are defined. The most common security classes are security class “C” with a symmetric signature calculation and security class “CCC” with an asymmetric calculation.

2.2 Application Validation

The application validation step has to ensure that all required modules are present on the ECU and that the downloaded modules are compatible with each other and with the used hardware/Bootloader combination.

The Bootloader has to decide if the ECU’s application is started based on the application validation step. The application validation step can either be done at the end of the download sequence or during ECU start-up. If the application validity is determined at the end of the download sequence, the validity information has to be stored within a non-volatile memory (see next chapter).

Note
If the application validity is checked on every start-up of the ECU, the checks will be based on the module validity flags and the startup time will be increased. On the other side, there is no additional non-volatile memory needed to store the application validity.

3.0 Validation Approaches

There are several approaches to check the integrity and authenticity of downloaded data. This chapter describes some variants.

3.1 Pattern

The check for one or more patterns located in the memory area of downloaded data is a simple approach to check if the written data is the expected data. The validation function checks if the patterns are found at a known location. If the patterns are found, the validation function assumes the download has been correctly programmed.

Usually, two patterns are used with this approach: one at the beginning and one at the end of the download.

Caution
With a pattern check, not the complete data written is checked. Errors outside the patterns won’t be found.

3.2 Checksum/CRC

A checksum/CRC check uses a checksum read from the downloaded data or sent by an additional service for the integrity check. The checksum is calculated over the complete downloaded data of a module. If the checksum is correct, the integrity of a complete module assumed.

Caution
Checksum algorithms like CRC32 don't provide an authenticity check because there's no secret parameter involved during the checksum calculation.

3.3 Signatures

Signature calculation algorithms combine a hash calculation and cryptographic routines to ensure the downloaded software's integrity and authenticity.

Note
Some Bootloader configurations skip the integrity check if a signature calculation is done.

3.4 Validation Structure

A validation structure can be used to embed validity information into the downloaded data. The validation structure consists of block information including checksums/signatures which can be compared against the checksums/signatures that are calculated over the actual memory content. A validation structure can be used if no specific diagnostic service for transmission of the checksum/signature values is specified by the OEM.

4.0 Storage of Validation Information

4.1 EEPROM

If an EEPROM is used to store the validation information, some bytes have to be reserved in the EEPROM's memory. Data in EEPROM is writeable without needing to erase larger blocks and can be written very often compared to flash memory. On the other side, storing the data in EEPROM requires loading an additional driver to access the EEPROM.

Note
During the ECU's start-up, the EEPROM driver has to be loaded twice: First, in the Bootloader to check if the application is valid, and a second time in the application.

4.2 Emulated EEPROM

Emulated EEPROMs basically are used as the same way than EEPROMs. They use software modules like EepM or FEE to achieve EEPROM like behavior in flash memory.

Caution
EEPROM emulations are often more complicated than EEPROM drivers. Especially the start-up times can be bigger than the start-up times of an EEPROM driver and the access times can depend on the amount of previously written data.

4.3 Presence Patterns

Presence patterns (also called Mask/Pattern mechanism) use the memory inside each module to store the validity information. A pattern value is used to indicate that a module is valid and a mask value is used to invalidate an already written pattern.

As presence patterns are stored inside the module's memory, they normally are memory mapped and can be accessed without loading additional drivers. This can reduce the start-up time compared to the EEPROM/ Emulated EEPROM variants, but some memory from the application/data area has to be reserved for presence patterns.

Caution
If a memory area with error recognition or correction is used to store the presence pattern, special care has to be taken to make sure that the software can handle the error. If the ECU reacts e.g. with a reset if an error occurs during reading the pattern, the ECU will be locked because the presence pattern will be read during every start-up.

5.0 Contacts

Germany
and all countries not named below:

Vector Informatik GmbH
Ingersheimer Str. 24
70499 Stuttgart
GERMANY
Phone: +49 711-80670-0
Fax: +49 711-80670-111
E-mail: info@de.vector.com

France, Belgium, Luxemburg:

Vector France SAS
168 Boulevard Camélinat
92240 Malakoff
FRANCE
Phone: +33 1 42 31 40 00
Fax: +33 1 42 31 40 09
E-mail: information@fr.vector.com

Sweden, Denmark, Norway, Finland, Iceland:

VecScan AB
Theres Svenssons Gata 9
41755 Göteborg
SWEDEN
Phone: +46 31 764 76 00
Fax: +46 31 764 76 19
E-mail: info@se.vector.com

United Kingdom, Ireland:

Vector GB Ltd.
Rhodium, Central Boulevard
Blythe Valley Park
Solihull, Birmingham
West Midlands B90 8AS
UNITED KINGDOM
Phone: +44 121 50681-50
Fax: +44 121 50681-69

E-mail: info@uk.vector.com

China:

Vector Automotive Technology (Shanghai) Co., Ltd.
Sunyoung Center
Room 1701, No.398 Jiangsu Road
Changning District
Shanghai 200050
P.R. CHINA
Phone: +86 21 6432 5353
Fax: +86 21 6432 5308
E-mail: info@cn.vector.com

India:

Vector Informatik India Pvt. Ltd.
4/1/1/1, Sutar Icon, Sus Road,
Pashan, Pune - 411 021
INDIA

Phone: +91 20 2587 2023
Fax: +91 20 2587 2025

E-mail: info@in.vector.com

USA, Canada, Mexico:

Vector CANtech, Inc.
39500 Orchard Hill Place, Suite 550
Novi, MI 48375
USA

Phone: +1 248 449 9290
Fax: +1 248 449 9704
E-mail: info@us.vector.com

Japan:

Vector Japan Co. Ltd.
Tennozu Yusen Bldg. 16F
2-2-20 Higashi-shinagawa,
Shinagawa-ku,
Tokyo 140-0002
JAPAN
Phone: +81 3 5769 7800
Fax: +81 3 5769 6975
E-mail: info@jp.vector.com

Korea:

Vector Korea IT Inc.
#1406, Mario Tower,
222-12 Guro-dong, Guro-gu
Seoul, 152-848
REPUBLIC OF KOREA

Phone: +82 2 807 0600
Fax: +82 2 807 0601
E-mail: info@kr.vector.com