Application Note - Boot Loader Solutions for LEON and NOEL-V Processors

Technical and Application Note

Product	
Project	
Document number	PMD-0085
Version	1.0
Status	Approved
Approved date	2026-10-30
Approved by	Martin Rönnbäck, HW SH
Contract	
Deliverable	
Template	TEM-0003, 9.0

Frontgrade Gaisler AB
Kungsgatan 12
411 19 Göteborg, Sweden

TABLE OF CONTENTS

1 Introduction 2
2 In Scope 3
3 Reference documents 3
4 Definitions and Abbreviations 3
5 Overview 3
6 GRBOOT 4
6.1 NR-field initialization considerations 4
7 GR712RC Boot Software 4
7.1 NR-field initialization considerations 5
8 MKPROM2 5
8.1 NR-field initialization considerations 5
9 Custom Boot Loader 5
9.1 NR-field initialization considerations 6
10 Identifying NR fields 6
11 Change Record 7

1 INTRODUCTION

This document provides a short overview of Frontgrade Gaisler bootloader products and describes the handling of

registers with NR fields. Information is also provided about how NR fields should be handled in custom bootloaders.

The work has been performed by Frontgrade Gaisler AB, Göteborg, Sweden.

2 IN SCOPE

This document is applicable to the following:

- IP cores in the GRLIB IP library, including the LEON and NOEL-V processors
- Custom boot loader solutions and the boot-loader solutions from Frontgrade Gaisler: GRBOOT, GR712RC Boot Software, and MKPROM2
- Components from Frontgrade Gaisler: GR740, GR712, GR716.

3 REFERENCE DOCUMENTS

[RD1] GR740-UM-DS-2-10.pdf

[RD2] https://www.gaisler.com/products/grboot

[RD3] https://www.gaisler.com/products/mkprom2

[RD4] https://www.gaisler.com/products/gr712rc-boot-software

4 DEFINITIONS AND ABBREVIATIONS

ESA	European Space Agency
ESTEC	European Space Research and Technology Centre
TBC	To Be Confirmed
TBD	To Be Defined
NR	Non Reset
RMW	Read Modify Write
SW	Software

5 OVERVIEW

Boot loaders perform the initial software configuration required to bring a processor system into a known state and transfer control to the application. Depending on the target hardware and mission requirements, projects can use an existing boot-loader solution or develop a custom implementation.

Frontgrade Gaisler provides the following boot-loader solutions for LEON-based or NOEL-V based systems.

- GRBOOT [RD2]
- MKPROM [RD3]
- GR712RC Boot SW [RD4]

Custom boot loaders are also an option when the available solutions do not meet specific hardware, storage, security, or mission requirements. The supported processors and features differ between solutions and must be verified for the target platform.

An important consideration when initializing a processor system is the handling of registers containing non-reset (NR) fields. These fields may retain their previous values across a SoC reset. Consequently, a register must be explicitly initialized before software reads its value and performs a read-modify-write (RMW) operation. Otherwise, a subsequent register write may unintentionally preserve stale NR-field values.

GRBOOT, GR712RC Boot Software, and MKPROM2 address this concern for the registers they initialize by writing initialization values without first performing RMW operations. However, these boot loaders do not necessarily initialize every peripheral. Applications and custom boot loaders must therefore ensure that registers containing NR fields are initialized correctly before they are read or modified.

The following sections describe the available boot-loader approaches and the relevant considerations for NR-field initialization.

6 GRBOOT

GRBOOT is a boot-software solution from Frontgrade Gaisler for payload and onboard computers based on the GR765, GR740, GR712RC, and UT700 processors. It is intended for flight applications and implements the ESA flight-computer start-up specification, SAVOIR-GS-002. Its development follows ECSS-E-ST-40C and ECSS-Q-ST-80C, tailored to software criticality category B.

GRBOOT consists of four main components:

1. Hardware and software initialization.
2. Built-in self-tests.
3. A standby extension point for maintenance functionality.
4. An application loader.

The application loader selects one of two application images, verifies the selected image, and can fall back to the alternative image if verification fails. GRBOOT supports several deployment configurations, including a standard boot-loader configuration, a configuration with a custom maintenance or standby mode running from ROM, and a configuration with a small application incorporated into the boot image.

The software supports both symmetric multiprocessing (SMP) and asymmetric multiprocessing (AMP) configurations. It prepares the system for operating systems and runtime environments such as RTEMS, VxWorks, Linux, BCC, and PikeOS. Results from self-tests of the CPU, caches, ROM, and external memory are recorded in a boot report that can be accessed by the application.

Boot-loader storage is separated from application storage, allowing application updates without modifying the boot software. Application images use an ELF-like format that supports in-flight patching and optional compression. Images can be loaded from memory-mapped storage, including PROM, flash, and MRAM, or from SPI flash. The S25FL256S, S79FL256S, and S79FL512S SPI flash devices are supported directly; other SPI flash devices may require custom extensions.

Available ports include the GR-CPCI-GR740, GR712RC development board, and UT700-LEAP. Verification includes automated unit tests on target hardware and in the TSIM3 simulator, code-coverage measurement in TSIM3, and a validation test suite executed on hardware.

6.1 NR-field initialization considerations

GRBOOT initializes CPU and memory-controller registers containing NR fields without relying on RMW operations for their initial configuration. Relevant examples include the PSR, WIM, TBR, and CCTRL (L1 Cache Control Register) on LEON3/LEON4 processors.

After the initial configuration has established known register values, subsequent reconfiguration may use RMW operations where appropriate. For example, the boot software may enable the L1 cache after completing the cache self-test.

GRBOOT does not necessarily initialize all peripheral registers. Examples of peripherals that may require application-level initialization include GPIO and GRETH interfaces. If these peripherals contain NR fields, their registers must be initialized before software reads them and performs RMW operations.

GRBOOT provides APIs for peripheral-register initialization that avoid RMW operations, supporting correct initialization of registers containing NR fields. If used, initialization of mentioned peripheral cores will be performed by GRBOOT during boot.

7 GR712RC BOOT SOFTWARE

GR712RC Boot SW is a flight system software suite targeting GR712RC systems. The purpose of GR712RC Boot SW is to provide initialization, maintenance and application loading functionality to a payload or on-board computer. GR712RC Boot SW is an implementation of the ESA Requirements document Flight Computer Initialization Sequence (TEC-SWS/10-373). The software is developed in accordance with European Space Agency software engineering standards ECSS-E-ST-40C and ECSS-Q-ST-80C, tailored software criticality category B, reviewed by ESA and third party (ISV&V). GR712RC Boot SW is used in 8 scientific instruments in the ESA JUICE satellite.

The software supports both symmetric multiprocessing (SMP) and asymmetric multiprocessing (AMP) configurations. It prepares the system for operating systems and runtime environments such as RTEMS, VxWorks, bare metal, etc. Results from self-tests of the CPU, caches, ROM, and external memory are recorded in a boot report that can be accessed by the application.

GR712RC Boot SW is legacy software, meaning it is no longer actively developed or maintained. GRBOOT replaces GR712RC Boot SW and provides similar solutions and functions.

7.1 NR-field initialization considerations

GR712RC Boot SW initializes CPU and memory-controller registers containing NR fields without relying on RMW operations for their initial configuration. Relevant examples include the PSR, WIM, TBR, and CCTRL (L1 Cache Control Register) on LEON3/LEON4 processors.

After the initial configuration has established known register values, subsequent reconfiguration may use RMW operations where appropriate. For example, the boot software may enable the L1 cache after completing the cache self-test.

GR712RC Boot SW does not initialize all peripheral registers. Examples of peripherals that require application-level initialization include GPIO and GRETH interfaces. If these peripherals contain NR fields, their registers must be initialized before software reads them and performs RMW operations.

8 MKPROM2

MKPROM2 is a production tool from Frontgrade Gaisler that converts a LEON program built to execute from RAM into an image that can boot from ROM. The resulting image contains start-up code, an application loader, and the application itself.

The application is compressed using a modified LZSS algorithm, typically reducing its size by approximately half. During start-up, the boot code initializes the CPU and FPU registers, configures the memory controller, initializes the UARTs and timer, and initializes EDAC-protected memory. It then decompresses the application from ROM into RAM and transfers control to it, with the stack pointer set to the top of RAM.

MKPROM2 supports ERC32 and LEON2, LEON3, LEON4, and LEON5 systems. The generated image can be stored in ROM, PROM, MRAM, or flash, enabling stand-alone operation.

MKPROM2 is available as a development tool for Linux and Windows. Version 2.0.69 was released in December 2022, and the binary archives include the source code. The associated documentation includes the user's manual and the GR-CPCI-GR740 Quick Start Guide, which describes the use of MKPROM2 on the GR740, including memory-controller and L2-cache configuration.

Frontgrade Gaisler states that MKPROM2 was not developed for flight use and does not provide the safety mechanisms typically required for such applications. GRBOOT is therefore the recommended solution for flight applications requiring the corresponding safety and robustness features.

MKPROM2 is a simple option for development and non-flight systems, whereas GRBOOT provides additional functionality intended for flight applications, including self-tests, application-image integrity checks, and fallback between redundant images.

8.1 NR-field initialization considerations

MKPROM2 initializes CPU and memory-controller registers containing NR fields without performing RMW operations as part of their initial configuration. This prevents values retained in NR fields across SoC resets from being unintentionally carried into the initialization values.

As with other boot loaders, the scope of initialization is limited to the registers and peripherals configured by the boot software. Any registers containing NR fields that are left uninitialized must be handled by subsequent software before being read and modified through RMW operations.

9 CUSTOM BOOT LOADER

A custom boot loader can be developed for a specific board or mission when an existing solution does not satisfy the system requirements. It is the first software executed by the processor after power-on or reset, typically from ROM, PROM, MRAM, or flash. Its primary purpose is to initialize the hardware into a known state and transfer control to the main application.

A typical custom boot loader performs the following operations:

- Initializes processor registers, the FPU, and caches.
- Configures the memory controller and system clocks.

- Initializes essential peripherals, such as UARTs and timers.
- Clears or scrubs EDAC-protected memory to prevent uninitialized memory contents from causing misleading error reports.
- Locates the application image in non-volatile storage.
- Verifies the image and copies or decompresses it into RAM.
- Configures the stack and required execution environment before transferring control to the application.

Custom boot loaders are commonly used when the target system has an unusual memory layout, requires unsupported boot or storage devices, or needs a particular initialization sequence or timing configuration. Another motivation is supporting specific maintenance modes.

For flight and other safety-critical systems, additional requirements may include:

- Built-in self-tests, with results recorded for the application.
- Application-image integrity checks, using CRCs, checksums, or other appropriate verification mechanisms.
- Redundant application images with automatic fallback if an image is found to be corrupt.
- Watchdog configuration and handling.
- A minimal safe mode to support recovery and onboard updates.

Keeping the boot software separate from application storage can allow application updates without changing the boot code in ROM. However, a custom boot loader requires substantial development, verification, and documentation effort, particularly when standards such as ECSS-E-ST-40C or SAVOIR-GS-002 apply. Consequently, projects may benefit from starting with an existing solution and adapting it to the target hardware rather than implementing all functionality from scratch.

9.1 NR-field initialization considerations

When developing a custom boot loader, particular care must be taken to initialize registers containing NR fields.

Because NR fields may retain their values across SoC resets, the boot loader must establish a known register state before reading the register or performing an RMW operation. The initial register configuration should therefore be performed using a direct write of the intended initialization value, without first reading the register to preserve its existing contents.

Failing to do this can result in stale NR-field values being unintentionally preserved by an RMW operation. The resulting register configuration may differ from the intended start-up state and can affect subsequent hardware initialization.

The recommended sequence is:

1. Identify the registers containing NR fields that require initialization. Refer section 10 to know how to identify NR register fields.
2. Write the required initialization values directly to those registers, without first reading them or using RMW operations.
3. Perform any required read-back or verification after initialization, where supported and appropriate.
4. Use RMW operations for subsequent configuration changes only after the register has been initialized and the operation is appropriate for its register semantics.

This approach should be applied to all relevant processor, memory-controller, and peripheral registers, rather than only to registers initialized by the boot-loader framework. Particular attention is required for peripherals that are not initialized automatically by the boot loader.

10 IDENTIFYING NR FIELDS

The registers and fields that are not reset are identified in the device user manual.

For example, the GR740, the register description convention is defined in section 1.11 of the GR740 Data Sheet and User's Manual [RD1]. Each register table contains a reset-value row beneath the field names, and Table 4 of [RD1] defines the value "NR" as "Field not reset". Section 6.2.18 further states that, apart from the registers listed in Table 38, processor registers maintain their value or are undefined after system reset. NR fields are present in processor registers (for example the PIL field of %psr, %wim, and the DF/IF fields of the cache control register) as well as in peripheral registers such as the GRGPIO interrupt polarity and edge registers, the UART control register, and several GRETH_GBIT registers.

Similarly, the respective device datasheet shall be referred to identify the NR register fields while developing or using the bootloader solutions for a device.

When developing boot software, the register tables of every core that is used should be reviewed for NR entries, and the field type should be taken into account. For example, write-to-clear (wc) fields must be written with one to establish a

known state. For other devices, the corresponding device user manual should be consulted, since the set of NR fields depends on the device and core configuration

11 CHANGE RECORD

VERSION	DATE	SECTION / PAGE	DESCRIPTION
1.0	30-10-2026	All	First revision
1.1	30-10-2026	1,2,3,5,10	Updated review comments

This document includes Frontgrade Gaisler's confidential and or proprietary information that shall not be duplicated, used, or disclosed, in whole or in part, except for the limited purpose for which it has been furnished.

Copyright © Frontgrade Gaisler AB