# R-IN32M3 Series

## Serial Bootloader

R01AN2580EU0100
Rev. 1.00
Feb 9, 2015

## Introduction

This serial terminal controlled bootloader provides a programming and control mode to let a user download or update the non-volatile program storage area with a new user image.

The host and the R-IN32M3 are connected via a UART serial link. This enables connection to the board from a PC using a virtual COM port over USB to enable text-based interactive control.

The user image is stored in the board's external flash memory. After reflash, the bootloader will reboot and copy the user image to RAM and execute it from reset.

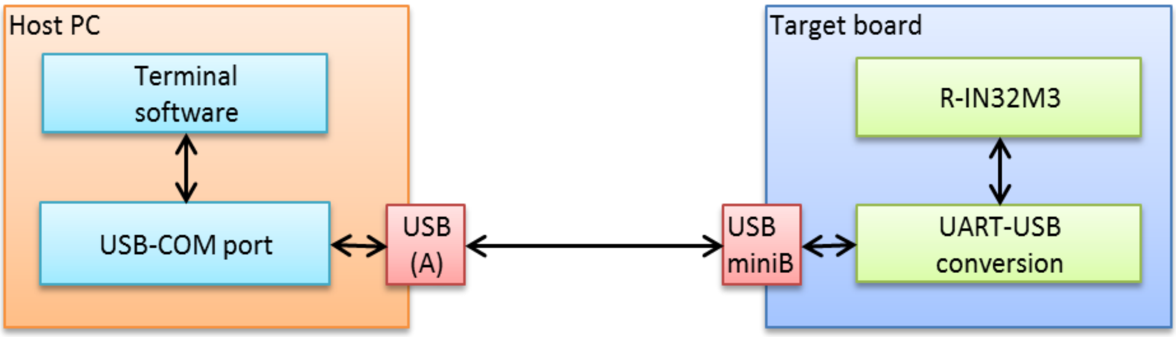


**Figure 1. The host PC and the R-IN32M3 are connected via a virtual UART serial communication link. A PC can thereby send bootloader commands and download a user image.**

## Target Device

**R-IN32M3**. A Renesas Industrial Networking LSI.

Applicable boards are the IAR R-IN32M3-EC KickStart Kit, for which the code is configured by default, the R-IN32M3-EC made by Tessera Technologies, the Tessera R-IN32M3-CL, or, any other board with the same external memory design. Note that on some boards the RP10 pin that controls boot behavior cannot conveniently be controlled by a switch.

## Related Documents

1. IAR R-IN32 Starter Kit documentation: http://www.iar.com/Products/Starter-kits--debug-probes, IAR Systems AB
2. TS-R-IN32M3-EC User Manual, (R-IN32M3-EC Evaluation Board). Tessera Tech. LTD.
3. TS-R-IN32M3-CL User Manual, (R-IN32M3-CL Evaluation Board). Tessera Tech. LTD.
4. r18uz0013ej, R-IN32M3-EC StarterKit Setup Procedure. Renesas Electronics Corp.
5. r18uz0024ej, R-IN32M3-CL StarterKit Setup Procedure. Renesas Electronics Corp.
6. r18uz0007ej, R-IN32M3 Series Peripherals Function Manual. Renesas Electronics Corp.

## Contents

# 1. Source Code

With this application note comes source code that should be merged into the R-IN32 Driver/Middleware standard source code tree *r-in32m3_samplesoft*. The latest R-IN32 middleware can be found at

https://www.renesas.com/en-us/products/factory-automation/multi-protocol-communication.html

Click on the Sample Software tab, and then "Driver/Middleware".

Merge the code into the middleware tree at directory level *..\r-in32m3_samplesoft\Device\Renesas\RIN32M3\Source\Project*.

- Include paths needed to compile the bootloader project:

```
$PROJ_DIR$/..
$PROJ_DIR$/../../../../Include
$PROJ_DIR$/../../../../../../../CMSIS/Include
$PROJ_DIR$/../../../Middleware/flash
$PROJ_DIR$/../../../Middleware/sflash
```

- Defined symbols

  Define **OSLESS** if not using an RTOS.

  Define **RIN32M3_CL** if using a "CL" type board for CC-Link IE.

## 2. R-IN32 Startup Basics

All R-IN32 example code, including the serial bootloader and the user image, are stored in non-volatile memory but executed from RAM. This is because execution directly from flash would be slower.

The source code projects exist in different configurations depending on where they are mapped to permanent memory, and thus from where the image must be copied at startup.

The project mode selected for a project must match the board setting as explained below - except for user images where the project mode must be set to RAM Debug. This is because the user image mapping in the RAM Debug projects are what the bootloader anticipates. That is, the bootloader has macros determining where the user image is expected to reside. (The user project mappings in turn are determined by the by the linker configuration file.) This can be changed. See 5.4.

### 2.1 Project Modes

The bootloader can be set to the project modes “Serial Flash Boot”, “NOR Boot”, or RAM Debug. For the user project, use *only* RAM Debug mode.

In section 3.2 we will study how the code boots from memory depending on project mode selection.

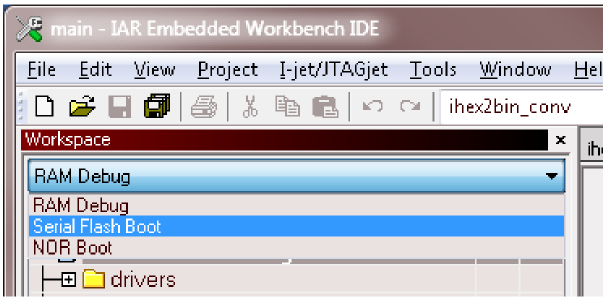

main - IAR Embedded Workbench IDE
File Edit View Project I-jet/JTAGjet Tools Window Hel
ihex2bin_conv
Workspace
RAM Debug
RAM Debug
Serial Flash Boot
NOR Boot
drivers

**Figure 2. Use Serial or NOR Boot mode for the bootloader project. For the user project use *only* RAM Debug mode since this matches the bootloader memory region expectations of the user image.**

#### 2.1.1 NOR Boot Mode

R-IN32 initialization code (Flashloader) copies the serial bootloader startup code from the NOR flash (parallel flash, U2 on the R-IN32M3-EC board). The code is copied to the RAM instruction memory at address 0 for execution.

Do not use this mode for the user image, as explained in 5.1.

#### 2.1.2 Serial Flash Boot Mode

R-IN32 initialization code copies the serial bootloader startup code from serial flash (U3 on R-IN32M3-EC board). The code is copied to the RAM instruction memory at address 0 for execution.

Do not use this mode for the user image, as explained in 5.1.

#### 2.1.3 RAM Debug Mode

Boot from RAM. That is, boot from volatile, non-permanent storage. Typically only used by a toolchain when developing and debugging. For the bootloader the Nor Boot and Serial Flash can be used when debugging.

Use this mode for the user project as explained in 5.1. Do not use this mode for the bootloader. The bootloader itself will be expanded to RAM. At a user image download, the bootloader will be overwritten

### 2.2 Boot Stages

Below are the stages wherewith execution proceeds from reset for the R-IN32. This is only given in order to understand the complete boot sequence. The most important aspect here is the second sequence; that Data RAM (at 0x2000 0000) is reserved for the bootloader.

- Stage 1. At RESET the R-IN32 copies initialization code to the instruction RAM area that starts at address 0. The initialization code is taken from storage depending on project mode and the BOOT pin settings. The initialization code comes with the toolchain, and the correct settings for the external memory type is added automatically when selecting project mode. For a parallel flash project for example, the init code (e.g.

*copy_init* and *data_init* functions) will be copied from the parallel flash. Figure 4 illustrates this copying of initialization code.

- Stage 2. The copied bootloader init code is executed. This code in turn copies the actual bootloader from serial or parallel flash to the destination addresses as can be seen in the bootloader map file (INIT TABLE). The destination address for the bootloader code is data RAM at 0x2000 0000. This area is therefore reserved for the bootloader. The area is determined by the bootloader project linker configuration file.
- Stage 3. The bootloader is executed.

Note that the pin setting (RP10) that determines whether the user image will be executed is not part of these initial stages, but comes into play after the last stage.

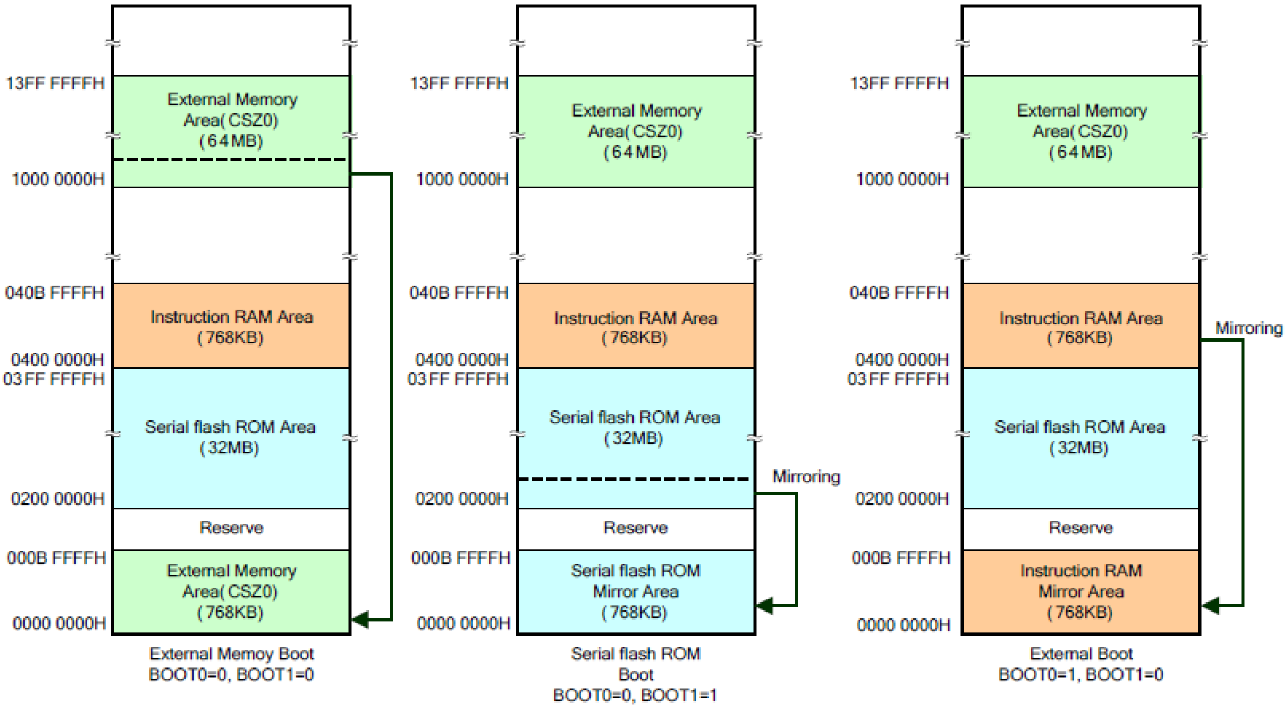

13FF FFFFH
External Memory Area( CSZ0) ( 64 MB)
1000 0000H
040B FFFFH
Instruction RAM Area ( 768KB)
0400 0000H
03 FF FFFFH
Serial flash ROM Area ( 32MB)
0200 0000H
Reserve
000B FFFFH
External Memory Area( CSZ0) ( 768KB)
0000 0000H
External Memoy Boot BOOT0=0, BOOT1=0
Serial flash ROM Mirror Area ( 768KB)
Mirroring
Serial flash ROM Boot BOOT0=0, BOOT1=1
Instruction RAM Mirror Area ( 768KB)
Mirroring
External Boot BOOT0=1, BOOT1=0

**Figure 3. A general view of how the R-IN32 copies code to RAM at reset. The code that is copied is initialization code that comes with all projects - except for the RAM Debug projects (which are used for the user images) - and is built in with the toolchain. From left; NOR Boot mode, Serial Boot mode, and RAM mode. The instruction RAM area at address 0 is always used for this purpose and is therefore reserved. See also [6] in Related Documents. A more detailed view of all memory is shown in Figure 14.**

# 3. Bootloader Download and Startup

*This guide does not cover how to set up IAR Embedded Workbench for ARM, and how to download the code and use the debugger for a Renesas R-IN32M3 board.*

The serial bootloader must first be downloaded with the toolchain. Not until after the bootloader is installed is it possible to update a user main program via a serial port connection.

At boot, the bootloader is copied into RAM for execution. (The user image is copied into RAM later by the bootloader.) The R-IN32 board has different storage locations from where to copy an image; parallel (NOR) flash, serial flash, or even RAM memory when debugging rom-less boards. As discussed in this section, project mode and boot pin settings must match.

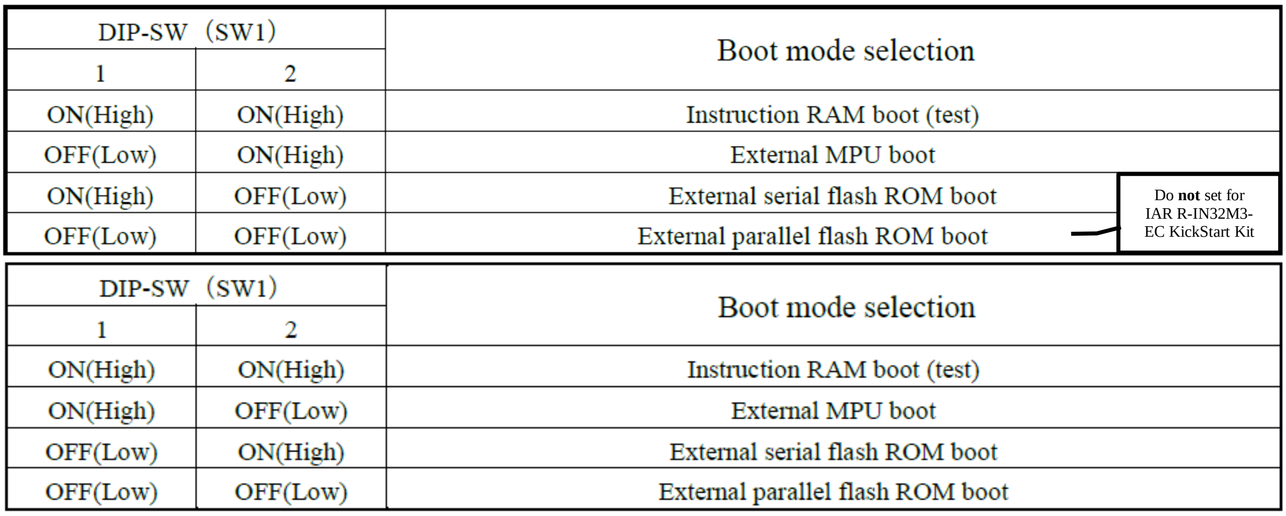
## 3.1 Settings

### 3.1.1 Board

See your R-IN32 board user manual under Related Documents above for specific jumper settings etc.

### 3.1.2 Boot Pins

The R-IN32M3-EC has two terminals BOOT0 and BOOT1. These determine from where the bootloader will be fetched. The voltage setting of these pins must match the bootloader project mode. For the R-IN32M3-EC for example, these pins are set via the dip switches “MODE_SW” (SW1). Boot mode settings are is reprinted in Figure 4  for convenience. (Related Documents [4, 5].)



| DIP-SW（SW1） | | Boot mode selection |
|---|---|---|
| 1 | 2 | |
| ON(High) | ON(High) | Instruction RAM boot (test) |
| OFF(Low) | ON(High) | External MPU boot |
| ON(High) | OFF(Low) | External serial flash ROM boot |
| OFF(Low) | OFF(Low) | External parallel flash ROM boot |

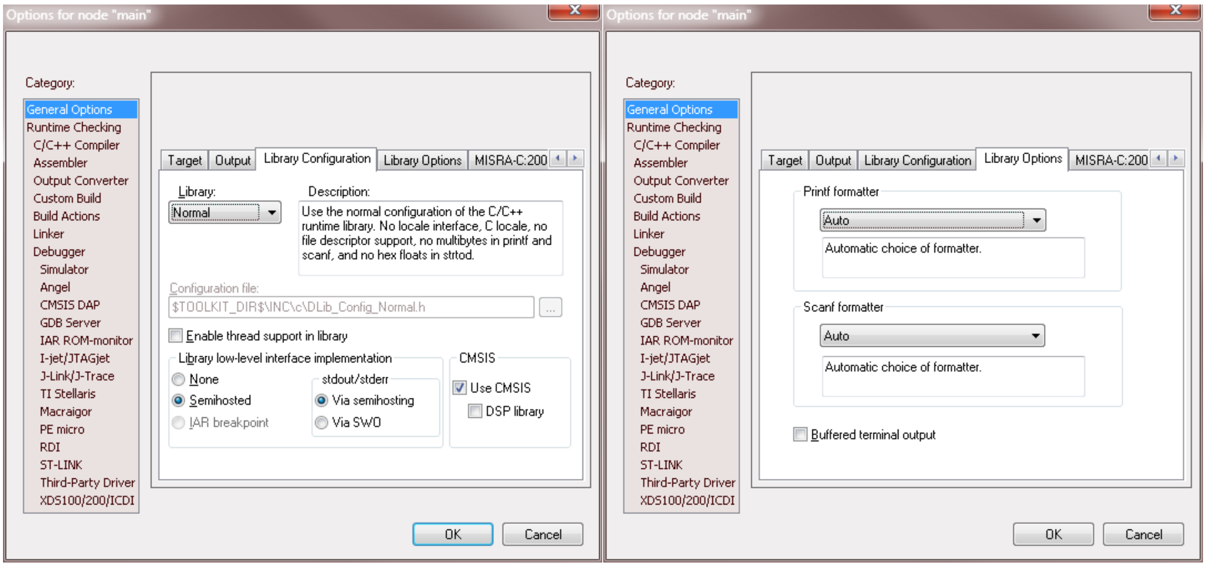
Do **not** set for IAR R-IN32M3-EC KickStart Kit

| DIP-SW（SW1） | | Boot mode selection |
|---|---|---|
| 1 | 2 | |
| ON(High) | ON(High) | Instruction RAM boot (test) |
| ON(High) | OFF(Low) | External MPU boot |
| OFF(Low) | ON(High) | External serial flash ROM boot |
| OFF(Low) | OFF(Low) | External parallel flash ROM boot |

**Figure 4. Boot mode selection tables for the Tessera R-IN32-EC board and the IAR R-IN32M3-EC KickStart Kit at top, and the –CL board at the bottom. Select a corresponding project mode as shown in Figure 2. Ignore “External MPU boot” in this application note.**

### 3.1.3 Project

To make sure that bootloader command-line strings that are to be sent to the serial port do indeed go to hardware and not a debug terminal, use the settings as shown in Figure 5 for IAR Embedded Workbench.



Library: Normal

Description: Use the normal configuration of the C/C++ runtime library. No locale interface, C locale, no file descriptor support, no multibytes in printf and scanf, and no hex floats in strtod.

Configuration file: $TOOLKIT_DIR$\INC\c\DLib_Config_Normal.h

□ Enable thread support in library

Library low-level interface implementation: ○ None ● Semihosted ○ IAR breakpoint

stdout/stderr: ● Via semihosting ○ Via SWO

CMSIS: ☑ Use CMSIS □ DSP library

Printf formatter: Auto — Automatic choice of formatter.

Scanf formatter: Auto — Automatic choice of formatter.

□ Buffered terminal output

**Figure 5. To make sure that bootloader serial output data will properly arrive at the serial port terminal when using the IAR toolchain, use semihosting and *un*check "Buffered terminal output".**

To run the bootloader on the target board, connect the debugger, power the board, compile, link, download and run the binary. Section 6 describes a hands-on session where the bootloader is run and a user image is downloaded.

## 3.2 Bootloader Startup Behavior

After startup of the serial bootloader, its further behavior depends on *both* the value of a hardware port and on whether there is user flash code at the expected address.

### 3.2.1 Hardware Port Setting

If pin RP10 (RealTime port 10) is at the high voltage level (switch is ON) the bootloader program is always executed regardless of the state of the user image. This corresponds to SW6-1 in the OFF position on the Tessera EC board.

### 3.2.2 User Flash Image Content

(a) **Serial flash storage**

If the 16 bytes at the start address of serial flash, that is at address 0x0202 0000 (SFLASH_BASE_ADDRESS + MAIN_PROGRAM_ADDRESS), are all 0xFF, the bootloader is executed. Otherwise execution is started at the main user code program. That is, the normal product application is run.

(b) **NOR (parallel) flash**

If the 16 bytes at the start address of parallel flash, at 0x1002 0000, are all 0xFF, the bootloader is run. Otherwise the main user code program is executed. Observe that using parallel flash is not available when using the IAR KSK.

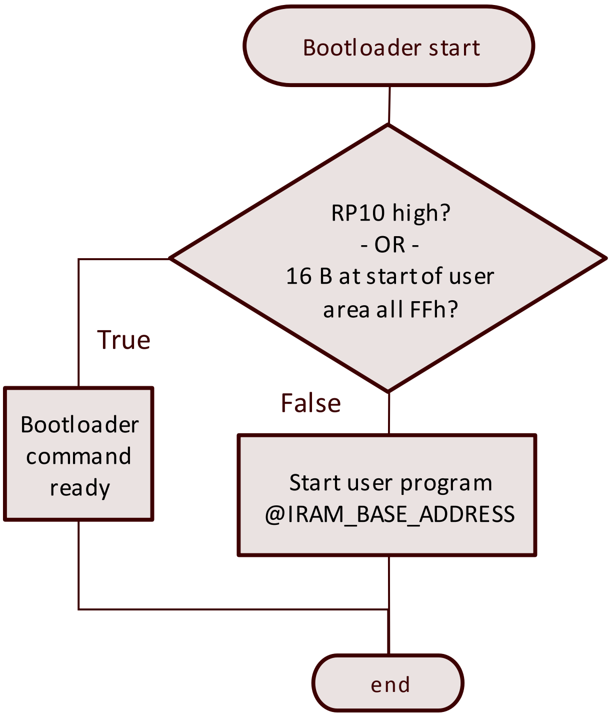


**Figure 6. After a reset, bootloader startup behavior determines whether to execute the user image. The user image is executed from the reset vector if it deemed to be OK and if RP10 is low. Otherwise the bootloader is started and enters the command mode.**

### 3.2.3 Faulty Image Protection

If there is code in the user image area as determined above, the user application should be checked further for correctness before execution. The above user execution startup scheme is provided as example only. The user should provide for added protection against execution of a faulty image by means of an image checksum e.g. within a user image header field with a fixed address known by the bootloader. A user header can contain fields such as image checksum, version number, etc. The header field should be added to the user image, and the address to it and checksum calculation routine etc added to the bootloader.

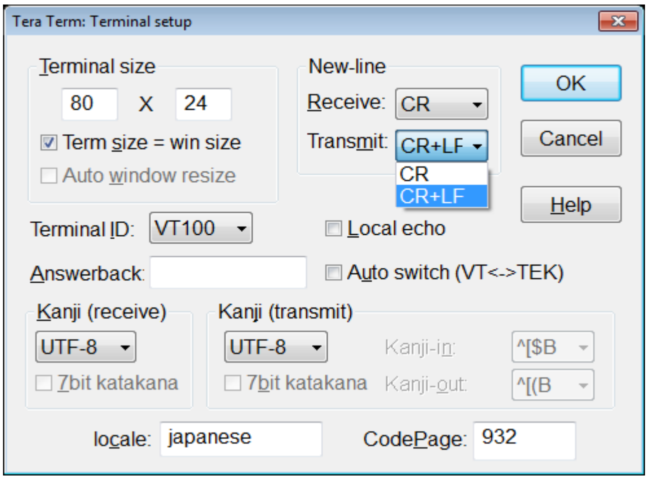
# 4. Serial Terminal

## 4.1 PC Settings

Control of the serial boot loader is done using a command set from over a COM port connection to the R-IN32 board. Connect a USB A to Mini-B cable between a PC and the board's UART (USB) connector. Start a serial terminal program and select the virtual COM port number that belongs to the R-IN32. The settings used for the PC's serial terminal program in order to be able to access the R-IN32 are shown in Figure 7 and Figure 8.

| Item | Setting |
|---|---|
| Baud rate | 115200 bps |
| Data bit length | 8 bit |
| Parity bit | None |
| Stop bit length | 1 bit |
| Flow control | Hardware control |
| Terminal setting | TX: CR+LF |

**Figure 7. COM port settings for the PC serial terminal program.**



Tera Term: Terminal setup

Terminal size: 80 X 24

☑ Term size = win size

□ Auto window resize

New-line

Receive: CR

Transmit: CR+LF

Terminal ID: VT100

□ Local echo

Answerback:

□ Auto switch (VT<->TEK)

Kanji (receive): UTF-8

□ 7bit katakana

Kanji (transmit): UTF-8

□ 7bit katakana

Kanji-in: ^[$B

Kanji-out: ^[(B

locale: japanese

CodePage: 932

**Figure 8. Note the CR+LF setting needed in the PC serial terminal settings.**

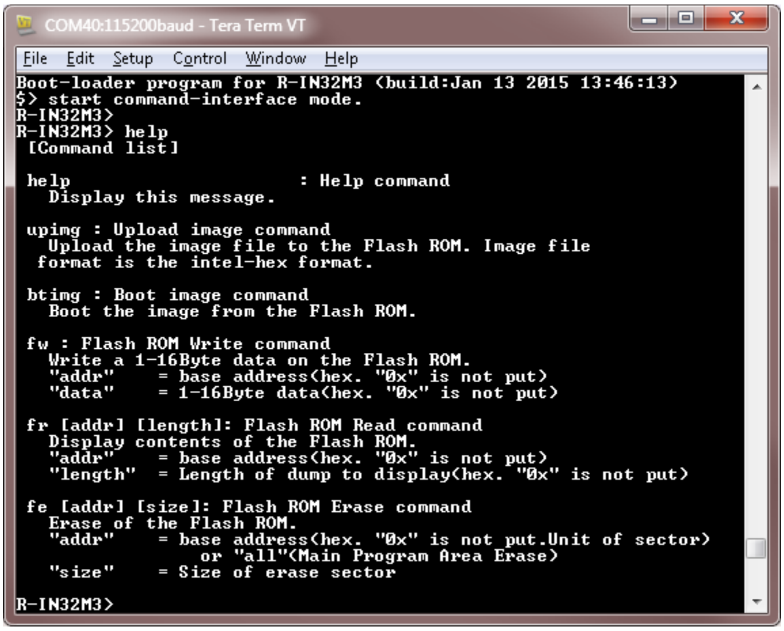
If the connection is correct, you should see a prompt in your serial terminal program:

```
R-IN32M3>
```

You can find the COM-port number for the PC by e.g. opening Device Manager.

Type <help>. A help text showing the bootloader's control commands should be shown.

```
R-IN32M3>help
```


COM40:115200baud - Tera Term VT
File Edit Setup Control Window Help

```
Boot-loader program for R-IN32M3 (build:Jan 13 2015 13:46:13)
$> start command-interface mode.
R-IN32M3>
R-IN32M3> help
 [Command list]

 help                     : Help command
   Display this message.

 upimg : Upload image command
   Upload the image file to the Flash ROM. Image file
  format is the intel-hex format.

 btimg : Boot image command
   Boot the image from the Flash ROM.

 fw : Flash ROM Write command
   Write a 1-16Byte data on the Flash ROM.
   "addr"    = base address(hex. "0x" is not put)
   "data"    = 1-16Byte data(hex. "0x" is not put)

 fr [addr] [length]: Flash ROM Read command
   Display contents of the Flash ROM.
   "addr"    = base address(hex. "0x" is not put)
   "length"  = Length of dump to display(hex. "0x" is not put)

 fe [addr] [size]: Flash ROM Erase command
   Erase of the Flash ROM.
   "addr"    = base address(hex. "0x" is not put.Unit of sector)
                 or "all"(Main Program Area Erase)
   "size"    = Size of erase sector

R-IN32M3>
```

**Figure 9. If the bootloader is running and the PC’s USB virtual COM port is set up correctly, typing <help> in the PC terminal should show the bootloader command set.**

## 4.2 Bootloader Commands

Here follows the complete command set.

The command is followed by none, one or more arguments.

**Command [Argument1 [Argument2 [... [ArgumentN]]]**

Address parameter values are *not* absolute but relative to the serial flash memory. That is, the start of the memory device address is zero. The serial flash is at absolute address 0x0200 0000 but commands are relative to the serial flash internally and so addressing starts at zero. Since the user image starts at absolute address 0x0202 0000 one must to read the very beginning of the user image enter e.g.

```
  R-IN32M3>fr 20000 30
```

to read the first 30h bytes of the user image. Figure 14 includes the flash memory map to the right.

| Command | Task | Description |
|---|---|---|
| **help** | Help | Shows all commands and their parameters. |
| **upimg** | Write a user image to flash ROM. | Upload a user image file to the flash memory. The image stream format must be Intel Hex. See Figure 10.<br>Data record lines (type 00) should have 0x10 bytes of data. |
| **btimg** | Execute a SW image | Executes the main program uploaded to flash memory.<br>RP10 does not need to be set low before the command. *1 |
| **fw [address] [data]** | Write data to flash memory | Writes 1 to 16 bytes to the flash memory. *2, *3. |
| **fr [address] [length]** | Read flash memory | Reads [length] bytes from the flash memory. *2. |
| **fe [address] [length]** | Erase flash memory | Erases [length] sector of data from the flash memory.<br>The sector number is given as parameter, or use "all". |

*1. If RP10 pin is low, after RESET, user program will start without bootloader control.
*2. Arguments must be in hexadecimal format; without a leading "0x".
*3. Can be inserted with a variable length of 16 B from 1 to 2 [data].

# 5. User Main Project

Sample user projects can be found in the source tree at
*..\r-in32m3_samplesoft\r-in32m3_samplesoft\Device\Renesas\RIN32M3\Source\Project*

For example, try the *interval_timer* project. This will send a string every second to the serial output if the bootloader successfully writes the user image. See
*..\ R-IN32M3\r-in32m3_samplesoft\r-in32m3_samplesoft\Device\Renesas\RIN32M3\Source\Project\interval_timer\IAR/main.eww*.

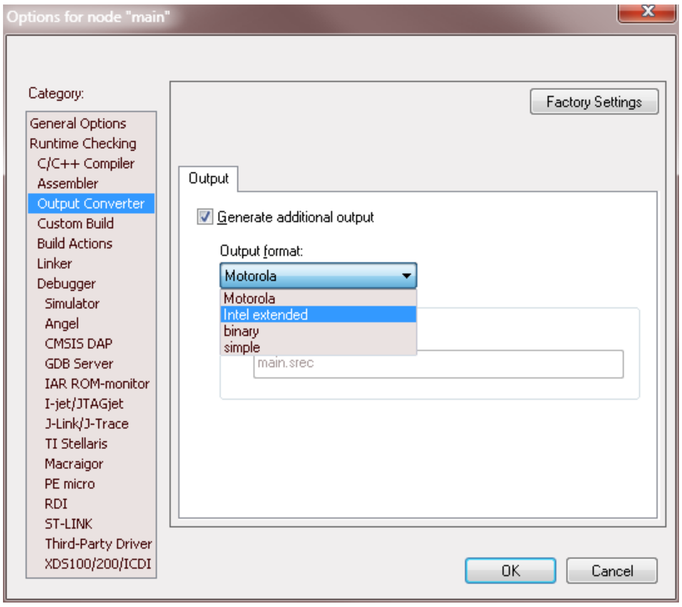
Here follows a brief description of user application limitations.

## 5.1 Project Mode

RAM Mode (Figure 2) must be used for the user application examples. This is because the user image must not occupy Data RAM as that is where the bootloader is located. The RAM Debug projects all place the application code in the Instruction RAM mirror area at 0x0400 0000. The bootloader will start by placing the image stream from the serial link to the same area, and then flash it to the memory device. Both of these happen with the one command <upimg>.

## 5.2 Build Output Format

The input format of the user image that is sent over the serial terminal to the bootloader must be of the Intel Hex format. To set this using IAR, see Figure 10.



☑ Generate additional output

Output format: Motorola

main.srec

**Figure 10. The build output format of the project must be Intel Extended.**

## 5.3 Maximum User Image Size

All programs on the R-IN23 execute from RAM after boot. This means that the user program cannot be larger than the 768 kB instruction RAM mirror area.

## 5.4 Changing User Image Location

The user image is executed from the 768 kB instruction RAM mirror area at IRAM_BASE_ADDRESS (0x0400 0000). To change the location of the user image should this be necessary, the following needs to be done. For complete memory disposition see Section 7.

### 5.4.1 Execution (RAM)

All code is executed from RAM. In the sample code, IRAM is the area for putting the instruction program, so this should be considered “read only” code. Stack pointer or heap should therefore be allocated to the data RAM.

To change where the RAM execution image resides, adjust the following.

1. Change IMG_SRC_AADR (IRAM_BASE_ADDRESS) and IMG_SRC_SIZE in *boot_loader.c*.

2. Change linker configuration of the user project to match above by changing the location of the *readonly* code. For some compilers this corresponds to the C, D, L, and P code sections.

The ideal solution is to have a header with a fixed, by the bootloader known, address in user application image that contains the address of the user image reset vector. This way it does not have to be "hardcoded" to IRAM_BASE_ADDRESS in the bootloader. See 3.2.3, Faulty Image.

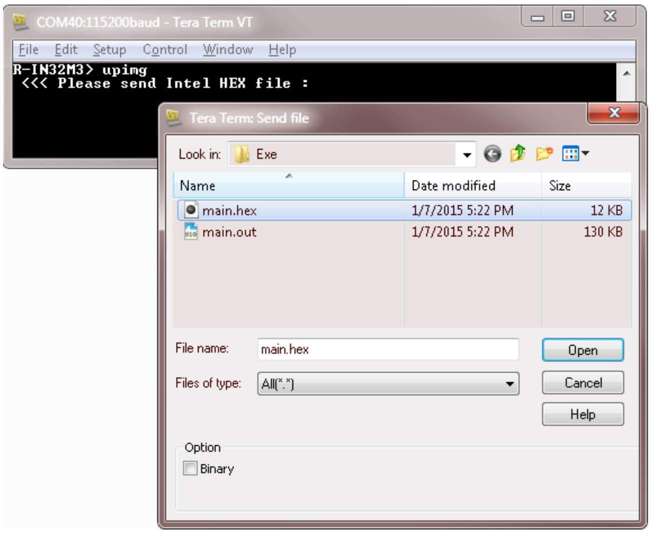
### 5.4.2 Storage (ROM)

The bootloader and user image will be stored in serial or parallel flash depending on project mode, and the setting of the BOOT0 and BOO1 pins. The bootloader will only interface the serial flash for the IAR KSK.

Both the bootloader and user images reside at the relative addresses (within the storage flash) given by BOOT_PARAM_ADDRESS and MAIN_PROGRAM_ADDRESS. This is the offset from the absolute address of the beginning of storage memory. The memory storage disposition is shown respectively for the two storage types with the brown and blue lines in Figure 14.

## 6. Procedure Loading a User Image

This section illustrates how a user image is uploaded to the R-IN32.



COM40:115200baud - Tera Term VT
File Edit Setup Control Window Help
Tera Term: Send file
Open
Cancel
Help

```
R-IN32M3> upimg
 <<< Please send Intel HEX file :
```

Look in: Exe

| Name | Date modified | Size |
|---|---|---|
| main.hex | 1/7/2015 5:22 PM | 12 KB |
| main.out | 1/7/2015 5:22 PM | 130 KB |

File name: main.hex

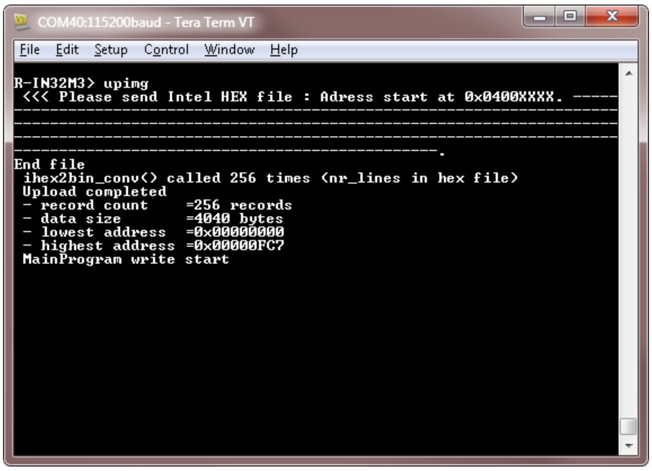
Files of type: All(*.*)

Option

□ Binary

**Figure 11. After entering command <upimg>, the bootloader is waiting for an Intel hex record file.**

The records from the file (one record per line) will be written to the RAM location IRAM_BASE_ADDRESS. This is the same location to which the bootloader copies the user image at boot, after the R-IN32 has been programmed with a user image.

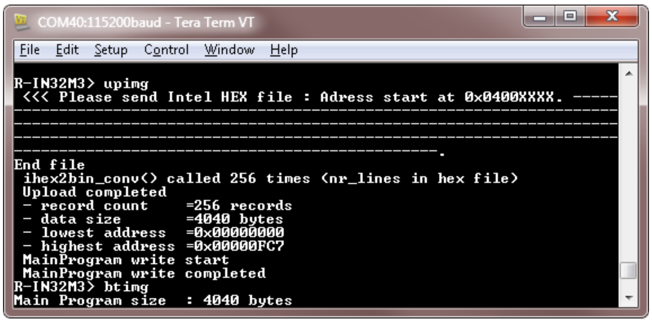


COM40:115200baud - Tera Term VT
File Edit Setup Control Window Help

```
R-IN32M3> upimg
 <<< Please send Intel HEX file : Adress start at 0x0400XXXX. -----
-------------------------------------------------------------------
-------------------------------------------------------------------
-------------------------------------------------------------------
-----------------------------------------------.
End file
 ihex2bin_conv() called 256 times (nr_lines in hex file)
 Upload completed
 - record count    =256 records
 - data size       =4040 bytes
 - lowest address  =0x00000000
 - highest address =0x00000FC7
 MainProgram write start
```

**Figure 12. If the user image was downloaded and received, it will then be programmed to flash. This can take many seconds due to flash erase time.**

The user image will be written to RAM, the flash will be erased, then written to with the same image.



COM40:115200baud - Tera Term VT
File Edit Setup Control Window Help

```
R-IN32M3> upimg
 <<< Please send Intel HEX file : Adress start at 0x0400XXXX. -----
-------------------------------------------------------------------
-------------------------------------------------------------------
-------------------------------------------------------------------
-----------------------------------------------.
End file
 ihex2bin_conv() called 256 times (nr_lines in hex file)
 Upload completed
 - record count    =256 records
 - data size       =4040 bytes
 - lowest address  =0x00000000
 - highest address =0x00000FC7
 MainProgram write start
 MainProgram write completed
R-IN32M3> btimg
Main Program size  : 4040 bytes
```

**Figure 13. Using the command <btimg>, the user image will be booted into if the image is valid.**

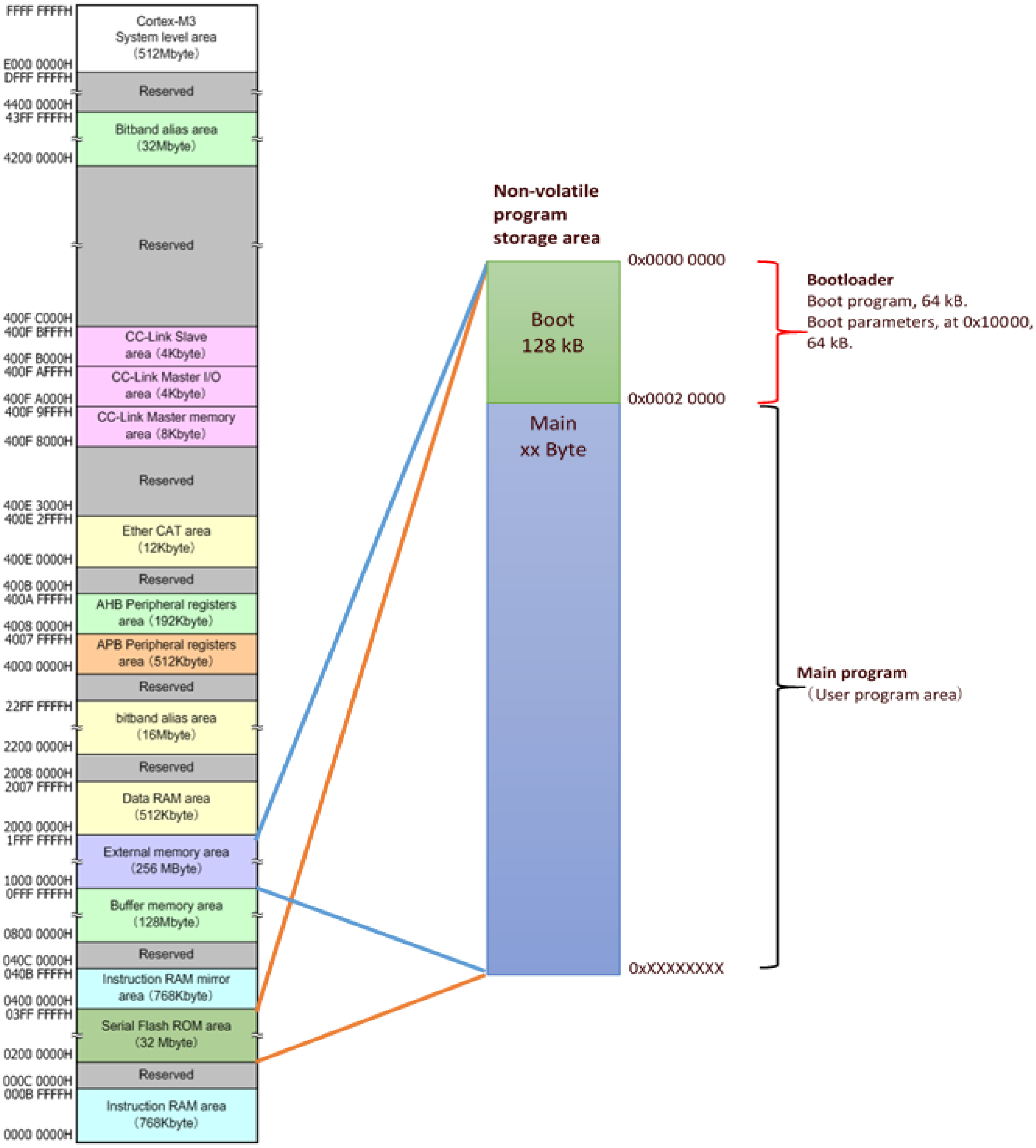
To test run the user image, use the command *upimg*. The user image location in flash will be inspected, and if it exists, it will be copied to the instruction RAM and executed.

## 7. Bootloader and User Image Storage

Both bootloader and user code are stored in permanent storage and are at startup copied to and then executed from RAM.

The bootloader is stored in NOR flash or serial flash, depending on selected project mode (section 8). The user image is stored in serial flash or NOR flash, depending on project mode, and will always be at an offset of 20000h bytes from the beginning of the permanent storage area. This is so that both bootloader and user image can share the same storage media.

The bootloader image is copied to data RAM at 0x2000 0000 (SRAM_BASE), and the user image to the instruction RAM mirror area at 0x0400 0000 (IRAM_BASE_ADDRESS).



.

**Figure 14. The bootloader can be stored in the Serial flash (32 MB) or in NOR flash (256 MB). The user image is stored at an offset of 20000h bytes from the beginning of the permanent storage area so that bootloader and user image can share the same storage media. The addresses on the far right are relative addresses, to be added to the start address for each memory area. The brown lines of Figure 9 are for the case using serial flash and the blue when using for the parallel (NOR) flash.**

## 7.1 Storage Addresses

The absolute storage addresses where the images are permanently stored before being copied to RAM for execution, are as follows.

### 7.1.1 Serial Flash

#### Bootloader

When stored in serial flash, the bootloader is at 0x0200 0000 (SFLASH_BASE_ADDRESS) and space is reserved up to 0x02020000. The bootloader's Serial Boot project mode and corresponding BOOT0 and 1 pin settings must be used.

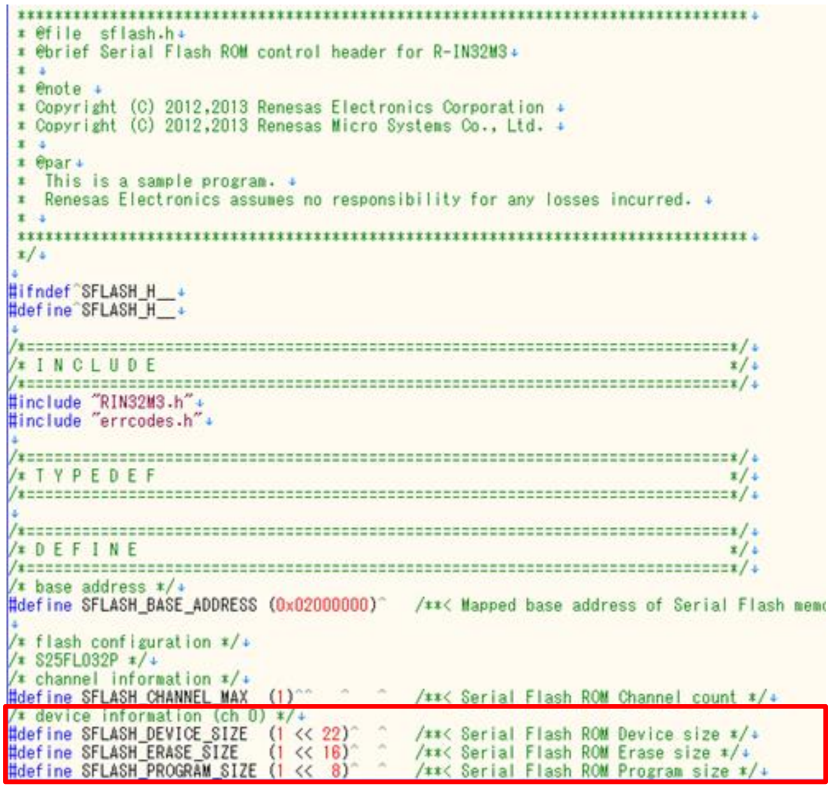
#### User application

The user application starts at 0x0202 0000, and the rest of the serial flash chip is reserved for it.

#### Bootloader modification depending on serial flash

Depending on the serial flash device, the user may need to modify macros in the bootloader. Device definitions for serial flash are in *..\Device\Renesas\RIN32M3\Source\Middleware\sflash\sflash.h*.
The user should modify the following device information according to the serial flash chip's specification.

SFLASH_DEVICE_SIZE = Size of flash device.
SFLASH_ERASE_SIZE = Sector size supported by the sector erase command.
SFLASH_PROGRAM_SIZE = Page size supported for programming.



```
*******************************************************************************
 * @file  sflash.h
 * @brief Serial Flash ROM control header for R-IN32M3
 *
 * @note
 * Copyright (C) 2012,2013 Renesas Electronics Corporation
 * Copyright (C) 2012,2013 Renesas Micro Systems Co., Ltd.
 *
 * @par
 *  This is a sample program.
 *  Renesas Electronics assumes no responsibility for any losses incurred.
 *
*******************************************************************************
 */

#ifndef SFLASH_H__
#define SFLASH_H__

/*============================================================================*/
/* I N C L U D E                                                              */
/*============================================================================*/
#include "RIN32M3.h"
#include "errcodes.h"

/*============================================================================*/
/* T Y P E D E F                                                              */
/*============================================================================*/

/*============================================================================*/
/* D E F I N E                                                                */
/*============================================================================*/
/* base address */
#define SFLASH_BASE_ADDRESS (0x02000000)    /**< Mapped base address of Serial Flash mem

/* flash configuration */
/* S25FL032P */
/* channel information */
#define SFLASH_CHANNEL_MAX  (1)             /**< Serial Flash ROM Channel count */
/* device information (ch 0) */
#define SFLASH_DEVICE_SIZE  (1 << 22)       /**< Serial Flash ROM Device size */
#define SFLASH_ERASE_SIZE   (1 << 16)       /**< Serial Flash ROM Erase size */
#define SFLASH_PROGRAM_SIZE (1 <<  8)       /**< Serial Flash ROM Program size */
```

**Figure 15. Device information that needs to match the serial memory chip.**
**Setting examples for each evaluation board are shown in the next figure.**

| **Memory chip attribute** | **Macro** | **Tessera**<br>**R-IN32M3-EC**<br>**R-IN32M3-CL** | **IAR KickStart Kit**<br>**KSK-RIN32M3EC-LT-IL** |
|---|---|---|---|
| Memory size | `SFLASH_DEVICE_SIZE` | `(1 << 22)` | `(1 << 21)` |
| Sector size supported by sector erase command | `SFLASH_ERASE_SIZE` | `(1 << 16)` | `(1 << 16)` |
| Page size supported by page program command | `SFLASH_PROGRAM_SIZE` | `(1 << 8)` | `(1 << 8)` |

**Figure 16. Serial flash code settings for respective evaluation board.**

For IAR KickStart Kit, modify the source code as per above and as stated for flash size in manual "IAIS-AB-14-0181-3_BootLoaderSample".

### 7.1.2 Parallel Flash

NOR flash is not available in the IAR KSK.

When stored in NOR flash, the bootloader is at 0x1000 0000 to 0x1002 0000. The NOR Boot project mode and corresponding BOOT0 and 1 settings must be used. When using this project mode, the user application is also stored in the NOR (parallel) flash, but at an offset to this as described in section 7.

# 8. Modifying Bootloader for Other Boards

The bootloader code is by default for the IAR R-IN32M3-EC KickStart Kit.

Besides following a different board's startup documentation (see Related Documents) there are a few things that need to be changed.

## 8.1 General

(a) **Serial flash configuration**

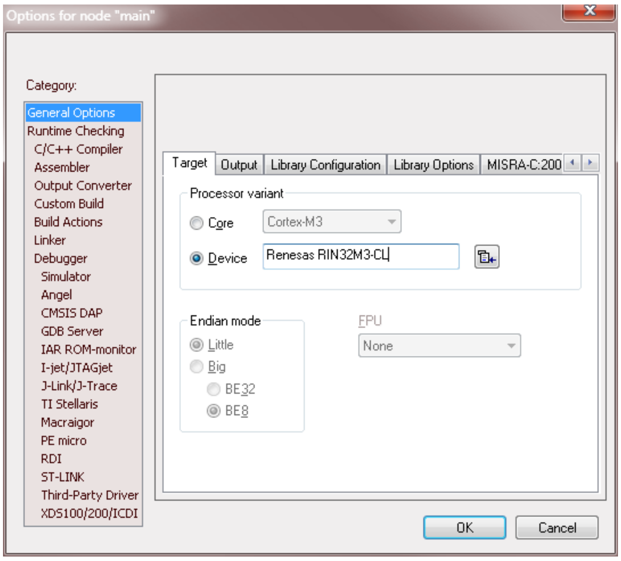
Make sure that serial flash memory chip has the right attributes in the source code as shown in 7.1.1, as this is where the user image is stored.

## 8.2 R-IN32M3-CL

Below is an example of changes needed to the default bootloader project when using the bootloader on the Tessera R-IN32M3-CL board.

(b) **Because different kinds of serial/parallel flash chips are mounted on the -EC and -CL boards, IAR target config files must be changed. Therefore, for a –CL board, copy the content of folder**
***..\r-in32m3_samplesoft\IAR_flashloader\flashloader\Renesas\FlashRIN32M3_CL_flashloader***
**into folder**
***C:\Program Files (x86)\IAR Systems\Embedded Workbench 7.0_2\arm\config\flashloader\Renesas***
**to replace the existing files.**

(c) **Change the board type to R-IN32M3-CL.**



Options for node "main"

Category: General Options

Target

Processor variant

○ Core Cortex-M3

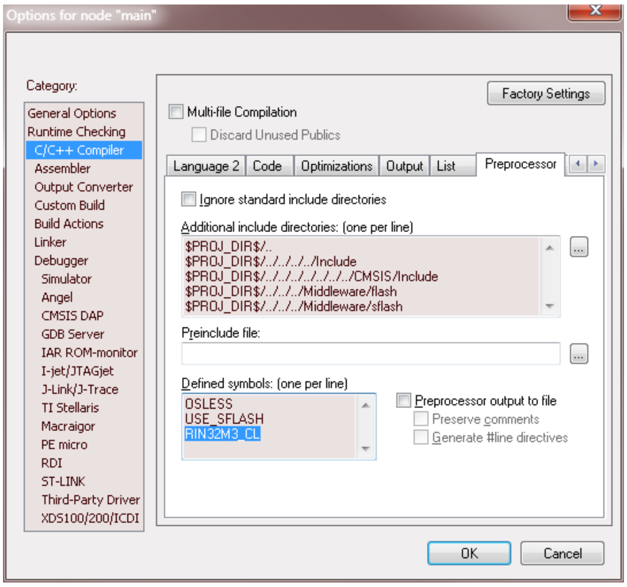
● Device Renesas RIN32M3-CL

Endian mode: ● Little ○ Big (○ BE32 ● BE8)

FPU: None

**Figure 17. For the IAR toolchain, change the board type by going to the project options -> General options, and change "Device".**

(d) **Use compile macro option RIN32M3_CL.**



☐ Multi-file Compilation

☐ Discard Unused Publics

☐ Ignore standard include directories

Additional include directories: (one per line)

```
$PROJ_DIR$/..
$PROJ_DIR$/../../../../Include
$PROJ_DIR$/../../../../../../../CMSIS/Include
$PROJ_DIR$/../../../Middleware/flash
$PROJ_DIR$/../../../Middleware/sflash
```

Preinclude file:

Defined symbols: (one per line)

```
OSLESS
USE_SFLASH
RIN32M3_CL
```

☐ Preprocessor output to file

☐ Preserve comments

☐ Generate #line directives

Figure 18. Adding the compile macro option RIN32M3_CL for IAR.

## Website and Support

Renesas Electronics Website
http://www.renesas.com/

Inquiries
http://www.renesas.com/contact/

## Revision History

| Rev. | Date | Description | |
|---|---|---|---|
| | | Page | Summary |
| 1.00 | Feb 9, 2015 | | Initial version. |

## General Precautions in the Handling of MPU/MCU Products

The following usage notes are applicable to all MPU/MCU products from Renesas. For detailed usage notes on the products covered by this document, refer to the relevant sections of the document as well as any technical updates that have been issued for the products.

1. Handling of Unused Pins

   Handle unused pins in accordance with the directions given under Handling of Unused Pins in the manual.

   — The input pins of CMOS products are generally in the high-impedance state. In operation with an unused pin in the open-circuit state, extra electromagnetic noise is induced in the vicinity of LSI, an associated shoot-through current flows internally, and malfunctions occur due to the false recognition of the pin state as an input signal become possible. Unused pins should be handled as described under Handling of Unused Pins in the manual.

2. Processing at Power-on

   The state of the product is undefined at the moment when power is supplied.

   — The states of internal circuits in the LSI are indeterminate and the states of register settings and pins are undefined at the moment when power is supplied.
   In a finished product where the reset signal is applied to the external reset pin, the states of pins are not guaranteed from the moment when power is supplied until the reset process is completed.
   In a similar way, the states of pins in a product that is reset by an on-chip power-on reset function are not guaranteed from the moment when power is supplied until the power reaches the level at which resetting has been specified.

3. Prohibition of Access to Reserved Addresses

   Access to reserved addresses is prohibited.

   — The reserved addresses are provided for the possible future expansion of functions. Do not access these addresses; the correct operation of LSI is not guaranteed if they are accessed.

4. Clock Signals

   After applying a reset, only release the reset line after the operating clock signal has become stable. When switching the clock signal during program execution, wait until the target clock signal has stabilized.

   — When the clock signal is generated with an external resonator (or from an external oscillator) during a reset, ensure that the reset line is only released after full stabilization of the clock signal. Moreover, when switching to a clock signal produced with an external resonator (or by an external oscillator) while program execution is in progress, wait until the target clock signal is stable.

5. Differences between Products

   Before changing from one product to another, i.e. to a product with a different part number, confirm that the change will not lead to problems.

   — The characteristics of an MPU or MCU in the same group but having a different part number may differ in terms of the internal memory capacity, layout pattern, and other factors, which can affect the ranges of electrical characteristics, such as characteristic values, operating margins, immunity to noise, and amount of radiated noise. When changing to a product with a different part number, implement a system-evaluation test for the given product.

## Notice

1. Descriptions of circuits, software and other related information in this document are provided only to illustrate the operation of semiconductor products and application examples. You are fully responsible for the incorporation of these circuits, software, and information in the design of your equipment. Renesas Electronics assumes no responsibility for any losses incurred by you or third parties arising from the use of these circuits, software, or information.
2. Renesas Electronics has used reasonable care in preparing the information included in this document, but Renesas Electronics does not warrant that such information is error free. Renesas Electronics assumes no liability whatsoever for any damages incurred by you resulting from errors in or omissions from the information included herein.
3. Renesas Electronics does not assume any liability for infringement of patents, copyrights, or other intellectual property rights of third parties by or arising from the use of Renesas Electronics products or technical information described in this document. No license, express, implied or otherwise, is granted hereby under any patents, copyrights or other intellectual property rights of Renesas Electronics or others.
4. You should not alter, modify, copy, or otherwise misappropriate any Renesas Electronics product, whether in whole or in part. Renesas Electronics assumes no responsibility for any losses incurred by you or third parties arising from such alteration, modification, copy or otherwise misappropriation of Renesas Electronics product.
5. Renesas Electronics products are classified according to the following two quality grades: "Standard" and "High Quality". The recommended applications for each Renesas Electronics product depends on the product's quality grade, as indicated below.

   "Standard": Computers; office equipment; communications equipment; test and measurement equipment; audio and visual equipment; home electronic appliances; machine tools; personal electronic equipment; and industrial robots etc.

   "High Quality": Transportation equipment (automobiles, trains, ships, etc.); traffic control systems; anti-disaster systems; anti-crime systems; and safety equipment etc.

   Renesas Electronics products are neither intended nor authorized for use in products or systems that may pose a direct threat to human life or bodily injury (artificial life support devices or systems, surgical implantations etc.), or may cause serious property damages (nuclear reactor control systems, military equipment etc.). You must check the quality grade of each Renesas Electronics product before using it in a particular application. You may not use any Renesas Electronics product for any application for which it is not intended. Renesas Electronics shall not be in any way liable for any damages or losses incurred by you or third parties arising from the use of any Renesas Electronics product for which the product is not intended by Renesas Electronics.
6. You should use the Renesas Electronics products described in this document within the range specified by Renesas Electronics, especially with respect to the maximum rating, operating supply voltage range, movement power voltage range, heat radiation characteristics, installation and other product characteristics. Renesas Electronics shall have no liability for malfunctions or damages arising out of the use of Renesas Electronics products beyond such specified ranges.
7. Although Renesas Electronics endeavors to improve the quality and reliability of its products, semiconductor products have specific characteristics such as the occurrence of failure at a certain rate and malfunctions under certain use conditions. Further, Renesas Electronics products are not subject to radiation resistance design. Please be sure to implement safety measures to guard them against the possibility of physical injury, and injury or damage caused by fire in the event of the failure of a Renesas Electronics product, such as safety design for hardware and software including but not limited to redundancy, fire control and malfunction prevention, appropriate treatment for aging degradation or any other appropriate measures. Because the evaluation of microcomputer software alone is very difficult, please evaluate the safety of the final products or systems manufactured by you.
8. Please contact a Renesas Electronics sales office for details as to environmental matters such as the environmental compatibility of each Renesas Electronics product. Please use Renesas Electronics products in compliance with all applicable laws and regulations that regulate the inclusion or use of controlled substances, including without limitation, the EU RoHS Directive. Renesas Electronics assumes no liability for damages or losses occurring as a result of your noncompliance with applicable laws and regulations.
9. Renesas Electronics products and technology may not be used for or incorporated into any products or systems whose manufacture, use, or sale is prohibited under any applicable domestic or foreign laws or regulations. You should not use Renesas Electronics products or technology described in this document for any purpose relating to military applications or use by the military, including but not limited to the development of weapons of mass destruction. When exporting the Renesas Electronics products or technology described in this document, you should comply with the applicable export control laws and regulations and follow the procedures required by such laws and regulations.
10. It is the responsibility of the buyer or distributor of Renesas Electronics products, who distributes, disposes of, or otherwise places the product with a third party, to notify such third party in advance of the contents and conditions set forth in this document, Renesas Electronics assumes no responsibility for any losses incurred by you or third parties as a result of unauthorized use of Renesas Electronics products.
11. This document may not be reproduced or duplicated in any form, in whole or in part, without prior written consent of Renesas Electronics.
12. Please contact a Renesas Electronics sales office if you have any questions regarding the information contained in this document or Renesas Electronics products, or if you have any other inquiries.

(Note 1) "Renesas Electronics" as used in this document means Renesas Electronics Corporation and also includes its majority-owned subsidiaries.

(Note 2) "Renesas Electronics product(s)" means any product developed or manufactured by or for Renesas Electronics.

RENESAS

**SALES OFFICES** Renesas Electronics Corporation http://www.renesas.com

Refer to "http://www.renesas.com/" for the latest and detailed information.

**Renesas Electronics America Inc.**
2801 Scott Boulevard Santa Clara, CA 95050-2549, U.S.A.
Tel: +1-408-588-6000, Fax: +1-408-588-6130

**Renesas Electronics Canada Limited**
9251 Yonge Street, Suite 8309 Richmond Hill, Ontario Canada L4C 9T3
Tel: +1-905-237-2004

**Renesas Electronics Europe Limited**
Dukes Meadow, Millboard Road, Bourne End, Buckinghamshire, SL8 5FH, U.K
Tel: +44-1628-585-100, Fax: +44-1628-585-900

**Renesas Electronics Europe GmbH**
Arcadiastrasse 10, 40472 Düsseldorf, Germany
Tel: +49-211-6503-0, Fax: +49-211-6503-1327

**Renesas Electronics (China) Co., Ltd.**
Room 1709, Quantum Plaza, No.27 ZhiChunLu Haidian District, Beijing 100191, P.R.China
Tel: +86-10-8235-1155, Fax: +86-10-8235-7679

**Renesas Electronics (Shanghai) Co., Ltd.**
Unit 301, Tower A, Central Towers, 555 Langao Road, Putuo District, Shanghai, P. R. China 200333
Tel: +86-21-2226-0888, Fax: +86-21-2226-0999

**Renesas Electronics Hong Kong Limited**
Unit 1601-1611, 16/F., Tower 2, Grand Century Place, 193 Prince Edward Road West, Mongkok, Kowloon, Hong Kong
Tel: +852-2265-6688, Fax: +852 2886-9022

**Renesas Electronics Taiwan Co., Ltd.**
13F, No. 363, Fu Shing North Road, Taipei 10543, Taiwan
Tel: +886-2-8175-9600, Fax: +886 2-8175-9670

**Renesas Electronics Singapore Pte. Ltd.**
80 Bendemeer Road, Unit #06-02 Hyflux Innovation Centre, Singapore 339949
Tel: +65-6213-0200, Fax: +65-6213-0300

**Renesas Electronics Malaysia Sdn.Bhd.**
Unit 1207, Block B, Menara Amcorp, Amcorp Trade Centre, No. 18, Jln Persiaran Barat, 46050 Petaling Jaya, Selangor Darul Ehsan, Malaysia
Tel: +60-3-7955-9390, Fax: +60-3-7955-9510

**Renesas Electronics India Pvt. Ltd.**
No.777C, 100 Feet Road, HAL II Stage, Indiranagar, Bangalore, India
Tel: +91-80-67208700, Fax: +91-80-67208777

**Renesas Electronics Korea Co., Ltd.**
12F., 234 Teheran-ro, Gangnam-Gu, Seoul, 135-080, Korea
Tel: +82-2-558-3737, Fax: +82-2-558-5141

Colophon 5.0