Application Note

RZ/N1D Group
RZ/N1S Group
RZ/N1L Group

Application Note:
Management Quick Start Guide

RZ Family RZ/N1 Series

Preliminary

Renesas Electronics
www.renesas.com

Rev. 1.4.3 July 09, 2019

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.

General Precautions in the Handling of the Product

The following usage notes are applicable to all 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.

Table of Content

Figures

Tables

1. Introduction

This document describes the setup of the management software for the CM3 core of the RZ/N1D, RZ/N1D together with RZ/N1D-EB board, RZ/N1S and RZ/N1L.

Please note that the software was tested using hardware version
EESS-0401-130-04 (RZ/N1D),
EESS-0401-131-03 (RZ/N1D-EB),
EESS-0401-141-02 (RZ/N1S),
EESS-0401-155-01 (RZ/N1L),
of the CPU and extension board.

Please note that the RZ/N1S requires at least hardware version EESS-0401-141-02 to work with the extension board correctly.

2. Project Setup

The following chapter describes the setup and usage of the Management Software.

2.1. Requirements

Please extract the RZ/N archive to the workspace.

Please make sure the following components are installed on the computer:

Tool	Version
IAR Embedded Workbench for ARM	8.32.3.20228
IAR C/C++ Compiler for ARM	8.32.3.193
GCC	8.2.0

Table 1: Development Tools required by the Management Software

Furthermore, the additional software is needed:

- DHCP server
  - the device will issue DHCP requests by default
- Terminal Emulator
  - log messages will be printed to the UART of the CPU Board

2.2. Hardware

Please take care to follow the setup guidelines for the RZ/N1 Demo Board from the Linux and U-Boot documentation - *RZN1x-Quick-Start-Guide.pdf*

Please follow these initial steps to setup the UART and DFU connection.

1. Connect the board to a PC via the UART and the DFU interface. After the driver for the device has been installed, additional serial ports will show up.
   a. On Linux PCs, if you have no other serial-over-USB devices attached, this is /dev/ttyUSB2.
   b. On Windows PCs, open the *device manager* and look up for new USB Serial Ports on section *ports*. The RZ/N1D and RZ/N1S board uses the 3rd of the 4 COM ports.

2. Open and configure a suitable terminal emulator on PC.
   a. On Linux PCs, open a serial terminal e.g. with

      cu -e -o -115200 -l /dev/XXX

      Replace the “XXX” with the serial device where the UART of the board is connected to.
   b. On Windows PCs, open a serial terminal program e.g. PuTTY and select the COM port where the UART is connected to. The following settings must be configured for the connection:

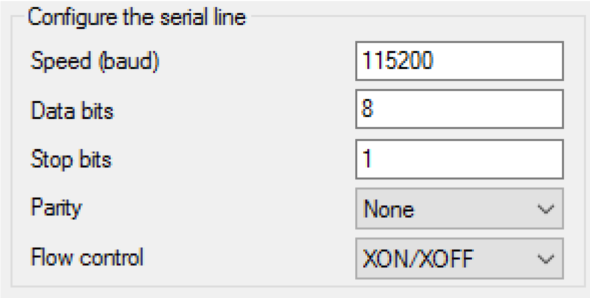


Figure 2-1: Serial Terminal settings RZ/N1

2.3. Sample Application

The management software provides several example projects for the IAR Workbench IDE showing the different functionalities for the switch management and GOAL.
All example projects are in the folder *projects/00410_goal/* and *projects/goal_http*. The project workspaces ending on *_eb contain the configuration for the CPU Board together with the extension board (4 switch ports). The other project workspaces contain the configuration for working with the CPU Board only.
The following examples are provided:

- **00410_goal/ cfg_cli:** Demonstrates the functionalities of Config Manager using the CLI
- **00410_goal/ cfg_demo:** Demonstrates the functionalities of the Config Manager
- **00410_goal/ chase_lights:** LED chasing lights demo
- **00410_goal/ cli:** Command Line Interface demo
- **00410_goal/ cli_udp:** Command Line Interface via UDP demo
- **00410_goal/ dhcp:** DHCP client demo
- **00410_goal/ eth_dump:** Dumps src/dst MAC addresses of incoming frames
- **00410_goal/ eth_state_ctc_ac*:** Displaying Ethernet statistics via MCTC on AC from CC
- **00410_goal/ eth_state_ctc_cc*:** Sending Ethernet statistics via MCTC from AC to CC.
- **00410_goal/ eth_stats:** Demo for statistics counters of the 5-port switch
- **00410_goal/ eth_stats:** Demo for statistics counters of the 5-port switch
- **00410_goal/ iface_info:** Demo for link, speed and duplex recognition
- **00410_goal/ igmp_snoop:** IGMP snooping demo
- **00410_goal/ iperf2_client_tcp:** TCP client for benchmarking with iperf2
- **00410_goal/ iperf2_client_udp:** UDP client for benchmarking with iperf2
- **00410_goal/ iperf2_server_tcp_goal:** Internal GOAL server for TCP benchmarking with iperf2
- **00410_goal/ iperf2_server_tcp_lwiperf** Internal lwIP server for TCP benchmarking with iperf2
- **00410_goal/ iperf2_server_udp:** UDP server for benchmarking with iperf2
- **00410_goal/ linked_list:** GOAL linked list feature usage demo
- **00410_goal/ lm:** GOAL Log Manager feature usage demo
- **00410_goal/ mailbox_ac*:** GOAL Mailbox API demo (application core side)
- **00410_goal/ mailbox_cc*:** GOAL Mailbox API demo (communication core side)
- **00410_goal/ mctc_ac*:** GOAL MCTC cyclic and acyclic usage demo (application core side)
- **00410_goal/ mctc_cc*:** GOAL MCTC cyclic and acyclic usage demo (communication core side)
- **00410_goal/ mem_buf:** GOAL memory buffer allocation usage demo
- **00410_goal/ mem_deny_delay:** Delaying GOAL memory allocation lock demo

- **00410_goal/ no_net:** Simple GOAL demo without network
- **00410_goal/ rb:** GOAL ring buffer usage demo
- **00410_goal/ rpc_ac*:** RPC application core demo
- **00410_goal/ rpc_cc***: RPC communication core demo
- **00410_goal/ snmp:** SNMPv2c agent with MIB II implementation (no UDP and TCP group)
- **00410_goal/ task:** Demo for working with tasks in GOAL
- **00410_goal/ task_lock:** Shows synchronization of two tasks via locking in GOAL
- **00410_goal/ tcp_client:** TCP client demo
- **00410_goal/ tcp_server:** TCP server demo
- **00410_goal/ template:** Project template for creating new projects
- **00410_goal/ udp_jumbo_frames:** Demo for receiving UDP packets and jumbo frames
- **00410_goal/ udp_receive:** Demo for receiving UDP packets
- **goal_http/01_get:** Simple Webserver demo (HTTP get)
- **goal_http/02_post:** Simple Webserver demo (HTTP post)
- **goal_http/03_list_res:** Simple Webserver demo listing system ressources
- **goal_http/04_auth:** Simple Webserver demo using basic authentication
- **goal_http/05_template_cm:** Simple Webserver template using CM variables and callbacks
- **goal_http/06_template_list** Simple Webserver template for unnumbered lists
- **goal_http/07_template_table:** Simple Webserver template for generating tables

(*) These projects are used for communication with a peer core by Core To Core. Please note the separate sections for flashing and debugging.of these examples.

Additional functionalities of the switch are accessible via the command line interface. Please see *r11an0229ed0131-rzn1-goal-cli-guidelines.docx* and *r11uz0009ed0131-rzn1-goal-cli-specifications.docx* for a more detailed description. The availability of specific commands may differ depending on the activated features.
The CLI also allows to read arbitrary memory data of the R-IN. This includes access to all 5-port switch registers. For reference, see command description of command “dbg mem<b/w/d> show” in the CLI guide. The memory location of the different switch registers can be found in RZ/N1 user’s manual - *r01uh0753ej0xxx-rzn1-ether.pdf “RZ/N1 Group User’s Manual: R-IN Engine and Ethernet Peripherals”.*

2.4. Running a sample application

The RZ/N1D and RZ/N1S use the U-Boot bootloader for initial setup of the hardware and loading of the CM3 firmware. Additionally, the RZ/N1D U-Boot bootloader is used for booting the Linux Kernel. The RZ/N1L is working without any bootloader. This chapter describes how to install the management software on the flash of the board. If no bootloader was yet installed on the RZ/N1D and RZ/N1S please refer to the Linux documentation - Quick Start Guide for U-Boot and Linux - *RZN1x-Quick-Start-Guide.pdf.*

There are many similarities between the derivatives of the RZ/N1 series but some minor differences, too. Therefore, here is a more detailed explanation how to run a sample application on each.

All standalone projects and the “_cc” project of the *Core To Core* variant contain different workspaces for each board variant. The project workspaces ending on “_eb” contain the configuration for the CPU Board together with the extension board (4 switch ports). The other project workspaces contain the configuration for working with the CPU Board only.

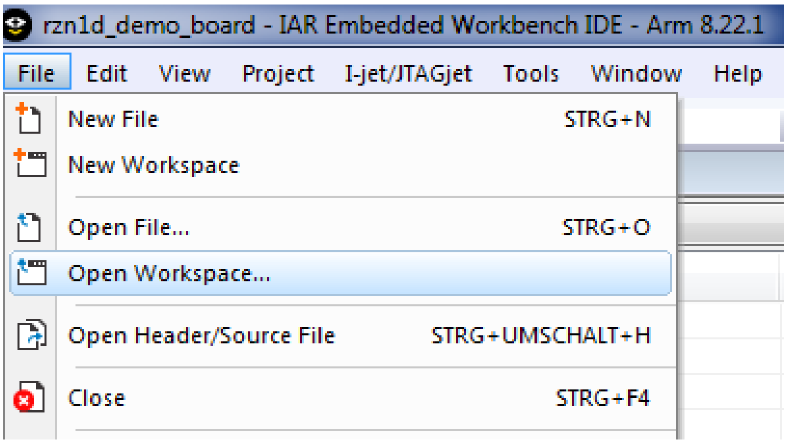
2.4.1. Standalone Variant – RZ/N1D and RZ/N1S

It is possible to load the code via debugger into RAM, which is a very fast approach to test the user application. Any project located in *goal\projects* must be built using IAR Embedded Workbench.

2.4.6.1. Loading application into RAM via IAR Embedded Workbench

To compile a project, follow these steps:

1. Start the IAR Workbench IDE.
2. Open a project via “File/Open Workspace”.



3. Go to the workspace folder and open it. In case the CPU board is used together with the extension board, please ensure to select the correct IAR-project.
4. Compile the project via “Project/Compile” or “Project/Rebuild all”.
5. Power up the device.
6. Open a serial terminal according to section 2.2**.**
7. Press any key on your keyboard to interrupt the bootloader.
8. Ensure to configure the U-Boot boot command to release the CM3 core after reset. This is done by the command:

   setenv bootcmd "mw 0x04000004 1 && rzn1_start_cm3 && loop 0 1"

   followed by

   saveenv

   and reset the board.
9. Connect the debugger to the system via the “Download and Debug” button of the IAR Workbench.
10. After the Debug view opened, click on the “Go” button.

2.4.6.2. Loading application into flash via dfu-util

The board uses the U-Boot bootloader for initial setup of the hardware and loading of the CM3 core firmware. This chapter describes how to install the compiled management software on the flash of the

board. If no bootloader was yet installed on the board, please refer to the Linux documentation - Quick Start Guide for U-Boot and Linux - *RZN1x-Quick-Start-Guide.pdf.*
The following steps describe the installation of the management software:

1. Connect a Linux PC to the board according to section 2.2.
2. Power up the board.
3. Open a serial terminal according to section 2.2.
4. Hit any key to stop the autoboot of the U-Boot.
5. Type *"dfu"* in the serial terminal of the board and hit enter.
6. On a Linux terminal start the command

   sudo dfu-util -a "sf_cm3" -D FIRMWARE.bin

   Replace *FIRMWARE.bin* with the file name of the software to install. The binary is placed at the subfolder *Debug-RAM\Exe* of the IAR project folder.
7. When the download process is complete, press Ctrl+C on U-Boot.
8. If the autoboot command was already configured, go to step 10.
9. Set the autoboot command in the U-Boot:

   setenv bootcmd "sf probe && sf read 0x4000000 d0000 90000 && rzn1_start_cm3 && loop 0 1"

10. Save the command to the flash:

    saveenv

11. Reset the device

2.4.1. Standalone Variant – RZ/N1L

The RZ/N1L does not use any bootloaders. If any application is stored in flash, it will be started automatically. Both, loading into RAM and flash can be done using IAR workbench.

1. Start the IAR Workbench IDE.
2. Open a project via "File/Open Workspace".
3. Go to the workspace folder and open it.
4. Compile the project via "Project/Compile" or "Project/Rebuild all".
5. Power up the device.
6. Open a serial terminal according to section 2.2.
7. Choose either the Debug-RAM or the Debug-ROM configuration. First is used for debugging via IAR, second is loading the application into the flash.

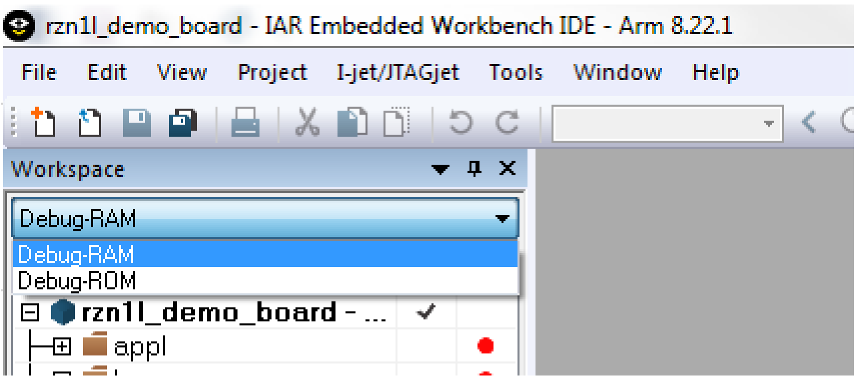


Figure 2-2: IAR Configurations RAM and ROM for RZ/N1L

8. Follow these steps for the Debug-RAM configuration
    a. Compile the project via “Project/Compile” or “Project/Rebuild all”.
    b. Press and hold the devices software-reset button.
    c. Click on “Download and Debug” and release the software-reset button as soon as the “Busy” window opens.

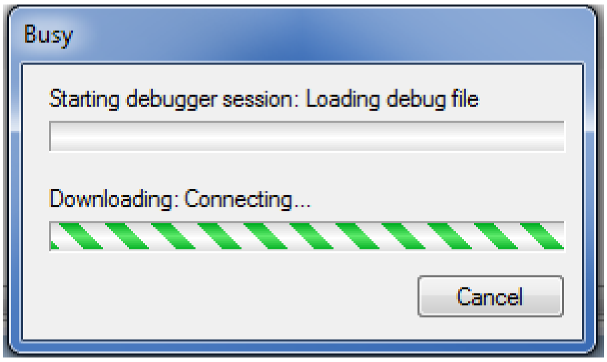


Figure 2-3: IAR Workbench "Busy"-Window

9. Follow these steps for the Debug-ROM configuration
    a. Click on “Download and Debug”.
    b. Set reset mode to “Hardware” and press “Make & Restart Debugger”.
    c. Check, if the reset mode is still on “Hardware”. If not, repeat the previous step.

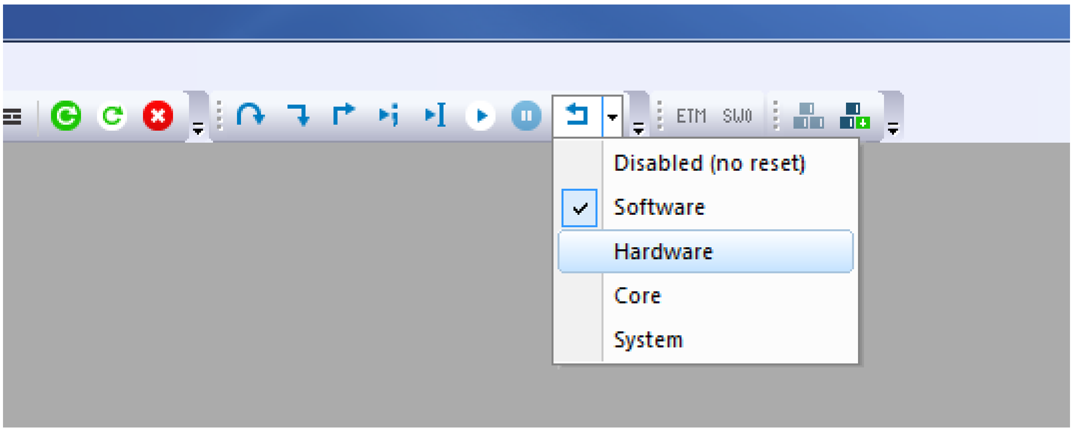


Figure 2-4: Changing Reset mode of RZ/N1L in Debug-ROM configuration

10. After the Debug view opened, click on the “Go” button.

2.4.1. Core To Core variant – RZ/N1D (Communication Core)

This section relates to communication core projects of the GOAL sample application. After building the binary by IAR Embedded Workbench, the file for the CM3 core is located in the subdirectory *Debug-RAM\Exe*.

Load the binary file to the flash according to the following steps.
1. Connect a Linux PC to the board according to section 2.2.
2. Power up the board.
3. Open a serial terminal according to section 2.2.

4. Hit any key to stop the autoboot of the U-Boot.
5. Type *dfu* in the serial terminal of the board and hit enter.
6. On a Linux terminal start the command

   sudo dfu-util -a "sf_cm3" -D FIRMWARE

   Replace *FIRMWARE* with the file name of the software to install.
7. When the download process is complete, press Ctrl+C on U-Boot.
8. If the autoboot command was already configured, go to step 10.
9. Set the autoboot command in the U-Boot:

   setenv bootcmd "sf probe && sf read 0x4000000 d0000 90000 && sf read 0x8ffe0000 b0000 20000 && sf read 0x80008000 1d0000 f00000 && rzn1_start_cm3 && sleep 4 && bootm 0x80008000 - 0x8ffe0000"

10. Save the command to the flash: *saveenv*
11. Reset the device

It is also possible to debug the RZ/N1D communication core. The steps accored to section 2.4.1, but the boot command has to be set to

setenv bootcmd "mw 0x04000004 1 && rzn1_start_cm3 && sleep 20 && sf probe && sf read 0x80008000 1d0000 f00000 && sf read 0x8ffe0000 b0000 20000 && bootm 0x80008000 - 0x8ffe0000"

This delays the boot of Linux about 20 seconds. Meanwhile the CM3 core has to be startet.

2.4.1. Core To Core variant – RZ/N1D (Application Core)

The user application runs on the Linux system of the CA7. Its binary must be created by GCC and downloaded to the RZ/N board manually.

2.4.9.1. Building and downloading the user application

The following steps describe, how to build a binary and download it to the RZ/N1D board.

1. Navigate with the terminal of a Linux PC to the project of the application core.
2. Start the build process by executing the Makefile by typing
   make
3. Select as target platform “rzn_a7_demo_board”.
4. Power up the board and wait till Linux booted successfully.
5. Copy the created binary file *build/rzn_a7_demo_board/goal_rzn_a7_demo_board.bin* to the RZ/N1 board by e.g. secure copy (scp).
6. Start the binary file on the target by typing the commands
   ./goal_rzn_a7_demo_board.bin

The GOAL setups the connection to the communication core via Core To Core and starts the user application. The initialization is done when the log message “*GOAL initialized”* is printed at the terminal, if logging is activated.

2.4.9.2. Auto start the user application

The Linux Kernel can start the user application on the CA7 automatically with the help of the start script

S99goal_app.sh

This script is placed at *linux_ctc/* of the release. Download the file to the CA7, like the user application binary, and place it at */etc/rc5.d/ if this file is not present*. Please ensure, that *goal_rzn_a7_demo_board.bin* at */home/root/*.

Disabling the start script is possible by adding the boot argument *GOAL_APPL_LINUX_PREV*.

1. Power up the board.
2. Hit any key to stop the autoboot of the U-Boot.
3. Add the boot argument for preventing the application autoboot by

   setenv bootargs "${bootargs} GOAL_APPL_LINUX_PREV"

4. Save the command to the flash by:

   saveenv

5. Reset the device.

Reenabling the start script is possible by deleting the boot argument *GOAL_APPL_LINUX_PREV.*

1. Power up the board.
2. Hit any key to stop the autoboot of the U-Boot.
3. Display the environment by

   env print

4. The latest boot arguments are listed at the line *bootargs=*
5. Copy these arguments, except *GOAL_APPL_LINUX_PREV* and paste them at <paste> on the following command

   setenv bootargs "<paste>"

6. Save the command to the flash by:

   saveenv

7. Reset the device.

2.4.1. Core To Core variant – RZ/N1S

Similar to the standalone variant the Core To Core variant is also capable to run from the RAM while debugging the application core and the communication core at the same time.

The IAR Embedded Workbench runs two instances of the IDE, one for each core, in a master-slave-system to share the access to the board keeping both instances synchronous.

The usage and setup of the multicore debugging will be exemplary described for the RPC application example.

To run the RPC example please perform the following steps:

1. Open the corresponding AC IAR project workspace, e.g.:

   projects\00410_goal\rpc_ac\iar\renesas\rzn1s_a7_threadx\rzn1s_a7_threadx.eww

2. Open the project options and navigate to the subcategory “Multicore” in the category “Debugger”.
3. Enable Multicore master mode and select the slave workspace to use. Please note, that the “slave project” and the “slave configuration” is already preconfigured for the GOAL slave projects.
   The Core To Core variant requires the corresponding “_cc” project running on the CM3 which is the same project as for the Core To Core variant under Linux on the RZ/N1D but for the RZ/N1S demo board instead.
   The slave workspace *rzn1s_demo_board.eww* is located in the following project directory:

   projects\00410_goal\rpc_ac\iar\renesas\rzn1s_demo_board

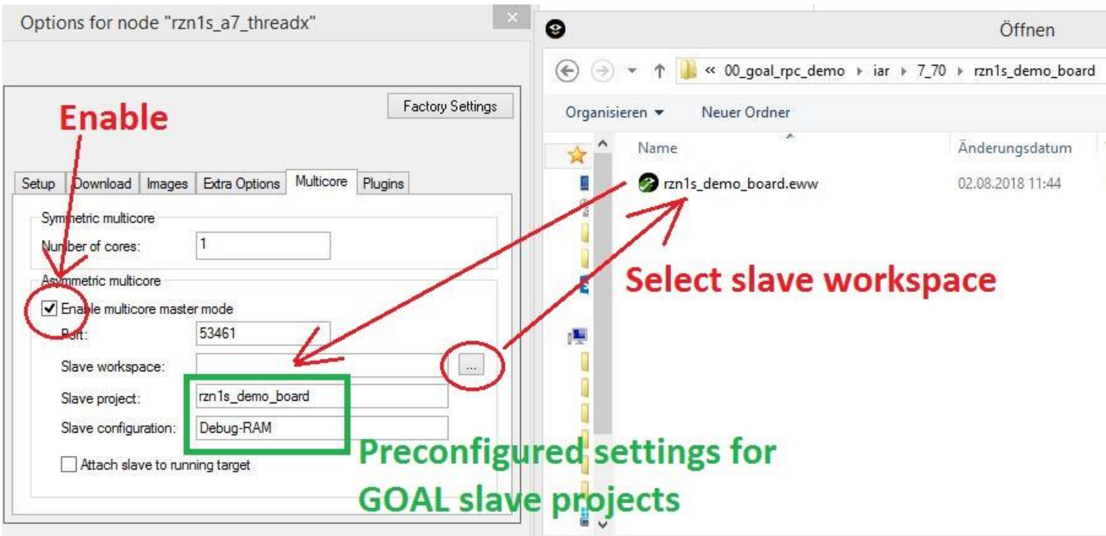


Figure 2-5: Multicore Debug Option

   When using the RZ/N1S expansion board, please ensure to select the correct CC project located at the projects rzn1s_demo_board_eb directory. Additionally, adjust the entry “Slave project” in the subcategory “Multicore” to *rzn1s_demo_board_eb*.
4. Compile the project via “Project/Compile” or “Project/Rebuild all”.
5. Press the “Download and debug” button or Ctrl+D
   This will cause IAR to open the slave workspace as an additional IAR workbench instance, builds the slave project and load both – the master and the slave project – to the board sharing the debugger.
6. When the software from both instances is loaded to the board and the IDE switches in the debug mode an additional dialog for multicore debugging is available giving the following options:
   - start all cores at once
   - stop all cores at once
   - toggle execution mode

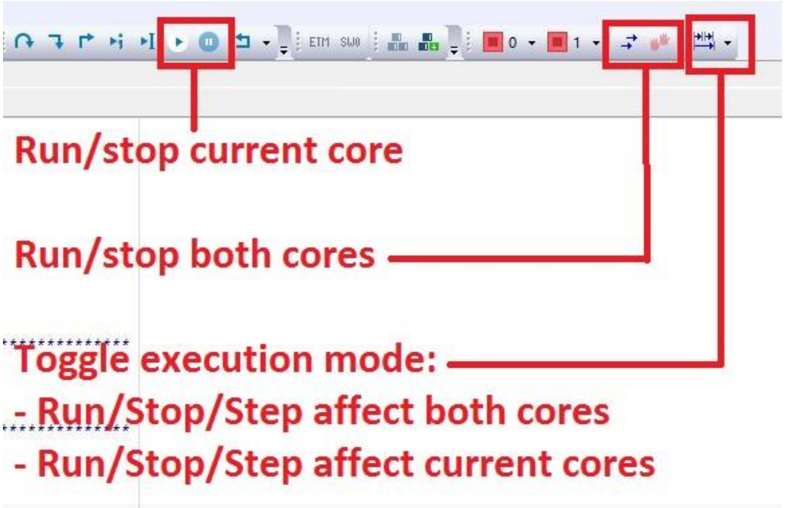


Figure 2-6: Multicore Debug Interface

2.4.1. Core To Core variant – RZ/N1L (Communication Core)

Please refer section 2.4.1 for building and downloading the Core To Core variant on RZ/N1L. It is handled the same as the standalone variant.

For mulit core projects, the RZ/N1L is used as communication core, while the e.g. Synergy S7GS-SK is used as application core. Data exchanging is done by SPI. The boards are connected as followed.

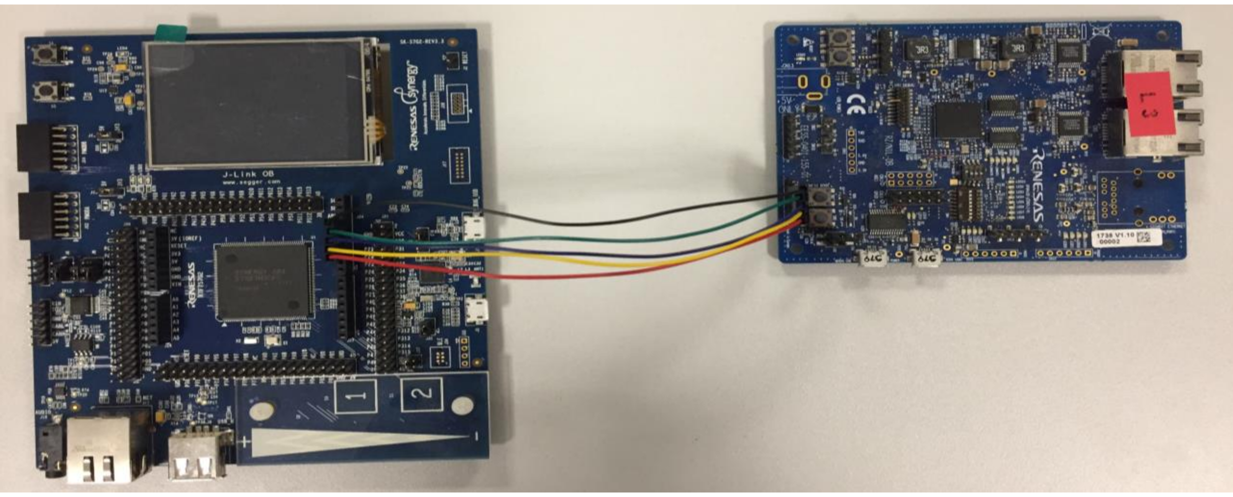


Figure 2-7: SPI connection of Synergy S7GS-SK (left) and RZ/N1L (right)

function	color	S7GS-SK	RZ/N1L
GND	Black	J24-7	CN20-5
SPI Clock	Green	J24-6	CN20-4
MISO	Blue	J24-5	CN20-3
MOSI	Yellow	J24-4	CN20-2
SPI chip select	Red	J24-3	CN20-1

Table 2-2: PINs for SPI usage

Please note the synergy quick start guide for setup the named core. By default, the RZ/N1L uses the SPI channel 5 and the following GPIOs

GPIO	Usage
62	SPI clock
63	MOSI
64	MISO
65	SPI chip select

Table 2-3: GPIOs for SPI usage

Note:
The board supports only SPI mode 1 and 3. Please set the SPI mode to 3 by defining *GOAL_GLOB_MA_SPI_ID_0_MODE_3* in *goal/goal_global/goal_global.h* to 1.

#define GOAL_GLOB_MA_SPI_ID_0_MODE_3 1 /**< set SPI mode 3 on MA ID 0 */

3. Revision History

Version	**Created**		**Validated**		**Released**	
	Date	**Name**	**Date**	**Name**	**Date**	**Name**
1.0	2016-10-26	Marcus Tangermann	2016-10-27	Marcus Züche		
Initial document						
1.1	2017-04-04	Marcus Züche	2016-04-04	Sven Bachmann	2016-04-04	Marcus Züche
Update Applications and Hardware description; add Requirements						
1.2	2017-05-04	Marcus Züche	2017-05-04	Marcus Tangermann	2017-05-04	Sten Mückenheim
Update description for U-Boot 2017.01. Change file path by document name. Minor text updates						
1.3	2017-08-07	Marcus Züche				
1.4	2018-05-28	Martin Herberg				
Added description for RZ/N1L, Minor text updates, Updated IAR graphics						
1.5	2018-08-22	Marcus Züche	2018-08-22	Martin Ehlert		
Added description for RZ/N1L RAM; Summary hardware initialization; List the SPI demo application; Fixed names of referenced documents						
1.6	2019-01-07	Marcus Züche				
Update description of running the application on CC and AC, name the GCC version, name the board versions, update boot commands.						
1.4.3 (1.7)	2019-07-09	Martin Ehlert	2019-07-12	Marcus Züche		
Updated example project list and IAR version information						

Sales Offices

Renesas Electronics Corporation

www.renesas.com

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

Renesas Electronics America Inc.
2880 Scott Boulevard Santa Clara, CA 95050-2554, U.S.A.
Tel: +1-408-588-6000, Fax: +1-408-588-6130

Renesas Electronics Canada Limited
1101 Nicholson Road, Newmarket, Ontario L3Y 9C3, Canada
Tel: +1-905-898-5441, Fax: +1-905-898-3220

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.
7th Floor, Quantum Plaza, No.27 ZhiChunLu Haidian District, Beijing 100083, P.R.China
Tel: +86-10-8235-1155, Fax: +86-10-8235-7679

Renesas Electronics (Shanghai) Co., Ltd.
Unit 204, 205, AZIA Center, No.1233 Lujiazui Ring Rd., Pudong District, Shanghai 200120, China
Tel: +86-21-5877-1818, Fax: +86-21-6887-7858 / -7898

Renesas Electronics Hong Kong Limited
Unit 1601-1613, 16/F., Tower 2, Grand Century Place, 193 Prince Edward Road West, Mongkok, Kowloon, Hong Kong
Tel: +852-2886-9318, Fax: +852 2886-9022/9044

Renesas Electronics Taiwan Co., Ltd.
7F, No. 363 Fu Shing North Road Taipei, Taiwan, R.O.C.
Tel: +886-2-8175-9600, Fax: +886 2-8175-9670

Renesas Electronics Singapore Pte. Ltd.
1 harbourFront Avenue, #06-10, keppel Bay Tower, Singapore 098632
Tel: +65-6213-0200, Fax: +65-6278-8001

Renesas Electronics Malaysia Sdn.Bhd.
Unit 906, 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 Korea Co., Ltd.
11F., Samik Lavied' or Bldg., 720-2 Yeoksam-Dong, Kangnam-Ku, Seoul 135-080, Korea
Tel: +82-2-558-3737, Fax: +82-2-558-5141

RZ/N1 Group

Renesas Electronics Corporation