RZ/N1L Group

CONNECT IT! ETHERNET RZ/N1L-DB

Introduction

RZ/N1 is a family of scalable ARM® based industrial Ethernet switches. Due to the integrated performant ARM® Cortex®-A7 CPU, they are ideal choice for building products such as Programmable Logic Controllers (PLC), complex remote-IO slaves, industrial switches, gateways or operator terminals.

RZ/N1 Family consists of three devices, out of which the most compact of them is the RZ/N1L, a 3-port, industrial ethernet communication System on Chip in a 196BGA Package.

This guide shall assist you to quickly start with your evaluation of the RZ/N1**L** device on the related Solution Kit board(s) and the Solution Kit software and documentation. It represents a short guide for the first-time users with the instructions for the first startup of the board.

In addition to this document, please be aware of the extensive documentation on RZ/N1 devices, evaluation boards and software that you may find in the Solution Kit Documentation directory.

The Solution Kit boards can solely be used for evaluation purposes. It is not allowed to use the boards for commercial purposes.

List of reference documents

Document name
User's Manual: System Introduction, Multiplexing, Electrical and Mechanical Information
User's Manual: System Control and Peripheral
User's Manual: Peripherals
User's Manual: R-IN Engine and Ethernet Peripherals
User's Manual: Generic Open Abstraction Layer
RZ/N1L Development Board Schematic
RZ/N1L Development Board Setup Notes

List of Abbreviations and Acronyms

Abbreviation	Full Form
API	Application Programming Interface
BSP	Board Support Package
CO	CANopen Bus Protocol
DLR	Device Level Ring Protocol
EtherCAT / ECAT	EtherCAT Network Protocol
EtherNet/IP / EIP	EtherNet/IP Network Protocol
EPL	Ethernet POWERLINK Network Protocol
PNIO	PROFINET IO Protocol
PROFINET	Process Field Network Protocol
lwIP	Lightweight IP Protocol
GOAL	Generic OS Abstraction Layer, Communication Framework on RZ/N1x
LLDP	Link Layer Discovery Protocol
NVM	Non-volatile memory
OS	Operating System
RT	Real-Time
API	Application Programming Interface
HW	Hardware
SW	Software
HSR	High-Availability Seamless Redundancy Protocol
PRP	Parallel Redundancy Protocol
IBIS	I/O Buffer Information Specification for pin signal simulation

1. Overview

The following chapters will help you setup the hardware and software environment and run a sample application on your RZ/N1L evaluation board.

- The Solution Kit is being actively updated and new versions of it are iteratively released
- Current version of Solution Kit provides a snapshot of the latest software and documentation versions at the release time

The **CONNECT IT! ETHERNET RZ/N1L** Solution Kit is based on the RZ/N1L, a 3-port, industrial Ethernet communication SoC in a 196BGA package. In contrast to the RZ/N1D and RZ/N1S the device has just the communication block based on the proven multi-protocol R-IN Engine with 6MB internal SRAM (no ARM® Cortex®-A7 CPU and no external DDR interface). Please refer to the RZ/N1 device documentation for further differences.

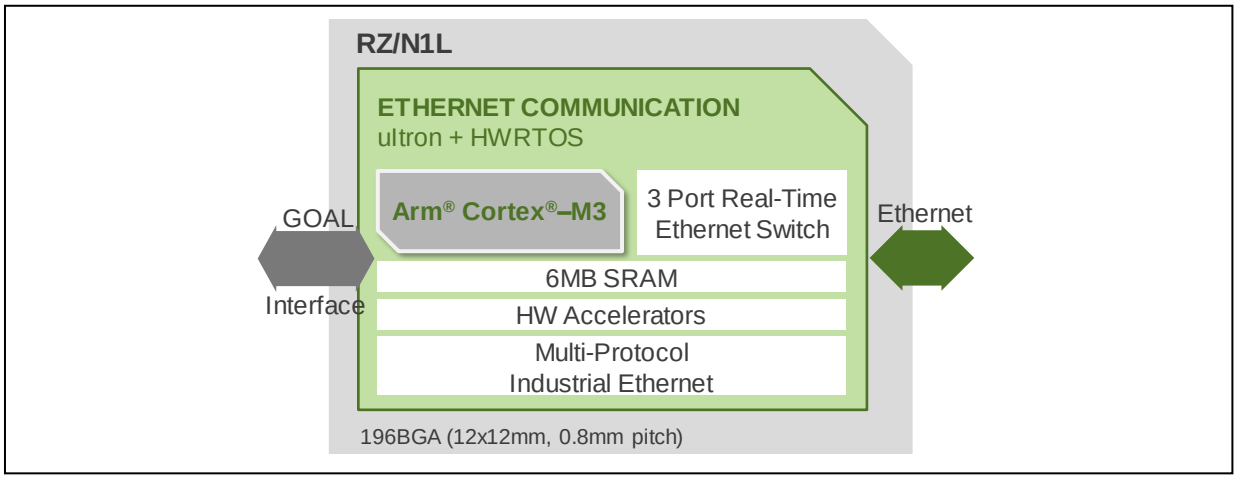


Figure 1-1: Single core for Simple Multi-protocol Implementation

The solution kit contains

- RZ/N1L CPU Development Board
- IAR I-jet Lite debugger (incl. 20pin flat ribbon and USB-micro/A cable)
- USB Cable Micro / Type-A
- DVD RZ/N1L-DB Solution Kit CD for all RZ/N1x devices / boards

2. Hardware Setup

To get started quickly with the board, please identify the connectors on the RZ/N1L-DB CPU Board.

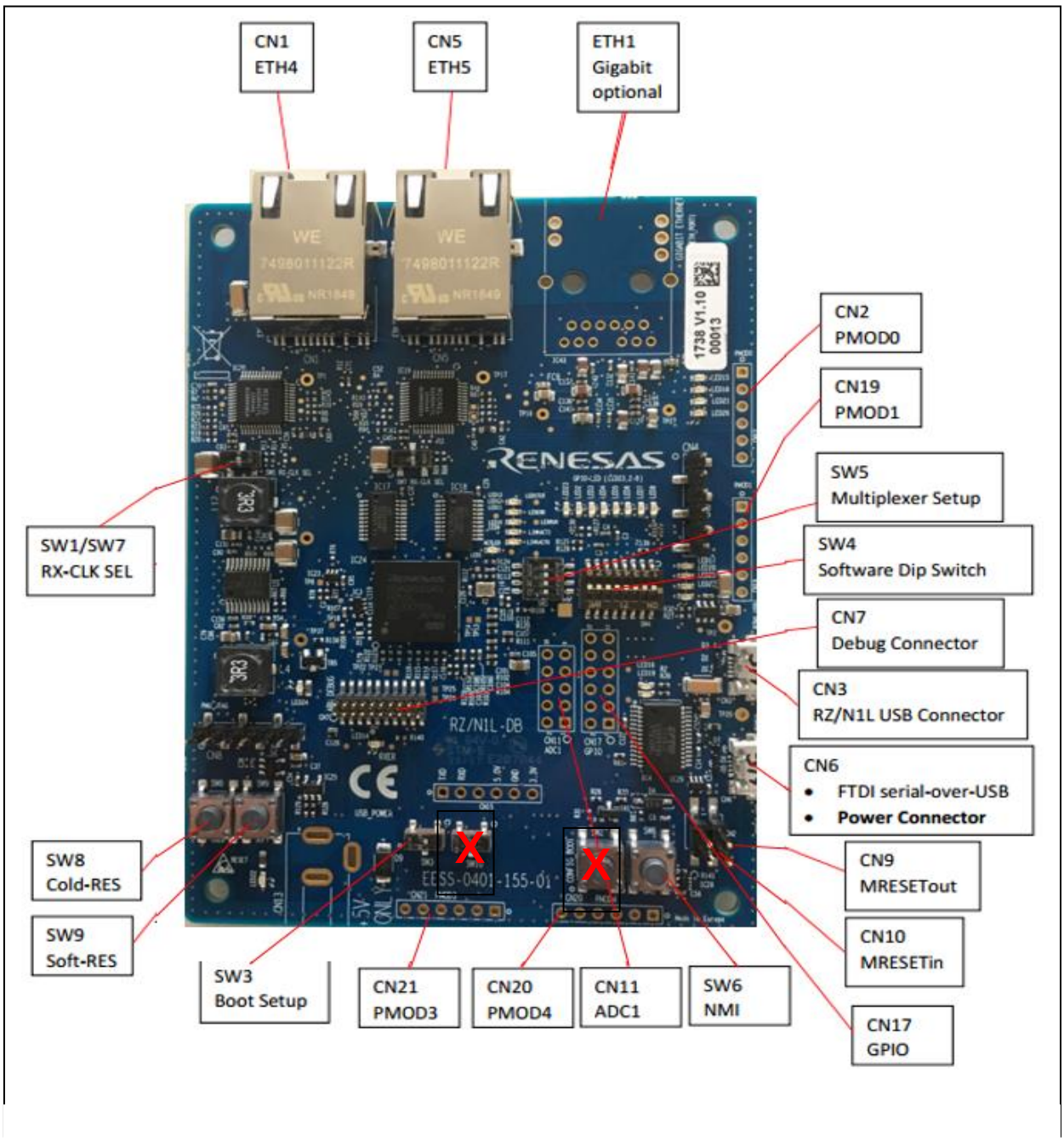


Figure 2-1: Board connector overview

1. Switches and Jumpers

Please take care to set the following switches as shown below

SW5: GPIO Multiplexer Setup

Number	SW ON (low)	SW OFF (high)	Default Setting
1	PMOD	RMII/MII	OFF
2	I2C Cat	I2C RZ/N1	OFF
3	Segger Debugger	IAR I-Jet Debugger	OFF
4	USB OTG Mode	USB Peripheral Mode	OFF

Other switches

Switch	SW ON (white bar)	SW OFF	Default Setting
SW3	JTAG Mode	ARM Coresight Mode	OFF
SW1	RXCLK4 from PHY	RXCLK4 from GPIO61	**ON**
SW7	RXCLK5 from PHY	RXCLK5 from GPIO61	**ON**

The Jumper **CN10** needs to be **ON** to assure that RZ/N1L is connected with the SRESET from the debugger, otherwise the debugger cannot reset the CPU.

Jumper	SW ON (white bar)	SW OFF	Default Setting
CN9	MRESETOUT is connected to the Debugger	Debugger cannot detect any internal CPU reset	OFF

2. Push buttons

- **SW8** is hard reset (POR) button
- **SW9** is soft reset button
- **SW6** generates CM3 NMI signal when ON.

3. JTAG Debug Connector

This connection is required for software development purposes, to be able to flash/load the software and debug it on the target. The IAR Debugger cable has one unused key pin, therefore pin 7 of the **CN7** connector has to be cut or bent, as shown below. After this step, you can attach your I-jet debugger to the board via JTAG cable to the PC via USB.

For more information on Board Setup, please refer to the board setup notes …\YCONNECT-IT-RZN_V1.x\Documents\RZN.Boards\RZN1L-DB\RZ_N1L_DB_Board_Setup_Notes.Vxxx.pdf

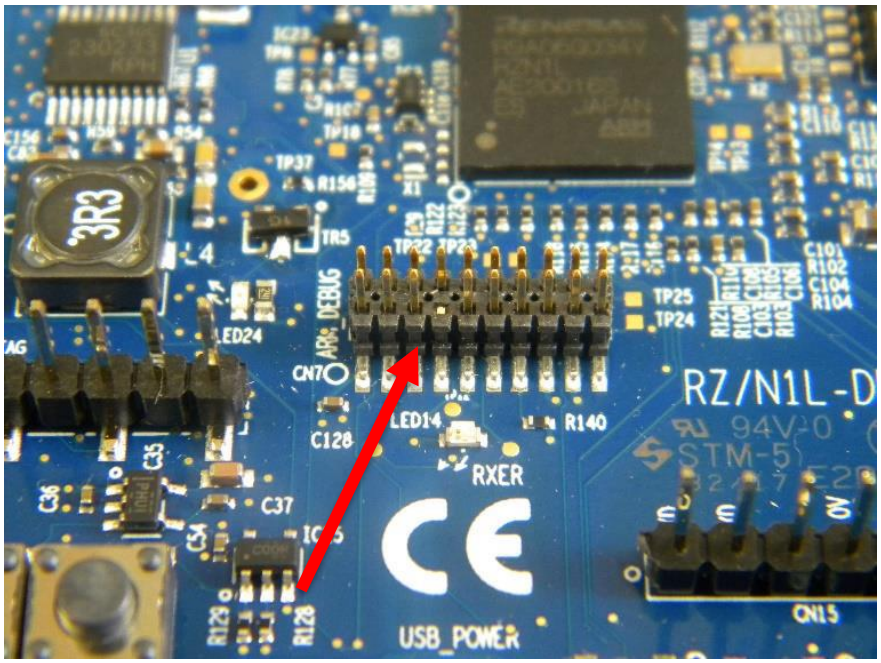


Figure 2-2: Connector CN10 with removed pin 7

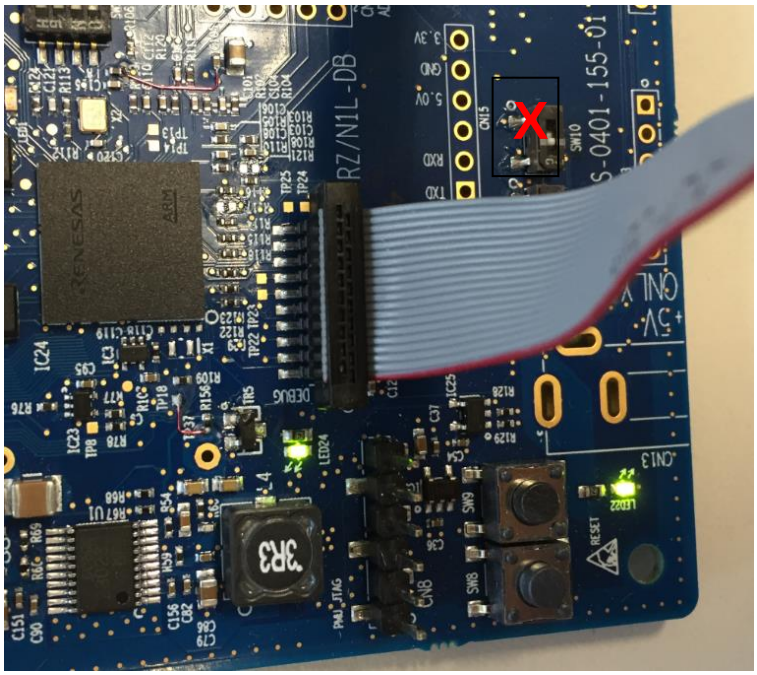


Figure 2-3: I-Jet Debugger connection

4. **Power connector and serial-over-USB**
   The board is powered via USB connector CN6. Please connect the board to your PC via USB cable to the connector **CN6**. The same interface is used for serial UART communication with the board, so if you have a windows PC, after connecting the board via USB, you should be able to see a new device registered on one of the USB Serial COM ports. On the linux host machines, the serial-over-usb device should be accessed using /dev/ttyUSB0, provided you do not have any other USB devices attached. You may use any of the terminal emulator on your PC to open a serial port connection to the board.

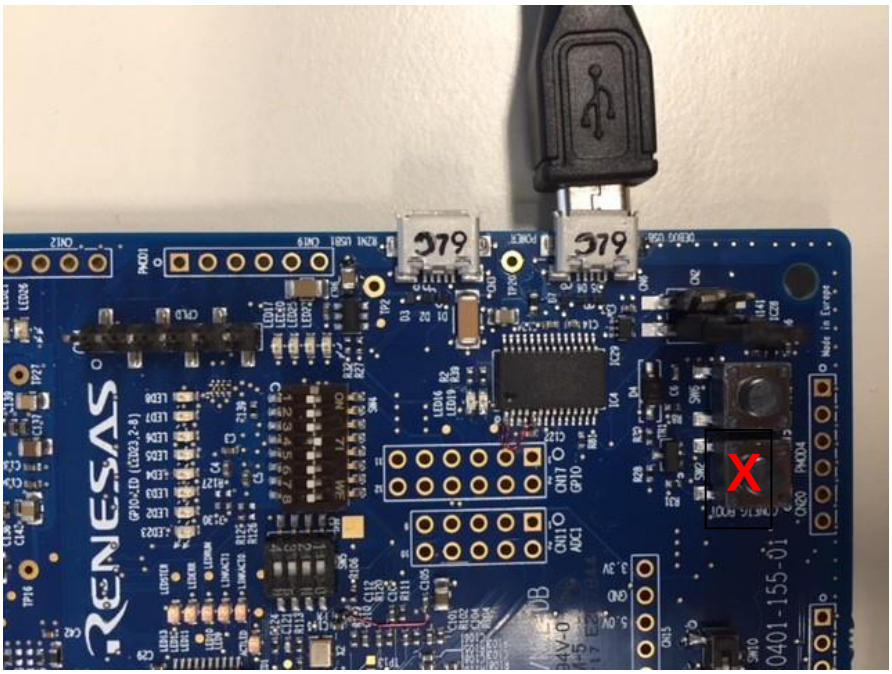


Figure 2-4: Power connector and serial over USB

If the board is not recognized by Windows, please install the FTDI driver that you may find under …\YCONNECT-IT-RZN_V1.x\Tools\FTDI.

If the required 3.3V are supplied to RZ/N1L, the green LED24 will be lit and stay ON.

You can perform a soft-reset of the board by pressing **SW9**.

5. Ethernet Interface

The Ethernet connection to the PC or a PLC can be prepared using one of the two RJ-45 connectors CN5 or CN1, as shown in

Figure 2-5: Ethernet RJ-45 connectors

Figure 2-5. For running the EtherCAT sample program you can use any of the two connectors. There are two PHY Micrel chips connected to these RJ-45 connectors.

Figure 2-5: Ethernet RJ-45 connectors

3. Software Setup

RZ/N1L Solution Kit contains a software package on a DVD that enables the user to evaluate various functionalities of our industrial automation device.

The software for RZ/N1L comes as IAR compilable code and you need IAR Embedded Workbench for ARM (EWARM) for compiling and running the software, but also for flashing the software to the external QSPI flash device on the board.

1. **Install IAR EWARM**
   As the first step in software setup, please install the IAR EWARM -> installation file for IAR EWARM 8.22.1 can be found under **YCONNECT-IT-RZN_V1.x\Tools\IAREWARM**.

   Please note that the current RZ/N1L software from the Solution Kit was tested only with IAR EWARM v8.22.1. During the installation process, you will be prompted to select your license type. For evaluation purposes, you can first choose the 30-day license.

2. **Locate the RZ/N1L IAR software projects in the Solution Kit**
   The software is stored in the folder **YCONNECT-IT-RZN_V1.x\Software** in the Solution Kit. This folder consists of software packages for all RZ/N1 platforms, not only for RZ/N1L. We provide below a list of IAR EWARM projects that are designed/adapted for RZ/N1L. More detailed software documentation can be found under **YCONNECT-IT-RZN_V1.x\Documents\RZN.Software**. The IAR projects represent a collection of required source files to build a Cortex M3 executable and run it on the target.

   A list of some available examples of RZ/N1**L** IAR Projects:

GOAL sample IAR Projects	**Description**	**Path to the workspace file in the Sol. Kit** YCONNECT-IT-RZN_V1.x\Software\GOAL\goal\projects\...
Chase Lights	User LEDs 1..8 blinking cyclically from one to other direction	**…**\00410_goal**chase_lights**\ iar\7_70\rzn1l_demo_board\rzn1l_demo_board.eww
Simple HTTP Web Server		**…\goal_http**\01_get\ iar\7_70\rzn1l_demo_board\rzn1l_demo_board.eww
Eth. Interface Info	Displays the status changes on the Ethernet Interface over UART in the serial console	**…**\00410_goal**iface_info**\ iar\7_70\rzn1l_demo_board\rzn1l_demo_board.eww
TCP Server	Runs a TCP server on the board, it can be accessed via port 1234, default IP address 192.168.0.10	**…**\00410_goal**tcp_server**\ iar\7_70\rzn1l_demo_board\rzn1l_demo_board.eww
Template	Template project, for the users to base their application development on	**…**\00410_goal**template**\ iar\7_70\rzn1l_demo_board\rzn1l_demo_board.eww
UDP Receiver	Prints out the received UDP telegram in the console	**…**\00410_goal**udp_receive**\ iar\7_70\rzn1l_demo_board\rzn1l_demo_board.eww

TCP Server and UDP receiver can be tested by sending TCP traffic to the address 192.168.0.10, port 1234. One well known cross-platform tool for network performance measurements is iperf. There is also a graphical frontend for the tool, that is called *jperf* and can be used for generating traffic from your PC to the board.

For more information on GOAL sample projects, please refer to the

\YCONNECT-IT-RZN_V1.X\Software\GOAL\doc\r11qs0008ed0131-rzn1-goal-quick-startguide-management.pdf document.

Protocol stack IAR Projects	**Description**	**Path to the workspace file in the Sol. Kit** YCONNECT-IT-RZN_V1.x\Software\GOAL\goal\projects\...
EtherCAT Slave	Runs a sample EtherCAT slave application, which is running an I/O communication between EtherCAT master and RZ/N1L	…**goal_ecat**\01_io_data\ iar\7_70\rzn1l_demo_board_eb\rzn1l_demo_board.eww
PROFINET Slave	Runs a sample PROFINET RT slave application	…**goal_pnio_lib**\10_led_demo\ iar\7_70\rzn1l_demo_board\rzn1l_demo_board.eww
EtherNet/IP Slave	Runs a sample EtherNet/IP slave application	…**goal_eip_lib**\01_simple_io\ iar\7_70\rzn1l_demo_board\rzn1l_demo_board.eww
Modbus TCP	Runs a sample Modbus TCP slave application	…**goal_mbs**\01_tcp_server\ iar\7_70\rzn1l_demo_board\rzn1l_demo_board.eww

Please check the protocol stack application notes in the folder :

\YCONNECT-IT-RZN_V1.X\Software\GOAL\doc\

Protocol Stack	**Application note**
EtherCAT Slave	r01an4239ej0090-rzn1-ethercat.pdf
PROFINET Slave	r11an0207ed0131-rzn1-goal-user-manual-profinet.pdf r11qs0010ed0131-rzn1-goal-quick-startguide-profinet.pdf
EtherNet/IP Slave	r11an0205ed0131-rzn1-goal-user-manual-eip.pdf r11qs0007ed0131-rzn1-goal-quick-startguide-eip.pdf
Modbus TCP	r01an4412ej0090-rzn1-modbus.pdf

Protocol stack IAR Projects	**Description**	**Path to the workspace file in the Sol. Kit** YCONNECT-IT-RZN_V1.x\Software\...
OS-Less Driver Sample Application**	Sample UART user interface application for controlling user LEDs and reading out dip switch status	\BaremetalDrivers\renesas_app**led_blink**\ iar\rzn1l_demo_board\rzn1l_demo_board.eww

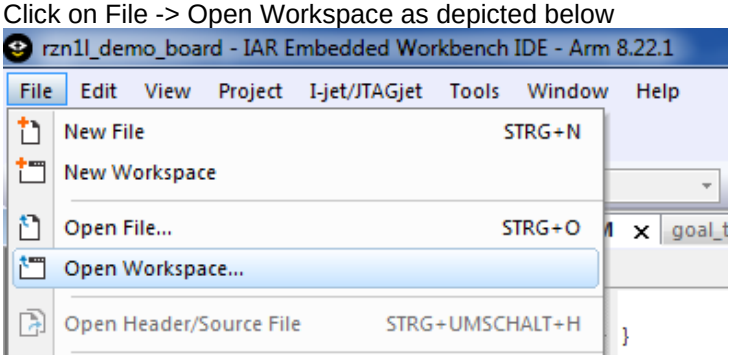
The Application Note for Baremetal drivers you may find in

\YCONNECT-IT-RZN_V1.X\Software\BaremetalDrivers\r11an0282ej0001-rzn1-baremetal-drivers.pdf

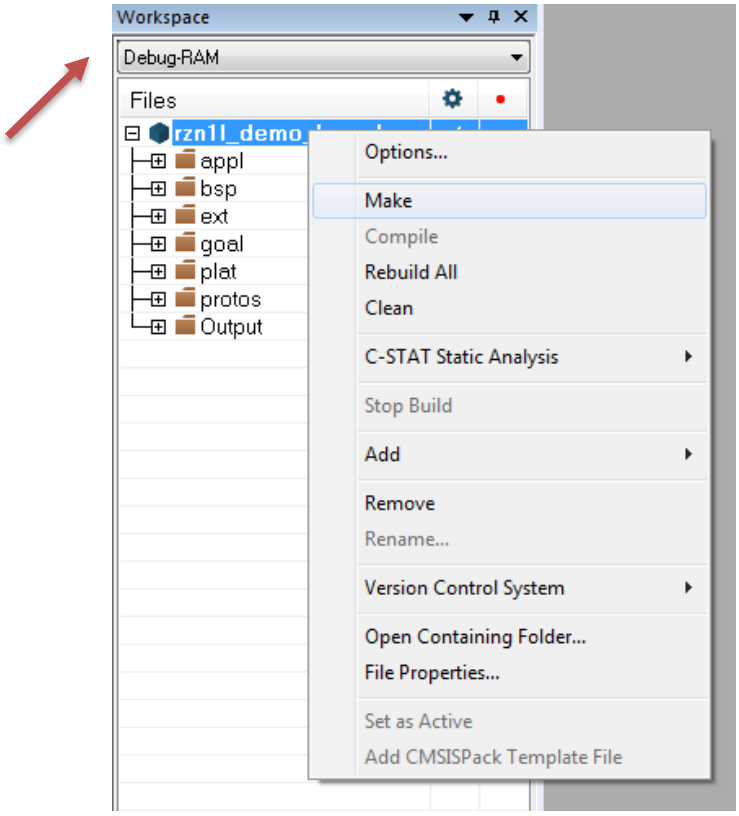
3.1 Running a sample application on the RZ/N1L

It is straightforward to run an application on the RZ/N1L. Provided you followed the steps in the hardware setup chapter you need to follow the steps below to load the Cortex M3 image in SRAM and execute it from there. This chapter describes the steps based on the chase_lights example, but the instructions are applicable in the case of any of the above mentioned IAR projects.

1. Open the IAR EWARM that you previously installed
2. Click on File -> Open Workspace as depicted below

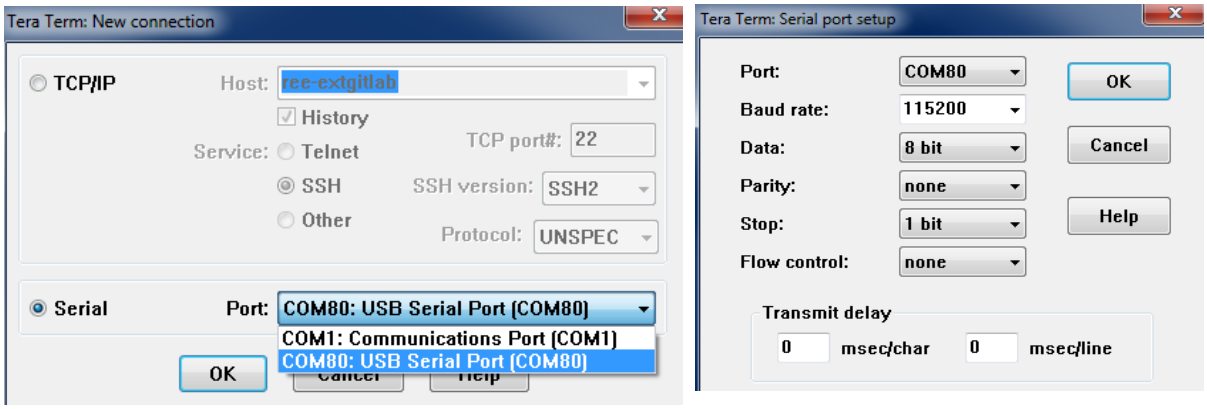


3. Select the workspace file
   *YCONNECT-IT-RZN_V1.x\Software\GOAL\goal\projects\00410_goal**chase_lights**\iar\7_70\rzn1l_demo_board\rzn1l_demo_board.eww*
4. You can see now a workspace with a project inside called rzn1l_demo_board. Select the "*Debug-RAM"* project configuration from the above panel as shown below

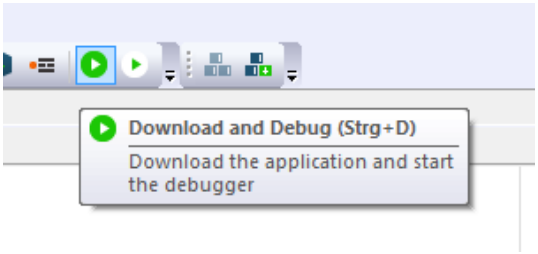


5. Right-click on the project name and click on "Make".

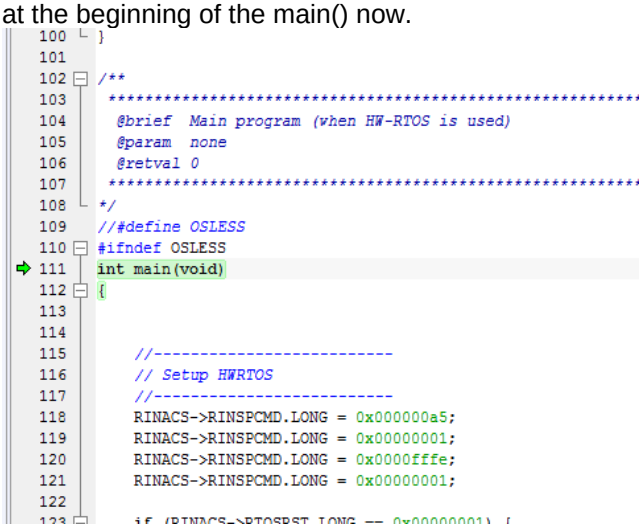
6. After the project compiled with no errors nor warnings, make sure your board is powered on and that the IAR I-Jet Debugger is connected to the JTAG connector.
7. Open a terminal emulator client like TeraTerm or putty, choose the corresponding serial port, where you board is registered to and set the following serial port settings. (COM port number may vary, please adapt to your environment settings).



8. Go back to the IAR EWARM and click on the “Download and Debug” button in the program control panel, as shown below.

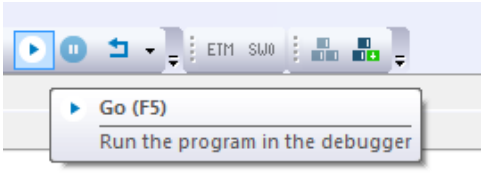


9. In the Debug Log window there should be no errors nor warnings and the program should be stopped at the beginning of the main() now.



```
100 }
101
102 /**
103  ************************************************************
104   @brief  Main program (when HW-RTOS is used)
105   @param  none
106   @retval 0
107  ************************************************************
108 */
109 //#define OSLESS
110 #ifndef OSLESS
111 int main(void)
112 {
113
114
115     //--------------------------
116     // Setup HWRTOS
117     //--------------------------
118     RINACS->RINSPCMD.LONG = 0x000000a5;
119     RINACS->RINSPCMD.LONG = 0x00000001;
120     RINACS->RINSPCMD.LONG = 0x0000fffe;
121     RINACS->RINSPCMD.LONG = 0x00000001;
122
123     if (RINACS->RTOSRST.LONG == 0x00000001) {
```

10. At this stage you can click F5 on your keyboard or “Go” button in the IAR program flow panel.

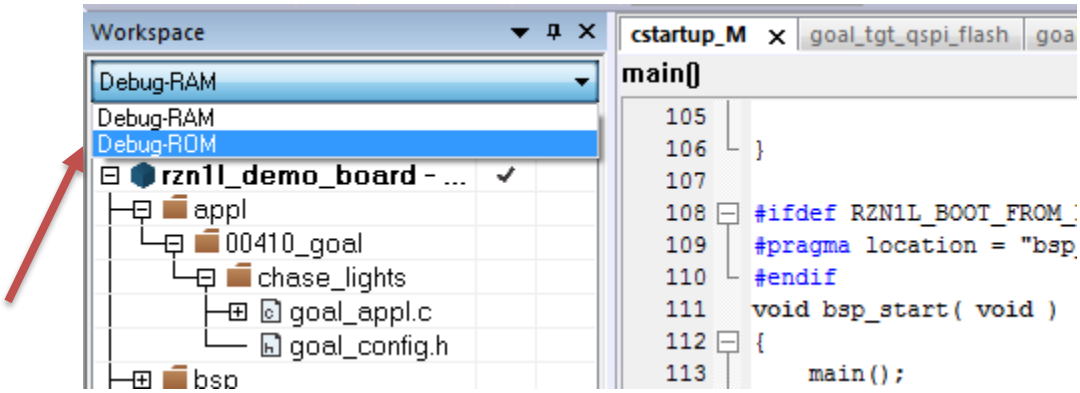


The program should be running now on your board, the LEDs are toggling from one direction to another and you should see some info output on your serial console as shown below.

```
[CC_I|goal_lmLogLegacy:1152] [I|goal_cmInit:209] Calculated config size of 656 in 3 modules
[CC_I|goal_lmLogLegacy:1152] [I|goal_queuePoolBufsReq:776] ID(35) requests buffers[1556]
[CC_I|goal_lmLogLegacy:1152] [I|goal_queuePoolBufsReq:777]        fixed/temp = 1, 0
[CC_I|goal_lmLogLegacy:1152] [I|goal_queuePoolBufsReq:776] ID(35) requests buffers[1556]
[CC_I|goal_lmLogLegacy:1152] [I|goal_queuePoolBufsReq:777]        fixed/temp = 2, 0
[CC_I|goal_lmLogLegacy:1152] [I|goal_queuePoolBufsReq:776] ID(35) requests buffers[1556]
[CC_I|goal_lmLogLegacy:1152] [I|goal_queuePoolBufsReq:777]        fixed/temp = 1, 0
[CC_I|goal_lmLogLegacy:1152] [I|goal_taskCreate:74] creating task: Timer
[CC_I|goal_lmLogLegacy:1152] [I|goal_init:190] GOAL initialized
[CC_I|goal_lmLogLegacy:1152] [I|appl_setup:85] Chase Lights application started.
[CC_I|goal_lmLogLegacy:1152] [I|goal_memInitDone:128] fixed memory usage: 16020/196608 bytes
[CC_I|goal_lmLogLegacy:1152] [I|goal_memInitDone:129] fixed memory usage: (9%)
```

3.2 Write the sample application in QSPI flash on the board

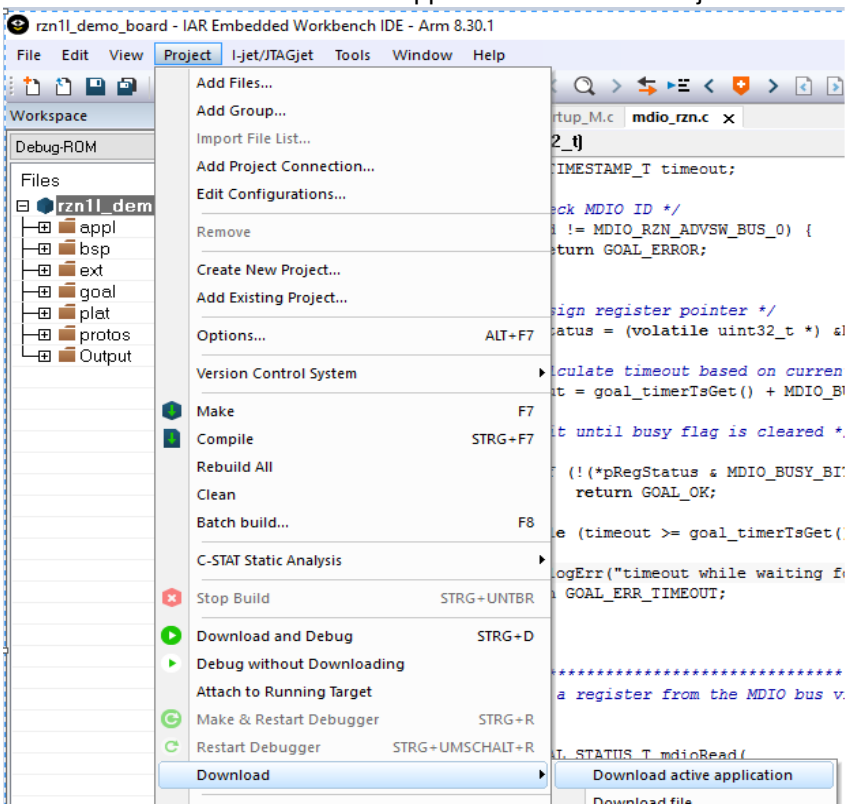
To write your binary in QSPI Flash and be able to execute it from there after a reset, we have provided a "Debug-ROM" project configuration with every RZ/N1L project. Please select the "*Debug-ROM*" project configuration from the panel above the project tree.



This project configuration allows you to compile your code for execution from the SPI Flash. RZ/N1L comes out of reset and starts executing code from QSPI. It can do this because the QSPI controller provides a memory-mapped interface and is set up with commands and clocks to access any QSPI.

In order to do this, please follow the next steps.

1. Select your rzn1l_demo_board application in the "*Debug-ROM*" configuration
2. Build this project in the same manner as described in previous chapter, by clicking on "Make"
3. Now select "Download active application" from IAR Project menu



4. After download has finished press the reset button SW-9 (see Figure 3-1).
   You should see LEDs blinking in couple of seconds after this.

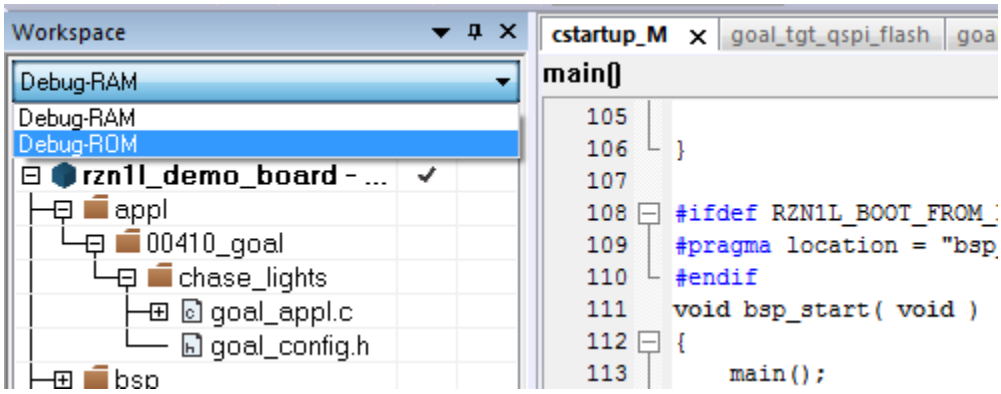
Important notes:

1. Since in this mode, the QSPI is operated at 5.2MHz with 1-bit wide data and the Cortex M3 does not have any caches, there will be a certain latency for loading the data from QSPI to SRAM. Therefore, compared to the direct RAM execution, booting will take couple of seconds when running from QSPI. This can be mitigated by configuring the QSPI speed prior to loading the program to SRAM

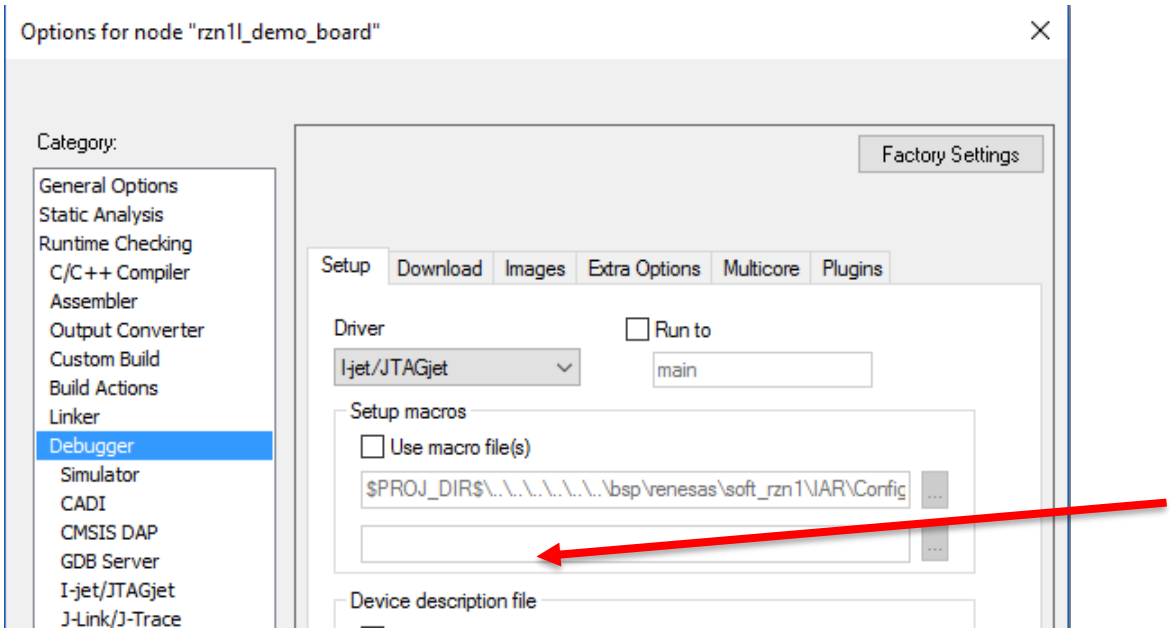
For the list of the known restrictions, please refer to the chapter 1.8 in the Solution Kit release notes – Readme.Release.V131, that can be found in the root directory of the solution kit.

3.3 Using Debug-ROM

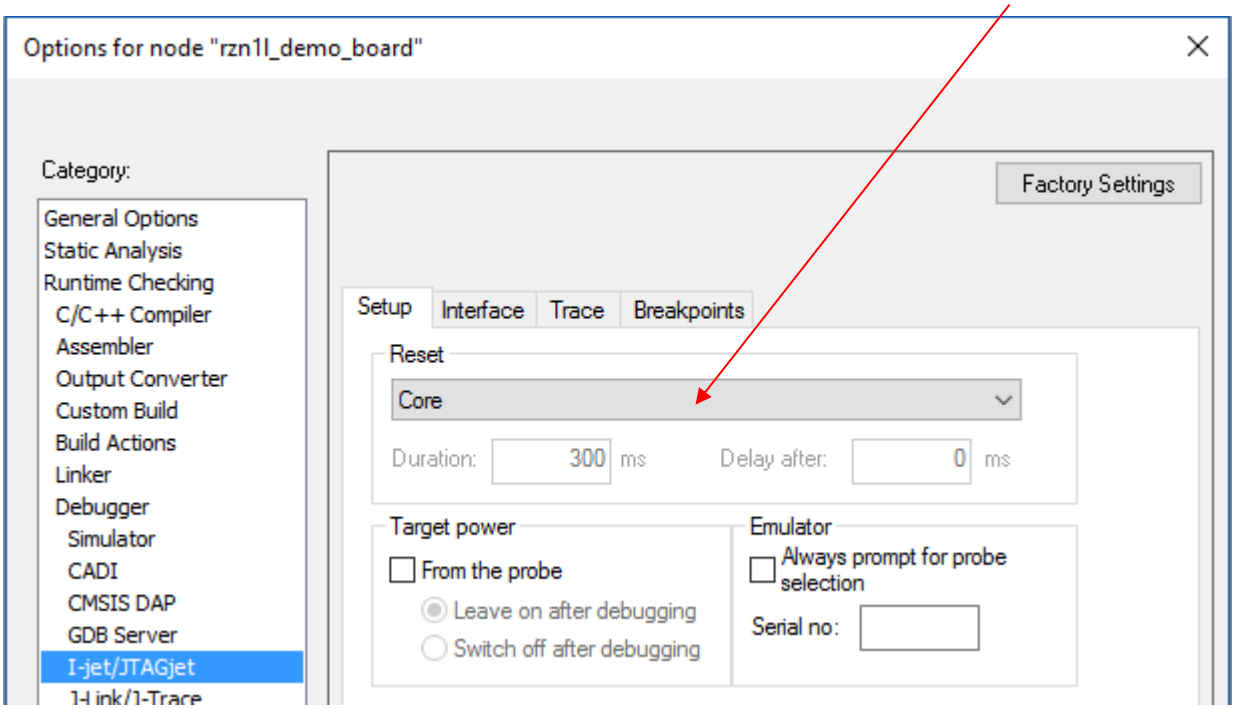
1) Make sure the program got loaded into QSPI flash (see section: 3.2 above)
2) Please select the "*Debug-ROM*" project configuration from the panel above the project tree.



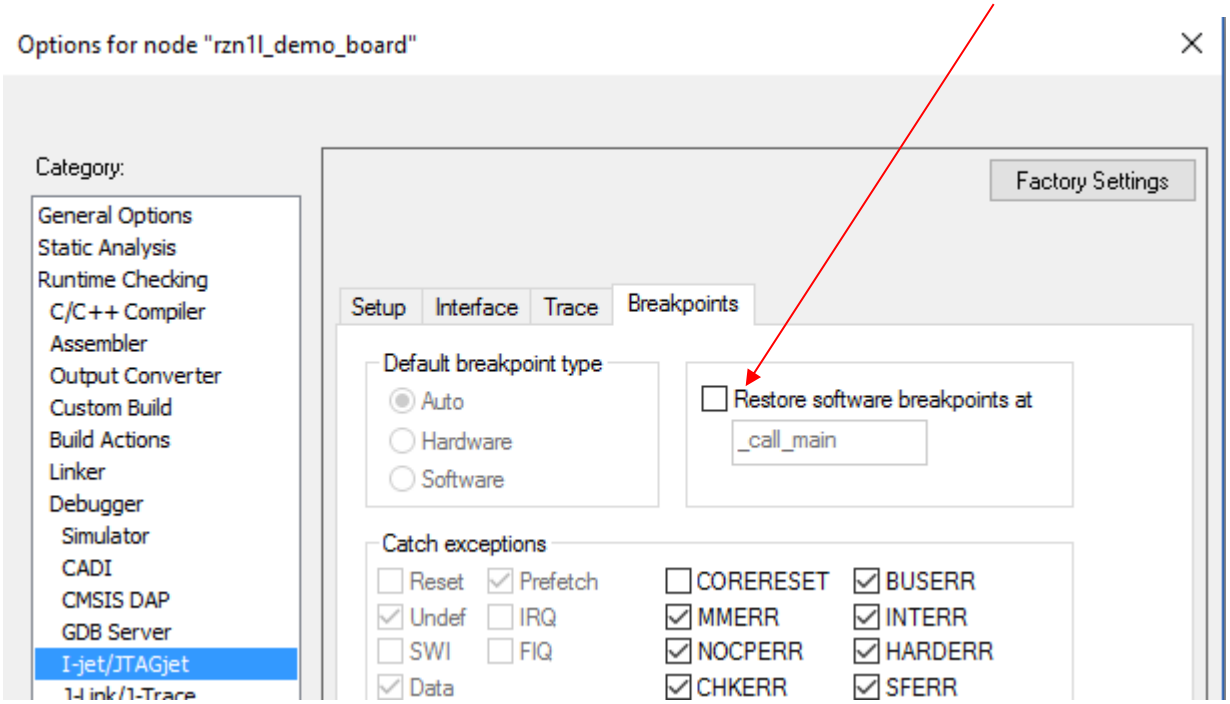
3) Now select "Options…" from the Project menu (shortcut ALT+F7) and choose "Debugger" from the Category selection, there you must disable "Use macro file(s)".



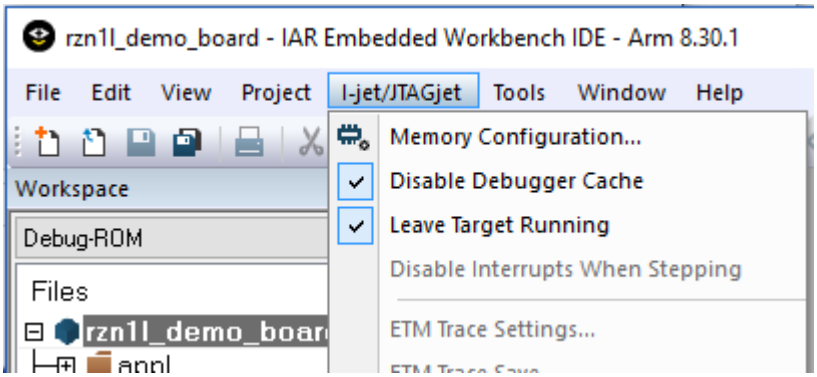
4) On Category I-Jet/JTAGjet select “Core” from the Reset pulldown in section “Setup”.



5) Ensure that you did not restore any software breakpoint in section “Breakpoints”

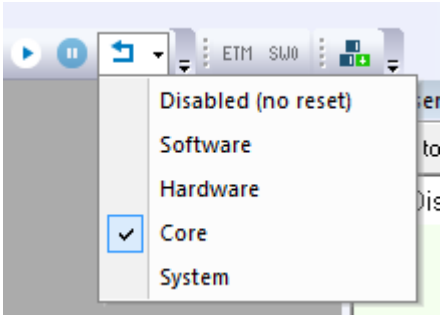


6) From the I-Jet/JTAGjet menu select “Disable Debugger Cache” and “Leave Target Running”.



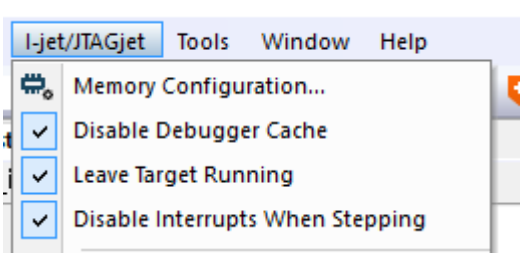
7) Now connect to the target by clicking  (shortcut: Strg+D), you will land on a random location, don’t do anything here and proceed with step 8.

8) On the reset button  select “Core” by clicking onto the little black triangle.



9) Now reset the system for to start QSPI flash debug by clicking onto the blue arrow  . You will land on “void __iar_program_start(void)”, please look at the Disassembly window, as you can see the instruction pointed to resides at a location starting with 0x1000…. This is the QSPI area, if you want to debug something here you must step through the assembly language. Stepping through the high-level Source-Code will most likely not succeed, because the next few lines will transfer the content from flash to RAM and handover the control to the Hardware RTOS. For that reason, High-level debugging will lose track here.

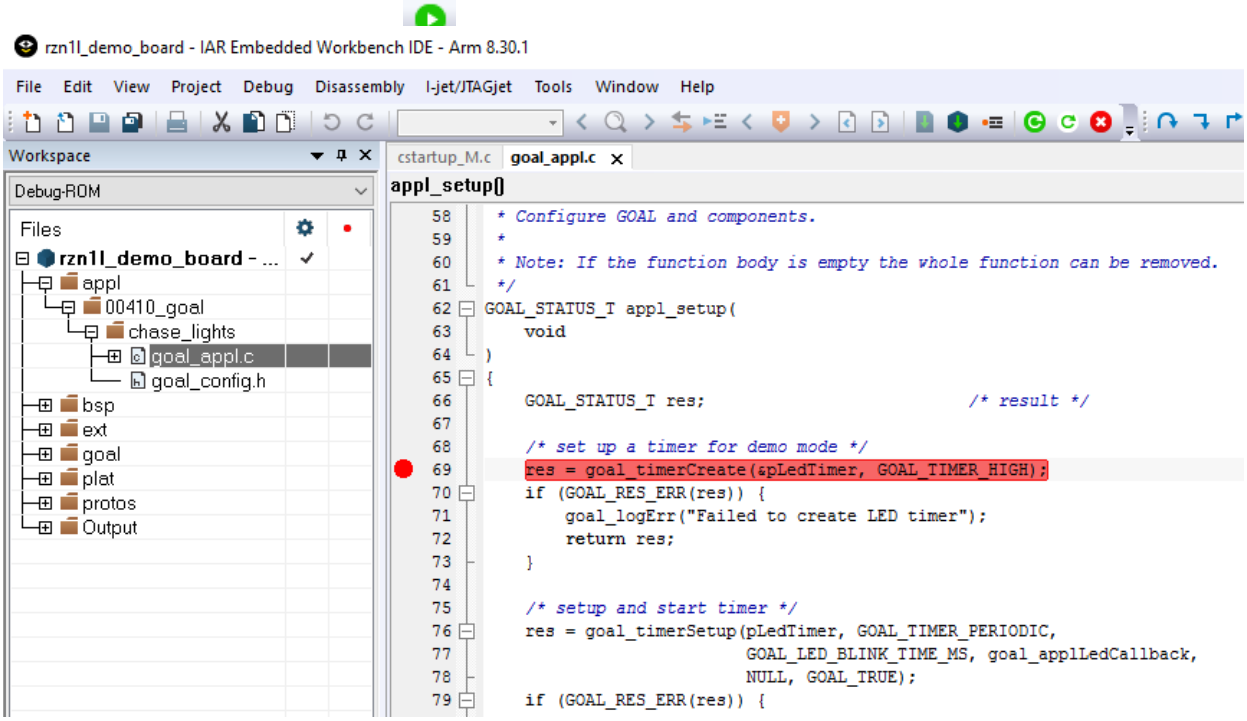
10) Now select “Disable Interrupts When Stepping” from the I-Jet/JTAGjet menu.



11) Set a proper breakpoint. A proper breakpoint is a line of code you wrote by yourself, or with other words a location that starts with 0x400… a.k.a. RAM in the disassembly window.

For the sake of simplicity let's assume you wrote the ***chase_lights*** application.
Set a breakpoint at line 69 in file: goal_appl.c, since this is the first line of user code that get executed by the HWRTOS.

Run to the breakpoint by clicking (shortcut: Strg+D).



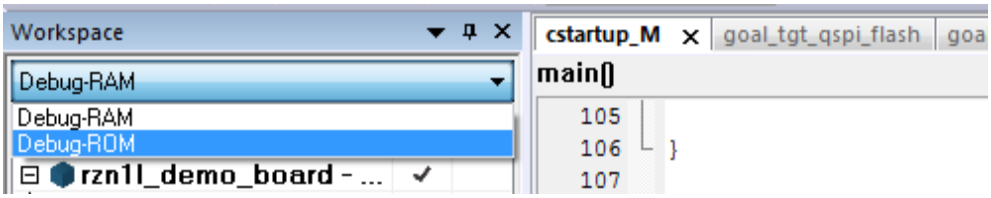
Important notice

Don't place any breakpoint in the file cstartup_M.c this includes stepping with the debugger. Only place breakpoints at user application code. To place a breakpoint somewhere else you have to understand every single line of machine code that got executed to the full extend. Otherwise you will rise an exception caused by a breakpoint that disturbed.
If you disturb the flash loader during transfer from flash to RAM you will end in an appropriate exception handler i.e. "dummy_handler_rom". This is not a bug, but an indicator that you missed to place a proper breakpoint.

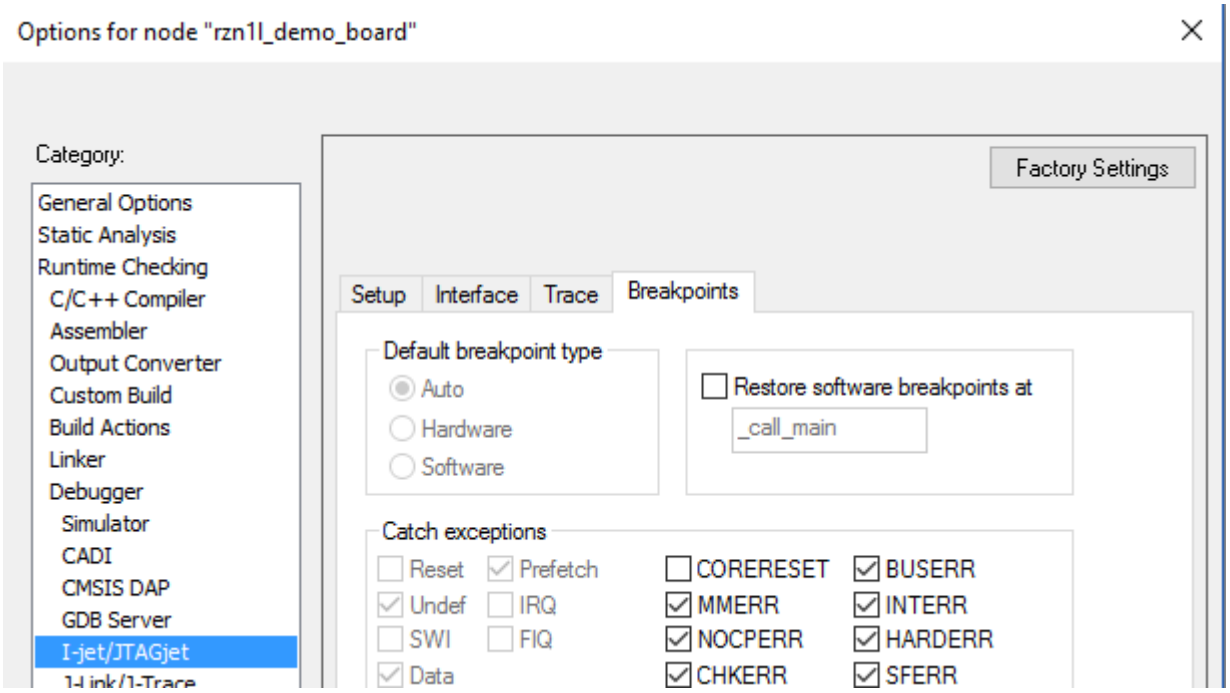
3.4 Attach to a running target

Sometimes you need to see what is going on, on a running target that misbehaves.
Before you can attach to a running target you must set the following options:

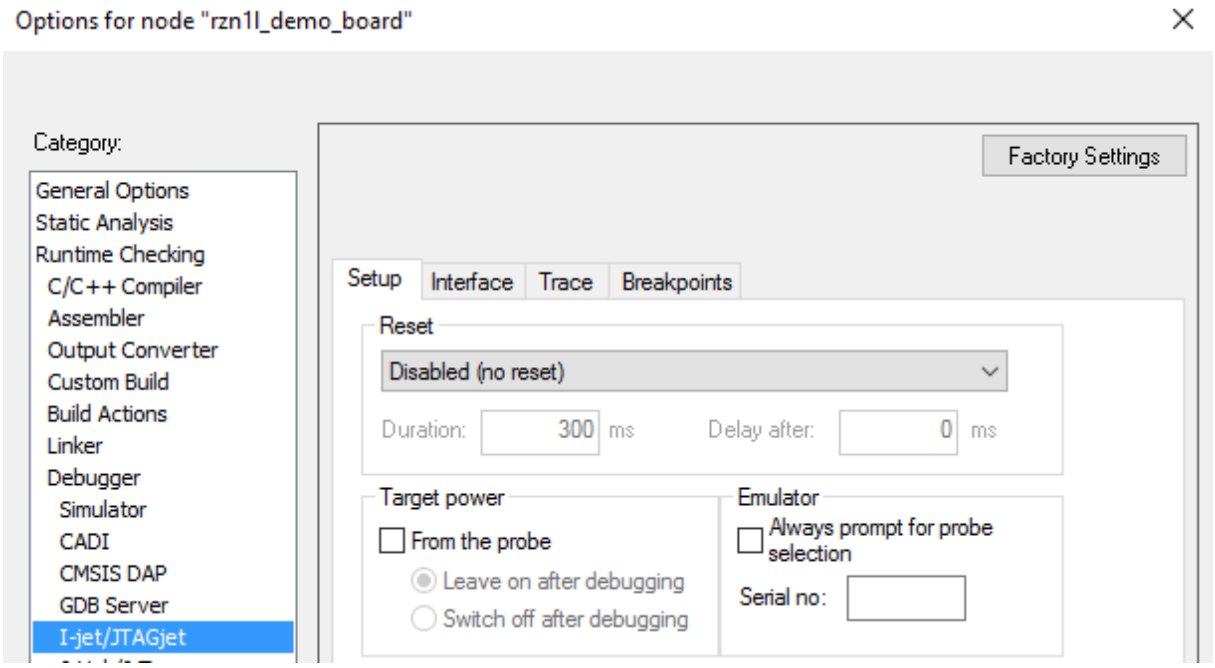
1) Please select the "*Debug-ROM*" project configuration from the panel above the project tree.



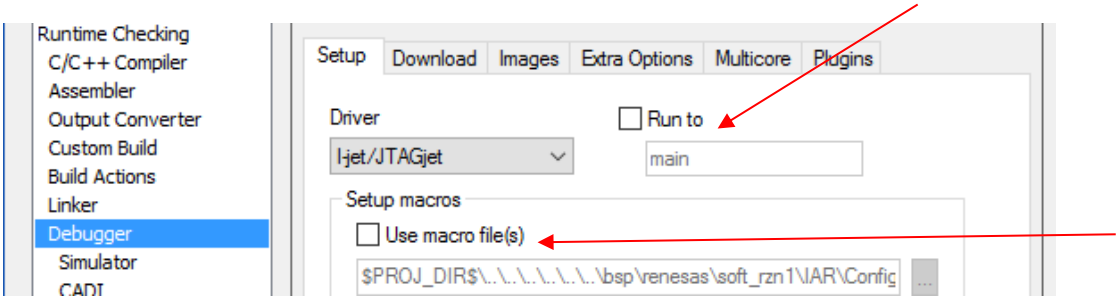
2) Select "Options…" from the Project menu (shortcut ALT+F7) and choose I-jet/JTAGjet from the Categories and verify that "Restore software breakpoints at" is disabled.



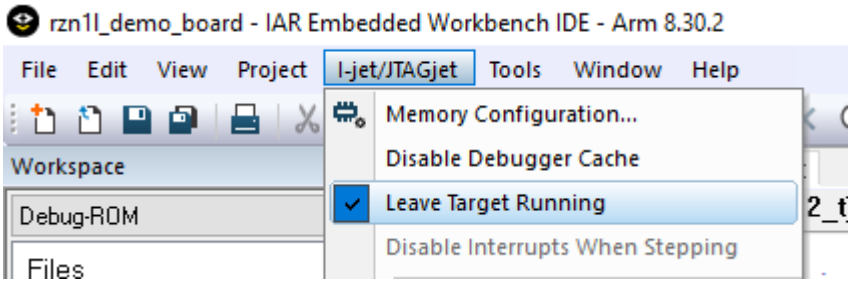
3) From the “I-Jet/JTAGjet” category select “Disable (no reset)”



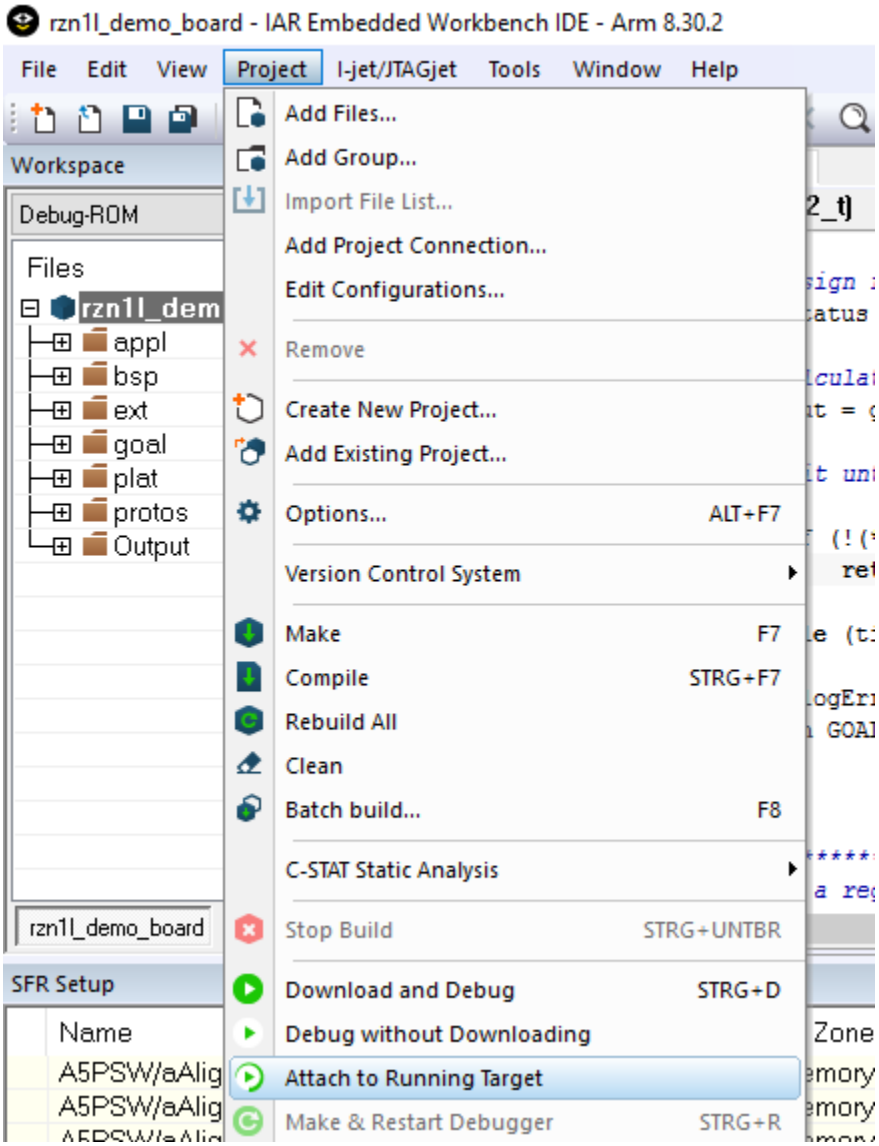
4) Proceed with category “Debugger” and disable “Run to” and “Use macro file(s)”



5) From the I-Jet/JTAGjet menu select “Leave Target Running”.



6) Assume your state is: Program is in flash you did invoke a software reset and verified that the program is running, now from the “Project” menu select “Attach to running target”.



Now you are connected to the program that got loaded from flash and is currently running in RAM.
Click  and you will land at the current executing line.

Revision History

Rev.	Date	Description	
		Page	Summary
0.1	20.Feb.2018	-	Draft – first revision
0.2	31.Jul.2018	-	Updates for the Solution Kit v1.3.1
0.3	06.Sep.2018	-	Minor image modifications applying the changes for removed components on board EESS-0401-155-02
0.4	22.Oct.2018	-	Added section 3.3 changed chapter 3.2
0.5	23.Oct.2018	-	Added section 3.4
0.6	24.Sep.2019	-	Modified licensing description
0.7	31.Jan.2020	-	Changed format

General Precautions in the Handling of Microprocessing Unit and Microcontroller Unit Products

The following usage notes are applicable to all Microprocessing unit and Microcontroller unit 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. Precaution against Electrostatic Discharge (ESD)

   A strong electrical field, when exposed to a CMOS device, can cause destruction of the gate oxide and ultimately degrade the device operation. Steps must be taken to stop the generation of static electricity as much as possible, and quickly dissipate it when it occurs. Environmental control must be adequate. When it is dry, a humidifier should be used. This is recommended to avoid using insulators that can easily build up static electricity. Semiconductor devices must be stored and transported in an anti-static container, static shielding bag or conductive material. All test and measurement tools including work benches and floors must be grounded. The operator must also be grounded using a wrist strap. Semiconductor devices must not be touched with bare hands. Similar precautions must be taken for printed circuit boards with mounted semiconductor devices.
2. Processing at power-on

   The state of the product is undefined at the time 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 time 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 time 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 time when power is supplied until the power reaches the level at which resetting is specified.
3. Input of signal during power-off state

   Do not input signals or an I/O pull-up power supply while the device is powered off. The current injection that results from input of such a signal or I/O pull-up power supply may cause malfunction and the abnormal current that passes in the device at this time may cause degradation of internal elements. Follow the guideline for input signal during power-off state as described in your product documentation.
4. 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 the 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.
5. Clock signals

   After applying a reset, only release the reset line after the operating clock signal becomes stable. When switching the clock signal during program execution, wait until the target clock signal is 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. Additionally, 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.
6. Voltage application waveform at input pin

   Waveform distortion due to input noise or a reflected wave may cause malfunction. If the input of the CMOS device stays in the area between V_{IL} (Max.) and V_{IH} (Min.) due to noise, for example, the device may malfunction. Take care to prevent chattering noise from entering the device when the input level is fixed, and also in the transition period when the input level passes through the area between V_{IL} (Max.) and V_{IH} (Min.).
7. Prohibition of access to reserved addresses

   Access to reserved addresses is prohibited. The reserved addresses are provided for possible future expansion of functions. Do not access these addresses as the correct operation of the LSI is not guaranteed.
8. Differences between products

   Before changing from one product to another, for example to a product with a different part number, confirm that the change will not lead to problems. The characteristics of a microprocessing unit or microcontroller unit products in the same group but having a different part number might differ in terms of 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 or any other use of the circuits, software, and information in the design of your product or system. Renesas Electronics disclaims any and all liability for any losses and damages incurred by you or third parties arising from the use of these circuits, software, or information.
2. Renesas Electronics hereby expressly disclaims any warranties against and liability for infringement or any other claims involving 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, including but not limited to, the product data, drawings, charts, programs, algorithms, and application examples.
3. 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 shall not alter, modify, copy, or reverse engineer any Renesas Electronics product, whether in whole or in part. Renesas Electronics disclaims any and all liability for any losses or damages incurred by you or third parties arising from such alteration, modification, copying or reverse engineering.
5. Renesas Electronics products are classified according to the following two quality grades: “Standard” and “High Quality”. The intended 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; industrial robots; etc.
   "High Quality": Transportation equipment (automobiles, trains, ships, etc.); traffic control (traffic lights); large-scale communication equipment; key financial terminal systems; safety control equipment; etc.

   Unless expressly designated as a high reliability product or a product for harsh environments in a Renesas Electronics data sheet or other Renesas Electronics document, Renesas Electronics products are not intended or 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 damage (space system; undersea repeaters; nuclear power control systems; aircraft control systems; key plant systems; military equipment; etc.). Renesas Electronics disclaims any and all liability for any damages or losses incurred by you or any third parties arising from the use of any Renesas Electronics product that is inconsistent with any Renesas Electronics data sheet, user’s manual or other Renesas Electronics document.
6. When using Renesas Electronics products, refer to the latest product information (data sheets, user’s manuals, application notes, “General Notes for Handling and Using Semiconductor Devices” in the reliability handbook, etc.), and ensure that usage conditions are within the ranges specified by Renesas Electronics with respect to maximum ratings, operating power supply voltage range, heat dissipation characteristics, installation, etc. Renesas Electronics disclaims any and all liability for any malfunctions, failure or accident arising out of the use of Renesas Electronics products outside of such specified ranges.
7. Although Renesas Electronics endeavors to improve the quality and reliability of Renesas Electronics products, semiconductor products have specific characteristics, such as the occurrence of failure at a certain rate and malfunctions under certain use conditions. Unless designated as a high reliability product or a product for harsh environments in a Renesas Electronics data sheet or other Renesas Electronics document, Renesas Electronics products are not subject to radiation resistance design. You are responsible for implementing safety measures to guard against the possibility of bodily injury, injury or damage caused by fire, and/or danger to the public in the event of a failure or malfunction of Renesas Electronics products, 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 and impractical, you are responsible for evaluating 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. You are responsible for carefully and sufficiently investigating applicable laws and regulations that regulate the inclusion or use of controlled substances, including without limitation, the EU RoHS Directive, and using Renesas Electronics products in compliance with all these applicable laws and regulations. Renesas Electronics disclaims any and all liability for damages or losses occurring as a result of your noncompliance with applicable laws and regulations.
9. Renesas Electronics products and technologies shall 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 shall comply with any applicable export control laws and regulations promulgated and administered by the governments of any countries asserting jurisdiction over the parties or transactions.
10. It is the responsibility of the buyer or distributor of Renesas Electronics products, or any other party who distributes, disposes of, or otherwise sells or transfers the product to a third party, to notify such third party in advance of the contents and conditions set forth in this document.
11. This document shall not be reprinted, 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.

(Note1) “Renesas Electronics” as used in this document means Renesas Electronics Corporation and also includes its directly or indirectly controlled subsidiaries.
(Note2) “Renesas Electronics product(s)” means any product developed or manufactured by or for Renesas Electronics.

(Rev.4.0-1 November 2017)

Corporate Headquarters

TOYOSU FORESIA, 3-2-24 Toyosu,
Koto-ku, Tokyo 135-0061, Japan
www.renesas.com

Trademarks

Renesas and the Renesas logo are trademarks of Renesas Electronics Corporation. All trademarks and registered trademarks are the property of their respective owners.

Contact information

For further information on a product, technology, the most up-to-date version of a document, or your nearest sales office, please visit:
www.renesas.com/contact/.