

RZ/T2M, RZ/N2L, RZ/T2N Group

TSN Sample application Package

Introduction

This document describes the system overview, supported features, and basic usage for evaluating TSN (Time-Sensitive Networking) using sample applications that implement TSN on the RZ/T2M, RZ/N2L and RZ/T2N.

Target Device

RZ/T2M, RZ/N2L, RZ/T2N

Contents

1. Overview	3
1.1 Abbreviations / Definitions	4
1.2 Reference	5
1.3 Limitation / Known Issue	5
2. Features	6
2.1 IEEE802.1AS (gPTP)	6
2.2 IEEE802.1Qbv (TDMA)	7
2.3 IEEE802.1Qbu (Frame Preemption)	7
2.4 IEEE802.1Qci (PSFP)	7
2.5 IEEE802.1Qch (CQF)	7
2.6 IEEE802.1AB (LLDP)	7
3. Project Setup	8
3.1 Environment during development	8
3.2 Hardware requirements	8
3.2.1 Hardware-setting of RZ/T2M or RZ/N2L RSK	9
3.2.2 Connect PC to RZ/T2M-RSK or RZ/N2L-RSK	23
3.3 Software requirements	28
3.3.1 Required software for download sample application	28
3.3.2 Third-Party software	32
4. Running sample application	33
4.1 Build and download	33
4.1.1 Open project	33
4.1.2 Pre-configuration for each device	37
4.1.3 Build and download to RZ/T2M-RSK or RZ/N2L-RSK	47
4.2 Run sample application	48
4.2.1 If using "EWARM"	51
4.2.2 If using "e² studio"	51

4.3	Example of Operation Check for Sample Application	51
4.4	Example of Operation Check for Sample Application (three-device setup)	53
5.	gPTP+TDMA sample application	54
5.1	Sample application Overview	54
5.1.1	Sample menu	56
5.1.2	Operation Specifications of gPTP	59
5.1.3	Operation Specifications of TDMA	60
5.1.4	Operation Specifications of RT frame and NRT frames	61
5.1.5	Operation Specifications of Frame Preemption	61
5.1.6	Operation Specifications of PSFP	64
5.1.7	Operation Specifications of CQF	66
5.1.8	Operation Specifications of LLDP	69
5.2	Sample application's outputs for checking operation	70
5.2.1	Outputs of log message	70
5.2.2	Outputs of GPIO Pin	77
5.3	Application structure	90
5.3.1	Module structure	90
5.3.2	Note - Task priorities in sample application	94
	Revision History	95

1. Overview

This document provides an overview of a TSN sample application for RZ/T2M, RZ/N2L and RZ/T2N.

TSN (Time-Sensitive-Networking) is a Networking system that can operate real-time traffic and another best-effort traffic simultaneously in same network. By utilizing mechanisms such as time synchronization, TSN enables deterministic communication behavior over standard Ethernet-based networks.

This sample application demonstrates how TSN-related functions can be used on the target devices and is intended to help users understand the basic configuration, operation, and evaluation procedures of TSN communication using Ethernet switch functionality and related software components.

The sample application includes functions such as time synchronization, scheduled traffic transmission, frame preemption, traffic control, cyclic queuing, and device information notification. These functions are provided through a combination of hardware features, such as the Ethernet switch module, and software components including the IEEE 802.1AS software stack and LLDP-related processing, allowing users to evaluate how they contribute to deterministic communication.

If you want to execute this sample application on target board, set up the target board and the sample application regarding this document.

1.1 Abbreviations / Definitions

Table 1-1 Abbreviations/Definitions

Index	Abbreviations /Definitions	Description
1	IP	Internet Protocol
2	TCP	Transmission Control Protocol
3	USB	Universal Serial Bus
4	PC	Personal Computer
5	SW	Switch
6	GPIO	General Purpose Input/Output
7	EWARM	Embedded Workbench® for ARM
8	lwIP	lightweight IP
9	xSPI	Expanded Serial Peripheral Interface
10	QSPI	Quad Serial Peripheral Interface
11	flash	Flash devices that use the xSPI
12	ETHSW	Ethernet Switch
13	GMAC	Ethernet MAC
14	TSN	Time-Sensitive Networking
15	IEEE	Institute of Electrical and Electronics Engineers
16	PTP / gPTP	Precision Time Protocol / Generalized Precision Time Protocol
17	TDMA	Time Division Multiple Access (IEEE802.1Qbv)
18	IDE	Integrated Development Environment
19	FSP	Flexible Software Package
20	NOR	Not OR
21	RAM	Random Access Memory
22	UART	Universal Asynchronous Receiver Transmitter
23	MAC	Media Access Control
24	RT	Real Time
25	NRT	Non-Real Time
26	UDP	User Datagram Protocol
27	CN	Connector
28	CH	Channel
29	JN	Jumper N
30	PSFP	Per-Stream Filtering and Policing (IEEE802.1Qci)
31	CQF	Cyclic Queuing and Forwarding (IEEE802.1Qch)
32	LLDP	Link Layer Discovery Protocol (IEEE 802.1AB)
33	MIB	Management Information Base
34	SNMP	Simple Network Management Protocol

1.2 Reference

Technical information about each board is available via Renesas website.

Table 1-2 Technical Inputs for RZ/T2M

Index	Technical Inputs
1	RZ/T2M, RZ/T2ME Group Renesas Starter Kit+ for RZ/T2M, RZ/T2ME User's Manual r20ut4939egxxxx-rskplus-rzt2m-v1-um.pdf
2	RZ/N2M Group User's Manual: Hardware r01uh0916ejxxxx-rzt2m.pdf
3	RZ/T2, RZ/N2 Getting Started with Flexible Software Package r01an6434ejxxxx-rzt2-rzn2-fsp-getting-started.pdf

Table 1-3 Technical Inputs for RZ/N2L

Index	Technical Inputs
1	RZ/N2L Group Renesas Starter Kit+ for RZ/N2L User's Manual r20ut4984egxxxx-rskplus-rzn2l-v1-um.pdf
2	RZ/N2L Group User's Manual: Hardware r01uh0955ejxxxx-rzn2l.pdf
3	RZ/T2, RZ/N2 Getting Started with Flexible Software Package r01an6434ejxxxx-rzt2-rzn2-fsp-getting-started.pdf

Table 1-4 Technical Inputs for RZ/T2N

Index	Technical Inputs
1	
2	RZ/T2N Group User's Manual: Hardware r01uh1084ejxxxx-rzt2n.pdf
3	RZ/T, RZ/N Series Getting Started with RZ Flexible Software Package r01an6434ej0420-rzt-rzn-series-fsp-getting-started.pdf

1.3 Limitation / Known Issue

- When using e²studio, if you repeatedly plug and unplug the Ethernet cable of a device that is operating as a timeReceiver, the device may hang up immediately after outputting the "Sync-TimeOut" log message.
- RZ/T2M-RSK does not support Qci features.
Therefore, samples related to Qci and tests using them cannot be implemented.

2. Features

TSN for this application includes the following components:

- IEEE802.1AS (gPTP)
 - It synchronizes time among devices on the network. In this sample application, it is used as the foundation for time-aware TSN operation.
- IEEE802.1Qbv (TDMA)
 - It controls the transmission timing of Ethernet frames according to a defined schedule. This function enables scheduled traffic transmission within specified time slots.
- IEEE802.1Qbu (Frame Preemption)
 - High-priority frames are allowed to preempt the transmission of low-priority frames. This feature reduces the latency of time-sensitive traffic.
- IEEE802.1Qci (PSFP)
 - It performs traffic filtering on a per-stream basis. This function controls abnormal or unexpected traffic and enhances the reliability of TSN communication.
- IEEE802.1Qch (CQF)
 - It forwards frames based on cyclic queue operations. This function enables deterministic network transmission.

In addition, this application includes the following components:

- IEEE802.1AB (LLDP)
 - It provides notification and retrieval of information about connected devices and network topology.

2.1 IEEE802.1AS (gPTP)

gPTP is one profile of IEEE1588 (PTP), a protocol for time synchronization.

Time synchronization in gPTP is realized as follows:

- At start of gPTP, each device is assigned as "timeTransmitter"(the GrandMaster device) or "timeReceiver" (devices other than the GrandMaster).
- "timeTransmitter" device act as source of time information.
- "timeReceiver" device synchronizes its own time based on received timeTransmitter's time information.

gPTP has the following features:

- High accurate time synchronization
 - Time synchronization with microsecond-level precision or better.
- measurement of transmission delay
 - gPTP calculates delay of transmission, so precision time synchronization is possible even in situations where transmission speeds vary be traffic routes.
- Automatic timeTransmitter device selection
 - Search automatically for the device has highest accuracy clock in network and select it as timeTransmitter.

In this sample application, it is implemented using the IEEE 802.1AS Software Stack, a Renesas-provided gPTP library compliant with IEEE 802.1AS-2020.

2.2 IEEE802.1Qbv (TDMA)

TDMA(Time Division Multiple Access) is the traffic control system that divides transmission timing of communication traffic by time.

TDMA is realized as follows:

- Divide the TDMA-cycle based on the user-specified time period.
- Assign some traffic to each period and ensure that each traffic is sent only during the assigned period.

Thereby, each traffic will be sent at a specific interval.

By using TDMA, multiple traffic can be operated multilaterally with guaranteed maximum delay times.

That makes it possible to realize TSN.

Note: TDMA is required to be guaranteed time-synchronization by gPTP (or other time synchronization protocol).

2.3 IEEE802.1Qbu (Frame Preemption)

Frame Preemption realizes "Traffic Interspersing".

When transmitting a high-priority frame while a low-priority frame is being transmitted, it sends the high-priority frame by interrupting the low-priority frame.

2.4 IEEE802.1Qci (PSFP)

PSFP discards received traffic according to certain criteria.

By enabling it on the timeTransmitter device, it is possible to filter traffic transmitted from the timeReceiver.

Note: RZ/T2M-RSK does not support PSFP.

2.5 IEEE802.1Qch (CQF)

CQF is intended to allow delivery of time-sensitive traffic in a deterministic latency that is dependent only on the number of hops through the network. To remove the hop-to-hop latency and switching delay, each intermediate device is required to buffer the frames received in a time period, while transmitting frames that were received the period before.

This results in a hop-to-hop latency equal to one cycle if the network and switch delay allows for the frames to be fully transmitted and received within the time period.

2.6 IEEE802.1AB (LLDP)

LLDP notifies its own information on the network.

3. Project Setup

3.1 Environment during development

Development environment is as follows:

Table 3-1 Environment during development

Item	Description		
Board	RZ/T2M	Renesas Started Kit+ for RZ/T2M (*1) (R9A07G075M24GBG)	
	RZ/N2L	Renesas Started Kit+ for RZ/N2L (*3) (R9A07G084M04GBG)	
	RZ/T2N	RZ/T2N Evaluation Board (*2) (R9A07G076M48GBG)	
IDE	IAR Systems Embedded Workbench for ARM (*4) (Version: 9.60.3)		
	Renesas Electronics e² studio (Version: 2026-07)	ToolChain	GNU ARM Embedded (Version: 13.3.1.arm-13-24)
FSP	RZ Flexible Software Package (FSP) (v4.2.0)		
	FSP Smart Configurator (Version: 2026-07)		

Note *1: referred to as RZ/T2M-RSK

Note *2: referred to as RZ/N2L-RSK

Note *3: referred to as RZ/N2L-RSK

Note *4: referred to as EWARM

3.2 Hardware requirements

The following items are required for running sample application:

Table 3-2 List of necessary items

Item	Description	Note
Any of the following: ● RZ/T2M-RSK ● RZ/N2L-RSK ● RZ/T2N-Evb	Evaluation board	(*1)
I-jet	Debugger	(*1)(*2)
USB cable (USB A to Micro-B)	for connection with debugger	(*1)
USB cable (USB A to Mini-B)	for serial connection + for communication to select functions	(*1)
Ethernet cables	for connection between devices	
(option) oscilloscope has 4 or more channel	for checking signal waveforms	

Note *1: Prepare two identical devices in total: one for timeTransmitter and one for timeReceiver. (*3)

Note *2: Required only if build by "EWARM".

Note *3: Prepare three identical devices in total when operating with three-unit setup like the "4.4 Example of Operation Check for Sample Application (three-device setup)".

3.2.1 Hardware-setting of the board

Below are the hardware configuration items for each board, respectively:

Table 3-3 Hardware-setting for RZ/T2M-RSK

Reference	Setting	Description
CN8	Shorted Pin 2-3	Enable QSPI (IC21).
CN17	Shorted Pin 2-3	Connect 1.8V Power rail to VCC1833_2. (When using Ethernet)
J9	(Open/Short)	External-emulator enable/disable selection (See 3.2.1.2)
SW4-1	(ON/OFF)	Boot mode selection (See 3.2.1.1)
SW4-2		
SW4-3		
SW4-4	ON	JTAG Authentication by Hash is disabled.
SW4-5	OFF	ATCM 1 wait
SW5-3	ON	Enable SCI_RTS
SW5-4	OFF	
SW5-5	ON	Enable SCI_RXD
SW5-6	OFF	
SW5-7	OFF	
SW5-8	OFF	Enable SCK3
SW5-9	ON	
SW5-10	OFF	
SW6-1	OFF	Enables signals other than the external bus. (CAN, Emulator, I2C, etc.)
SW6-3	ON	Enable TRACE_CTL
SW6-4	OFF	
SW6-5	OFF	Enable SCI_TXD
SW6-6	ON	
SW6-7	OFF	Enable MB_RST
SW6-8	ON	
SW6-9	OFF	Enable CAN_RX_OB
SW6-10	ON	

Table 3-4 Hardware-setting for RZ/N2L-RSK

Reference	Setting	Description
CN8	Shorted Pin 2-3	Enable QSPI (IC21).
CN17	Shorted Pin 2-3	Connect 1.8V Power rail to VCC1833_2. (When using Ethernet)
J9	(Open/Short)	External-emulator enable/disable selection (See 3.2.1.2)
SW4-1	(ON/OFF)	Boot mode selection (See 3.2.1.1)
SW4-2		
SW4-3		
SW4-4	ON	JTAG Authentication by Hash is disabled.
SW4-6	OFF	Enables signals other than the trace signal. (Motor, RS485, etc.)
SW4-7	ON	Enables signals other than the external bus. (CAN, Emulator, I2C, etc.)
SW4-8	OFF	Enable SW3.
SW8-1	OFF	Enable the "LED_GREEN" signal.
SW8-2	ON	
SW8-3	OFF	
SW8-4	ON	Enable the "LED5" signal.
SW8-5	OFF	
SW11-1	ON	Enable the "LED_RED2" signal.
SW11-2	OFF	
SW11-3	OFF	

Table 3-5 Hardware-setting for RZ/T2N-Evb

Reference	Setting	Description
SW8-1	ON	Boot mode setting: fixed to "xSPI0 boot mode (x1 boot serial flash)".
SW8-2	ON	
SW8-3	ON	
SW8-4	OFF	(Unused, factory default = OFF)
SW10-1	OFF	(Unused, factory default = OFF)
SW10-2	ON	VCC1833_1 = 1.8V (J15 : 1-2 short)
SW10-3	ON	VCC1833_2 = 1.8V (J18 : 1-2 short)
SW10-4	OFF	(Unused, factory default = OFF)
SW10-5	OFF	(Unused, factory default = OFF)
SW10-6	ON	VCC1833_5 = 1.8V (J39 : 1-2 short)
SW10-7	OFF	Cortex-R52 CPU0 ATCM wait cycle = 1 wait
SW10-8	OFF	Cortex-R52 CPU1 ATCM wait cycle = 1 wait
SW10-9	ON	JTAG mode = Normal mode
SW10-10	ON	2nd Boot CPU = Cortex-R52 CPU0
SW7-1	ON	P03_6,7, P04_0-3 and P15_0 are used as microSD card control signals.
SW7-2	ON	P23_1,2, 4-7, P24_0-4 and P25_0 are used as Ethernet Port2 control signals.
SW7-3	ON	P00_0-7, P01_0-5, P16_1 and P16_7 are used as Ethernet Port3 control signals.
SW7-4	ON	P01_6,7, P02_0-7, P03_0,1,4,5 and P15_1 are used as Ethernet Port4 control signals.
SW7-5	ON	P15_7 and P16_2-5 are used as QSPI Flash and extension SPI control signals.
SW7-6	ON	P17_5 and P18_3-6 are used as OctaFlash and HyperRAM control signals.
SW7-7	OFF	P04_4-7 and P05_0-6 are used as HSPI, inverter board INV3 and PMOD2 control signals.
SW7-8	OFF	signals.
J29	(Open/Short)	External-emulator enable/disable selection (See 3.2.1.2)

Note: For any items not specified, use the factory default settings.
Please refer to the "RZ/T2N Evaluation Board User's Manual" for details.

Depending on the execution environment, some hardware settings may need to be changed.

The following hardware settings need to be changed:

- Boot mode settings
- Enabling/disabling the external-emulator

3.2.1.1 Change boot mode

The boot mode of this sample application is selectable from RAM execution and xSPI boot.

Note: For the RZ/T2N-Evb, no hardware changes are required because it is fixed to "xSPI0 boot mode (x1 boot serial flash)".
However, "erasing the flash memory when switching operation modes" is mandatory.
For details, refer to "3.2.1.1.1 Erasing the Flash Memory."

- RAM execution

- To run this sample application by loading it into RAM from the debugger, set the boot mode to "16-bit bus boot mode (NOR flash)" or "xSPI0 boot mode (x1 boot serial flash)".

Note: When executing in RAM, it is necessary to delete the memory in advance.
For details, see "3.2.1.1.1 How to Erase Flash Memory".

- xSPI boot

- To run this sample application by programming and executing it from flash, set the boot mode to "xSPI0 boot mode (x1 boot serial flash)".

Below are the settings required to boot in each of these boot modes:

Table 3-6 Hardware-setting "Change boot mode"

Reference	16-bit bus boot mode (NOR flash)	xSPI boot (x1 boot serial flash)
SW4-1	ON	ON
SW4-2	OFF	ON
SW4-3	ON	ON

Note: Common to RZ/T2M-RSK and RZ/N2L-RSK.

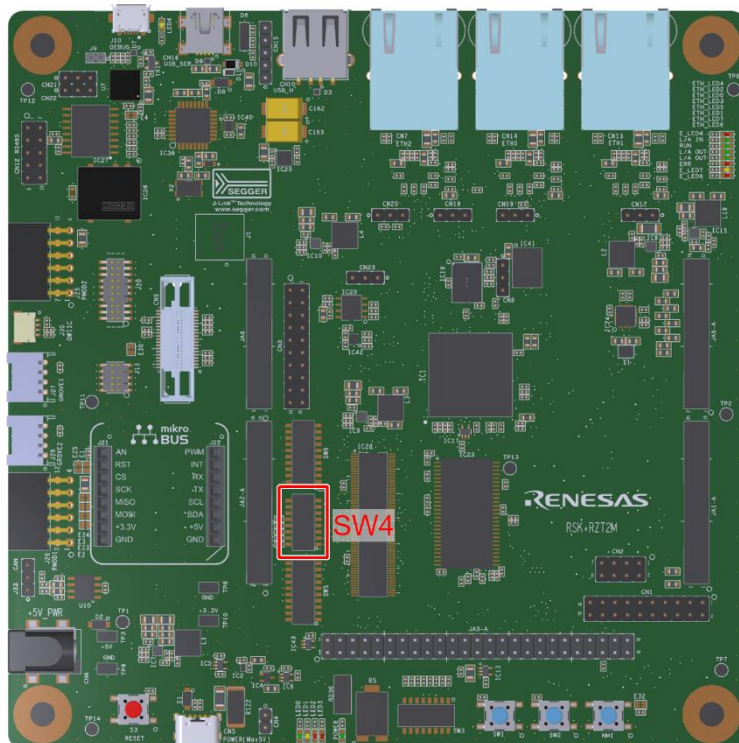


Figure 3-1 RZ/T2M-RSK board layout (SW4)

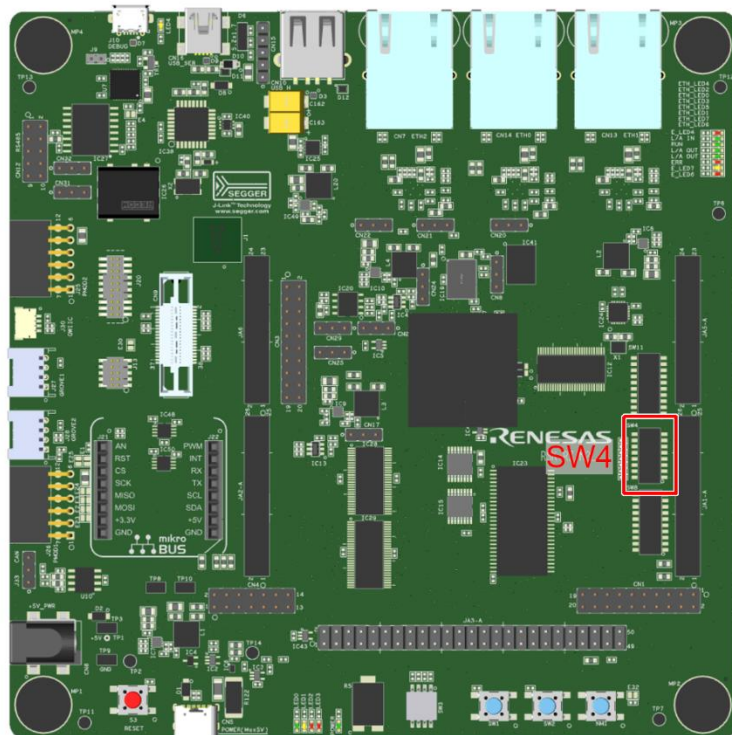


Figure 3-2 RZ/N2L-RSK board layout (SW4)

3.2.1.1.1 How to Erase Flash Memory

Note: This procedure, when switching from A to B, needs to be executed only once.
(After switching, it does not need to be executed repeatedly.)

See: "RZ/T2, RZ/N2 Getting Started with Flexible Software Package" : "9.1 How to Erase Flash Memory"

3.2.1.1.1.1 If using "EWARM"

Note: If the asymmetric multicore setting is enabled, the erase function cannot be used: it must be disabled.

Disable asymmetric multicore setting:

1. In toolbar, click Project > "Options..."
2. In "Debugger" > "Multicore", select "Disable" in Asymmetric multicore.

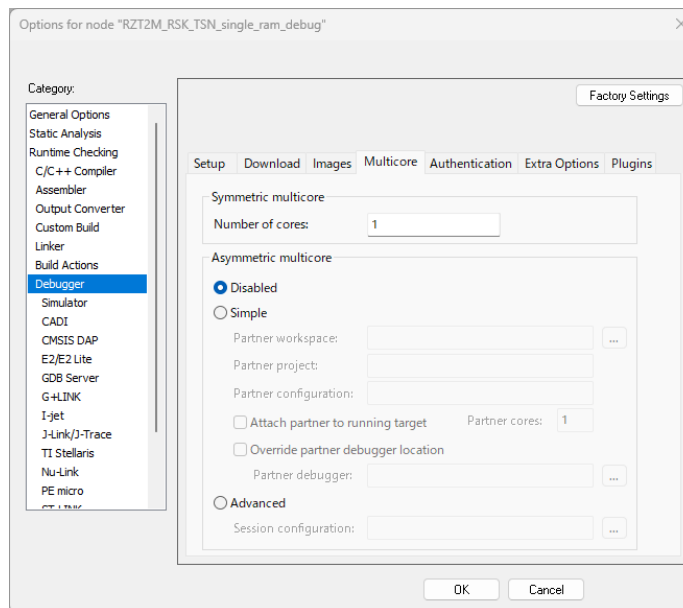


Figure 3-3 EWARM - Disabling asymmetric multicore

Erase flash memory by following the steps:

1. Open the workspace of a project.
(See: 4.1.1.1)
2. In toolbar, select "Project" > "Download" > "Erase memory".

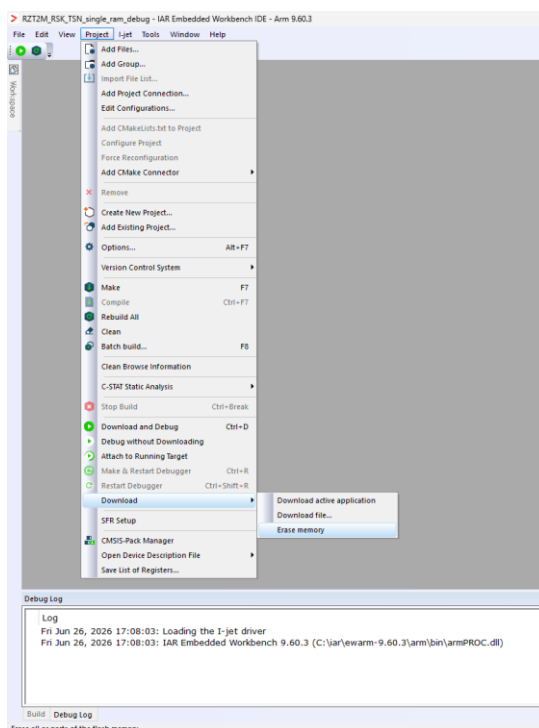


Figure 3-4 EWARM - Select "Erase memory"

3. Select erase memory space according to the current boot mode(See: 3.2.1.1) and start erasing.

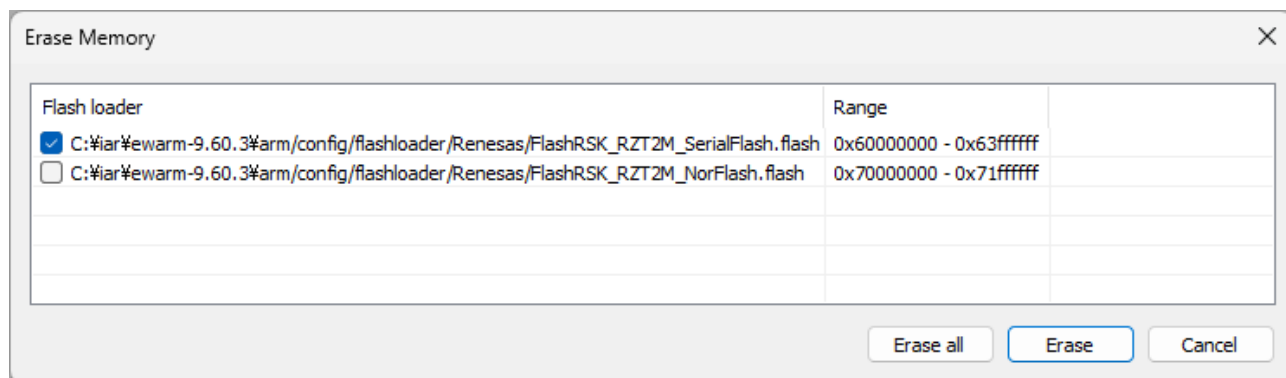


Figure 3-5 EWARM - Erase Memory

- The correspondence between boot mode and memory area is as follows:

Table 3-7 Correspondence between boot mode and memory area (When using the RZ/T2M or RZ/N2L)

Boot mode	Target memory area		
	Address space	Start address	End address
xSPI0 x1	xSPI0 CS0	0x60000000	0x63FFFFFF
16-bit bus	CS0	0x70000000	0x71FFFFFF

Note: Common to RZ/T2M-RSK and RZ/N2L-RSK.

Table 3-8 Correspondence between boot mode and memory area (When using the RZ/T2N)

Boot mode	Target memory area		
	Address space	Start address	End address
xSPI0 x1	xSPI0 CS0	0x40000000	0x43FFFFFF

4. After the following dialog appears, erasing of the flash is complete if no error occurs.

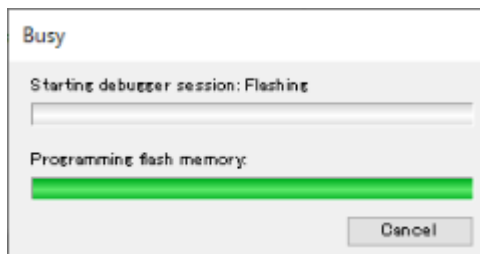


Figure 3-6 EWARM - Screen During Erasing

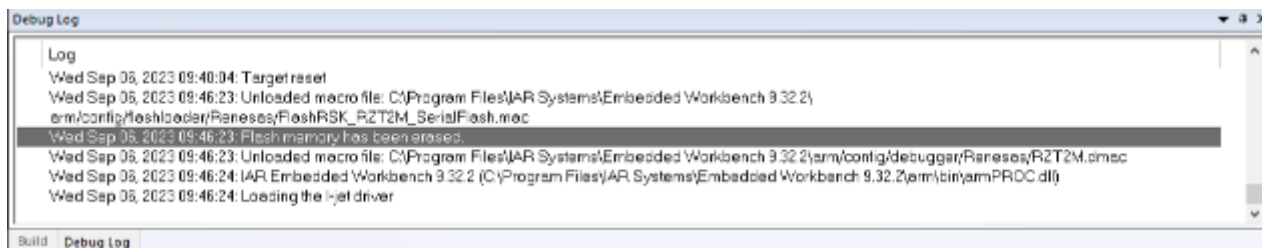
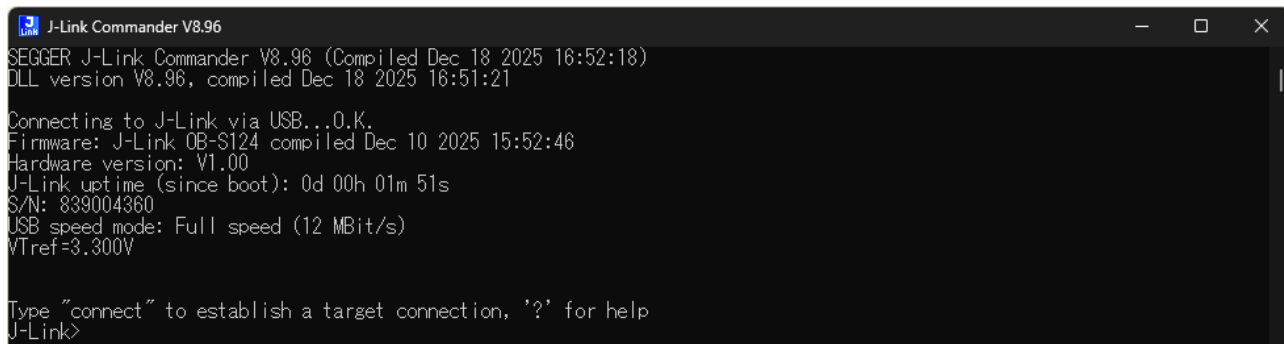


Figure 3-7 EWARM - Message of Flash Memory Erase Complete

3.2.1.1.1.2 If using "e² studio"

Erase flash memory by following the steps:

1. Connect board with PC(see: 3.2.2) and open the J-Link Commander.



```

J-Link Commander V8.96
SEGGER J-Link Commander V8.96 (Compiled Dec 18 2025 16:52:18)
DLL version V8.96, compiled Dec 18 2025 16:51:21

Connecting to J-Link via USB...O.K.
Firmware: J-Link OB-S124 compiled Dec 10 2025 15:52:46
Hardware version: V1.00
J-Link uptime (since boot): 0d 00h 01m 51s
S/N: 839004360
USB speed mode: Full speed (12 MBit/s)
Vtref=3.300V

Type "connect" to establish a target connection, '?' for help
J-Link>
  
```

Figure 3-8 J-Link Commander - Startup screen

2. Type `connect` in J-Link Commander.
3. At `Device >`, enter the device type in accordance with following:

Table 3-9 Device types

Board	Device type name
RZ/T2M-RSK	R9A07G075M24_CPU0
RZ/N2L-RSK	R9A07G084M04
RZ/T2N-Evb	R9A07G076M48_R52_0

4. At `TIF>`, enter `S`.
5. At `Speed>`, press the Enter key without entering anything.
(The default settings will be used.)

6. After that, confirm that the message “Cortex-R52 identified.” is displayed.

```

J-Link>connect
Please specify device / core. <Default>: R9A07G084M04
Type '?' for selection dialog
Device>R9A07G075M24_CPU0
Please specify target interface:
  J) JTAG (Default)
  S) SWD
  T) cJTAG
TIF>S
Specify target interface speed [kHz]. <Default>: 4000 kHz
Speed>
Device "R9A07G075M24_CPU0" selected.

Connecting to target via SWD
ConfigTargetSettings() start
ConfigTargetSettings() end - Took 178us
InitTarget() start
Authenticated device detected. Skipping authentication process.
  OCDREG_STATUS: 0x00000001
InitTarget() end - Took 205ms
Found SW-DP with ID 0x6BA02477
DPIDR: 0x6BA02477
CoreSight SoC-400 or earlier
AP map detection skipped. Manually configured AP map found.
AP[0]: APB-AP (IDR: Not set, ADDR: 0x00000000)
AP[1]: APB-AP (IDR: Not set, ADDR: 0x00000000)
AP[2]: AXI-AP (IDR: Not set, ADDR: 0x00000000)
Using preconfigured AP[1] as APB-AP
AP[1]: APB-AP found
DebugRegs + CTI manually specified. ROM table scan skipped.
Cortex-R52 @ 0x80410000 (configured)
CoreCTI @ 0x80420000 (configured)
Debug architecture: ARMv8.0
8 code breakpoints, 8 data breakpoints
Processor features:
  EL0 support: AArch32
  EL1 support: AArch32
  EL2 support: AArch32
  EL3 support: N/A
  FPU support: Single + Double + Conversion
Add. info (CPU temp. halted)
Current exception level: EL1
Exception level AArch usage:
  EL0: AArch32
  EL1: AArch32
  EL2: AArch32
  EL3: AArch32
Non-secure status: Non-secure
Cache info:
  Inner cache boundary: none
  LoU Uniprocessor: 1
  LoC: 1
  LoU Inner Shareable: 1
I-Cache L1: 16 KB, 64 Sets, 64 Bytes/Line, 4-Way
D-Cache L1: 16 KB, 64 Sets, 64 Bytes/Line, 4-Way
Memory zones:
  Zone: "Default" Description: Default access mode
  Zone: "AP0" Description: MEM-AP (APB-AP)
  Zone: "AP1" Description: MEM-AP (APB-AP)
  Zone: "AP2" Description: MEM-AP (AXI-AP)
Cortex-R52 identified.
J-Link>_

```

Figure 3-9 J-Link Commander - Connection successful

7. Type `exec EnableEraseAllFlashBanks`.
8. Use the `erase` commands with specifying erase memory space according to the current boot mode(See: 3.2.1.1).
 - The correspondence between boot mode and memory area is as follows:

Table 3-7 Correspondence between boot mode and memory area (When using the RZ/T2M or RZ/N2L) (repost)

Boot mode	Target memory area		
	Address space	Start address	End address
xSPI0 x1	xSPI0 CS0	0x60000000	0x63FFFFFF
16-bit bus	CS0	0x70000000	0x71FFFFFF

Note: Common to RZ/T2M-RSK and RZ/N2L-RSK.

Table 3-8 Correspondence between boot mode and memory area (When using the RZ/T2N) (repost)

Boot mode	Target memory area		
	Address space	Start address	End address
xSPI0 x1	xSPI0 CS0	0x40000000	0x43FFFFFF

9. After that, confirm that the message “Erasing done.” is displayed.

```

Cortex-R52 identified.
J-Link>exec EnableEraseAllFlashBanks
J-Link>erase 0x60000000 0x63FFFFFF
'erase': Performing implicit reset & halt of MCU.
ResetTarget() start
Authenticated device detected. Skipping authentication process.
OCDREG_STATUS: 0x00000001
Disabled core power domain detected.
Enabling debug mode...
ResetTarget() end - Took 537ms
Device specific reset executed.
Erasing selected range...
J-Link: Flash download: Total time needed: 238.444s (Prepare: 1.718s, Compare: 0.000s, Erase: 236.464s, Program: 0.000s, Veri
fy: 0.000s, Restore: 0.262s)
J-Link: Flash download:
Flash sectors within Range [0x60000000 - 0x63FFFFFF] deleted.
Erasing done.
J-Link>
  
```

Figure 3-10 J-Link Commander - Erasing done

10. Enter `q` to exit J-Link Commander.

3.2.1.2 Change external-emulator enable/disable

Setting of enable/disable external-emulator depends on the IDE used.

- If using "EWARM"
 - jet debugger (external-emulator) is used for debugging.
External-emulator must be set to "enable".
- If using "e² studio"
 - On-board debugger(J-LinkOB) is used for debugging.
External-emulator must be set to "disable".

External-emulator enable/disable requires the following operations:

Table 3-10 Hardware-setting "enable external-emulator"

Target Board	When using EWARM (with external emulator enabled)	When using e ² studio (with external emulator disabled)
RZ/T2M	J9:Short	J9:Open
RZ/N2L		
RZ/T2N	J29:Short	J29:Open

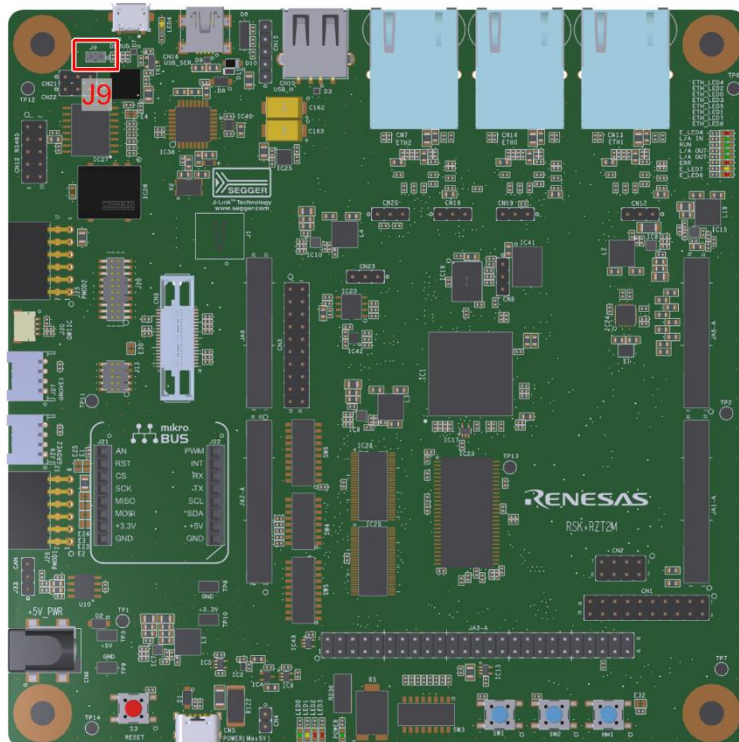


Figure 3-11 RZ/T2M-RSK board layout (J9)

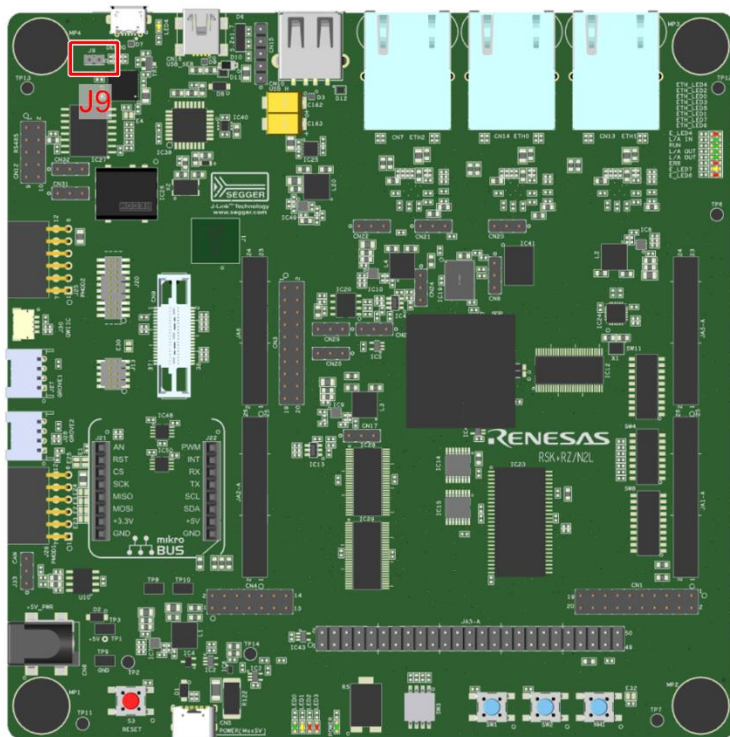


Figure 3-12 RZ/N2L-RSK board layout (J9)

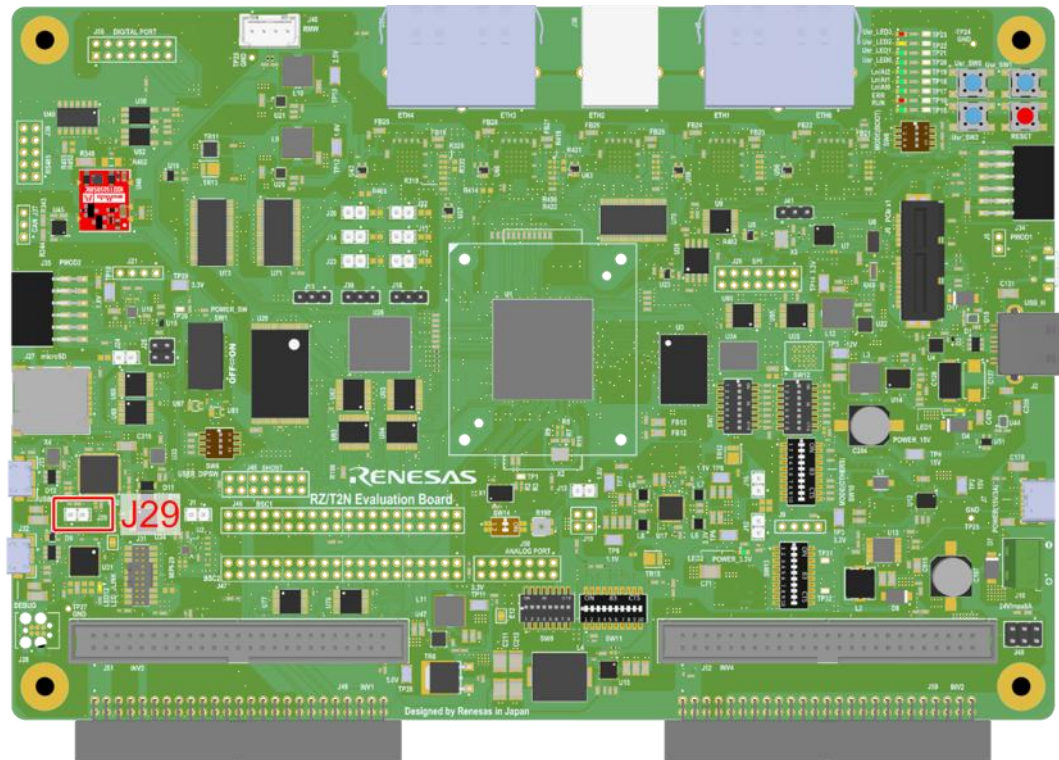


Figure 3-13 RZ/T2N-Evb board layout (J29)

3.2.2 Connect PC to the board

3.2.2.1 If using "EWARM"

Connection of Host PC to the board uses "I-jet debugger".

Connect the host PC to the I-jet debugger, and connect the I-jet debugger to the board.

The pin assignment for the connections and their locations on the board are shown below.

Table 3-11 Pin assignment for I-jet

Target Board	Assignment
RZ/T2M-RSK	J20
RZ/N2L-RSK	
RZ/T2N-Evb	J31

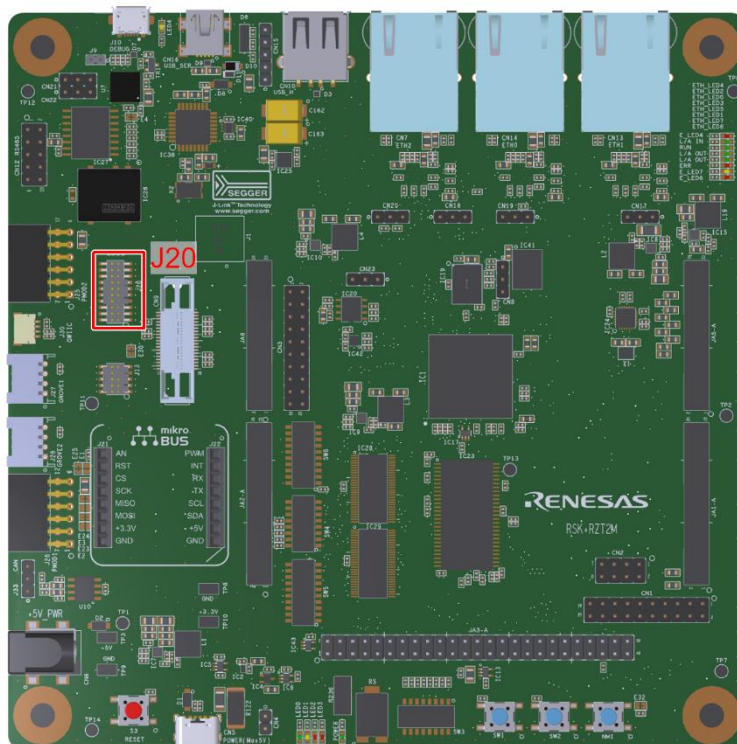


Figure 3-14 RZ/T2M-RSK board layout (J20)

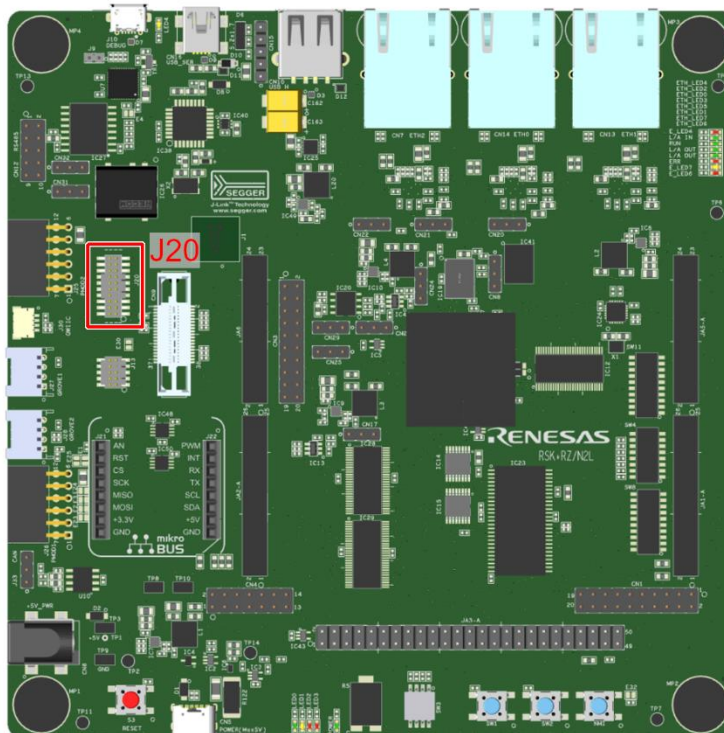


Figure 3-15 RZ/N2L-RSK board layout (J20)

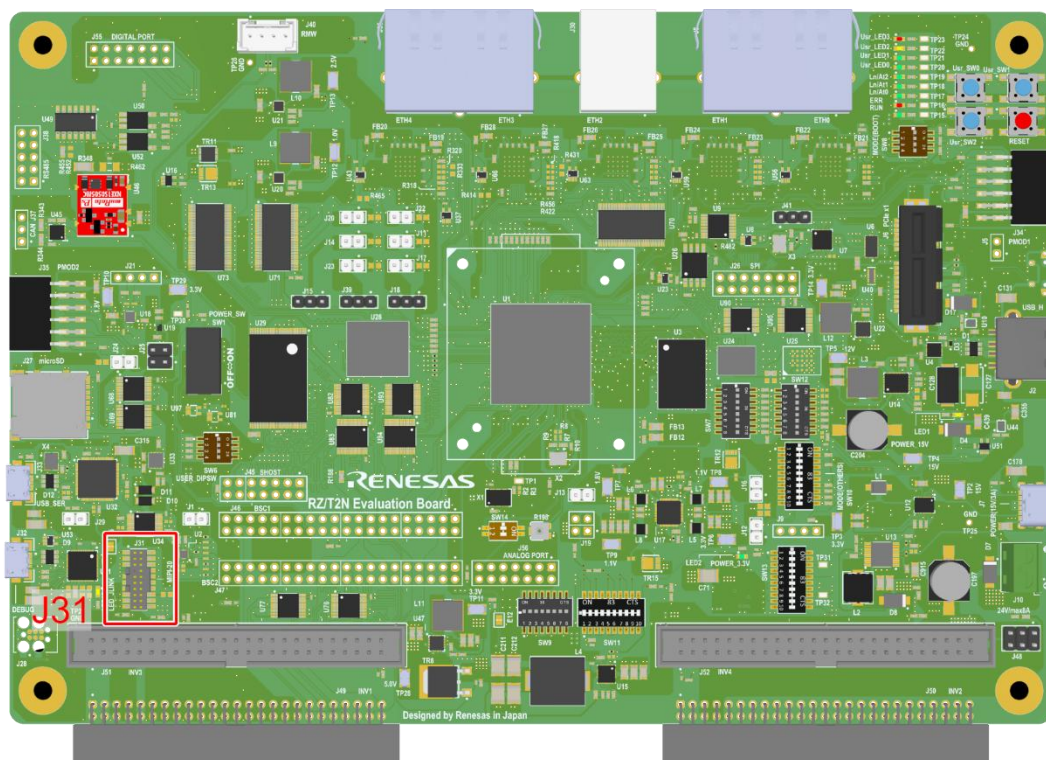


Figure 3-16 RZ/T2N-Evb board layout (J31)

When connection is successful, the lamps of "USB" and "DBG" on I-jet light up.

Note: The lamps of "TPWR" must be off.

3.2.2.2 If using "e² studio"

Connection of Host PC to the board uses "On-board debugger(J-LinkOB)".

Connect Host PC and J-Link OB USB Connector on the board.

The pin assignment for the connections and their locations on the board are shown below.

Table 3-12 Pin assignment for e² studio

Target Board	Assignment
RZ/T2M-RSK	J10
RZ/N2L-RSK	
RZ/T2N-Evb	J32

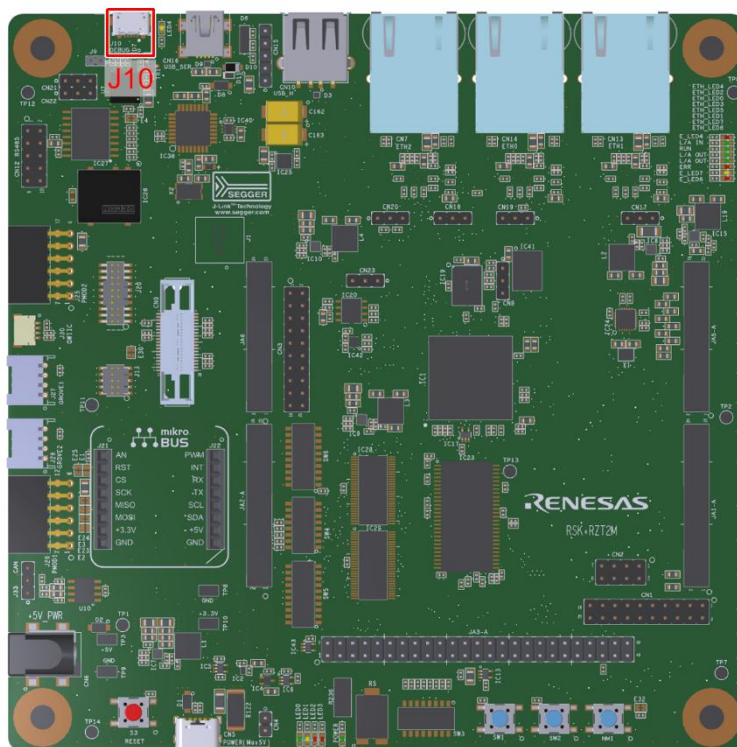


Figure 3-17 RZ/T2M-RSK board layout (J10)

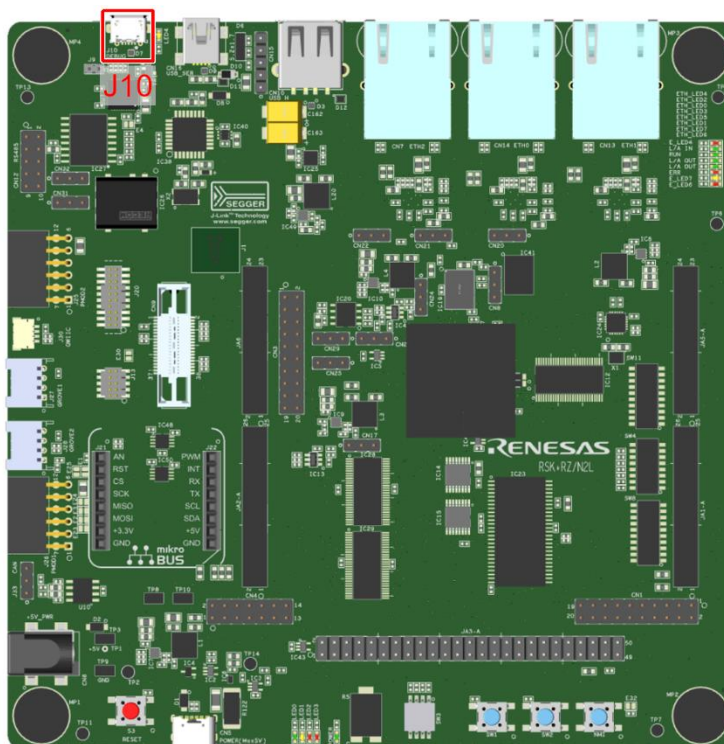


Figure 3-18 RZ/N2L-RSK board layout (J10)

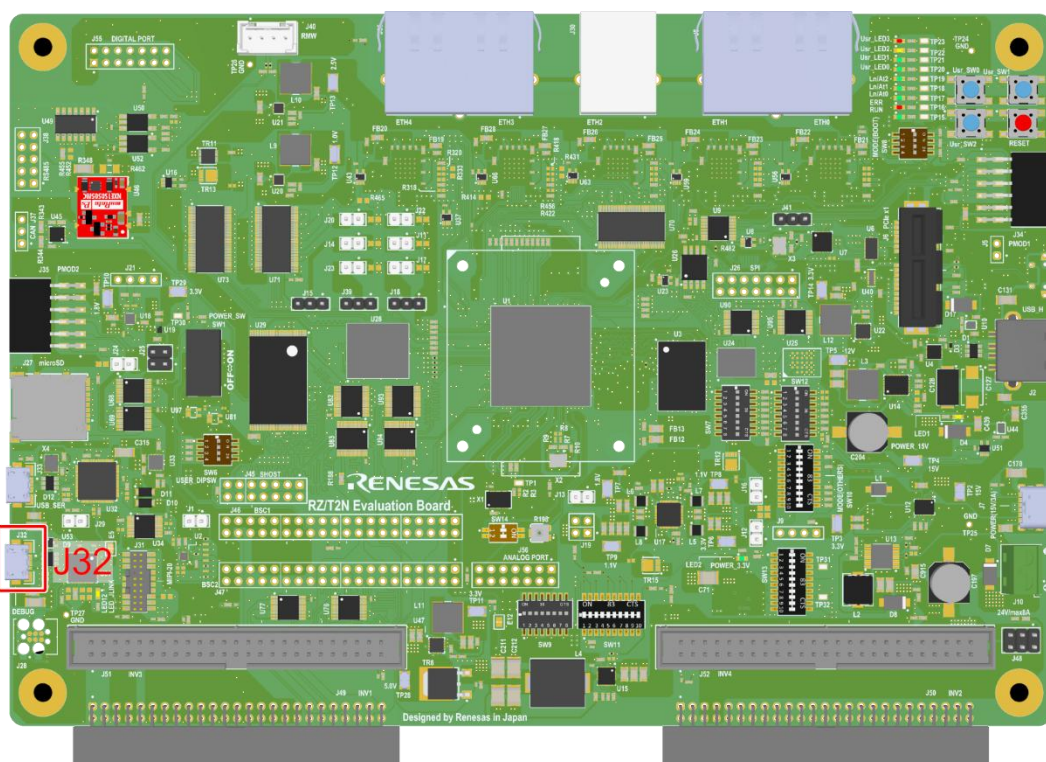


Figure 3-19 RZ/T2N-Evb board layout (J32)

Note: Whether the connection is correct can be confirmed by the following procedure:

1. On toolbar, click on the gear icon near "Launch Configuration".
2. In appeared dialog, select the "Debugger" tab.
3. In appeared screen, select the "Connection Setting" tab.

4. See the "J-Link" -> "J-Link Serial" field.
If the serial number of the connected board's J-Link can be selected, the connection is successful.

3.3 Software requirements

The following software is required for running sample application:

3.3.1 Required software for download sample application

The following software is required for building and downloading sample applications:

- IDE

EWARM or **e²studio** is selectable.

- A) **EWARM** (*1)(*2)

- Note *1: License activation is required to use this software.

- Note *2: The following software must be installed to use the board :

RZ FSP SC(Smart Configurator) and RZ FSP installer (for EWARM)

"setup_rzfsp_v4_2_0_rzsc_v2026-07.exe"

Note: This software is launched from above IDE.

For information on how to launch, see "3.3.1.1.1 Launch FSP Smart Configurator from IDE".

- B) **e² studio**

- e² studio and RZ/T2 FSP installer

- "setup_rzfsp_v4_2_0_e2s_v2026-07.exe"

Note: "RZ FSP SC(Smart Configurator) and RZ FSP installer (for EWARM)" and "e² studio and RZ/T2 FSP installer" are available on the following link:

<https://github.com/renesas/rz-fsp/releases>

Note: For detailed informations such as each software version, see "3.1 Environment during development".

3.3.1.1 Set-up of IDE

3.3.1.1.1 Launch FSP Smart Configurator from IDE

3.3.1.1.1.1 If using "EWARM"

1. On toolbar, select "Tools" -> "Configure Tools..."

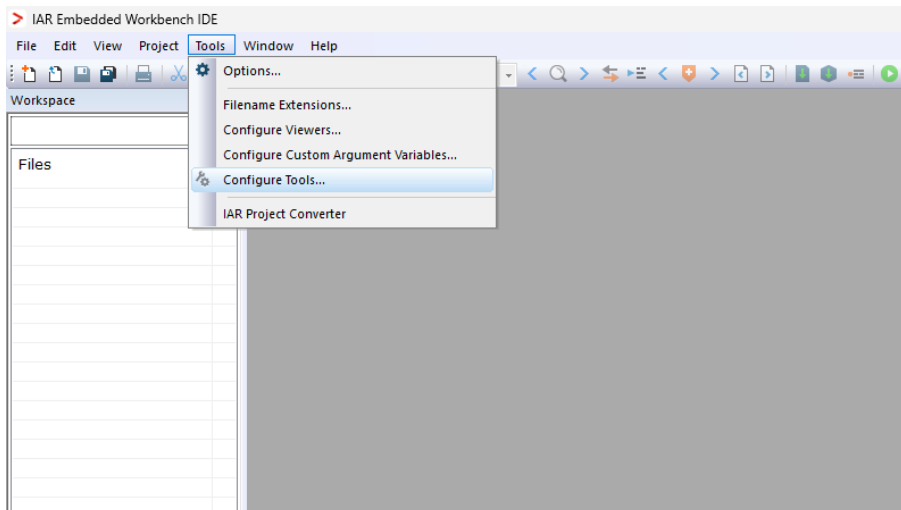


Figure 3-20 EWARM - toolbar (before setup)

2. In appeared dialog, select "New" button.
3. Fill in the fields as follows:

Table 3-1313 Setting in "Configure Tools"

Field	Input
Menu Text	FSP Smart Configurator
Command	<Path of "FSP Smart Configurator(rasc.exe)">
Argument	--compiler IAR configuration.xml
Initial Directory	\$PROJ_DIR\$

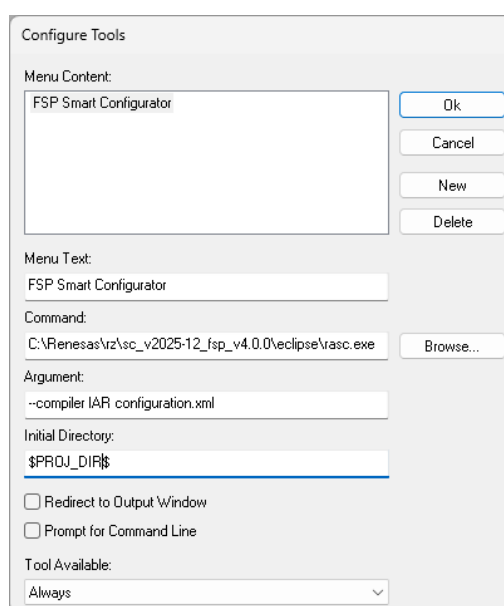


Figure 3-21 EWARM "Configure Tools"

- After setup, on toolbar, select "Tools" -> "FSP Smart Configurator".

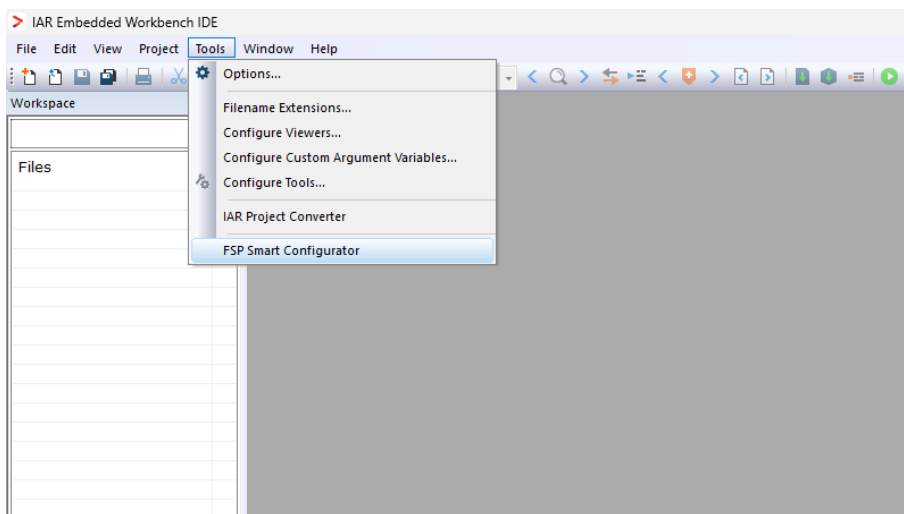


Figure 3-22 EWARM - toolbar (after setup)

FSP Smart Configurator will be launched.

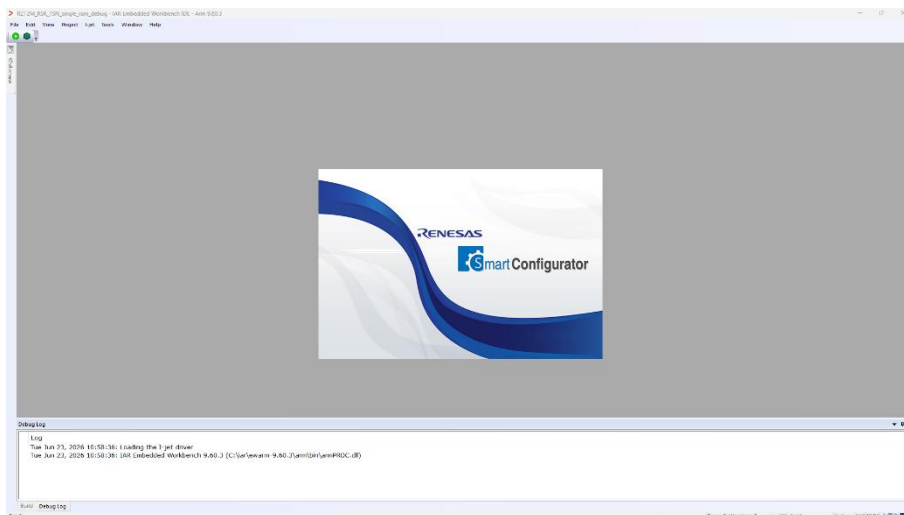


Figure 3-23 EWARM - Screen at FSP Smart Configurator start up

3.3.1.1.1.2 If using "e² studio"

- From the project explorer in e² studio, open the file named "configuration.xml".
FSP Smart Configurator will be launched.

3.3.1.1.2 Select the download destination when connecting multiple RZ/T2M-RSK or RZ/N2L-RSK

3.3.1.1.2.1 If using "EWARM"

1. On the toolbar, select "Project" -> "Options...".
Note: Or in IDE's file explorer, right click the project and select "Options...".
2. In appeared dialog, in "Category:" box, select "I-jet".
3. In "Setup" tab -> "Emulator", check the checkbox of "Always prompt for probe selection".

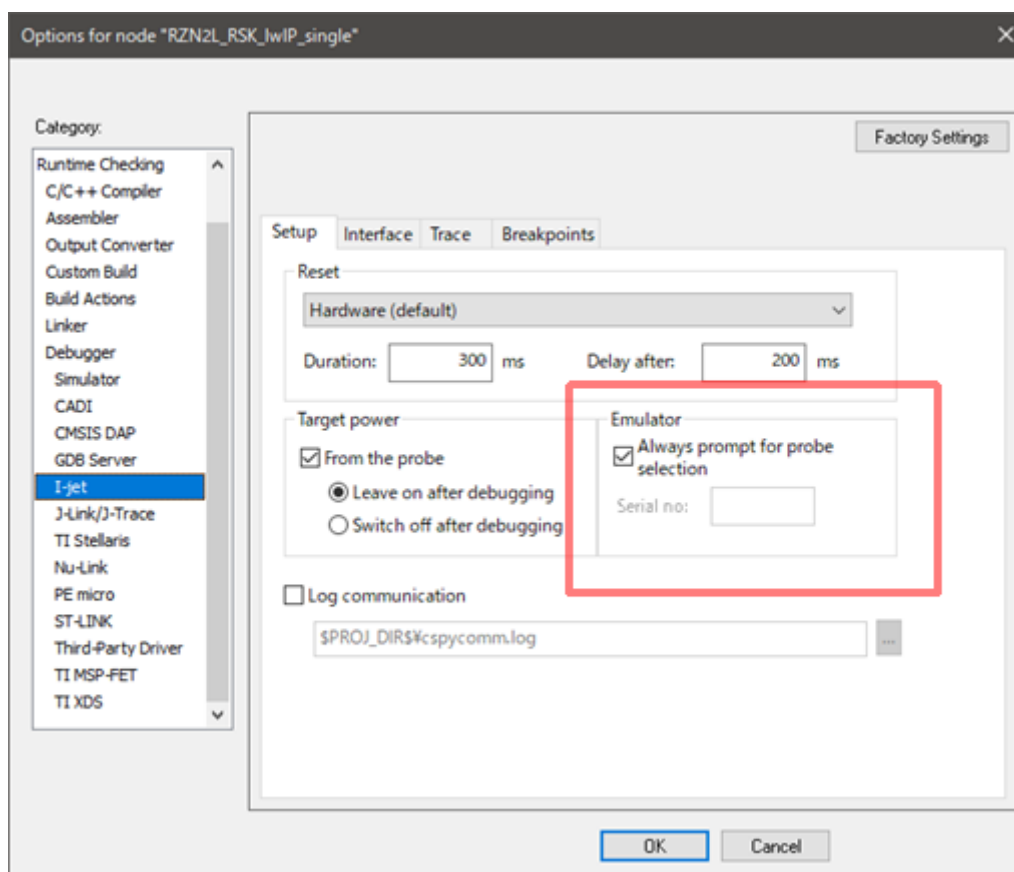


Figure 3-24 Setting dialog

After setup, the dialog of I-jet selection will appear just before application download.

In the dialog, select the I-jet connected to the board that you want to download the application to.

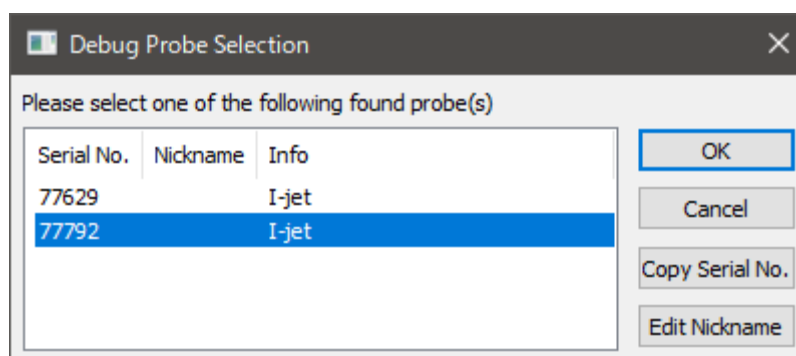


Figure 3-25 I-jet selection dialog

3.3.1.1.2.2 If using "e² studio"

1. On toolbar, click on the gear icon near "Launch Configuration".
2. From the dialog that appears, select the "Debugger" tab.
3. On the screen that appears, Select the "Connection Setting" tab.
4. On the "J-Link" -> "J-Link Serial" field, select J-Link serial number of RZ/T2M-RSK or RZ/N2L-RSK to be connected.

3.3.2 Third-Party software**3.3.2.1 Serial terminal tool**

The sample application outputs log messages via "UART".

The serial terminal tool is required to check the log messages.

• TeraTerm

<https://teratermproject.github.io/index-en.html>

The serial port settings in TeraTerm are the following:

Table 3-14 Serial connection configuration

component	setting
Baud rate	115200
Data	8bit (default)
Parity	none (default)
Stop bits	1bit (default)
Flow control	none (default)

4. Running sample application

This chapter shows the sequence of steps from building applications to running.

4.1 Build and download

4.1.1 Open project

This package contains the following 16 projects:

Table 4-1 List of projects

#	Board	IDE	Boot Mode	Directory of project	Notes
1	RZ/T2M-RSK	EWARM	RAM execution	/project/rzt2m_rsk/tsn_single/ewarm_ram_debug/	
2			xSPI boot	/project/rzt2m_rsk/tsn_single/ewarm_xspi_boot/	
3		e ² studio	RAM execution	/project/rzt2m_rsk/tsn_single/e2studio_ram_debug/	
4			xSPI boot	/project/rzt2m_rsk/tsn_single/e2studio_xspi_boot/	
5	RZ/N2L-RSK	EWARM	RAM execution	/project/rzn2l_rsk/tsn_single/ewarm_ram_debug/	
6			xSPI boot	/project/rzn2l_rsk/tsn_single/ewarm_xspi_boot/	
7		e ² studio	RAM execution	/project/rzn2l_rsk/tsn_single/e2studio_ram_debug/	
8			xSPI boot	/project/rzn2l_rsk/tsn_single/e2studio_xspi_boot/	
9	RZ/T2N-Evb	EWARM	RAM execution	/project/rzt2n_rsk/tsn_multi_cr52_cr52/ewarm_ram_debug_cpu0/	For CR52_0
10				/project/rzt2n_rsk/tsn_multi_cr52_cr52/ewarm_ram_debug_cpu1/	For CR52_1
11			xSPI boot	/project/rzt2n_rsk/tsn_multi_cr52_cr52/ewarm_xspi_boot_cpu0/	For CR52_0
12				/project/rzt2n_rsk/tsn_multi_cr52_cr52/ewarm_xspi_boot_cpu1/	For CR52_1
13		e ² studio	RAM execution	/project/rzt2n_rsk/tsn_multi_cr52_cr52/e2studio_ram_debug_cpu0/	For CR52_0
14				/project/rzt2n_rsk/tsn_multi_cr52_cr52/e2studio_ram_debug_cpu1/	For CR52_1
15			xSPI boot	/project/rzt2n_rsk/tsn_multi_cr52_cr52/e2studio_ram_xspi_boot_cpu0/	For CR52_0
16				/project/rzt2n_rsk/tsn_multi_cr52_cr52/e2studio_ram_xspi_boot_cpu1/	For CR52_1

Follow the steps below and open the project that matches each build environment.

4.1.1.1 If using "EWARM"

Unzip the zip file "RZT2M_RZN2L_TSN_RSK.zip" contains the project of sample application for EWARM.

Open the following unzipped file in EWARM.

- If using RZT2M-RSK

- RAM execution

```
/project/rzt2m_rsk/tsn_single/ewarm_ram_debug/
RZT2M_RSK_TSN_single_ram_debug.eww
```

- xSPI boot

```
/project/rzt2m_rsk/tsn_single/ewarm_xspi_boot/
RZT2M_RSK_TSN_single_xspi_boot.eww
```

- If using RZN2L-RSK

- RAM execution

```
/project/rzn2l_rsk/tsn_single/ewarm_ram_debug/
RZN2L_RSK_TSN_single_ram_debug.eww
```

- xSPI boot

```
/project/rzn2l_rsk/tsn_single/ewarm_xspi_boot/
RZN2L_RSK_TSN_single_xspi_boot.eww
```

- If using RZ/T2N-Evb

Note: Build and execution must be performed first on the CPU0-side project (the primary project).
For details, refer to "4.1.4 Build and Download to the Board (Multi-Process Version)".

- RAM execution

- ✧ CPU0-side (the primary project)

```
/project/rzt2n_evb/tsn_multi_cr52_cr52/ewarm_ram_debug_cpu0/  
RZT2N_EVB_TSN_multi_ram_debug_cpu0.eww
```

- ✧ CPU1-side (the secondary project)

```
/project/rzt2n_evb/tsn_multi_cr52_cr52/ewarm_ram_debug_cpu1/  
RZT2N_EVB_TSN_multi_ram_debug_cpu1.eww
```

- xSPI boot

- ✧ CPU0-side (the primary project)

```
/project/rzt2n_evb/tsn_multi_cr52_cr52/ewarm_xspi_boot_cpu0/  
RZT2N_EVB_TSN_multi_xspi_boot_cpu0.eww
```

- ✧ CPU1-side (the secondary project)

```
/project/rzt2n_evb/tsn_multi_cr52_cr52/ewarm_xspi_boot_cpu1/  
RZT2N_EVB_TSN_multi_xspi_boot_cpu1.eww
```

4.1.1.2 If using "e²studio"

Follow these steps to import the project of sample application.

1. Unzip the zip file " RZT2M_RZN2L_TSN_RSK.zip" contains the project of sample application for e² studios.
2. Move unzipped directory ("RZT2M_RZN2L_TSN_RSK.zip ") to the directory where you want to generate the project.
3. Launch e²studio and select "File" -> "Open Projects from fileSystem..." on toolbar.

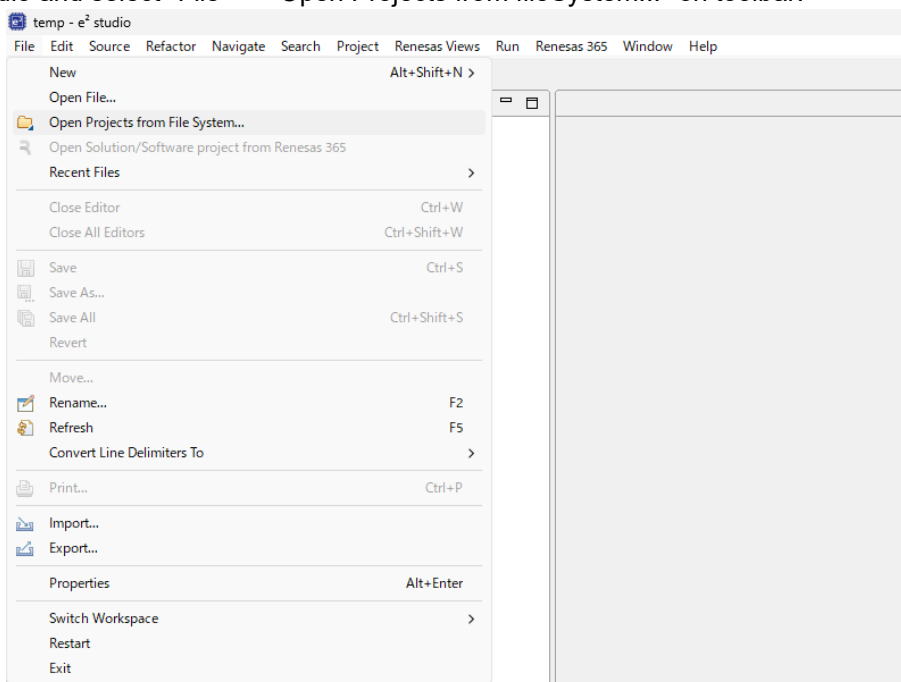


Figure 4-1 e² studio - toolbar

4. In appeared window, select "Directory" button and select the unzipped directory you just moved ("RZT2M_RZN2L_TSN_RSK").

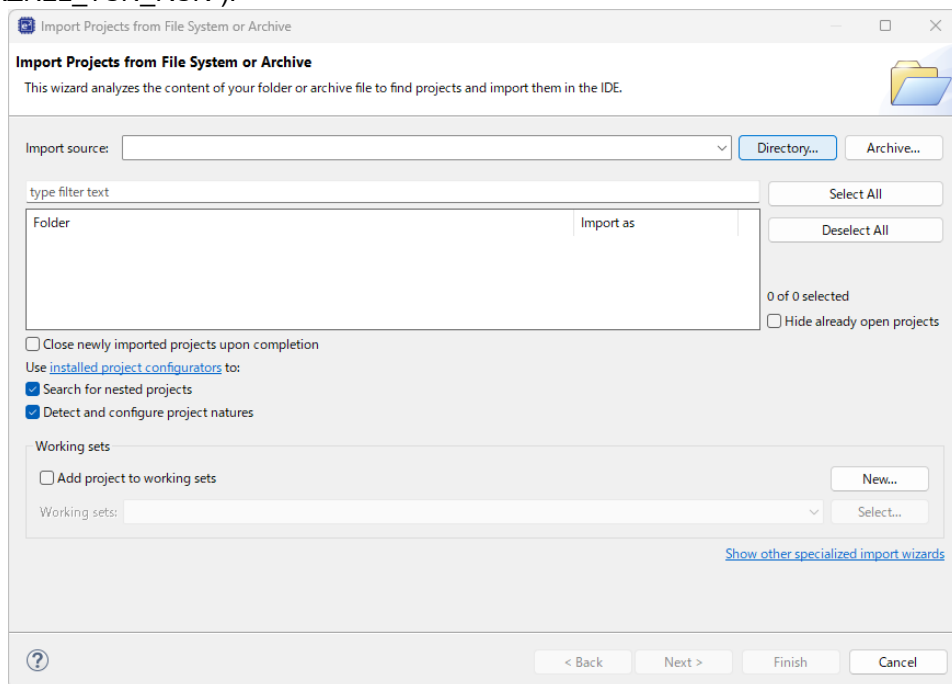


Figure 4-2 e² studio - Import Projects window (before directory selection)

5. A list of e² studio projects to be imported is displayed in the window.
Select the project to be imported. (Multiple selections possible)

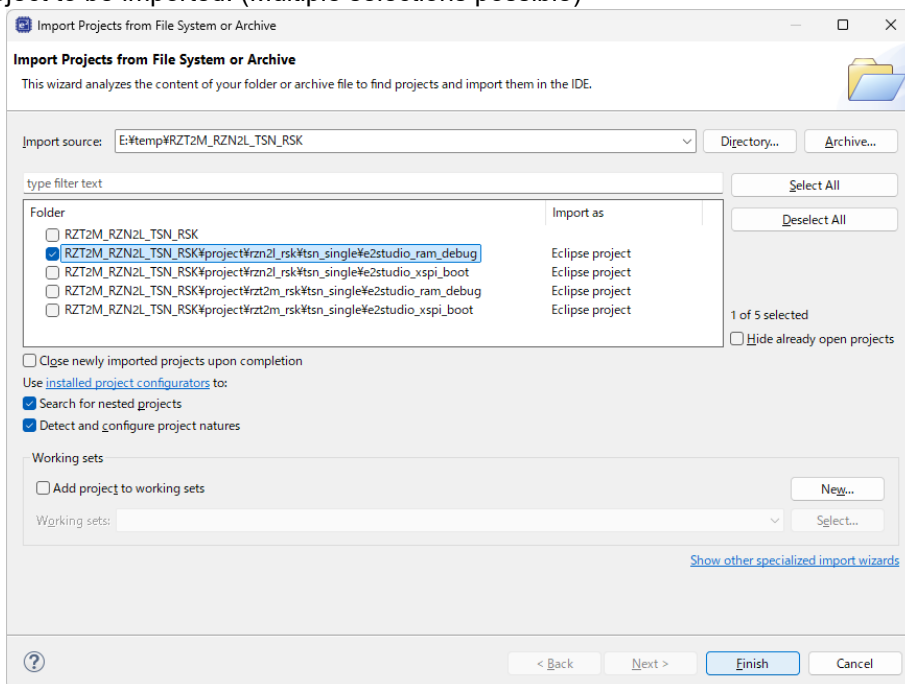


Figure 4-3 e² studio - Import Projects window (after project selection)

Note: "RZT2M_RZN2L_TSN_RSK"(the top item in the list) does not need to be selected.

6. Select "Finish button", the project will be imported.

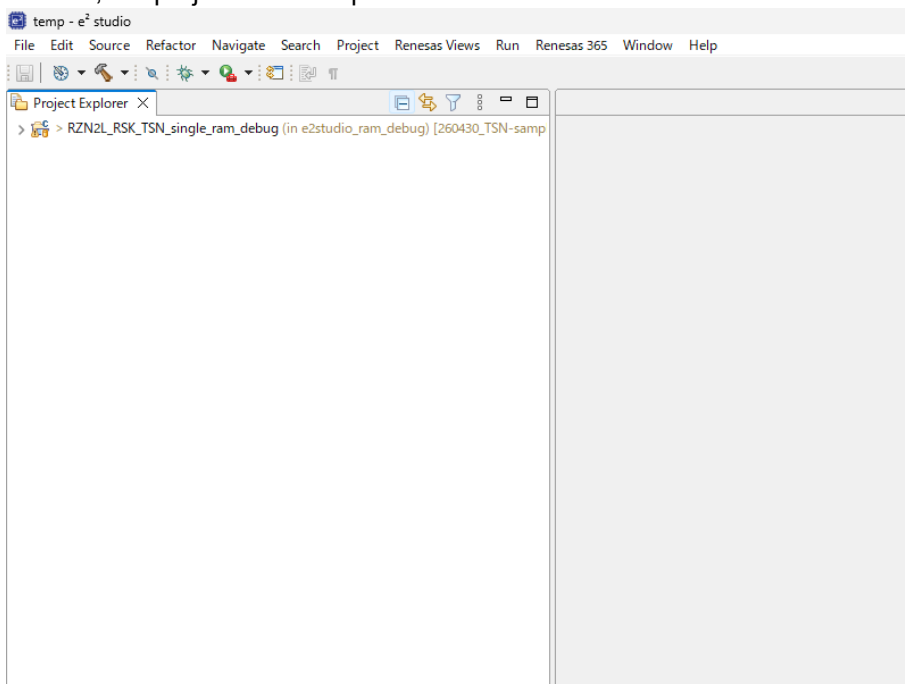


Figure 4-4 e² studio - After importing projects

4.1.2 Pre-configuration for each device

Before building, each device must be configured.

The required Pre-configurations are the as follows:

- MAC address of device
- IP address of device (*1)
- timeTransmitter/timeReceiver assignment
- Table of TDMA cycle (*2)
- Configuration of PROFINET frame (RT traffic) (*2)
- Configuration of UDP frame (NRT traffic) (*2)
- Clock pulse wave's period (*2)
- Configuration of Frame Preemption
- UDP source MAC address (*3)
- Configuration of PSFP
- Configuration of CQF(*4)

Note *1: This configuration is required only when operating with a three-device setup like the "4.4 Example of Operation Check for Sample Application (three-device setup)".

Note *2: This configuration must be consistent across all devices.

Note *3: This configuration is required when causing preemption without using preempt by queue closing (, and with TDMA table not changed from "Table 5-7 TDMA cycle table (for Frame Preemption)").

Note *4: This configuration is required only when operating with a three-device setup like the "4.4 Example of Operation Check for Sample Application (three-device setup)" for the purpose of check the operation of CQF.

Note *5: When using a multi-process project, configure these settings in the CPU1 project.

4.1.2.1 Change MAC address of device

1. Launch "FSP Smart Configurator" from IDE.

Note: For information on how to launch, see "3.3.1.1.1 Launch FSP Smart Configurator from IDE"

2. At FSP Smart Configurator, In bottom of "FSP Configuration" view, open "Stack" tab.
3. From List of "threads" in upper left of the "FSP Configuration" view, select "g_ether0 Ethernet Driver on r_ether".
4. In Properties view, open "Module g_ether0 Ethernet Driver on r_ether" -> "General".
5. At "General", The value of "MAC address" is device's MAC address.
Change the value depending to the address you want to assign.
Note: Must be set to a different value for each device.

4.1.2.2 Change IP address of device

Note: This configuration is required only when operating with a three-device setup like the "4.4 Example of Operation Check for Sample Application (three-device setup)".

Open "app_config.h" from project's directory.

```
/common/renesas/application/app_config.h
```

The IP addresses of the timeTransmitter and timeReceiver are specified by the macros TCPIP_IP_MASTER and TCPIP_IP_SLAVE in app_config.h.

```
#define TCPIP_IP_MASTER LWIP_MAKEU32(192,168, 10,170)
#define TCPIP_IP_SLAVE LWIP_MAKEU32(192,168, 10,171)
```

Note: If device is timeTransmitter, the IP address is assigned the value defined as "TCPIP_IP_MASTER".

Note: If device is timeReceiver, the IP address is assigned the value defined as "TCPIP_IP_SLAVE".

Change the value depending on the address you want to assign.

Note: Must be set to a different value for each device.

4.1.2.3 Change timeTransmitter/timeReceiver assignment

Open "app_config.h" from project's directory.

```
/common/renesas/application/app_config.h
```

TimeTransmitter and TimeReceiver are switchable by defining the macro GPTP_MODE_MASTER in app_config.h.

```
#define GPTP_MODE_MASTER
```

Note: Only timeReceiver devices must comment on the above macro definition.

4.1.2.4 Change Table of TDMA cycle

Open "gtp_sample_thread_entry.c" from project's directory.

```
/common/renesas/application/gtp_sample_thread_entry.c
```

The TDMA cycle table is configured by tdma_slot_config in gtp_sample_thread_entry.c.

```
static tdma_slot_config_t tdma_slot_config[] =
{
    /*      Period,Qgate,Pmask, GPIO,Callback,Preempt,In_ct,Out_ct */
    { 200 * 1000, 0xFC, 0x03, 0x03, false, false,false,false},
#ifdef GPTP_MODE_MASTER
    { 200 * 1000, 0x02, 0x03, 0x00, false, false,false,false},
#else
    { 200 * 1000, 0x02, 0x03, 0x00, true, false,false,false},
#endif /* GPTP_MODE_MASTER */
    { 400 * 1000, 0x01, 0x03, 0x02, false, false,false,false},
    { 200 * 1000, 0x00, 0x00, 0x00, false, false,false,false}
};
```

Change the table to what you want to realize.

Note: For various items in the table, See "5.1.3 Operation Specifications of TDMA".

4.1.2.5 Change Table of TDMA cycle (for Frame Preemption)

Open "gptp_sample_thread_entry.c" from project's directory.

/common/renesas/application/gptp_sample_thread_entry.c

The TDMA cycle table is configured by `tdma_slot_config_preemption` in `gptp_sample_thread_entry.c`.

```
static tdma_slot_config_t tdma_slot_config_preemption[] =
{
#ifdef GPTP_MODE_MASTER
    /* Period,Queue_Gate,Port_mask,GPIO,Callback,Preempt,In_ct,Out_ct */
    { 400000, 0x02/*(0000'0010)*/, 0x03, 0x02, false, false, false, false },
    { 1000, 0x01/*(0000'0001)*/, 0x03, 0x03, false, false, false, false },
    { 100000, 0xFC/*(1111'1100)*/, 0x03, 0x02, false, false, false, false },
    { 299000, 0x01/*(0000'0001)*/, 0x03, 0x03, false, false, false, false },
    { 200000, 0x00/*(0000'0000)*/, 0x03, 0x03, false, false, false, false }
#else /*GPTP_MODE_MASTER*/
    /* Period,Queue_Gate,Port_mask,GPIO,Callback,Preempt,In_ct,Out_ct */
    { 400000, 0x02/*(0000'0010)*/, 0x03, 0x02, true, false, false, false },
    { 1000, 0x01/*(0000'0001)*/, 0x03, 0x03, false, false, false, false },
    { 100000, 0xFC/*(1111'1100)*/, 0x03, 0x02, false, false, false, false },
    { 299000, 0x01/*(0000'0001)*/, 0x03, 0x03, false, false, false, false },
    { 200000, 0x00/*(0000'0000)*/, 0x03, 0x03, false, false, false, false }
#endif /*GPTP_MODE_MASTER*/
};
```

Change the table to what you want to realize.

Note: For various items in the table, See "5.1.5 Operation Specifications of Frame Preemption".

4.1.2.6 Change Table of TDMA cycle (for CQF)

Open "gptp_sample_thread_entry.c" from project's directory.

```
/common/renesas/application/gptp_sample_thread_entry.c
```

The TDMA cycle table is configured by `tdma_slot_config_cqf` in `gptp_sample_thread_entry.c`.

```
static tdma_slot_config_t tdma_slot_config_cqf[] =
{
#ifdef GPTP_MODE_MASTER || !defined(CQF_SEND_FRAME)
    /* Period,Qgate,Pmask, GPIO,Callback,Preempt,In_ct,Out_ct */
    { 800 * 1000, 0x0B, 0x03, 0x02, false, false,false,false},
    { 200 * 1000, 0x00, 0x00, 0x02, false, false,false,false},
    { 800 * 1000, 0x07, 0x03, 0x03, false, false,false,false},
    { 200 * 1000, 0x00, 0x00, 0x03, false, false,false,false},
    { 800 * 1000, 0x0B, 0x03, 0x00, false, false,false,false},
    { 200 * 1000, 0x00, 0x00, 0x00, false, false,false,false},
    { 800 * 1000, 0x07, 0x03, 0x01, false, false,false,false},
    { 200 * 1000, 0x00, 0x00, 0x01, false, false,false,false}
#else /* GPTP_MODE_MASTER */
    /* Period,Qgate,Pmask, GPIO,Callback,Preempt,In_ct,Out_ct */
    { 800 * 1000, 0x0B, 0x03, 0x02, true, false,false,false},
    { 200 * 1000, 0x00, 0x00, 0x02, false, false,false,false},
    { 800 * 1000, 0x07, 0x03, 0x03, true, false,false,false},
    { 200 * 1000, 0x00, 0x00, 0x03, false, false,false,false},
    { 800 * 1000, 0x0B, 0x03, 0x00, true, false,false,false},
    { 200 * 1000, 0x00, 0x00, 0x00, false, false,false,false},
    { 800 * 1000, 0x07, 0x03, 0x01, true, false,false,false},
    { 200 * 1000, 0x00, 0x00, 0x01, false, false,false,false}
#endif /* GPTP_MODE_MASTER */
};
```

Change the table to what you want to realize.

Note: For various items in the table, see "5.1.7 Operation Specifications of CQF".

4.1.2.7 Change configuration of PROFINET frame (RT traffic)

Open "app_config.h" from project's directory.

```
/common/renesas/application/app_config.h
```

PROFINET frame communication is configured using the macros `RT_APP_COUNT`, `RT_APP_TASK_PRIORITY_T`, and `RT_APP_TASK_PRIORITY_RX` in `app_config.h`.

```
/* profinet config */
#define RT_APP_COUNT 1
#define RT_APP_TASK_PRIORITY_TX 6 /* profinet send task */
#define RT_APP_TASK_PRIORITY_RX 7 /* profinet recv task */
#define RT_APP_ADDR_DEST {0x00, 0x11, 0x22, 0x33, 0x44, 0xaa}
```

For each item, the following settings are available:

1. `#define RT_APP_COUNT`
 - The number of PROFINET frames transmitted in one TDMA cycle.
Note: maximum is 4 frames.

2. `#define RT_APP_TASK_PRIORITY_TX`
`#define RT_APP_TASK_PRIORITY_RX`
 - Each sets the priority of the task that sends/receives frame.
3. `#define RT_APP_ADDR_DEST`
 - Specifies the destination MAC address of the RT frame.

4.1.2.8 Change configuration of UDP frame (NRT traffic)

Open "app_config.h" from project's directory.

```
/common/renesas/application/app_config.h
```

The UDP frame communication is configured using the macros NRT_APP_TASK_PRIORITY, NRT_APP_PORT, NRT_APP_INTERVAL, NRT_APP_TX_PAYLOAD_SIZE, and NRT_APP_TX_COUNT in app_config.h.

```
/* udp perf cofig */
#define NRT_APP_TASK_PRIORITY      2          /* udp perf task */
#define NRT_APP_PORT               5000U
#define NRT_APP_INTERVAL          (10 * 1000) /* 10sec */
#define NRT_APP_TX_PAYLOAD_SIZE   1472U
#define NRT_APP_TX_COUNT          (4U)
```

For each item, the following settings are available:

1. `#define NRT_APP_TASK_PRIORITY`
 - The Priority of task for UDP traffics.
2. `#define NRT_APP_PORT`
 - The Port number for UDP traffic.
3. `#define NRT_APP_INTERVAL`
 - The interval of throughput measurements.
4. `#define NRT_APP_TX_PAYLOAD_SIZE`
 - The Size of payload in UDP frame.
 - Note: The actual frame size to be sent is this value plus the size of Ethernet-Header and UDP-Header.
5. `#define NRT_APP_TX_COUNT`
 - The number of UDP frames transmitted in one TDMA cycle.
 - Note: If the number of UDP frames is set too great, it may not be possible to send or receive the specified number of frames, or log message output may stop.

4.1.2.9 Change configuration of the pulse generator period

Open "device_properties.h" from project's directory.

```
/common/renesas/module/ptp/device_properties.h
```

The pulse generator period can be config 1[s] or 1[ms] by defining the macro GPTP_OUTPUT_PULSE_PERIOD in device_properties.h.

- To set the pulse generator period to 1[s], set GPTP_PULSE_PERIOD_SEC.

```
#define GPTP_OUTPUT_PULSE_PERIOD  GPTP_PULSE_PERIOD_SEC
```

- To set the pulse generator period to 1[ms], set GPTP_PULSE_PERIOD_MSEC.

```
#define GPTP_OUTPUT_PULSE_PERIOD  GPTP_PULSE_PERIOD_MSEC
```

4.1.2.10 Change configuration of Frame preemption

4.1.2.10.1 Change detail configuration of Frame preemption

Open "gptp_sample_thread_entry.c" from project's directory.

```
/common/renesas/application/gptp_sample_thread_entry.c
```

The Frame preemption is configured by qbu_config in gptp_sample_thread_entry.c.

```
qbu_config_t qbu_config =
{
    /* queue,  qclose */
    { {0x01}, {0x01} },
    {
        /*P1*/{ true, { false, false, 0, 100, true, false, false, false, false} },
        /*P2*/{ true, { false, false, 0, 100, true, false, false, false, false} },
        /*P3*/{ true, { false, false, 0, 100, true, false, false, false, false} },
    }
};
```

The contents of each configurations are as follows:

Table 4-2 Configurations of Frame Preemption (1 of 2)

Components		Field name	Description	Note
Global	preempt_queues	preempt_queue.preempt_queues	Specification of the Preemptable Queues	
	preempt_on_qclose	preempt_queue.preempt_on_qclose	Specification of the queue to enable preempt by queue closing.	It becomes effective only when the specified queue is Preemptable queue.
Each ports	Enable	preempt_port_ctrl[Port number].enable	Enabling Preemption	
	Verify dis-enable	preempt_port_ctrl[Port number].config.verify_dis	Disabling verification process	
	Response dis-enable	preempt_port_ctrl[Port number].config.response_dis	Disabling response for verification frame	
	Frag size	preempt_port_ctrl[Port number].config.addfragsize	Minimum fragment size of the frame divided by Preemption	The size is adjusted in units of 64 bytes (64~256). The allocation of each value is as follows: • 0: 64 • 1: 128 • 2: 192 • 3: 256
	Verify time-out	preempt_port_ctrl[Port number].config.tx_verify_time	Timeout of verification process	1 ~ 127 ms Note: To satisfy the 20% accuracy required by 802.3br, a value of 5 ms or more is recommended.

Table 4-3 Configurations of Frame Preemption (2 of 2)

Components		Field name	Description	Note
Each ports	Pre-amble check	preempt_port_ctrl[Port number].config.rx_strict_pre	Selection of Preamble checking method	• true: the preamble is checked so all bytes except the SFD are 0x55. • false: only the last 2 bytes of the preamble are checked (SFD/SMD and FRAG_COUNT). Note: It is recommended to set this bit to "true" to comply with the 802.3br specification.
	No de-code SMD	preempt_port_ctrl[Port number].config.rx_br_smd_dis	Disabling decoding SMD	Note: When disabled, all frames are assumed as express frame.
	Strict check	preempt_port_ctrl[Port number].config.rx_strict_br_ctl	Enabling strict checking of VERIFY and RESPONSE frames	When set to true, the frame contents and frame length checks are also performed on these frames. Note: To be compliant with the functionality described in IEEE 802.3br, this field must be set to false.
	TX CRC in-version	preempt_port_ctrl[Port number].config.tx_mcrc_inv	Enabling inversion of the 32-bit XOR mask used to calculate the mCRC for transmitted frames	Note: This configuration is only used for debugging. During normal operation, this field must always be set to "false".
	RX CRC in-version	preempt_port_ctrl[Port number].config.rx_mcrc_inv	Enabling inversion of the 32-bit XOR mask used to calculate the mCRC for received frames	Note: This configuration is only used for debugging. During normal operation, this field must always be set to "false".

4.1.2.11 Change UDP source MAC address

Note: This configuration is required when causing preemption without using preempt by queue closing (, and with TDMA table not changed from "Table 5-7 TDMA cycle table (for Frame Preemption)").
 For details, see: "5.1.5.1 How to check Preemption without using a Close Preempt queue".

Open "gtp_sample_thread_entry.c" from project's directory.

```
/common/renesas/application/gtp_sample_thread_entry.c
```

The UDP source MAC addresses is specified by the macro TESTPACKET_MACADDR in gtp_sample_thread_entry.c.

```
#define TESTPACKET_MACADDR {0x00,0x00,0x5E,0x00,0x53,0x00}
```

4.1.2.12 Change configuration of PSFP

4.1.2.12.1 Change configuration of filters

Open "gptp_sample_thread_entry.c" from project's directory.

```
/common/renesas/application/gptp_sample_thread_entry.c
```

PSFP filters can be configured by editing the following variables in "gptp_sample_thread_entry.c":

Table 4-4 Filter configurations for PSFP

Item	Variable	Member	description
Configuration of "Stream filter"	g_efp_cfg_qci_stream_filter_entry	MAC_select	MAC address selection
		MAC	MAC address designation
		MAC_mask	MAC address designation mask
Enabling of "Stream filter"	g_efp_cfg_qci_stream_filter_ena		Stream filter enabling
Configuration of "Stream gate filter"	g_efp_cfg_gating_check_entry	enable	Stream gate filter enabling
		gateID	Stream gate filter ID
		gate mode	Stream gate filter mode
Configuration of "SDUmax filter"	g_efp_cfg_sdu_max_entry	enable	SDUmax filter enabling
		maxSDUsize	SDUmax filter size
		MSDU mode	SDUmax filter mode
Configuration of "Flow metering"	g_efp_cfg_flow_meter_entry	id_enable	Flow metering ID enable
		id	Flow metering ID
		coupling	Coupling flag
		mode	Flow metering mode
		red_drop	Red frame rejection enabling
		burst_size	CBS (Committed Burst Size)
		info_rate	CIR (Committed Info Rate)
		enable	Flow metering enabling

4.1.2.12.2 Change configuration of PSFP's log

Open "app_config.h" from project's directory.

```
/common/renesas/application/app_config.h
```

For each item, the following settings are available:

1. `#define PSFP_PORT_MASK`
 - specifies the port for outputting PSFP logs.
 - Specify the port numbers to which the filter settings are applied using a bitmask.
2. `#define PSFP_STREAM_MASK`
 - specifies the stream ID for outputting PSFP logs.
 - Specify the stream IDs to which the filter settings are applied using a bitmask.

4.1.2.13 Change configuration of CQF

Note: This configuration is required only when operating with a three-device setup for the purpose of check the operation of Qch.

Open "app_config.h" from project's directory.

```
/common/renesas/application/app_config.h
```

The definition of `CQF_SEND_FRAME` in "app_config.h" enables the sending of PROFINET frames from `timeReceiver` in the CQF sample.

```
#define CQF_SEND_FRAME
```

Note: In a three-device setup like the "4.4 Example of Operation Check for Sample Application (three-device setup)", comment out this macro only in `timeReceiver` (client) that is directly connected to `timeTransmitter` (server).

4.1.3 Build and download to the board

Note: If you are using a multi-process project, refer to "4.1.4 Build and Download to the Board (Multi-Process Version)"

4.1.3.1 Generate Project contents

Before building, generate project contents using FSP Smart Configurator.

1. Launch "FSP Smart Configurator" from IDE.
Note: For information on how to launch, see "3.3.1.1.1 Launch FSP Smart Configurator from IDE".
2. In FSP Smart Configurator, select "Generate Project Content" in the upper right of the "FSP Configuration" view. Project content will be created.

4.1.3.2 Build and download

4.1.3.2.1 If using "EWARM"

1. After the creation of project content, select "Project" -> "Make" on toolbar.
The application will be built.
2. After building, select "Project" -> "Download and Debug" on toolbar.
The application downloading to target board will be started.

4.1.3.2.2 If using "e² studio"

1. After the creation of project content, select "Project" -> "build" on toolbar.
The application will be built.
2. After building, select "Run" -> "Debug" on toolbar.
The application downloading to target board will be started.

4.1.4 Build and Download to the Board (Multi-Process Version)

4.1.4.1 Generate Project Source and Run the Build (Multi-Process Version)

1. Execute code generation in the CPU0-side project (primary project).
Note: For details on code generation, refer to "4.1.3.1 Generate Project contents".
2. Build the CPU0-side project.
Note: For details on how to build, refer to "4.1.3.2 Build and download".
Note: When building an xSPI project in EWARM, additional steps described in "4.1.4.1.1 Notes on Building with EWARM" are required.
3. Execute code generation in the CPU1-side project.
Note: You cannot perform code generation on the CPU1 side unless you have already built the CPU0-side project.
4. Build the CPU1-side project.
5. Clean the CPU0-side project and build it again.
The cleaning procedure is as follows:
 - a. For EWARM
 - i. Select "Clean (L)" from the "Project (P)" tab.
 - b. For e²studio
 - i. Right-click the project you want to clean in the Project Explorer.
 - ii. Select "Clean Project" from the context menu.

4.1.4.1.1 Notes on Building with EWARM

When building a multi-process xSPI project in EWARM, the bin file in the Debug folder of the CPU1-side project must be present.

Note: If it is not present, a build error will occur in the CPU0-side project.

Because the bin file does not exist during the first build, place a dummy bin file by following the steps below:

1. ...

4.1.4.2 Download (Multi-Process Version)

4.1.4.2.1 For EWARM

1. Disable the debugger function "Asymmetric Multicore".
Note: If you do not perform this operation, you will not be able to carry out some of the subsequent steps.

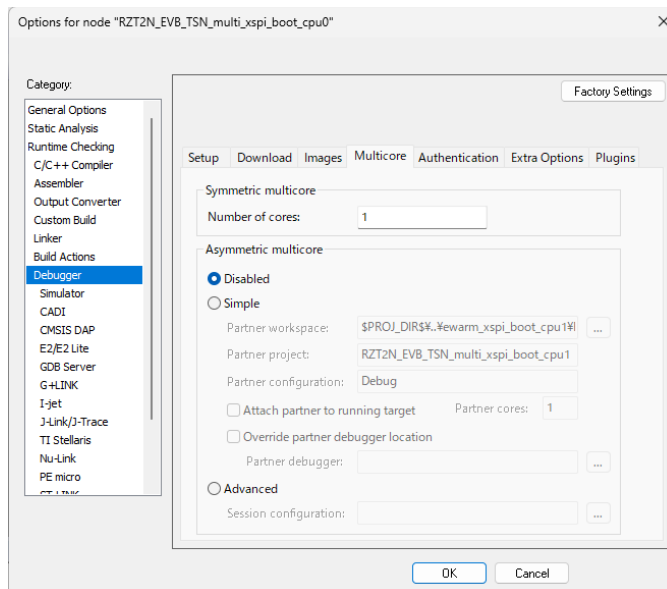


Figure 4-5 EWARM Disabling "Asymmetric Multi-Core"

2. (If necessary) Delete flash memory.
For details, refer to "3.2.1.1.1 How to Erase Flash Memory".
3. (xSPI execution only)
In the CPU0-side project, select "Project (P)" tab → "Download (W)" → "Download File... (O)", and then select the *.out file for CPU0 generated in the Debug/Exe/ directory.

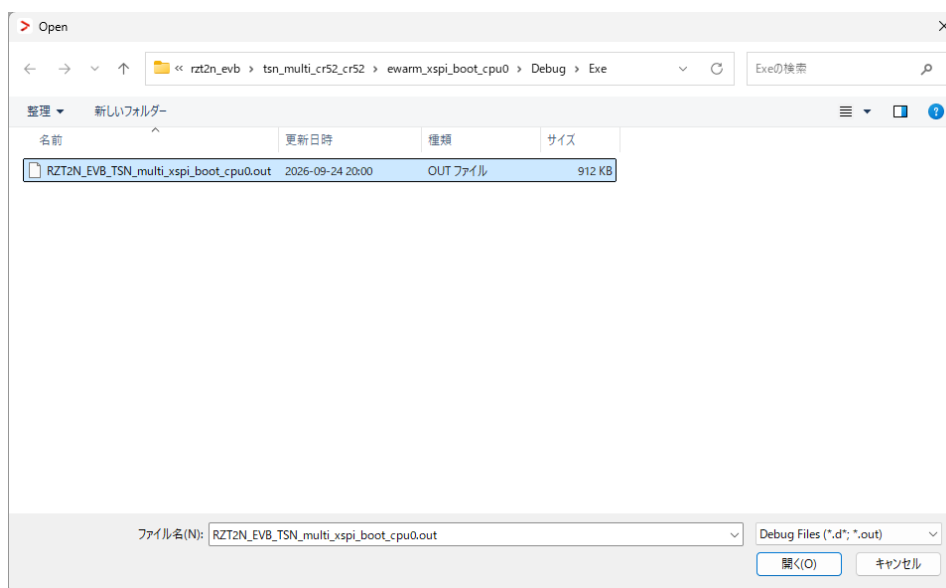


Figure 4-6 File Download

Note: If you perform this operation, you must disable the debugger function "Asymmetric Multicore" in advance.

This downloads the CPU0 program to the CPU0 side of the board.

4. Enable the debugger function "Asymmetric Multicore" (restore the original setting).

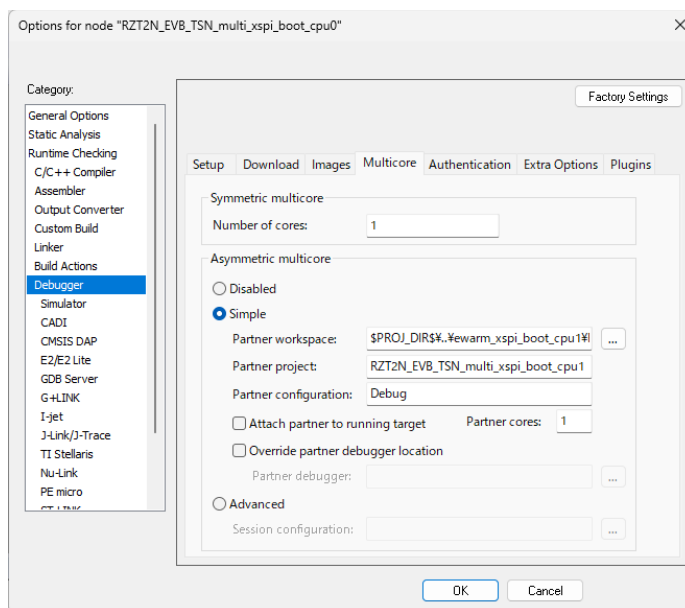


Figure 4-7 EWARM Re-enabling "Asymmetric Multicore"

5. Launch the debugger on the CPU0 side.

Note: The procedure differs depending on whether the system is running from RAM or xSPI.

- (When running from RAM)
In the CPU0-side project, select "Project (P)" → "Download and Debug (D)".
- (When running from xSPI)
In the CPU0-side project, select "Project (P)" → "Debug Without Downloading (H)".

Note: When the debugger is started on the CPU0 side, the debugger on the CPU1 side is also started automatically.
(A separate window will open.)

6. (When running from xSPI)
Start the debugger on the CPU0 side.

When running from xSPI, the CPU1 program is downloaded to the CPU1 side of the board at this point.

4.1.4.2.2 For e² studio

1. (If necessary) Delete flash memory.

For details, refer to "3.2.1.1.1 How to Erase Flash Memory".

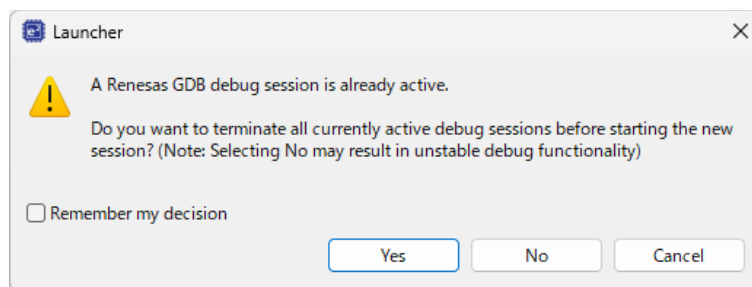
2. On the CPU0 side, select "Run" tab → "Debug" to start the debugger.

The CPU0 program is downloaded to the CPU0 side of the board.

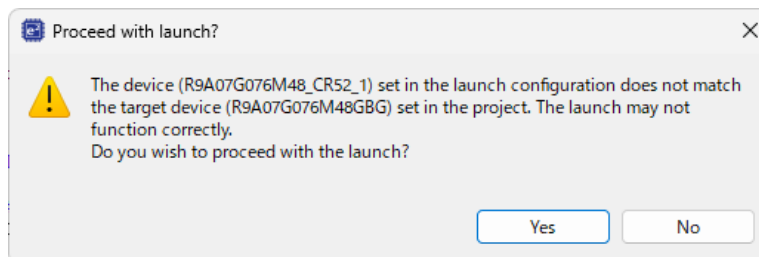
Note: Do not start the debugger yet at this point.

Leave the CPU0-side debugger in the waiting-to-start state, and proceed to the next step.

3. On the CPU1 side, select "Run" tab → "Debug" to start the debugger.
 - a. In the dialog box that says "Stop the currently running debugger?", select "No."

Figure 4-8 e² studio before starting the Debugger (Part 1)

- b. If a dialog box saying "Proceed with launch?" appears, select "Yes."

Figure 4-9 e² studio before starting the Debugger (Part 2)

This downloads the CPU1 program to the CPU1 side of the board.

4.2 Run sample application

4.2.1 When Running from RAM

For RAM execution, when the program download to the board is completed, the application enters the standby state for execution.

Follow these steps to start running application.

Note: For details on the operation check, see "5.2 Sample application's outputs for checking operation".

4.2.1.1 When using a single process

4.2.1.1.1 If using "EWARM"

Select "Debug" -> "Go" on toolbar.

The application will start running.

4.2.1.1.2 If using "e² studio"

Select "Run" -> "Resume" on toolbar.

The application will start running.

4.2.1.2 When using multiple processes

Note: For a multi-process project, this application runs on the CPU1 side.
The program on the CPU0 side does not execute anything.

4.2.1.2.1 If using "EWARM"

1. In the CPU0-side project, select "Debug" → "Run" from the toolbar.
2. In the CPU1-side project, select "Debug" → "Run" from the toolbar.

This will launch the application.

Note: If you want to stop the application, execute Stop on the CPU1 side.
When CPU1 is stopped, CPU0 is also stopped automatically.

Note: If you stop CPU0, the program may not start correctly the next time you debug it.
In that case, execute Stop from the CPU1 side.
(This will allow the application to start normally from the next time onward.)

4.2.1.2.2 If using "e² studio"

1. From the Debug window, right-click the debugger on the CPU0 side and select "Resume".

Note: If execution breaks, select "Resume" again.

2. From the Debug window, right-click the debugger on the CPU1 side and select "Resume".

Note: If execution breaks, select "Resume" again.

This will launch the application.

4.2.2 When Running from xSPI

For xSPI execution, no additional operation is required.

After downloading the program, the program will start by turning the board power off and on again.

Note: For details of the operation check, refer to "5.2 Sample application's outputs for checking operation".

4.3 Example of Operation Check for Sample Application

Below are the operation verification steps for a configuration using two RSK boards, with the one as the master and the other one as the slave:

1. Set the device as master/slave.

(A) Set the first device as *timeTransmitter*:

(1) Configure the device as *timeTransmitter* (see 4.1.2.3).

(2) Set the MAC address to `11:22:33:44:55:aa` (see 4.1.2.1).

(B) Set the second device as *timeReceiver*:

(1) Configure the device as *timeReceiver* (see 4.1.2.3).

(2) Set the MAC address to `11:22:33:44:55:bb` (see 4.1.2.1).

Note: After configuration, build and write the program on each device.

2. Connect the **ETH0** ports of each device with an Ethernet cable.

The configuration is as follows:

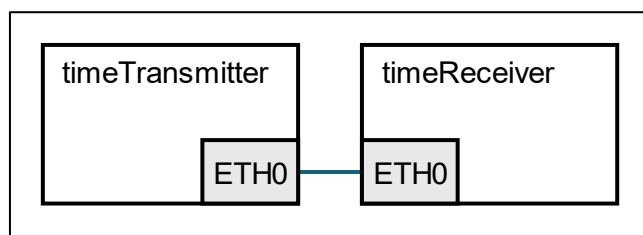


Figure 4-10 Example of Connection

3. Start the program on each device.
4. (Then, the operation of gPTP + TDMA can be verified using the method described in section 5.2.)

4.4 Example of Operation Check for Sample Application (three-device setup)

Below are the operation verification steps for a configuration using three RSK boards, where the first board operates as the timeTransmitter and the second and third boards operate as timeReceivers.

1. Set the device as master/slave.

(A) Set the first device as timeTransmitter:

- (1) Configure the device as *timeTransmitter* (see 4.1.2.3).
- (2) Set the MAC address to `11:22:33:44:55:aa` (see 4.1.2.1).
- (3) Set the IP address to `192.168.10.170` (see 4.1.2.13).

(B) Set the second device as timeReceiver1:

- (1) Configure the device as *timeReceiver* (see 4.1.2.3).
- (2) Set the MAC address to `11:22:33:44:55:bb` (see 4.1.2.1).
- (3) Set the IP address to `192.168.10.171` (see 4.1.2.2).
- (4) Comment out the `#define CQF_SEND_FRAME` of CQF configuration (See 4.1.2.13).

(C) Set the third devices as timeReceiver2:

- (1) Configure the device as *timeReceiver* (see 4.1.2.3).
- (2) Set the MAC address to `11:22:33:44:55:cc` (see 4.1.2.1).
- (3) Set the IP address to `192.168.10.172` (see 4.1.2.2).

Note: After configuration, build and write the program on each device.

2. Connect the **ETH0** ports of each device with an Ethernet cable.

The configuration is as follows:

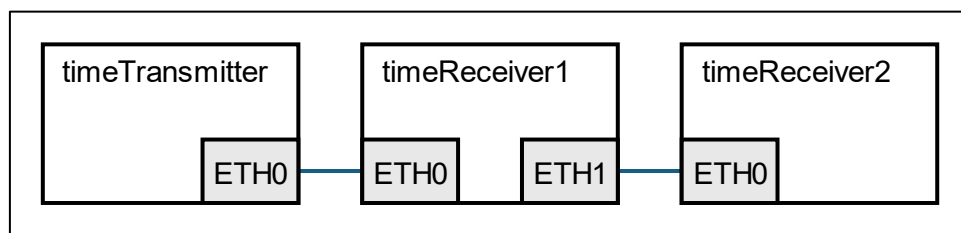


Figure 4-11 Example of Connection

3. Start the program on each device.

5. gPTP+TDMA sample application

This sample application is the simple TSN application realized with gPTP and TDMA used in the environment of face-to-face connection between boards.

5.1 Sample application Overview

By connecting ETH0 port of each boards to each other(*2)(*3), gPTP time synchronization starts. At time synchronization has been successful, TDMA is activated and the application begins operation.

At start of gPTP, each RZ/T2M-RSK or RZ/N2L-RSK are assigned as "timeTransmitter" or "timeReceiver".(*1)

Note *1: In this application, the packet-receiving (server) device operates as the *timeTransmitter*, and the packet-sending (client) device operates as the *timeReceiver*.

Therefore, it is necessary to configure the device in advance to be assigned either as the *timeTransmitter* or the *timeReceiver*.

See "4.1.2.3 Change timeTransmitter/timeReceiver assignment".

Note *2: It is possible to connect via ETH1 as well, but PROFINET frames are transmitted only from ETH0. Therefore, the port that connects to the timeTransmitter device must use ETH0.

Note *3: ETH2 is not available.

Note *4: In a multi-process project, this application runs on the CPU1 side.
The program on the CPU0 side is designed to perform no execution.

At TSN is active, This sample application sends PROFINET frame as RT(Real-Time)traffic and UDP frame as NRT(Non-Real-Time)traffic(*1)(*2).

In TDMA within this application, PROFINET frames are transmitted in dedicated TDMA slots(*3), while UDP frames are transmitted in default-frame TDMA slots. (*2)

When Frame preemption is active, this sample application sends RT traffic as express frame. Then, NRT traffic is preempted. (*4)

When Qci is active, frames PROFINET frame and UDP frame are targeted for filtering, and the filtering is performed when the timeTransmitter receives them.

When Qch is active, the transmission of PROFINET frames are performed periodically. See "2.5 IEEE802.1Qch (CQF)".

Additionally, in this case, the number of transmissions of PROFINET frame is fixed at the maximum of 4. When LLDP is active, the LLDP frame is also transmitted in addition to the PROFINET frame and the UDP frame.

Note *1: The "timeReceiver" device sends a frame (operating as a client), and the "timeTransmitter" device receives it (operating as a server).

Note *2: The TDMA cycle table is editable.
For editing instructions, refer to "4.1.2.4 Change Table of TDMA cycle".

Note *3: PROFINET frames are used only as test RT frames; this application does not implement or operate the PROFINET protocol.

Note *4: However, while hold_req is enabled, the preemptable queue remains blocked (UDP frames will not be transmitted), so preemption will not occur.

Each frame traffic has the following specifications:

- PROFINET frames are sent 1 frame per TDMA cycle.
 - The number of PROFINET frames sent per TDMA cycle are changeable(*1).
See "4.1.2.7 Change configuration of PROFINET frame (RT traffic)".
 - Note *1: maximum is 4 frames.
 - When multiple transmissions occur, frame IDs are assigned sequentially starting from 0x101 for each transmitted frame.
 - ✧ The queues to send frames are selected based on the frame ID.
(See the table below)
 - ✧ At timeTransmitter device has received PROFINET frame, device toggles the LEDs on the board according to the ID of received frame.
 - ✧ The correspondence table between Queue and LEDs in each frame ID is as follows:

Table 5-1 The correspondence table between Queue and LEDs in each frame ID

Frame ID	Queue	LED
0x101	2	LED0
0x102	3	LED1
0x103	4	LED2
0x104	5	LED3

- UDP frames are sent the specified number per TDMA cycle.
 - The number and size of UDP frames sent are changeable.
See "4.1.2.8 Change configuration of UDP frame (NRT traffic)".

Also, this sample application has the following functions for operation checks:

- UART outputs information of applications states.
- GPIO pins outputs clock pulse wave (for checking time synchronization) and information of TSN's states.

5.1.1 Sample menu

The menu launches immediately after the device starts up.

When a function is selected from the menu, the corresponding sample application will start.

This application performs output and input in the menu via UART from the USB to Serial Port of the boards.

The assignment and placement of the "USB to Serial Port" on each board are shown below.

Table 5-2 Assignment of "USB to Serial Port"

Target board	Assignment
RZ/T2M-RSK	CN16
RZ/N2L-RSK	
RZ/T2N-Evb	J33

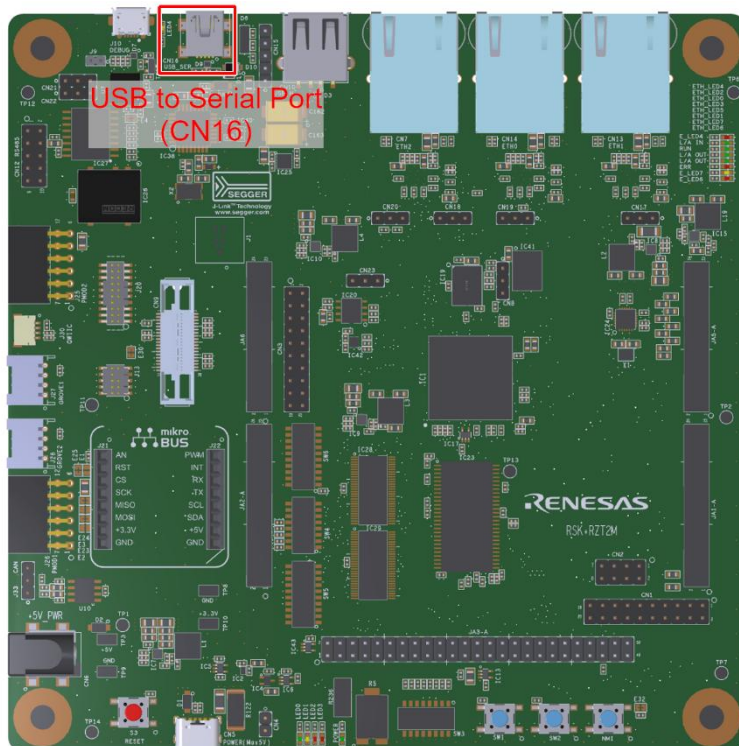


Figure 5-1 RZ/T2M board layout (USB to Serial Port USB Connector)

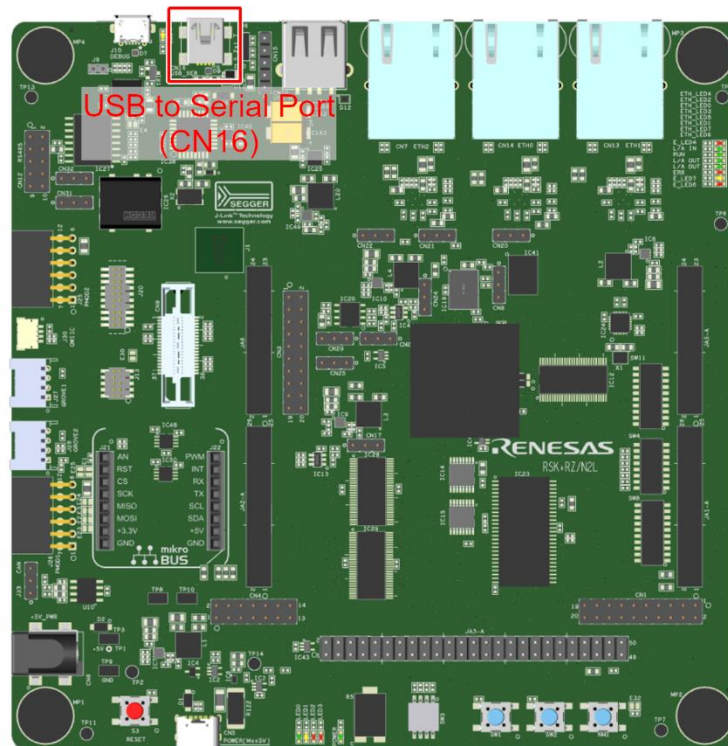


Figure 5-2 RZ/N2L board layout (USB to Serial Port USB Connector)

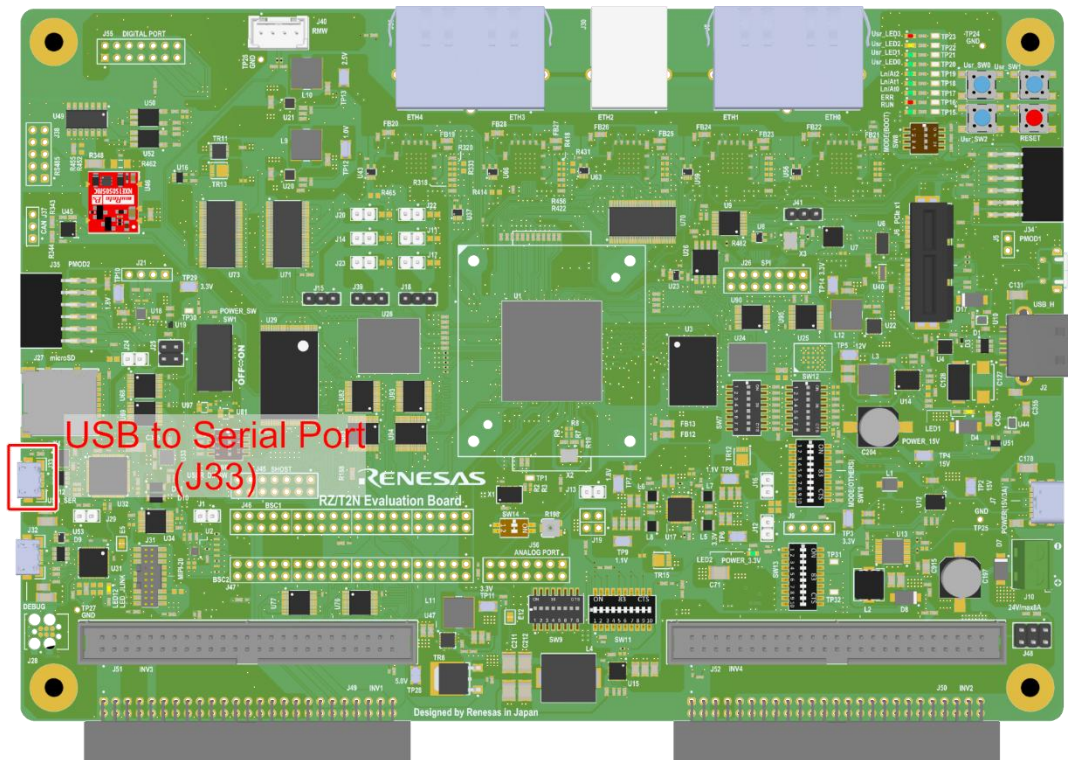


Figure 5-3 RZ/T2N board layout (USB to Serial Port USB Connector)

The displayed menu is as follows:

```
[Menu]
[0]: gPTP
[1]: Preemption
[2]: Preemption(Hold Request)
[3]: Qch
[4]: Qci
[5]: LLDP
>>
```

The functions corresponding to each are as follows:

Table 5-3 List of Menu Items and Corresponding Functions

number	title	function
0	gPTP	gPTP
1	Preemption	gPTP + Preemption
2	Preemption(Hold Request)	gPTP + Preemption (use hold_req)
3	Qch	gPTP + CQF
4	Qci	gPTP + PSFP
5	LLDP	gPTP + LLDP

Menu selection is made by inputting the number assigned to each function.

5.1.2 Operation Specifications of gPTP

Operation Specifications of gPTP are as follows:

Table 5-4 settings of gPTP

SYNC send interval	PDELAY send interval
125msec	1000msec

5.1.3 Operation Specifications of TDMA

Operation Specifications of TDMA are as follows:

- Timing of "start/stop"

Table 5-5 Timing of start/stop

device	Timing of TDMA start	Timing of TDMA stop
timeTransmitter	"NETDEV_UP" notified	"NETDEV_DOWN" notified
timeReceiver	"CLOCK_SYNCED" notified	"CLOCK_NOTSYNC" notified

- Settings of each TDMA slot (TDMA cycle table)

Note: The table of TDMA cycle is customizable.

See "4.1.2.4 Change Table of TDMA cycle".

Note: If Frame preemption is active, the TDMA cycle table is not same this.

See "5.1.5 Operation Specifications of Frame Preemption".

Note: If CQF is active, the TDMA cycle table is not same this.

See "5.1.7 Operation Specifications of CQF".

Table 5-6 TDMA cycle table

Slot	Period [us]	Queue-Gate (*2)(*3)								Port Mask (*2)				Trigger (activate callback)	Preemption hold_req	In cut- through	Out cut- through	Description (*3)
		7	6	5	4	3	2	1	0	2	1	0	1	0				
0	200	0	0	0	0	0	0			0	0	H	H	false	false	false	false	RT traffic slot
1	200							0		0	0	L	L	true(*1)	false	false	false	PTP traffic slot
2	400								0	0	0	H	L	false	false	false	false	NRT traffic slot
3	200											L	L	false	false	false	false	Guard band

Note *1: Set the parameter as follows:

"timeReceiver" device set "true".

"timeTransmitter" device set "false".

Note *2: "O" indicates queue open, and a blank indicates queue close.

Note *3: In this application, Queue 0 is assigned to NRT frames, Queue 1 is assigned to PTP, and Queue 2-7 are assigned to RT frames.

- TDMA cycle (The sum of the Period of each slot)

: **1000[us]**

5.1.4 Operation Specifications of RT frame and NRT frames

When RT frame and NRT frame are active, each device performs the following operations.

- "TimeReceiver" device
 - TimeReceiver sends PROFINET frame and UDP frame to GMAC at triggered by the slot1 actives while TSN is running.

Note: The timing of ETHSW send each frame actually are as follows:

 - PROFINET: Slot0
 - UDP.....: Slot2
- "TimeTransmitter" device
 - toggles the LEDs when it receives a PROFINET frame.
 - Which LED is toggled is decided by the received PROFINET frame's ID.

Note: The correspondence table is in "5.1 Sample application Overview".

5.1.5 Operation Specifications of Frame Preemption

This sample application controls frame transmission timing using TDMA and intentionally causes collisions in the transmission timing of frames from different queues to simulate conditions where preemption occurs.

This allows verification of the device's Frame Preempt functionality.

Therefore, a dedicated TDMA cycle table, different from the normal one, is applied when Frame Preemption is enabled.

The TDMA cycle table when Frame Preemption is enabled is as follows.

Note: The TDMA cycle table is editable.

For editing table, see "4.1.2.5 Change Table of TDMA cycle (for Frame Preemption)".

Table 5-7 TDMA cycle table (for Frame Preemption)

Slot	Period [us]	Queue-Gate (*2)(*3)								Port Mask (*2)			gpio pattern (TDMAOUTn)		Trigger (activate callback)	Preemption hold_req	Out cut- through	In cut- through	Description (*3)
		7	6	5	4	3	2	1	0	2	1	0	1	0					
0	400						0			0	0	L	L	true(*1)	false	false	false	PTP traffic slot	
1	1							0		0	0	H	L	false	false	false	false	NRT traffic slot	
2	100	0	0	0	0	0	0			0	0	H	H	false	false	false	false	RT traffic slot	
3	299							0		0	0	H	L	false	false	false	false	NRT traffic slot	
4	200											L	L	false	false	false	false	Guard band	

Note *1: Set the parameter as follows:
 "timeReceiver" device set "true".
 "timeTransmitter" device set "false".

Note *2: "O" indicates queue open, and a blank indicates queue close.

Note *3: In this application, Queue 0 is assigned to NRT frames, Queue 1 is assigned to PTP, and Queue 2-7 are assigned to RT frames.

And setting of Frame preemption are as follows.

- Preempt queue Queue0
- Queue enabled preempt by queue closing.....Queue0

Table 5-8 Configuration of Frame Preemption

Port	Enable	Verify dis-enable	Response dis-enable	Frag size	Verify time-out [ms]	Pre-amble check	No de-code SMD	Strict check	TX CRC in-version	RX CRC in-version
1	True	False	False	0	100	True	False	False	False	False
2	True	False	False	0	100	True	False	False	False	False
3	True	False	False	0	100	True	False	False	False	False

5.1.5.1 How to check Preemption without using a Close Preempt queue

If you want to cause preemption without using preempt by queue closing, it can be verified using the following method:

1. Set qclose of qbu_config to 0x00
2. the two-devices setup and connect external device such as a PC (for UDP frame transmission) to timeReceiver (client).
(See "Figure 5-4".)
3. From the external device, continuously send UDP packets with the following settings toward timeReceiver (client).

Table 5-9 Configurations of UDP frame from external device

Components	Configuration	Note
Size	1512 Byte	
Send rate	8000 Packet/s	
VLAN TAGs	None (Untagged)	
MAC address (destination)	11:22:33:44:55:BB	timeReceiver's MAC address
MAC address (source)	The MAC address defined as the macro "TESTPACKET_MACADDR"	See 4.1.2.11
IP address (destination)	192.168.10.170	timeTransmitter's IP address
IP address (source)	(Any address)	Do not overlap with others.

In this way, timeReceiver (client) is made to continue forwarding the above UDP frames to timeTransmitter (server).

4. In the above state, by starting time-synchronization between boards, the preemption can be caused.
 - The time-synchronization is started by disconnecting once the LAN cable connecting timeReceiver (client) and timeTransmitter (server) and then reconnecting it.
 - If preempt is not caused, reconnect LAN cable again.

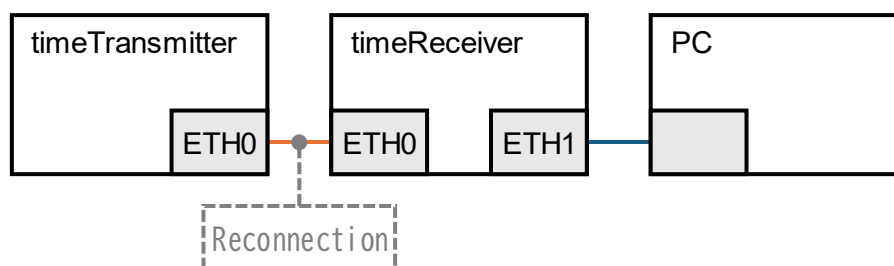


Figure 5-4 Example of Connection (Preemption without using a Close Preempt queue)

5.1.6 Operation Specifications of PSFP

When PSFP is enabled, the filtering setting is applied only to the frame reception side.

By default, Stream ID(SID) 0 of port 0 is used.

The PSFP settings for port 0, SID0 are as follows.

For each configuration item, see "4.1.2.12.1 Change configuration of filters".

Table 5-10 PSFP default configurations (port 0, SID0)

Function	Item	Setting	Description
Stream filter configuration	MAC_select	1	Frames with the source MAC address {00:11:22:33:44:BB} (*1) are subject to filtering. Note*1: This applies to the frames transmitted from the timeReceiver.
	MAC	{00:11:22:33:44:BB}	
	MAC_mask	{00:FF:FF:FF:FF:FF}	
Stream filter enabling	([0])	1	Enable SID0
Stream gate filter configuration	enable	true	Enable stream gate filter for SID0
	gateID	0	Use TDMAOUT0
	gate mode	false	Clearing the frame discard status is not necessary.
SDUmax verification configuration	enable	true	Enable SDUmax verification for SID0
	maxSDUsize	100	Data with a size of 100 bytes or more (including FCS) will be discarded(*2). Note*2: This applies to NRT frames (UDP frames).
	MSDU mode	false	Clearing the frame discard status is not necessary.
Flow metering configuration	id_enable	true	Enable flow metering for SID0
	id	0	Use meter0 for SID0
	coupling	true	Enable coupling of meter0
	mode	false	Clearing the frame discard status is not necessary.
	red_drop	true	Drop red frame
	burst_size	0x0FFF	CBS (Token max size)
	info_rate	0xA000	CIR (Token regenerate quantity)
	enable	true	Enable meter0

With the above settings, RT and NRT frames from the timeReceiver are classified as SID0 and subject to filtering.

Note: PTP frames are recognized as management frames by the ETHSW and are therefore excluded from filtering.

For details, refer to the EFP (Extended Frame Parser) chapter in the hardware manual for each board.

Table 5-11 The EFP-related chapter in the hardware manual.

Hardware Manual	Chapter
RZ/N2L Group	28.3.8 Extended Frame Parser Registers
User's Manual: Hardware	28.4.19 Extended Frame Parser (EFP)
RZ/T2N Group	29.3.8 Extended Frame Parser Registers
User's Manual: Hardware	29.4.19 Extended Frame Parser (EFP)

In the default setting, only NRT frames are discarded by the stream gate filter based on the value of TDMAOUT0 in the slot at the time of frame reception.

(TDMAOUT0 is set to L when NRT frames are received. For details, see "5.1.3 Operation Specifications of TDMA".)

RT frames pass through the stream gate filter, maximum SDU size filter, and flow meter in that order without being discarded.

Otherwise, it operates the same as gPTP sample.

See "5.1.2 Operation Specifications of gPTP".

5.1.7 Operation Specifications of CQF

In the CQF sample, TDMA is used to control the frame transmission timing.

The CQF sample uses a dedicated TDMA cycle table that differs from the usual one.

The TDMA table used in the CQF sample is as follows:

Note: The TDMA cycle table is editable.

For editing table, see "4.1.2.6 Change Table of TDMA cycle (for CQF)".

Table 5-12 TDMA cycle table (for CQF)

Slot	Period [us]	Queue-Gate (*2)(*3)								Port Mask (*2)			gpio pattern (TDMAOUTn) (*4)		Trigger (activate callback)	Preemption hold_req	In cut- through	Out cut- through	Description (*3)
		7	6	5	4	3	2	1	0	2	1	0	1	0					
0	800					0		0	0		0	0	H	L	true(*1)	false	false	false	EVEN(ID=0x101)
1	200												H	L	false	false	false	false	Guard band
2	800						0	0	0		0	0	H	H	true(*1)	false	false	false	ODD(ID=0x102)
3	200												H	H	false	false	false	false	Guard band
4	800					0		0	0		0	0	L	L	true(*1)	false	false	false	EVEN(ID=0x103)
5	200												L	L	false	false	false	false	Guard band
6	800						0	0	0		0	0	L	H	true(*1)	false	false	false	ODD(ID=0x104)
7	200												L	H	false	false	false	false	Guard band

Note *1: Set the parameter as follows:

"timeReceiver" device set "true".

"timeTransmitter" device set "false".

Note *2: "O" indicates queue open, and a blank indicates queue close.

Note *3: In this application, Queue 0 is assigned to NRT frames, Queue 1 is assigned to PTP, and Queue 2-3 are assigned to RT frames.

And the Queue 2-3 are set to CQF queue.

Note *4: In this application, the EVEN/ODD of CQF queue is switched based on TDMAOUT0.

Also, according to the Queue-Gate in the above TDMA table, while one side is queueing, the other side is open so that frames are transmitted.

Table 5-13 Operation of CQF queue

TDMAOUT0	CQF queue	Queueing	Opening (Transmitting)
Low	EVEN	Queue2	Queue3
High	ODD	Queue3	Queue2

The correspondence of the PROFINET frame IDs and TDMAOUT in each slot is as follows:

Table 5-14 Correspondence Table of Slots and Frame IDs

Slot	TDMAOUT1	TDMAOUT0	PROFINET frame ID
0	H	L	0x101
2	H	H	0x102
4	L	L	0x103
6	L	H	0x104

Each CQF cycle switches at the timing of Slots 0, 2, 4, and 6 within the TDMA cycle.

Therefore, one TDMA cycle corresponds to four CQF cycles.

When CQF is activated, the profinet frame is queued/dequeued according to the following flow:

		Slot0		Slot2	
timeReceiver	Application	0x101			
			↓Send		
	ETHSW	Queue3	Queue2 0x101	Queue3	Queue2 0x101
	Queue Gate		Close	Close	↓Send
timeTransmitter	ETHSW	Queue2	Queue3	Queue2	Queue3 0x101
					↓Send
	Application			0x101	

Figure 5-5 Queueing flow of CQF (two-devices setup)

Also, when operating with three-devices setup like the "4.4 Example of Operation Check for Sample Application (three-device setup)", the queueing flow is as follows:

		Slot0		Slot2		Slot4	
timeReceiver2	Application	0x101					
			↓Send				
	ETHSW	Queue3	Queue2 0x101	Queue3	Queue2 0x101	Queue3	Queue2
	Queue Gate		Close	Close	↓Send		Close
timeReceiver1	ETHSW	Queue2	Queue3	Queue2	Queue3 0x101	Queue2	Queue3 0x101
	Queue Gate	Close			Close	Close	↓Send
timeTransmitter	ETHSW	Queue3	Queue2	Queue3	Queue2	Queue3	Queue2 0x101
							↓Send
	Application					0x101	

Figure 5-6 Queueing flow of CQF (three-devices setup)

As described above, since the queueing/dequeueing(sending) of frames is alternately switched, the reception timing increases (shifts) by one cycle of CQF cycle according to the number of hops.

Abort actual operation, see "5.2.2.3 GPIO Pin's outputs of CQF".

5.1.8 Operation Specifications of LLDP

In the LLDP sample, device send LLDP frames periodically.

This sample also supports MIB(SNMP), and each items can be viewed using the console.

Otherwise, it operates the same as gPTP sample.

See "5.1.2 Operation Specifications of gPTP".

5.2 Sample application's outputs for checking operation

The contents that can be confirmed through various outputs are shown below.

5.2.1 Outputs of log message

This Application outputs log messages through UART from Serial Port of each board.

For the connector layout on the board, see "5.1.1 Sample menu".

The log message shows the following content:

- State of gPTP
 - State of PROFINET traffic (only when lost frames occur)
 - State of Frame Preemption
 - State of PSFP
 - MIB informations (LLDP)
- Note: CQF sample does not output any information to the log.

5.2.1.1 Log messages of gPTP

Log messages of gPTP are following:

5.2.1.1.1 Log output at connection (log emitted when a connection is established)

When a device is connected, the following logs are output.

The content of the output varies depending on the device assignment (timeTransmitter/timeReceiver).

- "timeTransmitter" device

```
NETDEV_UP. port=[0]
```

- "timeReceiver" device

```
NETDEV_UP. port=[0]  
CLOCK_SYNCED
```

Each log messages shows the following states.

- `NETDEV_UP. port=[0]`
 - Port0 is Link-Upped.
- `CLOCK_SYNCED`
 - Stable synchronization with timeTransmitter clock".

5.2.1.1.2 Periodic log output (regularly occurring/logs emitted at set intervals)

Various gPTP information is logged periodically at 5-second intervals.

This allows the operational status of gPTP for each device to be checked.

The contents of the output vary depending on the device state (disconnected / operating as timeTransmitter / operating as timeReceiver).

- Not connecting

```

-----NO SYNC ROLE-----
instance number 0 with domain number 0x0
CLOCK_IDENTITY 0x 00 11 22 33 44 aa 00 00
AS_CAPABLE P1 0 | P2 0 | P3 0 |
PORT_STATE P1 3 | P2 3 | P3 3 |

-----
MEAN_LINK_DELAY P1 0 | P2 0 | P3 0 |
time 4 999881808
=====

```

- Operating as timeTransmitter

```

-----MASTER-----
instance number 0 with domain number 0x0
CLOCK_IDENTITY 0x 00 11 22 33 44 aa 00 00
AS_CAPABLE P1 1 | P2 0 | P3 0 |
PORT_STATE P1 6 | P2 3 | P3 3 |

-----
MEAN_LINK_DELAY P1 13 | P2 0 | P3 0 |
time 14 999880608
=====

```

- Operating as timeReceiver

```

-----SLAVE-----
GM_CLOCK_IDENTITY 0x 00 11 22 33 44 aa 00 00
STEPS_REMOVED 1
OFFSET_FROM_MASTER -53
instance number 0 with domain number 0x0
CLOCK_IDENTITY 0x 00 11 22 33 44 bb 00 00
AS_CAPABLE P1 1 | P2 0 | P3 0 |
PORT_STATE P1 9 | P2 3 | P3 3 |

-----
MEAN_LINK_DELAY P1 15 | P2 0 | P3 0 |
time 14 996934159
=====

```

The meaning of each item is as follows:

Table 5-15 Output items in PTP cyclic log

Field	Description	Note
(first row)	Current assignment of the device	NO_SYNC_ROLE Not connected MASTER Operating as Master(timeTransmitter) SLAVE Operating as Slave (timeReceiver)
GM_CLOCK_IDENTITY (*1)	GrandMaster's CLOCK_IDENTITY	
STEP_REMOVED (*1)	Number of steps from the GrandMaster	Increases according to the number of relay devices from the GrandMaster. (Becomes 1 when directly connected.)
OFFSET_FROM_MASTER (*1)	Offset value relative to the GrandMaster's time	
instance number...	The number of the running instance and the domain number it belongs to	
CLOCK_IDENTITY	Own CLOCK_IDENTITY	In this application, CLOCK_IDENTITY is automatically generated based on the MAC address.
AS_CAPABLE	Whether the peer device on the port is AS_CAPABLE	1 : Enabled 0 : Disabled
PORT_STATE	States of each port	3 : PTP is not operating 6 : Sending time information 9 : Receiving time information
MEAN_LINK_DELAY	Propagation delay value calculated per port	
time	Value of the PTP timer	Timer adjusted by time synchronization

Note *1: Displayed only when operating as a *timeReceiver*.

In addition to the above, additional log output is generated depending on the sample launched.

For details on each content, see each chapter.

5.2.1.2 Log messages of PROFINET traffic

Log messages of PROFINET traffics are following:

Note: This message is output only when lost frames is detected(*1).

```
profinet(0101):lost frame. [0x41e0,0xef80]
```

Note *1: Lost frame detection is performed at the timing of PROFINET frame reception.
When a new PROFINET frame is received, a consistency check is performed against the CycleCounter value of the previously received PROFINET frame (which is incremented by a fixed value for each frame transmission), and based on the result, the occurrence of lost frames is detected.
Therefore, a lost frame is actually detected at the timing of the PROFINET frame reception immediately after the lost frame occurred.

5.2.1.3 Log messages of Frame Preemption

Log messages of Frame preemption traffics are as follows:

port	[0]	[1]	[2]	[3]	[4]	[5]	[6]	[7]
<0>	0x3	0	0	0x00	0x04	0x00000000	0x00000012	0x0000
[8]	[9]	[A]	[B]	[C]	[D]	[E]	[F]	[G]
0x0000	0x00000012	0x00000000	0x01	0x00	0x01	0x00	0x01	0x03

The meaning of each item is as follows:

Table 5-16 Log messages of Frame Preemption

Number	Name	Description
port	port	Port number
0	verify_state	Result of verify process. The values are as follows: 0x1: Not active Frame preemption 0x3: Success 0x4: Failed 0x5: Not active verify process
1	preempt_state	Indicates whether the port is preempted.
2	hold_req_state	Indicates whether Hold request is active.
3	min_pfrm_adj	Minimum mPacket size
4	last_pfrm_adj	Adjustment Value of mPacket size
5	TxFragCount	Count of sent mPacket
6	RxFragCount	Count of received mPacket
7	RxSmdErrCount	Count of frames received with SMD-Cx but assembly not in progress
8	RxAssyErrCount	Count of preemption frame assembly failures
9	RxAssyOkCount	Count of preemption frame assembly success
A	TxHoldReqCount	Count of Hold request changed from FALSE to TRUE
B	RxResponseOkCount	Count of received response frame
C	RxResponseBadCount	Count of received error response frame
D	RxVerifyOkCount	Count of received verify frame
E	RxVerifyBadCount	Count of received error verify frame
F	TxResponseOkCount	Count of sent response frame
G	TxVerifyOkCount	Count of sent verify frame

When the frame is operating correctly, the following behavior can be checked via the logs of each device:

Table 5-17 Operation confirmations of Frame Preemption

Operation	Device	Log content	Detailed
Success of Verify process	timeReceiver (client)	[G] incremented	The device sent the verify frame
		[B] incremented	The device received the response frame from timeTransmitter (server) to the verify frame sent itself.
		[0] is 0x3	The status of verify process shows "Success".
	timeTransmitter (server)	[D] incremented	The device received the verify frame from timeReceiver (client)
		[F] incremented	The device sent the response frame to the verify frame from timeReceiver (client)
Preemption	timeReceiver (client)	[5] incremented	The frames are divided by causing of frame preemption.
	timeTransmitter (server)	[6] incremented	The device received the frames fragmented by frame preemption.
		[9] incremented	The reassembly of the received fragmented frames is successful.

5.2.1.4 Log messages of PSFP

Log messages of PSFP are following:

Qci Stream Match Packet Count	[port:0 > sid:0]	: 25000
Qci MSDU Passed Packet Count	[port:0 > sid:0]	: 5000
Qci MSDU Reject Packet Count	[port:0 > sid:0]	: 0
Qci Gate g Passed Packet Count	[port:0 > gaid:0]	: 5000
Qci Gate g Dropped Packet Count	[port:0 > gaid:0]	: 20000
Qci Meter Green Packet Count	[port:0 > meid:0]	: 5000
Qci Meter Red Packet Count	[port:0 > meid:0]	: 0
VLAN Ingress Check Drop Frame Counter	[port:0]	: 0
DST Address Lookup Hit Counter	[port:0]	: 0
Error Interrupt - StatusQci Stream Filter	[port:0]	: 0x00
Error Interrupt - StatusQci Gate	[port:0]	: 0x00
Error Interrupt - StatusQci Meter	[port:0]	: 0x00
Frame Parser Runtime Error Status	[port:0]	: 0x00

The meaning of each item is as follows:

Table 5-18 Log messages of PSFP

Item	Description
Qci Stream Match Packet Count	Count of frames that matched the configured stream filter
Qci MSDU Passed Packet Count	Count of frames that passed the maxSDU size filter
Qci MSDU Reject Packet Count	Count of frames that dropped by the maxSDU size filter
Qci Gate g Passed Packet Count	Count of frames that passed the stream gate filter
Qci Gate g Dropped Packet Count	Count of frames that dropped by the stream gate filter
Qci Meter Green Packet Count	Count of frames that passed the flow meter
Qci Meter Red Packet Count	Count of frames that dropped by the flow meter
VLAN Ingress Check Drop Frame Counter	Count of frames that dropped by the VLAN Ingress verification
DST Address Lookup Hit Counter	Count of frames that hit by the active stream identification
Error Interrupt - StatusQci Stream Filter	Error interrupt status of stream filter
Error Interrupt - StatusQci Gate	Error interrupt status of stream gate filter
Error Interrupt - StatusQci Meter	Error interrupt status of flow meter
Frame Parser Runtime Error Status	Error interrupt status of frame parser runtime error

5.2.1.5 MIB informations (LLDP)

When LLDP sample is launched, the following menu is displayed:

```
[main menu]
[0]: Info gPTP
[1]: MIB console
>>
```

When [0] is selected from this menu, the following display is output, followed by the gPTP log output.

```
[main menu]
[q]: quit
>>
```

Entering [q] returns to the menu at LLDP startup.

When [1] is selected from this menu, the following display is output, and the console starts.

```
Started MIB Console.
>>
```

Note: In the console, press Ctrl + Enter ("CRLF") to complete the input of each command.

In MIB console, entering the command "snmp getnet 1" outputs all MIB information of the target device.

Entering the command "q" returns to the menu at LLDP startup.

5.2.2 Outputs of GPIO Pin

This Application outputs clock pulse wave (for checking time synchronization) and information of TSN's states from GPIO Pins.

5.2.2.1 GPIO Pin's outputs of gPTP

In this application, for checking of time-synchronization by gPTP, a clock waveform is output to PTPOUT.

This signal is clock pulse generated by the internal clock of the device.

By time-synchronizing this internal timer by gPTP, the phase of the signals for each device is adjusted so that they match.

By this, the accuracy of time-synchronization can be measured based on these signals.

Note: The period of the clock waveform can be selected from "1[s]" (default setting) and "1[ms]".
To change it, refer to "4.1.2.9 Change configuration of the pulse generator period".

The assignment and placement of the various pins on each board are shown below.

Table 5-19 Assignment of PTPOUT

Target Borad	Assignment
RZ/T2M-RSK	CN2-8
RZ/N2L-RSK	CN1-11
RZ/T2N-Evb	J55-8

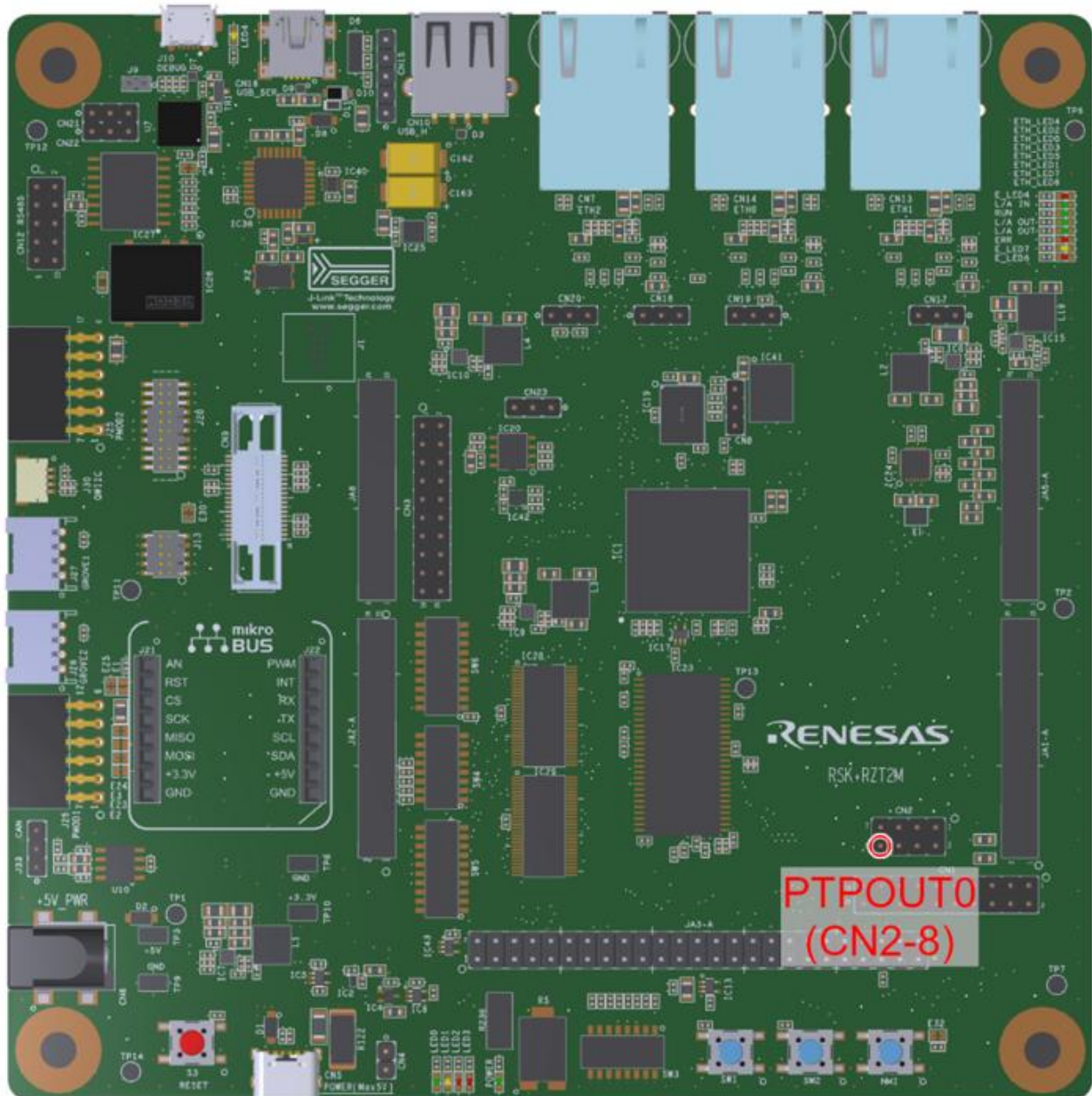


Figure 5-7 RZ/T2M-RSK board layout (gPTP Pins)

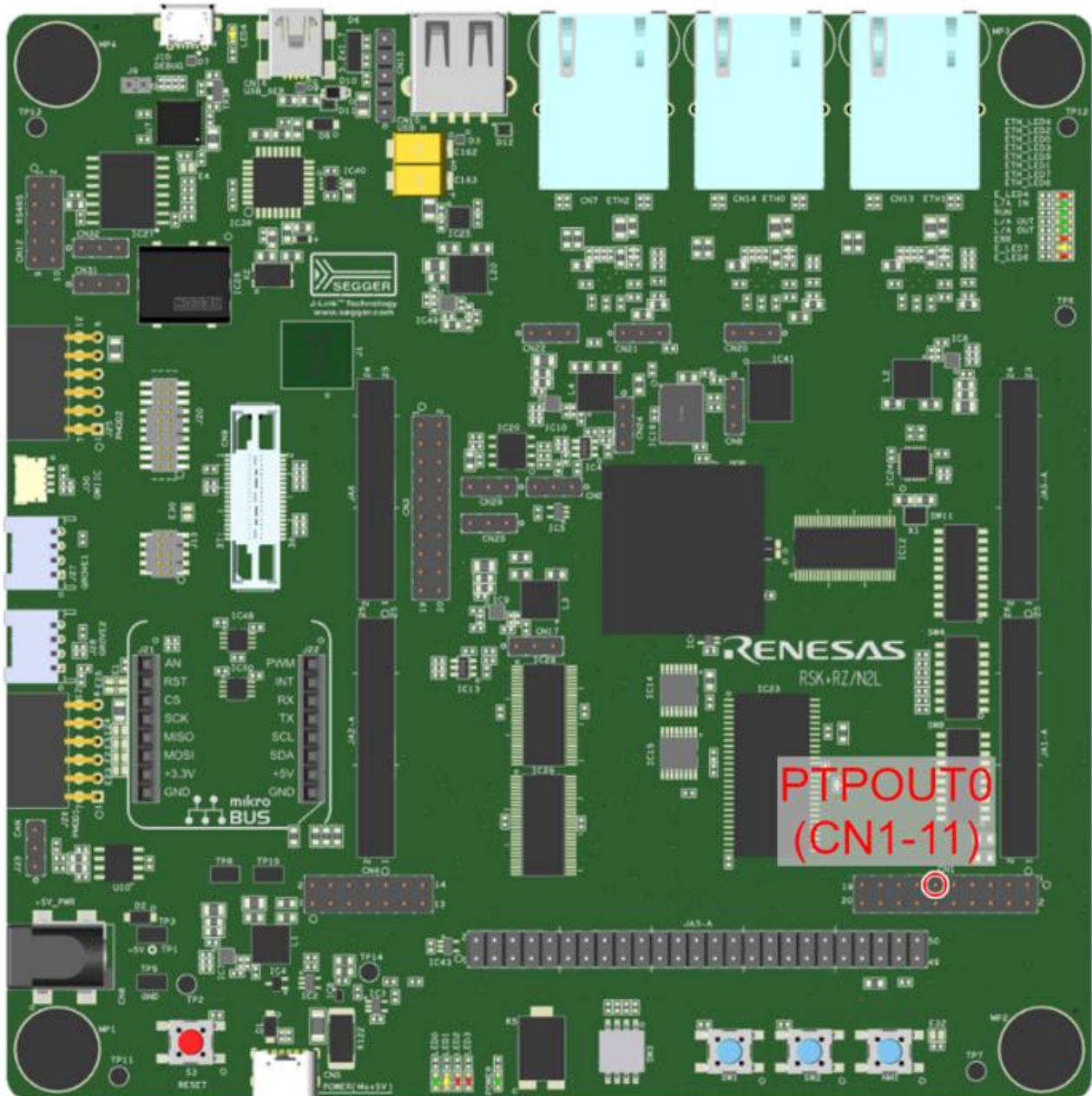


Figure 5-8 RZ/N2L-RSK board layout (gPTP Pins)

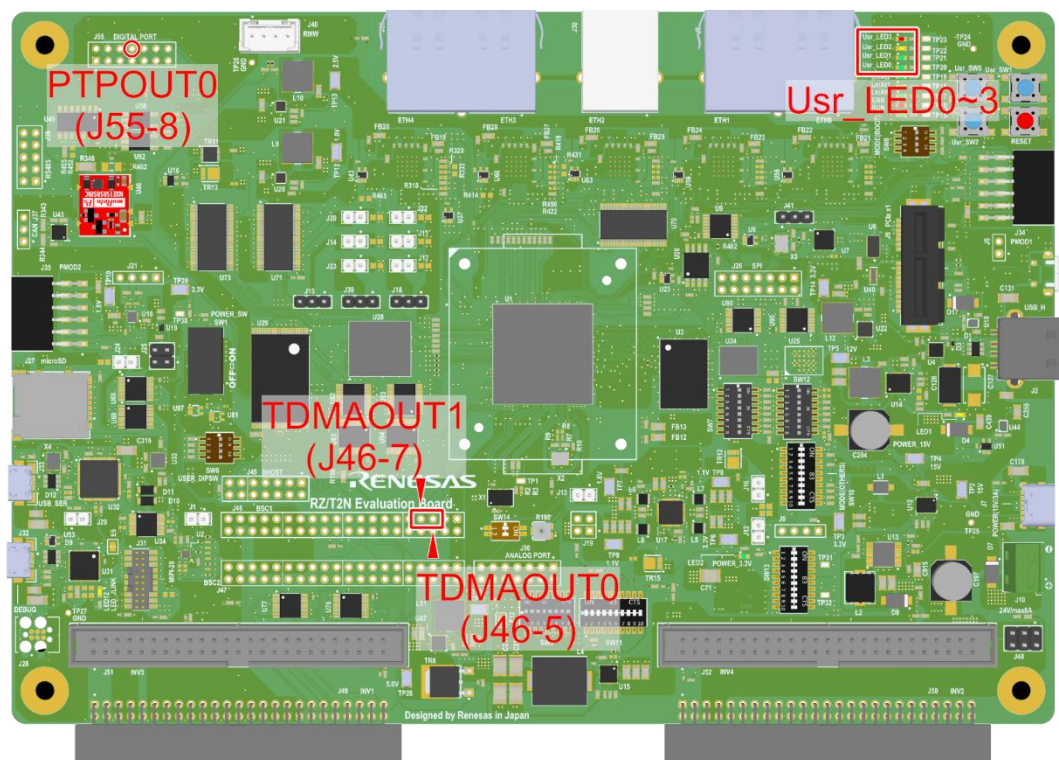


Figure 5-9 RZ/T2N-Evb board layout (gPTP Pins)

Examples of each pin's operation when gPTP is active are shown below.

EDUX1052A, CN63430212: Tue Jun 30 17:36:00 2026

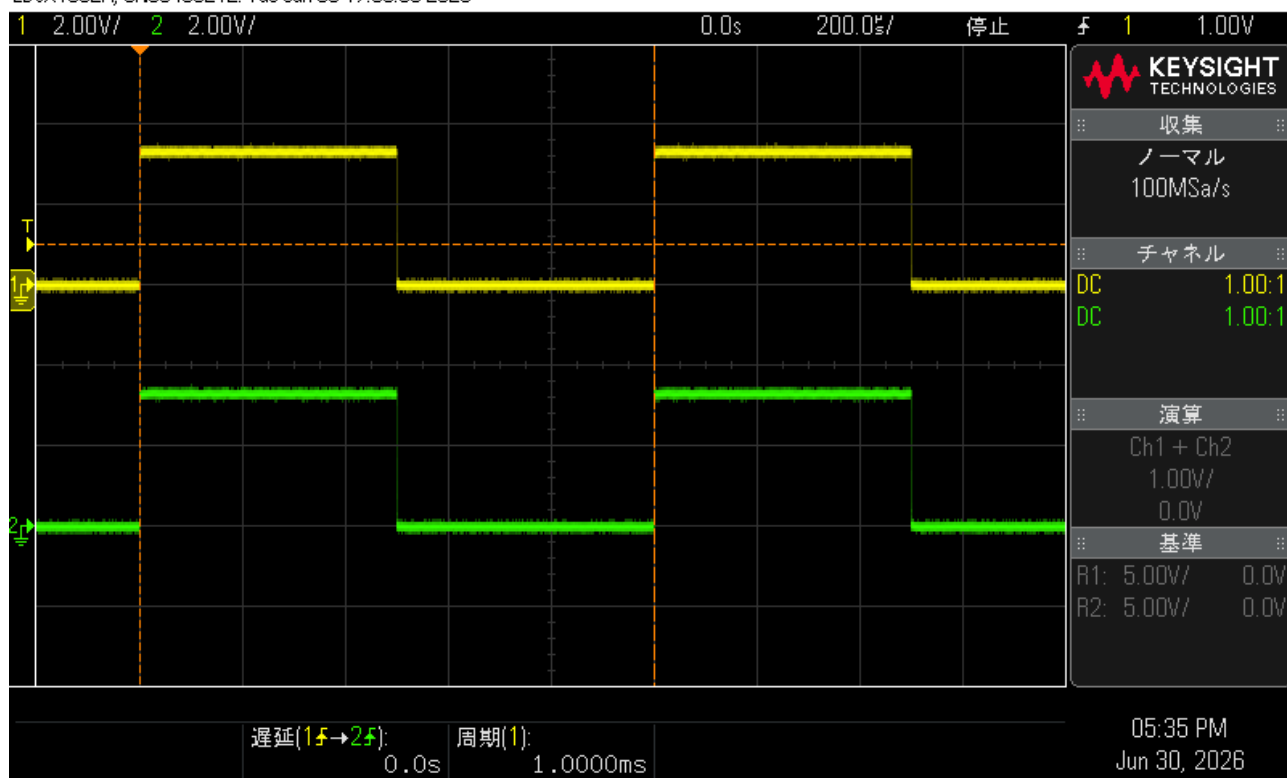


Figure 5-10 PTPOUT during gPTP operation

Note: In the above figure, the image is for when using RZ/T2M.

Note: In the above figure, the period of PTPOUT0 is set to 1[ms].

In the above figure, the description of each channel is the following:

Table 5-20 Description of each channel

channel	color	Output pin	description
CH1	Yellow	PTPOUT0 (timeTransmitter)	Output waveform according to current clock.
CH2	Green	PTPOUT0 (timeTransmitter)	(Same as above)

5.2.2.2 GPIO Pin's outputs of TDMA

Depending on the current state of TSN, output of TDMA-Pins and LED-Pins are Switched.

Therefore, the current states of TSN can be checked by using an oscilloscope.

The descriptions of each pin's outputs are the following:

Table 5-21 TDMA output pins

Pin	description	detailed
TDMAOUT0	Switches according to current active TDMA slot	Set "1" when Slot0.
TDMAOUT1	(Same as above)	Set "1" when Slot0 & Slot2.
LED0,1,2,3 (*1)	Toggles at timeTransmitter device has received PROFINET frame	Toggles the LED on the board according to the ID of received frame

Note *1: TimeTransmitter device only.

The assignment and placement of the various pins on each board are shown below.

Table 5-22 Assignment of pins for TDMA operation verification

Target Board	TDMAOUT0	TDMAOUT1	LED0	LED1	LED2	LED3
RZ/T2M-RSK	CN1-4	ETH_LED0	LED0	LED1	LED2	LED3
RZ/N2L-RSK						
RZ/T2N-Evb	J46-5	J46-7	Usr_LED0 (LED8)	Usr_LED1 (LED9)	Usr_LED2 (LED10)	Usr_LED3 (LED11)

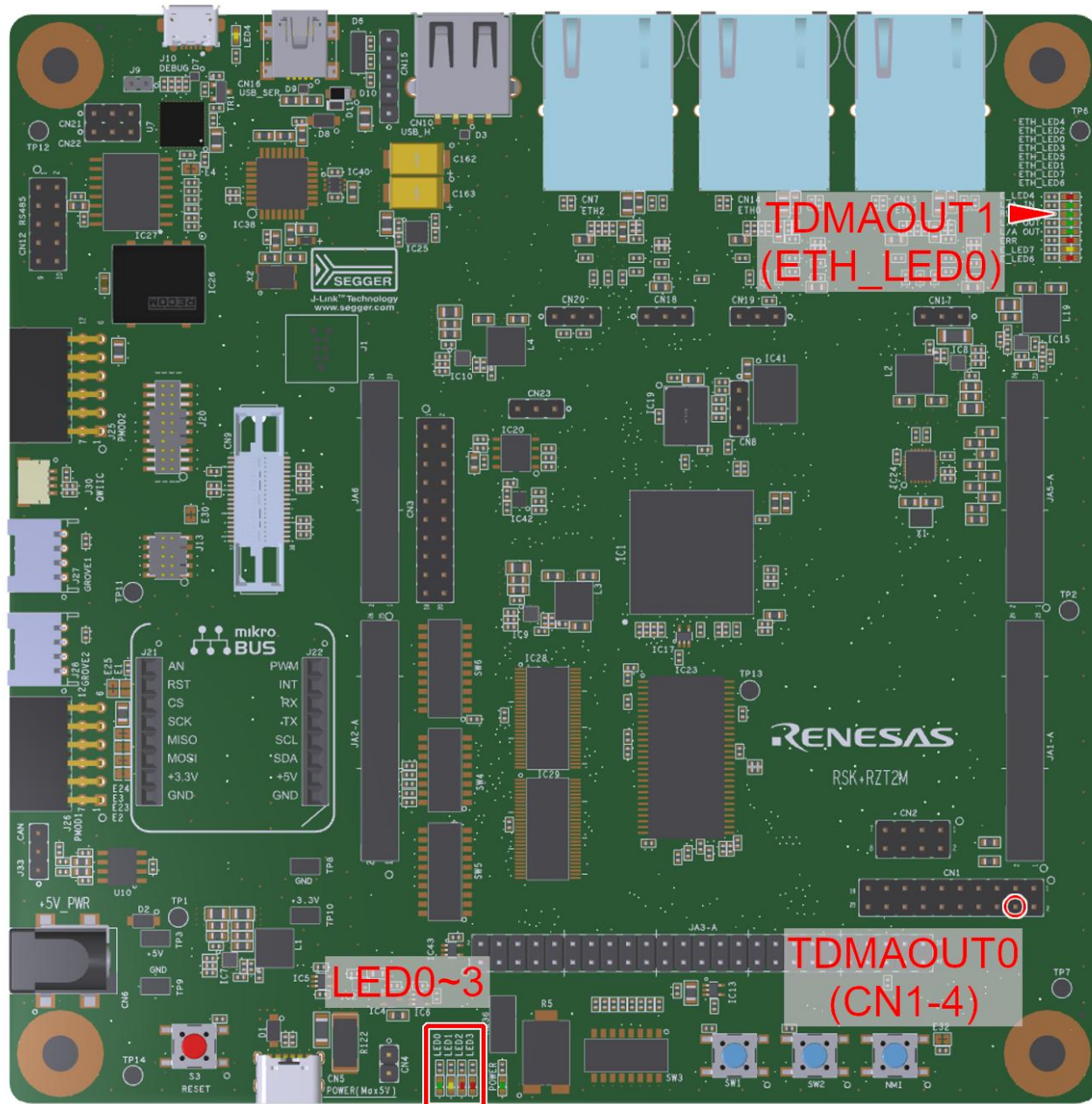


Figure 5-11 RZ/T2M-RSK board layout (TDMA Pins)

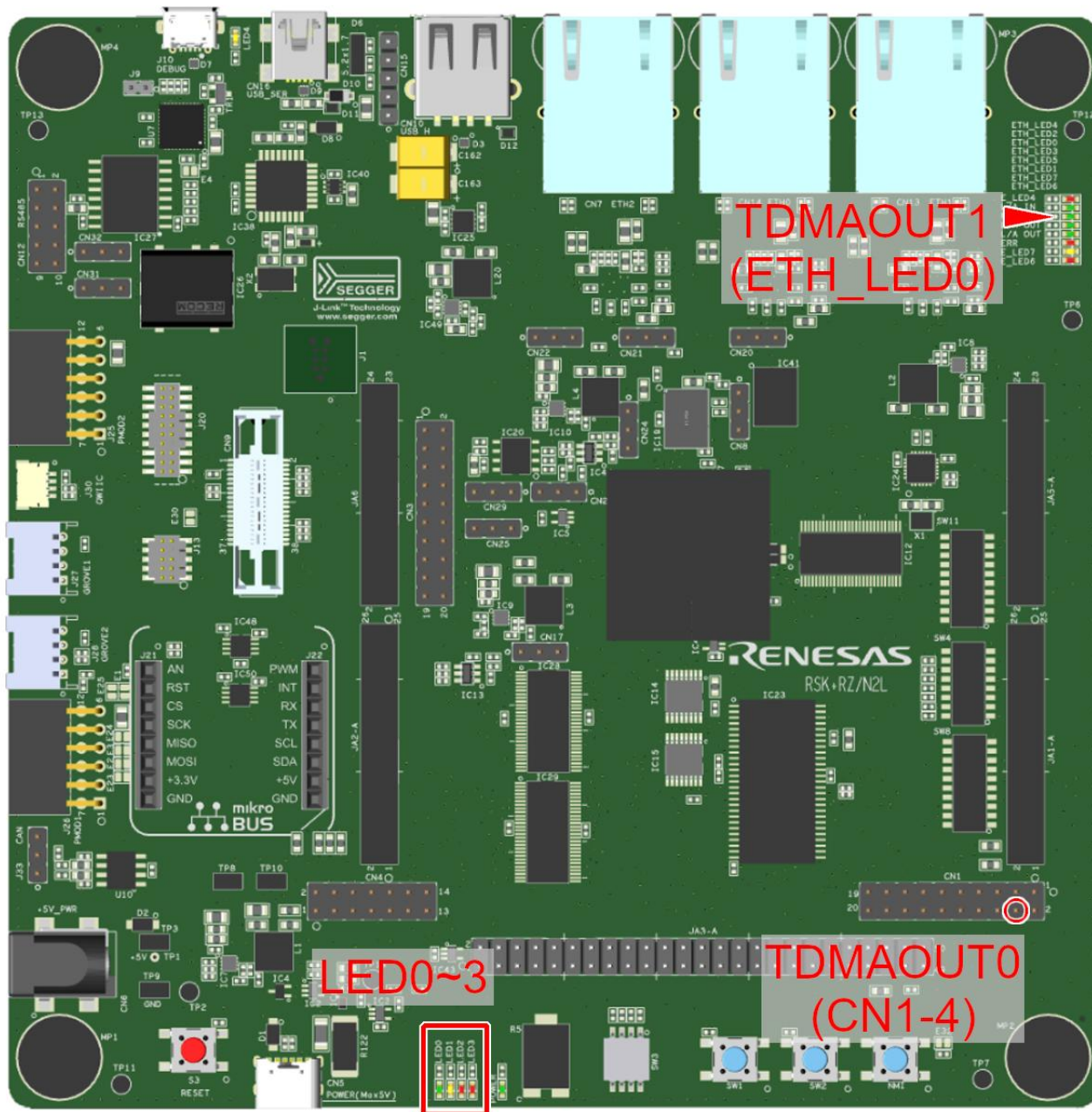


Figure 5-12 RZ/N2L-RSK board layout (TDMA Pins)

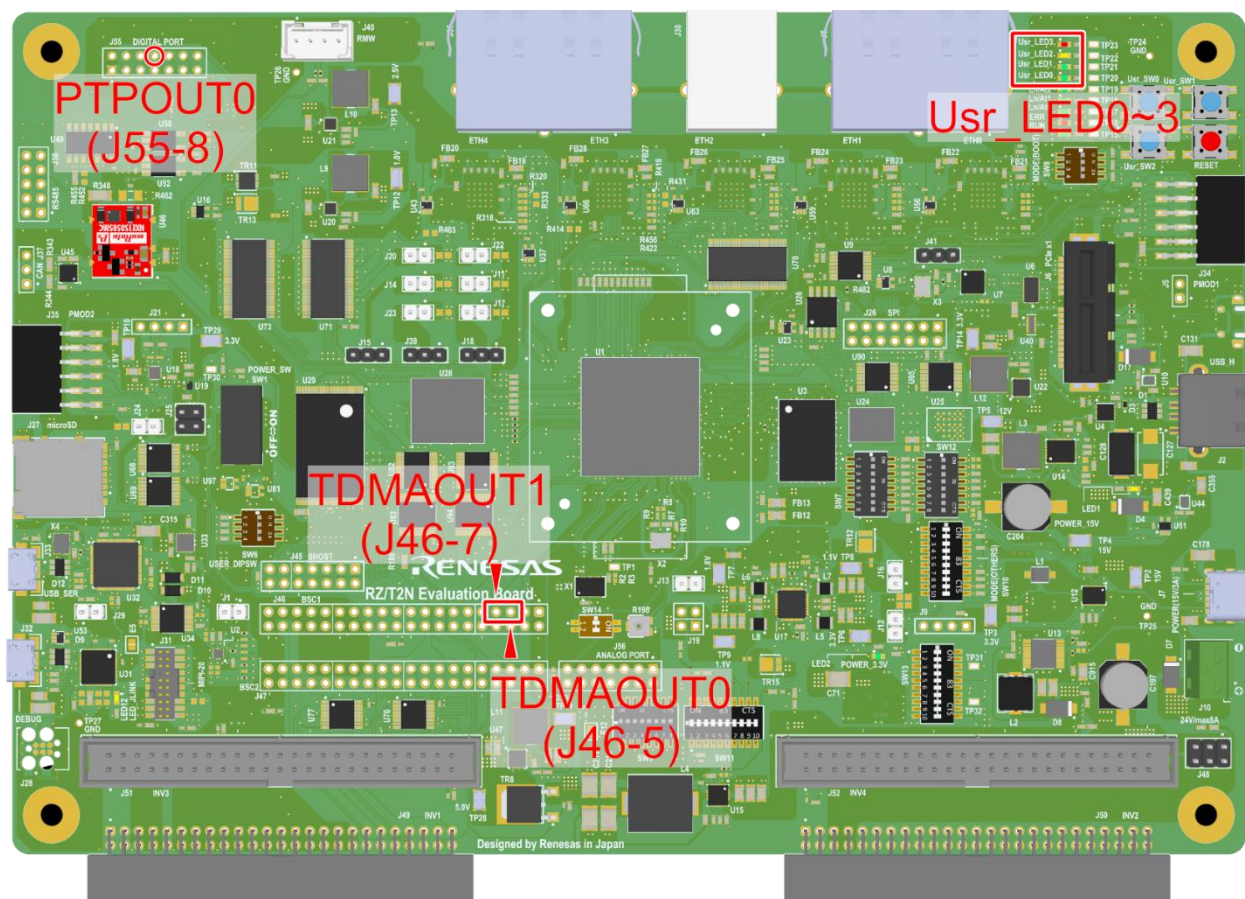


Figure 5-13 RZ/T2N-Evb board layout (TDMA Pins)

Examples of each pin's operation when TSN is active are shown below.

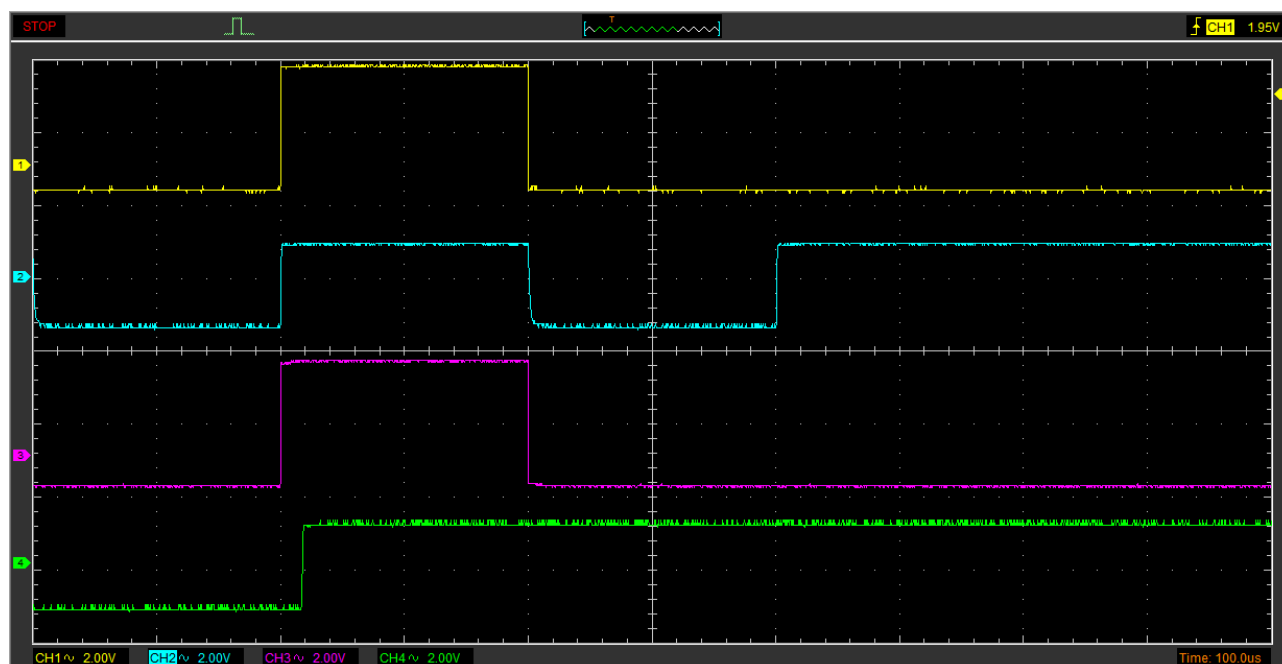


Figure 5-14 Example of pin's operation in TDMA

In the above figure, the description of each channel is the following:

Table 5-23 Description of each channel

channel	color	Pin	description
CH1	Yellow	TDMAOUT0 (timeTransmitter)	Switches according to current active TDMA slot
CH2	Cyan	TDMAOUT1 (timeTransmitter)	(Same as above)
CH3	Magenta	TDMAOUT0 (timeReceiver)	(Same as above)
CH4	Green	LED0 (timeTransmitter)	Toggles at timeTransmitter device has received PROFINET frame

Each Pin in the above figure shows the following:

- CH1(timeTransmitter's TDMAOUT0) and CH2(timeTransmitter's TDMAOUT1) show the timing of each TDMA slot.
- CH1(timeTransmitter's TDMAOUT0) and CH3(timeReceiver's TDMAOUT1) show the state of time synchronization.
- CH1(timeTransmitter's TDMAOUT0) and CH4(timeTransmitter's LED0) shows that timeTransmitter is receiving PROFINET frames at the proper timing.

The kind of sending and receiving packet and sending timing, see “5.1 Sample application Overview”.

5.2.2.3 GPIO Pin's outputs of CQF

Depending on the current state of CQF, output of TDMA-Pins and LED-Pins are Switched.

Therefore, the current states of CQF can be checked by using an oscilloscope.

The descriptions of each pin's outputs are the following:

Table 5-24 TDMA output pins (for CQF sample)

Pin	description	detailed
TDMAOUT0	Switches according to current active TDMA slot	Set "1" when CQF cycle is ODD. (Set "1" when Slot2, 3, 6, 7.)
TDMAOUT1	(Same as above)	Set "1" when Slot0, 1, 2, 3.
LED0,1,2,3 (*1)	Toggles at timeTransmitter device has received PROFINET frame	Toggles the LED on the board according to the ID of received frame

Note *1: TimeTransmitter device only.

For the assignment of the various pins on each board, refer to "Table 5-22 Assignment of pins for TDMA operation verification".

Examples of each pin's operation when CQF in two-devices setup is active are shown below:

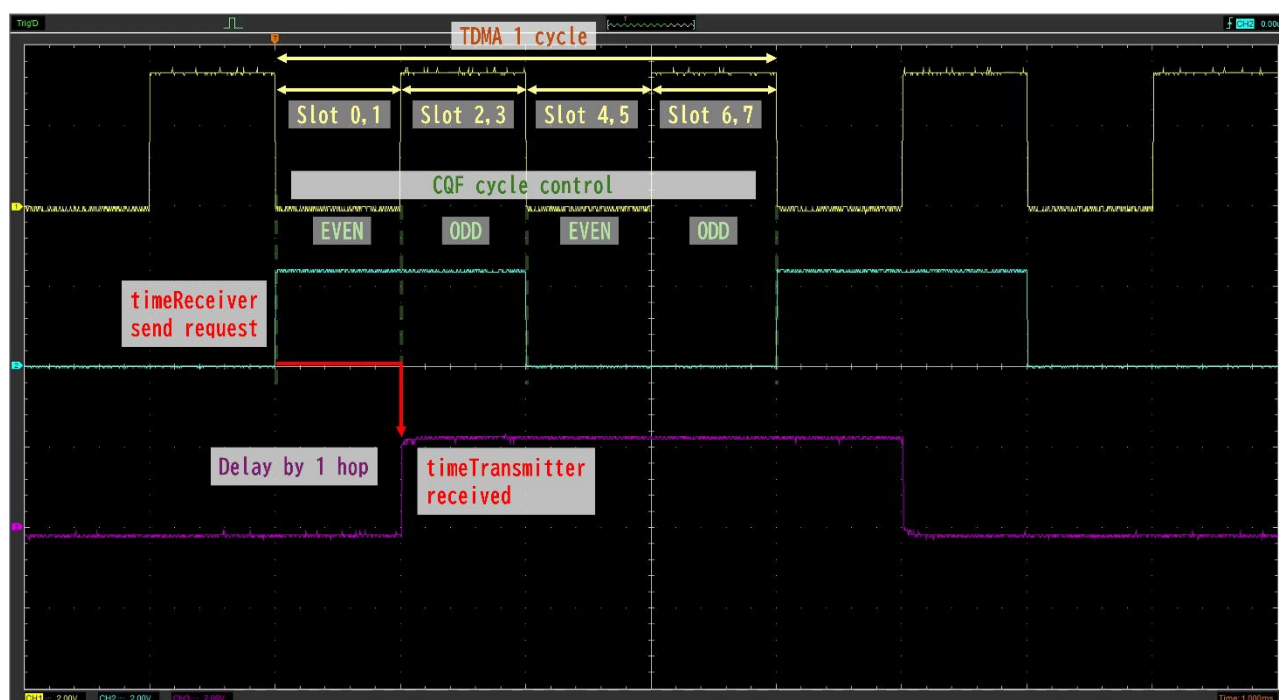


Figure 5-15 Example of pin's operation in CQF (two-devices setup)

In the above figure, the description of each channel is the following:

Table 5-25 Description of each channel

channel	color	Pin	description
CH1	Yellow	TDMAOUT0 (timeReceiver)	Switches according to current active TDMA slot
CH2	Cyan	TDMAOUT1 (timeReceiver)	(Same as above)
CH3	Magenta	LED0 (timeTransmitter)	Toggles at timeTransmitter device has received PROFINET frame

Each Pin in the above figure shows the following:

- It can be confirmed from CH1(timeReceiver's TDMAOUT0), CH2(timeReceiver's TDMAOUT1), and CH3(timeTransmitter's LED0) that a PROFINET frame with frame ID 0x101 is being received simultaneously with the timing when a PROFINET frame with frame ID 0x102 is transmitted. (Due to CQF, the reception timing is delayed by one cycle for one hop.)

Examples of each pin's operation when CQF in three-devices setup like the "4.4 Example of Operation Check for Sample Application (three-device setup)" is active are shown below:

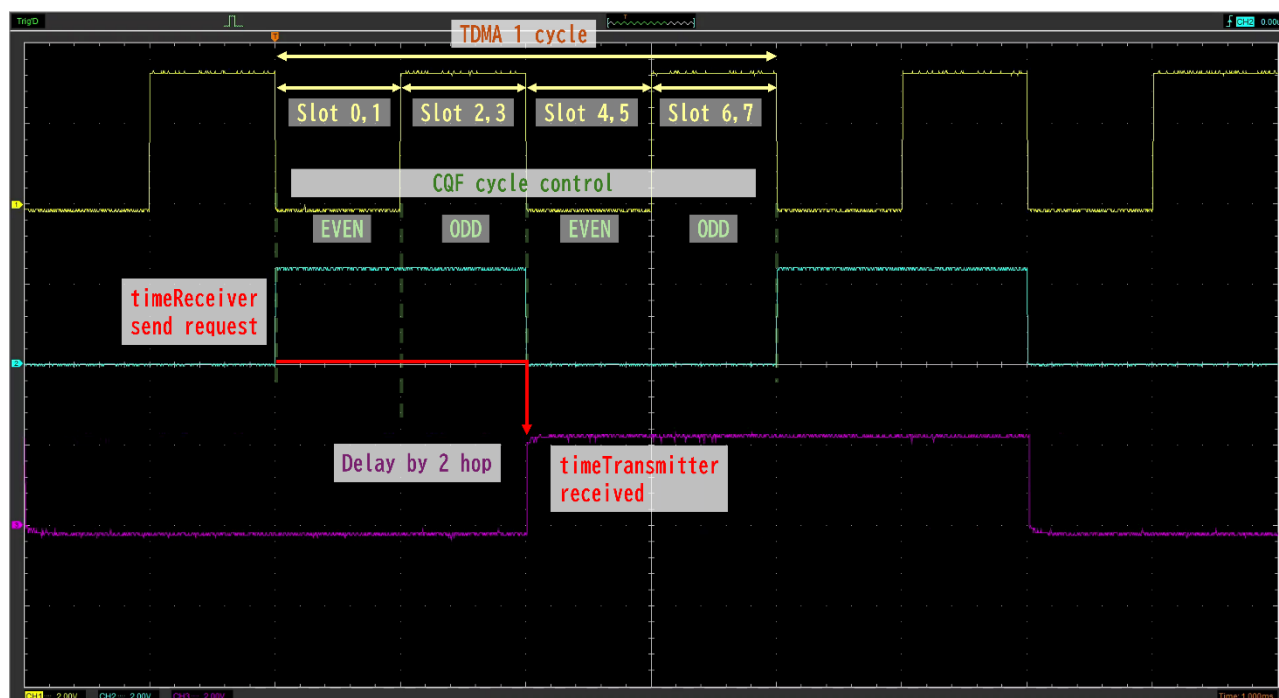


Figure 5-16 Example of pin's operation in CQF (three-devices setup)

In the above figure, the description of each channel is the following:

Table 5-26 Description of each channel

channel	color	Pin	description
CH1	Yellow	TDMAOUT0 (timeReceiver2)	Switches according to current active TDMA slot
CH2	Cyan	TDMAOUT1 (timeReceiver2)	(Same as above)
CH3	Magenta	LED0 (timeTransmitter)	Toggles at timeTransmitter device has received PROFINET frame

Each Pin in the above figure shows the following:

- It can be confirmed from CH1(timeReceiver2's TDMAOUT0), CH2(timeReceiver2's TDMAOUT1), and CH3(timeTransmitter's LED0) that a PROFINET frame with frame ID 0x101 is being received simultaneously with the timing when a PROFINET frame with frame ID 0x103 is transmitted. (Due to CQF, the reception timing is delayed by two interval for two hop.)

5.3 Application structure

5.3.1 Module structure

5.3.1.1 Overall Description

In this sample, there is a "NETIF" module between the "TCPIP" module and the "GMAC" module.

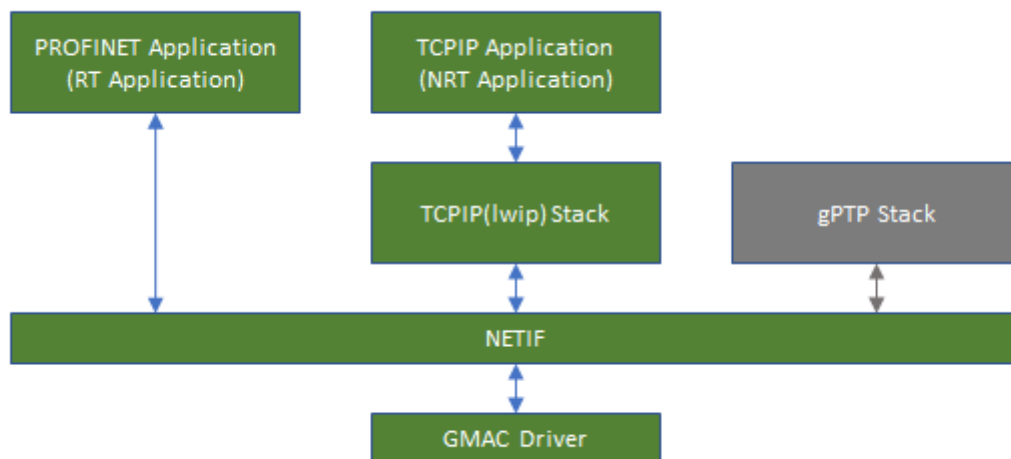


Figure 5-17 Software module structure

5.3.1.1.1 Notes - Differences depending on development environment

In the case of e2studio, it is the "NETWORK" module instead of the "NETIF" module.

"NETIF" module and "NETWORK" module have different structures.

5.3.1.2 Detailed explanation of Modular structure

5.3.1.2.1 For "Rx frames"

The "Ethernet Reader Task" of the "NETIF" module receives the frames.

After receiving some frame, the "Ethernet Reader Task" queues it to each RxQueue.

The frames in each RxQueue are got by the application(protocol) corresponding to that RxQueue.

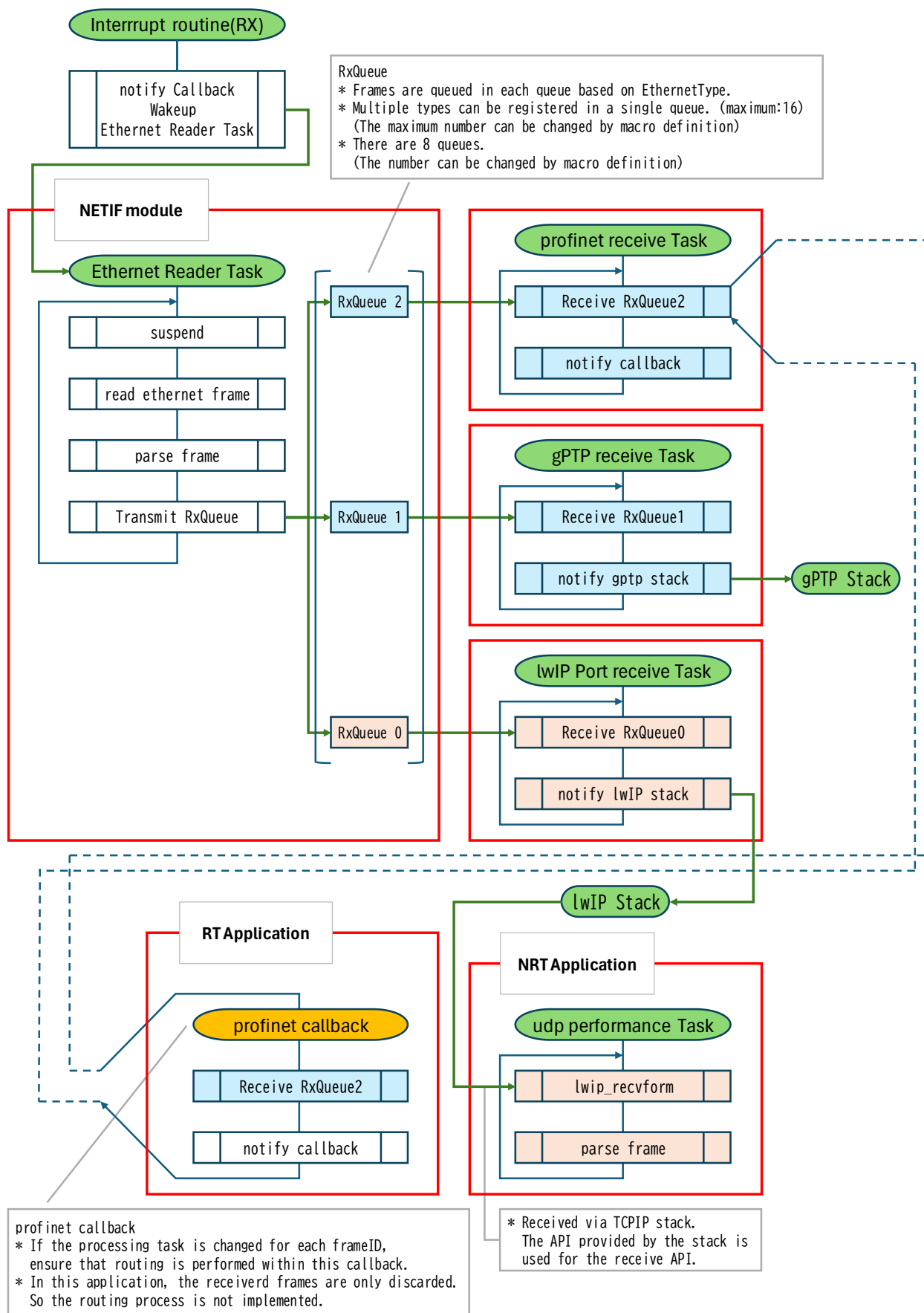


Figure 5-18 Software module flow (Rx frames)

5.3.1.2.2 For "Tx frames"

The "Ethernet Writer Task" of the "NETIF" module sends the frames.

Frame sending by the "Ethernet writer task" prioritizes frames from higher priority queues.

In queueing frames into each TxQueue, the "QueueID" is referenced.

The "QueueID" is in the management tag of the Tx frame.

Each application (protocol) does not need to be aware of TxQueue when sending frames.

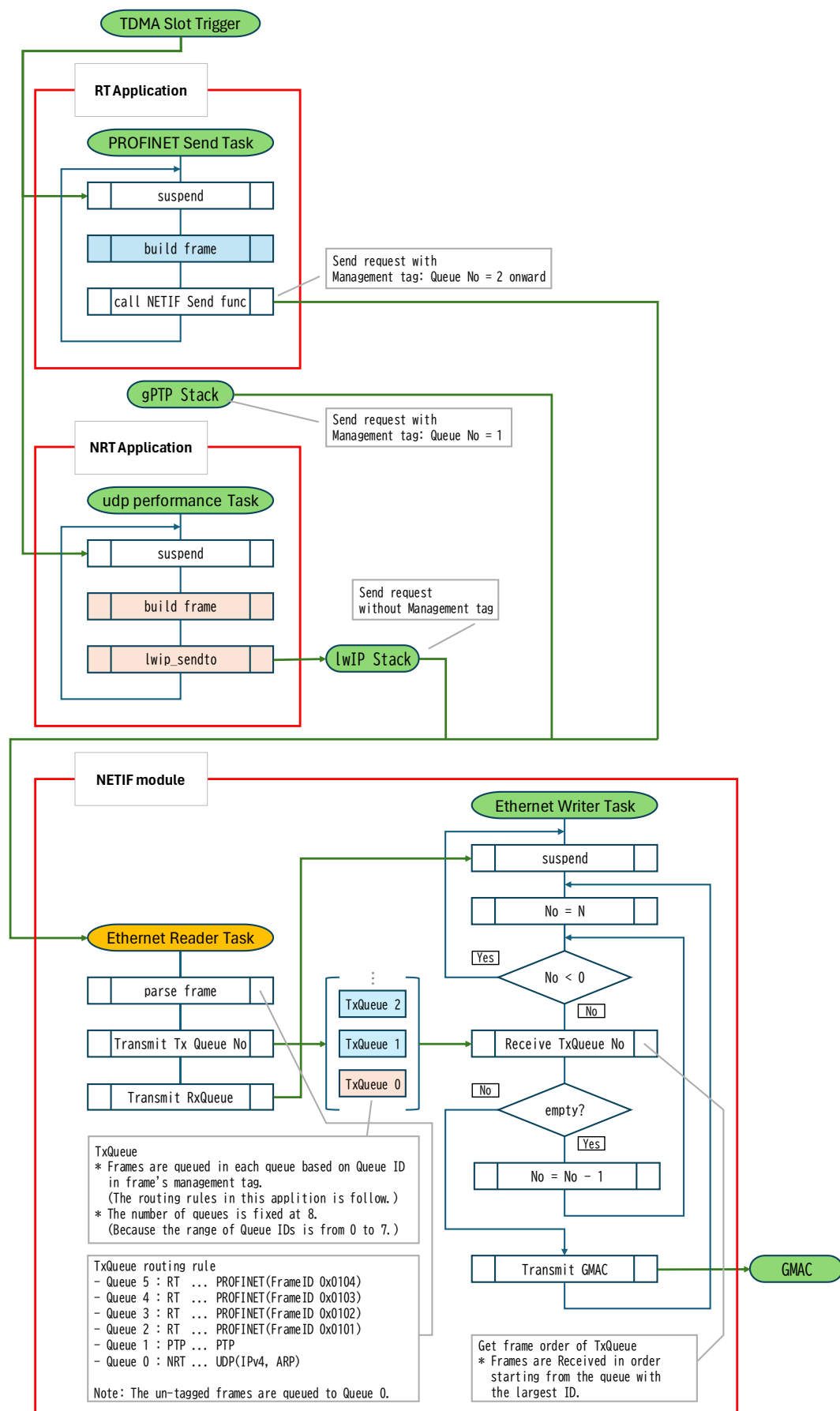


Figure 5-19 Software module flow (Tx frames)

5.3.2 Note - Task priorities in sample application

Task prioritization must be in the following order:

Table 5-27 Task Priorities

Priority	Task	Note
Highest	- LLDP task - Serial input task (for menu controls)	
	NETIF sender task	Set the receiver task and NETIF event issuing task to a lower priority than these (see below).
	RT transfer tasks	Needs the processing time of gPTP lower than the others (see below).
	gPTP tasks	Set the gPTP event issuing task to a lower priority than these (see below).
	NETIF receiver task	
	- LINKED event issuing task (NETIF event issuing task) "Ethernet Monitor Task" - SYNCED event issuing task (gPTP event issuing task) "EventHandler"	
	NRT transfer task	
Lowest	Serial output task (for log outputs)	

Note: When changing task priorities, please configure them in accordance with the above ranking. Operation is not guaranteed if the above conditions are not met.

5.3.2.1 Note - Priority setting in *configuration.xml* due to change in task priority

Basic tasks in the RTOS, such as the timer task and main task, have various settings including priority configured in *configuration.xml*.

If changing task priorities as mentioned above, it is necessary to consider the priorities of these tasks as well.

The task priority settings in *configuration.xml* shall be as follows:

Table 5-28 Priority settings in *configuration.xml*

Stacks	Priority	Default
Main Thread Priority ("main_thread")	Highest than other tasks	12
Timer Task Priority ("Tmr Svc")	Highest than other tasks but lower than "Main Thread Priority"	28

Revision History

Rev.	Date	Description	
		Page	Summary
1.00	Dec. 19, 2025	-	First edition issued
1.01	Mar. 17, 2026		Added descriptions related to IEEE 802.1Qbu (Frame Preemption).
1.02	May. 29, 2026		Added descriptions related to following: IEEE802.1Qci IEEE802.1Qch IEEE802.1AB
1.03	Sep. 24, 2026		Added descriptions related to RZ/T2N-Evb and multi-process support.

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 be responsible for determining what licenses are required from any third parties, and obtaining such licenses for the lawful import, export, manufacture, sales, utilization, distribution or other disposal of any products incorporating Renesas Electronics products, if required.
5. 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.
6. 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.

7. No semiconductor product is absolutely secure. Notwithstanding any security measures or features that may be implemented in Renesas Electronics hardware or software products, Renesas Electronics shall have absolutely no liability arising out of any vulnerability or security breach, including but not limited to any unauthorized access to or use of a Renesas Electronics product or a system that uses a Renesas Electronics product. RENESAS ELECTRONICS DOES NOT WARRANT OR GUARANTEE THAT RENESAS ELECTRONICS PRODUCTS, OR ANY SYSTEMS CREATED USING RENESAS ELECTRONICS PRODUCTS WILL BE INVULNERABLE OR FREE FROM CORRUPTION, ATTACK, VIRUSES, INTERFERENCE, HACKING, DATA LOSS OR THEFT, OR OTHER SECURITY INTRUSION ("Vulnerability Issues"). RENESAS ELECTRONICS DISCLAIMS ANY AND ALL RESPONSIBILITY OR LIABILITY ARISING FROM OR RELATED TO ANY VULNERABILITY ISSUES. FURTHERMORE, TO THE EXTENT PERMITTED BY APPLICABLE LAW, RENESAS ELECTRONICS DISCLAIMS ANY AND ALL WARRANTIES, EXPRESS OR IMPLIED, WITH RESPECT TO THIS DOCUMENT AND ANY RELATED OR ACCOMPANYING SOFTWARE OR HARDWARE, INCLUDING BUT NOT LIMITED TO THE IMPLIED WARRANTIES OF MERCHANTABILITY, OR FITNESS FOR A PARTICULAR PURPOSE.
8. 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.
9. 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.
10. 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.
11. 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.
12. 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.
13. This document shall not be reprinted, reproduced or duplicated in any form, in whole or in part, without prior written consent of Renesas Electronics.
14. 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.5.0-1 October 2020)

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/.