

# **Renesas RA Family**

# Tracealyzer® for FreeRTOS debugging

# Introduction

FreeRTOS is an RTOS from Amazon Web Services, which is based on a high-performance embedded kernel.

Percepio Tracealyzer® is the premier solution for visual trace diagnostics for developers of RTOS- or Linuxbased embedded software systems.

This application note provides procedures to check FreeRTOS thread and object states (referred to as resources) during the development of applications in e<sup>2</sup> studio. The procedure for starting Tracealyzer® is also explained.

### **Target Device**

RA6M3 MCU Group (R7FA6M3AH)

### **Operating Environment (using UART)**

Target Board	EK-RA6M3
IDE	e <sup>2</sup> studio version 2021-04 and FSP v3.0.0
Trace Tool	Percepio Tracealyzer® v4.4.2
OS	FreeRTOS 10.4.3
Toolchains	GNU Arm Embedded Toolchain: 9-2020-q2-major
	(GNU ARM Embedded 9.3.1.20200408)
Cable	FTDI TTL-232R-3V3 (USB to TTL Serial Cable)

# **Operating Environment (using J-Link RTT)**

Target Board	EK-RA6M3
IDE	e <sup>2</sup> studio version 2021-04 and FSP v3.0.0
Trace Tool	Percepio Tracealyzer® v4.4.2
OS	FreeRTOS 10.4.3
Toolchains	GNU Arm Embedded Toolchain: 9-2020-q2-major
	(GNU ARM Embedded 9.3.1.20200408)

Note: Please download and install tools from the following URL in advance.

- Quick Start Guide for e<sup>2</sup> studio for RA download site: Quick Start Guide for e<sup>2</sup> studio for RA
- FSP with e<sup>2</sup> studio installer download site: <u>https://github.com/renesas/fsp/releases</u>
- EK-RA6M3 Example Project Bundle Sample Code download site: <u>EK-RA6M3 Example Project Bundle - Sample Code</u>
- Tracealyzer® for FreeRTOS User Manual site: <u>Tracealyzer® for FreeRTOS - User Manual</u>
- Percepio Tracealyzer® download site:
   <u>Download Tracealyzer® Percepio AB</u>



# Contents

1.	Install FSP with e <sup>2</sup> studio	4
2.	Install Tracealyzer®	4
3.	Creating a project in e <sup>2</sup> studio	4
4.	Debugging via UART with Tracealyzer®	
4.1	Copy and remove Tracealyzer® for FreeRTOS into a project	7
4.1.1	Copy Tracealyzer® for FreeRTOS source under the Tracealyzer® installation folder	7
4.1.2	Remove unnecessary folders	7
4.2	FSP Configuration	8
4.2.1	UART driver settings	8
4.2.2	Blinky Thread settings	. 11
4.2.3	Generate Project Content	. 14
4.3	Code editing for Tracealyzer® connections	. 15
4.3.1	Create folder and file for UART	. 15
4.3.2	Add include file to task.c and timers.c	. 17
4.3.3	Add include files to trcKemelPort.c and trcStreamingRecorder.c	. 17
4.3.4	Change macro definitions in trcConfig.h	. 18
4.3.5	Add Include path to e <sup>2</sup> studio properties	. 18
4.3.6	Add code to hal_entry.c	. 19
4.3.7	Build the project	. 20
4.4	Connect PC and EK-RA6M3 Board	. 20
4.5	Using the RTOS Resource View	. 21
4.5.1	Displaying the RTOS Resources View	
4.5.2		
4.5.3	Stack setting	. 23
4.5.4	5	
4.6	Start debugging a project with Tracealyzer®	
4.6.1	Launch debugger on e <sup>2</sup> studio	
4.6.2		
4.6.3		
4.6.4		
4.6.5	-	
5.	Debugging via J-Link RTT with Tracealyzer®	.28
5.1	Copy and Remove Tracealyzer® for FreeRTOS into a project	. 28
5.1.1	Copy Tracealyzer® for FreeRTOS source under the Tracealyzer® installation folder	. 28
5.1.2	Remove unnecessary folders	. 28
5.2	FSP Configuration	. 30



# Renesas RA Family

5.2.2	Generate Project Content	. 33
5.3	Code editing for Tracealyzer® connections	. 34
5.3.1	Add include file to task.c and timers.c	. 34
5.3.2	Add include files to trcKemelPort.c and trcStreamingRecorder.c	. 34
5.3.3	Change macro definitions in trcConfig.h	. 35
5.3.4	Add Include path on e <sup>2</sup> studio properties	. 35
5.3.5	Add code to hal_entry.c	. 36
5.3.6	Build the project	. 37
5.4	Connect PC and EK-RA6M3 Board	. 37
5.5	Using the RTOS Resource View	. 38
5.6	Start debugging a project with Tracealyzer®	. 38
5.6.1	Launch debugger on e <sup>2</sup> studio	. 38
5.6.2	Launch Tracealyzer®	. 38
5.6.3	Set to Recording Settings	. 38
5.6.4	Start Recording a Trace	. 40
5.6.5	Trace information is displayed	. 41
Revis	sion History	.43



# 1. Install FSP with e<sup>2</sup> studio

Refer to "2.1 Installing the FSP with e<sup>2</sup> studio Installer" of "Renesas e<sup>2</sup> studio 2021-04 or higher User's Manual: Quick Start Guide".

#### 2. Install Tracealyzer®

Refer to Tracealyzer® for FreeRTOS User Manual.

#### 3. Creating a project in e<sup>2</sup> studio

A project generation wizard is available in e<sup>2</sup> studio to generate an RA project with a project name and the associated device and board, including drivers.

Launch e<sup>2</sup> studio and choose a workspace folder in the e<sup>2</sup> studio Launcher. To create a new RA project, follow these steps:

#### 1. Select File menu > New > Renesas C/C++ Project > Renesas RA.

2. Select the Renesas RA: Renesas RA C/C++ Project template. Click Next to continue.

😰 New C/C++ Proj	ect	_		×
Templates for Rer	nesas RA Project			
All C/C++	Renesas RA C/C++ Project Create an executable or static library C/C++	project foi	r Renesa:	as RA.
?	< Back Next > Finish		Cance	el

Figure 1. Template Selection

3. In the next dialog box, enter a project name and click **Next**.

Renesas RA C/C++ Project
Renesas RA C/C++ Project       Project Name and Location
Project name          RATraceAlyzer         Use default location
Choose file system: default
You can download more Renesas packs here
Cancel

Figure 2. Project Name and Location



- 4. In the device selection dialog, enter device and tool information as follows.
  - FSP version: 3.0.0
  - Board: EK-RA6M3
  - Device: Auto selected
  - Language: C
  - Toolchain version: Latest GNU Arm Embedded Toolchain approved for use with Renesas RA. (for example, GCC ARM Embedded 9.3.1.20200408)
  - Debugger: J-Link ARM
  - Click **Next** to continue

Renesas RA C/C++ Project	– 🗆 X
Renesas RA C/C++ Project Device and Tools Selection	, in the second se
Device Selection FSP Version: 3.0.0 Board: EK-RA6M3 Device: R7FA6M3AH3CFC Language: © C O C++	Board Description       Evaluation kit for RA6M3 MCU Group       Visit https://www.renesas.com/ra/ek-ra6m3 to get kit user's manual, quick start guide, errata, design package, example projects, etc.       Device Details       TrustZone     No       Pins     176       Processor     Cortex-M4
Toolchains GNU ARM Embedded	Debugger J-Link ARM
9.3.1.20200408 ~	
?	< Back Next > Finish Cancel

Figure 3. Create New Project for EK-RA6M3

5. Build Artifact Selection: **Executable**. RTOS Selection: **FreeRTOS** 

Renesas RA C/C++ Project		— 🗆 ×	
Renesas RA C/C++ Project Build Artifact and RTOS Selection			
Build Artifact Selection	RTOS Selection		
<ul> <li>Executable</li> <li>Project builds to an executable file</li> </ul>	FreeRTOS (v3.0.0)	~	
<ul> <li>Static Library</li> <li>Project builds to a static library file</li> </ul>			
<ul> <li>Executable Using an RA Static Library</li> <li>Project builds to an executable file</li> <li>Project uses an existing RA static library project</li> </ul>			
?	k Next > Fir	ish Cancel	
	Renesas RA C/C++ Project Build Artifact and RTOS Selection Build Artifact Selection Executable Project builds to an executable file Static Library Project builds to a static library file Executable Using an RA Static Library Project builds to an executable file Project uses an existing RA static library project	Renesas RA C/C++ Project         Build Artifact and RTOS Selection         Build Artifact Selection	Renesas RA C/C++ Project         Build Artifact and RTOS Selection         Build Artifact Selection

Figure 4. Build Artifact and RTOS Selection



#### 6. In the project template dialog, select **FreeRTOS – Binky – Static Allocation** and click **Finish**.

Renesas RA C	/C++ Project	- 0
Project Templa	te Selection	2
Project Temp	late Selection	
•	FreeRTOS - Blinky - Static Allocation FreeRTOS FSP project that includes BSP and will blink LEDs if available. FreeRTOS is pre-configured for static memory allocation. Th project will initialize the MCU using the BSP. FreeRTOS will also be initialized and a single thread to blink the LEDs will be started. [Renesas.RA.3.0.0.pack]	s
	FreeRTOS - Minimal - Static Allocation Empty FreeRTOS FSP project with no threads. FreeRTOS is pre-configured for static memory allocation. This project will initialize the MCU using the BSR	
	🖳 Empty FreeRTOS FSP project with no threads. FreeRTOS is pre-configured for static memory allocation. This project will initialize the	*
Code Generat	Empty FreeRTOS FSP project with no threads. FreeRTOS is pre-configured for static memory allocation. This project will initialize the MCU using the BSR [Renesas.RA.3.0.0.pack]	
	Empty FreeRTOS FSP project with no threads. FreeRTOS is pre-configured for static memory allocation. This project will initialize the MCU using the BSR [Renesas.RA.3.0.0.pack]	
	Empty FreeRTOS FSP project with no threads. FreeRTOS is pre-configured for static memory allocation. This project will initialize the MCU using the BSR [Renesas.RA.3.0.0.pack]	
	Empty FreeRTOS FSP project with no threads. FreeRTOS is pre-configured for static memory allocation. This project will initialize the MCU using the BSR [Renesas.RA.3.0.0.pack]	

Figure 5. Project Template Selection

7. Once complete, e<sup>2</sup> studio creates a new project with the FSP Configuration perspective open and ready for project configuration.

🀔 🐐 🔳 🕸 Debug	✓ RATraceAlyzer Debug_Flat ✓ ♦	9 • 🗐 🐚   🛞 • 🔨 • 🗟 💋   🔗	- 📮 😳	💸   🎄 - 9	<b>•</b> •	
∿, • 卷 ⊪ □□ 🗑 🖏 🕹 😻 🖢 •				۹ 🗄	Ec C/C++	FSP Configuration
🍐 Project Explorer 🛛 📄 😫 🍸 🖇 🕯	ិ 🗇 🎲 [RATraceAlyzer] FSP Configuration 🖂			🌮 FSP Visuali	zation 🛿 👩	Package 🗖 🗖
✓ C RATraceAlyzer > ĵĵ) Includes > 29 ra	Summary	Generate Proje	t Content	The active edit	tor element doe	s not use this view
<ul> <li>&gt; 28 ra_gen</li> <li>&gt; 29 src</li> <li>&gt; 20 src,</li> <li>&gt; 20 sript</li> <li>☆ configuration.xml</li> <li>&gt; R7FA6M3AH3CFC.pincfg</li> <li>&gt; RATaceAlyzer Debug_Flat.launch</li> <li>&gt; ⑦ Developer Assistance</li> </ul>	Project Summary         Board:       EK-RA6M3         Device:       R7FA6M3AH3CFC         Toolchain:       GCC ARM Embedd         Toolchain Version:       9.3.1.20200408         FSP Version:       3.0.0         Project Type:       Flat         Selected software components         Simple application that blinks an LED us         Board Support Package Common Files         I/O Port         FreeRTOS         Arm CMSIS Version 5 - Core (M)         Image: Support         Support         Support         Support         Support         Support	ing FreeRTOS. v3.0.0 v3.0.0 v3.0.0 v3.0.0 v3.0.0 v5.7.0+fsp.3.0.0	~			
🗌 Properties 🔀 💦 Problems 👒 Smart B	rowser 📑 🖬 🖓 🗔 🕴 🗖					7 i - E
Property Va	lue	0 items Description		Module	Pin	Location
		<				

Figure 6. New Project for EK-RA6M3

# 4. Debugging via UART with Tracealyzer®

This section describes how to use Tracealyzer® with UART.

# 4.1 Copy and remove Tracealyzer® for FreeRTOS into a project

### 4.1.1 Copy Tracealyzer® for FreeRTOS source under the Tracealyzer® installation folder

Copy the <code>Program Files\Percecio\Tracealyzer 4\FreeRTOS\TraceRecorder folder into workspace folder src using File Explorer.</code>

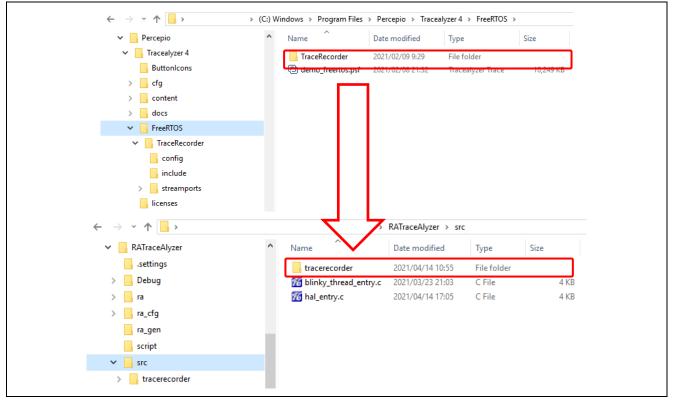


Figure 7. Copy Folder

# 4.1.2 Remove unnecessary folders

 $Remove \ all \ sub-folders \ in \ workspace \ folder \ \texttt{src/TraceRecorder/streamports}.$ 

RATraceAlyzer	^ Name	Date modified	Туре	Size
.settings	AFR_WIFI_LOCAL	2021/04/15 14:36	File folder	
Debug	ARM_ITM	2021/04/15 14:36	File folder	
- ra		2021/04/15 14:36	File folder	
ra_cfg	Jlink_RTT	2021/04/15 14:36	File folder	
ra_gen	STM32_USB_CDC	2021/04/15 14:36	File folder	
script	ТСРІР	2021/04/15 14:36	File folder	
src	TCPIP_Win32	2021/04/15 14:36	File folder	
tracerecorder				
config				
🔒 include				
streamports				

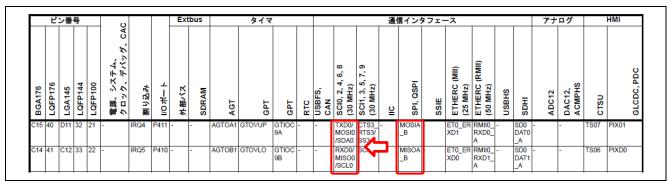
Figure 8. Remove Folder (UART)



# 4.2 FSP Configuration

#### 4.2.1 UART driver settings

Use P410/P411 (SCI0) for UART communication. EK-RA6M3 is selected SPI0 as default.



#### Figure 9. EK-RA6M3 on Pin Configuration

Disable SPI0 that is assigned "P410/P411" to use as SCI0 (UART Driver).

- Move to the **Pins** tab.
- Select Peripherals > Connectivity:SPI > SPI0.
- Operation Mode: **Disabled**.

Pin Group
Pin Group
^
>

Figure 10. Disable SPI0 on Pin Configuration



Move to the Stacks tab. Add the UART Driver by clicking New Stack > Driver > Connectivity > UART Driver on r\_sci\_uart.

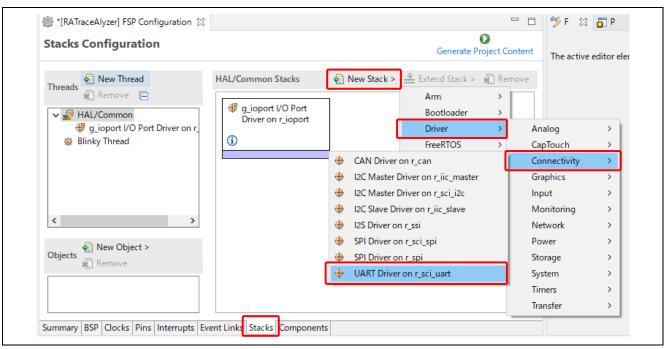


Figure 11. Add UART Driver on Stacks Configuration



Change the **Properties** of the UART driver as follows.

- Baud Rate: 921600
- Callback: sci\_callback\_tracealyzer
- TXD\_MOSI: **P411**
- RXD\_MISO: P410

Prope	ties 💥 💦 Problems 👒 Smart Browser	<b>1</b> 8 <b>1</b>		
_uart0	UART Driver on r_sci_uart			
Settings	Property	Value		
API Info	✓ Common			
	Parameter Checking	Default (BSP)		
	FIFO Support	Disable		
	DTC Support	Disable		
	Flow Control Support	Disable		
	<ul> <li>Module g_uart0 UART Driver on r_sci_uart</li> </ul>			
	> General			
	✓ Baud			
	Baud Rate	921600		
	Baud Rate Modulation	Disabled		
	Max Error (%)	5		
	> Flow Control			
	> Extra			
	✓ Interrupts			
	Callback	sci_callback_tracealyzer		
	Receive Interrupt Priority	Priority 12		
	Transmit Data Empty Interrupt Priority	Priority 12		
	Transmit End Interrupt Priority	Priority 12		
	Error Interrupt Priority	Priority 12		
	✓ Pins			
	TXD_MOSI	P411		
	RXD_MISO	P410		
	CTS_RTS_SS	<unavailable></unavailable>		

Figure 12. UART Properties



# 4.2.2 Blinky Thread settings

Add the Heap to the Blinky Thread as follows.

- Move to the **Stacks** tab.
- Select Blinky Thread.
- Click New Stack > FreeRTOS > Memory Management > Heap 1.

Stacks Configuration		0		
		Generate Project C	The active editor e	ement do
🕢 New Thread	Blinky Thread Stacks 🛛 👰 New Stack >	🚢 Extend Stack > 🛛 🐔 Re	move	
Threads Remove	Add stacks to the selected thread by us	Arm >		
V 🖉 HAL/Common	button (above), or by pasting here from			
g_ioport I/O Port Driver on r_		Driver >		
🕀 g_uart0 UART Driver on r_sci		FreeRTOS >	Libraries	>
Blinky Thread	<b>+</b>	Buffer Allocation 2	Memory Management	>
	<b>+</b>	Heap 1	Secure Sockets	>
	<b>+</b>	Heap 2	Transport	>
	<b>+</b>	Heap 3	1	
< >	<b>+</b>	Heap 4		
	<b>+</b>	Heap 5		
Objects			1	
Remove				

Figure 13. Add Heap on Stacks Configuration



Change the **Properties > General** on **Blinky Thread** as follows.

- Minimal Stack Size: 512
- Use Mutexes: Enabled
- Use Recursive Mutexes: Enabled
- Use Queue Sets: Enabled
- Enable Backward Compatibility: Enabled

Properties 🔀 🚮 Problems 🏟 Smart Browser	<b>1</b>
inky Thread	
Settings Property	Value ^
✓ Common	
✓ General	
Custom FreeRTOSConfig.h	
Use Preemption	Enabled
Use Port Optimised Task Selection	Disabled
Use Tickless Idle	Disabled
Cpu Clock Hz	SystemCoreClock
Tick Rate Hz	1000
Max Priorities	5
Minimal Stack Size	512
Max Task Name Len	16
Use 16-bit Ticks	Disabled
Idle Should Yield	Enabled
Use Task Notifications	Enabled
Use Mutexes	Enabled
Use Recursive Mutexes	Enabled
Use Counting Semaphores	Enabled
Queue Registry Size	10
Use Queue Sets	Enabled
Use Time Slicing	Disabled
Use Newlib Reentrant	Disabled
Enable Backward Compatibility	Enabled
Num Thread Local Storage Pointers	5
Stack Depth Type	uint32_t
Message Buffer Length Type	size_t
Library Max Syscall Interrupt Priority	Priority 1
Assert	assert(x)
Include Application Defined Privileged Functions	Disabled
> Hooks	v

#### Figure 14. Blinky Thread Properties 1



Change the **Properties > Hooks**, **Stats**, **Memory Allocation**, **Timers** on **Blinky Thread** as follows.

- Use Idle Hook: Disabled
- Use Malloc Failed Hook: Enabled
- Use Trace Facility: Enabled
- Use Stats Formatting Functions: Enabled
- Support Dynamic Allocation: Enabled
- Total Heap Size: **262,144** (256 \* 1,024)
- Timer Task Static Depth: **3,072** (1024 \* 3)

ettings	Property	Value	· · · · · · · · · · · · · · · · · · ·
	✓ Common		
	> General		
	✓ Hooks		
	Use Idle Hook	Disabled	
	Use Malloc Failed Hook	Enabled	
	Use Daemon Task Startup Hook	Disabled	
	Use Tick Hook	Disabled	
	Check For Stack Overflow	Disabled	
	✓ Stats		
	Use Trace Facility	Enabled	
	Use Stats Formatting Functions	Enabled	
	Generate Run Time Stats	Disabled	
	<ul> <li>Memory Allocation</li> </ul>		
	Support Static Allocation	Enabled	
	Support Dynamic Allocation	Enabled	
	Total Heap Size	262144	
	Application Allocated Heap	Disabled	
	✓ Timers		
	Use Timers	Enabled	
	Timer Task Priority	3	
	Timer Queue Length	10	

Figure 15. Blinky Thread Properties 2



Change the Properties (Optional Functions, RA, Logging) on Blinky Thread as follows.

- uxTaskGetStackHighWaterMark() Function: Enabled ٠
- eTaskGetState() Function: Enabled •
- xTimerPendFunctionCall() Function: Enabled •
- xTaskAbortDelay() Function: Enabled •
- Hardware Stack Monitor: Enabled •
- Logging Include Time and Task Name: Enabled •

Blinky T	lifeau			
Settings	Property	Value		^
-	✓ Common			
	> General			
	> Hooks			
	> Stats			
	> Memory Allocation			
	> Timers			
	<ul> <li>Optional Functions</li> </ul>			
	vTaskPrioritySet() Function	Enabled		
	uxTaskPriorityGet() Function	Enabled		
	vTaskDelete() Function	Enabled		
	vTaskSuspend() Function	Enabled		
	xResumeFromISR() Function	Enabled		
	vTaskDelayUntil() Function	Enabled		
	vTaskDelay() Function	Enabled		
	xTaskGetSchedulerState() Function	Enabled		
	xTaskGetCurrentTaskHandle() Function	Enabled		
	uxTaskGetStackHighWaterMark() Function	Enabled		
	xTaskGetIdleTaskHandle() Function	Disabled		
	eTaskGetState() Function	Enabled		
	xEventGroupSetBitFromISR() Function	Enabled		
	xTimerPendFunctionCall() Function	Enabled		
	xTaskAbortDelay() Function	Enabled		
	xTaskGetHandle() Function	Disabled		
	xTaskResumeFromISR() Function	Enabled		
	✓ RA			
	Hardware Stack Monitor	Enabled		
	✓ Logging			
	Print String Function	printf(x)		
	Logging Max Message Length	192		
	Logging Include Time and Task Name	Enabled		

Figure 16. Blinky Thread Properties 3

# 4.2.3 Generate Project Content

0

Click on the Generate Project Content button to generate the source files.



# 4.3 Code editing for Tracealyzer® connections

# 4.3.1 Create folder and file for UART

- Create the uart folder and trcSteamingPort.c and trcStreamingPort.h files, for example.
- Make trcSteamingPort.c and trcStreamingPort.h as shown in Figure 18 and Figure 19.

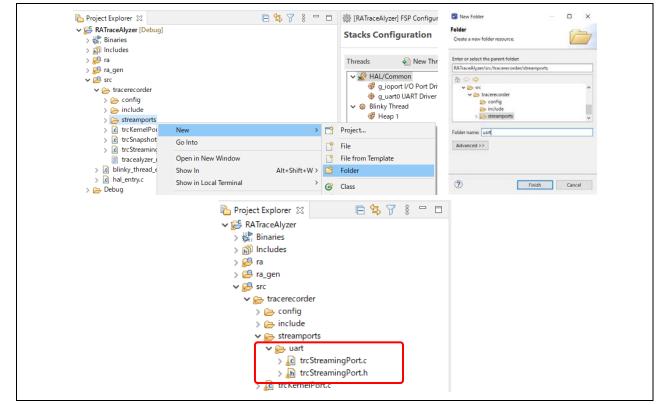


Figure 17. Create folder and file



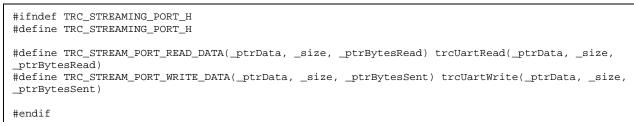
— Create file: trcStreamingPort.c

```
#include "bsp api.h"
#include "trcRecorder.h"
#include "r_sci_uart.h"
#include "r_uart_api.h"
#include <string.h>
#include "semphr.h"
#if (TRC_CFG_RECORDER_MODE == TRC_RECORDER_MODE_STREAMING)
#if (TRC_USE_TRACEALYZER_RECORDER == 1)
static uint8_t s_u8_string[1024];
static signed portBASE_TYPE xHigherPriorityTaskWoken;
static uint8_t sci_buffer[1024];
static uint32_t sci_current_received_size = 0;
extern sci_uart_instance_ctrl_t g_uart0_ctrl;
extern SemaphoreHandle_t semaphore_handle_1;
int32_t trcUartWrite(void* data, uint32_t size, int32_t *ptrBytesWritten)
{
    fsp_err_t err = FSP_SUCCESS;
   int32_t error_code = -1;
    if(size < sizeof(s_u8_string))</pre>
    {
           memcpy(s_u8_string, data, size);
           /* Writing to terminal */
           err = R_SCI_UART_Write (&g_uart0_ctrl, s_u8_string, size);
           if(err == FSP_SUCCESS)
           {
                   xSemaphoreTake( semaphore_handle_1, portMAX_DELAY );
                   *ptrBytesWritten = size;
                   error_code = 0;
           }
   return error code;
}
int32_t trcUartRead(void* data, uint32_t size, int32_t *ptrBytesRead)
{
   if(sci_current_received_size == size)
    {
           memcpy(data, sci_buffer, sci_current_received_size);
           *ptrBytesRead = sci_current_received_size;
           sci_current_received_size = 0;
    }
   return 0;
}
void sci_callback_tracealyzer(uart_callback_args_t *p_args)
{
    if(UART_EVENT_RX_CHAR == p_args->event)
    ł
           sci_buffer[sci_current_received_size] = (uint8_t ) p_args->data;
           if(sci_current_received_size == (sizeof(sci_buffer) - 1)) /* -1 means string
terminator after "\n" */
           {
                   sci_current_received_size = 0;
           }
           else
           {
                   sci current received size++;
           }
   else if(UART_EVENT_TX_COMPLETE == p_args->event)
           xHigherPriorityTaskWoken = pdFALSE;
           xSemaphoreGiveFromISR(semaphore_handle_1, &xHigherPriorityTaskWoken);
           portYIELD_FROM_ISR( xHigherPriorityTaskWoken );
    else
    }
#endif /*(TRC_USE_TRACEALYZER_RECORDER == 1)*/
#endif /*(TRC_CFG_RECORDER_MODE == TRC_RECORDER_MODE_STREAMING)*/
```

Figure 18. Create trcStreamingPort.c



— Create file: trcStreamingPort.h



#### Figure 19. Create the trcStreamingPort.h

#### 4.3.2 Add include file to task.c and timers.c

• #include "trcRecorder.h"

Project Explorer 🛛	🖻 🔄 🏹 🕴 🗖 🖬	
🗸 📂 RATraceAlyzer [Debug]		
> 🖑 Binaries		
> 🔊 Includes		
🗸 🚰 ra		
> 🔁 arm		
🗸 🗁 aws		
🗸 📂 amazon-freertos		
∽ 📂 freertos_kernel		
> 📂 include		
> 🗁 License		
> 👝 portable		
> c event_groups.c		
> 🖻 list.c		
> c queue.c		🏟 [RATraceAlyzer] FSP Configuration 🗈 tasks.c 🔀 🔎 timers.c
stream_buffer.c		37 /* FreeRTOS includes. */
> 💽 tasks.c		38 <b>#include</b> "trcRecorder.h"
> C timers.c		39 <b>#include</b> "FreeRTOS.h"
asks.c.bak		40 #include "task.h"
itimers.c.bak		41 #include "timers.h"
		42 #include "stack_macros.h"
> 👝 board		42 #Include Stack_macros.n

Figure 20. Add include to freertos\_kernel

#### 4.3.3 Add include files to trcKemelPort.c and trcStreamingRecorder.c

- #include "bsp\_api.h"
- #include "trcRecorder.h"

Project Explorer 🛛 📄 🔄 🖓 🖇 🖓
✓ 😴 RATraceAlyzer [Debug]
> 💥 Binaries
> 🔊 Includes
> 🚰 ra
> 🔑 ra_gen
✓ 🔑 src
✓ 😥 tracerecorder
> 😂 config
> 🗁 include
> 🚌 streamports
trcKernelPort.c
i.c) trcSnapshotRecorder.c
trcStreamingRecorder.c
🖹 tracealyzer_readme.txt
(ii) [RATraceAlyzer] FSP Configuration Icon trace in the image of the ima
2 ⊕ * Trace Recorder Library for Tracealyzer v4.4.2.
44 <b>#include</b> "bsp api.h"
45 <b>#include</b> "trcRecorder.h"
46 #include "FreeRTOS.h"
A 77

Figure 21. Add include tracerecorder



#### 4.3.4 Change macro definitions in trcConfig.h

Change only the red part of the macro definitions in trcConfig.h as follows.

- #include "bsp\_api.h"
- //#error "Trace Recorder: Please include your processor's header file here and remove this line."
- #define TRC\_CFG\_HARDWARE\_PORT TRC\_HARDWARE\_PORT\_ARM\_Cortex\_M
- #define TRC\_CFG\_RECORDER\_MODE TRC\_RECORDER\_MODE\_STREAMING
- #define TRC\_CFG\_FREERTOS\_VERSION TRC\_FREERTOS\_VERSION\_10\_4\_1

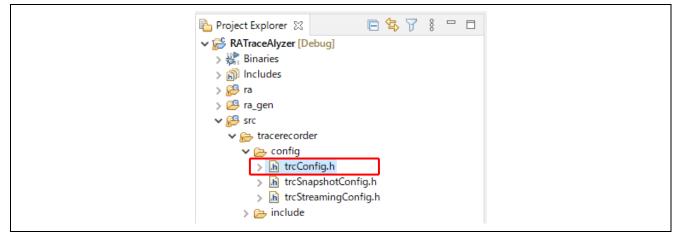


Figure 22. Change the define in trcConfig.h

#### 4.3.5 Add Include path to e<sup>2</sup> studio properties

Select menu **Project > Properties**, then click **Settings > Includes** to add the Include path.

- "\${workspace\_loc:/\${ProjName}/src/tracerecorder}"
- "\${workspace\_loc:/\${ProjName}/src/tracerecorder/config}"
- "\${workspace\_loc:/\${ProjName}/src/tracerecorder/include}"
- "\${workspace\_loc:/\${ProjName}/src/tracerecorder/streamports/uart}"

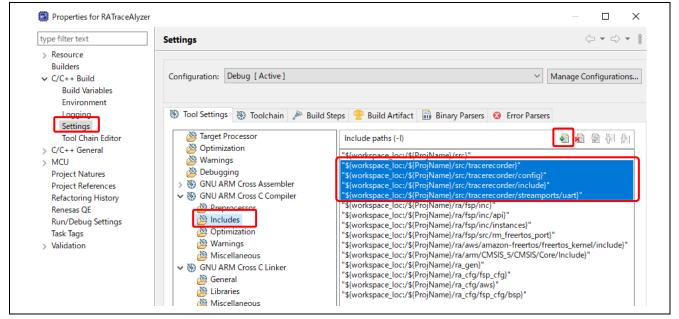


Figure 23. Setting Include paths in project properties



#### 4.3.6 Add code to hal\_entry.c

Add source code in red to hal\_entry.c.

```
#include "bsp_api.h"
#include "trcRecorder.h"
#include "FreeRTOS.h"
#include "semphr.h"
#include "hal_data.h"
SemaphoreHandle_t semaphore_handle_1;
StaticSemaphore t semaphore handle 1 memory;
void R_BSP_WarmStart(bsp_warm_start_event_t event);
*****************//**
 * This function is called at various points during the startup process. This implementation
uses the event that is
* called right before main() to set up the pins.
 * @param[in] event Where at in the start up process the code is currently at
void R_BSP_WarmStart (bsp_warm_start_event_t event)
{
   if (BSP_WARM_START_RESET == event)
#if BSP_FEATURE_FLASH_LP_VERSION != 0
       /* Enable reading from data flash. */
      R_FACI_LP->DFLCTL = 1U;
       /* Would normally have to wait tDSTOP(6us) for data flash recovery. Placing the enable
here, before clock and
       * C runtime initialization, should negate the need for a delay since the initialization
will typically take more than 6us. */
#endif
   }
   if (BSP_WARM_START_POST_C == event)
   {
       /* C runtime environment and system clocks are setup. */
       /* Configure pins. */
      R_IOPORT_Open(&g_ioport_ctrl, &g_bsp_pin_cfg);
   }
   fsp_err_t err = FSP_SUCCESS;
   /* Initialize UART channel with baud rate 115200 */
   err = R_SCI_UART_Open (&g_uart0_ctrl, &g_uart0_cfg);
   if (FSP_SUCCESS != err)
   semaphore_handle_1 = xSemaphoreCreateBinaryStatic (&semaphore_handle_1_memory);
   vTraceEnable(TRC_INIT);
}
```

Figure 24. Change the UART Streaming hal\_entry.c



### 4.3.7 Build the project

Right-click on the project and select Build Project.

Project Explorer 🔀	🖻 🔩 🍸 🕴 🗖
✓ RATraceAlyzer [Debug]	
> 🖓 Binaries	
> 🔊 Includes	
> 🚰 ra	
> 🚰 ra_gen	
🗸 🔁 src	
✓ → tracerecorder	
> 🔁 config	
> 🔁 include	
Die Conflicte Di Conselle M	
Pin Conflicts 📮 Console 🔀	
CDT Build Console [RATraceAlyzer]	
arm-none-eabi-gcc @"RATraceAlyzer.elf.in" 'Finished building target: RATraceAlyzer.elf' '.'	,
'Invoking: GNU ARM Cross Create Flash Image'	
arm-none-eabi-objcopy -O srec "RATraceAlyzer.	.elf" "RATraceAlyzer.srec"
'Invoking: GNU ARM Cross Print Size' arm-none-eabi-sizeformat=berkeley "RATrace	eAlvzer.elf"
text data bss dec hex filer	name
	aceAlyzer.elf
'Finished building: RATraceAlyzer.srec' 'Finished building: RATraceAlyzer.siz'	
• •	
11:02:54 Build Finished. 0 errors, 17 warning	gs. (took 12s.57ms)

Figure 25. Build Project

# 4.4 Connect PC and EK-RA6M3 Board

The figure below shows the connection between the host PC and the EK-RA6M3 board.

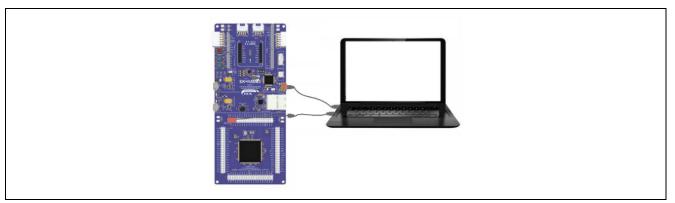


Figure 26. EK-RA6M3 Board Connection

The hardware settings are as follows:

Table 1.	Jumper	Connection	Summary for	Different	Debug Modes
----------	--------	------------	-------------	-----------	-------------

Debug Modes	J8	<b>J</b> 9	J29
Debug on-board	Jumper on pins 1-2	Open	Jumpers on pins 1-2, 3-4, 5-6, 7-8



Terminal connection (UART):

- PC port TXD: ORANGE EK-RA6M3 board P410 (RXD\_MISO)
- PC port RXD: YELLOW EK-RA6M3 board P411 (TXD\_MOSI)
- PC port GND: BLACK EK-RA6M3 board GND

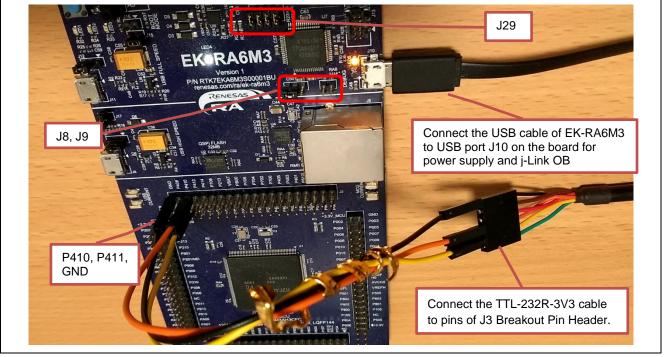


Figure 27. Connection between PC and EK-RA6M3 Board

### 4.5 Using the RTOS Resource View

The e<sup>2</sup> studio has an RTOS resource view function that displays the state of FreeRTOS resources. This procedure describes how to use the RTOS resource view.

#### 4.5.1 Displaying the RTOS Resources View

Because the **RTOS Resources** view functions only with the debugger running, start the debugger and select **Renesas Views > Partner OS > RTOS Resources**. When the **Select OS** dialog box is displayed, select FreeRTOS as shown in Figure 28. The **RTOS Resources** view appears view appears as shown in Figure 29.

🖓 RTOS Re Select OS	sources 🛛	
OS:		
	Never show display the Select OS at download.	
	ОК	

Figure 28. Selecting the OS



	OS Re	sources	X					
Stack	Task	Queue	Timer					
No.	Task	Name		Base/ActualPriority	State	EventObject	TotalTickCount	DeltaTickCount
1	Blin	ky Threa	d	1/1	BLOCKED	None	-(-%)	-(-%)
2	IDLE			0/0	READY	None	-(-%)	-(-%)
3	Tmr Svc			3/3	SUSPENDED	None	-(-%)	-(-%)
4								
5								

#### Figure 29. RTOS Resources view

#### 4.5.2 Context menu

Display the context menu by right-clicking the mouse on the RTOS Resources view.

Ċ,	Real-time Refresh Column	>
	Real-time Refresh Interval	-
	Stack Setting	
Ś	Update information	
	Jump to source	
	Save File	
ାଷ୍ପ	Select OS	

Figure 30. Context menu

Explanation:

- Real-time Refresh Column: Allows real-time display for the displayed items. This is not valid while the program is running.
- Real-time Refresh Interval: Specifies interval time for updating of the real-time display. The specifiable range is 500ms to 10000ms. This is not valid while the program is running.

#### Stack Setting: Enables/disables Stack Loading and stack threshold setting for stack alert function. This is not valid while a program is running.

- Update information: Updates the information.
- Jump to source:

Opens an editor view displaying the source code of the task/thread or handler. Double-clicking the task/thread or handler also opens an editor view.

This is not valid while the program is running.

• Save File:

Saves the data of the current tab in the text file (\*.txt).

This is not valid while the program is running.

• Select OS:

Opens the **Select OS** dialog box.

This is not valid while the program is running.



#### 4.5.3 Stack setting

Enable load stack data and set stack threshold.

- 1. Open the context menu and select **Stack Setting**.
- 2. To load stack data to the **RTOS Resource** view, check **Enable loading Stack data** checkbox in the **Stack Setting** dialog. If this option is not enabled, stack data will not be loaded in the next debugging session.

Stack Setting				×
Enable loading Stack o				
Stack Threshold (%)	80.00			▲ ▼
		OK	Cancel	

#### Figure 31. Enable loading stack data

3. The desired threshold value can be set in the **Stack Threshold (%)** textbox. Click **OK** to save the setting.

Stack Setting					$\times$
Enable loading Stack of					
Stack Threshold (%)	80.00				▲ ▼
		ОК	C	ancel	

#### Figure 32. Set up threshold value

4. Run then suspend the target project to load stack data. The stack threshold warning will pop up if the threshold set is met.

There are 2 types of warning popup: **Stack Threshold Warning** (list of threads which reached stack threshold value set as above) and **Stack Overflow Warning** (reached 100%).

Stack Threshold Warning	×	Stack Overflow Warning	×
Blinky Thread (No.1)	~	Blinky Thread (No.1)	^
IDLE (No.2)		IDLE (No.2)	
Tmr Svc (No.3)	$\checkmark$	Tmr Svc (No.3)	$\checkmark$
Click here to view Stack	c tab	Click here to view Stack	tab

Figure 33. Stack Threshold Warning popup (left) and Stack Overflow Warning popup (right)



### 4.5.4 Tab menu

Table 2 show display items for each tab.

#### Table 2. Contents of each tabbed window

Name of tabbed window in the RTOS Resources view	Displayed information andselections	Information to be displayed
Stack	No.	Row index
	TaskName	The name assigned to the task upon creation
	StartOfStack	The address of the start of stack
	EndOfStack	The address of the end of stack
	TopOfStack	The address of the top of the stack where it is last writter to when the context of the stack was saved
	StackSize(bytes)	Total stack size
	StackUsageSize	Stack usage at high water mark
	StackUsageRatio	Percentage of usage at high water mark relative to total stack size
Task	No.	Row index
	TaskName	The name assigned to the task upon creation
	Base/ActualPriority	The base priority used by the priority inheritance mechanism/The actual priority used by the task
	State	State of the task which includes "RUNNING", "READY", "BLOCKED" and "SUSPENDED"
	EventObject	The name of the queue which causes the task to be blocked
	TotalTickCount	The total number of tick count for the task to be active
	DeltaTickCount	The number of tick count for the task to be active since previous suspend event
Queue	No.	Row index
	Name (Type)	The name assigned to the queue upon registration and its type (Queue, Semaphore, or Mutex)
	Address	The address of the queue handle
	MaxLength	Size per item in the queue (in bytes)
	ItemSize	Message size
	CurrentLength	Number of items currently stored in the queue
	#WaitingTx	Number of tasks blocked while waiting to send to the queue
	#WaitingRx	Address where the message queue starts
	EndAddress	Address where the message queue ends
Timer	No.	Row index
	Name	The current period of the timer in system ticks
	Period	Automatic reload enable/disable. "On" when auto reload is enabled, which resets the timer each time it expires, "Off" when auto reload is disabled which does nothing when the timer expires
	CallbackFn	Address and <name> of the callback function which executes each time the timer ends</name>
	TimerID	The numeric ID of the timer assigned in hexadecimal format when it was created

# 4.6 Start debugging a project with Tracealyzer®

# 4.6.1 Launch debugger on e<sup>2</sup> studio

Select menu **Run > Debug** to launch the debugger.



#### 4.6.2 Launch Tracealyzer®

Launch installed Tracealyer 4 on PC.

### 4.6.3 Set Recording Settings

Click **Recording settings** on Tracealyzer and select **PSF Streaming Settings** on Tracealyzer. Set the following.

- Device: COM8 (user pc system port)
- Data bits: 8
- Data rate: **921600**
- Handshake: None
- Parity: None
- Stop bits: One

			Settings - PSF Streaming Settings				-		$\times$
			Enter text to filter controls	PSF Streamin	e Settines				
			Global Settings	Target Connec	ction: SerialPort	~			
			Project Settings						
			View Settings	Device	COM8				
			- Automatic Event Filtering - Docking Settings	Data bits	8				
🔁 Fercepin Tracealyzer - Weidlaw 1	-	□ ×	- Navigation Bar Settings	Data rate	921600				
File Trace View Window Help		Feedback	- API Settings	Handshake	None				
Welcome to Percepio Traceal ×			- Performance Settings - J-Link Settings		None				
Contraction of the second seco		i i	ST-LINK Settings	Parity	None 🗸				
🔘 percepio*			PSF Streaming Settings	Stop bits	One 🗸				
		_	Snapshot Settings						
Welcome to Tracealyzer	Record a Trace	- 1							
Percepio Tracealyzer is a powerful tool for tracing and visualization of RTOS- and Linux-based embedded software systems. More than 25 views offers amazing insight into the real- time behavior, speeding up debugging, validation and performance optimization.	Recording Settings								
To enable tracing in your target system, follow the step-by-step guide provided in the User Manual.	Read Snapshot Trace								
Getting Started     Vser Manual	Traces								
Percepio News	trace-2021-04-05_16 39.55 psf								
Traceelyzer Version 4.4.2 Is Out With Support for Azure RTOS	2021/04/05 16:39:55, 337.3 KB								
ThreadX SMP	trace-2021-04-05_16 27.11 psf								
We released Tracealyzer version 4.4.2 yesterday, with support for	2021/04/05 16:27 11, 1.3 MB								
symmetric multiprocessing systems running Azure RTOS	trace.2021.04.05 16 22 41 nst			Help		ОК	Apply	Car	incel

Figure 34. Set to Record UART Settings



### 4.6.4 Start Recording a Trace

Click Record Streaming Trace to start recording a Trace.

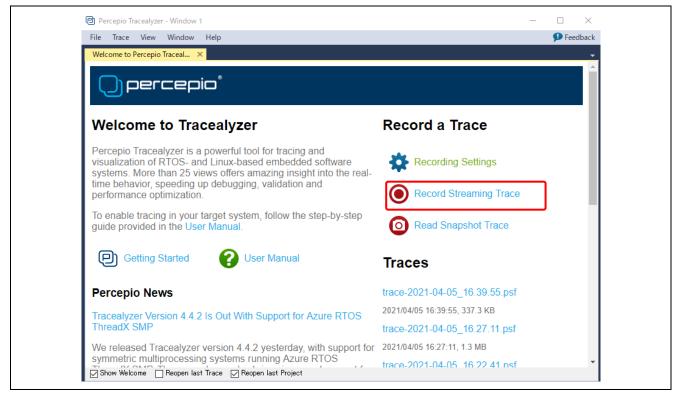


Figure 35. Start Record Streaming Trace



### 4.6.5 Trace information is displayed

Various analysis modes are provided. For more information, see Help tab.

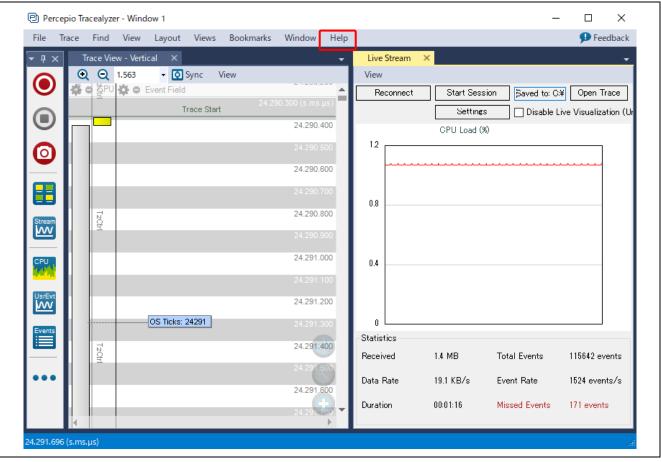


Figure 36. Displayed Trace information



# 5. Debugging via J-Link RTT with Tracealyzer®

This section describes how to use Tracealyzer® in J-Link RTT.

# 5.1 Copy and Remove Tracealyzer® for FreeRTOS into a project

### 5.1.1 Copy Tracealyzer® for FreeRTOS source under the Tracealyzer® installation folder

Copy the Program Files\Percecio\Tracealyzer 4\FreeRTOS\TraceRecorder folder into workspace folder src using File Explorer.

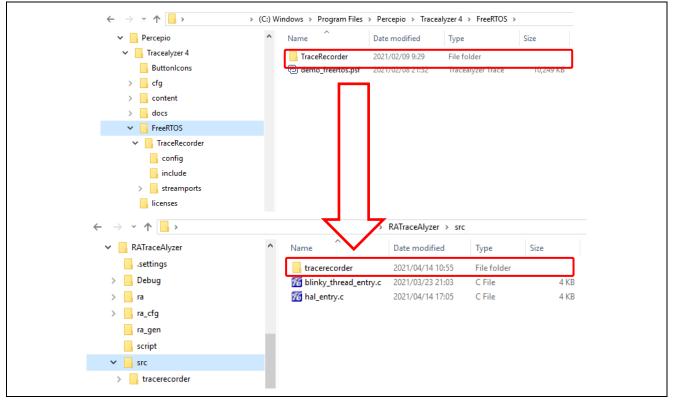


Figure 37. Copy Folder

# 5.1.2 Remove unnecessary folders

1. Remove sub-folders marked in red in workspace folder src/TraceRecorder/streamports as shown in Figure 5-2.

RATraceAlyzer	`	Name	Date modified	Туре
.settings		AFR_WIFI_LOCAL	2021/04/23 16:20	File folder
Debug		ARM_ITM	2021/04/23 16:20	File folder
ra		File	2021/04/23 16:20	File folder
ra_cfg	Ι.	Jlink_RTT	2021/04/22 10:22	File folder
ra_gen		STM32_USB_CDC	2021/04/23 16:20	File folder
script		TCPIP	2021/04/23 16:20	File folder
		TCPIP_Win32	2021/04/23 16:20	File folder
src				
TraceRecorder				
📙 config				
include				
🔒 streamports				

Figure 38. Remove Folder (J-Link RTT)



#### 2. Copy J-Link RTT files in EK-RA6M3 Example Project Bundle - Sample Code. Overwrite the files in the

\ek\_ra6m3\sci\_uart\sci\_uart\_ek\_ra6m3\_ep\e2studio\src\SEGGER\_RTT folder into
workspace folder src/TraceRecorder/streamports" using File Explorer.

- File segger\_rttc
- File SEGGER\_RTT\_printfc
- File SEGGER\_RTTh
- File SEGGER\_RTT\_Confh

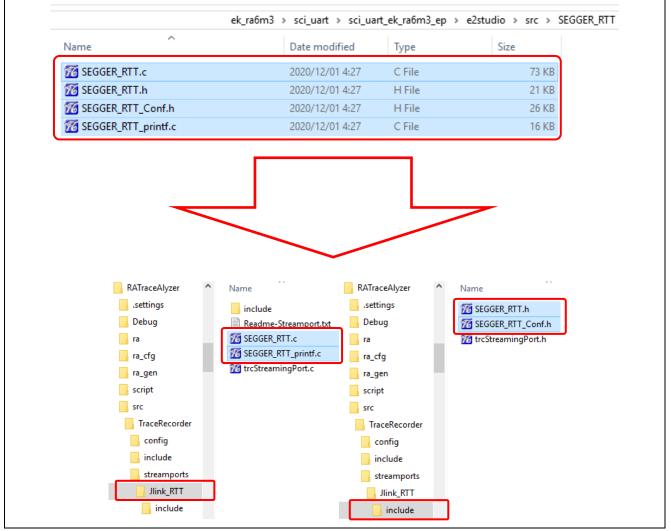


Figure 39. Copy the SEGGER\_RTT files



# 5.2 **FSP** Configuration

### 5.2.1 Blinky Thread Settings

Add the Heap to the Blinky Thread as follows.

- Move to the **Stacks** tab.
- Select Blinky Thread.
- Click New Stack > FreeRTOS > Memory Management > Heap 1.

			D	° 🗖 🦘 F 💥 👼 F	
Stacks Configuration		Generate Pr	-	tent The active editor	r element do
New Thread	Blinky Thread Stacks	► 🐣 Extend Stack >	Remo	ove	
Remove	Add stacks to the selected thread by	Arm	>		
V 🖉 HAL/Common	button (above), or by pasting here fr		>		
🔮 g_ioport I/O Port Driver on r_		Driver	>		
🕀 g_uart0 UART Driver on r_sci		FreeRTOS	>	Libraries	>
Blinky Thread		Buffer Allocation 2		Memory Managemer	nt ≻
		🕀 Heap 1		Secure Sockets	>
	•	🕀 Heap 2		Transport	>
	€	🕀 Heap 3	i i		
< >	€	🕀 Heap 4			
E New Objects	€	🕀 Heap 5			
Objects					
Remove					
Summary BSP Clocks Pins Interrupts Ev	ent Links Stacks Components				

Figure 40. Add Heap on Stacks Configuration



Change the **Properties > General** on **Blinky Thread** as follows.

- Minimal Stack Size: 512
- Use Mutexes: Enabled
- Use Recursive Mutexes: Enabled
- Use Queue Sets: Enabled
- Enable Backward Compatibility: Enabled

Properties 🛛 🔐 Problems 🌸 Smart Browser	<b>1</b>
linky Thread	
Settings Property	Value ^
✓ Common	
✓ General	
Custom FreeRTOSConfig.h	
Use Preemption	Enabled
Use Port Optimised Task Selection	Disabled
Use Tickless Idle	Disabled
Cpu Clock Hz	SystemCoreClock
Tick Rate Hz	1000
Max Priorities	5
Minimal Stack Size	512
Max Task Name Len	16
Use 16-bit Ticks	Disabled
Idle Should Yield	Enabled
Use Task Notifications	Enabled
Use Mutexes	Enabled
Use Recursive Mutexes	Enabled
Use Counting Semaphores	Enabled
Queue Registry Size	10
Use Queue Sets	Enabled
Use Time Slicing	Disabled
Use Newlib Reentrant	Disabled
Enable Backward Compatibility	Enabled
Num Thread Local Storage Pointers	5
Stack Depth Type	uint32_t
Message Buffer Length Type	size_t
Library Max Syscall Interrupt Priority	Priority 1
Assert	assert(x)
Include Application Defined Privileged Functions	Disabled
> Hooks	~

#### Figure 41. Blinky Thread Properties 1



Change the **Properties > Hooks**, **Stats**, **Memory Allocation**, **Timers** on **Blinky Thread** as follows.

- Use Idle Hook: Disabled
- Use Malloc Failed Hook: Enabled
- Use Trace Facility: Enabled
- Use Stats Formatting Functions: Enabled
- Support Dynamic Allocation: Enabled
- Total Heap Size: 262,144 (256 \* 1,024)
- Timer Task Static Depth: 3,072 (1024 \* 3)

	hread		
ttings	Property	Value	-
-	✓ Common		
	> General		
	✓ Hooks		
	Use Idle Hook	Disabled	
	Use Malloc Failed Hook	Enabled	
	Use Daemon Task Startup Hook	Disabled	
	Use Tick Hook	Disabled	
	Check For Stack Overflow	Disabled	
	✓ Stats		
	Use Trace Facility	Enabled	
	Use Stats Formatting Functions	Enabled	
	Generate Run Time Stats	Disabled	
	<ul> <li>Memory Allocation</li> </ul>		
	Support Static Allocation	Enabled	
	Support Dynamic Allocation	Enabled	
	Total Heap Size	262144	
	Application Allocated Heap	Disabled	
	✓ Timers		
	Use Timers	Enabled	
	Timer Task Priority	3	
	Timer Queue Length	10	
	Timer Task Stack Depth	3072	

Figure 42. Blinky Thread Properties 2



Change the **Properties > Optional Functions**, **RA**, **Logging** on **Blinky Thread** as follows.

- uxTaskGetStackHighWaterMark() Function: Enabled ٠
- eTaskGetState() Function: Enabled •
- xTimerPendFunctionCall() Function: Enabled •
- xTaskAbortDelay() Function: Enabled •
- Hardware Stack Monitor: Enabled •
- Logging Include Time and Task Name: Enabled •

Blinky T	hread					
Settings	Property	Value		^		
_	✓ Common					
	> General					
	> Hooks					
	> Stats					
	> Memory Allocation					
	> Timers					
	<ul> <li>Optional Functions</li> </ul>					
	vTaskPrioritySet() Function	Enabled				
	uxTaskPriorityGet() Function	Enabled				
	vTaskDelete() Function	Enabled				
	vTaskSuspend() Function	Enabled				
	xResumeFromISR() Function	Enabled				
	vTaskDelayUntil() Function	Enabled				
	vTaskDelay() Function	Enabled				
	xTaskGetSchedulerState() Function	Enabled				
	xTaskGetCurrentTaskHandle() Function	Enabled				
	uxTaskGetStackHighWaterMark() Function	Enabled				
	xTaskGetIdleTaskHandle() Function	Disabled				
	eTaskGetState() Function	Enabled				
	xEventGroupSetBitFromISR() Function	Enabled				
	xTimerPendFunctionCall() Function	Enabled				
	xTaskAbortDelay() Function	Enabled				
	xTaskGetHandle() Function	Disabled				
	xTaskResumeFromISR() Function	Enabled				
	✓ RA					
	Hardware Stack Monitor	Enabled				
	✓ Logging					
	Print String Function	printf(x)				
	Logging Max Message Length	192				
	Logging Include Time and Task Name	Enabled				

Figure 43. Blinky Thread Properties 3

#### 5.2.2 Generate Project Content

0

Click on the Generate Project Content button to generate the source files.



# 5.3 Code editing for Tracealyzer® connections

### 5.3.1 Add include file to task.c and timers.c

• #include "trcRecorder.h"

Project Explorer 🐹	E 4	5	000	- 0	
∽ 😴 RATraceAlyzer [Debug]					
> 🐝 Binaries					
> 🔊 Includes					
🗸 🚰 ra					
> 🗁 arm					
🗸 🗁 aws					
∨ 🕞 amazon-freertos					
∽ 📂 freertos_kernel					
> 📂 include					
> 🔁 License					
> 👝 portable					
> c event_groups.c					
> 💽 list.c					
> c queue.c					🍈 [RATraceAlyzer] FSP Configuration 🔒 tasks.c 💥 🔎 timers.c
stream_buffer.c					37 /* FreeRTOS includes. */
tasks.c					38 <b>#include</b> "trcRecorder.h"
timers.c					39 #include "FreeRTOS.h"
tasks.c.bak					40 #include "task.h"
timers.c.bak					41 #include "timers.h"
					41 #include cimers.n 42 #include "stack macros.h"
> 📂 board					42 #Include Stack_Macros.n

Figure 44. Add include to freertos\_kernel

### 5.3.2 Add include files to trcKemelPort.c and trcStreamingRecorder.c

- #include "bsp\_api.h"
- #include "trcRecorder.h"

Project Explorer 🛛	🖻 🔄 🏹 🕴 🗖 🛛	
✓ 📂 RATraceAlyzer [Debu	9]	
> 🖗 Binaries		
> 🔊 Includes		
> 🔁 ra		
> 🔑 ra_gen		
V 🔁 src		
✓		
> 🔁 config		
> 🔁 include		
> 🕞 streamports		
> 🔂 trcKernelPor		
> c) trcSnapshot		
> 🔂 trcStreaming		
tracealyzer_r	eadme.txt	
(RATraceAlyzer] FSP Configuration	🖟 trcKernelPort.c 🛛	
	rder Library for Tracealyz	ver v4.4.2.
44 #include "bsp		
45 #include "tro		
46 #include "Fre	eRTOS.h"	
A 77		

Figure 45. Add include tracerecorder



#### 5.3.3 Change macro definitions in trcConfig.h

Change only the red part of macro definitions trcConfig.h as follows.

- #include "bsp\_api.h".
- //#error "Trace Recorder: Please include your processor's header file here and remove this line.".
- #define TRC\_CFG\_HARDWARE\_PORT TRC\_HARDWARE\_PORT\_ARM\_Cortex\_M.
- #define TRC\_CFG\_RECORDER\_MODE TRC\_RECORDER\_MODE\_STREAMING.
- #define TRC\_CFG\_FREERTOS\_VERSION TRC\_FREERTOS\_VERSION\_10\_4\_1.

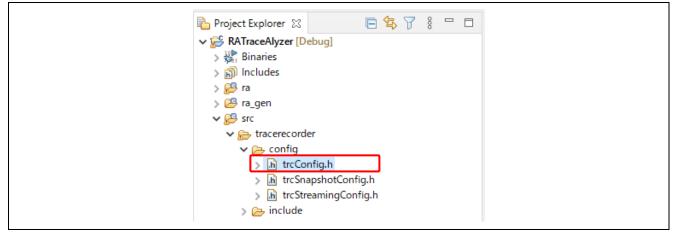


Figure 46. Change the define in trcConfig.h

#### 5.3.4 Add Include path on e<sup>2</sup> studio properties

Select menu Project > Properties, then click Settings > Includes to add the Include path .

- "\${workspace\_loc:/\${ProjName}/src/tracerecorder}"
- "\${workspace\_loc:/\${ProjName}/src/tracerecorder/config}"
- "\${workspace\_loc:/\${ProjName}/src/tracerecorder/include}"
- "\${workspace\_loc:/\${ProjName}/src/tracerecorder/streamports/Jlink\_RTT/include
  - } "

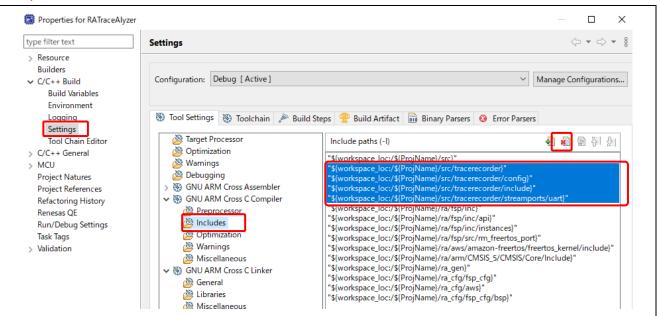


Figure 47. Settings Include paths of project properties



#### 5.3.5 Add code to hal\_entry.c

Add source code in red to hal\_entry.c.

```
#include "bsp_api.h"
#include "trcRecorder.h"
#include "FreeRTOS.h"
#include "semphr.h"
#include "hal_data.h"
void R_BSP_WarmStart(bsp_warm_start_event_t event);
****************//**
* This function is called at various points during the startup process. This implementation
uses the event that is
* called right before main() to set up the pins.
 *
* @param[in] event
                   Where at in the start up process the code is currently at
****
******
void R_BSP_WarmStart (bsp_warm_start_event_t event)
{
   if (BSP_WARM_START_RESET == event)
#if BSP_FEATURE_FLASH_LP_VERSION != 0
       /* Enable reading from data flash. */
      R_FACI_LP->DFLCTL = 1U;
      /* Would normally have to wait tDSTOP(6us) for data flash recovery. Placing the enable
here, before clock and
       * C runtime initialization, should negate the need for a delay since the initialization
will typically take more than 6us. \star/
#endif
   }
   if (BSP_WARM_START_POST_C == event)
   {
      /* C runtime environment and system clocks are setup. */
      /* Configure pins. */
      R_IOPORT_Open(&g_ioport_ctrl, &g_bsp_pin_cfg);
   }
   vTraceEnable(TRC INIT);
}
```

Figure 48. Change the J-Link RTT Streaming hal\_entry.c



### 5.3.6 Build the project

Right-click on the project and select Build Project.

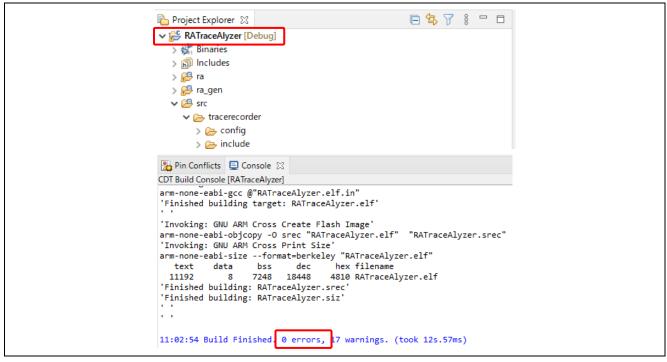


Figure 49 Build Project

# 5.4 Connect PC and EK-RA6M3 Board

The picture below shows the connection between the host PC and the EK-RA6M3 board.

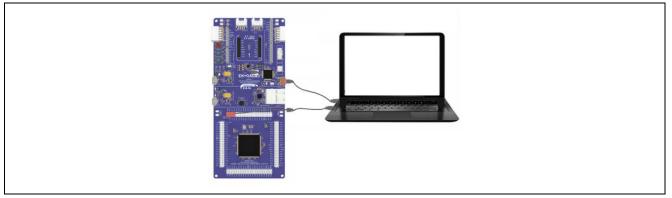


Figure 50. EK-RA6M3 Board Connection

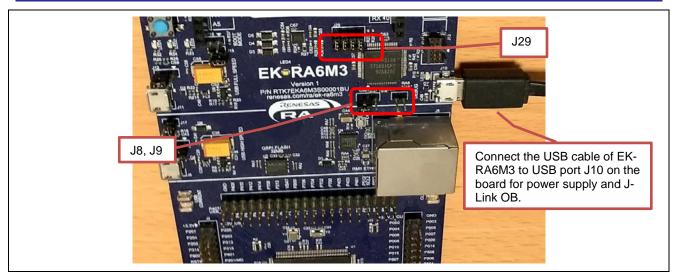
The hardware settings are as follows:

#### Table 3. Jumper Connection Summary for Different Debug Modes

Debug Modes	J8	J9	J29
Debug on-board	Jumper on pins 1-2	Open	Jumpers on pins 1-2, 3-4, 5-6, 7-8



# **Renesas RA Family**





# 5.5 Using the RTOS Resource View

Refer to section 4.5, Using the RTOS Resource View.

# 5.6 Start debugging a project with Tracealyzer®

#### 5.6.1 Launch debugger on e<sup>2</sup> studio

Select menu **Run > Debug** to launch the debugger.

#### 5.6.2 Launch Tracealyzer®

Launch installed Tracealyer 4 on PC.

#### 5.6.3 Set to Recording Settings

Refer to Figure 52 to get the RTT control block address in the map file, See code in red.

```
*(COMMON)

COMMON 0xlffe2fa4

0xa8 ./src/TraceRecorder/streamports/Jlink_RTT/SEGGER_RTT.o

0xlffe2fa4 _SEGGER_RTT
```





Click **Recording settings** on Tracealyzer, then select **J-Link Settings** and **PSF Streaming Settings** on Tracealyzer. Set the following.

- Select Target Device: **R7FA6M3AH**. (EK-RA6M3)
- Set RTT Control Block Address: 0x1FFE2FA4.See Error! Reference source not found.

O percepio"		
Welcome to Tracealyzer	Record a Trace	Settings - J-Link Settings - X
Percepto Traceetyzer is a powerful tool for tracing and systems. More than 25 weres offers anal.zing in sport tend for the step-by-step performance offers.canc To enable tracing in your target system, follow the step-by-step greed on the User Narasi. Percepto News Tracentyzer Version 4.4.2 is Out With Support for Azure RTOS Traced XMM Traces 2021-04-05_16.39.55 prf 2014405 fit 39.65.337.3KB trace-2021-04-05_16.39.55 prf 2014405 fit 39.65.337.3KB trace-2021-04-05_16.39.55 prf 2014405 fit 39.65.337.3KB trace-2021-04-05_16.39.55 prf 2014405 fit 39.65.337.3KB trace-2021-04-05_16.39.55 prf 2014405 fit 39.65.337.3KB trace-2021-04-05_16.29.24 tred Percepto News Traces Traces Traces Traces Percepto News Traces T		Enter text to filter controls  Filter text to filter controls  Global Settings  View Settings  Ocking Settings  API Settings  API Settings  Settings  Streaming Settings  Help OK Apply Cancel  Help OK Apply Cancel  Control Block Address: Dx1FFE2FA4
		et RTT Up Buffer Index: 1 et RTT Down Buffer Index: 1 eset Target On Connect arget Starts Tracing elp OK Apply Cancel

Figure 53. Set to Recording J-Link RTT Settings



### 5.6.4 Start Recording a Trace

Click Record Streaming Trace to start recording a trace.

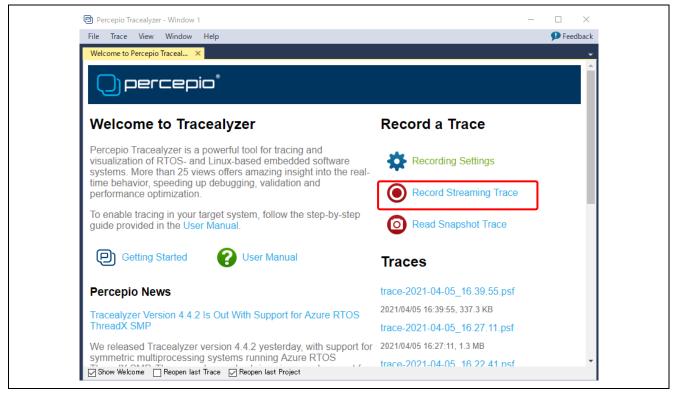


Figure 54. Start Record Streaming Trace



### 5.6.5 Trace information is displayed

Various analysis modes are provided. For more information, see the Help tab.

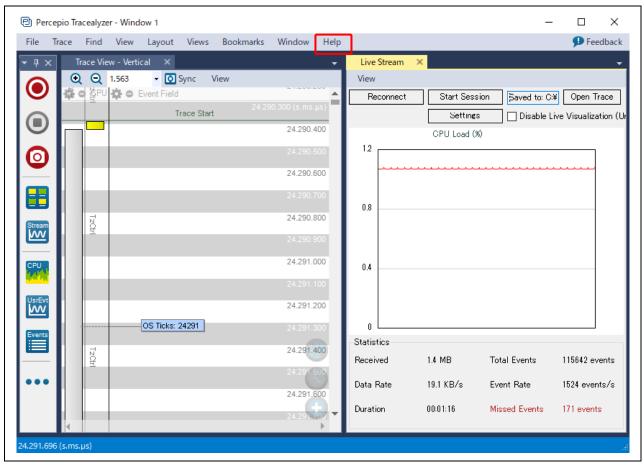


Figure 55. Displayed Trace information



### Website and Support

Visit the following URLs to learn about key elements of the RA family, download components, and related documentation, and get support.

RA Product Information RA Product Support Forum RA Flexible Software Package Renesas Support www.renesas.com/ra/forum www.renesas.com/FSP www.renesas.com/support



# **Revision History**

		Description	
Rev.	Date	Page	Summary
1.00	Jul.22.21		First release document



#### 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 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

- 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 and viewed applications are unimport. text and measurement and instant and ins

"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, 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.
- This document shall not be reprinted, reproduced or duplicated in any form, in whole or in part, without prior written consent of Renesas Electronics.
   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: <a href="http://www.renesas.com/contact/">www.renesas.com/contact/</a>.