RL78 Family

Example of Integration of SPI Mode Multimedia Card Driver into M3S-TFAT-Tiny Open-Source FAT File System

Introduction

This application note describes the integration method for enabling use of the M3S-TFAT-Tiny open-source FAT file system (referred to below as the TFAT library) and SPI mode multimedia card driver (referred to below as the MMC driver) in combination.

Target Devices

Target MCUs

MCUs compatible with both the M3S-TFAT-Tiny open-source FAT file system for the RL78 Family and the SPI mode multimedia card driver for the RL78 Family.

When using this application note with other Renesas MCUs, careful evaluation is recommended after making modifications to comply with the alternate MCU.

Contents

1. Specifications

Sample code is provided for memory driver interface functions to be included in the TFAT library, which enables storage media file operations.

The functionality provided is summarized below.

- Provides a memory driver interface function when using the MMC driver, enabling file operations.
- The TFAT library’s R_tfat_f_sync() (disk cache control) function is not supported.

2. Operation Confirmation Conditions

The operation of the sample code accompanying this application note has been confirmed under the following conditions.

Table 2.1 Operation Confirmation Conditions (CC-RL)

Item	Description
MCU	RL78/G23 (program ROM: 128 KB, RAM: 16 KB)
Operating frequency	Main system clock 32 MHz Peripheral module clock: 32 MHz
Operating voltage	3.3 V
Integrated development environment	Renesas Electronics, CS+ for CC V8.05.00
C compiler	Renesas Electronics, CC-RL V1.10.00
Board used	RL78G23-64p Fast Prototyping Board (RTK7RLG230CLG000BJ)
Device used	SanDisk microSD, 2 GB

Table 2.2 Operation Confirmation Conditions (LLVM)

Item	Description
MCU	RL78/G23 (program ROM: 768 KB, RAM: 48 KB)
Operating frequency	Main system clock 32 MHz Peripheral module clock: 32 MHz
Operating voltage	3.3 V
Integrated development environment	Renesas Electronics, e^2 studio 2022-07 (22.7.0)
C compiler	LLVM for Renesas RL78 10.0.0.202207 open-source compiler
Board used	RL78G23-64p Fast Prototyping Board (RTK7RLG230CLG000BJ)
Device used	Team microSD, 2 GB

3. Related Application Notes and User's Manuals

Application notes and user's manuals related to this application note are listed below. Refer to them in conjunction with this document.

- FAT File System (M3S-TFAT-Tiny) (R20UW0078EJ)
- RL78 Family Open Source FAT File System M3S-TFAT-Tiny: Introduction Guide (R20AN0159EJ)
- RL78 Family SPI mode MultiMediaCard Driver: Introduction Guide (R20AN0158EJ)

4. Software

4.1 Operation Overview

The memory driver interface functions of the TFAT library are modified for the MMC driver, enabling storage media file operations.

Figure 4.1 shows the configuration of the TFAT library and MMC driver.

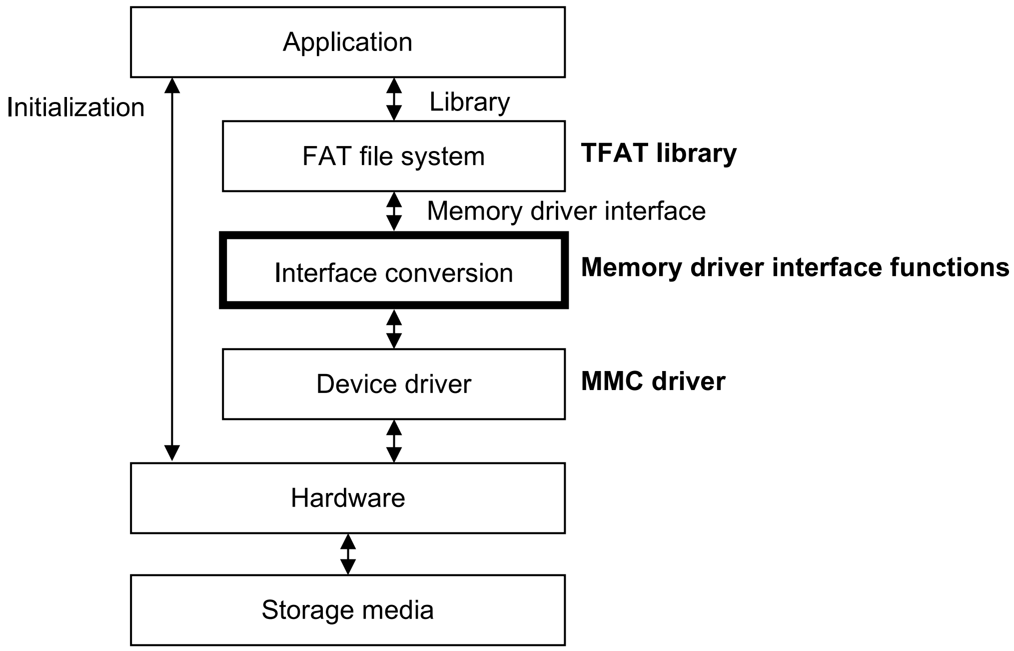

Figure 4.1 Configuration of TFAT Library and MMC Driver

4.2 Required Memory Sizes

4.2.1 CC-RL

Table 4.1 lists the required memory sizes.

Table 4.1 Required Memory Sizes (CC-RL)

Memory used	Size	Remarks
ROM	454 bytes	drv_if_sub.c r_tfat_drv_if.c r_target_io.c
RAM	20 bytes	drv_if_sub.c r_tfat_drv_if.c r_target_io.c
Maximum user stack used	82 bytes	
Maximum interrupt stack used	—	Interrupts not used

Notes: 1. The required memory sizes differ depending on the version of the C compiler and the compile options used.
2. The ROM and RAM sizes shown do not include the sizes of the TFAT library and MMC driver.
3. The stack size shown includes the stack RAM size of the MMC driver.

4.2.2 LLVM for Renesas RL78

Table 4.2 lists the required memory sizes.

Table 4.2 Required Memory Sizes (LLVM)

Memory used	Size	Remarks
ROM	489 bytes	drv_if_sub.c r_tfat_drv_if.c r_target_io.c
RAM	20 bytes	drv_if_sub.c r_tfat_drv_if.c r_target_io.c
Maximum user stack used	76 bytes	
Maximum interrupt stack used	—	Interrupts not used

Notes: 1. The required memory sizes differ depending on the version of the C compiler and the compile options used.
2. The ROM and RAM sizes shown do not include the sizes of the TFAT library and MMC driver.
3. The stack size shown includes the stack RAM size of the MMC driver.

4.3 File Structure

Table 4.3 lists the files used for the sample code. Note that files generated automatically by the integrated development environment are not listed.

Table 4.3 Files Used for Sample Code

\r01an1643xx0201-rl78-tfat-mmc				<DIR>	Sample code folder
	\doc			<DIR>	Document folder
		\en		<DIR>	English folder
			r01an1643ej0201-rl78-tfat-mmc.pdf		Application Note (This manual)
		\ja		<DIR>	Japanese folder
			r01an1643jj0201-rl78-tfat-mmc.pdf		Application Note
	\source			<DIR>	Program storage folder
		\drv_if		<DIR>	Memory driver interface folder
			drv_if_sub.c		Memory driver interface internal function file
			r_tfat_drv_if.c		Memory driver interface function file
		\mcu_io		<DIR>	Board setup interface folder
			r_target_io.c		Board setup interface source file

4.4 Functions

4.4.1 Memory Driver Interface Functions

Table 4.4 lists the memory driver interface functions.

Table 4.4 Memory Driver Interface Functions

Function Name	Description
R_tfat_disk_initialize()	Initializes disk drive
R_tfat_disk_read()	Reads from disk
R_tfat_disk_write()	Writes to disk
R_tfat_disk_ioctl()	Controls other drive
R_tfat_disk_status()	Gets disk drive status
R_tfat_get_fattime()	Gets date and time

Note: The R_tfat_outstream function (see TFAT library user’s manual) is required in order to use the R_tfat_f_forward function. Create it if necessary.

4.4.2 Memory Driver Interface Internal Functions

Table 4.5 lists the memory driver interface internal functions.

Table 4.5 Memory Driver Interface Internal Functions

Function Name	Description
R_card_insertion_chk()	Card insertion checking processing
R_init_card_detect_chat()	Initializes chattering counter

4.4.3 Board Setup Interface Function

Table 4.6 shows the board setup interface function.

Table 4.6 Board Setup Interface Function

Function Name	Description
storage_driver_init()	Initializes MMC driver

4.5 Function Specifications

Refer to the TFAT library user's manual.

4.5.1 Modifying Memory Driver Interface Functions

(1) Modifying Included Files

Configure settings in the MMC driver header file and shared functions header file as follows.

Approximately line 36 in r_tfat_drv_if.c

```
/******************************************************************************
Includes <System Includes> , "Project Includes"
******************************************************************************/
#include "r_cg_macrodriver.h"
#include "r_stdint.h"
#include "r_tfat_lib.h"
#include "r_mtl_com.h"
#include "r_mmc.h"
```

(2) Modifying R_tfat_disk_initialize()

This function executes MMC detection processing (by calling the R_mmc_Chk_Detect() function internally) and configures device initialization settings (by calling the R_mmc_Init_Slot() function).

Consequentially, it is necessary to perform MMC driver initialization processing (R_mmc_Init_Driver()) before calling the MMC driver API from the TFAT library.

Refer to the MMC driver user's manual for instructions on initializing the MMC driver.

Approximately line 77 in r_tfat_drv_if.c

```
DSTATUS R_tfat_disk_initialize(void)
{
/* Please put the code for disk_initalize driver interface function over here   */
/* Please refer the application note for details                                */
/* Please put the code for memory driver initialization, if any, over here      */
  int16_t ret_value;

  ret_value = R_card_insertion_chk(MMC_SLOT0);
  if (ret_value < MMC_OK)
  {
   R_init_card_detect_chat();
   return TFAT_STA_NODISK;
  }

  if (ret_value != MMC_TRUE)
  {
   return TFAT_STA_NODISK;
  }

  ret_value = R_mmc_Init_Slot(MMC_SLOT0);
  if (ret_value < MMC_OK)
  {
   R_init_card_detect_chat();
   return TFAT_STA_NOINIT;
  }

  ret_value = R_mmc_Get_MmcInfo(MMC_SLOT0, &card_info);
  if (ret_value < MMC_OK)
  {
   R_init_card_detect_chat();
   return TFAT_STA_NOINIT;
  }
  return  TFAT_RES_OK;
}
```

(3) Modifying R_tfat_disk_read()

Calls the MMC driver read function.

Approximately line 124 in r_tfat_drv_if.c

```
DRESULT R_tfat_disk_read(
    uint8_t Drive,           /* Physical drive number              */
    uint8_t* Buffer,         /* Pointer to the read data buffer    */
    uint32_t SectorNumber,   /* Start sector number                */
    uint8_t SectorCount      /* Number of sectors to read          */
)
{
  /*Please put the code for R_tfat_disk_read driver interface function over her
  */
  /*Please refer the application note for details                         */
  int16_t    Ret;

  Ret = R_mmc_Read_Data(0,SectorNumber,SectorCount,Buffer,MMC_MODE_NORMAL);
  if (Ret < MMC_OK)
  {
   return TFAT_RES_ERROR;
  }
  return TFAT_RES_OK;
}
```

(4) Modifying R_tfat_disk_write()

Calls the MMC driver write function.

Approximately line 154 in r_tfat_drv_if.c

```
DRESULT R_tfat_disk_write(
    uint8_t Drive,            /* Physical drive number        */
    const uint8_t* Buffer,    /* Pointer to the write data    */
    uint32_t SectorNumber,    /* Sector number to write       */
    uint8_t SectorCount       /* Number of sectors to write   */
)
{
  /*Please put the code for R_tfat_disk_write driver interface function over here    */
  /*Please refer the application note for details                                    */
  int16_t Ret;

  Ret = R_mmc_Write_Data(0,SectorNumber,SectorCount, (uint8_t *)Buffer,MMC_MODE
_NORMAL);
  if (Ret < MMC_OK)
  {
   return TFAT_RES_ERROR;
  }
  return TFAT_RES_OK;
}
```

(5) Modifying R_tfat_disk_ioctl()

To provide disk cache functionality, supply code for the appropriate processing.

Refer to the TFAT library user's manual for details.

The processing example shown below performs a normal end without doing anything. Therefore, do not use R_tfat_f_sync().

Approximately line 183 in r_tfat_drv_if.c

```
DRESULT R_tfat_disk_ioctl(
    uint8_t Drive,            /* Drive number           */
    uint8_t Command,          /* Control command code   */
    void* Buffer              /* Data transfer buffer   */
)
{
  /*Please put the code for R_tfat_disk_ioctl driver interface function over here   */
  /*Please refer the application note for details                                   */
  return TFAT_RES_OK;
}
```

(6) Modifying R_tfat_disk_status()

To provide the ability to get the disk drive status, supply code for the appropriate processing.

Refer to the TFAT library user's manual for details.

Approximately line 202 in r_tfat_drv_if.c

```
DSTATUS R_tfat_disk_status(
    uint8_t Drive             /* Physical drive number   */
)
{
  /*Please put the code for R_tfat_disk_status driver interface function over here  */
  /*Please refer the application note for details                                   */
  return TFAT_RES_OK;
}
```

(7) Modifying R_tfat_get_fattime()

This is not related to the MMC driver, so no description is provided.

4.5.2 Modifying Memory Driver Interface Internal Functions

(1) Modifying Included Files

Configure settings in the MMC driver header file and shared functions header file as follows.

Approximately line 1 in drv_if_sub.c

```
#include "r_cg_macrodriver.h"
#include <string.h>
#include "r_stdint.h"
#include "r_tfat_lib.h"
#include "r_data_file.h"
#include "r_board.h"
#include "r_mtl_com.h"
#include "r_mmc.h"
```

(2) Modifying Static Variable Definitions

Modify the index definitions of arrays defined as static.

Approximately line 16 in drv_if_sub.c

```
static uint16_t          gDetChatCnt[MMC_SLOT_NUM];
static uint16_t          gDetSts_Old[MMC_SLOT_NUM];
```

(3) Modifying R_card_insertion_chk()

Calls the MMC driver's insertion checking function.

Approximately line 27 in drv_if_sub.c

```
int16_t R_card_insertion_chk(uint8_t slot_num)
{
  uint8_t          DetSts;               /* Detection status                        */
  int16_t          api_ret;

  /* Check MMC insertion. */
  api_ret = R_mmc_Chk_Detect(slot_num,&DetSts);
  if (api_ret < MMC_OK)
  {
   return api_ret;
  }
  if (DetSts != gDetSts_Old[slot_num])     /* Status Changed!                        */
  {
   if (DetSts == MMC_TRUE)                 /* Removal -> Insertion                   */
   {
      DetChatCnt[slot_num]--;              /* Chattering counter decrement           */
      if (gDetChatCnt[slot_num] == 0)      /* counter =0                             */
      {
         gDetChatCnt[slot_num] = MMC_INS_CHAT;
         gDetSts_Old[slot_num] = DetSts;   /* Initialize MMC slot.                   */
      }
   }
   else
   {
      /* Insertion -> Removal     */
      /* Do not care chattering. */
      gDetChatCnt[slot_num] = MMC_INS_CHAT;
      gDetSts_Old[slot_num] = DetSts;
   }
  }
  else
  {
   /* No change    */
   gDetChatCnt[slot_num] = MMC_INS_CHAT;
  }
  return gDetSts_Old[slot_num];
}
```

(4) Modifying R_init_card_detect_chat()

Initializes the chattering counter of the target MMC.

Approximately line 73 in drv_if_sub.c

```
void R_init_card_detect_chat(void)  /* Initial process                         */
{
  uint8_t          slot_num;                /* Slot number                         */

  for (slot_num = MMC_SLOT0; slot_num < MMC_SLOT_NUM; slot_num++)
  {
    gDetChatCnt[slot_num] = MMC_INS_CHAT;   /* Reset the counter of chattering d
etection.*/
    gDetSts_Old[slot_num] = MMC_FALSE;      /* Set the previous status of detect
ion   to "removal".   */
  }
}
```

4.5.3 Modifying Board Setup Interface Function

(1) Modifying Included Files

Configure settings in the MMC driver header file and shared functions header file as follows.

Approximately line 35 in r_target_io.c

```
#include "r_cg_macrodriver.h"

#include "r_mtl_com.h"
#include "r_mmc.h"
#include "r_board.h"
```

(2) Modifying storage_driver_init()

Calls the MMC driver's initialization function.

Approximately line 54 in r_target_io.c

```
void storage_driver_init(void)
{
  R_mmc_Init_Driver();
```

5. Usage Notes

5.1 Included Header Files

In the application software include the MMC driver header file and shared functions header file.

```
#include "r_mtl_com.h"
#include "r_mmc.h"
```

5.2 Required Files when Integrating MMC Driver

The MMC driver is software for controlling the MMC.

This software is bundled with the following application note. It can be obtained from the Renesas Electronics website.

RL78 Family SPI mode MultiMediaCard Driver: Introduction Guide (R20AN0158EJ)

5.3 Usage Limitations

The code shown in (5) Modifying R_tfat_disk_ioctl() under 4.5.1, Modifying Memory Driver Interface Functions, does not implement memory driver interface processing with disk cache functionality. When using the code provided with this application note, do not call R_tfat_f_sync() under the TFAT library specification.

5.4 MMC Driver Initialization Procedure

Follow the steps below to initialize the MMC driver and the device.

1. Running the storage_driver_init() function
   Run the MMC driver's driver initialization function (R_mmc_Init_Driver()).
2. Running the R_tfat_disk_initialize() function
   Run the MMC detection and device initialization function (R_mmc_Init_Slot()).

5.5 Confirmation of Operation

The operation of the sample code accompanying the following application note has been confirmed.

RL78/G14 Sound Playback/Compression Demonstration for RL78/G14 CPU Board (R20AN0194EJ)

Revision History

Rev.	Date	Description	
		Page	**Summary**
1.00	Mar. 29, 2013	—	First edition issued
2.00	Jul. 14, 2021	—	Added support for RL78/G23 Changed memory card driver from “SPI mode MMC/SD card driver” to “SPI mode multimedia card driver”
2.01	Nov. 09, 2022	—	Added support for LLVM
		5	Content for LLVM added to 2. Operation Confirmation Conditions
		8	Content for LLVM added to 4.2 Required Memory Sizes

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

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/.