RL78 Family

SPI mode MultiMediaCard Driver: Introduction Guide

Introduction

This manual shows the software configuration of MultiMediaCard device driver for the RL78 family and how to use it.

And, we prepared Sound Playback/Compression demonstration software for the YRDKRL78G14 as sample application program for this driver software.

Please refer to the following URL for details.

RL78/G14 Sound Playback/Compression Demonstration for RL78/G14 CPU Board - Sample Code | Renesas
(Document No.: R20AN0194)

Target Device

RL78/G13, RL78/G14, RL78/G23

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

Contents

1. Summary

1.1 Purpose

The purpose is to provide an interface that connects RL78 family MCU to MultiMediaCard (hereafter referred to as "MMC") in SPI mode.

This manual provides information to create the application.

1.2 Function Description

This device driver (hereafter referred to as "MMC driver") is software that enables communication with MMC by RL78 family. This software achieves accessing to MMC using SPI mode in 3-lines serial array unit that is RL78 Family peripherals.

MMC driver

- Reference MMCA System Specifications; Version 3.2
- This is only used in MMC SPI mode
- This is a block type device driver that defines one sector as 512Byte.
  The commands of READ_MULTIPLE_BLOCK and WRITE_MULTIPLE_BLOCK are used.
  As for cards that not support aforesaid two MULTIPLE_BLOCK commands, it is operated by commands of READ_SINGLE_BLOCK and WRITE_SINGLE_BLOCK.
- It supports multiple devices controlled by CS signals.
- It is independent of OS.
- MMC driver in this manual: SPI mode MultiMediaCard Driver for RL78 Family V.2.01 Release 00

1.3 File configuration

Table 1 MMC driver product files

Directory Configuration <Directory name> ,File name	Reference
\doc <DIR>	**Document Directory**
\en <DIR>	**English Directory**
r20an0158ej0201-rl78-mmc.pdf	Application Note (This manual)
\ja <DIR>	**Japanese Directory**
r20an0158jj0201-rl78-mmc.pdf	Application Note
\mmc_driver <DIR>	**MMC Driver Source Program Directory**
\com <DIR>	**Common Function Directory**
r_mtl_com.c	Common Function (logging function)
r_mtl_com2.h	Common Header file
r_mtl_endi.c	Common Function (endian function)
r_mtl_mem.c	Common Function (standard library function)
r_mtl_str.c	Common Function (standard library function)
r_mtl_tim.c r_mtl_tim.h	Common Function (software loop timer)
r_stdint.h	Integer type definitions header file
\ rl78_32MHz_CCRL <DIR>	**Definitions Directory for 32MHz Operation (CCRL)**
r_mtl_com.h	Definitions Header file
\ rl78_24MHz_CCRL <DIR>	**Definitions Directory for 24MHz Operation (CCRL)**
r_mtl_com.h	Definitions Header file
\ rl78_20MHz_CCRL <DIR>	**Definitions Directory for 20MHz Operation** (CCRL)
r_mtl_com.h	Definitions Header file
\ rl78_32MHz_IAR <DIR>	**Definitions Directory for 32MHz Operation (IAR)**
r_mtl_com.h	Definitions Header file
\ rl78_24MHz_IAR <DIR>	**Definitions Directory for 24MHz Operation (IAR)**
r_mtl_com.h	Definitions Header file
\ rl78_20MHz_IAR <DIR>	**Definitions Directory for 20MHz Operation (IAR)**
r_mtl_com.h	Definitions Header file
\ rl78_32MHz_LLVM <DIR>	**Definitions Directory for 32MHz Operation (LLVM)**
r_mtl_com.h	Definitions Header file

Directory Configuration <Directory name> ,File name	**Reference**
\ mmc <DIR>	**Directory of Device Driver for MMC**
r_mmc.h	Common Header File
r_mmc_io.c r_mmc_io.h	I/O Module for SPI Mode
r_mmc_mmc.c	MMC Module for SPI Mode
r_mmc_sub.c r_mmc_sub.h	Sub Module for SPI Mode
r_mmc_usr.c	API for SPI Mode
r_mmc_mcu_pragma.h	Header file for pragma define MMC driver
\ rl78 <DIR>	**Directory of RL78 Family**
r_mmc_csi.c	MMC driver SPI mode Communication module using CSI
rl78g14_csi_ccrl <DIR>	**SFR definition directory for RL78/G14 group CSI (CC**RL)
r_mmc_sfr.h.	Individual definitions of SFR header file
r_mmc_user_config.h	User definitions header file
rl78g14_csi_iar <DIR>	**SFR definition directory for RL78/G14 group CSI (IAR)**
r_mmc_sfr.h.	Individual definitions of SFR header file
r_mmc_user_config.h	User definitions header file
rl78g23_csi_ccrl <DIR>	**SFR definition directory for RL78/G23 group CSI (CCRL)**
r_mmc_sfr.h.	Individual definitions of SFR header file
r_mmc_user_config.h	User definitions header file
rl78g23_csi_iar <DIR>	**SFR definition directory for RL78/G23 group CSI (IAR)**
r_mmc_sfr.h.	Individual definitions of SFR header file
r_mmc_user_config.h	User definitions header file
rl78g23_csi_llvm <DIR>	**SFR definition directory for RL78/G23 group CSI (LLVM)**
r_mmc_sfr.h.	Individual definitions of SFR header file
r_mmc_user_config.h	User definitions header file

2. Program type definitions

This section gives the details about the type definitions used in the program.

DataType	Typedef
unsigned char	uint8_t
unsigned short	uint16_t
unsigned long	uint32_t
signed char	int8_t
signed short	int16_t
signed long	int32_t
int16_t	natural_int_t
uint16_t	natural_uint_t

3. Device Driver

3.1 Device driver function details

Initialization function

Function Name	Function description
R_mmc_Init_Driver ()	Slot initialization process

Function of device control

Function Name	Function description
R_mmc_Init_Slot()	Slot initialization process
R_mmc_Detach()	Slot stop process
R_mmc_Chk_Detect()	Insertion check process

Data access control function

Function Name	Function description
R_mmc_Read_Data()	Data reading process
R_mmc_Write_Data()	Data writing process
R_mmc_Get_MmcInfo()	MMC information obtaining process

Command list of internal use

Command index	Name of command
CMD0	GO_IDLE_STATE
CMD1	SEND_OP_COND
CMD9	SEND_CSD
CMD10	SEND_CID
CMD12	STOP_TRANSMISSION
CMD13	SEND_STATUS
CMD17	READ_SINGLE_BLOCK
CMD18	READ_MULTIPLE_BLOCK
CMD24	WRITE_BLOCK
CMD25	WRITE_MULTIPLE_BLOCK
CMD58	READ_OCR
CMD59	CRC_ON_OFF

Note: User needs to respond to unsupported commands

3.2 Function details

3.2.1 Initialization process of driver (R_mmc_Init_Driver)

clause	detail
Function Name	void R_mmc_Init_Driver(void)
Argument	None
Function	Initialize driver. Initialize SFR for card control. The following process is done in every slot. (1) Open card control port. (2) Initialize card control RAM. Execute once when the system starts up.
Return Value	None

3.2.2 Initialization of card slot (R_mmc_Init_Slot)

clause	detail
Function Name	int16_t R_mmc_Init_Slot(uint8_t SlotNo)
Argument	uint8_t SlotNo : Slot number
Function	Initialize card slot. Initialize card control RAM. Initialization of card. Execute when card insertion is detected.
Return Value	Returns initialization result. MMC_OK : Successful operation MMC_ERR_PARAM : Parameter error MMC_ERR_HARD : Hardware error MMC_ERR_CRC : CRC error MMC_ERR_IDEL : Idle state error MMC_ERR_OTHER : Other error

3.2.3 Card slot stop process (R_mmc_Detach)

clause	detail
Function Name	int16_t R_mmc_Detach(uint8_t SlotNo)
Argument	uint8_t SlotNo : Slot number
Function	Process when removing card from designated slot. -Initialize card control SFR. -Open card control port. -Initialize card control RAM. Execute when card removal is detected.
Return Value	Returns removal result. MMC_OK : Successful operation MMC_ERR_PARAM : Parameter error

3.2.4 Card insertion checking process (R_mmc_Chk_Detect)

clause	detail
Function Name	int16_t R_mmc_Chk_Detect(uint8_t SlotNo, uint8_t* pDetSts)
Argument	uint8_t SlotNo : Slot number uint8_t *pDetSts : Buffer pointer for card insertion condition
Function	Check the condition of card being inserted. If returns "MMC_OK", The port status of card detecting will be in buffer 'pDetSts'. — MMC_TRUE :The port status of card detecting is active — MMC_FALSE:The port status of card detecting Non is non-active Cannot remove chattering in this process. Remove chattering in upper system if needed. Recommend confirming card insertion by periodic polling.
Return Value	Returns the check result. MMC_OK : Successful operation MMC_ERR_PARAM : Parameter error

3.2.5 Data reading process (R_mmc_Read_Data)

clause	detail
Function Name	int16_t R_mmc_Read_Data(uint8_t SlotNo, uint32_t BlkNo, uint32_t BlkCnt, uint8_t *pData, uint8_t Mode)
Argument	uint8_t SlotNo : Slot number uint32_t BlkNo : Block number to start readout uint32_t BlkCnt : Number of readout blocks uint8_t *pData : Pointer to the area where the data which is read must be stored uint8_t Mode : Transfer mode of reading data
Function	Readout the data from card by block (512byte) Readout the data in the designated number of blocks from the designated block. Choose MMC_MODE_NORMAL(transfers data to the designated buffer 'pData'.) in "Mode". The readout from MMC is possible among MMC information handed from R_mmc_Get_MmcInfo() function only when card classification (MmcInfo.Card) is not 'MMC_CARD_UNDETECT'. Maximum block number is 'pMmcInfo.MaxBlkNum' from the "R_mmc_Get_MmcInfo()" function. Maximum number of blocks is 'pMmcInfo.MaxBlkNum' +1.
Return Value	Returns the result of reading. MMC_OK : Successful operation MMC_ERR_PARAM : Parameter error MMC_ERR_HARD : Hardware error MMC_ERR_CRC : CRC error MMC_ERR_OTHER : Other error

3.2.6 Data writing process (R_mmc_Write_Data)

clause	detail	
Function Name	int16_t R_mmc_Write_Data(uint8_t SlotNo, uint32_t BlkNo, uint32_t BlkCnt, uint8_t *pData, uint8_t Mode)	
Argument	uint8_t SlotNo	: Slot number
	uint32_t BlkNo	: Block number to start writing
	uint32_t BlkCnt	: Number of writing blocks
	uint8_t *pData	: Pointer to the area where the data which is written must be stored
	uint8_t Mode	: Transfer mode of writing data
Function	Write the data to card by block (512byte). Write the data in the designated number of blocks to the designated block. Choose MMC_MODE_NORMAL (This is a mode that transfers data from the designated buffer 'pData') in "Mode". The transfers to MMC is possible among MMC information handed from R_mmc_Get_MmcInfo() function only when card classification (MmcInfo.Card) is not 'MMC_CARD_UNDETECT'. Maximum block number is 'pMmcInfo.MaxBlkNum' from the "R_mmc_Get_MmcInfo()" function. Maximum number of blocks is 'pMmcInfo.MaxBlkNum' +1.	
Return Value	Returns the result of writing.	
	MMC_OK	: Successful operation
	MMC_ERR_PARAM	: Parameter error
	MMC_ERR_HARD	: Hardware error
	MMC_ERR_WP	: Write-protection error
	MMC_ERR_OTHER	: Other error

3.2.7 Card information obtaining process (R_mmc_Get_MmcInfo)

clause	detail
Function Name	int16_t R_mmc_Get_MmcInfo(uint8_t SlotNo, MMC_INFO* pMmcInfo)
Argument	uint8_t SlotNo : Slot number MMC_INFO *pMmcInfo : Buffer pointer for card information
Function	It returns MMC information. The buffer 'pMmcInfo' holds card information. • pMmcInfo.Card : Card types — MMC_CARD_UNDETECT : Card not detected — MMC_CARD_MMC : MMC — MMC_CARD_OTHER : Other card • pMmcInfo.WProtect : Write-protection status — MMC_NO_PROTECT : Write-protection cancel — bit1: MMC_W_PROTECT_SOFT : Software write-protection pMmcInfo.MemSize : Card capacity(byte) pMmcInfo.MaxBlkNum : Maximum block number of the media When 'pMmcInfo.MemSize' is 0xFFFFFFFF, 'pMmcInfo.MaxBlkNum' +1 inidicates the number of the media and the size is ('pMmcInfo.MaxBlkNum'+1)*512.
Return Value	Returns the result of obtaining card information. MMC_OK : Successful operation MMC_ERR_PARAM : Parameter error MMC_ERR_OTHER : Other error

3.3 Data Structure

Data structure is showed as follow.

Definition of Card Information Data Structure

```
typedef struct {
  uint8_t    Card;             /* Card type                      */
  uint8_t    WProtect;         /* Write-protection status        */
  uint32_t   MemSize;          /* Card capacity                  */
  uint32_t   MaxBlkNum;        /* The number of the max blocks   */
} MMC_INFO;
```

3.4 Definitions

Definitions are showed as follow.

```
/*----------- Definitions of return value ------------*/
#define MMC_OK              (int16_t)( 0)        /* Successful operation     */
#define MMC_ERR_PARAM       (int16_t)(-1)        /* Parameter error          */
#define MMC_ERR_HARD        (int16_t)(-2)        /* Hardware error           */
#define MMC_ERR_CRC         (int16_t)(-3)        /* CRC error                */
#define MMC_ERR_WP          (int16_t)(-4)        /* Write-protection error   */
#define MMC_ERR_MBLKCMD     (int16_t)(-5)        /* Multi-block command error*/
#define MMC_ERR_IDLE        (int16_t)(-6)        /* Idle state error         */
#define MMC_ERR_OTHER       (int16_t)(-7)        /* Other error              */

/*--------------- Definitions of flag ----------------*/
#define MMC_TRUE            (uint8_t)0x01        /* Flag "ON"                */
#define MMC_FALSE           (uint8_t)0x00        /* Flag "OFF"               */

/*-------------- Definition of card type --------------*/
#define MMC_CARD_UNDETECT   (uint8_t)0x00        /* Card is not found        */
#define MMC_CARD_MMC        (uint8_t)0x01        /* MMC                      */
#define MMC_CARD_OTHER      (uint8_t)0xFF        /* Other card               */

/*------ Definitions of write-protection status ------*/
#define MMC_NO_PROTECT      (uint8_t)0x00        /* None setting             */
#define MMC_W_PROTECT_SOFT  (uint8_t)0x02        /* Software write-protection*/
```

4. Setup Examples

The following example is for the usage of RL78/G14, RL78/G23. Please refer to the section 4.3 if you need the information about porting to RL78/G13.

4.1 r_mtl_XXX : Variable Data Setup Example

This section is for setting the resources of each user system

The setting should be made in the [/**SET**/] comment of each file.

An excerpt of each file is provided with detailed comments.

4.1.1 r_mtl_com.h

This file is a common header file.

r_mtl_com.h is prepared for each MCU and system clock settings.

Please select using directory for your environment.

If these are not suitable for your environment, please make directory and setting files for your environment.

MCU - System clock	Include Directory (source directory)
RL78/G14, RL78/G23 – 32MHz (CCRL)	\com\ rl78_32MHz_CCRL
RL78/G14, RL78/G23 – 24MHz (CCRL)	\com\ rl78_24MHz_CCRL
RL78/G14, RL78/G23 – 20MHz (CCRL)	\com\ rl78_20MHz_CCRL
RL78/G14, RL78/G23 – 32MHz (IAR)	\com\ rl78_32MHz_IAR
RL78/G14, RL78/G23 – 24MHz (IAR)	\com\ rl78_24MHz_IAR
RL78/G14, RL78/G23 – 20MHz (IAR)	\com\ rl78_20MHz_IAR
RL78/G23 – 32MHz (LLVM)	\com\ rl78_32MHz_LLVM

(1) Define the software loop timer

— When using the loop timer, include following header file.
The loop timer process is used for waiting duration of device driver.
The following is an example of the setting when using the software loop timer.
And please define the macro that is suitable for your system clock in r_mtl_tim.h. In case, running RL78 in 32MHz, "MTL_TIM_RL78__32MHz_noWait" should be defined.

```
/* When not using the loop timer, put the following 'include' as comments.  */
#define MTL_TIM_RL78__32MHz_noWait

#include "r_mtl_tim.h"
```

(2) Define Endian type

— Little endian should be selected in RL78 Family

```
#define MTL_MCU_LITTLE              /* Little Endian           */ /** SET **/
```

(3) Specify type of user standard library

— Specify the type of standard library in the user system.

When using the library bundled with the compiler for the processes stated below, add the listed define definitions as comments.

When using the optimized library, define the optimized library.

```
/* Specify the type of user standard library.                              *//** SET **/
/* When using the compiler-bundled library for the following processes,    *//** SET **/
/* put the following 'define' as comments.                                 *//** SET **/
/* memcmp()/memmove()/memcpy()/memset()/strcat()/strcmp()/strcpy()/strlen()*//** SET**/
#define MTL_USER_LIB                /* use optimized library              */ /** SET**/
```

4.1.2 r_mtl_tim.h

When including r_mtl_tim.h, it is enable.

The value depends on clock frequency and wait of MCU.

Set the software timer to be used for internal operations.

If there are no "define" that is suitable for user system, user has to make own "define".

The following count value is actual measurement value.

```
/* Define the counter value for the timer.                                 */
/* Specify according to the user MCU, clock and wait requirements.         */
/*                                                                         */
/* Set the reference value to 10% more than the actual calculated value.   */
/*=========================================================================*/

/*=========================================================================*/
#ifdef MTL_TIM_RL78__32MHz_noWait
/* Setting for 32.0MHz no wait */
#define MTL_T_1US       3               /* loop Number of 1us   */ /** SET **/
#define MTL_T_2US       8               /* loop Number of 2us   */ /** SET **/
#define MTL_T_4US      17               /* loop Number of 4us   */ /** SET **/
#define MTL_T_5US      21               /* loop Number of 5us   */ /** SET **/
#define MTL_T_10US     44               /* loop Number of 10us  */ /** SET **/
#define MTL_T_20US     90               /* loop Number of 20us  */ /** SET **/
#define MTL_T_30US    136               /* loop Number of 30us  */ /** SET **/
#define MTL_T_50US    227               /* loop Number of 50us  */ /** SET **/
#define MTL_T_100US   456               /* loop Number of 100us */ /** SET **/
#define MTL_T_200US   913               /* loop Number of 200us */ /** SET **/
#define MTL_T_300US  1370               /* loop Number of 300us */ /** SET **/
#define MTL_T_400US  1827               /* loop Number of 400us */ /** SET **/
#define MTL_T_1MS    4572               /* loop Number of 1ms   */ /** SET **/
#endif

#ifdef MTL_TIM_RL78__24MHz_noWait
/* Setting for 24.0MHz no wait */
(omit)
#endif

#ifdef MTL_TIM_RL78__20MHz_noWait
/* Setting for 20.0MHz no wait */
(omit)
#endif
```

4.2 MMC Driver : Variable Data Setup Example

This section is for setting the resources of each user system.

The setting should be made in the [/**SET**/] comment of each file.

An excerpt of each file is provided with detailed comments.

4.2.1 r_mmc.h (Driver common definitions header file)

(1) Define number of slots (devices) and slot number

— Specify number of slots (devices) and slot number.

```
/*  Define number of required card slots. (1-N slots)                          */
/*  Define slot number in accordance with the number of card slots to be connected.  */
/*-----------------------------------------------------------------------------*/
/* Define number of slots (devices). */
#define MMC_SLOT_NUM       1                   /* 1slots              */ /** SET **/

/* Define slot number. */
#define MMC_SLOT0             0                /* Slot 0              */ /** SET **/
#define MMC_SLOT1             1                /* Slot 1              */ /** SET **/
```

(2) Define use of single block commands with SPI mode

— Do not make any changes.

```
/* When use the card which does not support a multi-block command, please define it.      */
/* Use single block commands in the case of the card which does not support multiple block */
/* commands.                                                                              */
#define MMC_SBLK_CMD          /* Support single block commands     */  /** SET **/
```

(3) Define card type

— Define MMC_SUPPORT_MMC.

```
/*-----------------------------------------------------------------------*/
/*  Please define the media to support.                                  */
/*-----------------------------------------------------------------------*/
#define MMC_SUPPORT_MMC                    /* MMC             */ /** SET **/
```

4.2.2 r_mmc_user_config.h (MCU individual definitions header file)

(1) Selecting MUC

r_mmc_user_config.h is prepared for each MCU. Please include r_mmc_user_config.h in directory that is suitable for user system.

MCU – Communication Module	Include Directory (source directory)
RL78G14 – CSI (CCRL)	\mmc\rl78\rl78g14_csi_ccrl
RL78G14 – CSI (IAR)	\mmc\rl78\rl78g14_csi_iar
RL78G23 – CSI (CCRL)	\mmc\rl78\rl78g23_csi_ccrl
RL78G23 – CSI (IAR)	\mmc\rl78\rl78g23_csi_iar
RL78G23 – CSI (LLVM)	\mmc\rl78\rl78g23_csi_llvm

(2) “define” for channel number of communication unit

Please define MMC_SAU_UNIT for communication unit that user uses.

Please define MMC_SAU_CHANNEL for communication unit channel number.

In case, using SAU0-channel CSI00 or channel CSI10, user has to select using pins. Please select MMC_CSI_PIN for pins that user uses.

```
/* Serial Array Unit(SAU) Select ( 0 or 1 )*/
#define MMC_SAU_UNIT            1                   /** SET **/
/* SAU Channel Select ( 0 or 1 or 2 or 3 ) */
#define MMC_SAU_CHANNEL         1                   /** SET **/
/* CSI PIN select  ( 'A' or 'B' ) */
#define MMC_CSI_PIN             'A'                 /** SET **/

/*
                                ||         Select Port          |
-----------+-----------+----------++-------------------------------+
  MMC_SAU_ |  MMC_SAU_ | MMC_CSI_ ||    SI    |   SCK    |    SO    |
  UNIT     |  CHANNEL  | PIN      ||  Select  |  Select  |  Select  |
   Value   |   Value   |  Value   ||   port   |   port   |   port   |
===========+===========+==========++=========+==========+==========+
     0     |     0     |    'A'   ||   P50    |   P30    |   P51    |
(=Use SAU0)|           |----------++---------+----------+----------+
           |(=Use CSI00)|   'B'   ||   P16    |   P55    |   P17    |
           +-----------+----------++---------+----------+----------+
           |     1     | (invalid) ||   P74    |   P75    |   P73    |
           |(=Use CSI01)|          ||         |          |          |
           +-----------+----------++---------+----------+----------+
           |     2     |    'A'   ||   P03    |   P04    |   P02    |
           |           |----------++---------+----------+----------+
           |(=Use CSI10)|   'B'   ||   P81    |   P80    |   P82    |
          -+-----------+----------++---------+----------+----------+
           |     3     | (invalid) ||   P11    |   P10    |   P12    |
           |(=Use CSI11)|          ||         |          |          |
-----------+-----------+----------++---------+----------+----------+
     1     |     0     | (invalid) ||   P14    |   P15    |   P13    |
(=Use SAU1)|(=Use CSI20)|          ||         |          |          |
           +-----------+----------++---------+----------+----------+
           |     1     | (invalid) ||   P71    |   P70    |   P72    |
           |(=Use CSI21)|          ||         |          |          |
           +-----------+----------++---------+----------+----------+
           |     2     | (invalid) ||   P143   |   P142   |   P144   |
           |(=Use CSI30)|          ||         |          |          |
           +-----------+----------++---------+----------+----------+
           |     3     | (invalid) ||   P53    |   P54    |   P52    |
           |(=Use CSI31)|          ||         |          |          |
-----------+-----------+----------++---------+----------+----------+*/
```

(3) Define control ports

— Please define the macro for DETECT(detecting card insertion) pins or CS(card select) pins suitable for user's circuit.

— When DETECT pin is not connect, comment out MMC_DETECT0_CONNECTION macro.

```
/*-----------------------------------------------------------------------*/
/*  Define the control port.                                             */
/*-----------------------------------------------------------------------*/
#define MMC_CS0_PORTNO          0             /* CS0 Port No. */  /** SET **/
#define MMC_CS0_BITNO           5             /* CS0 Bit  No. */  /** SET **/

/* Please define the MMC_DETECT0_CONNECTION macro when the MMC slot Card
detect pin connect to MCU. */
#define MMC_DETECT0_CONNECTION       /* DETECT0 Port Connection */ /** SET **/

#if defined(MMC_DETECT0_CONNECTION)
#define MMC_DETECT0_PORTNO      0          /* DETECT0 Port No. */ /** SET **/
#define MMC_DETECT0_BITNO       0          /* DETECT0 Bit  No. */ /** SET **/
#endif /* #if defined(MMC_DETECT0_CONNECTION) */


#if (MMC_SLOT_NUM > 1)

#define MMC_CS1_PORTNO                       /* CS1 Port No. */  /** SET **/
#define MMC_CS1_BITNO                        /* CS1 Bit  No. */  /** SET **/


#define MMC_DETECT1_CONNECTION       /* DETECT1 Port Connection */ /** SET **/

#if defined(MMC_DETECT1_CONNECTION)
#define MMC_DETECT1_PORTNO                /* DETECT1 Port No. */ /** SET **/
#define MMC_DETECT1_BITNO                 /* DETECT1 Bit  No. */ /** SET **/
#endif /* #if defined(MMC_DETECT0_CONNECTION) */
```

(4) Definition of detecting communication timeout

— This macro can omit detecting timeout during communication.
If user omits detecting timeout, please define “MMC_NOCHK_TIMEOUT”. If this macro is defined, processing speed would be increased, but there is a possibility program stops when abnormal communication status occurs. If user does not omit this macro definition, please set time to this macro.

- Setting time unit using MMC_T_CSI_WAIT macro. Please select setting macro from r_mtl_tim.h.
- Please define transmit timeout time using MMC_CSI_TX_WAIT macro.
- Please define reception timeout time using MMC_CSI_RX_WAIT macro.
- Setting values of each timeout time macro are [timeout time/unit].

```
/*-------------------------------------------------------------------------*/
/* Macro "MMC_NOCHK_TIMEOUT" omits detecting timeout during communication.  */
/* If user omits detecting timeout, please define this macro.              */
/* If this macro is defined, processing speed would be increased.          */
/*-------------------------------------------------------------------------*/
#define MMC_NOCHK_TIMEOUT  /* No Check Communication Timeout    */ /** SET **/

/*-------------------------------------------------------------------------*/
/* If MMC_NOCHK_TIMEOUT would be not defined, please set timeout time.     */
/* MMC_T_CSI_WAIT is unit of measuring timeout.                            */
/* Please select value from "r_mtl_tim.h"                                  */
/* Please set value of (timeout time/unit) to MMC_CSI_TX_WAIT(transmitting) */
/* and MMC_CSI_RX_WAIT(receiving).                                         */
/*-------------------------------------------------------------------------*/
/* CSI transmit&receive completion waiting polling time   */
#define MMC_T_CSI_WAIT   (natural_uint_t)MTL_T_1US                 /** SET **/

/* CSI transmission completion waiting time 50000 * 1us = 50ms */
#define MMC_CSI_TX_WAIT  (natural_uint_t)50000                     /** SET **/
/* CSI receive completion waiting time  50000 * 1us = 50ms */
#define MMC_CSI_RX_WAIT  (natural_uint_t)50000                     /** SET **/
```

(5) Define resources

— The data transfer depends on MCU resource for use with.
Select one of the following for use as your system.

```
/*-------------------------------------------------------------------------------*/
/*  Define the combination of the MCU's resources.                               */
/*-------------------------------------------------------------------------------*/
//#define MMC_OPTION_1   /* CSI                          */ /** SET **/
#define MMC_OPTION_2   /* CSI + CRC calculation circuit  */ /** SET **/
```

(6) Define bit rate

— As for transfer speed setting, it is necessary to meet tODLY of both Identification mode and Data Transfer mode in spec.
In addition, it is necessary to meet tOD (100kHz <= tOD <= 400kHz) at Identification mode and tPP (0.1MHz <= tPP <= 20MHz (*)) at Data Transfer mode.
The frequency of tOD and tPP mean the frequency of SCLK in this device driver.
As for maximum frequency, make a confirmation of each MCU datasheet.

- MMC_FCLK_PRESCALER_SELECT macro
  This macro sets fMCK using fCLK(source of clock). Setting value is reflected to PRSm3 - PRSm0 bit.
  Please set the value “0-15” to MMC_FCLK_PRESCALER_SELECT macro.
- MMC_OPERATION_CLK_SELECT macro
  This macro selects fMCK for each channel (max 4 channel) of SAU(Serial Array Unit). Setting value is reflected to CKSmn bit in SMSm register.
  Please sets any values (0 or 1) to MMC_OPERATION_CLK_SELEC macro.
- MMC_UBRG_IDENTIFICATION macro
  This macro sets transfer clock when user uses “Identification mode”. Setting value is reflected to upper 7bit in SDRmn register, and becomes transfer clock (fTCLK) from divided fMCK. Please set “0-63” to MMC_UBRG_IDENTIFICATION macro.

- MMC_UBRG_D_TRANSFER macro
  This macro sets transfer clock when user uses “Transfer mode”. Setting value is reflected to upper 7bit in SDRmn register, and becomes transfer clock (fTCLK) from divided fMCK. Please set “0-63” to MMC_UBRG_D_TRANSFER macro.

- MMC_CLK_D_TRANSFER macro
  Please set frequency of “Transfer mode” to MMC_CLK_D_TRANSFER macro. The setting value is used for checking “NAC Cycles”.

— This section shows example of setting for communication baud rate

- fCLK=32MHz(HOCO) , (fMCK=16MHz,) fTCLK=8MHz
  Please define baud rate in r_mmc_user_config.h. Example is below.

```
/*-----------------------------------------------------------------------------------------*/
/* Define the value of the bit rate register according to a communication baud rate.       */
 (omit)
/* fCLK = 32MHz , fMCK = 16MHz , fTCLK = 8MHz */
#define MMC_FCLK_PRESCALER_SELECT 1              /* SPSm.PRSmk[3:0]                */  /** SET **/

#define MMC_OPERATION_CLK_SELECT  0              /* select SMRm.CKmX 0:CKm0  1:CKm1*/  /** SET **/

#define MMC_UBRG_IDENTIFICATION  (uint8_t)19     /* BRR identification mode setting*/  /** SET **/
/*                                        ++----------------- 400KHz               */  /** SET **/
#define MMC_UBRG_D_TRANSFER      (uint8_t)0      /* BRR data Transfer mode setting */  /** SET **/
/*                                        ++----------------- 8.0MHz               */  /** SET **/
#define MMC_CLK_D_TRANSFER  (uint32_t)8000000 /* Data Transfer mode clock frequency */  /** SET **/
```

- fCLK=24MHz(HOCO) , (fMCK=24MHz,) fTCLK=12MHz
  Please define baud rate in r_mmc_user_config.h. Example is below.

```
/*-----------------------------------------------------------------------------------------*/
/* Define the value of the bit rate register according to a communication baud rate.      */
 (omit)
/* fCLK = 24MHz , fMCK = 24MHz , fTCLK = 12MHz */
#define MMC_FCLK_PRESCALER_SELECT 0             /* SPSm.PRSmk[3:0]                */  /** SET **/

#define MMC_OPERATION_CLK_SELECT  0             /* select SMRm.CKmX 0:CKm0  1:CKm1*/  /** SET **/

#define MMC_UBRG_IDENTIFICATION   (uint8_t)29    /* BRR identification mode setting*/  /** SET **/
/*                                        ++---------------- 400KHz               */  /** SET **/
#define MMC_UBRG_D_TRANSFER       (uint8_t)0     /* BRR data Transfer mode setting */  /** SET **/
/*                                        ++---------------- 12.0MHz              */  /** SET **/
#define MMC_CLK_D_TRANSFER (uint32_t)12000000 /* Data Transfer mode clock frequency*/  /** SET **/
```

- fCLK=20MHz(XIN) , (fMCK=20MHz,) fTCLK=10MHz
  Please define baud rate in r_mmc_user_config.h. Example is below.

```
/*-----------------------------------------------------------------------------------------*/
/* Define the value of the bit rate register according to a communication baud rate.      */
 (omit)
/* fCLK = 20MHz , fMCK = 20MHz , fTCLK = 10MHz */
#define MMC_FCLK_PRESCALER_SELECT 0             /* SPSm.PRSmk[3:0]                */  /** SET **/

#define MMC_OPERATION_CLK_SELECT  0             /* select SMRm.CKmX 0:CKm0  1:CKm1 */  /** SET **/

#define MMC_UBRG_IDENTIFICATION   (uint8_t)24    /* BRR identification mode setting */  /** SET **/
/*                                        ++---------------- 400KHz               */  /** SET **/
#define MMC_UBRG_D_TRANSFER       (uint8_t)0     /* BRR data Transfer mode setting  */  /** SET **/
/*                                        ++---------------- 10.0MHz              */  /** SET **/
#define MMC_CLK_D_TRANSFER (uint32_t)10000000 /* Data Transfer mode clock frequency*/  /** SET **/
```

4.3 Scheme of porting to RL78/G13

The different exist in only section 4.2.2 (1) and (2).

If you needs the other information, please refer to the section 4.1 and 4.2.

This section explains for section 4.2.2 [“define” for channel number of communication unit].

r_mmc_user_config.h and r_mmc_sfr.h needs to change.

Please rename the include directory about section 4.2.2 [Selecting MUC].

4.3.1 r_mmc_user_config.h

Communication unit channel number and usable pins combinations are different in each MCU type and number of pins.

[User’s Manual Hardware] section “Serian Array Unit” explains communication unit (MMC_SAU_UNIT) and communication unit channel number (MMC_SAU_CHANNEL), CSI channel combination.

Please update following table.

MMC_SAU_ UNIT Value	MMC_SAU_ CHANNEL Value	MMC_CSI_ PIN Value	SI Select port	SCK Select port	SO Select port
0 (=Use SAU0)	0 (=Use CSI00)	'A'	P50	P30	P51
		'B'	P16	P55	P17
	1 (=Use CSI01)	(invalid)	P74	P75	P73
	2	'A'	P03	P04	P02
	(=Use CSI10)	'B'	P81	P80	P82
	3 (=Use CSI11)	(invalid)	P11	P10	P12
1 (=Use SAU1)	0 (=Use CSI20)	(invalid)	P14	P15	P13
	1 (=Use CSI21)	(invalid)	P71	P70	P72
	2 (=Use CSI30)	(invalid)	P143	P142	P144
	3 (=Use CSI31)	(invalid)	P53	P54	P52

*/

```
For example, RL78/G13 32pin type product table as below.
------------+------------+-----------++-------------------------------+
  MMC_SAU_  |  MMC_SAU_  | MMC_CSI_  ||    SI     |   SCK    |    SO    |
  UNIT      |  CHANNEL   | PIN       ||  Select  |  Select  |  Select  |
   Value    |   Value    |  Value    ||   port    |   port   |   port   |
============+============+===========++==========+==========+==========+
     0      |     0      | (invalid) ||   P11    |   P10    |   P12    |
 (=Use SAU0)|(=Use CSI00)|           ||          |          |          |
            +------------+-----------++----------+----------+----------+
            |     1      |    ---    ||   ---    |   ---    |   ---    |
            +------------+-----------++----------+----------+----------+
            |     2      |    ---    ||   ---    |   ---    |   ---    |
            +------------+-----------++----------+----------+----------+
            |     3      | (invalid) ||   P50    |   P30    |   P51    |
            |(=Use CSI11)|           ||          |          |          |
------------+------------+-----------++----------+----------+----------+
     1      |     0      | (invalid) ||   P14    |   P15    |   P13    |
(=Use SAU1) |(=Use CSI20)|           ||          |          |          |
            +------------+-----------++----------+----------+----------+
            |     1      |    ---    ||   ---    |   ---    |   ---    |
------------+------------+-----------++----------+----------+----------+*/
Allocation port to SI, SCK, SO information is shown in RL78/G13 User's Manual
section 1 outline.
```

- 30, 32-pin products

Unit	Channel	Used as CSI	Used as UART	Used as Simplified I^2C
0	0	CSI00	UART0	IIC00
	1	–		–
	2	–	UART1	–
	3	CSI11		IIC11
1	0	CSI20	UART2 (supporting LIN-bus)	IIC20
	1	–		–

Figure 1 Allocation of communication function (Quotes from RL78/G13 User's Manual rev3.10)

```
Please update communication unit channel definition fitting to user circuit.
This example, CSI20 channel is used.

/* Serial Array Unit(SAU) Select ( 0 or 1 )*/
#define MMC_SAU_UNIT            1                   /** SET **/
/* SAU Channel Select ( 0 or 1 or 2 or 3 ) */
#define MMC_SAU_CHANNEL         0                   /** SET **/

In RL78/G13 32 pin product, one-pair pins are allocated for CSI channel.
"#define MMC_CSI_PIN" is not necessary.
```

4.3.2 r_mmc_sfr.h

This section explains about select port settings.

Target of changings are following 12 defines. Please unchanged for others.

1. #define MMC_CSI_UNIT
   Setting for CSI unit number. (“m” of CSImn)
   For example, CSI20 -> “2”

2. #define MMC_CSI_CHANNEL
   Setting for CSI channel number (“n” of CSImn)
   For example CSI20 -> “0”

3. #define MMC_DATAI_PORTNO
   Setting for port number about allocate as data input line(SI)
   For example P14 -> “1”

4. #define MMC_DATAI_BITNO
   Setting for bit number about allocate as data input line (SI)
   For example P14 -> “4”

5. #define MMC_CLK_PORTNO
   Setting port number about allocate as CLOCK line (SCK)
   For example P15 -> “1”

6. #define MMC_CLK_BITNO
   Setting bit number about allocate as CLOCK line (SCK)
   For example P15 -> “5”

7. #define MMC_DATAO_PORTNO
   Setting port number about allocate as data output line (SO)
   For example P13 -> “1”

8. #define MMC_DATAO_BITNO
   Setting bit number about allocate as data output line (SO)
   For example P13 -> “3”

9. #define MMC_CSI_SIR_CLEAR
   This is setting value to clear the serial flag clear trigger register (SIR)
   Clear flag when set bit to target bit.
   FECT bit (framing error flag clear trigger) is limited about valid
   unit/channel combination. Please refer to the MCU User’s Manual.
   Please set “7” when FECT bit is available combination.
   Please set “3” when FECT bit is unavailable combination.

5. Method for connecting to MCU and MCU resource for use with

5.1 MCU resource for use with

This software controls as follows:

Data input/output is controlled by clock synchronous serial I/O (internal clock).

Allocate CMOS output port and set CMOS output of the clock synchronous serial I/O in order to perform high-speed processing.

Please do CMOS output setting.

The transmission control detects the space of the transmission buffer, and use a transmission interrupt request bit without using an interrupt. Therefore, I set it about an interrupt as follows.

- Set “1” (disable interrupt process)
- Connect Card CS# pin to RX Port and control it by RX general port setting.

Resources	RL78/G14, RL78/G23
SCI (clocked synchronous mode)	M
CRC calculation circuit	R
Port for CS#: 1port/Card	M
Port for Card detection: 1port/Card	R
Port for Power Control: 1port/Card	M

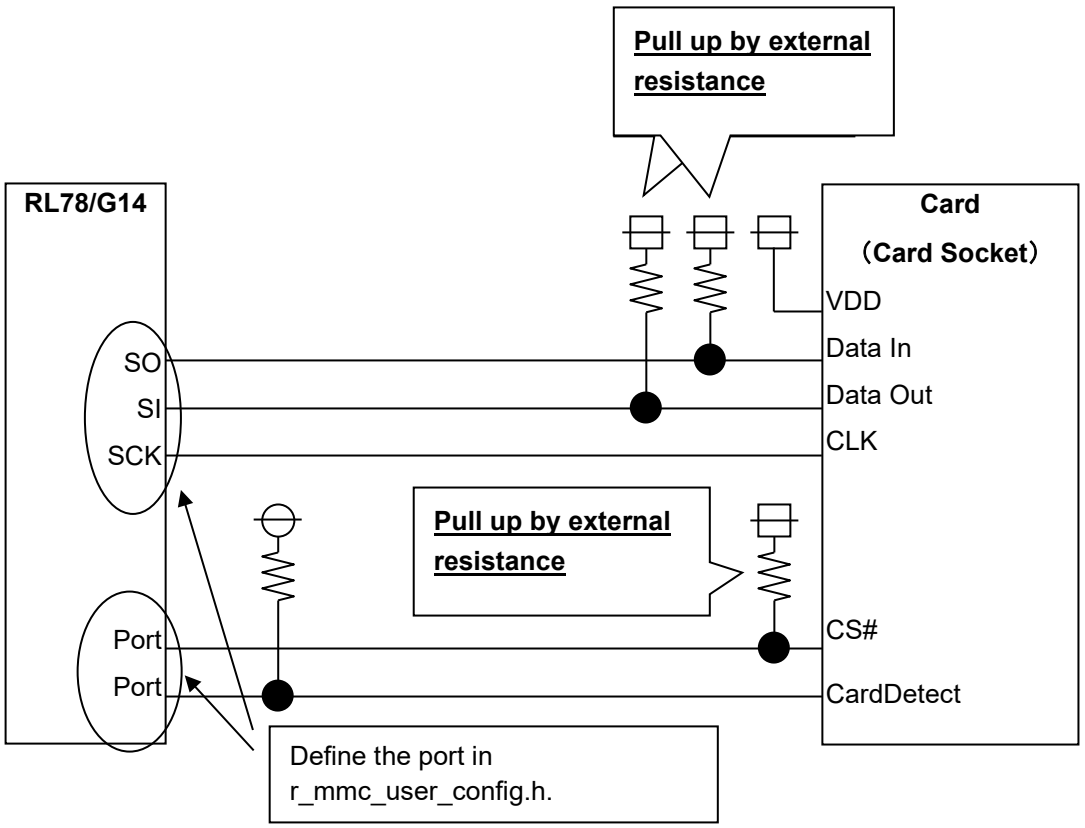
M: mandatory

R: recommended (high-speed processing is enabled when unique resource of RL78 is used)

5.2 Method for connecting to MCU

An example of connecting to RL78/G14 is showed.

In case of other RL78 family MCUs, the same connection is made



Power control circuit for card is required to control inserting and removing.

(power control circuit is not described).

Supply power to the card after detection of insertion.

Cut off the power supply to a card before removing.

In case of using multiple cards in the same SPI bus, assigne CS# port/CardDetect port for each card.

6. Notes for Application Development

6.1 Notes for use

- Configure the software according to the hardware.
- Remove card after deactivation, setting signals between MCU and card into Hi-z and cutting off power supply to card.
- In case that insertion/removable circuit is not realized, inserting/removing card might cause the power source to be unstable and reset MCU.

6.2 Development environment

Requirement items

When user develops, choose newer version than below.

6.2.1 CC-RL (C compiler)

-Integrated Development Environment

CS+ for CC V8.05.00
e^2 studio 2021-04 (21.4.0)

-C compiler

CC-RL V1.10.00

-Code Generator tool

(CS+) : Renesas Smart Configurator for RL78 V1.00.00.04
(e^2 studio) : Renesas Smart Configurator for RL78 21.4.0.v20210315-0928

6.2.2 IAR C/C++ Compiler for Renesas RL78 (C compiler)

-Integrated Development Environment and C compiler

IAR Embedded Workbench for Renesas RL78 version 4.21.1

-C compiler

IAR C/C++ Compiler for RL78 version : 4.20.1.2260 (4.20.1.2260)

-Code Generator tool

Renesas Smart Configurator for RL78 Version: 1.0.1

6.2.3 LLVM for Renesas RL78 (C compiler)

-Integrated Development Environment

e^2 studio 2022-07 (22.7.0)

-C compiler

LLVM for Renesas RL78 10.0.0.202207

-Code Generator tool

(e^2 studio) : Renesas Smart Configurator for RL78 22.7.0.v20220620-0602

6.2.4 Sample Project

The sample program that uses MMC driver is in the following Application note.

Document title: Sound Playback/Compression demonstration software for RL78/G14 CPU board
(Document number: R20AN0194)

Please download the sample code clicking following URL.

RL78/G14 Sound Playback/Compression Demonstration for RL78/G14 CPU Board - Sample Code | Renesas

6.3 Notes for embedding

6.3.1 Files for includding

Include "r_mmc.h" and "r_mtl_com.h" when embedding this device driver.

It is necessary for include to do "r_mtl_com.h" first.

6.3.2 Note of configuration (r_mmc_user_config.h)

- Note of select “B” for MMC_CSI_PIN macro
  User can change CSI channel (CSI00, CSI01) according to the number of pins of MCU. MMC_CSI_PIN macro can select using pins but if user select “B”, user has to set the PIOR0 resister.

6.3.3 Limitation of program

Program sets SAUmEN bit to “1” (enable) when program starts communication, but the program does not set “0” (disable) when program finishes communication. Please set “0” (disable) if necessary.

6.4 ROM size / RAM size / Stack size

MMC driver requires ROM/RAM/Stack size as below.

6.4.1 CC-RL (C compiler)

ROM size : about 6.5KByte

RAM size : about 1.8KByte (*)

Stack size : about 80byte

6.4.2 IAR C/C++ Compiler for Renesas RL78 (C compiler)

ROM size : about 6.3KByte

RAM size : about 2KByte (*)

Stack size : about 70byte

6.4.3 LLVM for Renesas RL78 (C compiler)

ROM size : about 7KByte

RAM size : about 1.6KByte (*)

Stack size : about 70byte

(*) 512 bytes for each necessary to writing and reading from upper layer program are included.

6.5 Notes on insertion/removal of the card

Enable to detect the insertion or removal of the card using the function "R_mmc_Chk_Detect()" with the card detection pins, which comes with the card connector.

Therefore, it is recommended to detect the insertion or removal of the card by software polling.

The driver returns an error when the card is removed in a data transmission eventually.

However, the driver may not return an error when the card is removed momentary in a data transmission in case of the following conditions:

- Data transmission is operated properly when no response error from the card occurs, because the driver cannot detect the insertion or removal of the card by software polling.
- The driver may recognize the completion of writing to the card when the card is inserted or removed momentary in writing stage. This is because of the specification that a writing completion will be detected by "H" signal of DetaIn pin. DataIn pin is pulled up.

Please avoid this problem by the system hardware such as a hardware interrupt control and polling period time etc.

6.6 Note of the Hi-z setting processing of the port about the exclusion and adding of the card

In the insertion of the card, Please insert card after having set CS#, DataIn, DataOut, and Clock terminal in Hi-z. Please supply the power supply to a card afterwards.

In the extraction of the card, after the power supply supply stop to a card, after the power supply supply stop to a card, please extract a card, after having set CS#, DataIn, DataOut, and Clock terminal in Hi-z.

The CS#, DataIn, DataOut, CLK terminal of the card is assigned to CSI or a port terminal of MCU, but does not process Hi-z by this driver because the case that the port is assigned to other resources is expected.

Therefore, please make the Hi-z processing of the MCU terminal in the high rank side in the exclusion and adding of the card.

7. MMC Driver Information

Ver	change	Release date
2.01	・Supported LLVM. Added SFR definitions r_user_config.h and r_mmc_sfr for RL78 / G23 group CSI (LLVM). Added definitions r_mtl_com.h for 32MHz operation (LLVM).	Nov.09.2022
2.00	・Supported RL78/G23. ・Added SFR definitions r_user_config.h and r_mmc_sfr for RL78 / G23 group CSI.	Jul.14.2021
1.03	・Supported CS+ for CC. ・Deleted MMC_OPTION_3 and MMC_OPTION_4 from r_mmc_user_config.h	Oct.01.2015
1.02	Add information about porting to RL78/G13.	Apr.01.2015
1.01	Supported IAR Embedded Workbench. Added config settings for the case that detect pin is not connected to MCU.	Sep.01.14
1.00	first release	Mar.31.12

Website and Support

Renesas Electronics Website
http://www.renesas.com/

Inquiries
http://www.renesas.com/contact/

Revision History

Rev.	Date	Description	
		Page	Summary
2.01	Nov.09.2022	-	Supported LLVM.
		5	Added LLVM content to 1.3 File configuration.
		14	Added LLVM content to 4.1.1 r_mtl_com.h.
		17	Added LLVM content to 4.2.2 r_mmc_user_config.h (MCU individual definitions header file).
		28	Added LLVM content to 6.2 Development environment.
		30	Added LLVM content to 6.4 ROM size / RAM size / Stack size.
2.00	Jul.14.2021	-	Deleted CS+ for CA,CX. Supported e^2 studio. Supported RL78/G23. Added SFR definition for RL78 / G23 group CSI.
1.03	Oct.01.2015	-	Changed CubeSuite+ to CS+ for CA,CX Supported CS+ for CC. Deleted sample program. Deleted description that is communication using DTC function.
1.02	Apr.01.2015	-	Add information about porting to RL78/G13.
1.01	Sep.01.2014	-	Release with V.1.01 Release 00
1.00	Mar.31.2012	-	First edition issued

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

The following usage notes are applicable to all Microprocessing unit and Microcontroller unit products from Renesas. For detailed usage notes on the products covered by this document, refer to the relevant sections of the document as well as any technical updates that have been issued for the products.

1. Precaution against Electrostatic Discharge (ESD)

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

2. Processing at power-on

   The state of the product is undefined at the time when power is supplied. The states of internal circuits in the LSI are indeterminate and the states of register settings and pins are undefined at the time when power is supplied. In a finished product where the reset signal is applied to the external reset pin, the states of pins are not guaranteed from the time when power is supplied until the reset process is completed. In a similar way, the states of pins in a product that is reset by an on-chip power-on reset function are not guaranteed from the time when power is supplied until the power reaches the level at which resetting is specified.

3. Input of signal during power-off state

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

4. Handling of unused pins

   Handle unused pins in accordance with the directions given under handling of unused pins in the manual. The input pins of CMOS products are generally in the high-impedance state. In operation with an unused pin in the open-circuit state, extra electromagnetic noise is induced in the vicinity of the LSI, an associated shoot-through current flows internally, and malfunctions occur due to the false recognition of the pin state as an input signal become possible.

5. Clock signals

   After applying a reset, only release the reset line after the operating clock signal becomes stable. When switching the clock signal during program execution, wait until the target clock signal is stabilized. When the clock signal is generated with an external resonator or from an external oscillator during a reset, ensure that the reset line is only released after full stabilization of the clock signal. Additionally, when switching to a clock signal produced with an external resonator or by an external oscillator while program execution is in progress, wait until the target clock signal is stable.

6. Voltage application waveform at input pin

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

7. Prohibition of access to reserved addresses

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

8. Differences between products

   Before changing from one product to another, for example to a product with a different part number, confirm that the change will not lead to problems. The characteristics of a microprocessing unit or microcontroller unit products in the same group but having a different part number might differ in terms of internal memory capacity, layout pattern, and other factors, which can affect the ranges of electrical characteristics, such as characteristic values, operating margins, immunity to noise, and amount of radiated noise. When changing to a product with a different part number, implement a system-evaluation test for the given product.

Notice

1. Descriptions of circuits, software and other related information in this document are provided only to illustrate the operation of semiconductor products and application examples. You are fully responsible for the incorporation or any other use of the circuits, software, and information in the design of your product or system. Renesas Electronics disclaims any and all liability for any losses and damages incurred by you or third parties arising from the use of these circuits, software, or information.
2. Renesas Electronics hereby expressly disclaims any warranties against and liability for infringement or any other claims involving patents, copyrights, or other intellectual property rights of third parties, by or arising from the use of Renesas Electronics products or technical information described in this document, including but not limited to, the product data, drawings, charts, programs, algorithms, and application examples.
3. No license, express, implied or otherwise, is granted hereby under any patents, copyrights or other intellectual property rights of Renesas Electronics or others.
4. You shall be responsible for determining what licenses are required from any third parties, and obtaining such licenses for the lawful import, export, manufacture, sales, utilization, distribution or other disposal of any products incorporating Renesas Electronics products, if required.
5. You shall not alter, modify, copy, or reverse engineer any Renesas Electronics product, whether in whole or in part. Renesas Electronics disclaims any and all liability for any losses or damages incurred by you or third parties arising from such alteration, modification, copying or reverse engineering.
6. Renesas Electronics products are classified according to the following two quality grades: “Standard” and “High Quality”. The intended applications for each Renesas Electronics product depends on the product’s quality grade, as indicated below.
   "Standard": Computers; office equipment; communications equipment; test and measurement equipment; audio and visual equipment; home electronic appliances; machine tools; personal electronic equipment; industrial robots; etc.
   "High Quality": Transportation equipment (automobiles, trains, ships, etc.); traffic control (traffic lights); large-scale communication equipment; key financial terminal systems; safety control equipment; etc.

   Unless expressly designated as a high reliability product or a product for harsh environments in a Renesas Electronics data sheet or other Renesas Electronics document, Renesas Electronics products are not intended or authorized for use in products or systems that may pose a direct threat to human life or bodily injury (artificial life support devices or systems; surgical implantations; etc.), or may cause serious property damage (space system; undersea repeaters; nuclear power control systems; aircraft control systems; key plant systems; military equipment; etc.). Renesas Electronics disclaims any and all liability for any damages or losses incurred by you or any third parties arising from the use of any Renesas Electronics product that is inconsistent with any Renesas Electronics data sheet, user’s manual or other Renesas Electronics document.
7. No semiconductor product is absolutely secure. Notwithstanding any security measures or features that may be implemented in Renesas Electronics hardware or software products, Renesas Electronics shall have absolutely no liability arising out of any vulnerability or security breach, including but not limited to any unauthorized access to or use of a Renesas Electronics product or a system that uses a Renesas Electronics product. RENESAS ELECTRONICS DOES NOT WARRANT OR GUARANTEE THAT RENESAS ELECTRONICS PRODUCTS, OR ANY SYSTEMS CREATED USING RENESAS ELECTRONICS PRODUCTS WILL BE INVULNERABLE OR FREE FROM CORRUPTION, ATTACK, VIRUSES, INTERFERENCE, HACKING, DATA LOSS OR THEFT, OR OTHER SECURITY INTRUSION (“Vulnerability Issues”). RENESAS ELECTRONICS DISCLAIMS ANY AND ALL RESPONSIBILITY OR LIABILITY ARISING FROM OR RELATED TO ANY VULNERABILITY ISSUES. FURTHERMORE, TO THE EXTENT PERMITTED BY APPLICABLE LAW, RENESAS ELECTRONICS DISCLAIMS ANY AND ALL WARRANTIES, EXPRESS OR IMPLIED, WITH RESPECT TO THIS DOCUMENT AND ANY RELATED OR ACCOMPANYING SOFTWARE OR HARDWARE, INCLUDING BUT NOT LIMITED TO THE IMPLIED WARRANTIES OF MERCHANTABILITY, OR FITNESS FOR A PARTICULAR PURPOSE.
8. When using Renesas Electronics products, refer to the latest product information (data sheets, user’s manuals, application notes, “General Notes for Handling and Using Semiconductor Devices” in the reliability handbook, etc.), and ensure that usage conditions are within the ranges specified by Renesas Electronics with respect to maximum ratings, operating power supply voltage range, heat dissipation characteristics, installation, etc. Renesas Electronics disclaims any and all liability for any malfunctions, failure or accident arising out of the use of Renesas Electronics products outside of such specified ranges.
9. Although Renesas Electronics endeavors to improve the quality and reliability of Renesas Electronics products, semiconductor products have specific characteristics, such as the occurrence of failure at a certain rate and malfunctions under certain use conditions. Unless designated as a high reliability product or a product for harsh environments in a Renesas Electronics data sheet or other Renesas Electronics document, Renesas Electronics products are not subject to radiation resistance design. You are responsible for implementing safety measures to guard against the possibility of bodily injury, injury or damage caused by fire, and/or danger to the public in the event of a failure or malfunction of Renesas Electronics products, such as safety design for hardware and software, including but not limited to redundancy, fire control and malfunction prevention, appropriate treatment for aging degradation or any other appropriate measures. Because the evaluation of microcomputer software alone is very difficult and impractical, you are responsible for evaluating the safety of the final products or systems manufactured by you.
10. Please contact a Renesas Electronics sales office for details as to environmental matters such as the environmental compatibility of each Renesas Electronics product. You are responsible for carefully and sufficiently investigating applicable laws and regulations that regulate the inclusion or use of controlled substances, including without limitation, the EU RoHS Directive, and using Renesas Electronics products in compliance with all these applicable laws and regulations. Renesas Electronics disclaims any and all liability for damages or losses occurring as a result of your noncompliance with applicable laws and regulations.
11. Renesas Electronics products and technologies shall not be used for or incorporated into any products or systems whose manufacture, use, or sale is prohibited under any applicable domestic or foreign laws or regulations. You shall comply with any applicable export control laws and regulations promulgated and administered by the governments of any countries asserting jurisdiction over the parties or transactions.
12. It is the responsibility of the buyer or distributor of Renesas Electronics products, or any other party who distributes, disposes of, or otherwise sells or transfers the product to a third party, to notify such third party in advance of the contents and conditions set forth in this document.
13. This document shall not be reprinted, reproduced or duplicated in any form, in whole or in part, without prior written consent of Renesas Electronics.
14. Please contact a Renesas Electronics sales office if you have any questions regarding the information contained in this document or Renesas Electronics products.

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

(Rev.5.0-1 October 2020)

Corporate Headquarters

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

Trademarks

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

Contact information

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