Notice

1. All information included in this document is current as of the date this document is issued. Such information, however, is subject to change without any prior notice. Before purchasing or using any Renesas Electronics products listed herein, please confirm the latest product information with a Renesas Electronics sales office. Also, please pay regular and careful attention to additional and different information to be disclosed by Renesas Electronics such as that disclosed through our website.
2. Renesas Electronics does not assume any liability for infringement of 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. No license, express, implied or otherwise, is granted hereby under any patents, copyrights or other intellectual property rights of Renesas Electronics or others.
3. You should not alter, modify, copy, or otherwise misappropriate any Renesas Electronics product, whether in whole or in part.
4. 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 of these circuits, software, and information in the design of your equipment. Renesas Electronics assumes no responsibility for any losses incurred by you or third parties arising from the use of these circuits, software, or information.
5. When exporting the products or technology described in this document, you should comply with the applicable export control laws and regulations and follow the procedures required by such laws and regulations. You should not use Renesas Electronics products or the technology described in this document for any purpose relating to military applications or use by the military, including but not limited to the development of weapons of mass destruction. Renesas Electronics products and technology may 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.
6. Renesas Electronics has used reasonable care in preparing the information included in this document, but Renesas Electronics does not warrant that such information is error free. Renesas Electronics assumes no liability whatsoever for any damages incurred by you resulting from errors in or omissions from the information included herein.
7. Renesas Electronics products are classified according to the following three quality grades: "Standard", "High Quality", and "Specific". The recommended applications for each Renesas Electronics product depends on the product's quality grade, as indicated below. You must check the quality grade of each Renesas Electronics product before using it in a particular application. You may not use any Renesas Electronics product for any application categorized as "Specific" without the prior written consent of Renesas Electronics. Further, you may not use any Renesas Electronics product for any application for which it is not intended without the prior written consent of Renesas Electronics. Renesas Electronics shall not be in any way liable for any damages or losses incurred by you or third parties arising from the use of any Renesas Electronics product for an application categorized as "Specific" or for which the product is not intended where you have failed to obtain the prior written consent of Renesas Electronics. The quality grade of each Renesas Electronics product is "Standard" unless otherwise expressly specified in a Renesas Electronics data sheets or data books, etc.

   "Standard": Computers; office equipment; communications equipment; test and measurement equipment; audio and visual equipment; home electronic appliances; machine tools; personal electronic equipment; and industrial robots.

   "High Quality": Transportation equipment (automobiles, trains, ships, etc.); traffic control systems; anti-disaster systems; anti-crime systems; safety equipment; and medical equipment not specifically designed for life support.

   "Specific": Aircraft; aerospace equipment; submersible repeaters; nuclear reactor control systems; medical equipment or systems for life support (e.g. artificial life support devices or systems), surgical implantations, or healthcare intervention (e.g. excision, etc.), and any other applications or purposes that pose a direct threat to human life.
8. You should use the Renesas Electronics products described in this document within the range specified by Renesas Electronics, especially with respect to the maximum rating, operating supply voltage range, movement power voltage range, heat radiation characteristics, installation and other product characteristics. Renesas Electronics shall have no liability for malfunctions or damages arising out of the use of Renesas Electronics products beyond such specified ranges.
9. Although Renesas Electronics endeavors to improve the quality and reliability of its products, semiconductor products have specific characteristics such as the occurrence of failure at a certain rate and malfunctions under certain use conditions. Further, Renesas Electronics products are not subject to radiation resistance design. Please be sure to implement safety measures to guard them against the possibility of physical injury, and injury or damage caused by fire in the event of the failure of a Renesas Electronics product, 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, please evaluate the safety of the final products or system 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. Please use Renesas Electronics products in compliance with all applicable laws and regulations that regulate the inclusion or use of controlled substances, including without limitation, the EU RoHS Directive. Renesas Electronics assumes no liability for damages or losses occurring as a result of your noncompliance with applicable laws and regulations.
11. This document may not be reproduced or duplicated, in any form, in whole or in part, without prior written consent of Renesas Electronics.
12. Please contact a Renesas Electronics sales office if you have any questions regarding the information contained in this document or Renesas Electronics products, or if you have any other inquiries.

(Note 1) "Renesas Electronics" as used in this document means Renesas Electronics Corporation and also includes its majority-owned subsidiaries.

(Note 2) "Renesas Electronics product(s)" means any product developed or manufactured by or for Renesas Electronics.

H8/300L SLP Series

Multiprocessor Communication

Introduction

Data transmission is performed via the serial communication interface using the multiprocessor communication function of the H8/38024. Transmit data format is specified as 8-bit length followed by a multiprocessor bit and a stop bit. Data transmission is at 31250 bit/sec.

Target Device

H8/38024

Contents

1. Specifications

1. Using the multiprocessor communication function, data H'B8 is transmitted to receiver A and H'DE to receiver B as shown in figure 1.1.
2. The transmit data format for communication is specified for 8-bit length, one bit of multiprocessor bit, and the stop bit length set to one bit.
3. Data is transmitted at the bit rate of 31250 bps. A break is output upon completion of data transmission.

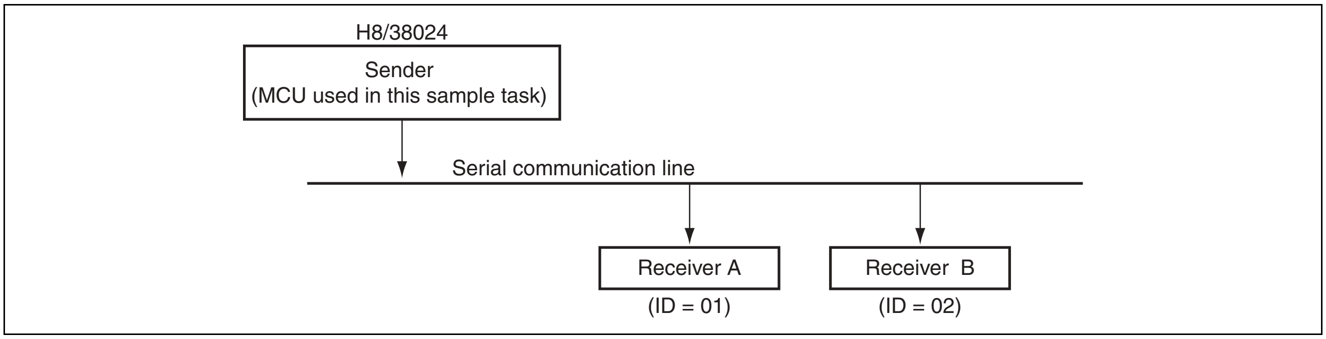


Figure 1.1 Multiprocessor Communication

2. Description of Functions

1. In this sample task, data transmission is performed via a serial communication interface (SCI) using the multiprocessor communication function. Figure 2.2 shows a block diagram of transmission in multiprocessor communication described below.
   — The multiprocessor communication function enables data to be exchanged among a number of processors on a shared communication line. Serial data communication is performed in asynchronous mode using the multiprocessor format (in which a multiprocessor bit is added to the transfer data).
   — In multiprocessor communication, each receiver is assigned its own ID code. The serial communication cycle consists of two cycles, an ID transmission cycle in which the receiver is specified, and a data transmission cycle in which the transfer data is sent to the specified receiver.
   — These two cycles are differentiated by means of the multiprocessor bit: 1 indicating an ID transmission cycle and 0, a data transmission cycle.
   — The sender first sends transfer data with a 1 multiprocessor bit added to the ID code of the receiver it wants to communicate with, and then sends transfer data with a 0 multiprocessor bit added to the transmit data.
   — When a receiver receives transfer data with the multiprocessor bit set to 1, it compares the ID code with its own ID code, and if they match, receives the transfer data sent next. If the ID codes do not match, it skips the transfer data until data with the multiprocessor bit set to 1 is sent again.
   — There is a choice of four data transfer formats. If a multiprocessor format is specified, the parity bit specification is invalid.
   — Since separate transmission and reception units are provided, transmission and reception can take place simultaneously. The transmission and reception units are both double-buffered, allowing continuous transmission and reception.
   — The internal baud rate generator allows any desired bit rate to be selected.
   — An internal or external clock can be selected as the transmit/receive clock source.
   — There are six possible interrupt sources: transmit end, transmit data empty, receive data full, overrun error, framing error, and parity error
   — The Receive Shift Register (RSR) is the register for receiving serial data. Serial data input from RXD32 pin is set in RSR in the receiving order starting from the LSB (Bit 0) to be converted to parallel data. When one byte of data has been received, the data is automatically transferred to RDR. RSR cannot be read from or written to directly by the CPU.

— The Receive Data Register (RDR) is an 8-bit register for storing received serial data. When one byte of serial data has been received, the data is transferred from RSR to RDR, which completes the receive operation. RSR is then ready to receive new data. RSR and RDR are both double-buffered, enabling continuous receive operations. RDR is a receive-only register and cannot be written to by the CPU.

— The Transmission Shift Register (TSR) is a register for transmission of serial data. Transmit data is temporarily transferred from TDR to TSR, from which the data bits are sent to TXD32 pin serially starting from LSB (Bit 0) for serial data transmission. When one byte of data has been transmitted, the next transmit data is automatically transferred from TDR to TSR to start the next transmission. If data is not written in TDR (1 is set in TDRE), data is not transferred from TDR to TSR. TSR cannot be read from or written to directly by the CPU.

— The Transmit Data Register (TDR) is an 8-bit register that stores transmit data. Upon detection of "TSR empty", transmit data written in TDR is transferred to TSR to start serial data transmission. Continuous transmission is possible by writing the next transmit data to TDR while the data in TSR is serially transmitted. TDR can always be read from or written to by the CPU.

— The Serial Mode Register (SMR) is an 8-bit register that specifies the serial data transfer format and select the clock source for the baud rate generator. SMR can always be read from or written to by the CPU.

— The Serial Control Register 3 (SCR3) is an 8-bit register that selects transmit/receive operation, clock output in asynchronous mode, interrupt request enable/disable, and transmit/receive clock source. SCR3 can always be read from or written to by the CPU.

— Figure 2.1 shows serial data in which the multiprocessor bit is used and table 2.1 shows the multiprocessor format.

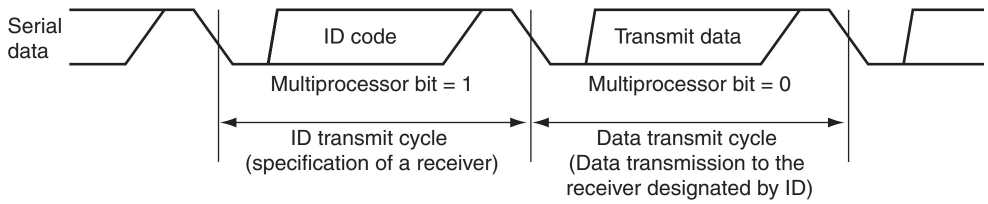


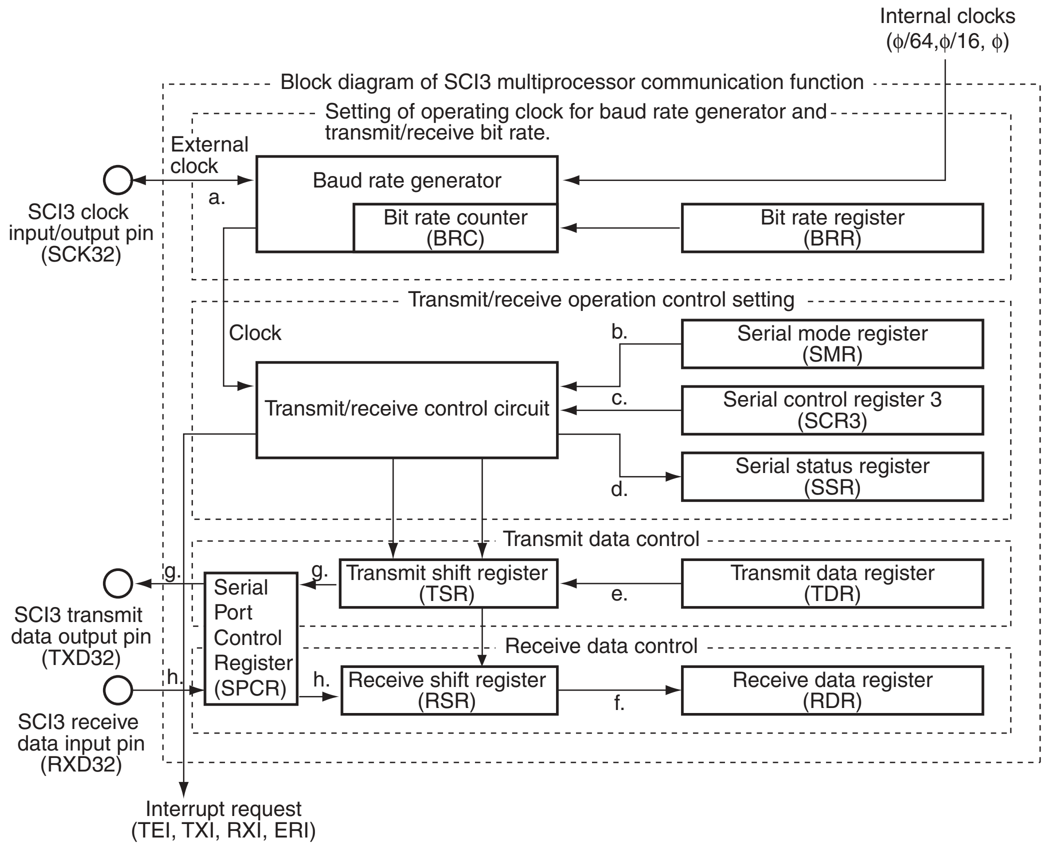
Figure 2.1 Serial Data in Which the Multiprocessor Bit is Used

Table 2.1 Multiprocessor Format

MSR				Serial Transfer Format and Frame Length											
CHR	PE	MP	STOP	1	2	3	4	5	6	7	8	9	10	11	12
0	*	1	0	S	8-bit data								MPB	STOP	
0	*	1	1	S	8-bit data								MPB	STOP	STOP
1	*	1	0	S	7-bit data							MPB	STOP		
1	*	1	1	S	7-bit data							MPB	STOP	STOP	

Note: * : Don't care

[Legend]
S: Start bit
STOP: Stop bit
MPB: Multiprocessor bit


Internal clocks
(ϕ/64,ϕ/16, ϕ)
Block diagram of SCI3 multiprocessor communication function
Setting of operating clock for baud rate generator and transmit/receive bit rate.
External clock
a.
SCI3 clock input/output pin (SCK32)
Baud rate generator
Bit rate counter (BRC)
Bit rate register (BRR)
Transmit/receive operation control setting
Clock
b.
Serial mode register (SMR)
c.
Serial control register 3 (SCR3)
Transmit/receive control circuit
d.
Serial status register (SSR)
Transmit data control
g.
g.
Serial Port Control Register (SPCR)
Transmit shift register (TSR)
e.
Transmit data register (TDR)
SCI3 transmit data output pin (TXD32)
Receive data control
h.
h.
Receive shift register (RSR)
f.
Receive data register (RDR)
SCI3 receive data input pin (RXD32)
Interrupt request (TEI, TXI, RXI, ERI)

Notes: a. The transmit/receive bit rate should be set in accordance with the operating clock source for the baud rate generator selected by SMR. The transmit bit rate is set to 31250 bps in this sample task.

b. The serial data transfer format is set and clock source for baud rate generator is selected. In this sample task, the serial data transfer format is set to asynchronous mode as the operating mode, the data length is set to 8 bits, parity bit addition and check are disabled, the multiprocessor communication function is enabled, the stop bit length is set to one bit, and the clock source for the built-in baud rate generator is set to ϕ.

c. The transmit/receive operation, clock output in asynchronous mode, and interrupt request enable/disable are selected. In this sample task, the clock output in asynchronous mode is set up as follows: clock source is an internal clock and SCK32 pin functions as a clock output pin. Interrupt requests by transmit data empty and by receive data full are disabled.

d. The status flags (transmit data register empty, receive data register full, overrun error, framing error, parity error and transmission end) indicate operational states of SCI3.

e. On detection of "TSR empty", transmit data written in TDR is transferred to TSR.

f. On receiving one byte of data, the received data is transferred from RSR to RDR.

g. Transmit data

h. Receive data

Figure 2.2 Block Diagram of Multiprocessor Communication Function

- The serial status register (SSR) is an 8-bit register that contains status flags indicating operation status of SCI3 and the bits for storing the state of the multiprocessor bit. SSR can always be read from or written to by the CPU, except that 1 cannot be written to TDRE, RDRF, OER, PER and FER. Before clearing these bits by writing 0, 1 must be read from them. TEND and MPBR are for read-only and cannot be written to.
- The bit rate register (BRR) is an 8-bit register that specifies the bit rate for transmit/receive in accordance with the operating clock for the baud rate generator selected by CSK1 and CKS0 in SMR. BRR can always be read from or written to by the CPU.
- Table 2.2 shows examples of BRR setting in asynchronous mode. Table 2.2 shows values in the active mode when OSC is 10 MHz.

Table 2.2 Examples of BRR Setting for Bit Rates (Asynchronous Mode)

R Bit Rate (Bps)	**110**	**150**	**200**	**250**	**1200**	**2400**	**31250**
n	2	2	1	0	0	0	0
N	88	64	48	38	129	64	4
Error (%)	–0.25	+0.16	-0.35	+0.16	+0.16	+0.16	0.00

Notes: 1. BBR must be set ensuring that the error is within 1%.

2. Values to be set in BRR can be found using the following equation:

$$N = \frac{OSC}{64 \times 2^{2n} \times B} \times 10^6 - 1$$

B: Bit rate (bps)
N: Value to be set in BBR for the baud rate generator ($0 \leq N \leq 255$)
OSC: Value of ϕ_{OSC} (MHz) = 10 MHz or subclock ϕw = 32.768 KHz
n: value set in CKS1 and CKS0 in SMR ($0 \leq n \leq 3$)
(See table 2.3 for the relation between n and clock.)

Table 2.3 Relation between n and Clock

		Values set in SMR	
N	Clock	CKS1	CKS0
0	ϕ	0	0
1	$\phi w/4$, ϕw	0	1
2	$\phi/16$	1	0
3	$\phi/64$	1	1

3. The error shown in table 2.2 is given by the following equation.
(rounded off to two decimal places)

$$\text{Error (\%)} = \left\{ \frac{\phi \times 10^6}{(N+1) \times B \times 64 \times 2^{2n-1}} - 1 \right\} \times 100$$

4. When OSC is 10 MHz, the maximum bit rate (asynchronous mode) is 31250 bps, provided n = 0 and N = 4.

- In asynchronous mode, character data to which a start bit indicating the start of communication and a stop bit indicating the end of communication are added is transmitted and received; and serial communication is performed with synchronization provided on a character by character basis.
- Since separate transmission and reception units are provided in SCI3, full duplex communications are possible. Both the transmission and reception units are double-buffered, data writing during transmission and data reading during reception are possible, which makes for continuous transmission and reception.
- In asynchronous communications, the communication line is normally kept in the marked state (high level). SCI3 monitors the communication line and starts serial communication when it detects a space (low level), regarding it as the start bit.
- One character in communication data consists of the start bit (low level), followed by transmit/receive data (LSB first), parity bit (high or low level) and a stop bit (high level) at the end.
- In asynchronous mode, operation is synchronized with the falling edge of the start bit during reception. Data is sampled on the eighth edge of the clock of a frequency 16 times the one bit period (functions as the. Accordingly, communication data is fetched in the center of each bit.
- SCK32 is a clock input/output pin of SCI3.
- RXD32 is a receive data input pin of SCI3.
- TXD32 is a transmit data output pin of SCI3.
- There are six SCI3 interrupt sources: transmit end, transmit data empty, receive data full, and three reception errors of overrun error, framing error, and parity error. Common vector address is assigned to them.
- Each type of interrupt requests can be enabled/disabled by TIE and RIE in SCR3.
- If TDRE in SSR is set to 1, a transmit data empty interrupt request (TXI) is generated. If TEND in SSR is set to 1, a transmit end interrupt request (TEI) is generated. These two interrupts are generated during transmission.
- The initial value of TDRE in SSR is 1. Therefore, by setting TIE in SCR3 to 1 and by enabling TXI before transmit data is transferred to TDR, TXI is generated even when transmit data is not ready.
- The initial value of TEND in SSR is 1. Therefore, by setting TEIE in SCR3 to 1 and by enabling TEI before transferring transmit data to TDR, TEI is generated even when transmit data is not sent.
- These interrupts can be used effectively by building the system so that the transfer of transmit data to TDR is performed within the interrupt handling routine. To prevent these interrupt requests (TXI and TEI), the their enable bits (TIE and TEIE) should only be set to 1 after transmit data has been transferred to TDR.
- RXI is generated when RDRF in SSR is set to 1. ERI is generated when OER, PER or FER is set to 1. These two interrupt requests are generated during reception.

2. Table 2.4 shows assignment of functions in this sample task. Multiprocessor communication is performed by assigning the functions as shown in table 2.4.

Table 2.4 Function Allocation

Function	Description
TSR	A register for transmitting serial data
TDR	A register for storing transmit data
SMR	Sets a serial data communication format and clock source for the baud rate generator
SSR	Status flags to indicate operation states of SCI3
BRR	Sets transmit/receive bit rate
SCR3	Enables transmit operation, sets TXD32 output pin, and sets the SCK32 pin as a clock output pin
SCK32	SCI3 clock output pin
TXD32	SCI3 transmit data output pin
SPCR	Sets TXD32 output pin

3. Principle of Operation

1. Figure 3.1 illustrates the operation of this sample task. Data transmission is performed by the multiprocessor communication function through hardware and software processing as shown in figure 3.1.

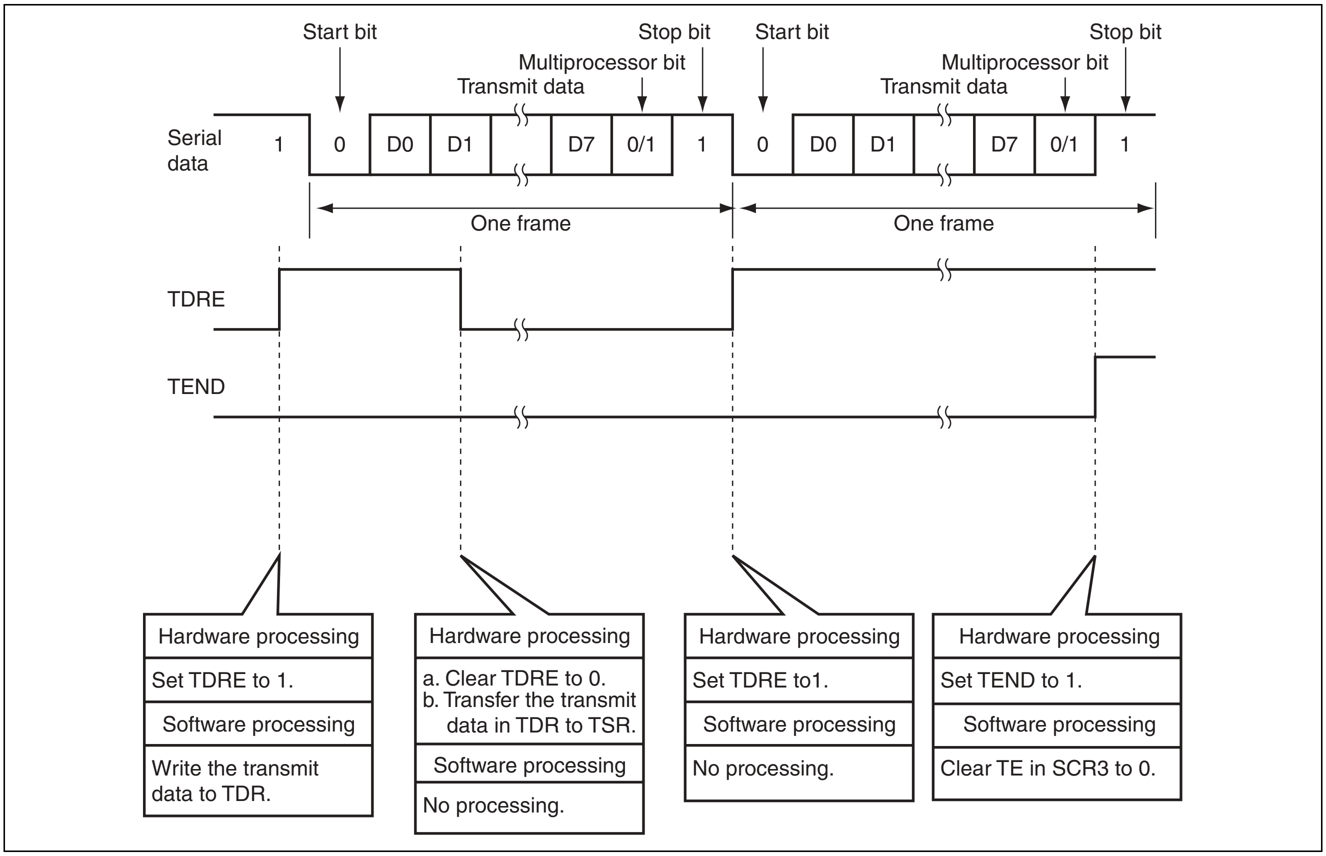


Figure 3.1 Operation Principle of Transmission by Multiprocessor Communication Function

4. Description of Software

4.1 Modules

Table 4.1 describes the module in this sample task.

Table 4.1 Description of Module

Module	Label	Function
Main Routine	main	Sets transfer data, sets up the multiprocessor communication function, and ends when 4-byte data has been sent.

4.2 Arguments

Table 4.2 describes the arguments used in this sample task.

Table 4.2 Description of Arguments

Argument	Function	Used in	Data Length	Input/ Output
STD[0] to STD[3]	Serial transmit data in asynchronous mode	Main Routine	1 byte	Input

4.3 Internal registers

Table 4.3 describes the internal registers in this sample task.

Table 4.3 Description of Internal Registers

Register		Function	Address	Setting
SMR	COM	Serial mode register (Communication mode) If COM = 0, the communication mode is set to asynchronous mode. If COM = 1, the communication mode is set to synchronous mode.	H'FFA8 Bit 7	0
	CHR	Serial mode register (Character length) If CHR = 0, the data length for asynchronous mode is set to 8 bits. If CHR = 1, the data length for asynchronous mode is set to 7 bits.	H'FFA8 Bit 6	0
	PE	Serial mode register (Parity enable) If PE = 0, parity bit addition and check during transmission are disabled in asynchronous mode. If PE = 1, parity bit addition and check during transmission are enabled in asynchronous mode.	H'FFA8 Bit 5	0
	STOP	Serial mode register (Stop bit length) If STOP = 0, the stop bit length in asynchronous mode is set to one. If STOP = 1, the stop bit length in asynchronous mode is set to two.	H'FFA8 Bit 3	0

Register		Function	Address	Setting
SMR	MP	Serial mode register (Multiprocessor mode) If MP = 0, the multiprocessor communication function is disabled. If MP = 1, the multiprocessor communication function is enabled.	H'FFA8 Bit 2	1

	CKS1
	CKS0
		Serial mode register (Clock select 1, 0)
		If CKS1 = 0 and CKS0 = 0, the clock source for the built-in baud rate generator is set to ϕ.
			H'FFA8
			Bit 1
			Bit 0
				CKS1 = 0
				CKS0 = 0

BRR		Bit rate register If BRR = H'04, the transmit bit rate matched to the operating clock for the baud rate generator selected by CKS1 and CKS0 in SMR is set to 31250 bps.	H'FFA9	H'04
SCR3	TE	Serial Control Register 3 (Transmit Enable) If TE = 0, transmission is disabled. If TE = 1, transmission is enabled.	H'FFAA Bit 5	0

	CKE1
	CKE0
		Serial control register 3 (Clock enable 1, 0)
		If CKE1 = 0 and CKE0 = 1, the clock source is set to an internal clock and SCK32 pin function to clock output in asynchronous mode.
			H'FFAA
			Bit 1
			Bit 0
				CKE1 = 0
				CKE0 = 1

TDR		Transmit data register An 8-bit register that stores transmit data	H'FFAB	—
SSR	TDRE	Serial status register (Transmit data register empty) If TDRE = 0, transmit data written in TDR has not yet been transferred to TSR. If TDRE = 1, transmit data is not written to TDR or transmit data written to TDR has been transferred to TSR.	H'FFAC Bit 7	1
	TEND	Serial status register (Transmit end) If TEND = 0, data is being transmitted. If TEND = 1, data transmission has been completed.	H'FFAC Bit 2	—
	MPBR	Serial status register (Multiprocessor bit receive) If MPBR = 0, data with the multiprocessor bit valued 0 has been received. If MPBR = 1, data with the multiprocessor bit valued 1 has been received.	H'FFAC Bit 1	0
	MPBT	Serial status register (Multiprocessor bit transfer) If MPBT = 0, 0-valued multiprocessor bit is sent. If MPBT = 1, 1-valued multiprocessor bit is sent.	H'FFAC Bit 0	0
SPCR	SPC32	Serial port control register (P42/TXD32 pin function switching) If SPC32 = 0, P42/TXD32 pin functions as the P42 pin. If SPC32 = 1, P42/TXD32 pin functions as the TXD32 pin.	H'FF91 Bit 5	1
	SCINV3	Serial port control register (TXD32 pin output data inversion) If SCINV3 = 0, TXD32 output data is not inverted. If SCINV3 = 1, TXD32 output data is inverted.	H'FF91 Bit 3	0

4.4 Description of RAM

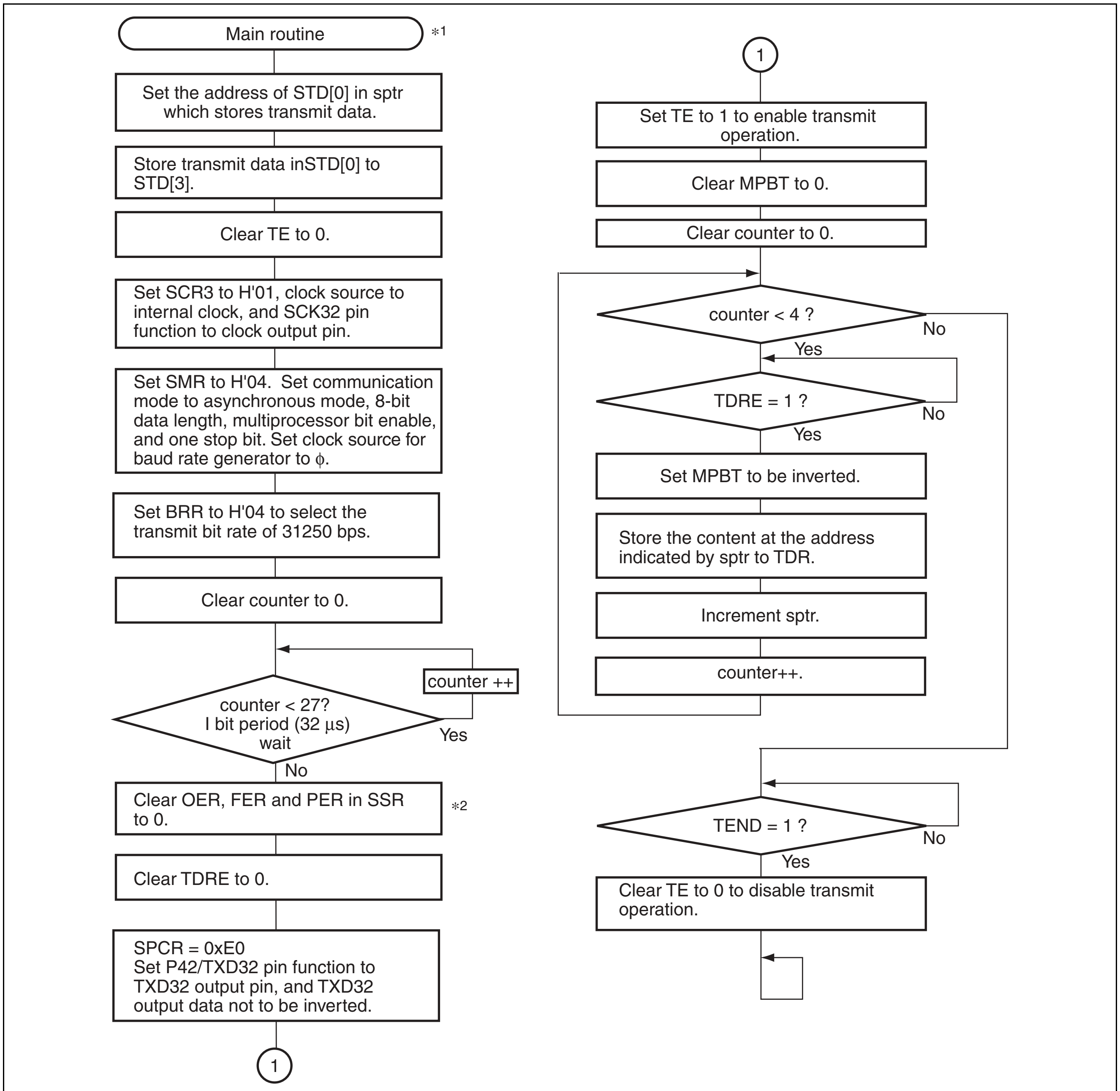
Table 4.4 describes the RAM area used in this sample task.

Table 4.4 Description of RAM

Label	Function	Address	Used in
STD[0]	Stores the first byte of serial transmit data in asynchronous mode.	H'FB80	Main Routine
STD[1]	Stores the second byte of serial transmit data in asynchronous mode.	H'FB81	Main Routine
STD[2]	Stores the third byte of serial transmit data in asynchronous mode.	H'FB82	Main Routine
STD[3]	Stores the fourth byte of serial transmit data in asynchronous mode.	H'FB83	Main Routine

5. Flowchart

1. Main routine


Main routine
*1
Set the address of STD[0] in sptr which stores transmit data.
Store transmit data inSTD[0] to STD[3].
Clear TE to 0.
Set SCR3 to H'01, clock source to internal clock, and SCK32 pin function to clock output pin.
Set SMR to H'04. Set communication mode to asynchronous mode, 8-bit data length, multiprocessor bit enable, and one stop bit. Set clock source for baud rate generator to ϕ.
Set BRR to H'04 to select the transmit bit rate of 31250 bps.
Clear counter to 0.
counter ++
counter < 27? I bit period (32 µs) wait
Yes
No
Clear OER, FER and PER in SSR to 0.
*2
Clear TDRE to 0.
SPCR = 0xE0 Set P42/TXD32 pin function to TXD32 output pin, and TXD32 output data not to be inverted.
1
1
Set TE to 1 to enable transmit operation.
Clear MPBT to 0.
Clear counter to 0.
counter < 4 ?
No
Yes
TDRE = 1 ?
No
Yes
Set MPBT to be inverted.
Store the content at the address indicated by sptr to TDR.
Increment sptr.
counter++.
TEND = 1 ?
No
Yes
Clear TE to 0 to disable transmit operation.

Notes: 1. In this sample task, the stack pointer is set in INIT.SRC (assembly language).
2. OER, FER, PER must be cleared to 0 in asynchronous mode (multiprocessor mode).

6. Program Listing

INIT.SRC (Program listing)

```
    .EXPORT    _INIT
    .IMPORT    _main
 ;a
    .SECTION  P,CODEa
 _INIT:a
    MOV.W     #H'FF80,R7
    LDC.B     #B'10000000,CCR
    JMP       @_main
 ;
    .END
```

```
/**************************************************************************************************************/
/*                                                                                                            */
/*  H8/300L Super Low Power Series                                                                            */
/*      -H8/38024 Series-                                                                                     */
/*  Application Note                                                                                          */
/*                                                                                                            */
/*  'MultiProcessor Communications'                                                                           */
/*                                                                                                            */
/*  Function                                                                                                  */
/*  : Serial Communication Interface                                                                          */
/*     -Multi-Processor Communication                                                                         */
/*                                                                                                            */
/*  External Clock : 16MHz                                                                                    */
/*  Internal Clock : 5MHz                                                                                     */
/*  Sub Clock      : 32.768kHz                                                                                */
/*                                                                                                            */
/**************************************************************************************************************/

#include    <machine.h>

/**************************************************************************************************************/
/*  Symbol Definition                                                                                         */
/**************************************************************************************************************/
struct BIT {
    unsigned char   b7:1;       /* bit7 */
    unsigned char   b6:1;       /* bit6 */
    unsigned char   b5:1;       /* bit5 */
    unsigned char   b4:1;       /* bit4 */
    unsigned char   b3:1;       /* bit3 */
    unsigned char   b2:1;       /* bit2 */
    unsigned char   b1:1;       /* bit1 */
    unsigned char   b0:1;       /* bit0 */
};

#define     SMR        *(volatile unsigned char *)0xFFA8         /* Serial Mode Register                      */
#define     SMR_BIT    (*(struct BIT *)0xFFA8)                   /* Serial Mode Register                      */
#define     COM        SMR_BIT.b7                                /* Communication Mode                        */
#define     CHR        SMR_BIT.b6                                /* Character Length                          */
#define     PE         SMR_BIT.b5                                /* Parity Enable                             */
```

```
#define     PM         SMR_BIT.b4                                /* Parity Mode                              */
#define     STOP       SMR_BIT.b3                                /* Stop Bit Length                          */
#define     MP         SMR_BIT.b2                                /* Multiprocessor Mode                      */
#define     CKS1       SMR_BIT.b1                                /* Clock Select 1                           */
#define     CKS0       SMR_BIT.b0                                /* Clock Select 0                           */
#define     BRR        *(volatile unsigned char *)0xFFA9         /* Bit Rate Register                        */
#define     SCR3       *(volatile unsigned char *)0xFFAA         /* Serial Control Register 3                */
#define     SCR3_BIT   (*(struct BIT *)0xFFAA)                   /* Serial Control Register 3                */
#define     TIE        SCR3_BIT.b7                               /* Transmit Interrupt Enable                */
#define     RIE        SCR3_BIT.b6                               /* Receive Interrupt Enable                 */
#define     TE         SCR3_BIT.b5                               /* Transmit Enable                          */
#define     RE         SCR3_BIT.b4                               /* Receive Enable                           */
#define     MPIE       SCR3_BIT.b3                               /* Multiprocessor Interrupt Enable          */
#define     TEIE       SCR3_BIT.b2                               /* Transmit End Interrupt Enable            */
#define     CKE1       SCR3_BIT.b1                               /* Clock Enable 1                           */
#define     CKE0       SCR3_BIT.b0                               /* Clock Enable 0                           */
#define     TDR        *(volatile unsigned char *)0xFFAB         /* Transmit Data Register                   */
#define     SSR        *(volatile unsigned char *)0xFFAC         /* Serial Status Register                   */
#define     SSR_BIT    (*(struct BIT *)0xFFAC)                   /* Serial Status Register                   */
#define     TDRE       SSR_BIT.b7                                /* Transmit Data Register Empty             */
#define     RDRF       SSR_BIT.b6                                /* Receive Data Register Full               */
#define     OER        SSR_BIT.b5                                /* Overrun Error                            */
#define     FER        SSR_BIT.b4                                /* Framing Error                            */
#define     PER        SSR_BIT.b3                                /* Parity Error                             */
#define     TEND       SSR_BIT.b2                                /* Transmit End                             */
#define     MPBR       SSR_BIT.b1                                /* Multiprocessor Bit Receive               */
#define     MPBT       SSR_BIT.b0                                /* Multiprocessor Bit Transfer              */
#define     SPCR       *(volatile unsigned char *)0xFF91         /* Transmit Data Register                   */
#define     SPCR_BIT   (*(struct BIT *)0xFF91)                   /* Port Mode Register 1                     */
#define     SPC32      SPCR_BIT.b5                               /* TXD Output Terminal                      */
#define     RDR        *(volatile unsigned char *)0xFFAD         /* Receive data Register                    */

/*************************************************************************************************************/
/*  Function define                                                                                          */
/*************************************************************************************************************/
extern void INIT ( void );                                       /* SP Set                                   */
void        main ( void );

/*************************************************************************************************************/
/*  RAM define                                                                                               */
/*************************************************************************************************************/
unsigned char   STD[4];

/*************************************************************************************************************/
/*  Vector Address                                                                                           */
/*************************************************************************************************************/
#pragma section     V1                                           /* Vector Section Set                       */
void (*const VEC_TBL1[])(void) = {                               /* 0x0000 - 0x000F                          */
    INIT                                                         /* 0x0000 Reset Vector                      */
};

#pragma section                                                  /* P                                        */
```

```
/*************************************************************************************************************/
/*  Main Program                                                                                             */
/*************************************************************************************************************/
void main ( void )
{
    unsigned char   *sptr;
    unsigned char   counter;

    sptr = &STD[0];                                             /* Initialize Serial Transmitting Data      */
                                                                /*                                  Address */
    STD[0] = 0x01;                                              /* Set Serial Transfer Data 0               */
    STD[1] = 0xB8;                                              /* Set Serial Transfer Data 1               */
    STD[2] = 0x02;                                              /* Set Serial Transfer Data 2               */
    STD[3] = 0xDE;                                              /* Set Serial Transfer Data 3               */

    TE = 0;                                                     /* Clear Serial Transmitting                */
    SCR3 = 0x01;                                                /* Initialize Serial Control Register 3     */

    SMR = 0x04;                                                 /* Initialize Serial Mode Register          */

    BRR = 0x04;                                                 /* Initialize Bit Rate  Register            */
    for(counter = 0; counter < 27; counter++);                  /* dummy Wait                               */

    OER = 0;                                                    /* Clear OER                                */
    FER = 0;                                                    /* Clear FER                                */
    PER = 0;                                                    /* Clear PER                                */

    TDRE = 0;                                                   /* Clear TDRE                               */
    TEND = 0;                                                   /* Clear TEND                               */

    SPCR = 0xE0;                                                /* Initialize Output Port TXD               */

    TE = 1;                                                     /* Start Serial Transmitting                */

    MPBT = 0;                                                   /* Clear Multiprocessor Bit Transfer        */

    for(counter = 0; counter < 4; counter++){                   /* Serial Transmitting Data Counter 4 Loop  */

        while(TDRE ! = 1){                                      /* End Serial Transmitting                  */
            ;
        }

        MPBT = ~MPBT;                                           /* Initialize Multiprocessor Bit Transfer   */

        TDR = *sptr;                                            /* Save Serial Transmitting Data            */

        sptr++;                                                 /* Increment Serial Transmitting Data       */
                                                                /*                                  Address */
    }

    while(TEND ! = 1){                                          /* End Serial Transmitting                  */
        ;
    }
```

```
        TE = 0;                                                      /* Initialize Transmitting Enable          */

        while(1){
            ;
        }
    }
```

Link address specifications

Section Name	Address
CV1	H'0000
P	H'0100
B	H'FB80

Revision Record

		Description	
Rev.	Date	Page	Summary
1.00	Dec.19.03	—	First edition issued

Keep safety first in your circuit designs!

1. Renesas Technology Corp. puts the maximum effort into making semiconductor products better and more reliable, but there is always the possibility that trouble may occur with them. Trouble with semiconductors may lead to personal injury, fire or property damage.
   Remember to give due consideration to safety when making your circuit designs, with appropriate measures such as (i) placement of substitutive, auxiliary circuits, (ii) use of nonflammable material or (iii) prevention against any malfunction or mishap.

Notes regarding these materials

1. These materials are intended as a reference to assist our customers in the selection of the Renesas Technology Corp. product best suited to the customer's application; they do not convey any license under any intellectual property rights, or any other rights, belonging to Renesas Technology Corp. or a third party.
2. Renesas Technology Corp. assumes no responsibility for any damage, or infringement of any third-party's rights, originating in the use of any product data, diagrams, charts, programs, algorithms, or circuit application examples contained in these materials.
3. All information contained in these materials, including product data, diagrams, charts, programs and algorithms represents information on products at the time of publication of these materials, and are subject to change by Renesas Technology Corp. without notice due to product improvements or other reasons. It is therefore recommended that customers contact Renesas Technology Corp. or an authorized Renesas Technology Corp. product distributor for the latest product information before purchasing a product listed herein.
   The information described here may contain technical inaccuracies or typographical errors. Renesas Technology Corp. assumes no responsibility for any damage, liability, or other loss rising from these inaccuracies or errors.
   Please also pay attention to information published by Renesas Technology Corp. by various means, including the Renesas Technology Corp. Semiconductor home page (http://www.renesas.com).
4. When using any or all of the information contained in these materials, including product data, diagrams, charts, programs, and algorithms, please be sure to evaluate all information as a total system before making a final decision on the applicability of the information and products. Renesas Technology Corp. assumes no responsibility for any damage, liability or other loss resulting from the information contained herein.
5. Renesas Technology Corp. semiconductors are not designed or manufactured for use in a device or system that is used under circumstances in which human life is potentially at stake. Please contact Renesas Technology Corp. or an authorized Renesas Technology Corp. product distributor when considering the use of a product contained herein for any specific purposes, such as apparatus or systems for transportation, vehicular, medical, aerospace, nuclear, or undersea repeater use.
6. The prior written approval of Renesas Technology Corp. is necessary to reprint or reproduce in whole or in part these materials.
7. If these products or technologies are subject to the Japanese export control restrictions, they must be exported under a license from the Japanese government and cannot be imported into a country other than the approved destination.
   Any diversion or reexport contrary to the export control laws and regulations of Japan and/or the country of destination is prohibited.
8. Please contact Renesas Technology Corp. for further details on these materials or the products contained therein.