To our customers,

Old Company Name in Catalogs and Other Documents

On April 1st, 2010, NEC Electronics Corporation merged with Renesas Technology Corporation, and Renesas Electronics Corporation took over all the business of both companies. Therefore, although the old company name remains in this document, it is a valid Renesas Electronics document. We appreciate your understanding.

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

April 1st, 2010
Renesas Electronics Corporation

Issued by: Renesas Electronics Corporation (http://www.renesas.com)

Send any inquiries to http://www.renesas.com/inquiry.

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

Simultaneous Transmission/Reception in Asynchronous Mode

Introduction

Using the serial data transfer function in asynchronous mode, four bytes of 8-bit data are simultaneously transmitted/received. The data transfer format for transmit data is set to eight bits for the data length, an odd parity, and one bit for the stop bit length. Data is sent at the bit rate of 31250 bps. Data transmitting and receiving is completed when 4-byte data is transmitted/received.

Target Device

H8/38024

Contents

1. Specifications

1. Using the serial data transfer function in asynchronous mode, four bytes of 8-bit data are simultaneously transmitted/received as shown in figure 1.1.
2. The data transfer format for transmit data is set to eight bits for the data length, an odd parity, and one bit for the stop bit length.
3. Data is sent at the bit rate of 31250 bps. Data transmitting and receiving is completed when 4-byte data is transmitted/received.

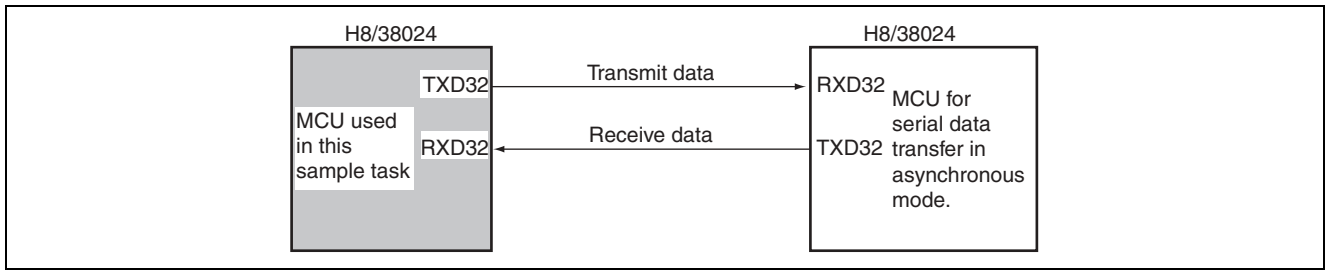
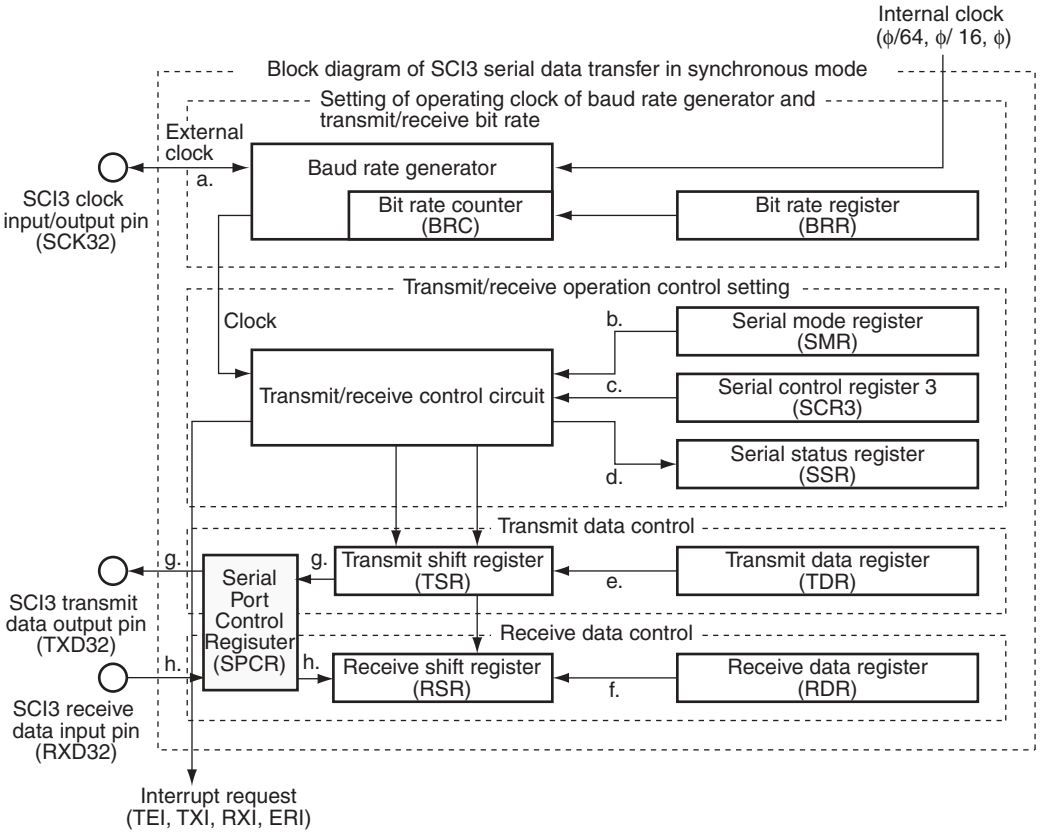


Figure 1.1 Simultaneous Serial Data Transmission/Reception in Asynchronous Mode

2. Description of Functions

1. In this sample task, a Serial Communication Interface (SCI) is used for simultaneous serial data transmission/reception in asynchronous mode. Figure 2.1 shows a block diagram of simultaneous serial data transmission/reception in asynchronous mode which is described below.
   - In asynchronous mode, serial data communication is performed asynchronously, with synchronization provided character by character.
   - Serial data can be communicated with standard asynchronous communication LSIs such as Universal Asynchronous Receiver/Transmitter (UART) and Asynchronous Communication Interface Adapter (ACIA).
   - A multi-processor communication function is provided to enable serial data communications with multiple processors.
   - The transfer format can be selected from 16 format types.
   - The transmitter and receiver are independent, enabling simultaneous transmission and reception. Both the transmitter and receiver have a double-buffer architecture to achieve continuous transmission and reception.
   - Any desired bit rate can be selected using an on-chip baud rate generator.
   - The transmit/receive clock source can be selected from internal or external clocks.
   - There are six interrupt factors, namely, transmit complete, transmit data empty, receive data full, overrun error, framing error and parity error.
   - The Receive Shift Register (RSR) is a register to receive serial data. Serial data input from RXD32 pin is set in RSR in the receiving order that is starting from the LSB (Bit 0), and is converted into parallel data. When one-byte data is received, the data is transferred automatically to RDR. RSR cannot be read from or written to directly by the CPU.
   - The Receive Data Register (RDR) is an 8-bit register to store received serial data. Upon receiving one-byte data, the received data is transferred from RSR to RDR to complete receive operation. RSR is then ready to receive data. RSR and RDR have a double buffer, 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 to transmit serial data. Transmit data is temporarily transferred from TDR to TSR and is sent to TXD32 pin starting from the LSB (Bit 0) for serial data transmission. Transmitting one-byte data, the next transmit data is transferred automatically from TDR to TSR to start transmitting. 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 to store transmit data. Detecting that TSR is "empty", transmit data written in TDR is transferred to TSR to start serial data transmission. By writing next transmit data in TDR while transmitting serial data of TSR, continuous transmission is possible. TDR can always be read from or written to the CPU.

— The Serial Mode Register (SMR) is an 8-bit register for setting of a serial data transfer format and selection of a 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 for selecting 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.



Notes: a. Matched to the operating clock source for baud rate generator selected by SMR, the transmit/receive bit rate is set. The transmit bit rate is set at 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 of 8 bits, with the parity mode of odd parity and the stop bit length of one bit, and the clock source for the on-chip baud rate generator is set to φ clock.
c. The transmit/receive operation, clock output in asynchronous mode and interrupt request enable/disable are selected. In this sample task, communication mode is set to asynchronous mode for clock output, clock source is set to internal clock, and SCK32 pin is set to function as clock output pin. Interrupt requests by transmit data empty, receive data full, and receive error are disabled.
d. The status flags (transmit data register empty, receive data register full, overrun error, framing error, parity error and transmission end) indicate operation status of SCI3.
e. Detecting that TSR is "empty", transmit data written in TDR is transferred to TSR.
f. Received data is transferred from RSR to RDR when receiving of one-byte data is completed.
g. Transmit data
h. Receive data

Figure 2.1 Block Diagram of Simultaneous Serial Data Transmission/Reception in Asynchronous Mode

— The Serial Status Register (SSR) is an 8-bit register with an on-chip status flag indicating operation status of SCI3 and on-chip multi-processor bits. SSR can always be read from or written to by the CPU, except 1 cannot be written in TDRE, RDRF, OER, PER or FER. 1 must be read in advance to clear them by writing 0. TEND and MPBR are for read-only and data cannot be written in them.

— The Bit Rate Register (BRR) is an 8-bit register to set a transmit/receive bit rate matched to 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 2.1 shows an example of BRR setting in asynchronous mode. Table 2.1 shows values in the active mode when OSC is 10 MHz.

Table 2.1 Example of BRR Settings for Bit Rates (Asynchronous Mode)

R Bit Rate (Bps)	110	150	200	250	1200	2400	31250
n	2	2	2	2	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. Set errors to be less than 1%.

2. BRR set values can be calculated as follows:

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

B: Bit rate (bps)
N: Set value of baud rate generator BRR (0 ≤ N ≤ 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 ≤ n ≤ 3)
(See Table 2 for the relation between n and clock.)

Table 2.2 Relation between n and Clock

		Set Value of SMR	
N	**Clock**	**CKS1**	**CKS0**
0	ϕ	0	0
1	ϕ w /4, ϕ w	0	1
2	ϕ/16	1	0
3	ϕ/64	1	1

3. The error shown in Table 1 is given by the following equation (rounded off to two decimals)

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

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

— In asynchronous mode, serial communication is performed with synchronization provided character by character, transmitting and receiving characters added with a start bit indicating the start of communication and a stop bit indicating the end of communication.
— The transmitter and receiver are independent inside SCI3 and full duplex communications are possible. Both the transmitter and receiver have a double-buffer architecture, therefore it is possible to perform data writing during transmission and data reading during reception, and continuous transmission and reception is then possible.
— Figure 2.2 shows data format of asynchronous communications. In asynchronous communications, the communication line is normally maintained in the mark state ("High" level). SCI3 monitors communication line and starts serial communications when it detects the place which has become a space ("Low" level) to serve as a start bit.
— One character in communication data consists of the start bit ("Low" level), followed by transmit/receive data (LSB first, starting from the least significant bit), parity bit ("High" or "Low" level) and stop bit ("High" level) at the end.
— In asynchronous mode, synchronization is achieved by the falling edge of the start bit during reception. Data is sampled on the eighth clock of a frequency obtained by multiplying 16 times the one bit period and communication data is fetched in the center of each bit.

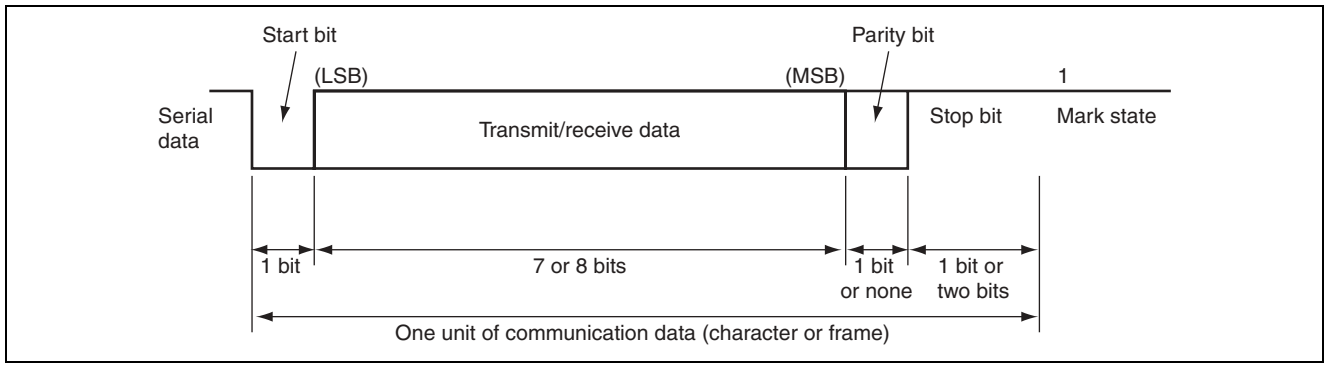


Figure 2.2 Data Format in Asynchronous Communications

— SCI3 clock (SCK32) is a clock input/output pin of SCI3.
— SCI3 receive data input (RXD32) is a receive data input pin of SCI3.
— SCI3 transmit data output (TXD32) is a transmit data output pin of SCI3.
— SCI3 interrupt factors total six, transmit complete, transmit data empty, receive data full and three receive errors (overrun error, framing error and parity error). Common vector address is assigned to them.
— Each interrupt request can be enabled/disabled by TIE and RIE in SCR3.
— If TDRE in SSR is set to 1, TXI is generated. If TEND in SSR is set to 1, 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 a transmit data empty interrupt request (TXI) before transferring transmit data 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 a transmit end interrupt request (TEI) before transferring transmit data to TDR, TEI is generated even when transmit data is not sent.
— By processing which transfer transmit data to TDR within the interrupt handling routine, these interrupts can be utilized effectively. To prevent these interrupt requests (TXI and TEI), the enable bits (TIE and TEIE) interacting to these interrupt requests should 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 2.3 shows assignment of functions in this sample task. Serial data transmission in asynchronous mode is performed by assigning the functions as shown in table 2.3.

Table 2.3 Assignment of Functions

Function	Assignment
TSR	A register to transmit serial data
TDR	A register to store transmit data
RSR	A register to receive serial data
RDR	A register to store receive data
SMR	Sets a serial data transfer format and clock source for the baud rate generator
SSR	Status flags to indicate operation status of SCI3
BRR	Sets transmit/receive bit rate
SCR3	Enables transmit/receive operation, sets TXD32 output pin, and sets RXD32 input pin
TXD32	SCI3 transmit data output pin
RXD32	SCI3 receive data input pin
SPCR	Sets TXD32 output pin

3. Principle of Operation

1. Figure 3.1 illustrates the principle of operation of this sample task. Serial data is simultaneously transmitted and received in asynchronous mode by hardware and software processing as shown in figure 3.1.

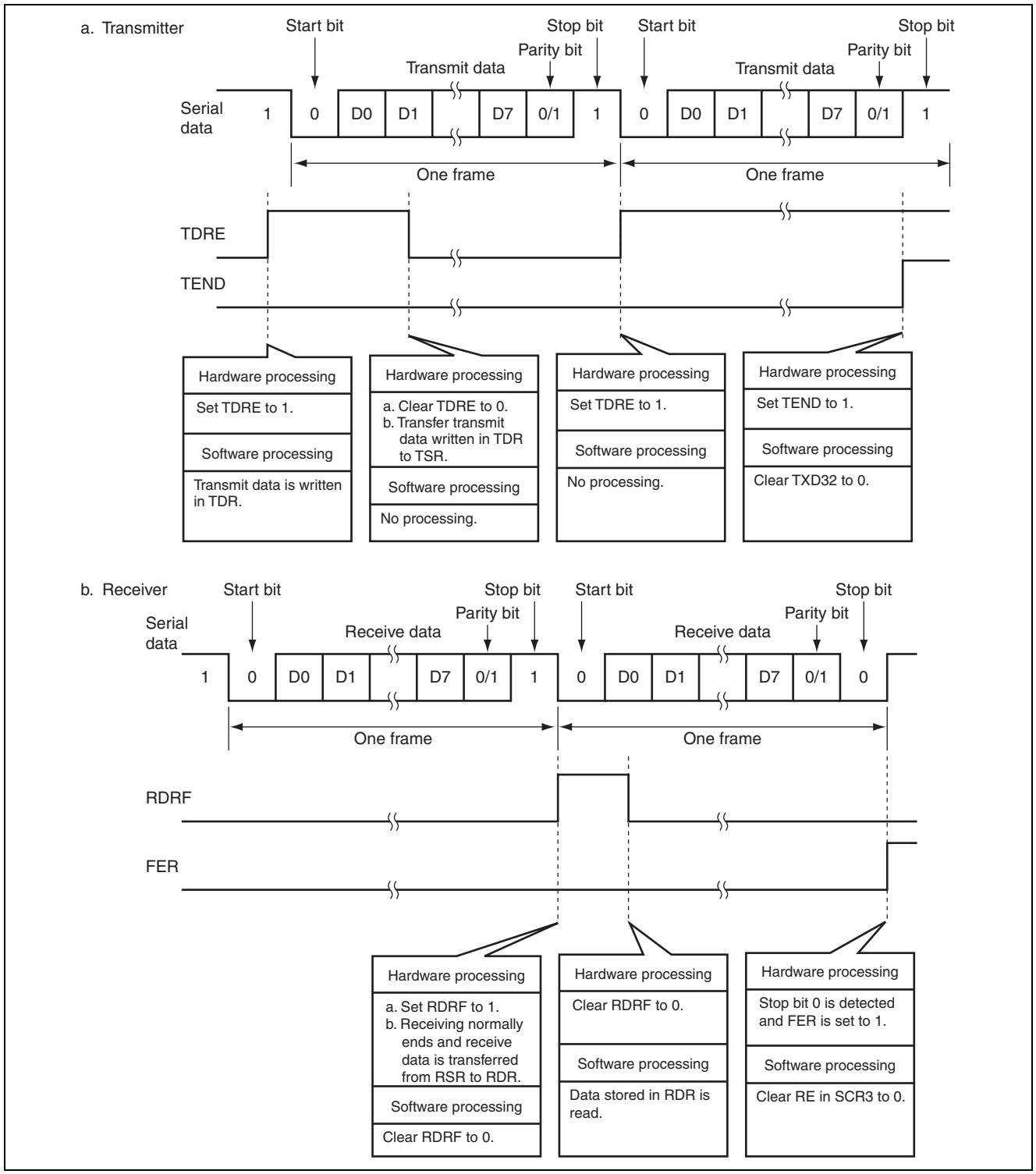


Figure 3.1 Operation Principle of Simultaneous Serial Data Transmission and Reception in Asynchronous Mode

4. Description of Software

4.1 Modules

Table 4.1 describes the modules in this sample task.

Table 4.1 Description of Modules

Module	Label	Function
Main Routine	main	Serial data transmission and reception in asynchronous mode setting, branching to receive error processing subroutine if receive error occurs, and end of operation when 4-byte data is transmitted/received.
Receive Error Handling	er-sub	Judges which error has occurred, OER, FER or PER, and processes the error accordingly.

4.2 Arguments

Table 4.2 describes the arguments used in this sample task.

Table 4.2 Description of Arguments

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

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 in asynchronous mode is set to 8-bit data. If CHR = 1, the data length in asynchronous mode is set to 7-bit data.	H'FFA8 Bit 6	0
	PE	Serial Mode Register (Parity Enable) If PE = 0, parity bit addition and check are disabled during transmission in asynchronous mode. If PE = 1, parity bit addition and check are enabled during transmission in asynchronous mode.	H'FFA8 Bit 5	1
	PM	Serial Mode Register (Parity Mode) If PM = 0, parity addition and check are set to even parity. If PM = 1, parity addition and check are set to odd parity.	H'FFA8 Bit 4	1

Register		Function	Address	Setting
SMR	STOP	Serial Mode Register (Stop Bit Length) If STOP = 0, the stop bit length in asynchronous mode is set to one bit. If STOP = 1, the stop bit length in asynchronous mode is set to two bits.	H'FFA8 Bit 3	0
	MP	Serial Mode Register (Multi-Processor Mode) If MP = 0, the multi-processor communication function is disabled. If MP = 1, the multi-processor communication function is enabled.	H'FFA8 Bit 2	0
	CKS1 CKS0	Serial Mode Register (Clock Select 1, 0) If CKS1 = 0 and CKS0 = 0, the clock source for the on-chip baud rate generator is set to ϕ clock.	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, transmit operation is disabled. If TE = 1, transmit operation is enabled.	H'FFAA Bit 5	0
	RE	Serial Control Register 3 (Receive Enable) If RE = 0, receive operation is disabled. If RE = 1, receive operation is enabled.	H'FFAA Bit 4	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 to store transmit data.	H'FFAB	—
SSR	TDRE	Serial Status Register (Transmit Data Register Empty) If TDRE = 0, transmit data written in TDR is not transferred to TSR. If TDRE = 1, transmit data is not written in TDR or transmit data written in TDR is transferred to TSR.	H'FFAC Bit 7	1
	RDRF	Serial Status Register (Receive Data Register Full) If RDRF = 0, receive data is not stored in RDR. If RDRF = 1, receive data is stored in RDR.	H'FFAC Bit 6	1
	OER	Serial Status Register (Overrun Error) If OER = 0, data is being received or reception has been completed. If OER = 1, an overrun error has occurred during data reception.	H'FFAC Bit 5	0
	FER	Serial Status Register (Framing Error) If FER = 0, data is being received or reception has been completed. If FER = 1, a framing error has occurred during data reception.	H'FFAC Bit 4	0
	PER	Serial Status Register (Parity Error) If PER = 0, data is being received or reception has been completed. If PER = 1, a parity error has occurred during data reception.	H'FFAC Bit 3	0

Register		Function	Address	Setting
SSR	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	—
RDR		Receive Data Register An 8-bit register to store receive data.	H'FFAD	—
SPCR	SPC32	Serial Port Control Register (P42/TXD32 Pin Function Switch) If SPC32 = 0, P42/TXD32 pin is set to P42 pin function. If SPC32 = 1, P42/TXD32 pin is set to TXD32 pin function.	H'FF91 Bit 5	1
	SCINV3	Serial Port Control Register (TXD32 Pin Output Data Inversion Switch) If SCINV3 = 0, TXD32 output data in not inverted. If SCINV3 = 1, TXD32 output data in inverted.	H'FF91 Bit 3	0
	SCINV2	Serial port Control Register (RXD32 Pin Input Data Inversion Switch) If SCINV2 = 0, RXD32 input data in not inverted. If SCINV2 = 1, RXD32 input data in inverted.	H'FF91 Bit 2	0

4.4 Description of RAM

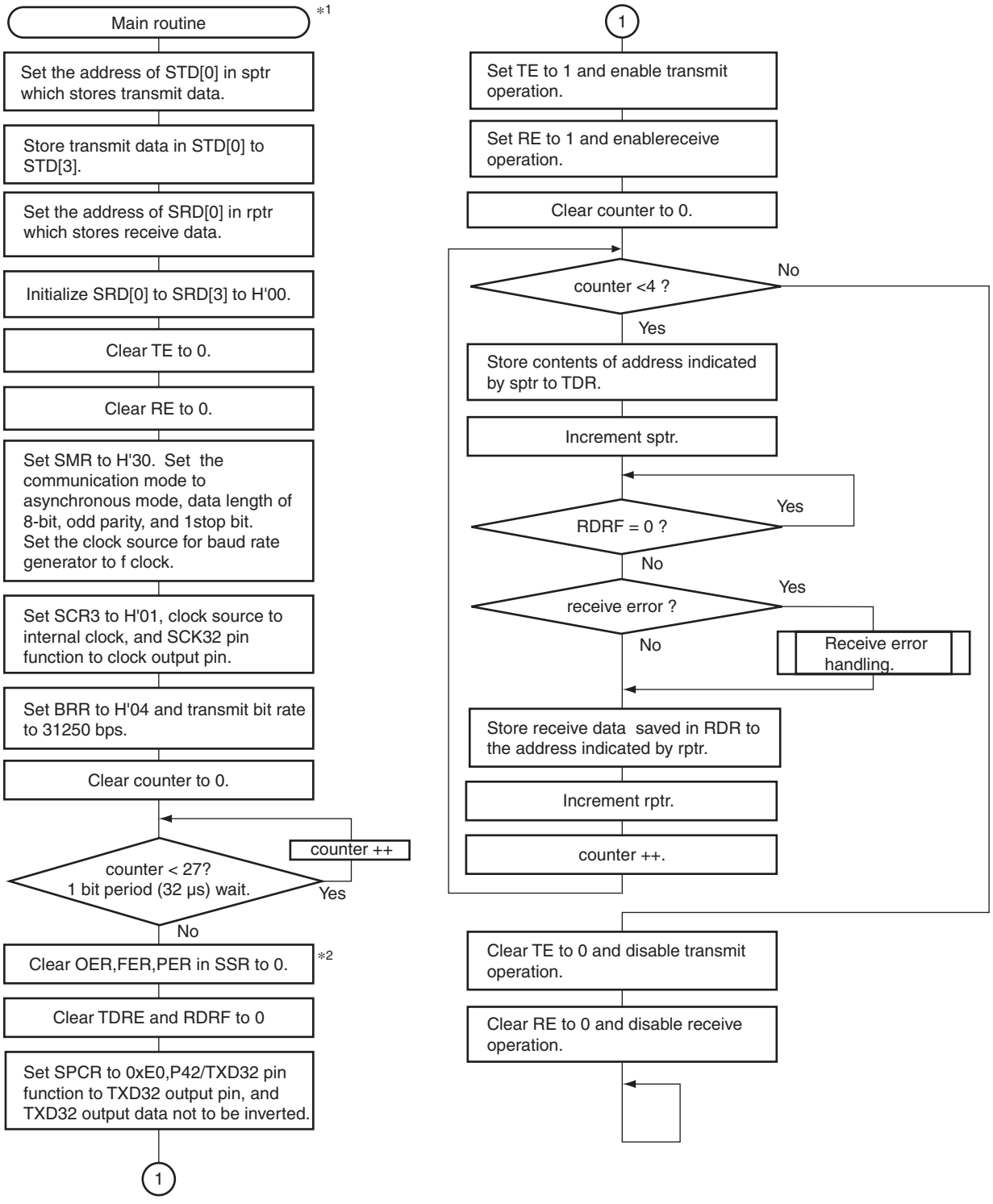
Table 4.4 describes the RAMs used in this sample task.

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
SRD[0]	Stores the first byte of serial receive data in asynchronous mode.	H'FB84	Main Routine
SRD[1]	Stores the second byte of serial receive data in asynchronous mode.	H'FB85	Main Routine
SRD[2]	Stores the third byte of serial receive data in asynchronous mode.	H'FB86	Main Routine
SRD[3]	Stores the fourth byte of serial receive data in asynchronous mode.	H'FB87	Main Routine

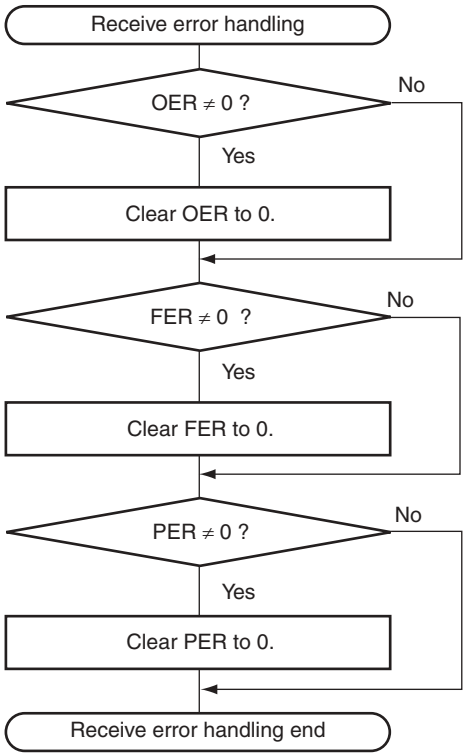
5. Flowchart

1. Main routine



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

2. Subroutine



6. Program Listing

INIT.SRC (Program listing)

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

```
/************************************************************************************************************************/
/*                                                                                                                      */
/*  H8/300L Super Low Power Series                                                                                      */
/*      -H8/38024 Series-                                                                                               */
/*  Application Note                                                                                                    */
/*                                                                                                                      */
/*  'Asynchronous Serial Data Simultaneous                                                                              */
/*   Transmission and Reception'                                                                                        */
/*                                                                                                                      */
/*  Function                                                                                                            */
/*  : Serial Communication Interface                                                                                    */
/*     Asynchronous Serial Interface                                                                                    */
/*     -Transmitting/Receiving                                                                                          */
/*                                                                                                                      */
/*  External Clock : 16MHz                                                                                              */
/*  Internal Clock : 16MHz                                                                                              */
/*  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 );
void            er_sub ( void );

/************************************************************************************************************************/
/*  RAM define                                                                                                          */
/************************************************************************************************************************/
unsigned char   STD[4];
unsigned char   SRD[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   stus;
    unsigned char   *sptr,*rptr;
    unsigned char   counter;

    sptr = &STD[0];                                     /* Initialize Serial Transmitting           */
                                                        /*                              Data Address */

    STD[0] = 0x00;                                      /* Set Serial Transfer Data 0               */
    STD[1] = 0x55;                                      /* Set Serial Transfer Data 1               */
    STD[2] = 0xAA;                                      /* Set Serial Transfer Data 2               */
    STD[3] = 0xbb;                                      /* Set Serial Transfer Data 3               */

    rptr = &SRD[0];                                     /* Initialize Serial Receiving Data Address */
    SRD[0] = 0x00;                                      /* Initialize Serial Receiving Data 0       */
    SRD[1] = 0x00;                                      /* Initialize Serial Receiving Data 1       */
    SRD[2] = 0x00;                                      /* Initialize Serial Receiving Data 2       */
    SRD[3] = 0x00;                                      /* Initialize Serial Receiving Data 3       */

    TE = 0;                                             /* Clear Serial Transmitting                */
    RE = 0;                                             /* Clear Serial Receiving                   */

    SMR = 0x30;                                         /* Initialize Serial Mode Register          */

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

    BRR = 4;                                            /* 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                               */
    RDRF = 0;                                           /* Clear RDRF                               */
    TEND = 0;                                           /* Clear TEND                               */

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

    TE = 1;                                             /* Start Serial Transmitting                */
    RE = 1;                                             /* Start Serial Receiving                   */

    for(counter = 0; counter < 4; counter++){           /* Serial Transmitting/Receiving            */
                                                        /*                      Data Counter 4 Loop */
        TDR = *sptr;                                    /* Save Serial Transmitting Data            */
        sptr++;                                         /* Increment Serial Transmitting            */
                                                        /*                              Data Address */

        while(RDRF == 0){                               /* End Serial Receive End ?                 */
            ;
        }

        if((SSR & 0x38) != 0){                          /* Error Flag = 1 ?                         */
            er_sub();
        }
```

```
        else{
            *rptr = RDR;                                /* Save Serial Receiving Data              */
        }

        rptr++;                                         /* Increment Serial Receiving Data Address */
    }

    TE = 0;                                             /* Initialize Transmitting Enable          */
    RE = 0;                                             /* Initialize Receiving Enable             */

    while(1){
        ;
    }
}

void er_sub(void)
{
    if(OER != 0){
        OER = 0;                                        /* Clear OER                               */
    }
    if(FER != 0){
        FER = 0;                                        /* Clear FER                               */
    }
    if(PER != 0){
        PER = 0;                                        /* Clear PER                               */
    }
}
```

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.