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.

32176 Group

Application Using the Serial Interface (CSIO Mode)

1. Overview

This documentation describes a sample task using 32176 Group microcomputer's on-chip serial interface (serial I/O).

2. Introduction

The example application described in this documentation uses the following microcomputer under the respective conditions.

- Microcomputer: 32176 Group (M32176FnVFP, M32176FnTFP)
- Operating frequency: 20 to 40 MHz (The sample program is compiled assuming a frequency of 40 MHz)
- Operating Board: Starter kit for 32176 Group

3. Explanation of the Technology Applied

3.1 Outline of the Serial Interface

The 32176 contains a total of four serial interface channels, SIO0, SIO1, SIO2, and SIO3. SIO0 and SIO1 can be selected between CSIO mode (clock-synchronous serial interface) and UART mode (clock-asynchronous serial interface). Channels SIO2 and SIO3 are UART mode only.

- CSIO Mode (clock-synchronous serial interface)
  Communication is performed synchronously with a transfer clock, using the same clock for both transmit and receive sides. The transfer data is 8 bits long (fixed).
- UART Mode (clock-asynchronous serial interface)
  Communication is performed at any transfer rate in any transfer data format. The transfer data length can be selected from 7, 8, or 9 bits.

Channels SIO0, SIO1, SIO2, and SIO3 each have transmit DMA transfer and a receive DMA transfer request. These serial interfaces, when combined with the internal DMA Controller (DMAC), allow serial communication to be performed at high speed, as well as reduce the data communication load of the CPU.

In the CSIO mode, the transfer rate when using the internal clock is determined by the clock divider frequency division value and the values set in the individual baud rate registers for each channel. The table below shows sample setting values for each baud rate.

Table 3.1.1 Sample Baud Rate Register Settings

Item / Baud Rate (bps)	When f (BCLK) 20 MHz		
	Clock Divider Frequency Division Value (Frequency Division)	BRG Setting Value	Actual Baud Rate (bps)
250	256	155	250.40
500	256	77	500.80
1000	256	38	1001.60
2500	32	124	2500.00
5000	8	249	5000.00
10000	8	124	10000.00
25000	8	49	25000.00
50000	1	199	50000.00
100000	1	99	100000.00
250000	1	39	250000.00
500000	1	19	500000.00
1000000	1	9	1000000.00
2000000	1	4	2000000.00

In the CSIO mode, the value to be set in the baud rate register can be calculated as follows:

$$\text{BRG setting value} = \frac{\text{f (BCLK)}}{\text{Baud rate (bps)} \times \text{clock divider frequency division value} \times 2} - 1$$

f (BCLK)	: Peripherial clock operating frequency
Baud rate register setting value (BRG)	: H'00 to H'FF
Clock divider frequency division value	: 1, 8, 32, 256

For details of the serial interface, refer to the 32176 Group User's Manual.

4. Sample Program Using the Serial Interface

4.1 Outline of the Sample Program

In this sample program, the SIO0 is for transmission using the internal clock, and SIO1 is for reception using an external clock. For transmission, data for the number of transmission bytes is transmitted from the address of the transmit data storage area specified by the parameter. For reception, interrupt processing is used. By connecting the SIO0 and the SIO1, CSIO transmission and reception are possible by using only the on-chip functions of the 32176.

In this sample program, the transmission rate is set to 500kbps. When the internal clock is selected 2Mbps, f(BCLK)/16 becomes the maximum transmission rates when the external clock is selected.

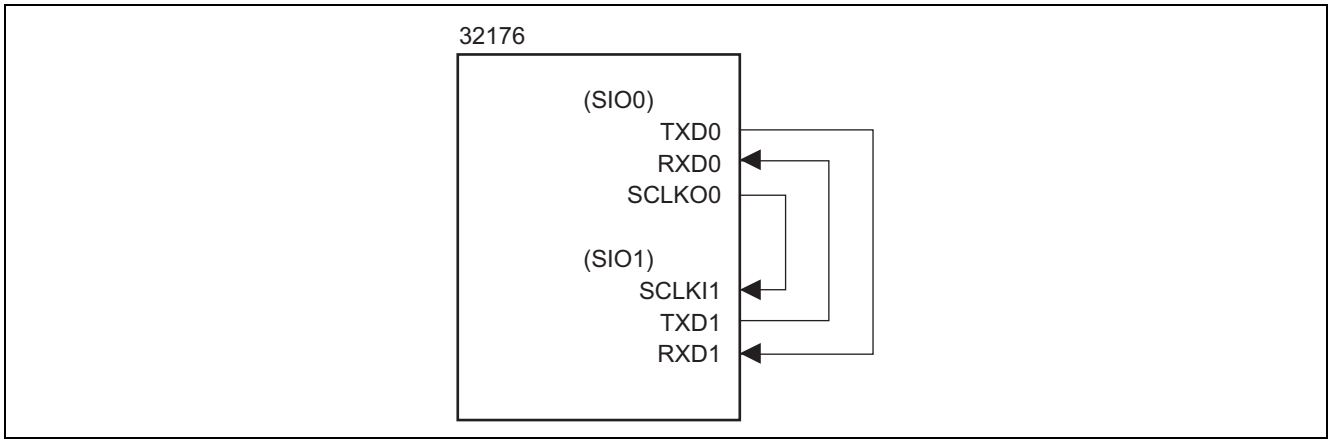
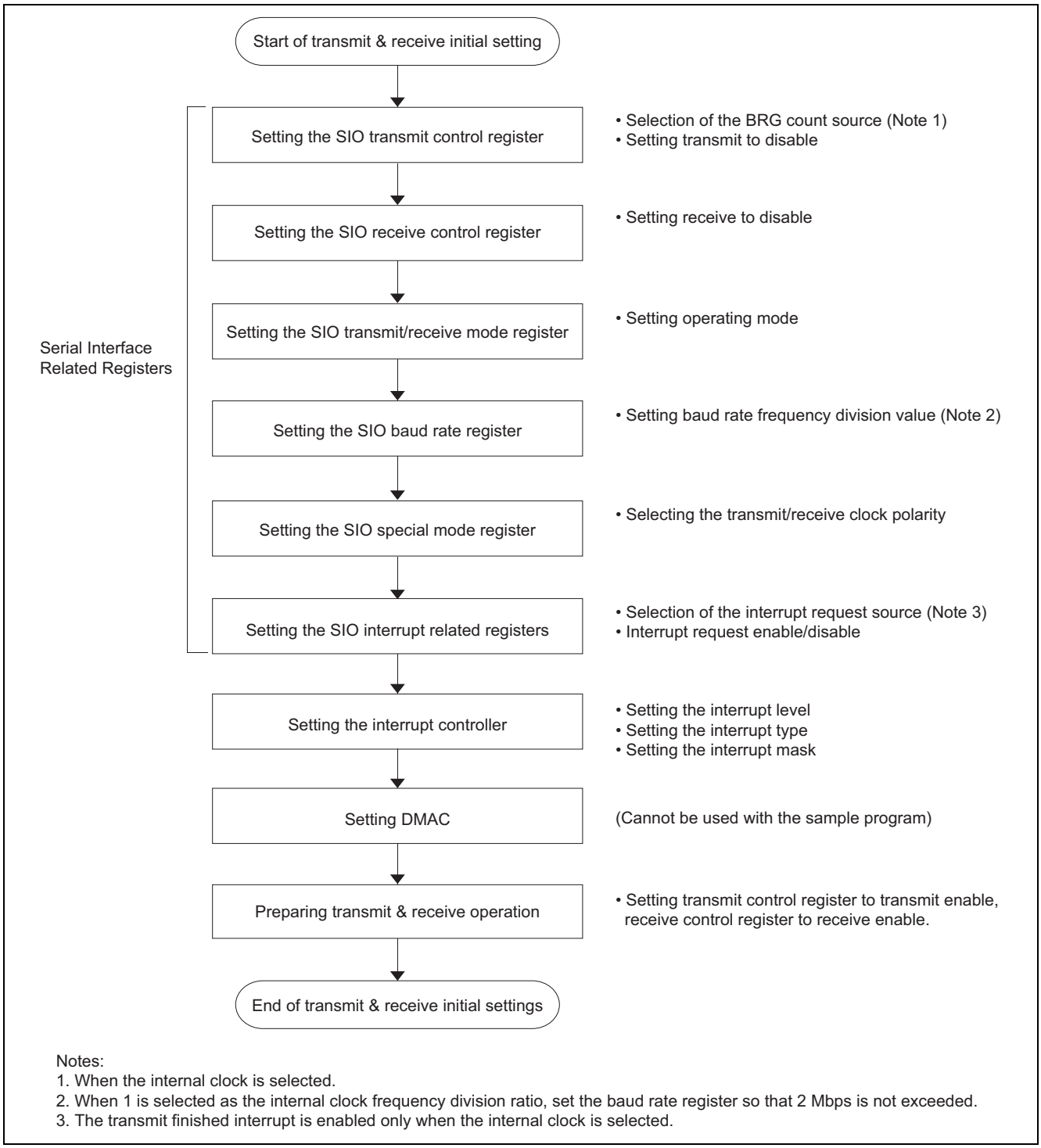


Figure 4.1.1 Connection Diagram

4.2 CSIO Transmit/Receive Processing

The flowchart for the initial settings of transmission and reception in the CSIO mode is shown in figure 4.2.1.



Notes:
1. When the internal clock is selected.
2. When 1 is selected as the internal clock frequency division ratio, set the baud rate register so that 2 Mbps is not exceeded.
3. The transmit finished interrupt is enabled only when the internal clock is selected.

Figure 4.2.1 Flowchart for the Initial Settings of Transmission and Reception in the CSIO Mode

4.3 Description of the Sample Program

Note: The registers used are indicated as (register name: bit name).

4.3.1 Various Initialization Function (init_func ())

(1) Call the function for port initialization.

4.3.2 Port Initialization Function (port_init ())

(1) Set the output port.

- Set the port input enable bit of the port input special function control register to input enabled. (PICNT: PIEN0)
- Initialize the P11 data register. (P11DATA)
- Set the P11 Direction Register to the output mode. (P11DIR)
- Set the P11 Operation Mode Register to general-purpose port. (P11MOD)

Note: If a Direction Register is set as output before setting up a Data Register, an unfixed value is outputted until writing will be performed to a Data Register.

4.3.3 Main Function (main ())

(1) Initialize local variables.

- Received data buffer pointer (rx_cnt1), previous received data buffer pointer (rx_cnt1_bak), and transmission data (send_data) are initialized to “0”.

(2) Call the function for disabling interrupts.
(3) Call the function for various initialization.
(4) Call the function for SIO0 transmit processing initial settings.
(5) Call the function for SIO1 receive processing initial settings.
(6) Call the function for enabling interrupts.
(7) Transmit/receive processing infinite loop.

- Write dummy data to the SIO1 (receiving side) transmit buffer register.
  (In the CSIO mode, the receive shift clock can be obtained by operating the transmission circuit. Accordingly, it is necessary to perform transmission operation even when only reception operation is required.)
- Transmit one byte of variable send_data from SIO0.
- Wait for the SIO1 receive interrupt.
- Retrieve received data from the SIO1 receive data buffer.
- Re-set variable rx_cnt1 to “0” when overflow occurs.
- Store the current received data buffer pointer.
- When SIO0 transmission data and SIO1 received data have the same value:
  1) Displays the received data on the LED.
  2) Increments the transmission data and start next transmission processing.
- When the SIO0 transmission data and SIO1 received data differ:
  1) Enters an infinite loop, and displays H'55 and H'AA alternatively on the LED.

4.3.4 Initial Setting Function for CSIO Transmission on SIO0 (SIO0_CLKTinit ())

(1) Setting the transfer mode.

- Set the SIO0 receive control register's receive enable bit to receive disabled. (S0RCNT: REN)
- Set the SIO0 transmit control register. (S0TCNT: CDIV, TEN)
  Set the BRG count source to f (BCLK), transmission disabled.
- Set the P8 operation mode registers. (P8MOD: P82MOD, P84MOD)
  Set port P82 to TXD0, and port P84 to SCLKI0/SCLKO0.
- Set the SIO0 transmit/receive mode register. (S0MOD: SMOD, CKS)
  Set the 8-bit clock-synchronous serial mode and to use the internal clock.
- Set the SIO0 baud rate frequency division value in the baud rate register. (S0BAUR)
  (At 500 kbps when CPU clock is 40 MHz)
- Set the transmit/receive clock polarity select bit of the SIO0 special mode register. (S0SMOD: CKPOL)
  Output transmission data on the falling edge of SCLK, and fetch received data on the rising edge of SCLK.

(2) Interrupt settings.

- Set the interrupt level in the SIO0 transmit interrupt control register to interrupt disabled.
  (ISIO0TXCR: ILEVEL)
- Set the interrupt level in the SIO0 receive interrupt control register to interrupt disabled.
  (ISIO0RXCR: ILEVEL)
- Set the SIO03 interrupt request mask registers. (SI03MASK: T0MASK, R0MASK)
  Disable SIO0 transmit interrupts, and SIO0 receive interrupts.

(3) Setting to enable transmission

- Set the transmit enable bit of the SIO0 transmit control register to transmit enabled (S0TCNT: TEN)

4.3.5 Initial Setting Function of CSIO receive for SIO1 (SIO1_CLKRinit ())

(1) Setting the transfer mode.

- Set the SIO1 receive control register's receive enable bit to receive disabled. (S1RCNT: REN)
- Set the SIO1 transmit control register (S1TCNT: CDIV, TEN)
  Set the BRG count source to f (BCLK), transmit disabled
- Set the P8 operation mode register. (P8MOD: P86MOD, P87MOD)
  Set port P86 to RXD1, and port P87 to SCLKI1/SCLKO1.
- Set the SIO1 transmit/receive mode register. (S1MOD: SMOD, CKS)
  Set the 8-bit clock-synchronous serial mode and to use the external clock.
- Set the transmit/receive clock polarity select bit of the SIO1 special mode register. (S1SMOD: CKPOL)
  Set to output transmission data on the falling edge of SCLK, fetch received data on the rising edge of SCLK.

(2) Interrupt settings.

- Set the interrupt level in the SIO1 transmit interrupt control register to interrupts disabled.
  (ISIO1TXCR: ILEVEL)
- Set the interrupt level in the SIO1 receive interrupt control register to "0" (highest priority).
  (ISIO1RXCR: ILEVEL)
- Set the SIO1 receive interrupt request source select bit of the SIO03 interrupt request source select register to reception finished interrupt. (SI03SEL: ISR1)
- Set the SIO03 interrupt request mask register. (SI03MASK: T1MASK, R1MASK)
  Disable SIO1 transmit interrupts, and enable SIO1 receive interrupts.

(3) Enable transmission and reception.

- Set the transmit enable bit of the SIO1 transmit control register to transmission enabled. (S1TCNT: TEN)
- Set the receive enable bit of the SIO1 receive control register to reception enabled. (S1RCNT: REN)

4.3.6 SIO0 CSIO Transmit Processing Function (SIO0_Tr ())

(1) Transmit processing.

- Confirm that the SIO0 transmit buffer is empty. (S0TCNT: TBE)
- Write one byte of transmission data to transmit buffer register. (S0TXB_L)
- Repeat transmission for a specified number of times.
- Wait for SIO0 transmission completion. (S0TCNT: TSTAT)

4.3.7 CSIO Receive Processing Function by SIO1 Receive Interrupt (SIO1_RcvInt ())

(1) Receive processing.

- Read the SIO1 receive control register (receive status). (S1RCNT)
- Read the SIO1 receive buffer register. (S1RXB_L)
- Re-read the SIO1 receive control register.
  For detecting overrun occurring between reading the receive status and reading the receive buffer register.
- Check the SIO1 receive error sum bit. (S1RCNT: ERS)
  1) When no error occurs, the received data is stored in the receive data buffer and the pointer is incremented.
  2) When an error occurs, the receive enable bit is first set to receive disabled, then set to receive enabled. (Each bit in the receive control registers is cleared.)

4.3.8 Startup Routine (startup.ms)

(1) Interrupt settings.

- Set the start address of the SIO1 receive interrupt routine (SIO1_RcvInt ()) in the SIO1 receive interrupt handler (H'0000 00CC) as the interrupt source in the ICU vector table.

4.4 Sample Program

The transmit/receive sample program for the CSIO mode is shown below.

Note that the sample program below requires the SFR definition file. The latest SFR definition file can be downloaded from Renesas Technology website. When using the SFR definitions file, adjust the path setting to match the operating computer environment.

4.4.1 csio_main.c

```
/*""FILE COMMENT""*****************************************************
 *      M32R C Programming              Rev. 1.01
 *               < Sample Program for 32176 >
 *               < Serial Interface (CSIO) (main routine) >
 *
 *      Copyright (c) 2004 Renesas Technology Corporation
 *                         All Rights Reserved
 *********************************************************************/

/*********************************************************************/
/*              Include file                                         */
/*********************************************************************/

#include                "..\inc\sfr32176_pragma.h"

/*********************************************************************/
/*              Function prototype declaration                       */
/*********************************************************************/

        void            main(void);                             /* Main function */
        void            init_func(void);                        /* Initial setup function */
        void            port_init(void);                        /* Initialize port */

/*********************************************************************/
/*              Externally referenced variable                       */
/*********************************************************************/

extern  void            DisInt( void );                         /* Interrupt disable function
*/
extern  void            EnInt( void );                          /* Interrupt enable function
*/

extern  void            SIO0_CLKTinit( void);                   /* Initialize SIO0 */
extern  void            SIO1_CLKRinit( void);                   /* Initialize SIO1 */
extern  void            SIO0_Tr( UCHAR *, ULONG);               /* Transmit from SIO0 */

/*              Externally referenced variable                          */
/*              Global variable                                         */
/*              Define macro                                            */

        volatile UCHAR  RcvBuf1[10];                            /* Receive buffer */
        volatile UCHAR  rx_cnt1;                                /* Pointer to receive buffer
*/

/*""FUNC COMMENT""*****************************************************
 * Function name: init_func()
 *--------------------------------------------------------------------
 * Description  : Call various initialization functions
 *--------------------------------------------------------------------
 * Argument     : -
 *--------------------------------------------------------------------
 * Returns      : -
 *--------------------------------------------------------------------
 * Notes        :
 *""FUNC COMMENT END""************************************************/
void init_func(void)
{
        port_init();                                            /* Initialize port */
}

/*""FUNC COMMENT""*****************************************************
 * Function name: port_init()
 *--------------------------------------------------------------------
```

```
 61   * Description  : Initialize port
 62   *------------------------------------------------------------------------
 63   * Argument     : -
 64   *------------------------------------------------------------------------
 65   * Returns      : -
 66   *------------------------------------------------------------------------
 67   * Notes        :
 68   *""FUNC COMMENT END""*************************************************/
 69  void port_init(void)
 70  {
 71          PICNT = PIEN0;                                          /* Enable port input */
 72
 73  /*** LED output port ***/
 74
 75          P11DATA = 0x00;                                         /*Output data (must be set
prior to mode)*/
 76          P11DIR = 0xff;                                          /* P110-P117 : Output mode */
 77          P11MOD = 0x00;                                          /* P110-P117 : Input/output
port */
 78  }
 79
 80  /*""FUNC COMMENT""*****************************************************
 81   * Function name: main()
 82   *------------------------------------------------------------------------
 83   * Description  : Input/output port Receives data (increment pattern) from SIO1 after being
transmitted in 500 Kbps
 84   *              : clock-synchronized serial mode (when the source clock frequency = 10 MHz)
from SIO0 and
 85   *              : if the received data is the same as transmitted, displays it on LED(port
11).
 86   *              : If unable to receive the same data, it displays 0x55 in blinking inverse
video.
 87   *------------------------------------------------------------------------
 88   * Argument     : -
 89   *------------------------------------------------------------------------
 90   * Returns      : -
 91   *------------------------------------------------------------------------
 92   * Notes        : Dummy transmission is required for data to be received from SIO1
 93   *""FUNC COMMENT END""*************************************************/
 94  void    main(void)
 95  {
 96  /*              +-------+
 97   *              |       |
 98   *              |   TXD0| --+
 99   *      (SIO0)  |   RXD0| <-|---+
100   *              | SCLKO0| ------|--+
101   *              |       |   |   |  |
102   *              |       |   |   |  |
103   *              |   TXD1| --|---+  |
104   *      (SIO1)  |   RXD1| <-+      |
105   *              | SCLKI1| <--------+
106   *              |       |
107   *              |       |
108   *              +-------+
109                  AR_GetPict( arpict );                  /* DRI start */
110
111          ULONG   j;
112          UCHAR   send_data;
113          UCHAR   rcv_data;
114          UCHAR   rx_cnt1_bak;
115
116          rx_cnt1 = 0;
117          rx_cnt1_bak = 0;
118
119          send_data = 0;
120
121          DisInt();                                               /* Disable interrupt */
122
123          init_func();                                            /* Initialize microcomputer
*/
124
125          SIO0_CLKTinit();                                        /* Initialize SIO0 */
126          SIO1_CLKRinit();                                        /* Initialize SIO1 */
127
128          EnInt();                                                /* Enable interrupt */
129
130          while(1) {
```

```
131                 S1TXB_L = rx_cnt1;                              /* Dummy write */
132                 SIO0_Tr( &send_data, 1ul);                      /* Send data from SIO0 */
133
134                 while( rx_cnt1 == rx_cnt1_bak){                 /* Wait for data to receive
*/
135                         ;
136                 }
137                 rcv_data = RcvBuf1[ rx_cnt1 - 1];               /* Read out received data */
138
139                 if( rx_cnt1 == 10) {                            /* Check for receive counter
overflow */
140                         rx_cnt1 = 0;
141                 }
142                 rx_cnt1_bak = rx_cnt1;
143
144                 if( rcv_data == send_data) {
145                         P11DATA = rcv_data;                     /* Display received
(transmitted) data */
146                         send_data++;                            /* Increment transmit data */
147                 } else {                                        /* Received data in error */
148                         P11DATA = 0x55;
149                         while(1) {
150                                 for( j = 0ul; j < 1000000ul; j++){      /* Wait */
151                                         ;
152                                 }
153                                 P11DATA ^= 0xffu;               /* Display 0x55 in blinking
inverse video*/
154                         }
155                 }
156         }
157  }
```

4.4.2 csio.c

```
/*""FILE COMMENT""*******************************************************
 *      M32R C Programming              Rev. 1.01
 *               < Sample Program for 32176 >
 *              < Serial Interface (CSIO) >
 *
 *      Copyright (c) 2004 Renesas Technology Corporation
 *                         All Rights Reserved
 ************************************************************************/

/************************************************************************/
/*              Include file                                            */
/************************************************************************/

#include                "..\inc\sfr32176_pragma.h"

/************************************************************************/
/*              Function prototype declaration                          */
/************************************************************************/

        void            SIO0_CLKTinit(void);                    /* Initialize SIO0 */
        void            SIO1_CLKRinit(void);                    /* Initialize SIO1 */
        void            SIO0_Tr( UCHAR *TrBuf, ULONG TrNum);    /* Send data from SIO0 */
        void            SIO1_RcvInt(void);                      /* SIO1 receive interrupt */

/*              Definition of external reference                        */
/*              Externally referenced variable                          */
/************************************************************************/

extern  volatile UCHAR  RcvBuf1[];                              /* Receive buffer */
extern  volatile UCHAR  rx_cnt1;                                /* Pointer to receive buffer
*/

/*              Externally referenced variable                          */
/*              Define macro                                            */
/************************************************************************/

/* Setting port operation mode */
                                                        /* 0123 4567    */
#define P8MOD_SCI0ClkTr         0x28u                   /* 0010 1000B   P8 operation mode
register      */
                                                        /* |||| |||+--- P87
*/
                                                        /* |||| ||+---- P86
*/
                                                        /* |||| |+----- P85
*/
                                                        /* |||| +------ SCLKI0/SCLKO0
*/
                                                        /* |||+-------- P83
*/
                                                        /* ||+--------- TXD0
*/
                                                        /* ++---------- don't care
*/

                                                        /* 0123 4567    */
#define P8MOD_SCI1ClkRcv        0x03u                   /* 0000 0011B   P8 operation mode
register      */
                                                        /* |||| |||+--- SCLKI1/SCLKO1
*/
                                                        /* |||| ||+---- RXD1
*/
                                                        /* |||| |+----- P85
*/
                                                        /* |||| +------ P84
*/
                                                        /* |||+-------- P83
*/
                                                        /* ||+--------- P82
*/
                                                        /* ++---------- don't care
*/

/* Setting serial I/O */
```

```
 58
 59                                                      /* 0123 4567    */
 60  #define SnTCNT_INI              0x00u               /* 0000 0000B   SIOn transmit control
register  */
 61                                                      /* |||| |||+--- Disable transmission
*/
 62                                                      /* |||| +++---- don't care
*/
 63                                                      /* ||++-------- f(BCLK)
*/
 64                                                      /* ++---------- don't care
*/
 65
 66                                                      /* 0123 4567    */
 67  #define SnMOD_CLK_INI           0x80u               /* 1000 0000B   SIOn mode register
*/
 68                                                      /* |||| ++++--- don't care
*/
 69                                                      /* |||+-------- Select
clock(specified separately)*/
 70                                                      /* +++--------- 8-bit CSIO
*/
 71
 72                                                      /* 0123 4567    */
 73  #define SnSMOD_CKPOL_INI        0x00u               /* 0000 0000B   SIOn special mode
register      */
 74                                                      /* |||| |||+--- Clock Polarity Select
*/
 75                                                      /* |||| ||+---- don't care
*/
 76
 77  /* Setting interrupt priority level */
 78
 79  #define SioILEVEL       0x0u                        /* Serial IO transmit/receive
interrupt priority level*/
 80  #define ILEVEL_7        0x7u                        /* Interrupt Disable
*/
 81
 82  /* Setting baud rate (Be sure to check actually set value when using) */
 83
 84  #define XIN             10                          /* 10MHz */
 85  #define ClkBAUD_500     (XIN * 2000000 / 2 / 500000 - 1)        /* 500Kbps */
 86
 87  /*""FUNC COMMENT""*****************************************************
 88   * Function name: SIO0_CLKTinit()
 89   *----------------------------------------------------------------------
 90   * Description  : Sets SIO0 for synchronous serial I/O transmission
 91   *              : - Program transmission
 92   *              : - Internal clock selected (clock output)
 93   *----------------------------------------------------------------------
 94   * Argument     : -
 95   *----------------------------------------------------------------------
 96   * Returns      : -
 97   *----------------------------------------------------------------------
 98   * Notes        : Must be executed while interrupts are disabled
 99   *""FUNC COMMENT END""*************************************************/
100  void    SIO0_CLKTinit( void)
101  {
102
103  /*** Setting transfer mode ***/
104
105          S0RCNT = 0x00;                                  /* Disable reception */
106          S0TCNT = SnTCNT_INI;                            /* f(BCLK), disable
transmission */
107          P8MOD |= P8MOD_SCI0ClkTr;                       /* Set P8 for CSIO mode */
108          S0MOD = SnMOD_CLK_INI;                          /* Set data format */
109          S0BAUR = ClkBAUD_500;                           /* Set baud rate */
110          S0SMOD = SnSMOD_CKPOL_INI;                      /* Set Transmit/Receive clock
polarity */
111
112  /*** Interrupt related settings ***/
113
114          ISIO0TXCR = ILEVEL_7;                           /* Set SIO0 transmit interrupt
priority level*/
115          ISIO0RXCR = ILEVEL_7;                           /* Set SIO0 receive interrupt
priority level */
```

```
116          SIO0MASK &= ~TOMASK;                                   /* Disable SIO0 transmit
interrupt request*/
117          SIO0MASK &= ~ROMASK;                                   /* Disable SIO0 receive
interrupt request */
118
119  /*** Starting transmission/reception ***/
120
121          S0TCNT |= TEN;                                         /* Enable transmission */
122  }
123
124  /*""FUNC COMMENT""*****************************************************
125   * Function name: SIO1_CLKRinit()
126   *----------------------------------------------------------------------
127   * Description  : Enable transmissionSets SIO1 for synchronous serial I/O reception
128   *              : - Reception by interrupt
129   *              : - External clock selected (clock input)
130   *----------------------------------------------------------------------
131   * Argument     : -
132   *----------------------------------------------------------------------
133   * Returns      : -
134   *----------------------------------------------------------------------
135   * Notes        : Port input function must be enabled
136   *              : To receive, it is necessary to enable transmission and write dummy data to
transmit buffer register
137   *              : Must be executed while interrupts are disabled
138   *""FUNC COMMENT END""*************************************************/
139  void    SIO1_CLKRinit( void)
140  {
141
142  /*** Setting transfer mode */
143
144          S1RCNT = 0x00;                                         /* Disable reception */
145          S1TCNT = SnTCNT_INI;                                   /* f(BCLK), disable
transmission */
146          P8MOD |= P8MOD_SCI1ClkRcv;                             /* Set P8 for CSI1 mode */
147          S1MOD = SnMOD_CLK_INI | CKS;                           /* Set data format */
148          S1SMOD = SnSMOD_CKPOL_INI;                             /* Set Transmit/Receive clock
polarity */
149
150  /*** Interrupt related settings ***/
151
152          ISIO1TXCR = ILEVEL_7;                                  /* Set SIO1 transmit interrupt
priority level*/
153          ISIO1RXCR = SioILEVEL;                                 /* Set SIO1 receive interrupt
priority level */
152          SIO3SEL &= ~ISR1;                                      /* Select reception-finished
interrupt*/
153          SIO3MASK &= ~T1MASK;                                   /* Disable SIO1 transmit
interrupt request */
156          SI03MASK |= R1MASK;                                    /* Disable SIO1 receive
interrupt request*/
157
158  /*** Starting transmission/reception ***/
159
160          S1TCNT |= TEN;                                         /* Enable transmission */
161          S1RCNT |= REN;                                         /* Enable reception */
162  }
163
164  /*""FUNC COMMENT""*****************************************************
165   * Function name: SIO0_Tr()
166   *----------------------------------------------------------------------
167   * Description  : Transmits data from SIO0
168   *----------------------------------------------------------------------
169   * Argument     : unsigned char *TrBuf  Pointer to transmit data buffer
170   *              : unsigned int TrNum    Number of transmit bytes
171   *----------------------------------------------------------------------
172   * Returns      : -
173   *----------------------------------------------------------------------
174   * Notes        : Do not always need to wait for end of transmission
175   *""FUNC COMMENT END""*************************************************/
176  void  SIO0_Tr( UCHAR *TrBuf, ULONG TrNum)
177  {
178          ULONG   i;
179
180          for( i = 0ul; i < TrNum; i++) {
181                  while(( S0TCNT & TBE) == 0u){                  /* Wait until transmit buffer
is empty */
```

```
182                         ;
183                 }
184                 S0TXB_L = *TrBuf++;                             /* Transfer return data */
185         }
186         while(( S0TCNT & TSTAT) != 0u){                         /* Wait for end of
transmission */
187                 ;
188         }
189 }
190
191 /*""FUNC COMMENT""*****************************************************
192  * Function name: SIO1_RcvInt()
193  *-----------------------------------------------------------------------
194  * Description  : Reads out received data
195  *-----------------------------------------------------------------------
196  * Argument     : -
197  *-----------------------------------------------------------------------
198  * Returns      : -
199  *-----------------------------------------------------------------------
200  * Notes        :
201  *""FUNC COMMENT END""*************************************************/
202 void    SIO1_RcvInt(void)
203 {
204         UCHAR   data;
205         UCHAR   status;
206
207         status = S1RCNT;                                        /* Read receive status */
208         data = S1RXB_L;                                         /* Read out received data */
209         status |= S1RCNT;                                       /* Handle overrun */
210
211         if(( status & ERS) == 0u) {                             /* Check receive error sum */
212                 RcvBuf1[ rx_cnt1++] = data;                     /* Store received data */
213
214         } else {                                                /* Process receive errors */
215
216                 /* Error processing by user */
217
218                 S1RCNT &= ~REN;                                 /* Disable reception */
219                 S1RCNT |= REN;                                  /* Enable reception */
220         }
221 }
222
```

4.4.3 startup.ms (partially omitted)

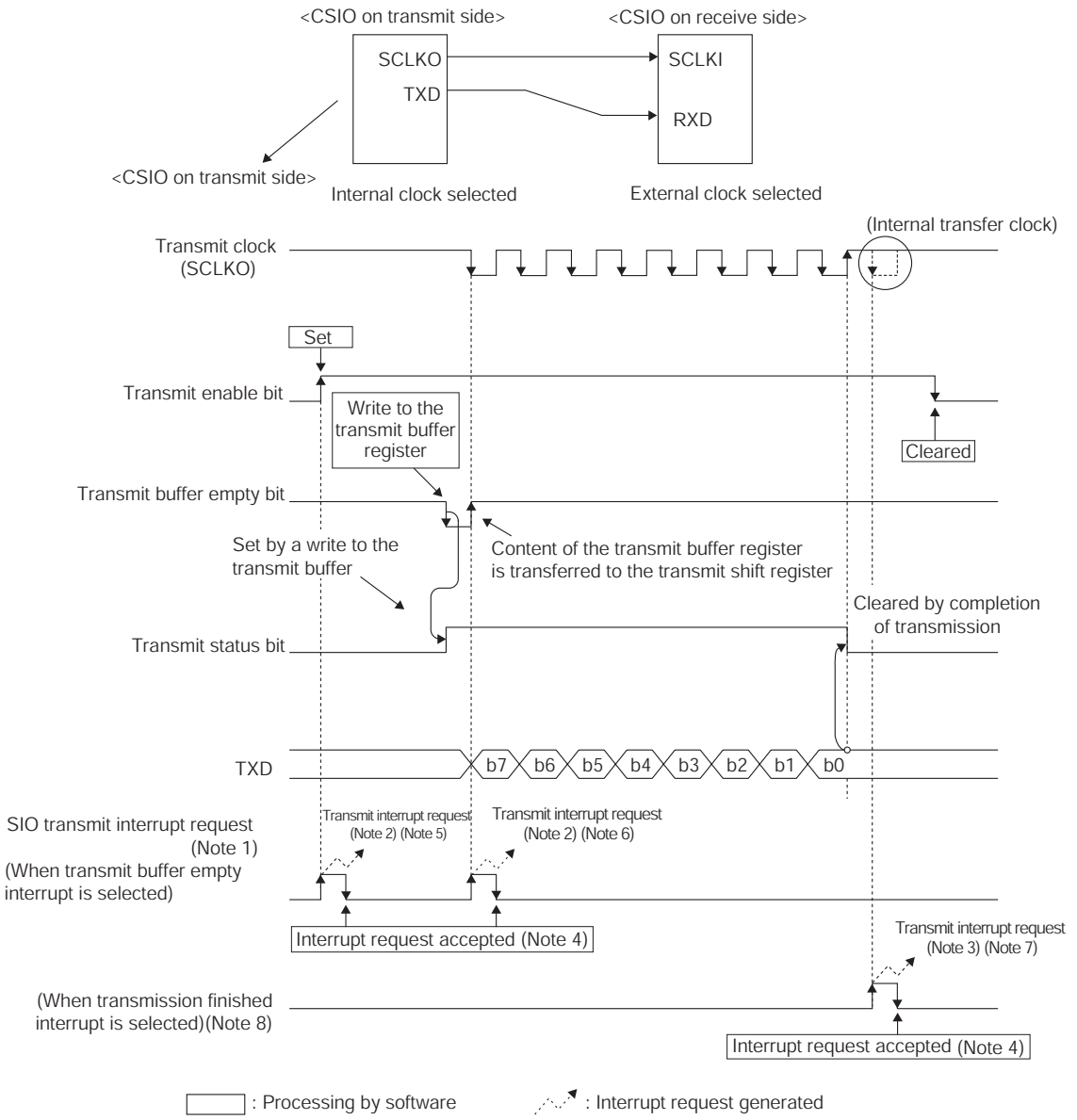
(Omitted)

```
 69  ;**************************************
 70  ; ICU Vector Table
 71  ;**************************************
 72  ;
 73          .SECTION        ICUVECT, DATA, ALIGN=4
 74  ;
 75          .IMPORT         $SIO1_RcvInt            ; SIO1 Receive Interrupt
 76  ;
 77  vectbl:
 78          .DATA.W         EIT_reset               ; H'0000 0094   MJT Input Interrupt 4:TIN3-
TIN6
 79          .DATA.W         EIT_reset               ; H'0000 0098   MJT Input Interrupt 3:TIN20-
TIN23
 80          .DATA.W         EIT_reset               ; H'0000 009C   MJT Input Interrupt 2:TIN12-
TIN19
 81          .DATA.W         EIT_reset               ; H'0000 00A0   MJT Input Interrupt 1:TIN0-
TIN2
 82          .DATA.W         EIT_reset               ; H'0000 00A4   MJT Input Interrupt 0:TIN7-
TIN11
 83          .DATA.W         EIT_reset               ; H'0000 00A8   MJT Output Interrupt
7:TMS0,TMS1
 84          .DATA.W         EIT_reset               ; H'0000 00AC   MJT Output Interrupt
6:TOP8,TOP9
 85          .DATA.W         EIT_reset               ; H'0000 00B0   MJT Output Interrupt 5:TOP10
 86          .DATA.W         EIT_reset               ; H'0000 00B4   MJT Output Interrupt 4:TIO4-
TIO7
 87          .DATA.W         EIT_reset               ; H'0000 00B8   MJT Output Interrupt
3:TIO8,TIO9
 88          .DATA.W         EIT_reset               ; H'0000 00BC   MJT Output Interrupt 2:TOP0-
TOP5
 89          .DATA.W         EIT_reset               ; H'0000 00C0   MJT Output Interrupt
1:TOP6,TOP7
 90          .DATA.W         EIT_reset               ; H'0000 00C4   MJT Output Interrupt 0:TIO0-
TIO3
 91          .DATA.W         EIT_reset               ; H'0000 00C8   DMAC0-4 Interrupt:DMA0-DMA4
 92          .DATA.W         $SIO1_RcvInt            ; H'0000 00CC   SIO1 Receive Interrupt
 93          .DATA.W         EIT_reset               ; H'0000 00D0   SIO1 Transmit Interrupt
 94          .DATA.W         EIT_reset               ; H'0000 00D4   SIO0 Receive Interrupt
 95          .DATA.W         EIT_reset               ; H'0000 00D8   SIO0 Transmit Interrupt
 96          .DATA.W         EIT_reset               ; H'0000 00DC   A-D0 Conversion Interrupt
 97          .DATA.W         EIT_reset               ; H'0000 00E0   TID0 Output Interrupt
 98          .DATA.W         EIT_reset               ; H'0000 00E4   TOD0 Output Interrupt
 99          .DATA.W         EIT_reset               ; H'0000 00E8   DMAC5-9 Interrupt:DMA5-DMA9
100          .DATA.W         EIT_reset               ; H'0000 00EC   SIO2,3 Transmit/Receive
Interrupt
101          .DATA.W         EIT_reset               ; H'0000 00F0   RTD Interrupt
102          .DATA.W         EIT_reset               ; H'0000 00F4   TID1 Output Interrupt
103          .DATA.W         EIT_reset               ; H'0000 00F8   TOD1,TOM0 Output Interrupt
104          .DATA.W         EIT_reset               ; H'0000 00FC   SIO4,5 Transmit/Receive
Interrupt
105          .DATA.W         EIT_reset               ; H'0000 0100   A-D1 Conversion Interrupt
106          .DATA.W         EIT_reset               ; H'0000 0104   TID2 Output Interrupt
107          .DATA.W         EIT_reset               ; H'0000 0108   TML1 Input Interrupt
108          .DATA.W         EIT_reset               ; H'0000 010C   CAN0 Transmit/Receive & Error
Interrupt
109          .DATA.W         EIT_reset               ; H'0000 0110   CAN1 Transmit/Receive & Error
Interrupt
```

(The remainder of the program has been omitted)

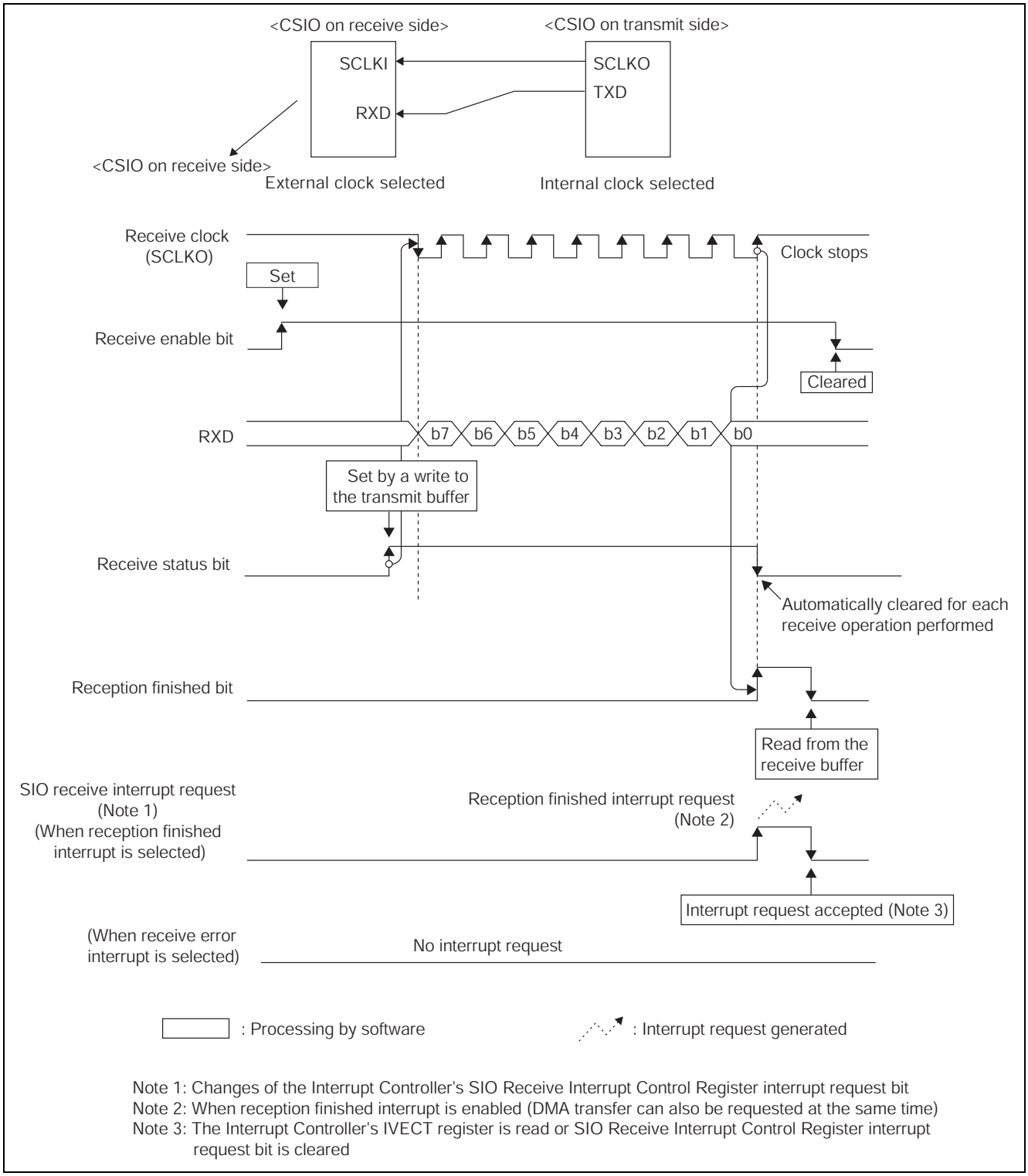
4.5 Operation Timing of the Serial Interface

The CSIO mode transmission and reception timing diagrams are shown below.



Note 1: Changes of the Interrupt Controller's SIO Transmit Interrupt Control Register interrupt request bit
Note 2: When transmit buffer empty interrupt is enabled (DMA transfer can also be requested at the same time)
Note 3: When transmission finished interrupt is enabled
Note 4: The Interrupt Controller's IVECT register is read or SIO Transmit Interrupt Control Register interrupt request bit is cleared.
Note 5: A transmit interrupt request is generated when transmission is enabled.
Note 6: Be aware that even after transmit data is written to the transmit buffer, a transmit interrupt request is generated when the data is transferred from the transmit buffer to the transmit shift register and the transmit buffer is thereby emptied.
Note 7: A transmission finished interrupt request is generated by a falling edge of the internal transfer clock pulse at which transmission of the transmit shift register data has finished.
Note 8: It is inhibited to select the transmission finished interrupt when an external clock is selected.

Figure 4.5.1 CSIO Mode Transmission Timing Diagram



Note 1: Changes of the Interrupt Controller's SIO Receive Interrupt Control Register interrupt request bit
Note 2: When reception finished interrupt is enabled (DMA transfer can also be requested at the same time)
Note 3: The Interrupt Controller's IVECT register is read or SIO Receive Interrupt Control Register interrupt request bit is cleared

Figure 4.5.2 CSIO Mode Receive Timing Diagram

5. Reference of documents

- 32176 Group User's Manual (Rev.1.01)
- M32R Family Software Manual (Rev.1.20)
- M3T-CC32R V.4.30 User's Manual (Compiler)
- M3T-CC32R V.4.30 User's Manual (Assembler)

(Please get the latest one from Renesas Technology Corp. website.)

6. Website and Support Center

- Renesas Technology Corp. website.

  http://www.renesas.com/

- Customer Support Center for all Products and Technical Support Center for M32R Family

  Customer Support Center: csc@renesas.com

Revision Record

Rev.	Date	Description	
		Page	Summary
1.00	Dec.09.05	—	First edition issued

Keep safety first in your circuit designs!

1. Renesas Technology Corporation 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 Corporation 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 Corporation or a third party.
2. Renesas Technology Corporation 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 Corporation without notice due to product improvements or other reasons. It is therefore recommended that customers contact Renesas Technology Corporation or an authorized Renesas Technology Corporation 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 Corporation 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 Corporation by various means, including the Renesas Technology Corporation 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 Corporation assumes no responsibility for any damage, liability or other loss resulting from the information contained herein.
5. Renesas Technology Corporation 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 Corporation or an authorized Renesas Technology Corporation 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 Corporation 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 Corporation for further details on these materials or the products contained therein.