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.

RENESAS

32176 Group

Application Using the Serial Interface (UART Mode)

1. Overview

This documentation describes a sample program 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 UART mode, the transfer rate is decided by clock divider frequency division value and the values set in the individual baud rate registers (BRG) 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)	Clock Divider Count Source = 20 MHz			
	Clock Divider Frequency Division Value (Frequency Division)	**BRG Setting Value**	**Error (%)**	**Actual Baud Rate (bps)**
300	32	129	0.16	300.48
600	32	64	0.16	600.96
1200	8	129	0.16	1201.92
2400	8	64	0.16	2403.85
4800	8	32	–1.36	4734.85
9600	1	129	0.16	9615.38
12500	1	99	0.00	12500.00
14400	1	86	–0.22	14367.82
19200	1	64	0.16	19230.77
28800	1	42	0.94	29069.77
31250	1	39	0.00	31250.00
38400	1	32	–1.36	37878.79
57600	1	21	–1.36	56818.18
62500	1	19	0.00	62500.00
115200	1	10	–1.36	113636.36
125000	1	9	0.00	125000.00
250000	1	4	0.00	250000.00
500000	1	2	–16.67	416666.67
625000	1	1	0.00	625000.00
1000000	—	—	—	—
1250000	1	0	0.00	1250000.00

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

$$\text{BRG setting value} = \frac{f\text{ (BCLK)}}{\text{Baud rate (bps)} \times \text{clock divider frequency division value} \times 16} - 1$$

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 the sample program, data is output from the SIO0 and data is received on the SIO1. Both communications use 8-bit UART. 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, full duplex communication is enabled by the on-chip UART functions of the 32176.

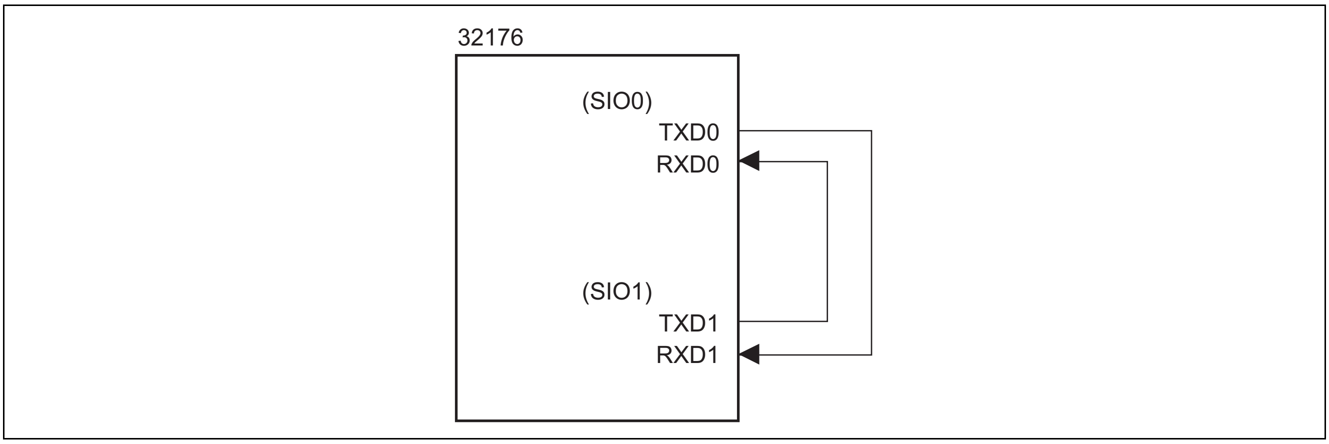


Figure 4.1.1 Connection Diagram

4.2 Transmit/Receive Processing

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

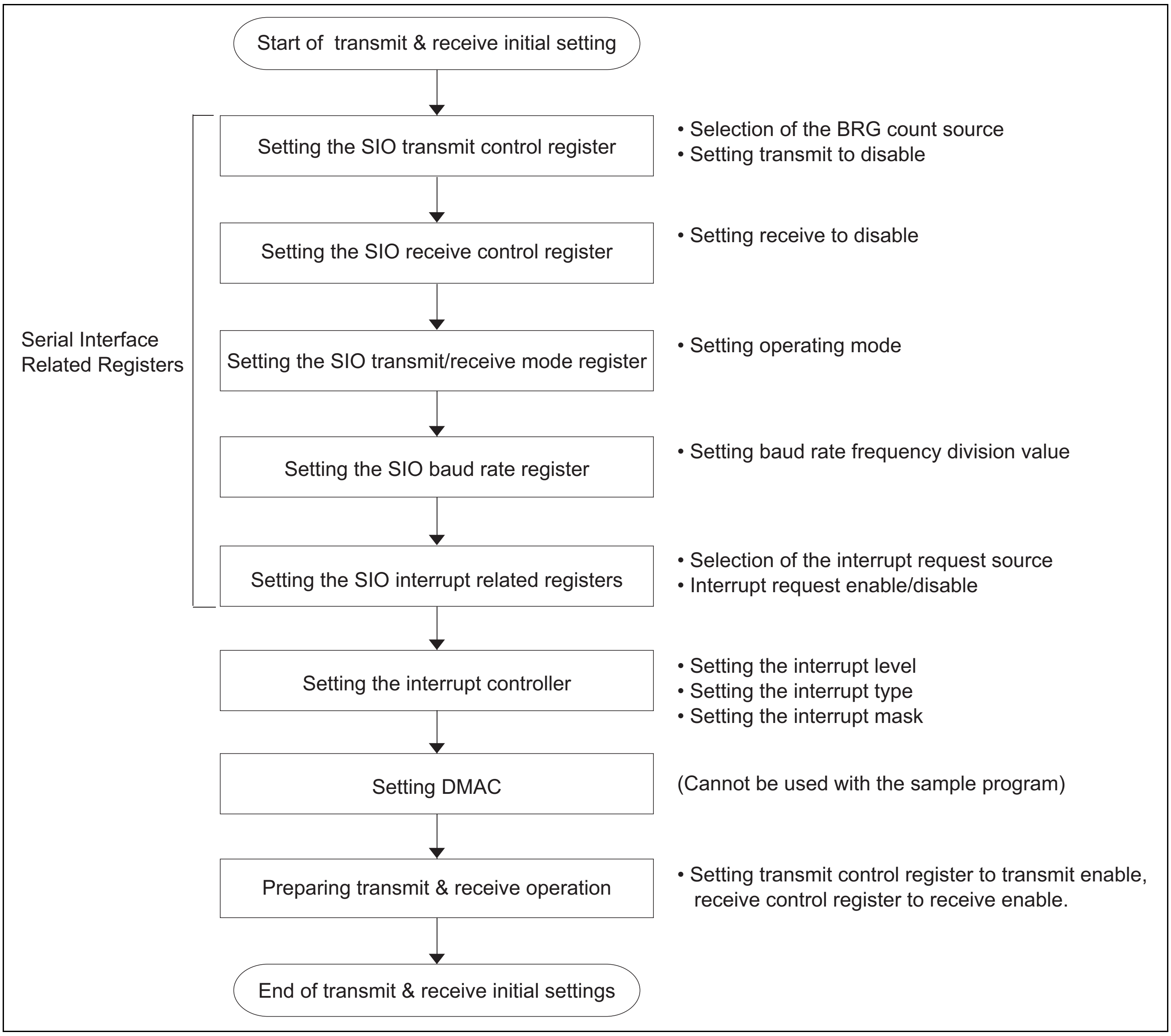


Figure 4.2.1 Flowchart for the Initial Settings of Transmission and Reception in the UART Mode

4.3 Description of the Sample Program

Note: 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 input/output port. (P11MOD)

Note: Setting a direction register to output before setting the corresponding data register, causes undefined values to be output until the data register is written to.

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 initialization.
(5) Call the function for SIO1 receive processing initialization.
(6) Call the function for enabling interrupts.
(7) Transmit/receive processing infinite loop.

- Transmit one byte of variable send_data from SIO0.
- Wait for the SIO1 receive interrupt.
- Retrieve received data from SIO1 receive data buffer.
- Re-set to 0 when the SIO1 receive data buffer pointer overflow occurs.
- Store the current receive data buffer pointer.
- When SIO0 transmission data and SIO1 received data have the same value:
  1) Displays the received data on LED.
  2) Increments the transmission data and start next transmission processing.
- When 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 SIO0 Initialization Function (SIO0_init ())

(1) Setting the transfer mode.

- Set the receive enable bit of the SIO0 receive control register to receive disabled. (S0RCNT: REN)
- Set the SIO0 transmit control register. (S0TCNT: CDIV, TEN)
  Set the BRG count source is set to f (BCLK), transmission disabled.
- Set the P8 operation mode register. (P8MOD: P82MOD, P83MOD)
  Set port P82 to TXD0, and port P83 to RXD0.
- Set the SIO0 transmit/receive mode register. (S0MOD: SMOD, CKS, STB, PEN, SEN)
  Set the 8-bit UART using the internal clock, with one stop bit, parity disabled, sleep function invalid.
- Set the baud rate frequency division value to SIO0 baud rate register. (S0BAUR)

(2) Interrupt settings.

- Set the interrupt level in the SIO0 transmission interrupt control register to interrupt disabled. (ISIO0TXCR: ILEVEL)
- Set the interrupt level in the SIO0 receive interrupt control register to “0” (highest priority). (ISIO0RXCR: ILEVEL)
- Set the SIO0 receive interrupt request source select bit of the SIO03 interrupt request source select register to reception finished interrupt. (SI03SEL: ISR0)
- Set the SIO03 interrupt request mask register. (SI03MASK: T0MASK, R0MASK)
  Disable SIO0 transmit interrupts, and enable SIO0 receive interrupts.

(3) Setting to enable transmission and reception.

- Set the transmit enable bit of the SIO0 transmit control register to transmit enabled. (S0TCNT: TEN)
- Set the receive enable bit of the SIO0 receive control register to receive enabled. (S0RCNT: REN)

4.3.5 SIO1 Initialization Function (SIO1_init ())

(1) Setting the transfer mode.

- Set the receive enable bit of the SIO1 receive control register to receive disabled. (S1RCNT: REN)
- Set the SIO1 transmit control register. (S0TCNT: CDIV, TEN)
  Set the BRG count source to f (BCLK), transmit disabled.
- Set the P8 operation mode register. (P8MOD: P85MOD, P86MOD)
  Set the port P85 to TXD1, and port P86 to RXD1.
- Set the SIO1 transmit/receive mode register. (S1MOD: SMOD, CKS, STB, PEN, SEN)
  Set the 8-bit UART using the internal clock, with one stop bit, parity disabled, sleep function invalid.
- Set the baud rate frequency division value in the SIO1 baud rate register. (S1BAUR)

(2) Interrupt settings.

- Set the interrupt level in the SIO1 transmission 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 reception 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 SIO2 Initialization Function (SIO2_init ())

Set in the same way as SIO0_init.

Note: This function is not used in this sample program.

4.3.7 SIO3 Initialization Function (SIO3_init ())

Set in the same way as SIO0_init.

Note: This function is not used in this sample program.

4.3.8 UART Transmit Processing Function (SIO0_Tr (), SIO1_Tr (), SIO2_Tr (), SIO3_Tr ())

(1) Transmit processing.

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

Note: SIO1_Tr (), SIO2_Tr (), and SIO3_Tr () are not used in this sample program.

4.3.9 The UART Receive Processing Function by SIO Receive Interrupt (SIO0_RcvInt (), SIO1_RcvInt (), SIO2_RcvInt (), SIO3_RcvInt ())

(1)Receive processing.

- Read the SIO receive control register (receive status). (S1RCNT)
- Read the SIO receive buffer register. (S1RXB_L)
- Re-read the SIO receive control register.
  For detecting overrun occurring between reading the receive status and reading the receive buffer register.
- Check SIO 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 and then set to receive enabled. (Each bit in the receive control registers is cleared.)

Note: SIO0_RcvInt (), SIO2_RcvInt (), SIO3_RcvInt () are not used in this sample program.

4.3.10 Interrupt Handler Function of SIO2, 3 (IntHand_SIO23 ())

Since the SIO2 and SIO3 interrupts are a group interrupt, which interrupt request occurred is determined by the interrupt request status bit of the SIO23 interrupt request status register.
(SI23STAT: IRQR2, IRQR3)

Note: This function is not used in this sample program.

4.3.11 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 UART 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 uart_main.c

```
/*""FILE COMMENT""***********************************************************
 *      M32R C Programming                Rev. 1.01
 *               < Sample Program for 32176 >
 *               < Serial I/O (UART) (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 */

/*************************************************************************/
/*               Definition of external reference                       */
/*************************************************************************/

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

extern  void            SIO0_init( void);                   /* Initialize SIO0 */
extern  void            SIO1_init( void);                   /* Initialize SIO1 */
extern  void            SIO0_Tr( UCHAR *TrBuf, ULONG TrNum);   /* Transmit from SIO0 */
extern  void            SIO1_Tr( UCHAR *TrBuf, ULONG TrNum);   /* Transmit from SIO1 */

/*               Global variable                                        */
/*               Global variable                                        */
/*************************************************************************/

        volatile UCHAR  RcvBuf0[10];                        /* Receive buffer */
        volatile UCHAR  RcvBuf1[10];                        /* Receive buffer */
        volatile UCHAR  RcvBuf2[10];                        /* Receive buffer */
        volatile UCHAR  RcvBuf3[10];                        /* Receive buffer */
        volatile UCHAR  rx_cnt0;                            /* Pointer to receive buffer */
        volatile UCHAR  rx_cnt1;                            /* Pointer to receive buffer */
        volatile UCHAR  rx_cnt2;                            /* Pointer to receive buffer */
        volatile UCHAR  rx_cnt3;                            /* 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()
 *-----------------------------------------------------------------------
 * Description  : Port initialization
 *-----------------------------------------------------------------------
 * Argument     : -
 *-----------------------------------------------------------------------
 * Returns      : -
 *-----------------------------------------------------------------------
 * Notes        : -
 *""FUNC COMMENT END""*****************************************************/
void port_init(void)
{
        PICNT = PIEN0;                                  /* Enable port input */

/*** LED output port ***/

        P11DATA = 0x00;                                 /* Output data(must be set prior to mode)*/
        P11DIR = 0xff;                                  /* P110-P117 : Output mode */
        P11MOD = 0x00;                                  /* P110-P117 : Input/output port */
}

/*""FUNC COMMENT""*******************************************************
 * Function name: main()
 *-----------------------------------------------------------------------
 * Description  : Receives data (increment pattern) from SIO1 after being transmitted in 500 Kbps clock-synchronized
 *              : serial mode (when the source clock frequency = 10 MHz) from SIO0 and
 *              : if the received data is the same as transmitted, displays it on LED(port 11).
 *              : If unable to receive the same data, it displays 0x55 in blinking inverse video.
 *-----------------------------------------------------------------------
 * Argument     : -
 *-----------------------------------------------------------------------
 * Returns      : -
 *-----------------------------------------------------------------------
 * Notes        : -
 *""FUNC COMMENT END""*****************************************************/
void    main(void)
{
/*              +-------+
 *              |       |
 *      (SIO0)  |   TXD0| --+
 *              |       |   |
 *              |       |   |
 *      (SIO1)  |   RXD1| <-+
 *              |       |
 *      (SIO1)  |   TXD1| --+
 *              |       |   |
 *              |       |   | (not use)
 *      (SIO0)  |   RXD0| <-+
 *              |       |
 *              +-------+
        SI03MASK &= ~T0MASK;                            /*Disable SIO0 transmit interrupt request*/

        ULONG   j;
        UCHAR   send_data;
        UCHAR   rcv_data;
        UCHAR   rx_cnt1_bak;

        rx_cnt1 = 0;
        rx_cnt1_bak = 0;

        send_data = 0;

        DisInt();                                       /* Disable interrupt */
```

```
        init_func();                                    /* Initialize microcomputer */

        SIO0_init();                                    /* Initialize SIO0 */
        SIO1_init();                                    /* Initialize SIO1 */

        EnInt();                                        /* Enable interrupt */

        while(1) {
                SIO0_Tr( &send_data, 1ul);              /* Send data from SIO0 */

                while( rx_cnt1 == rx_cnt1_bak){         /* Wait for data to receive */
                        ;
                }
                rcv_data = RcvBuf1[ rx_cnt1 - 1];       /* Read out received data */

                if( rx_cnt1 == 10) {                    /* Check for receive counter overflow */
                        rx_cnt1 = 0;
                }
                rx_cnt1_bak = rx_cnt1;

                if( rcv_data == send_data) {
                        P11DATA = rcv_data;             /* Display received (transmitted) data */
                        send_data++;                    /* Increment transmit data */
                } else {                                /* Received data in error */
                        P11DATA = 0x55;
                        while(1) {
                                for( j = 0ul; j < 1000000ul; j++){      /* Wait */
                                        ;
                                }
                                P11DATA ^= 0xffu;       /* Display 0x55 in blinking inverse video*/
                        }
                }
        }
}
```

4.4.2 uart.c

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

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

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

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

        void            SIO0_init( void);                   /* Initialize SIO0 */
        void            SIO1_init( void);                   /* Initialize SIO1 */
        void            SIO2_init( void);                   /* Initialize SIO2 */
        void            SIO3_init( void);                   /* Initialize SIO3 */
        void            SIO0_Tr( UCHAR *TrBuf, ULONG TrNum);    /* Transmit from SIO0 */
        void            SIO1_Tr( UCHAR *TrBuf, ULONG TrNum);    /* Transmit from SIO1 */
        void            SIO2_Tr( UCHAR *TrBuf, ULONG TrNum);    /* Transmit from SIO2 */
        void            SIO3_Tr( UCHAR *TrBuf, ULONG TrNum);    /* Transmit from SIO3 */
        void            SIO0_RcvInt(void);                  /* SIO0 receive interrupt */
        void            SIO1_RcvInt(void);                  /* SIO1 receive interrupt */
        void            SIO2_RcvInt(void);                  /* SIO2 receive interrupt */
        void            SIO3_RcvInt(void);                  /* SIO3 receive interrupt */
        void            IntHand_SIO23(void);                /* SIO2,3 transmit/receive interrupt handler */

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

extern  volatile UCHAR  RcvBuf0[];                          /* Receive buffer */
extern  volatile UCHAR  RcvBuf1[];                          /* Receive buffer */
extern  volatile UCHAR  RcvBuf2[];                          /* Receive buffer */
extern  volatile UCHAR  RcvBuf3[];                          /* Receive buffer */
extern  volatile UCHAR  rx_cnt0;                            /* Pointer to receive buffer */
extern  volatile UCHAR  rx_cnt1;                            /* Pointer to receive buffer */
extern  volatile UCHAR  rx_cnt2;                            /* Pointer to receive buffer */
extern  volatile UCHAR  rx_cnt3;                            /* Pointer to receive buffer */

/**********************************************************************/
/*              Define macro                                          */
/**********************************************************************/

/* Setting port operation mode */
                                                    /* 0123 4567    */
#define P8MOD_SCI0      0x30u                       /* 0011 0000B   P8 operation mode register      */
                                                    /* |||| |||+--- P87 */
                                                    /* |||| ||+---- P86 */
                                                    /* |||| |+----- P85 */
                                                    /* |||| +------ P84 */
                                                    /* |||+-------- RXD0 */
                                                    /* ||+--------- TXD0 */
                                                    /* ++---------- don't care */

```

```
                                                            /* 0123 4567    */
#define P8MOD_SCI1      0x06u                               /* 0000 0110B   P8 operation mode register      */
                                                            /* |||| |||+--- P87         */
                                                            /* |||| ||+---- RXD1        */
                                                            /* |||| |+----- TXD1        */
                                                            /* |||| +------ P84         */
                                                            /* |||+-------- P83         */
                                                            /* ||+--------- P82         */
                                                            /* ++---------- don't care  */

                                                            /* 0123 4567    */
#define P17MOD_SCI2     0x0cu                               /* 0000 1100B   P17 operation mode register     */
                                                            /* |||| |||+--- P177        */
                                                            /* |||| ||+---- P176        */
                                                            /* |||| |+----- RXD2        */
                                                            /* |||| +------ TXD2        */
                                                            /* |||+-------- P173        */
                                                            /* ||+--------- P172        */
                                                            /* ++---------- don't care  */

                                                            /* 0123 4567    */
#define P17MOD_SCI3     0x03u                               /* 0000 0011B   P17 operation mode register     */
                                                            /* |||| |||+--- RXD3        */
                                                            /* |||| ||+---- TXD3        */
                                                            /* |||| |+----- P175        */
                                                            /* |||| +------ P174        */
                                                            /* |||+-------- P173        */
                                                            /* ||+--------- P172        */
                                                            /* ++---------- don't care  */

/* Setting serial IO */
                                                            /* 0123 4567    */
#define SnTCNT_INI      0x00u                               /* 0000 0000B   SIOn transmit control register  */
                                                            /* |||| |||+--- Disable transmission */
                                                            /* |||| +++---- don't care  */
                                                            /* ||++-------- f(BCLK)     */
                                                            /* ++---------- don't care  */

                                                            /* 0123 4567    */
#define SnMOD_INI       0x20u                               /* 0010 0000B   SIOn mode register */
                                                            /* |||| |||+--- Sleep function disabled */
                                                            /* |||| ||+---- Parity inhibited */
                                                            /* |||| |+----- don't care(odd) */
```

```
                                                      /* |||| +------ 1 stop bit */
                                                      /* |||+-------- Internal clock */
                                                      /* +++--------- 8-bit UART */

/* Setting interrupt priority level */

#define SioILEVEL        0x0u                         /* Serial IO transmit/receive interrupt priority level*/
#define ILEVEL_7         0x7u                         /* Interrupt Disable */

/* Setting baud rate (Be sure to check actually set value when using) */

#define XIN              10                           /* 10MHz */

#define BAUD_62_5        (XIN * 2000000 / 16 / 62500 - 1)       /* 62.5Kbps */
#define BAUD_15_6        (XIN * 2000000 / 16 / 15600 - 1)       /* 15.6Kbps */
#define BAUD_9615        (XIN * 2000000 / 16 / 9615 - 1)        /* 9615bps */


/*""FUNC COMMENT""*******************************************************
 * Function name: SIO0_init()
 *----------------------------------------------------------------------
 * Description   : Sets SIO0 for 8-bit UART
 *               : - Reception by interrupt
 *               : - Program transmission
 *----------------------------------------------------------------------
 * Argument      : -
 *----------------------------------------------------------------------
 * Returns       : -
 *----------------------------------------------------------------------
 * Notes         : Port input function must be enabled
 *               : Must be executed while interrupts are disabled
 *""FUNC COMMENT END""**************************************************/
void    SIO0_init( void)
{

/*** Setting transfer mode */

        S0RCNT = 0x00;                                  /* Disable reception */
        S0TCNT = SnTCNT_INI;                            /* f(BCLK) and disable transmission */
        P8MOD |= P8MOD_SCI0;                            /* Set P8 for SIO mode */
        S0MOD = SnMOD_INI;                              /* Set data format */
        S0BAUR = BAUD_62_5;                             /* Set baud rate */

/*** Interrupt related settings ***/

        ISIO0TXCR = ILEVEL_7;                           /* Set SIO0 transmit interrupt priority level*/
        ISIO0RXCR = SioILEVEL;                          /* Set SIO0 receive interrupt priority level */
        SI03SEL &= ~ISR0;                               /* Select receive-finished interrupt */
        SI03MASK &= ~T0MASK;                            /* Disable SIO0 transmit interrupt request */
        SI03MASK |= R0MASK;                             /* Enable SIO0 receive interrupt request */

/*** Starting transmission/reception ***/

        S0TCNT |= TEN;                                  /* Enable transmission */
        S0RCNT |= REN;                                  /* Enable reception */
}

/*""FUNC COMMENT""*******************************************************
 * Function name: SIO1_init()
 *----------------------------------------------------------------------
 * Description   : Sets SIO1 for 8-bit UART
 *               : - Reception by interrupt
 *               : - Program transmission
 *----------------------------------------------------------------------
 * Argument      : -
 *----------------------------------------------------------------------
```

```
 * Returns      : -
 *-----------------------------------------------------------------------
 * Notes        : Port input function must be enabled
 *              : Must be executed while interrupts are disabled
 *""FUNC COMMENT END""*****************************************************/
void    SIO1_init( void)
{

/*** Setting transfer mode */

        S1RCNT = 0x00;                                  /* Disable reception */
        S1TCNT = SnTCNT_INI;                            /* f(BCLK) and disable transmission */
        P8MOD |= P8MOD_SCI1;                            /* Set P8 for SIO mode */
        S1MOD = SnMOD_INI;                              /* Set data format */
        S1BAUR = BAUD_62_5;                             /* Set baud rate */

/*** Interrupt related settings ***/

        ISIO1TXCR = ILEVEL_7;                           /* Set SIO1 transmit interrupt priority level*/
        ISIO1RXCR = SioILEVEL;                          /* Set SIO1 receive interrupt priority level */
        SI03SEL &= ~ISR1;                               /* Select receive-finished interrupt */
        SI03MASK &= ~T1MASK;                            /* Disable SIO1 transmit interrupt request */
        SI03MASK |= R1MASK;                             /* Enable SIO1 receive interrupt request */

/*** Starting transmission/reception ***/

        S1TCNT |= TEN;                                  /* Enable transmission */
        S1RCNT |= REN;                                  /* Enable reception */
}

/*""FUNC COMMENT""*******************************************************
 * Function name: SIO2_init()
 *-----------------------------------------------------------------------
 * Description  : Sets SIO2 for 8-bit UART
 *              : - Reception by interrupt
 *              : - Program transmission
 *-----------------------------------------------------------------------
 * Argument     : -
 *-----------------------------------------------------------------------
 * Returns      : -
 *-----------------------------------------------------------------------
 * Notes        : Port input function must be enabled
 *              : Must be executed while interrupts are disabled
 *              : Interrupt priority levels are common to both SIO2,3 transmission and reception
 *""FUNC COMMENT END""*****************************************************/
void    SIO2_init( void)
{

/*** Setting transfer mode */

        S2RCNT = 0x00;                                  /* Disable reception */
        S2TCNT = SnTCNT_INI;                            /* f(BCLK) and disable transmission */
        P17MOD |= P17MOD_SCI2;                          /* Set P17 for SIO mode */
        S2MOD = SnMOD_INI;                              /* Set data format */
        S2BAUR = BAUD_62_5;                             /* Set baud rate */

/*** Interrupt related settings ***/

        ISIO23CR = SioILEVEL;                           /* Set SIO2,3 transmission and reception interrupt priority level */
        SI03SEL &= ~ISR2;                               /* Select receive-finished interrupt */
        SI03MASK &= ~T2MASK;                            /* Disable SIO2 transmit interrupt request */
        SI03MASK |= R2MASK;                             /* Enable SIO2 receive interrupt request */
        SI23STAT = (~( IRQT2 | IRQR2)) & 0x0fu;        /* Clear interrupt request */

/*** Starting transmission/reception ***/
```

```c

        S2TCNT |= TEN;                                      /* Enable transmission */
        S2RCNT |= REN;                                      /* Enable reception */
}

/*""FUNC COMMENT""*******************************************************
 * Function name: SIO3_init()
 *-----------------------------------------------------------------------
 * Description  : Sets SIO3 for 8-bit UART
 *              : - Reception by interrupt
 *              : - Program transmission
 *-----------------------------------------------------------------------
 * Argument     : -
 *-----------------------------------------------------------------------
 * Returns      : -
 *-----------------------------------------------------------------------
 * Notes        : Port input function must be enabled
 *              : Must be executed while interrupts are disabled
 *              : Interrupt priority levels are common to both SIO2,3 transmission and reception
 *""FUNC COMMENT END""***************************************************/
void    SIO3_init( void)
{

/*** Setting transfer mode */

        S3RCNT = 0x00;                                      /* Disable reception */
        S3TCNT = SnTCNT_INI;                                /* f(BCLK) and disable transmission */
        P17MOD |= P17MOD_SCI3;                              /* Set P17 for SIO mode */
        S3MOD = SnMOD_INI;                                  /* Set data format */
        S3BAUR = BAUD_62_5;                                 /* Set baud rate */

/*** Interrupt related settings ***/

        ISIO23CR = SioILEVEL;                               /* Set SIO2,3 transmission and reception interrupt priority level */
        SI03SEL &= ~ISR3;                                   /* Select receive-finished interrupt */
        SI03MASK &= ~T3MASK;                                /* Disable SIO3 transmit interrupt request */
        SI03MASK |= R3MASK;                                 /* Enable SIO3 receive interrupt request */
        SI23STAT = (~( IRQT3 | IRQR3)) & 0x0fu;             /* Clear interrupt request */

/*** Starting transmission/reception ***/

        S3TCNT |= TEN;                                      /* Enable transmission */
        S3RCNT |= REN;                                      /* Enable reception */
}

/*""FUNC COMMENT""*******************************************************
 * Function name: SIO0_Tr()
 *-----------------------------------------------------------------------
 * Description  : Transmits data from SIO0
 *-----------------------------------------------------------------------
 * Argument     : unsigned char *TrBuf  Pointer to transmit data buffer
 *              : unsigned int TrNum    Number of transmit bytes
 *-----------------------------------------------------------------------
 * Returns      : -
 *-----------------------------------------------------------------------
 * Notes        : Do not always need to wait for end of transmission
 *""FUNC COMMENT END""***************************************************/
void  SIO0_Tr( UCHAR *TrBuf, ULONG TrNum)
{
        ULONG   i;

        for( i = 0ul; i < TrNum; i++) {
                while(( S0TCNT & TBE) == 0u){               /* Wait until transmit buffer is empty */
                        ;
                }
                S0TXB_L = *TrBuf++;                         /* Transfer return data */
        }
        while(( S0TCNT & TSTAT) != 0u){                     /* Wait for end of transmission */
                ;
```

```
        }
}

/*""FUNC COMMENT""*******************************************************
 * Function name: SIO1_Tr()
 *-----------------------------------------------------------------------
 * Description  : Transmits data from SIO1
 *-----------------------------------------------------------------------
 * Argument     : unsigned char *TrBuf  Pointer to transmit data buffer
 *              : unsigned int TrNum    Number of transmit bytes
 *-----------------------------------------------------------------------
 * Returns      : -
 *-----------------------------------------------------------------------
 * Notes        : Do not always need to wait for end of transmission
 *""FUNC COMMENT END""*************************************************/
void  SIO1_Tr( UCHAR *TrBuf, ULONG TrNum)
{
        ULONG   i;

        for( i = 0ul; i < TrNum; i++) {
                while(( S1TCNT & TBE) == 0u){                   /* Wait until transmit buffer is empty */
                        ;
                }
                S1TXB_L = *TrBuf++;                             /* Transfer return data */
        }
        while(( S1TCNT & TSTAT) != 0u){                         /* Wait for end of transmission */
                ;
        }
}

/*""FUNC COMMENT""*******************************************************
 * Function name: SIO2_Tr()
 *-----------------------------------------------------------------------
 * Description  : Transmits data from SIO2
 *-----------------------------------------------------------------------
 * Argument     : unsigned char *TrBuf  Pointer to transmit data buffer
 *              : unsigned int TrNum    Number of transmit bytes
 *-----------------------------------------------------------------------
 * Returns      : -
 *-----------------------------------------------------------------------
 * Notes        : Do not always need to wait for end of transmission
 *""FUNC COMMENT END""*************************************************/
void  SIO2_Tr( UCHAR *TrBuf, ULONG TrNum)
{
        ULONG   i;

        for( i = 0ul; i < TrNum; i++) {
                while(( S2TCNT & TBE) == 0u){                   /* Wait until transmit buffer is empty */
                        ;
                }
                S2TXB_L = *TrBuf++;                             /* Transfer return data */
        }
        while(( S2TCNT & TSTAT) != 0u){                         /* Wait for end of transmission */
                ;
        }
}

/*""FUNC COMMENT""*******************************************************
 * Function name: SIO3_Tr()
 *-----------------------------------------------------------------------
 * Description  : Transmits data from SIO3
 *-----------------------------------------------------------------------
 * Argument     : unsigned char *TrBuf  Pointer to transmit data buffer
 *              : unsigned int TrNum    Number of transmit bytes
 *-----------------------------------------------------------------------
 * Returns      : -
 *-----------------------------------------------------------------------
 * Notes        : Do not always need to wait for end of transmission
 *""FUNC COMMENT END""*************************************************/
void  SIO3_Tr( UCHAR *TrBuf, ULONG TrNum)
{
        ULONG   i;

```

```
        for( i = 0ul; i < TrNum; i++) {
                while(( S3TCNT & TBE) == 0u){                   /* Wait until transmit buffer
is empty */
                        ;
                }
                S3TXB_L = *TrBuf++;                             /* Transfer return data */
        }
        while(( S3TCNT & TSTAT) != 0u){                         /* Wait for end of
transmission */
                ;
        }
}

/*""FUNC COMMENT""*******************************************************
 * Function name: SIO0_RcvInt()
 *-----------------------------------------------------------------------
 * Description  : Reads out received data
 *-----------------------------------------------------------------------
 * Argument     : -
 *-----------------------------------------------------------------------
 * Returns      : -
 *-----------------------------------------------------------------------
 * Notes        : -
 *""FUNC COMMENT END""***************************************************/
void    SIO0_RcvInt(void)
{
        UCHAR   data;
        UCHAR   status;

        status = S0RCNT;                                        /* Read receive status */
        data = S0RXB_L;                                         /* Read out received data */
        status |= S0RCNT;                                       /* Handle overrun */

        if(( status & ERS) == 0u) {                             /* Check receive error sum */
                RcvBuf0[ rx_cnt0++] = data;                     /* Store received data */

        } else {                                                /* Disable reception */

                /* Error processing by user */

                S0RCNT &= ~REN;                                 /* Disable reception */
                S0RCNT |= REN;                                  /* Enable reception */
        }
}

/*""FUNC COMMENT""*******************************************************
 * Function name: SIO1_RcvInt()
 *-----------------------------------------------------------------------
 * Description  : Reads out received data
 *-----------------------------------------------------------------------
 * Argument     : -
 *-----------------------------------------------------------------------
 * Returns      : -
 *-----------------------------------------------------------------------
 * Notes        :
 *""FUNC COMMENT END""***************************************************/
void    SIO1_RcvInt(void)
{
        UCHAR   data;
        UCHAR   status;

        status = S1RCNT;                                        /* Read receive status */
        data = S1RXB_L;                                         /* Read out received data */
        status |= S1RCNT;                                       /* Handle overrun */

        if(( status & ERS) == 0u) {                             /* Check receive error sum */
                RcvBuf1[ rx_cnt1++] = data;                     /* Store received data */

        } else {                                                /* Disable reception */

                /* Error processing by user */

                S1RCNT &= ~REN;                                 /* Disable reception */
                S1RCNT |= REN;                                  /* Enable reception */
        }
}

```

```
/*""FUNC COMMENT""***********************************************************
 * Function name: SIO2_RcvInt()
 *-----------------------------------------------------------------------
 * Description   : Reads out received data
 *-----------------------------------------------------------------------
 * Argument      : -
 *-----------------------------------------------------------------------
 * Returns       : -
 *-----------------------------------------------------------------------
 * Notes         : -
 *""FUNC COMMENT END""*****************************************************/
void    SIO2_RcvInt(void)
{
        UCHAR   data;
        UCHAR   status;

        status = S2RCNT;                                        /* Read receive status */
        data = S2RXB_L;                                         /* Read out received data */
        status |= S2RCNT;                                       /* Handle overrun */

        if(( status & ERS) == 0u) {                             /* Check receive error sum */
                RcvBuf2[ rx_cnt2++] = data;                     /* Store received data */

        } else {                                                /* Disable reception */

                /* Error processing by user */

                S2RCNT &= ~REN;                                 /* Disable reception */
                S2RCNT |= REN;                                  /* Enable reception */
        }
}

/*""FUNC COMMENT""***********************************************************
 * Function name: SIO3_RcvInt()
 *-----------------------------------------------------------------------
 * Description   : Reads out received data
 *-----------------------------------------------------------------------
 * Argument      : -
 *-----------------------------------------------------------------------
 * Returns       : -
 *-----------------------------------------------------------------------
 * Notes         : -
 *""FUNC COMMENT END""*****************************************************/
void    SIO3_RcvInt(void)
{
        UCHAR   data;
        UCHAR   status;

        status = S3RCNT;                                        /* Read receive status */
        data = S3RXB_L;                                         /* Read out received data */
        status |= S3RCNT;                                       /* Handle overrun */

        if(( status & ERS) == 0u) {                             /* Check receive error sum */
                RcvBuf3[ rx_cnt3++] = data;                     /* Store received data */

        } else {                                                /* Disable reception */

                /* Error processing by user */

                S3RCNT &= ~REN;                                 /* Disable reception */
                S3RCNT |= REN;                                  /* Enable reception */
        }
}

/*""FUNC COMMENT""***********************************************************
 * Function name: IntHand_SIO23()
 *-----------------------------------------------------------------------
 * Description   : Processing by SIO2,3 transmit/receive interrupt handlers
 *-----------------------------------------------------------------------
 * Argument      : -
 *-----------------------------------------------------------------------
 * Returns       : -
 *-----------------------------------------------------------------------
 * Notes         : Transmit interrupt is not considering
 *""FUNC COMMENT END""*****************************************************/
void    IntHand_SIO23(void)
{
```

```
        if(( SI23STAT & IRQR3) != 0u) {                     /* Judge interrupt request */
                SI23STAT = (~IRQR3) & 0x0fu;                /* Clear interrupt request
status */
                SIO3_RcvInt();                              /* Process SIO3 receive
interrupt */
        }

        if(( SI23STAT & IRQR2) != 0u) {                     /* Judge interrupt request */
                SI23STAT = (~IRQR2) & 0x0fu;                /* Clear interrupt request
status */
                SIO2_RcvInt();                              /* Process SIO2 receive
interrupt */
        }
}

```

4.4.3 startup.ms (partially omitted)

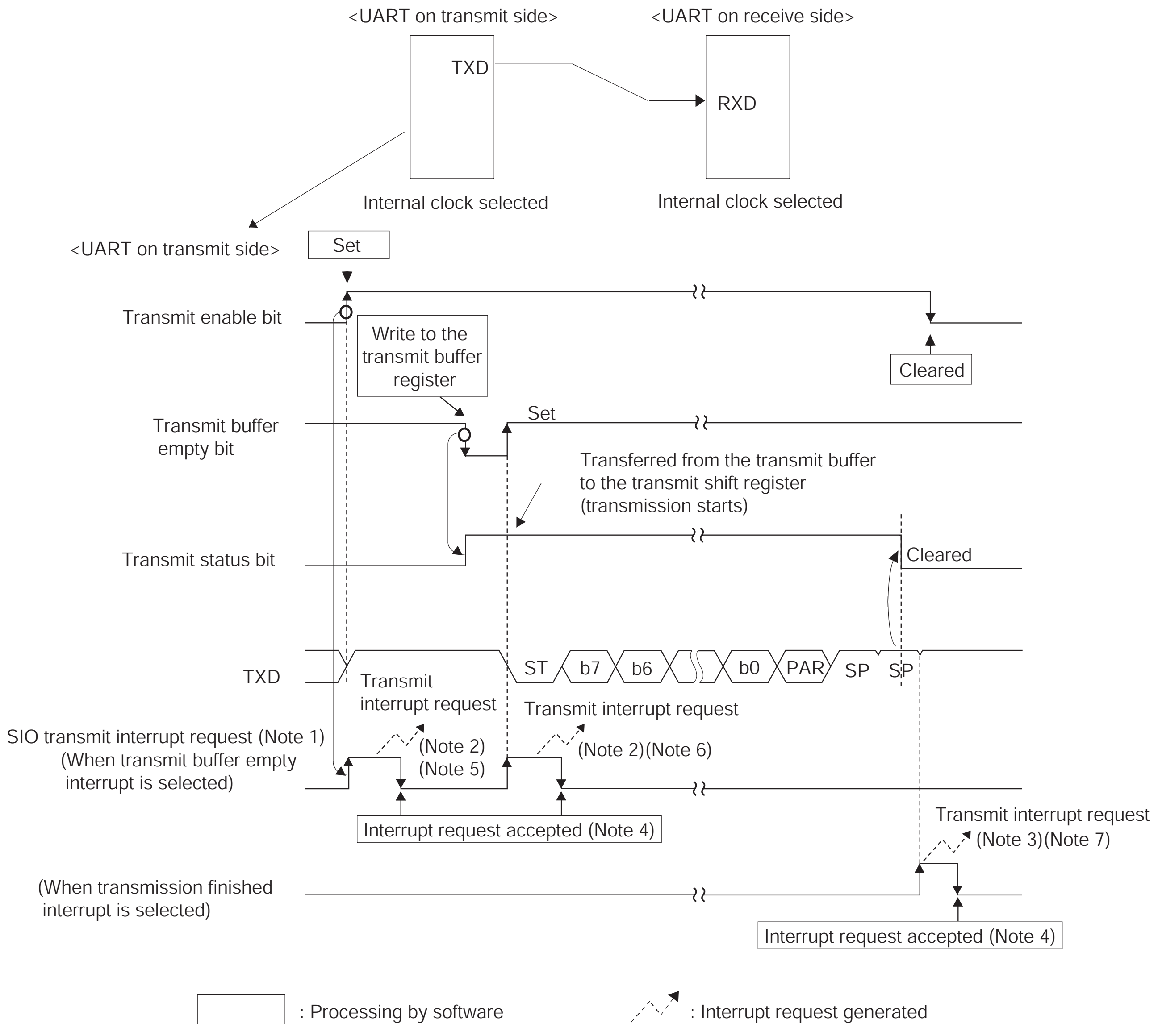
(Omitted)

```
;***************************************
; ICU Vector Table
;***************************************
;
        .SECTION        ICUVECT, DATA, ALIGN=4
;
        .IMPORT         $SIO1_RcvInt
;
vectbl:
        .DATA.W         EIT_reset               ; H'0000 0094   MJT Input Interrupt 4:TIN3-TIN6
        .DATA.W         EIT_reset               ; H'0000 0098   MJT Input Interrupt 3:TIN20-TIN23
        .DATA.W         EIT_reset               ; H'0000 009C   MJT Input Interrupt 2:TIN12-TIN19
        .DATA.W         EIT_reset               ; H'0000 00A0   MJT Input Interrupt 1:TIN0-TIN2
        .DATA.W         EIT_reset               ; H'0000 00A4   MJT Input Interrupt 0:TIN7-TIN11
        .DATA.W         EIT_reset               ; H'0000 00A8   MJT Output Interrupt 7:TMS0,TMS1
        .DATA.W         EIT_reset               ; H'0000 00AC   MJT Output Interrupt 6:TOP8,TOP9
        .DATA.W         EIT_reset               ; H'0000 00B0   MJT Output Interrupt 5:TOP10
        .DATA.W         EIT_reset               ; H'0000 00B4   MJT Output Interrupt 4:TIO4-TIO7
        .DATA.W         EIT_reset               ; H'0000 00B8   MJT Output Interrupt 3:TIO8,TIO9
        .DATA.W         EIT_reset               ; H'0000 00BC   MJT Output Interrupt 2:TOP0-TOP5
        .DATA.W         EIT_reset               ; H'0000 00C0   MJT Output Interrupt 1:TOP6,TOP7
        .DATA.W         EIT_reset               ; H'0000 00C4   MJT Output Interrupt 0:TIO0-TIO3
        .DATA.W         EIT_reset               ; H'0000 00C8   DMAC0-4 Interrupt:DMA0-DMA4
        .DATA.W         $SIO1_RcvInt            ; H'0000 00CC   SIO1 Receive Interrupt
        .DATA.W         EIT_reset               ; H'0000 00D0   SIO1 Transmit Interrupt
        .DATA.W         EIT_reset               ; H'0000 00D4   SIO0 Receive Interrupt
        .DATA.W         EIT_reset               ; H'0000 00D8   SIO0 Transmit Interrupt
        .DATA.W         EIT_reset               ; H'0000 00DC   A-D0 Conversion Interrupt
        .DATA.W         EIT_reset               ; H'0000 00E0   TID0 Output Interrupt
        .DATA.W         EIT_reset               ; H'0000 00E4   TOD0 Output Interrupt
        .DATA.W         EIT_reset               ; H'0000 00E8   DMAC5-9 Interrupt:DMA5-DMA9
        .DATA.W         EIT_reset               ; H'0000 00EC   SIO2,3 Transmit/Receive Interrupt
        .DATA.W         EIT_reset               ; H'0000 00F0   RTD Interrupt
        .DATA.W         EIT_reset               ; H'0000 00F4   TID1 Output Interrupt
        .DATA.W         EIT_reset               ; H'0000 00F8   TOD1,TOM0 Output Interrupt
        .DATA.W         EIT_reset               ; H'0000 00FC   SIO4,5 Transmit/Receive Interrupt
        .DATA.W         EIT_reset               ; H'0000 0100   A-D1 Conversion Interrupt
        .DATA.W         EIT_reset               ; H'0000 0104   TID2 Output Interrupt
        .DATA.W         EIT_reset               ; H'0000 0108   TML1 Input Interrupt
        .DATA.W         EIT_reset               ; H'0000 010C   CAN0 Transmit/Receive & Error Interrupt
        .DATA.W         EIT_reset               ; H'0000 0110   CAN1 Transmit/Receive & Error Interrupt
;
```

(The remainder of the program has been omitted)

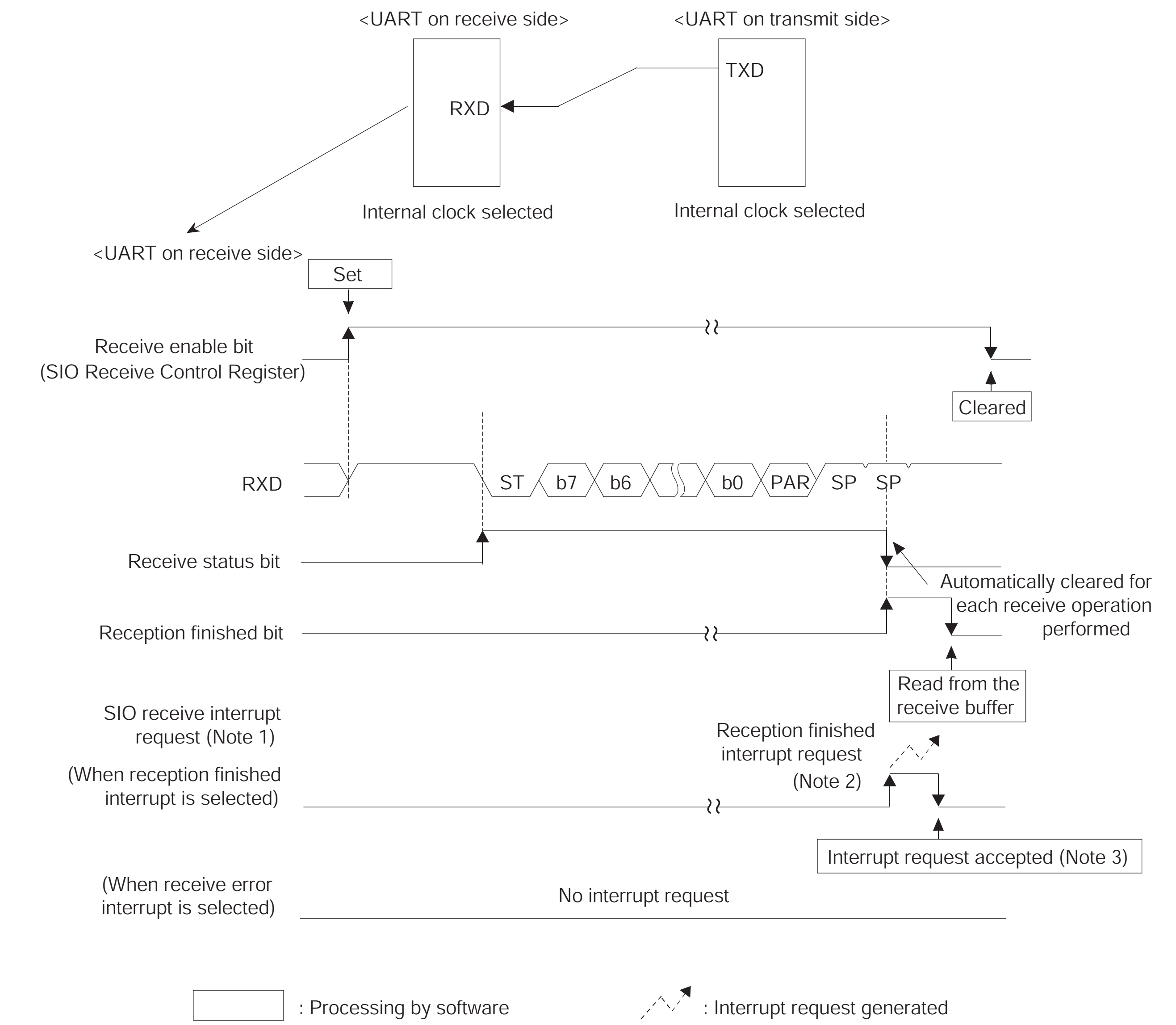
4.5 UART Operation Timing

The UART 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 when data in the transmit shift register has all been transmitted.

Figure 4.5.1 UART 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 UART Mode Receive Timing Diagram

5. Reference 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/

- Inquires for all Renesas products and technical inquiries for the M32R Family products, please contact:

  Customer Support Center: csc@renesas.com

Revision Record

Rev.	Date	Description	
		Page	Summary
1.00	Jan.12.06	—	First edition issued