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.

H8/300H Tiny Series

Serial Data Transmission in Synchronous Mode

Introduction

Four bytes of 8-bit data are transmitted by serial data transfer in synchronous mode.

Target Device

H8/300H Tiny Series H8/3664

Contents

1. Specification

1. Four bytes of 8-bit data are transmitted by serial data transfer in synchronous mode, as shown in figure 1.
2. Data is transmitted in a 4 µs clock cycle with an internal clock used as the transmit/receive clock (serial clock).
3. The data length of the transmit data is eight bits, and data is transmitted in the LSB-first format, starting from the least significant bit.

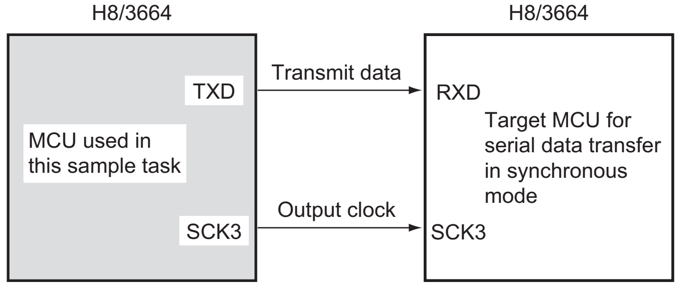
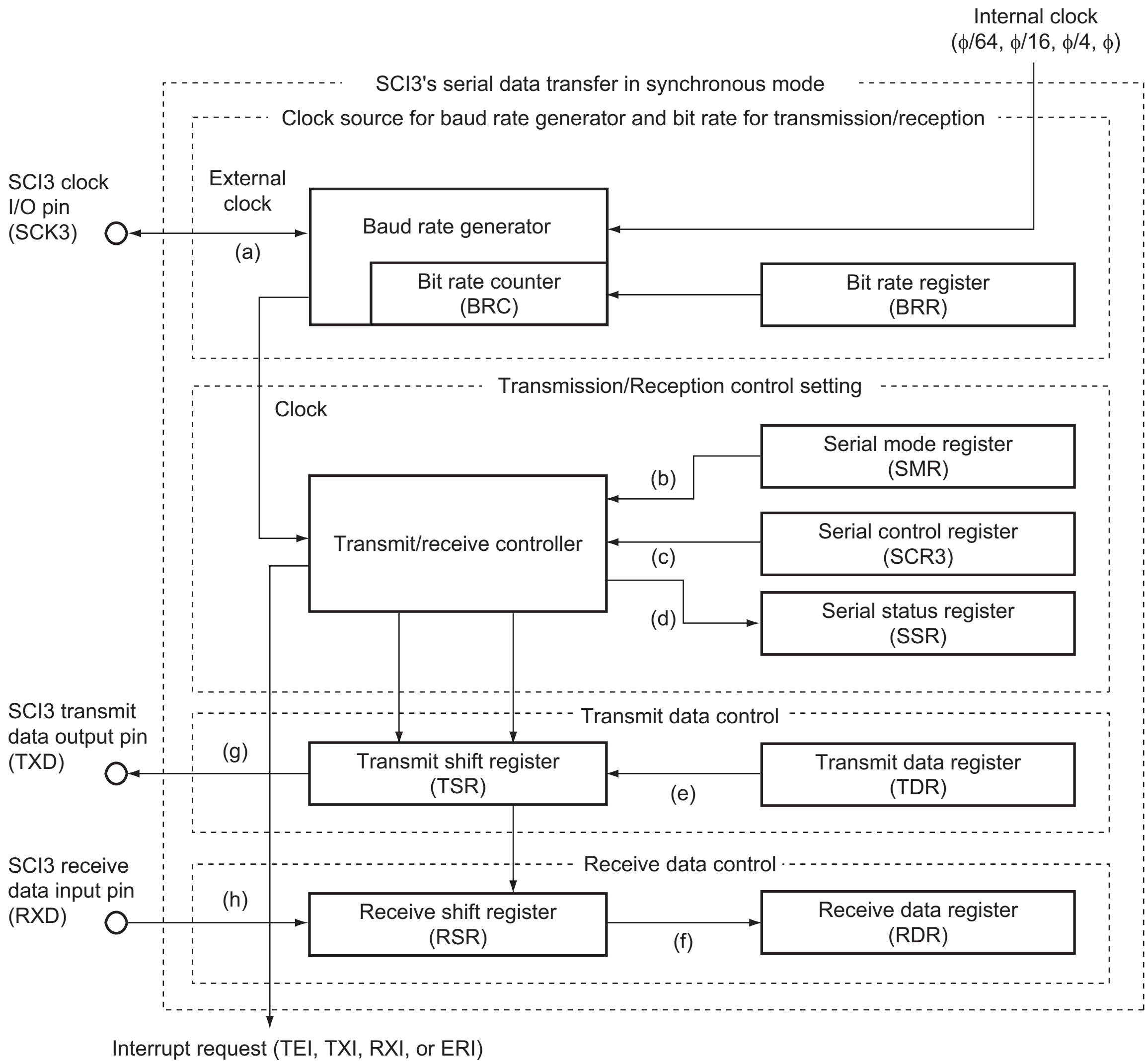


Figure 1 Serial Data Transmission in Synchronous Mode

2. Description of Functions Used

1. In this sample task, serial data is transmitted in synchronous mode via the serial communication interface (SCI). Figure 2 is a block diagram of serial data transmission in synchronous mode. The elements of the block diagram are described below.

- The system clock (ϕ) is a 16 MHz OSC clock that is used as a reference clock for operating the CPU and peripheral functions.
- In synchronous mode, the data length is eight bits.
- The receive shift register (RSR) is a register used to receive serial data. Serial data input to RSR from the RXD pin is set in the order in which it is received, starting from the LSB (bit 0), and converted to parallel data. When one byte of data is received, it is transferred to RDR automatically. RSR cannot be read from or written to directly by the CPU.
- The receive data register (RDR) is an 8-bit register that stores received serial data. When reception of one byte of data is finished, the received data is transferred from RSR to RDR, and the receive operation is completed. RSR is then enabled for reception. RSR and RDR are double-buffered, allowing consecutive receive operations. RDR is a read-only register, and cannot be written to by the CPU.
- The transmit shift register (TSR) is a register used to transmit serial data. Transmit data is first transferred from TDR to TSR, and serial data transmission is carried out by sending the data to the TXD pin in order, starting from the LSB (bit 0). When one byte of data is transmitted, the next byte of transmit data is automatically transferred from TDR to TSR, and transmission is started. Data transfer from TDR to TSR is not performed if no data has been written to TDR (if bit TDRE is set to 1). TSR cannot be read from or written to directly by the CPU.
- The transmit data register (TDR) is an 8-bit register that stores transmit data. When TSR is found to be empty, the transmit data written in TDR is transferred to TSR, and serial data transmission is started. Continuous transmission is possible by writing the next transmit data to TDR during TSR serial data transmission. TDR can be read from or written to by the CPU at any time.
- The serial mode register (SMR) is an 8-bit register used to set the serial data transfer format and to select the clock source for the baud rate generator.
- Serial control register 3 (SCR3) is an 8-bit register for selecting transmit or receive operation and the serial clock source.
- The serial status register (SSR) is an 8-bit register containing status flags that indicate the operational status of SCI3, and multiprocessor bits. Bits TDRE, RDRF, OER, PER, and FER can only be cleared to 0.
- The serial clock can be selected from a total of nine clocks: eight internal clocks and an external clock. When an internal clock is selected, the SCK3 pin functions as an output pin. When clock consecutive output mode is selected, the selected clock is consecutively output from the SCK3 pin. When an external clock is selected, the SCK3 pin functions as an input pin.
- In this sample task, the source of the serial clock (baud rate generator) is system clock/64 and the serial clock cycle is 4 µs.
- The SCI3 data transfer format uses a fixed 8-bit data length, and data is transmitted in the LSB-first format, starting from the least significant bit. Transmit data is output from one falling edge of the serial clock until the next falling edge. Receive data is latched at the rising edge of the serial clock.
- In this sample task, the data transfer format is set to an 8-bit data length, and 8-bit data is transmitted.
- The SCI3 clock (SCK3) pin is the SCI3 clock I/O pin.
- The SCI3 transmit data output (TXD) pin is the output pin for SCI3 transmit data.



Notes: (a) Outputs clock source (ϕ/64) for baud rate generator selected in SMR.
(b) Sets serial data transfer format and clock source for baud rate generator.
(c) Selects transmission or reception, and clock output pin in synchronous mode.
(d) Indicates operational status of SCI3 by a status flag (transmit data register empty).
(e) When TSR is found to be empty, sends the transmit data written in TDR to TSR.
(f) When one byte of data is received, sends receive data in RSR to RDR.
(g) Transmits transmit data.
(h) Receives receive data.

Figure 2 Serial Data Transmission in Synchronous Mode

2. Table 1 lists the function allocation for this sample task. The functions listed in table 1 are allocated for serial data transmission in synchronous mode.

Table 1 Function Allocation

Function	Function Assignment
TSR	Transmits serial data
TDR	Stores transmit data
SMR	Sets the serial data transfer format and clock source for the baud rate generator
SSR	Status flags indicating the operational status of SCI3
SCR3	Selects transmit operation and sets the SCK3 pin to function as a clock output pin
SCK3	SCI3 clock output pin
TXD	SCI3 transmit data output pin
PMR1	Sets TXD output pin

3. Operational Description

Figure 3 shows this sample task's principle of operation. The hardware and software processing shown in figure 3 performs serial data transmission in synchronous mode.

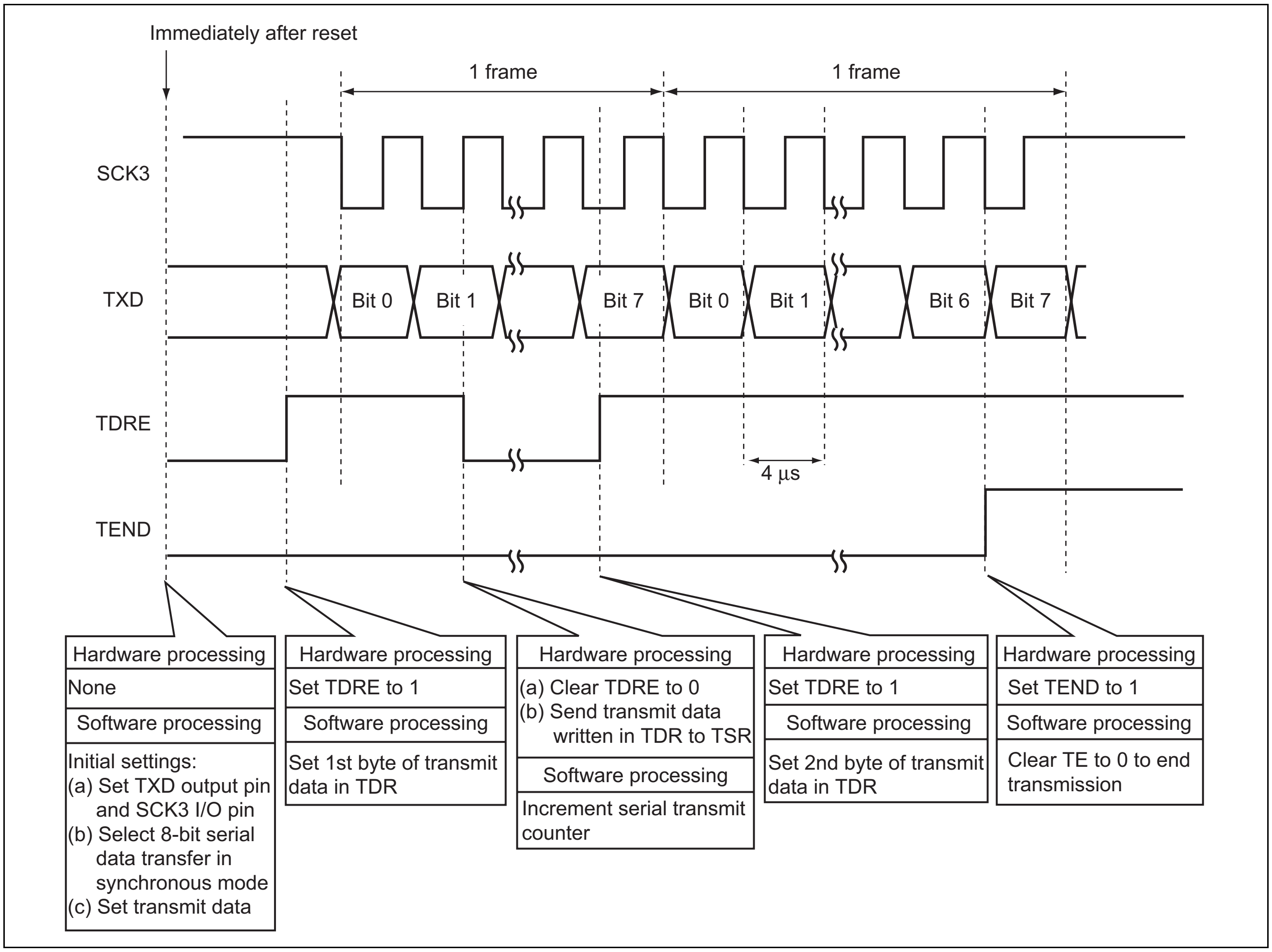


Figure 3 Operation Principle: Serial Data Transmission in Synchronous Mode

4. Description of Software

4.1 Description of Modules

Table 2 describes the software used in this sample task.

Table 2 Description of Module

Module Name	Label Name	Function
Main routine	main	Sets the transmit data, selects serial data transmission in synchronous mode, and stops SCI3 after four bytes of data have been transmitted.

4.2 Description of Argument

Table 3 describes the argument used in this sample task.

Table 3 Description of Argument

Argument Name	Function	Used in	Data Length	I/O
STD0 to STD3	Serial transmit data in synchronous mode	Main routine	1 byte	Output

4.3 Description of Internal Registers

Table 4 describes the internal registers used in this sample task.

Table 4 Description of Applicable Internal Registers

Register Name		Functional Description	Address	Setting
SMR	COM	Serial mode register (communication mode): When COM is set to 1, the communication mode is set to synchronous mode.	H'FFA8 Bit 7	1
	MP	Serial mode register (multiprocessor mode): This bit must be cleared to 0 (multiprocessor communication function disabled) in synchronous mode.	H'FFA8 Bit 2	0
	CKS1 CKS0	Serial mode register (clock select 1 and 0): When CKS1 and CKS0 are both set to 1, the clock source for the baud rate generator is set to system clock/64.	H'FFA8 Bit 1 Bit 0	CKS1 = 1 CKS0 = 1
SCR3	TE	Serial control register 3 (transmit enable): When TE is set to 1, transmit operation is enabled.	H'FFAA Bit 5	1
	CKE1 CKE0	Serial control register 3 (clock enable): When CKE1 and CKE0 are both cleared to 0, the clock source is set to an internal clock and the SCK3 pin functions as a clock output pin in synchronous mode.	H'FFAA Bit 1 Bit 0	CKE1 = 0 CKE0 = 0
TDR		Transmit data register: 8-bit register that stores the transmit data.	H'FFAB	—
SSR	TDRE	Serial status register (transmit data register empty): When TDRE is cleared to 0, the transmit data written in TDR has not been sent to TSR. When TDRE is set to 1, the transmit data has not been written to TDR, or the transmit data written in TDR has been sent to TSR.	H'FFAC Bit 7	1
	TEND	Serial status register (transmit end): When TEND is cleared to 0, transmission is in progress. When TEND is set to 1, transmission has completed.	H'FFAC Bit 2	1
PMR1	PMR11	Port mode register 1 (P22/TXD pin function switch): When PMR11 is set to 1, the P22/TXD pin functions as the TXD output pin.	H'FFE0 Bit 1	1

4.4 Description of RAM

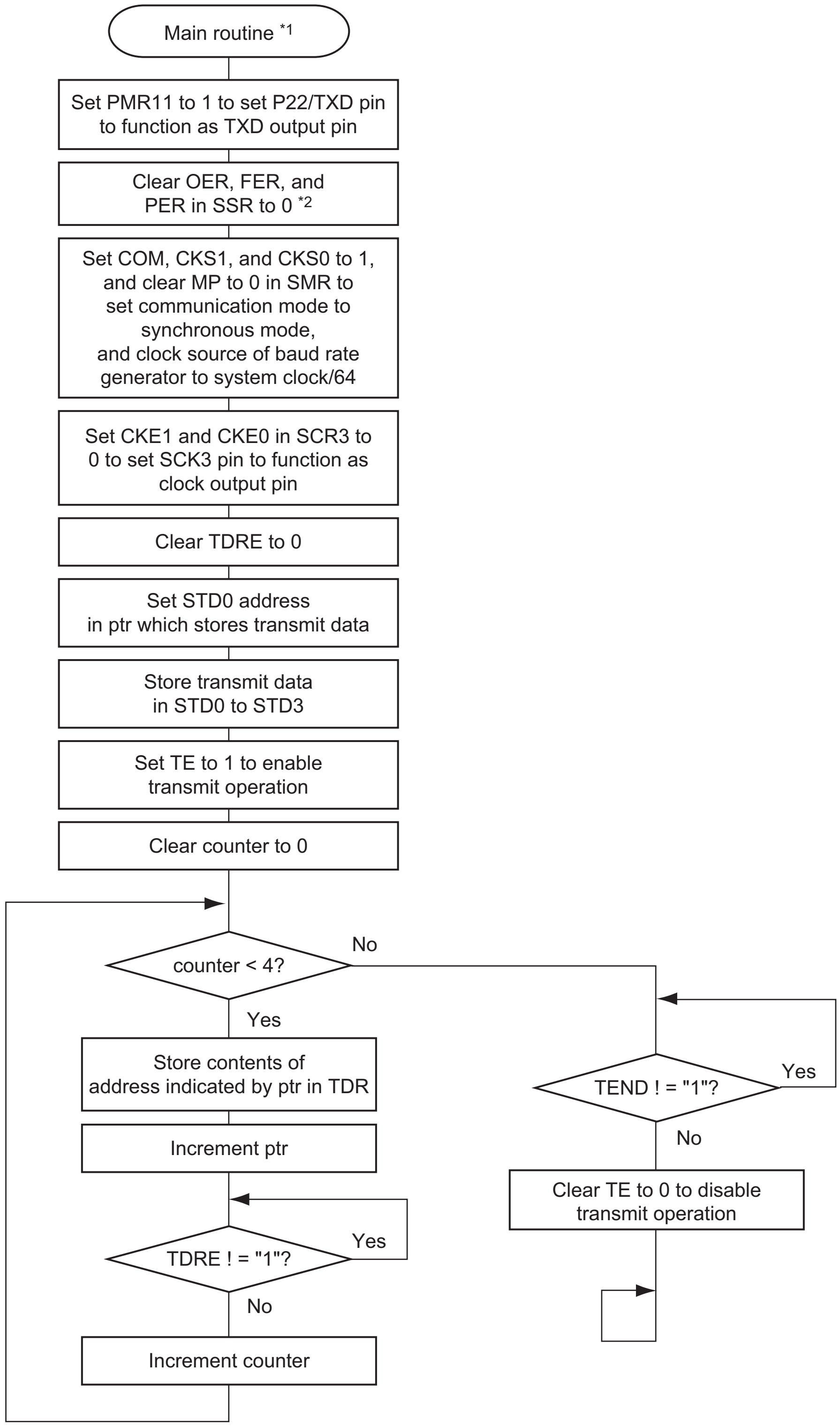
Table 5 describes the RAM used in this sample task.

Table 5 Description of RAM

Label Name	Function	Address	Used in
STD0	Stores the first byte of transmit data in serial data transmission in synchronous mode	H'FB80	Main routine
STD1	Stores the second byte of transmit data in serial data transmission in synchronous mode	H'FB81	Main routine
STD2	Stores the third byte of transmit data in serial data transmission in synchronous mode	H'FB82	Main routine
STD3	Stores the fourth byte of transmit data in serial data transmission in synchronous mode	H'FB83	Main routine
counter	8-bit counter for counting four transmit operations in serial data transmission in synchronous mode	H'FB84	Main routine

5. Flowchart

1. Main Routine



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

5.1 Link Address Designation

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

6. Program Listing

INIT.SRC (Program listing)

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

```
/**********************************************************/
/*                                                        */
/*  H8/300H Tiny Series -H8/3664-                         */
/*  Application Note                                      */
/*                                                        */
/*  'Synchronous Serial Data Transmission'                */
/*                                                        */
/*  Function                                              */
/*  : Serial Communication Interface                      */
/*     Synchronous Serial Interface                       */
/*     -Transmitting                                      */
/*                                                        */
/*  External Clock : 16MHz                                */
/*  Internal Clock : 16MHz                                */
/*  Sub Clock      : 32.768kHz                            */
/*                                                        */
/**********************************************************/

#include    <machine.h>

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

```
#define    SMR_BIT (*(struct BIT *)0xFFA8)           /* Serial Mode Register         */
#define    COM      SMR_BIT.b7                       /* Communication Mode                        */
#define    MP       SMR_BIT.b2                       /* Multiprocesor Mode                        */
#define    CKS1     SMR_BIT.b1                       /* Clock Select 1                            */
#define    CKS0     SMR_BIT.b0                       /* Clock Select 0                            */
#define    SCR3_BIT (*(struct BIT *)0xFFAA)          /* Serial Control Register 3                 */
#define    TE       SCR3_BIT.b5                      /* Transmit Enable                           */
#define    CKE1     SCR3_BIT.b1                      /* Clock Enable 1                            */
#define    CKE0     SCR3_BIT.b0                      /* Clock Enable 0                            */
#define    TDR      *(volatile unsigned char *)0xFFAB  /* Transmit Data Register                    */
#define    SSR_BIT (*(struct BIT *)0xFFAC)           /* Serial Status Register                    */
#define    TDRE     SSR_BIT.b7                       /* Transmit Data Register Empty              */
#define    OER      SSR_BIT.b5                       /* Overrun Erorr                             */
#define    FER      SSR_BIT.b4                       /* Framing Erorr                             */
#define    PER      SSR_BIT.b3                       /* Parity Erorr                              */
#define    TEND     SSR_BIT.b2                       /* Transmit End                              */
#define    PMR1_BIT (*(struct BIT *)0xFFE0)          /* Port Mode Register 1                      */
#define    PMR11    PMR1_BIT.b1                      /* TXD Output Terminal                       */

/***********************************************************/
/*     Function Definition                                 */
/***********************************************************/

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

/***********************************************************/
/*   RAM Allocation                                        */
/***********************************************************/
    unsigned char   STD0;
    unsigned char   STD1;
    unsigned char   STD2;
    unsigned char   STD3;
    unsigned char   counter;

/***********************************************************/
/*   Vector Address                                        */
/***********************************************************/
#pragma section     V1                               /* VECTOR SECTOIN SET                        */
void (*const VEC_TBL1[])(void) = {
/* 0x00 - 0x0f */
    INIT                                             /* 00 Reset                                  */
};

#pragma section                                      /* P                                         */
```

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

    PMR11 = 1;                                /* Initialize Output Port TXD               */

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

    COM = 1;                                  /* Initialize Communication Mode            */
    MP = 0;                                   /* Initialize Multiprocesor Mode            */
    CKS1 = 1;                                 /* Initialize Clock Select 1 φ/64           */
    CKS0 = 1;                                 /* Initialize Clock Select 0 φ/64           */

    CKE1 = 0;                                 /* Initialize Clock Enable 1                */
    CKE0 = 0;                                 /* Initialize Clock Enable 0                */

    TDRE = 0;                                 /* Clear TDRE                               */

    ptr = &STD0;                              /* Initialize Serial Transmitting Data Addreass */

    STD0 = 0x00;                              /* Set Serial Transfer Data 0               */
    STD1 = 0x55;                              /* Set Serial Transfer Data 1               */
    STD2 = 0xAA;                              /* Set Serial Transfer Data 2               */
    STD3 = 0xFF;                              /* Set Serial Transfer Data 3               */

    TE = 1;                                   /* Start Serial Transmitting                */

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

        TDR = *ptr;                           /* Write Serial Transmit Data to TDR        */

        ptr++;                                /* Increment Serial Transmittimg Data Address   */

        while(TDRE != 1){                     /* TDRE = 1 ?                               */
            ;
        }
    }

while(TEND != 1){                             /* End Serial Transmitting                  */
        ;
    }
    TE = 0;                                   /* Clear TE                                 */

    while(1)    {
        ;
```

Revision Record

Rev.	Date	Description	
		Page	Summary
1.00	Feb.26.03	—	First edition issued
2.00	Jul.22.05	—	Second edition issued