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.

SH7145F

Multiprocessor Communications

Summary

The SH7144 series is a single-chip microprocessor based on the SH-2 RISC (Reduced Instruction Set Computer) CPU core and integrating a number of peripheral functions.

This application note describes asynchronous multiprocessor communications using the SCI (Serial Communication Interface) module of the SH7145F. It is intended to be used as reference by users designing software applications.

The program examples contained in this application note have been tested. However, operation should be confirmed before using them in an actual application.

Device for Which Operation Has Been Confirmed

SH7145F

Contents

1. Specifications

As shown in figure 1, asynchronous multiprocessor communication is performed using channel 0 (ch0) of the SCI module of the SH7145F.

Multiprocessor communication allows data to be sent to multiple user-defined receiver stations by preceding the data with a specific receiver station ID. Each receiver station first checks the ID sent to it against its own ID. If they match, it receives the data that follows. The data that follows is not received if the IDs do not match.

In this task example serial data is transmitted from the SH7145 to user-defined receiver stations. The communication format is 192,000 bps, 8-bit, one stop bit, and no parity.

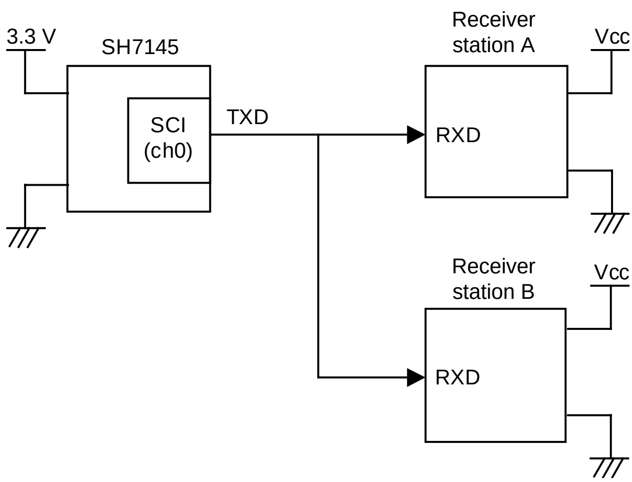


Figure 1 Asynchronous Serial Data Transmission by SH7145

Table 1 Asynchronous Multiprocessor Communications Format

Format Item	Setting
Bit rate	19200 bps
Data length	8 bits
Parity bit	No
Stop bit	1 bit
Serial/parallel conversion format	LSB first

2. Functions Used

In this task example the SCI (Serial Communication Interface) is used to perform asynchronous multiprocessor communications. Figure 2 shows a block diagram of channel 0 (ch0) of the SCI module. The functions of the elements shown in figure 2 are described below.

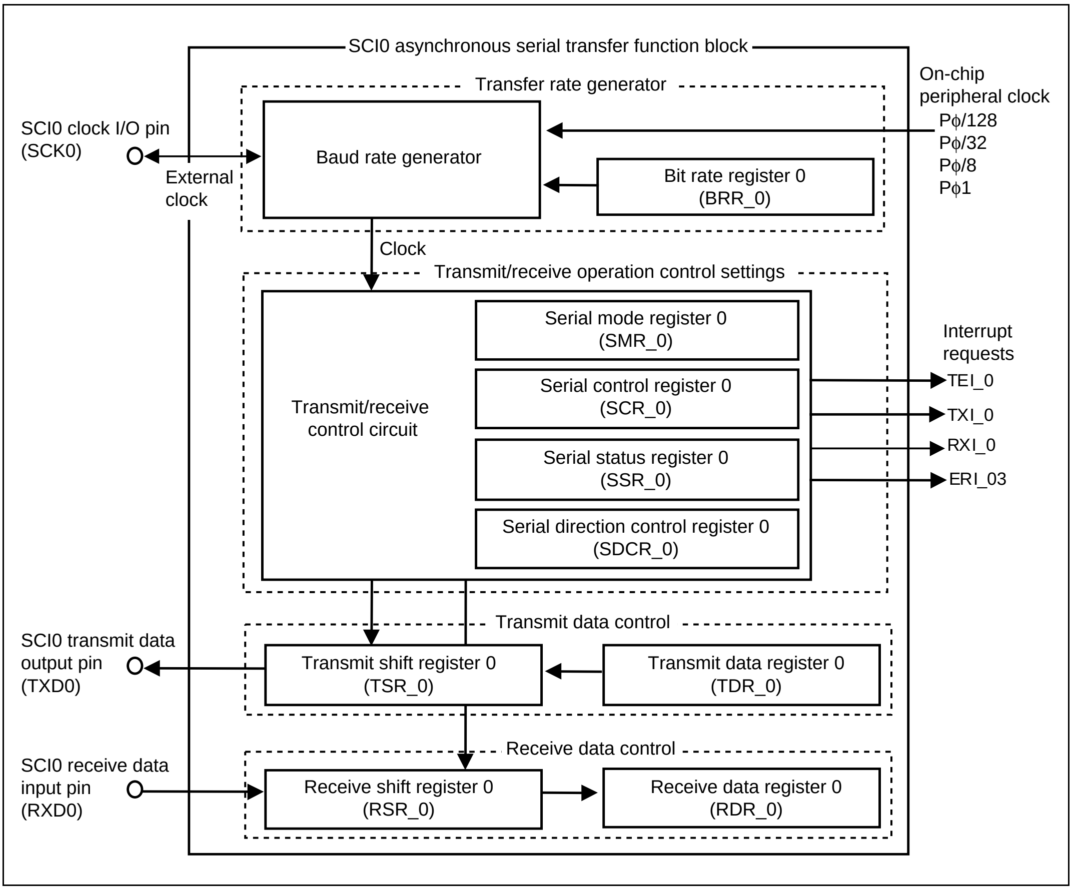


Figure 2 SCI (ch0) Block Diagram

- Asynchronous Mode

  Serial data communication is performed using synchronization by character unit. This allows serial communication with a standard dedicated asynchronous communication chip such as a Universal Asynchronous Receiver/Transmitter (UART) or Asynchronous Communication Interface Adapter (ACIA). In addition, the asynchronous mode supports serial communication among multiple processors (multiprocessor communication function).

- On-Chip Peripheral Clock Pϕ

  This is the reference clock for operation of on-chip peripheral functions. The clock signal is generated by a clock oscillator.

- Receive Shift Register (RSR_0)

  This register is used to receive serial data. Serial data is input to RSR_0 from the RxD_0 pin. When one frame of data has been received, it is automatically transferred to the receive data register (RDR_0). RSR_0 cannot be accessed by the CPU.

- Receive Data Register (RDR_0)

  Received data is stored in this 8-bit register. When one frame of data has been received, it is automatically transferred from RSR_0. RSR_0 and RDR_0 are in a double-buffer configuration, allowing continuous reception of data. RDR_0 is a receive-only register, so it can only be read by the CPU.

- Transmit Shift Register (TSR_0)

  This register is used to transmit serial data. In order to transmit data, the data is first transferred from the transmit data register (TDR_0) to TSR_1. Then the transmit data is output from the TxD_0 pin. TSR_0 cannot be accessed directly by the CPU.

- Transmit Data Register (TDR_0)

  Data to be transmitted is stored in this 8-bit register. When it is detected that TDR_0 is empty, data that has been written to TDR_0 is automatically transferred to TSR_0. TDR_0 and TSR_0 are in a double-buffer configuration. This allows data to be transferred to TSR_0 after one frame of data has been transmitted and the next frame of data is still being written to TDR_0, making possible continuous transmission of data. It is always possible to read or write to the TDR from the CPU, but before writing to the TDR it should be confirmed that the value of the TDRE bit in the serial status register (SSR_0) is 1.

- Serial Mode Register (SMR_0)

  This 8-bit register is used to select the serial data communication format and the clock source for the on-chip baud rate generator.

- Serial Control Register (SCR_0)

  This register is used for transmit and receive control, interrupt control, and to select the transmit and receive clock source.

- Serial Status Register (SSR_0)

  This register comprises the SCI1 status flag and the transmit and receive multiprocessor bits. TDRE, RDRF, ORER, PER, and FER can be cleared only.

- Serial Direction Control Register (SDCR_0)

  This register is used to select whether the LSB or MSB is first. For 8-bit communication either LSB-first or MSB-first may be selected, but LSB-first should be used for 7-bit communication.

- Bit Rate Register (BRR_0)

  This 8-bit register is used to adjust the bit rate. The SCI has independent baud rate generators for the individual channels, allowing different bit rates to be set for each. See the hardware manual for details on setting values, execution rate relationships, etc.

Table 2 shows the function allocations for the task example.

Table 2 Function Allocations

Function	Classification	Function Allocation
TXD0	Pin	Channel 0 transmit data output pin
RXD0	Pin	Channel 0 transmit data input pin
SMR_0	SCI0	Sets communication format to asynchronous mode
SCR_0	SCI0	Enables transmit operation
SSR_0	SCI0	Status flag showing SCI0 operation status
SDCR_0	SCI0	Specifies LSB-first
BRR_0	SCI0	Sets communication bit rate
TSR_0	SCI0	Register for transmitting serial data
TDR_0	SCI0	Register for storing transmit data
RSR_0	SCI0	Register for receiving serial data
RDR_0	SCI0	Register for storing receive data

3. Operation

Figure 3 shows reception operation when the do not IDs match, and figure 4 shows reception operation when they do match. To help explain figures 3 and 4, tables 3 and 4 list the software and hardware processing that is performed.

Transmission operation is identical to asynchronous serial data transmission, except that the MPBT bit in SSR_0 is set to 1 when transmitting the ID.

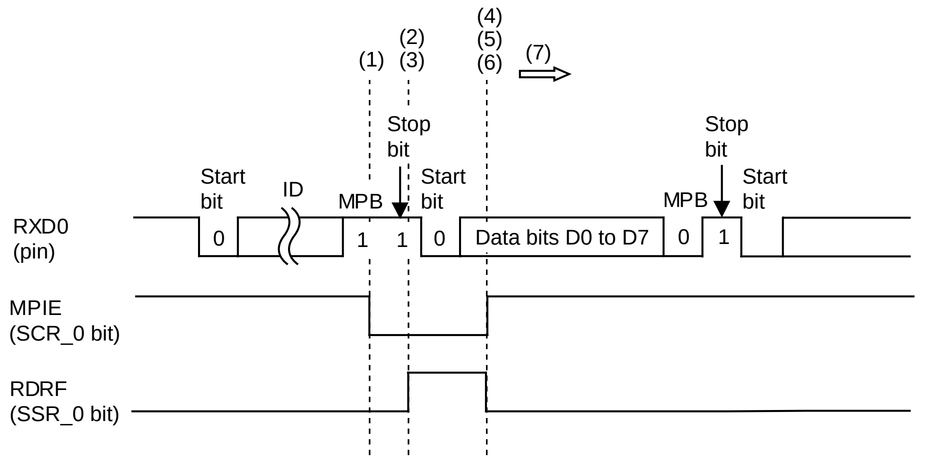


Figure 3 Reception Operation when IDs Do Not Match

Table 3 Processing

	Software Processing	Hardware Processing
(1)	—	Clear MPIE flag in SCR_0 to 0 (receive ID data)
(2)	—	RSR_0 receives serial data and transfers it to RDR_0
(3)	—	Set RDRF flag in SSR_1 to 1
(4)	Read data from RDR_1	—
(5)	Compare with own ID	—
(6)	Clear RDRF flag in SSR_1 and MPIE flag in SCR_0 to 0	—
(7)	—	Do not receive data transmitted when MPIE flag value was 1 because MPB bit is 0

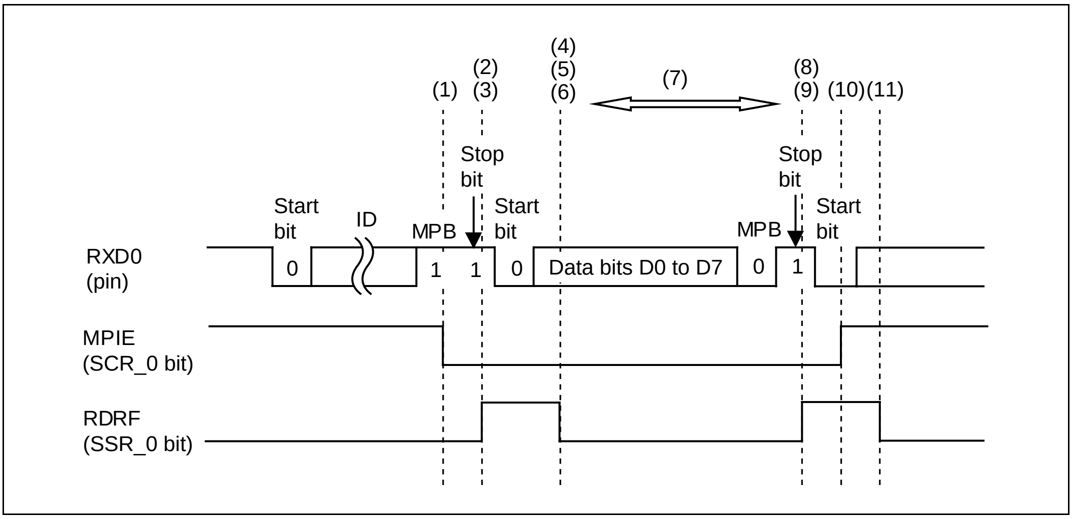


Figure 4 Reception Operation when IDs Match

Table 4 Processing

	Software Processing	Hardware Processing
(1)	—	Clear MPIE flag in SCR_0 to 0 (receive ID data)
(2)	—	RSR_0 receives serial data and transfers it to RDR_0
(3)	—	Set RDRF flag in SSR_1 to 1
(4)	Read data from RDR_1	—
(5)	Compare with own ID	—
(6)	Clear RDRF flag in SSR_1 to 0	—
(7)	—	Receive data transmitted when MPIE flag value was 1 because MPB bit is 1
(8)	—	RSR_0 receives serial data and transfers it to RDR_0
(9)	—	Set RDRF flag in SSR_1 to 1
(10)	Set MPIE flag in SCR_0 to 1	—
(11)	Read data from RDR_1	—

4. Software

(1) Module Descriptions

Table 5 lists the modules used in the task example.

Table 5 Module Descriptions

Module	Label	Function
Main routine	main	Calls modules
SCI routine	init_sci	Initial settings of SCI0
ID transmit routine	trans_id	Transmits ID data
Data transmit routine	trans_data	Transmits actual data

(2) Argument Descriptions

Table 6 lists the arguments used in the task example.

Table 6 Argument Descriptions

Argument	Function	Module
ID_1, ID_2	ID value	trans_id
DATA_1–4	Transmits data	trans_data

(3) On-Chip Register Descriptions

Table 7 lists the on-chip registers used in the task example. The set values shown are the values used in the task example and differ from the initial settings.

Table 7 On-Chip Register Descriptions

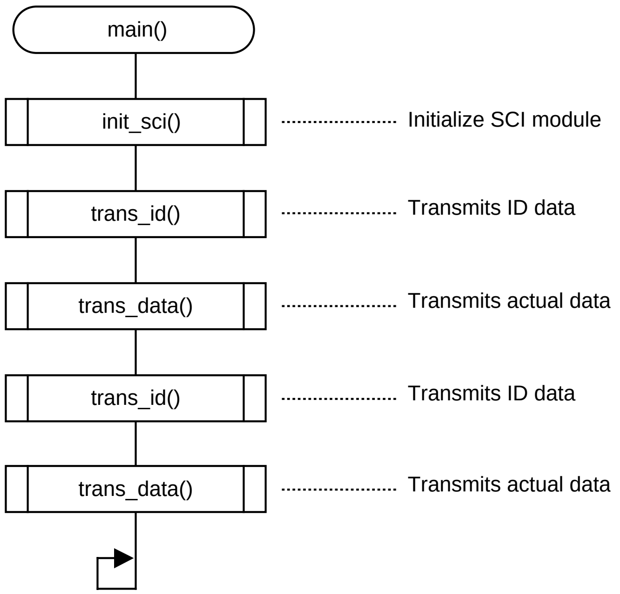
Register	Bit	Set Value	Function
MSTCR1	MSTP16	0	Module standby control register 1 SCI0 standby control bit Standby cancelled when MSTP16 = 0
SCR_0		H'20	Serial control register 0 (SCI_0) Transmit and receive control, interrupt control, transmit and receive clock source control
	TIE	0	Transmit interrupt enable TXI interrupt requests enabled when set to 1
	RIE	0	Receive interrupt enable RXI and ERI interrupt requests enabled when set to 1
	TE	1	Transmit enable Transmit operations enabled when set to 1
	RE	0	Receive enable Receive operations enabled when set to 1
	MPIE	0	Multiprocessor interrupt enable (In asynchronous mode, enabled when MP = 1 in SMR) ID data is received when MPBT = 1. Automatically cleared when ID data received
	TEIE	0	Transmit end interrupt enable TEI interrupt requests enabled when set to 1
	CKE1 CKE2	0 0	Clock enable 1, 0 Selects clock source and SCK pin function In the task example, clock source is on-chip clock and SCK pin is not used
SMR_0		H'04	Serial mode register 0 Selects communication format and the clock source for on-chip baud rate generator
	C/A	0	Communication mode Asynchronous mode when cleared to 0
	CHR	0	Character length (enabled in asynchronous mode only) 8-bit transmission and reception when 0
	PE	0	Parity enable (enabled in asynchronous mode only) No-parity transmission and reception when 0
	O/E	0	Parity mode (enabled in asynchronous mode when PE = 1) (In this example PE = 0 and this bit is disabled)

Register		Set Value	Function
	Bit		
SMR_0	STOP	0	Stop bit length (enabled in asynchronous mode only) 1-stop-bit transmission and reception when 0
	MP	1	Multiprocessor mode (enabled in asynchronous mode only) Multiprocessor communication enabled when 1
	CKS1 CKS2	0 0	Clock select 1, 0 When value is 00, Pϕ clock selected using on-chip baud rate generator as clock source
BRR_0		H'40	Bit rate register 1 8-bit register for adjusting bit rate
SDCR_0		H'F2	Serial direction control register 1 DIR bit (bit 3) selects LSB-first or MSB-first In task example, DIR = 0 (LSB-first)
SSR_0		H'xx	Serial status register 0 Comprises SCI0 status flag and transmit and receive multiprocessor bits Only 0 may be written to the status flag, to clear it
	TDRE	*	Transmit data register empty (status flag)
	RDRF	*	Receive data register full (status flag)
	ORER	*	Overrun error (status flag)
	FER	*	Framing error (status flag)
	PER	*	Parity error (status flag)
	TEND	*	Transmit end (status flag)
	MPB	0	Multiprocessor bit
	MPBT	0	Multiprocessor bit transfer
PACRL2	PA1MD1 PA1MD0	0 1	Port A control register L2 Function setting for port A multiplex pin (TXD0)
	PA0MD1 PA0MD0	0 1	Port A control register L2 Function setting for port A multiplex pin (RXD0)

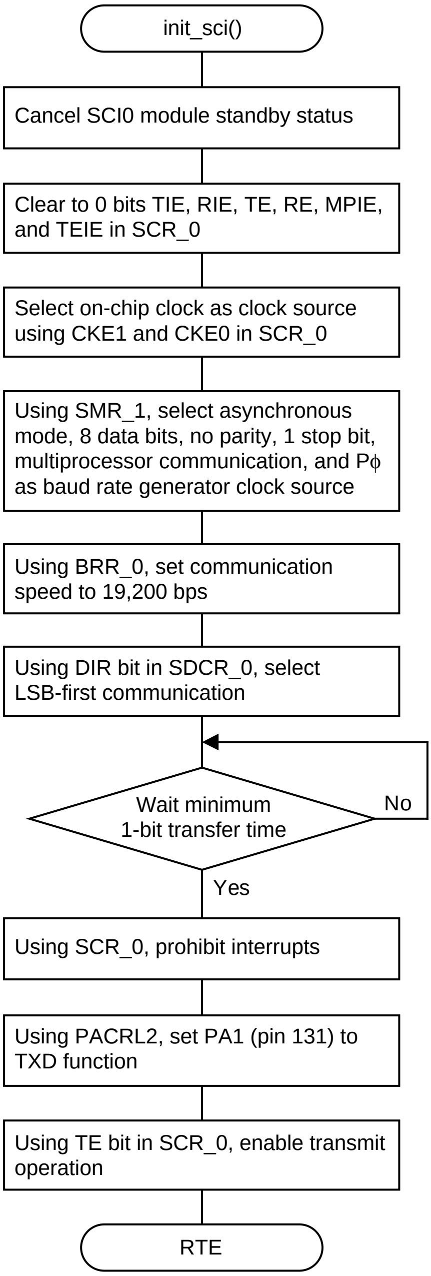
*: Can only be cleared to 0. Setting to 1 is performed by hardware.

5. Flowcharts

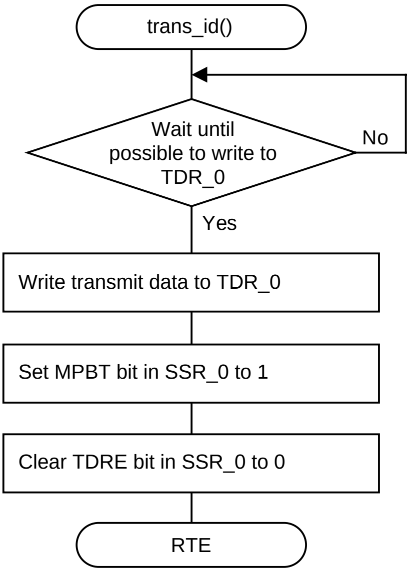
(1) Main Routine



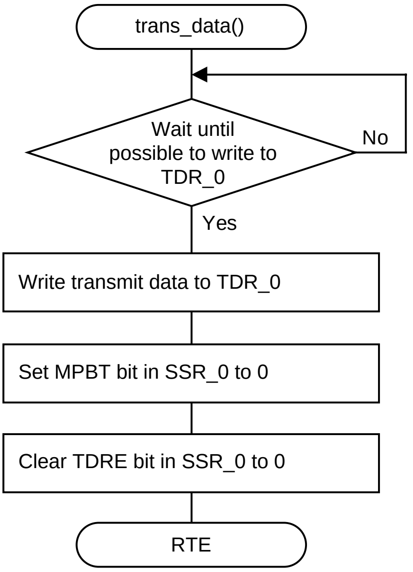
(2) SCI0 Initialize Routine



(3) Data Receive Routine



(4) Data Transfer Routine



6. Program Listing

```
/************************************************************/
/* SH7145F Application Note                                */
/*                                                          */
/* Function                                                 */
/*   :SCI0(multi processor mode)                           */
/*                                                          */
/* External input clock      : 12.5MHz                     */
/* Internal CPU clock        : 50MHz                       */
/* Internal peripheral clock : 25MHz                       */
/*                                                          */
/* Written                     2003/9   Rev.1.0            */
/************************************************************/

#include "iodefine.h"
#include <machine.h>

/*------------ Symbol Definition ---------------------------------------------*/

#define ID_1 0
#define ID_2 1

#define DATA_1 0x11
#define DATA_2 0x22

/*------------ Function Definition ------------------------------------------*/
void main(void);
extern void _INITSCT(void);

void init_sci(void);
void trans_id(char);
void trans_data(char);

void dummy_f(void);

/*------------ RAM allocation Definition -------------------------------------*/

/************************************************************/
/* main Program                                             */
/************************************************************/
void main( void )
{
   init_sci();              /* Initialize SCI             */

   trans_id(0);
   trans_data(DATA_1);

   trans_id(1);
   trans_data(DATA_2);
```

```
   while(1);                  /* LOOP                        */
}

/***************************************************************
  Function  : init_sci
  Operation : Initialize serial (sci0)
  Asynchronous multiprocessor communication
     -Data       : 8bit
     -Stop bit   : 1bit
     -Parity bit : No

***************************************************************/
void init_sci(void)
{
   unsigned long i;

   P_STBY.MSTCR1.BIT.MSTP16 = 0;       /* disable SCI0 standby mode         */

   /* Initialize SCI Asynchronous mode   *       */
   P_SCI0.SCR_0.BYTE &= 0x03 ;          /* clear TIE,RIE,TE,RE,MPIE,TEIE     */
   P_SCI0.SCR_0.BIT.CKE = 0;            /* clock:internal,SCK:output         */
   P_SCI0.SMR_0.BYTE = 0x04;            /* 8bit,No parity,1stop bit          */
         //   CA = 0;                   /* Asynchronous mode                 */
         //   CHR = 0;                  /* data length 8bits                 */
         //   PE = 0;                   /* No parity                         */
         //   OE = 0;                   /* (=0)even parity                   */
         //   STOP = 0;                 /* 1 stop bit                        */
         //   MP = 1;                   /* multiprocssor mode                */
         //   CKS = 0;                  /* clock source=Pφ(25MHz)           */

   P_SCI0.BRR_0 = 40;                   /* 19200bps@25MHz(Peripheral)        */
   P_SCI0.SDCR_0.BIT.DIR = 0;           /* LSB first send                    */

   for( i=0; i < 0x0500 ; i++);         /* Wait 1bit                         */

   P_SCI0.SCR_0.BIT.TIE = 0;            /* TXI3 interrupt disable            */
   P_SCI0.SCR_0.BIT.RIE = 0;            /* RXI3,ERI interrupt disable        */

   /* Initialize SCI1 PORT              */
   P_PORTA.PACRL2.BIT.PA1MD = 1;         /* set TXD0(PA1:131pin@SH7145)      */

   P_SCI3.SCR_3.BIT.TE = 1;              /* TE=1,Transmit Enable             */
}

/****************************************************************/
/* Function       : trans_id                                    */
/* Operation      : ID data output                              */
/* Argument       : num                                         */
/* Value returned : None                                        */
/****************************************************************/
```

```
void trans_id(char num){

   while(!(P_SCI0.SSR_0.BYTE & 0x80)){ /* Wait until data can be written to TDR */
      ;                                /* (until TDRE is set to 1)              */
   }

   if(num == 0){
      P_SCI0.TDR_0 = ID_1;              /* Write ID data to TDR                 */
   }
   else if(num == 1){
      P_SCI0.TDR_0 = ID_2;              /* Write ID data to TDR                 */
   }

   P_SCI0.SSR_0.BIT.MPBT = 1;           /* Set MPBT bit to 1                    */

   P_SCI0.SSR_0.BYTE &= 0x7F;           /* Clear flag, transmit                 */
}
/**************************************************************************/
/*  Function       : trans_data                                            */
/*  Operation      : Write 1 character to serial output                    */
/*  Argument       : d_num                                                 */
/*  Value returned : None                                                  */
/**************************************************************************/
void trans_data(char data){

   while(!(P_SCI0.SSR_0.BYTE & 0x80)){ /* Wait until data can be written to TDR */
      ;                                /* (until TDRE is set to 1)              */
   }

   P_SCI0.TDR_0 = data;                /* Write data to TDR                     */

   P_SCI0.SSR_0.BIT.MPBT = 0;          /* Clear MPBT bit to 0                   */

   P_SCI0.SSR_0.BYTE &= 0x7F;          /* Clear flag, transmit                  */
}

/**************************************
          Interrupt handling
**************************************/
#pragma interrupt(dummy_f)
void dummy_f(void)
{
   /*   Other Interrupt   */
}
```

Keep safety first in your circuit designs!

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

Notes regarding these materials

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