Introduction
The multiprocessor communication function of the serial communications interface (SCI) is used to transmit data to each processor.

Target Device
H8SX/1582F

Contents
1. Specifications .................................................................................................................................... 2
2. Conditions for Application ................................................................................................................. 3
3. Description of Modules Used ............................................................................................................ 4
4. Description of Operation ................................................................................................................... 8
5. Description of Software ................................................................................................................... 10
1. Specifications

Using the multiprocessor communication function, data H'B8 is transmitted to receiving station A and data H'DE is transmitted to receiving station B.

In multiprocessor communication, data can be transferred to the desired receiving station among the multiple serially-connected stations by transmitting an ID, uniquely assigned to each receiving station, followed by data. A receiving station compares the ID sent first with its own ID and receives the data that comes next only if the IDs match. If the IDs do not match, the receiving station does not receive the data that follows.

- Figure 1 shows an example of connection for inter-processor communication using multiprocessor format communication.
- The communication format is shown in table 1. A break is output when data transmission has finished.
- In this sample task, asynchronous transmission and reception of 128 bytes of data is controlled by software through the interrupt exception processing.

---

Figure 1  Inter-Processor Communication Using Multiprocessor Format Communication

<table>
<thead>
<tr>
<th>Format Item</th>
<th>Setting</th>
</tr>
</thead>
<tbody>
<tr>
<td>$P\phi$</td>
<td>16 MHz</td>
</tr>
<tr>
<td>Bit rate</td>
<td>19200 bps</td>
</tr>
<tr>
<td>Data length</td>
<td>8 bits</td>
</tr>
<tr>
<td>Parity bit</td>
<td>None</td>
</tr>
<tr>
<td>Stop bit</td>
<td>1 bit</td>
</tr>
<tr>
<td>Multiprocessor bit</td>
<td>1 bit</td>
</tr>
<tr>
<td>Serial/parallel conversion format</td>
<td>LSB-first</td>
</tr>
</tbody>
</table>
## 2. Conditions for Application

### Table 2  Conditions for Application

<table>
<thead>
<tr>
<th>Item</th>
<th>Contents</th>
</tr>
</thead>
<tbody>
<tr>
<td>Operating frequency</td>
<td>Input clock: 5 MHz</td>
</tr>
<tr>
<td></td>
<td>System clock (Iφ): 40 MHz</td>
</tr>
<tr>
<td></td>
<td>Peripheral module clock (Pφ): 20 MHz</td>
</tr>
<tr>
<td></td>
<td>External bus clock (Bφ): 20 MHz</td>
</tr>
<tr>
<td>Operating mode</td>
<td>Mode 3 (MD1 = 1, MD0 = 1)</td>
</tr>
<tr>
<td>Development tool</td>
<td>High-performance Embedded Workshop Version 4.00.02</td>
</tr>
<tr>
<td>C/C++ compiler</td>
<td>H8S, H8/300 Series C/C++ Compiler Version 6.01.00 (from Renesas Technology Corp.)</td>
</tr>
<tr>
<td>Compile option</td>
<td>-cpu = h8sxa:24:md, -code = machinecode, -optimize = 1, -regparam = 3, -speed = (register, shift, struct, expression)</td>
</tr>
</tbody>
</table>

### Table 3  Section Settings

<table>
<thead>
<tr>
<th>Address</th>
<th>Section Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>H'001000</td>
<td>P</td>
<td>Program area</td>
</tr>
<tr>
<td></td>
<td>C</td>
<td>Constant area</td>
</tr>
<tr>
<td>H'FF9000</td>
<td>B</td>
<td>Uninitialized data area (RAM area)</td>
</tr>
</tbody>
</table>
3. Description of Modules Used

3.1 Description of SCI_4

In this sample task, serial data transmission is performed in asynchronous mode using the SCI_4. Figure 2 shows a block diagram of the SCI_4 and the functions regarding figure 2 are described below.

- On-chip peripheral module clock P\(\Phi\)
  P\(\Phi\) is a reference clock used to drive the on-chip peripheral functions and is generated by the clock pulse generator.

- Receive shift register_4 (RSR_4)
  RSR_4 is a register used to receive serial data. When RSR_4 receives one frame of serial data input from the RxD4 pin, that data is automatically transferred to receive data register_4 (RDR_4). RSR_4 cannot be directly accessed by the CPU.

- Receive data register_4 (RDR_4)
  RDR_4 is an 8-bit register that stores receive data. When one frame of data has been received, the data is automatically transferred from RSR_4 to RDR_4. RSR_4 and RDR_4 form a double buffer, thus enabling continuous reception of data. Since RDR_4 is a register used only for data reception, RDR_4 can only be read by the CPU.

- Transmit shift register_4 (TSR_4)
  TSR_4 is a register used to transmit serial data. When transmitting data, data is first transferred from transmit data register_4 (TDR_4) to TSR_4, and then the transmit data is output from the TxD4 pin. TSR_4 cannot be directly accessed by the CPU.

- Transmit data register_4 (TDR_4)
  TDR_4 is an 8-bit register that stores transmit data. When TSR_4 is detected as empty, the data written to TDR_4 is automatically transferred to TSR_4. Since TDR_4 and TSR_4 form a double buffer, if the next data has been written to TDR_4 when one frame of data is transmitted, the written data is transferred to TSR_4, thus enabling continuous transmission of data. TSR_4 can always be read from or written to by the CPU. However, be sure to confirm that the TDRE bit in serial status register_4 (SSR_4) is 1 before writing to TDR_4.

- Serial mode register_4 (SMR_4)
  SMR_4 is an 8-bit register used to set the serial data transfer format and select the clock source for the on-chip baud rate generator.

- Serial control register_4 (SCR_4)
  SCR_4 is a register used to enable or disable transmission/reception or interrupt requests, and to select the transmit/receive clock source.

- Serial status register_4 (SSR_4)
  SSR_4 consists of the SCI_4 status flags and multiprocessor bits for transmission/reception. The TDRE, RDRF, ORER, PER, and FER bits in this register can only be cleared.

- Smart card mode register_4 (SCMR_4)
  SCMR_4 is a register used to select the smart card interface and its format. In this sample task, this register is set to select normal asynchronous or clock synchronous mode.
Figure 2   Block Diagram of SCI_4
3.2 Multiprocessor communication function

Multiprocessor communication is to perform data transmission and reception between multiple processors via a shared communication line by serial communication in asynchronous mode with a format (multiprocessor format) in which the multiprocessor bit is added.

When performing multiprocessor communication, a unique ID code is assigned to each receiving station. A serial communication cycle consists of an ID transmit cycle which specifies the receiving station and a data transmit cycle which transmits data to the specified receiving station.

The ID transmit cycle and data transmit cycle are distinguished by the multiprocessor bit. The multiprocessor bit is 1 for the ID transmit cycle and 0 for the data transmit cycle.

The transmitting station first transmits the ID code of the receiving station to which serial data is to be transmitted with a 1-valued multiprocessor bit added. Next, the transmitting station transmits the data with a 0-valued multiprocessor bit added.

A receiving station receives the communication data in which the multiprocessor bit is 1. It compares it with its own ID, and receives the subsequently transmitted data on a match. If the IDs do not match, the receiving station ignores the communication data until communication data with a 1-valued multiprocessor bit is transmitted again.

The transmit/receive format can be selected from four types when the multiprocessor format is specified. In these formats, parity bit setting is not available.

![Figure 3 Multiprocessor Format](image-url)
Table 4  Multiprocessor Communication Format

<table>
<thead>
<tr>
<th>SMR</th>
<th>CHR</th>
<th>MP</th>
<th>STOP</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
<th>11</th>
<th>12</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>START</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>8-bit data</td>
<td></td>
<td>MPB</td>
<td><strong>STOP</strong></td>
</tr>
<tr>
<td><strong>START</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>8-bit data</td>
<td></td>
<td>MPB</td>
<td><strong>STOP</strong></td>
</tr>
<tr>
<td><strong>START</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>7-bit data</td>
<td></td>
<td>MPB</td>
<td><strong>STOP</strong></td>
</tr>
<tr>
<td><strong>START</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>7-bit data</td>
<td></td>
<td>MPB</td>
<td><strong>STOP</strong></td>
</tr>
</tbody>
</table>

[Legend]

**START:** Start bit
**STOP:** Stop bit
**MPB:** Multiprocessor bit

Note: * This format is selected in this sample task.
4. Description of Operation

4.1 Overview of Operation

Receive operation when the IDs do not match is shown in figure 4 and receive operation when the IDs match is shown in figure 5. The hardware processing and software processing are shown in tables 5 and 6 for describing figures 4 and 5, respectively.

Transmit operation is the same as that for data transmission in asynchronous mode except for setting the MPBT bit in SSR_4 to 1 when transmitting the ID.

![Image of RXD_4 (pin), MPIE (bit in SCR_4), RDRF (bit in SSR_4), Start bit, ID, MPB, Stop bit, Data, MPB, Start bit, Stop bit](image)

**Figure 4  Receive Operation with Non-Matching IDs**

<table>
<thead>
<tr>
<th>Table 5  Hardware and Software Processing</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Hardware Processing</strong></td>
</tr>
<tr>
<td>(1) Clear the MPIE flag in SCR_4 to 0 (ID data reception).</td>
</tr>
<tr>
<td>(2) RSR_4 receives serial data and transfers it to RDR_4.</td>
</tr>
<tr>
<td>(3) Set the RDRF flag in SSR_4 to 1.</td>
</tr>
<tr>
<td>(4) —</td>
</tr>
<tr>
<td>(5) —</td>
</tr>
<tr>
<td>(6) —</td>
</tr>
<tr>
<td>(7) —</td>
</tr>
<tr>
<td>(8) Does not receive data because the MPB bit of the data sent while the MPIE flag being 1 is 0.</td>
</tr>
</tbody>
</table>
### Table 6  Hardware and Software Processing

<table>
<thead>
<tr>
<th></th>
<th>Hardware Processing</th>
<th>Software Processing</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Clear the MPIE flag in SCR_4 to 0 (ID data reception).</td>
<td>—</td>
</tr>
<tr>
<td>2</td>
<td>RSR_4 receives serial data and transfers it to RDR_4.</td>
<td>—</td>
</tr>
<tr>
<td>3</td>
<td>Set the RDRF flag in SSR_4 to 1.</td>
<td>—</td>
</tr>
<tr>
<td>4</td>
<td>—</td>
<td>Read the data in RDR_4.</td>
</tr>
<tr>
<td>5</td>
<td>—</td>
<td>Clear the RDRF flag in SSR_4 to 0.</td>
</tr>
<tr>
<td>6</td>
<td>—</td>
<td>Compare the data read from RDR_4 with the ID of its own.</td>
</tr>
<tr>
<td>7</td>
<td>Receive data.</td>
<td>—</td>
</tr>
<tr>
<td>8</td>
<td>RSR_4 receives serial data and transfers it to RDR_4.</td>
<td>—</td>
</tr>
<tr>
<td>9</td>
<td>Set the RDRF flag in SSR_4 to 1.</td>
<td>—</td>
</tr>
<tr>
<td>10</td>
<td>—</td>
<td>Read the data in RDR_4.</td>
</tr>
<tr>
<td>11</td>
<td>—</td>
<td>Clear the RDRF flag in SSR_4 to 0.</td>
</tr>
</tbody>
</table>

![Figure 5  Receive Operation with Matching IDs](image-url)
5. Description of Software

5.1 List of Functions

Table 7 List of Functions

<table>
<thead>
<tr>
<th>Function Name</th>
<th>Functions</th>
</tr>
</thead>
</table>
| init          | Initialization routine  
Cancels module stop mode, sets the clocks, and calls the main function. |
| main          | Main routine  
Makes the initial settings for the SCI and performs communication with multiple devices at a transfer rate of 19200 bps. |
| sci4_rcv1byte | 1-byte reception by receiving station  
Checks the received ID code and receives one byte of data. |
| sci4_trs1byte | 1-byte transmission by transmitting station  
Transmits an ID code and one byte of data. |

5.2 RAM Usage

Table 8 RAM Usage

<table>
<thead>
<tr>
<th>Type</th>
<th>Label Name</th>
<th>Description</th>
<th>Used In</th>
</tr>
</thead>
<tbody>
<tr>
<td>unsigned char</td>
<td>r_buf</td>
<td>Stores the receive data</td>
<td>main</td>
</tr>
</tbody>
</table>

5.3 Constants

Table 9 List of Constants

<table>
<thead>
<tr>
<th>Constant Name</th>
<th>Description</th>
<th>Used In</th>
</tr>
</thead>
</table>
| ID1           | ID number of receiving station A  
Setting: H’01 | main    |
| ID2           | ID number of receiving station B  
Setting: H’02 | main    |

5.4 Data Table

Table 10 Data Table

<table>
<thead>
<tr>
<th>Type</th>
<th>Array Name</th>
<th>Description</th>
<th>Used In</th>
</tr>
</thead>
</table>
| unsigned char | t_buf[2]   | Stores the transmit data  
t_buf[0] = H'B8, t_buf[1] = H'DE | main    |
5.5 Macro Definition

Table 11 Macro Definition

<table>
<thead>
<tr>
<th>Identifier</th>
<th>Description</th>
<th>Used In</th>
</tr>
</thead>
<tbody>
<tr>
<td>TRSSTA</td>
<td>Generates a program for the transmitting station</td>
<td>main</td>
</tr>
<tr>
<td>RCVSTA_A</td>
<td>Generates a program for receiving station A</td>
<td>main</td>
</tr>
<tr>
<td>RCVSTA_B</td>
<td>Generates a program for receiving station B</td>
<td>main</td>
</tr>
</tbody>
</table>

5.6 Description of Functions

5.6.1 init Function

(1) Functional overview

Initialization routine which cancels module stop mode, sets the clocks, and calls the main function.

(2) Argument

None

(3) Return value

None

(4) Description of internal registers

The internal registers used in this sample task are described below. The setting values shown below are the values used in this sample task and differ from their initial values.

- System clock control register (SCKCR)  
  Address: H'FFFDC4

<table>
<thead>
<tr>
<th>Bit</th>
<th>Bit Name</th>
<th>Setting</th>
<th>R/W</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>10</td>
<td>ICK2</td>
<td>0</td>
<td>R/W</td>
<td>System Clock (Iφ) Select</td>
</tr>
<tr>
<td>9</td>
<td>ICK1</td>
<td>0</td>
<td>R/W</td>
<td>These bits select the system clock frequency. The CPU, DMAC, and DTC modules are driven by the system clock. 000: Input clock x 8</td>
</tr>
<tr>
<td>8</td>
<td>ICK0</td>
<td>0</td>
<td>R/W</td>
<td>These bits select the system clock frequency. The CPU, DMAC, and DTC modules are driven by the system clock. 000: Input clock x 8</td>
</tr>
<tr>
<td>6</td>
<td>PCK2</td>
<td>0</td>
<td>R/W</td>
<td>Peripheral Module Clock (Pφ) Select</td>
</tr>
<tr>
<td>5</td>
<td>PCK1</td>
<td>0</td>
<td>R/W</td>
<td>These bits select the frequency of the peripheral module clock. 001: Input clock x 4</td>
</tr>
<tr>
<td>4</td>
<td>PCK0</td>
<td>1</td>
<td>R/W</td>
<td>001: Input clock x 4</td>
</tr>
<tr>
<td>2</td>
<td>BCK2</td>
<td>0</td>
<td>R/W</td>
<td>External Bus Clock (Bφ) Select</td>
</tr>
<tr>
<td>1</td>
<td>BCK1</td>
<td>0</td>
<td>R/W</td>
<td>These bits select the frequency of the external bus clock. 001: Input clock x 4</td>
</tr>
<tr>
<td>0</td>
<td>BCK0</td>
<td>1</td>
<td>R/W</td>
<td>001: Input clock x 4</td>
</tr>
</tbody>
</table>

- MSTPCRA, MSTPCRB, and MSTPCRC are the registers that control module stop mode. Setting the bits in these registers places the corresponding modules in module stop mode, and clearing the bits cancels module stop mode.
### Module stop control register A (MSTPCRA)

<table>
<thead>
<tr>
<th>Bit</th>
<th>Bit Name</th>
<th>Setting</th>
<th>R/W</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>15</td>
<td>ACSE</td>
<td>0</td>
<td>R/W</td>
<td>All-module-clock-stop mode enable</td>
</tr>
</tbody>
</table>

Enables or disables transition to all-module-clock-stop mode. If this bit is set to 1, all-module-clock-stop mode is entered when the SLEEP instruction is executed by the CPU while all the modules under control of the MSTPCR registers are placed in module stop mode. In all-module-clock-stop mode, even the bus controller and I/O ports are stopped to reduce the supply current.

0: Disables transition to all-module-clock-stop mode.
1: Enables transition to all-module-clock-stop mode.

<table>
<thead>
<tr>
<th>Bit</th>
<th>Bit Name</th>
<th>Setting</th>
<th>R/W</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>13</td>
<td>MSTPA13</td>
<td>1</td>
<td>R/W</td>
<td>DMA controller (DMAC)</td>
</tr>
<tr>
<td>12</td>
<td>MSTPA12</td>
<td>1</td>
<td>R/W</td>
<td>Data transfer controller (DTC)</td>
</tr>
<tr>
<td>4</td>
<td>MSTPA4</td>
<td>1</td>
<td>R/W</td>
<td>A/D converter (unit 1)</td>
</tr>
<tr>
<td>3</td>
<td>MSTPA3</td>
<td>1</td>
<td>R/W</td>
<td>A/D converter (unit 0)</td>
</tr>
<tr>
<td>1</td>
<td>MSTPA1</td>
<td>1</td>
<td>R/W</td>
<td>16-bit timer pulse unit (TPU channels 11 to 6)</td>
</tr>
<tr>
<td>0</td>
<td>MSTPA0</td>
<td>1</td>
<td>R/W</td>
<td>16-bit timer pulse unit (TPU channels 5 to 0)</td>
</tr>
</tbody>
</table>

### Module stop control register B (MSTPCRB)

<table>
<thead>
<tr>
<th>Bit</th>
<th>Bit Name</th>
<th>Setting</th>
<th>R/W</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>15</td>
<td>MSTPB15</td>
<td>1</td>
<td>R/W</td>
<td>Programmable pulse generator (PPG)</td>
</tr>
<tr>
<td>12</td>
<td>MSTPB12</td>
<td>0</td>
<td>R/W</td>
<td>Serial communication interface_4 (SCI_4)</td>
</tr>
<tr>
<td>11</td>
<td>MSTPB11</td>
<td>1</td>
<td>R/W</td>
<td>Serial communication interface_3 (SCI_3)</td>
</tr>
</tbody>
</table>

### Module stop control register C (MSTPCRC)

<table>
<thead>
<tr>
<th>Bit</th>
<th>Bit Name</th>
<th>Setting</th>
<th>R/W</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>10</td>
<td>MSTPC10</td>
<td>1</td>
<td>R/W</td>
<td>Synchronous serial communication unit 2 (SSU_2)</td>
</tr>
<tr>
<td>9</td>
<td>MSTPC9</td>
<td>1</td>
<td>R/W</td>
<td>Synchronous serial communication unit 1 (SSU_1)</td>
</tr>
<tr>
<td>8</td>
<td>MSTPC8</td>
<td>1</td>
<td>R/W</td>
<td>Synchronous serial communication unit 0 (SSU_0)</td>
</tr>
<tr>
<td>1</td>
<td>MSTPC1</td>
<td>0</td>
<td>R/W</td>
<td>On-chip RAM_1 (H'FF9000 to H'FFBFFF)</td>
</tr>
<tr>
<td>0</td>
<td>MSTPC0</td>
<td>0</td>
<td>R/W</td>
<td>Always write the same value to the MSTPC1 and MSTPC0 bits.</td>
</tr>
</tbody>
</table>
(5) Flowchart

```
init

CCR = H'80
   Initialize CCR to disable interrupts.

SCKCR = H'0011
   • Clock (×8, ×4, ×4)

MSTPCRA = H'3FFF
MSTPCRB = H'EFFF
MSTPCRC = H'FF00
   • Cancel module stop mode.

main()
   Execute the main routine.
```

5.6.2 main Function

(1) Functional overview
Main routine which transmits and receives one byte of data.

(2) Argument
None

(3) Return value
None

(4) Description of internal registers
The internal registers used in this sample task are described below. The setting values shown below are the values used in this sample task and differ from their initial values.

- Serial mode register_4 (SMR_4)
  Address: H'FFFE90

<table>
<thead>
<tr>
<th>Bit</th>
<th>Bit Name</th>
<th>Setting</th>
<th>R/W</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>7</td>
<td>C/A</td>
<td>0</td>
<td>R/W</td>
<td>Communication Mode</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0: Operates in asynchronous mode</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>1: Operates in clock synchronous mode</td>
</tr>
<tr>
<td>6</td>
<td>CHR</td>
<td>0</td>
<td>R/W</td>
<td>Character Length</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0: 8-bit length data is transmitted</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>1: 7-bit length data is transmitted</td>
</tr>
<tr>
<td>3</td>
<td>STOP</td>
<td>0</td>
<td>R/W</td>
<td>Stop Bit Length</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Selects the stop bit length for transmission.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0: 1 stop bit</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>1: 2 stop bits</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>In reception, only the first stop bit is checked, regardless of the setting of this bit. If the second stop bit is 0, it is treated as the start bit of the next transmit frame.</td>
</tr>
<tr>
<td>2</td>
<td>MP</td>
<td>1</td>
<td>R/W</td>
<td>Multiprocessor Mode</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0: Multiprocessor function is disabled</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>1: Multiprocessor function is selected</td>
</tr>
<tr>
<td>1</td>
<td>CKS1</td>
<td>0</td>
<td>R/W</td>
<td>Clock Select 1, 0</td>
</tr>
<tr>
<td>0</td>
<td>CKS0</td>
<td>0</td>
<td>R/W</td>
<td>00: Clock source of on-chip baud rate generator is Pϕ.</td>
</tr>
</tbody>
</table>

- Bit rate register_4 (BRR_4)
  Address: H'FFFE91
  Function: 8-bit register for adjusting the bit rate. When Pϕ = 20 MHz, CKS1 and CKS0 bits = B'00 in SMR_4, and BRR_4 = 32, the bit rate is set at 19200 bps.
  Setting: 32
- **Serial control register_4 (SCR_4)**
  
  **Address:** H'FFFE92

<table>
<thead>
<tr>
<th>Bit</th>
<th>Bit Name</th>
<th>Setting</th>
<th>R/W</th>
<th>Function</th>
</tr>
</thead>
</table>
| 7   | TIE      | 0       | R/W | Transmit Interrupt Enable  
|     |          |         |     | 0: TXI interrupt is disabled 
|     |          |         |     | 1: TXI interrupt is enabled |
| 6   | RIE      | 0       | R/W | Receive Interrupt Enable  
|     |          |         |     | 0: RXI and ERI interrupts are disabled 
|     |          |         |     | 1: RXI and ERI interrupts are enabled |
| 5   | TE       | 0       | R/W | Transmit Enable  
|     |          |         |     | 0: Transmission is disabled 
|     |          |         |     | 1: Transmission is enabled |
| 4   | RE       | 0       | R/W | Receive Enable  
|     |          |         |     | 0: Reception is disabled 
|     |          |         |     | 1: Reception is enabled |
| 3   | MPIE     | 0       | R/W | Multiprocessor Interrupt Enable  
|     |          |         |     | (Valid when the MP bit in SMR is 1 in asynchronous mode) 
|     |          |         |     | When this bit is set to 1, receive data whose multiprocessor bit is 0 is skipped and setting of the RDRF, FER, and OER status flags in SSR is disabled. When receiving data whose multiprocessor bit is 1, this bit is automatically cleared and normal receive operation is resumed. For details, refer to section 13.5, Multiprocessor Communication Function, in the hardware manual. 
|     |          |         |     | When data with a 0-valued multiprocessor bit is being received, transfer of receive data from RSR to RDR, receive error detection, and setting of the RDRF, FER, and OER flags in SSR are not performed. When data with a 1-valued multiprocessor bit is received, the MPB bit in SSR is set to 1, the MPIE bit is automatically cleared to 0, RXI and ERI interrupt requests are enabled (when the TIE and RIE bits in SCR are set to 1), and setting of the RDRF, FER and OER flags is enabled. |
| 2   | TEIE     | 0       | R/W | Transmit-End Interrupt Enable  
|     |          |         |     | 0: TEI interrupt is disabled 
|     |          |         |     | 1: TEI interrupt is enabled |
| 1   | CKE1     | 0       | R/W | Clock Enable 1, 0 |
| 0   | CKE0     | 0       | R/W | These bits select the clock source.  
|     |          |         |     | 00: On-chip baud rate generator |

- **Transmit data register_4 (TDR_4)**
  
  **Address:** H'FFFE93

  Function: 8-bit readable/writable register that stores transmit data

  Setting: Undefined
### Serial status register_4 (SSR_4)

Address: H'FFFE94

<table>
<thead>
<tr>
<th>Bit</th>
<th>Bit Name</th>
<th>Setting</th>
<th>R/W</th>
<th>Function</th>
</tr>
</thead>
</table>
| 7   | TDRE     | Undefined | R/(W)* | Transmit Data Register Empty  
Indicates whether TDR contains transmit data.  
[Setting conditions]  
• When the TE bit in SCR is 0  
• When data is transferred from TDR to TSR  
[Clearing conditions]  
• When 0 is written to TDRE after reading TDRE = 1  
• When the DMAC transfers transmit data to TDR due to a TXI interrupt |
| 6   | RDRF     | 0        | R/(W)* | Receive Data Register Full  
Indicates whether RDR contains receive data.  
[Setting condition]  
• When reception is finished successfully, and receive data is transferred from RSR to RDR  
[Clearing conditions]  
• When 0 is written to RDRF after reading RDRF = 1  
• When the DMAC or DTC transfers data from RDR due to an RXI interrupt  
Even if the RE bit in SCR is cleared to 0, the RDRF flag is unaffected and its previous state retained.  
If reception of the next data is completed while the RDRF flag is still set to 1, an overrun error occurs and the received data is lost. |
| 5   | ORER     | 0        | R/(W)* | Overrun Error  
[Setting condition]  
• When an overrun error occurs during reception  
[Clearing condition]  
• When 0 is written to ORER after reading ORER = 1 |
| 4   | FER      | 0        | R/(W)* | Framing Error  
[Setting condition]  
• When a framing error occurs during reception  
[Clearing condition]  
• When 0 is written to FER after reading FER = 1 |
| 3   | PER      | 0        | R/(W)* | Parity Error  
[Setting condition]  
• When a parity error occurs during reception  
[Clearing condition]  
• When 0 is written to PER after reading PER = 1 |
<table>
<thead>
<tr>
<th>Bit</th>
<th>Bit Name</th>
<th>Setting</th>
<th>R/W</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>TEND</td>
<td>Undefined</td>
<td>R</td>
<td>Transmit End</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td><strong>[Setting conditions]</strong></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>• When the TE bit in SCR is 0</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>• If the TDRE bit is 1 when the last bit in the transmit character is</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>transmitted</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td><strong>[Clearing conditions]</strong></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>• When 0 is written to TDRE after reading TDRE = 1</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>• When the DMAC writes transmit data to TDR due to a TXI interrupt</td>
</tr>
<tr>
<td>1</td>
<td>MPB</td>
<td>Undefined</td>
<td>R</td>
<td>Multiprocessor Bit</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Stores the multiprocessor bit value in the receive frame. This bit</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>value does not change when the RE bit in SCR is 0.</td>
</tr>
<tr>
<td>0</td>
<td>MPBT</td>
<td>Undefined</td>
<td>R/W</td>
<td>Multiprocessor Bit Transfer</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Sets the multiprocessor bit value that is to be added to the transmit</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>frame.</td>
</tr>
</tbody>
</table>

**Note:**  * Only 0 can be written to clear the flag.

- Receive data register_4 (RDR_4)  
  Address: H'FFFE95  
  Function: 8-bit read-only register that stores receive data  
  Setting: Undefined

- Smart card mode register_4 (SCMR_4)  
  Address: H'FFFE96

<table>
<thead>
<tr>
<th>Bit</th>
<th>Bit Name</th>
<th>Setting</th>
<th>R/W</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>SMIF</td>
<td>0</td>
<td>R/W</td>
<td>Smart Card Interface Mode Select</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0: Operates in normal asynchronous or clock synchronous mode</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>1: Operates in smart card interface mode</td>
</tr>
</tbody>
</table>

- Port 1 input buffer control register (P1ICR)  
  Address: H'FFFFB90

<table>
<thead>
<tr>
<th>Bit</th>
<th>Bit Name</th>
<th>Setting</th>
<th>R/W</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>P15ICR</td>
<td>1</td>
<td>R/W</td>
<td>0: Input buffer of pin P15 is disabled. Input signal is fixed high.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>1: Input buffer of pin P15 is enabled. Pin state is reflected in the</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>peripheral module.</td>
</tr>
</tbody>
</table>
(5) Flowchart

```
main

Clear TE and RE bits in SCR_4 to 0.
P15ICR = 1
Set P15 (RxD4) pin as input pin.

SCR_4 = H'00
Select on-chip baud rate generator as the clock source.

SMR_4 = H'04
SCMR_4 &= H'F2
Set communication format.
• Asynchronous mode
• Data length: 8 bits
• 1 stop bit
• Multiprocessor mode

BRR_4 &= 32
Bit rate: 19200 bps

i = 0

Wait for 1-bit period
i < 128?

Yes

SSR_4 = H'87
Clear RDRF, ORER, PER, and FER bits in SSR_4 to 0.

Receiving station A (RCVSTA_A)

Transmitting station (TRSSTA)

sci4_trs1byte (ID1, t_buf[0])
Transmit one byte by multiprocessor communication.
• ID code: ID1
• Transmit data: t_buf[0]

r_buf = sci4_rcv1byte (ID1)
Receive one byte by multiprocessor communication.
• ID code: ID1

sc4_trs1byte (ID2, t_buf[1])
Transmit one byte by multiprocessor communication.
• ID code: ID2
• Transmit data: t_buf[1]

End

Receiving station B (RCVSTA_B)

r_buf = sci4_rcv1byte (ID2)
Receive one byte by multiprocessor communication.
• ID code: ID2
```
### 5.6.3 sci4_rcv1byte Function

(1) Functional overview

Checks the received ID code and receives 1-byte data.

(2) Argument

<table>
<thead>
<tr>
<th>Type</th>
<th>Variable Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>unsigned char</td>
<td>id</td>
<td>ID code</td>
</tr>
</tbody>
</table>

(3) Return value

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>unsigned char</td>
<td>Receive data</td>
</tr>
</tbody>
</table>

(4) Description of internal registers

The internal registers used in this sample task are described below. The setting values shown below are the values used in this sample task and differ from their initial values.

- **Serial control register_4 (SCR_4)**
  
  Address: H'FFFE92

<table>
<thead>
<tr>
<th>Bit</th>
<th>Bit Name</th>
<th>Setting</th>
<th>R/W</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>4</td>
<td>RE</td>
<td>0</td>
<td>R/W</td>
<td>Receive Enable</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0: Reception is disabled</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>1: Reception is enabled</td>
</tr>
<tr>
<td>3</td>
<td>MPIE</td>
<td>0</td>
<td>R/W</td>
<td>Multiprocessor Interrupt Enable</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>(Valid when the MP bit in SMR is 1 in asynchronous mode)</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>When this bit is set to 1, receive data whose multiprocessor bit is 0 is skipped and setting of the RDRF, FER, and OER status flags in SSR is disabled. When receiving data whose multiprocessor bit is 1, this bit is automatically cleared and normal receive operation is resumed. For details, refer to section 13.5, Multiprocessor Communication Function, in the hardware manual. When data including MPB = 0 is being received, transfer of receive data from RSR to RDR, receive error detection, and setting of the RDRF, FER, and OER flags in SSR are not performed. When data including MPB = 1 is received, the MPB bit in SSR is set to 1, the MPIE bit is automatically cleared to 0, RXI and ERI interrupt requests are enabled (when the TIE and RIE bits in SCR are set to 1), and setting of the FER and ORER flags is enabled.</td>
</tr>
</tbody>
</table>
### Serial status register_4 (SSR_4)

- **Address:** H'FFFE94

<table>
<thead>
<tr>
<th>Bit</th>
<th>Bit Name</th>
<th>Setting</th>
<th>R/W</th>
<th>Function</th>
</tr>
</thead>
</table>
| 6   | RDRF     | 0       | R/W*| Receive Data Register Full  
  - Indicates whether RDR contains receive data.  
  - [Setting condition]  
    - When reception is finished successfully, and receive data is transferred from RSR to RDR  
  - [Clearing conditions]  
    - When 0 is written to RDRF after reading RDRF = 1  
    - When the DMAC or DTC transfers data from RDR due to an RXI interrupt  
  Even if the RE bit in SCR is cleared to 0, the RDRF flag is unaffected and its previous state retained.  
  If reception of the next data is completed while the RDRF flag is still set to 1, an overrun error occurs and the received data is lost. |
| 5   | ORER     | 0       | R/W*| Overrun Error  
  - [Setting condition]  
    - When an overrun error occurs during reception  
  - [Clearing condition]  
    - When 0 is written to ORER after reading ORER = 1 |
| 4   | FER      | 0       | R/W*| Framing Error  
  - [Setting condition]  
    - When a framing error occurs during reception  
  - [Clearing condition]  
    - When 0 is written to FER after reading FER = 1 |
| 3   | PER      | 0       | R/W*| Parity Error  
  - [Setting condition]  
    - When a parity error occurs during reception  
  - [Clearing condition]  
    - When 0 is written to PER after reading PER = 1 |

**Note:**  
* Only 0 can be written to clear the flag.

### Receive data register_4 (RDR_4)

- **Address:** H'FFFE95  
  - Function: 8-bit register that stores receive data  
  - Setting: Undefined
(5) Flowchart

- `sci4_rcv1byte`
- `RE = 1`
- `MPIE bit in SCR_4 = 1`
- `In SSR_4, ORER = 1, FER = 1, or PER = 1?`
  - Yes
  - `tmp = RDRF bit in SSR_4`
  - `tmp = 0?`
    - Yes
    - Receive ID.
    - `rcv = RDR_4`
    - `RDRF bit in SSR_4 = 0`
    - `rcv = id?`
      - Yes
      - Receive data ≠ ID code?
    - No
    - `In SSR_4, ORER = 1, FER = 1, or PER = 1?`
      - Yes
      - `tmp = RDRF bit in SSR_4`
      - `tmp = 0?`
        - Yes
        - Receive data.
        - `rcv = RDR_4`
        - `RDRF bit in SSR_4 = 0`
        - `RE = 0`
        - Return rcv.
  - No
  - `rcv = RDR_4`
  - `RDRF bit in SSR_4 = 0`
  - `RE = 0`
  - Return rcv.

- `End`
5.6.4 sci4_trs1byte Function

(1) Functional overview
Transmits the ID code and 1-byte data.

(2) Arguments

<table>
<thead>
<tr>
<th>Type</th>
<th>Variable Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>unsigned char</td>
<td>id</td>
<td>ID code</td>
</tr>
<tr>
<td>unsigned char</td>
<td>tdt</td>
<td>Transmit data</td>
</tr>
</tbody>
</table>

(3) Return value
None

(4) Description of internal registers
The internal registers used in this sample task are described below. The setting values shown below are the values used in this sample task and differ from their initial values.

- Serial control register_4 (SCR_4)  
  Address: H'FFFE92

<table>
<thead>
<tr>
<th>Bit</th>
<th>Bit Name</th>
<th>Setting</th>
<th>R/W</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>TE</td>
<td>0</td>
<td>R/W</td>
<td>Transmit Enable</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0: Transmission is disabled</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>1: Transmission is enabled</td>
</tr>
</tbody>
</table>

- Transmit data register_4 (TDR_4)  
  Function: 8-bit register that stores transmit data  
  Setting: Undefined

- Serial status register_4 (SSR_4)  
  Address: H'FFFE94

<table>
<thead>
<tr>
<th>Bit</th>
<th>Bit Name</th>
<th>Setting</th>
<th>R/W</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>7</td>
<td>TDRE</td>
<td>0</td>
<td>R/(W)*</td>
<td>Transmit Data Register Empty</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0: Indicates that the transmit data written to TDR has not been transferred to TSR</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>1: Indicates that transmit data has not been written to TDR or the transmit data written to TDR has been transferred to TSR</td>
</tr>
<tr>
<td>2</td>
<td>TEND</td>
<td>Undefined</td>
<td>R/(W)*</td>
<td>Transmit End</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0: Transmission is in progress</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>1: Transmission has ended</td>
</tr>
<tr>
<td>0</td>
<td>MPBT</td>
<td>0/1</td>
<td>R/W</td>
<td>Multiprocessor Bit Transfer</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Sets the multiprocessor bit value that is to be added to the transmit frame.</td>
</tr>
</tbody>
</table>

Note: * Only 0 can be written to clear the flag.
(5) Flowchart

- **sci4_trs1byte**
  - TE = 1

  - **TDRE bit in SSR_4 == 0?**
    - Yes
      - TDR_4 still has data
    - No
      - TDR_4 has no data

      - TDR_4 = id
        - Set ID code in TDR_4.
      - MPBT bit in SSR_4 = 1
        - Multiprocessor bit = 1
      - TDRE bit in SSR_4 = 0

      - **TDRE bit in SSR_4 == 0?**
        - Yes
          - TDR_4 still has data
        - No
          - TDR_4 has no data

          - TDR_4 = tdt
            - Set transmit data in TDR_4.
          - MPBT bit in SSR_4 = 0
            - Multiprocessor bit = 0
          - TDRE bit in SSR_4 = 0

          - **TEND bit in SSR_4 == 0?**
            - Yes
              - Transmission is in progress
            - No
              - Transmission is complete

              - TE = 0

              - End
Revision Record

<table>
<thead>
<tr>
<th>Rev.</th>
<th>Date</th>
<th>Page</th>
<th>Summary</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.00</td>
<td>Mar.10.06</td>
<td>—</td>
<td>First edition issued</td>
</tr>
</tbody>
</table>
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.