To our customers,

---

**Old Company Name in Catalogs and Other Documents**

On April 1\textsuperscript{st}, 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](http://www.renesas.com)

---

April 1\textsuperscript{st}, 2010
Renesas Electronics Corporation

Issued by: Renesas Electronics Corporation ([http://www.renesas.com](http://www.renesas.com))

Notice

1. All information included in this document is current as of the date this document is issued. Such information, however, is subject to change without any prior notice. Before purchasing or using any Renesas Electronics products listed herein, please confirm the latest product information with a Renesas Electronics sales office. Also, please pay regular and careful attention to additional and different information to be disclosed by Renesas Electronics such as that disclosed through our website.

2. Renesas Electronics does not assume any liability for infringement of patents, copyrights, or other intellectual property rights of third parties by or arising from the use of Renesas Electronics products or technical information described in this document. No license, express, implied or otherwise, is granted hereby under any patents, copyrights or other intellectual property rights of Renesas Electronics or others.

3. You should not alter, modify, copy, or otherwise misappropriate any Renesas Electronics product, whether in whole or in part.

4. Descriptions of circuits, software and other related information in this document are provided only to illustrate the operation of semiconductor products and application examples. You are fully responsible for the incorporation of these circuits, software, and information in the design of your equipment. Renesas Electronics assumes no responsibility for any losses incurred by you or third parties arising from the use of these circuits, software, or information.

5. When exporting the products or technology described in this document, you should comply with the applicable export control laws and regulations and follow the procedures required by such laws and regulations. You should not use Renesas Electronics products or the technology described in this document for any purpose relating to military applications or use by the military, including but not limited to the development of weapons of mass destruction. Renesas Electronics products and technology may not be used for or incorporated into any products or systems whose manufacture, use, or sale is prohibited under any applicable domestic or foreign laws or regulations.

6. Renesas Electronics has used reasonable care in preparing the information included in this document, but Renesas Electronics does not warrant that such information is error free. Renesas Electronics assumes no liability whatsoever for any damages incurred by you resulting in errors in or omissions from the information included herein.

7. Renesas Electronics products are classified according to the following three quality grades: “Standard”, “High Quality”, and “Specific”. The recommended applications for each Renesas Electronics product depend on the product’s quality grade, as indicated below. You must check the quality grade of each Renesas Electronics product before using it in a particular application. You may not use any Renesas Electronics product for any application categorized as “Specific” without the prior written consent of Renesas Electronics. Further, you may not use any Renesas Electronics product for any application for which it is not intended without the prior written consent of Renesas Electronics. Renesas Electronics shall not be in any way liable for any damages or losses incurred by you or third parties arising from the use of any Renesas Electronics product for an application categorized as “Specific” or for which the product is not intended where you have failed to obtain the prior written consent of Renesas Electronics. The quality grade of each Renesas Electronics product is “Standard” unless otherwise expressly specified in a Renesas Electronics data sheets or data books, etc.

- “Standard”: Computers; office equipment; communications equipment; test and measurement equipment; audio and visual equipment; home electronic appliances; machine tools; personal electronic equipment; and industrial robots.
- “High Quality”: Transportation equipment (automobiles, trains, ships, etc.); traffic control systems; anti-disaster systems; anti-crime systems; safety equipment; and medical equipment not specifically designed for life support.
- “Specific”: Aircraft; aerospace equipment; submersible repeaters; nuclear reactor control systems; medical equipment or systems for life support (e.g. artificial life support devices or systems); surgical implantations, or healthcare intervention (e.g. excision, etc.), and any other applications or purposes that pose a direct threat to human life.

8. You should use the Renesas Electronics products described in this document within the range specified by Renesas Electronics, especially with respect to the maximum rating, operating supply voltage range, movement power voltage range, heat radiation characteristics, installation and other product characteristics. Renesas Electronics shall have no liability for malfunctions or damages arising out of the use of Renesas Electronics products beyond such specified ranges.

9. Although Renesas Electronics endeavors to improve the quality and reliability of its products, semiconductor products have specific characteristics such as the occurrence of failure at a certain rate and malfunctions under certain use conditions. Further, Renesas Electronics products are not subject to radiation resistance design. Please be sure to implement safety measures to guard them against the possibility of physical injury, and injury or damage caused by fire in the event of the failure of a Renesas Electronics product, such as safety design for hardware and software including but not limited to redundancy, fire control and malfunction prevention, appropriate treatment for aging degradation or any other appropriate measures. Because the evaluation of microcomputer software alone is very difficult, please evaluate the safety of the final products or system manufactured by you.

10. Please contact a Renesas Electronics sales office for details as to environmental matters such as the environmental compatibility of each Renesas Electronics product. Please use Renesas Electronics products in compliance with all applicable laws and regulations that regulate the inclusion or use of controlled substances, including without limitation, the EU RoHS Directive. Renesas Electronics assumes no liability for damages or losses occurring as a result of your noncompliance with applicable laws and regulations.

11. This document may not be reproduced or duplicated, in any form, in whole or in part, without prior written consent of Renesas Electronics.

12. Please contact a Renesas Electronics sales office if you have any questions regarding the information contained in this document or Renesas Electronics products, or if you have any other inquiries.

(Note 1) “Renesas Electronics” as used in this document means Renesas Electronics Corporation and also includes its majority-owned subsidiaries.

(Note 2) “Renesas Electronics product(s)” means any product developed or manufactured by or for Renesas Electronics.
H8SX Family

Using the Average Transfer Rate Generator in Transmission and Reception via the SCI (Serial Communications Interface): DMAC Volume

Introduction

In the H8SX/1653, an average transfer rate generator can be selected as the clock source for serial communications interfaces 5 and 6 (SCI_5 and 6) in the asynchronous mode. In this sample task, the average transfer rate generator is used to drive the transfer of data at a rate of 921.569 kbps in operation at \( Pφ = 16 \) MHz.

Target Device

H8SX/1653

Contents

1. Specifications ............................................................................................................. 2
2. Applicable Conditions ................................................................................................. 3
3. Description of Modules Used ....................................................................................... 4
4. Description of Operation ............................................................................................. 10
5. Description of Software .............................................................................................. 14
1. Specifications

An average transfer rate generator can be selected as the clock source in the asynchronous mode on SCI_5 and 6 of the H8SX/1653. In this sample task, data are transmitted and received at a rate of 921.69 kbps by using the average transfer rate generator of SCI_5 with the peripheral clock (Pφ) running at 16 MHz.

- Connect the H8SX/1653 as shown in figure 1.
- Table 1 shows the communications format.
- After a power-on reset of the master side, pin P13 on the same side outputs a low-level trigger, and the master side starts operations for the simultaneous reception and transmission of 128-byte blocks of data.
- When the low-level trigger is input to the IRQ3 pin on the slave side, the slave side starts operations for the simultaneous transmission and reception of 128-byte blocks of data.
- In this sample task, interrupt-driven DMAC asynchronously handles transmission and reception of the 128-byte blocks.

<table>
<thead>
<tr>
<th>Item</th>
<th>Setting</th>
</tr>
</thead>
<tbody>
<tr>
<td>Pφ</td>
<td>16 MHz</td>
</tr>
<tr>
<td>Serial communications mode</td>
<td>Asynchronous</td>
</tr>
<tr>
<td>Clock source</td>
<td>Average transfer rate generator</td>
</tr>
<tr>
<td>Transfer rate</td>
<td>921.69 kbps</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>Format for serial/parallel conversion</td>
<td>LSB first</td>
</tr>
</tbody>
</table>

![Figure 1: Setup for Asynchronous Communications with Timing from the Average Transfer Rate Generator](image)
2. Applicable Conditions

Table 2  Applicable Conditions

<table>
<thead>
<tr>
<th>Item</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Operating frequency</td>
<td>Input clock: 16 MHz</td>
</tr>
<tr>
<td></td>
<td>System clock (Iφ): 16 MHz</td>
</tr>
<tr>
<td></td>
<td>Peripheral module clock (Pφ): 16 MHz</td>
</tr>
<tr>
<td></td>
<td>External bus clock (Bφ): 16 MHz</td>
</tr>
<tr>
<td>Operating mode</td>
<td>Mode 6 (MD2 = 1, MD1 = 1, MD0 = 0) MD_CLK = 0</td>
</tr>
<tr>
<td>Development tool</td>
<td>High-performance Embedded Workshop Ver. 4.00.02</td>
</tr>
<tr>
<td>C/C++ compiler</td>
<td>Renesas Technology Corp. H8S, H8/300 Series C/C++ Compiler Ver. 6.01.00</td>
</tr>
</tbody>
</table>
| Compiler options      | -cpu = h8sxa:24:md, -code = machinecode, -optimize = 1, -regparam = 3,  
|                       | -speed = (register, shift, struct, expression) |

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>Data table storage</td>
</tr>
<tr>
<td>H'FF7000</td>
<td>B</td>
<td>Non-initialized data area (RAM area)</td>
</tr>
</tbody>
</table>
3. Description of Modules Used

3.1 Description in Outline

Figure 2 shows the peripheral modules of the H8SX/1653 which are used in this sample task. The following description concerns the blocks shown in figure 2.

1. SCI_5
   The average transfer rate generator is used in data transmission and reception.
   a. During SCI Transmission
      • When TSR_5 is not full, data for transmission are written to TDR_5, transferred to TSR_5, and then output on the TxD5 pin.
      • When data are transferred from TDR_5 to TSR_5, a TXI_5 interrupt is generated.
   b. During SCI Reception
      • After one frame of data has been received via the RxD5 pin, the received data are transferred from RSR_5 to RDR_5.
      • Once the data have been successfully received and then transferred from RSR_5 to RDR_5, an RXI_5 interrupt is generated.

2. DMAC channels 0 and 1
   a. During SCI Transmission
      • Channel 0 is activated by TXI_5 (transmit data empty interrupt) of SCI_5 and transfers data from the area where data for transmission are stored to the TDR_5 register.
   b. During SCI Reception
      • Channel 1 is activated by RXI_5 (receive data full interrupt) of SCI_5 and transfers data from RDR_5 to the area where received data are stored.
Using the Average Transfer Rate Generator in Transmission and Reception via the SCI (Serial Communications Interface): DMAC Volume

Figure 2 Usage of H8SX/1653 Modules
3.2 SCI_5

In this sample task, SCI_5 is used for asynchronous serial data transmission. Figure 3 is a block diagram of SCI_5, and the following is a description of the functions in the diagram.

- **On-Chip Peripheral Clock Pφ**
  - This is the base clock for the operation of on-chip peripheral functions and is generated by a clock oscillator.

- **Receive Shift Register (RSR_5)**
  - This register is used to receive serial data. Serial data on RSR_5 are input via the RxD5 pin. When one frame of data has been received, the data bits are automatically transferred to the Receive Data Register (RDR_5). RSR_5 is not accessible by the CPU.

- **Receive Data Register (RDR_5)**
  - Received data are stored in this 8-bit register. After RSR_5 has received one frame, the data bits are automatically transferred from RSR_5 to RDR_5. Since RSR_5 and RDR_5 function as a double buffer, continuous reception is possible. RDR_5 is for reception only, and so is seen as a read-only register by the CPU.

- **Transmit Shift Register (TSR_5)**
  - This register is used to transmit serial data. In transmission, data are transferred from the Transmit Data Register (TDR_5) to TSR_5, and then output on the TxD5 pin. TSR_5 is not directly accessible from the CPU.

- **Transmit Data Register (TDR_5)**
  - Data for transmission are stored in this 8-bit register. When SCI_5 detects that TSR_5 is empty, data that have been written to TDR_5 are automatically transferred to TSR_5. Since TDR_5 and TSR_5 function as a double buffer, if the next byte is written to TDR_5 before transmission of the frame including the byte currently in TSR_5 is complete, the byte can be transferred to TSR_5 immediately on completion of the transmission. This allows continual transmission. Although TDR can be read from or written to by the CPU at all times, only write data for transmission data after having confirmed setting of the TDRE bit in the Serial Status Register (SSR_5) to 1.

- **Serial Mode Register (SMR_5)**
  - This 8-bit register is used to select the format of serial data communications and the clock source for the on-chip baud-rate generator.

- **Serial Control Register (SCR_5)**
  - This register is used to control transmission, reception, and interrupts, and to select the clock source for transmission and reception.

- **Serial Status Register (SSR_5)**
  - This register consists of status flags for SCI_5 and multiprocessor bits for transmission and reception. TDRE, RDRF, ORER, PER, and FER can only be cleared.

- **Smart Card Mode Register (SCMR_5)**
  - This register is used to select the smart-card or normal interface mode for SCMR_5, and to set up the format for the smart-card mode. For this task, the setting in SCMR_5 selects the normal asynchronous or clock synchronous mode.

- **Serial Extended Mode Register (SEMR_5)**
  - SEMR_5 and SEMR_6 are used to select the clock source for SCI_5 and SCI_6 in the asynchronous mode. The base (peripheral) clock is automatically specified when average transfer rate operation is selected. TMO output from timer units 2 and 3 can also be set as the base clock for serial transfer. Otherwise, specific average transfer rates are selectable according to whether the peripheral-clock frequency is 8, 10.667, 12, 16, 24, or 32 MHz. Table 4 shows the relationship between Pφ and average transfer rate.
Using the Average Transfer Rate Generator in Transmission and Reception via the SCI (Serial Communications Interface): DMAC Volume

Figure 3  SCI_5 Block Diagram

Table 4  Transfer Rates of Average Transfer Rate Generators on SCI_5 and 6

<table>
<thead>
<tr>
<th>( f_s ) (MHz)</th>
<th>Average Transfer Rate (kbps)</th>
</tr>
</thead>
<tbody>
<tr>
<td>8</td>
<td>460.784</td>
</tr>
<tr>
<td>10.667</td>
<td>115.152 460.606</td>
</tr>
<tr>
<td>12</td>
<td>230.263 460.526</td>
</tr>
<tr>
<td>16</td>
<td>115.196 460.784 720 921.569</td>
</tr>
<tr>
<td>24</td>
<td>115.132 460.526 720 921.053</td>
</tr>
<tr>
<td>32</td>
<td>720</td>
</tr>
</tbody>
</table>
3.3 DMAC Channels 0 and 1

In this sample task, DMAC channel 0 is activated by the TXI_5 interrupt of SCI_5 and DMAC channel 1 is activated by the RXI_5 interrupt of SCI_5. A block diagram of the DMAC is given as figure 4. The following description is with reference to figure 4.

- DMA source address register _0 (DSAR_0)
- DMA source address register _1 (DSAR_1)
  These registers are 32-bit readable/writable registers and specify the source address for the transfer. Each register is equipped with an address-updating function, so the source address is updated to that for the next transfer each time a transfer operation takes place.

- DMA destination address register _0 (DDAR_0)
- DMA destination address register _1 (DDAR_1)
  These registers are 32-bit readable/writable registers and specify the destination address for the transfer. Each register is equipped with an address-updating function, so the destination address is updated to that for the next transfer each time a transfer operation takes place.

- DMA transfer count register _0 (DTCR_0)
- DMA transfer count register _1 (DTCR_1)
  These registers are 32-bit readable/writable registers and specify the amount of data to be transferred (total size for transfer). After each data transfer operation, the value is reduced by the amount that corresponds to the transferred amount of data. In this sample task, both are set for 1536 bytes (H'00000600) of data, and the byte is selected as the unit of data access. Four is subtracted from the value on each DMAC operation, to indicate the amount still to be transferred.

- DMA mode control register _0 (DMDR_0)
- DMA mode control register _1 (DMDR_1)
  These registers control DMAC operation.

- DMA address control register_0 (DACR_0)
- DMA address control register_1 (DACR_1)
  These registers set the operating mode and transfer method.

- DMA module request select register_0 (DMRSR_0)
- DMA module request select register_1 (DMRSR_1)
  These registers set the activation source.
Internal activation sources

TXL_5

RXI_5

Internal activation source judgment

DMRSR_0,1

Control unit

Internal address bus

Address buffer

Arithmetic unit

DSAR_0,1

DDAR_0,1

DTCR_0,1

Module data bus

Figure 4 DMAC Block Diagram
4. Description of Operation

4.1 Outline

Figure 5 shows the operation of this task in outline. When 128 bytes of data is transmitted from the master side to the slave side, the same bytes of data is returned from the slave side to the master side.

---

**Figure 5  Outline of Operations**

---
4.2 Transmission

The timing of transmission operations is shown in figure 6. Table 5 is a list of the hardware and software processing at the numbered points in figure 6.

![Figure 6 Timing of Transmission](image-url)

**Table 5 Processing**

<table>
<thead>
<tr>
<th>Hardware Processing</th>
<th>Software Processing</th>
</tr>
</thead>
</table>
| (1) a. Set TDRE to 1.  
b. Activate DMAC_0, and transfer data for transmission from RAM to TDR_5.  
c. Clear TDRE to 0.  | None |
| (2) a. Decrement DTCR_0.  
b. Transfer the contents of TDR_5 to TSR_5.  
c. Output the contents of TSR_5 on the TxD5 pin.  | None |
| (3) a. Decrement DTCR_0 (producing DTCR_0 = 0).  
b. Transfer the contents of TDR_5 to TSR_5.  
c. Output the contents of TSR_5 on the TxD5 pin. | DMA transfer end interrupt processing  
a. Disable transmission and transmit end interrupts. |
| (4) a. Set TEND to 1.  | TEI interrupt processing  
a. Clear TE to 0.  
b. Disable TEI interrupts. |
4.3 Reception

The timing of reception operations is shown in figure 7. Table 6 is a list of the hardware and software processing at the numbered points in figure 7.

![Figure 7 Timing of Reception](image)

**Table 6 Processing**

<table>
<thead>
<tr>
<th>Hardware Processing</th>
<th>Software Processing</th>
</tr>
</thead>
<tbody>
<tr>
<td>(1) a. Set RDRF to 1.</td>
<td>None</td>
</tr>
<tr>
<td>b. Each time a byte is successfully received in RSR_5, transfer it to RDR_5.</td>
<td></td>
</tr>
<tr>
<td>c. Activate DMAC_1, and transfer data for transmission from RAM to RDR_5.</td>
<td></td>
</tr>
<tr>
<td>d. Clear RDRF5 to 0.</td>
<td></td>
</tr>
<tr>
<td>(2) a. Decrement DTCR_1.</td>
<td>None</td>
</tr>
<tr>
<td>(3) Decrement DTCR_1 (DTCR_1 = 0).</td>
<td>DMA transfer end interrupt processing</td>
</tr>
<tr>
<td>a. Disable reception and data-received interrupts.</td>
<td></td>
</tr>
</tbody>
</table>
4.4 Example of Internal Base Clock When $P_\phi = 16$ MHz and ACS3 to 0 = B’0011

Figure 8 shows an example of the internal base clock when $P_\phi = 16$ MHz and ACS3 to 0 = B’0011 = 16 MHz and ACS3 to 0 = B’0011 = 16 MHz and ACS3 to ACS0 = B’0011. When $P_\phi = 16$ MHz and ACS3 to 0 = B’0011, $P_\phi$ is divided by two and the cycle retention rate is 47/51. The following is the average transfer rate calculated from $P_\phi$ and the cycle retention rate.

Average transfer rate = \[
\frac{\text{Base clock frequency for average-rate transfer}}{8} \times \frac{P_\phi}{2} \times \frac{47}{51} = \frac{7.3725 \text{ MHz}}{8} = 921.569 \text{ kbps}
\]

4.5 One-Bit Period for Communications Data

The one-bit period corresponds to eight cycles of the internally generated base clock, but the actual period will differ according to whether or not a cycle of the frequency-divided peripheral clock has been omitted from the internal base clock. Therefore, the 1-bit interval is either of the following.

Higher period for one bit: \[
\frac{1}{P_\phi/2} \times 9 \text{ frequency-divided peripheral clock cycles} = 1.125 \mu s
\]

Lower period for one bit: \[
\frac{1}{P_\phi/2} \times 8 \text{ frequency-divided peripheral clock cycles} = 1.0 \mu s
\]
5. Description of Software

5.1 List of Functions

Functions used in this sample task are shown in table 7.

The hierarchical structure of this sample task is shown in figure 9.

Table 7 List of Functions

<table>
<thead>
<tr>
<th>Function Name</th>
<th>Description</th>
</tr>
</thead>
</table>
| Init              | Initialization routine  
Takes the module out of module stop mode, performs clock settings, and calls the main function |
| main              | Main routine  
Makes initial SCI settings for communications at the transfer rate of 921.569 kbps when operating at $P_\phi = 16$ MHz |
| DMAC0_trs_init    | DMAC_0 initialization  
Processing for transfer on TXI from ROM to TDR_5 |
| DMAC1_rcv_init    | DMAC_1 initialization  
Processing for transfer on RXI from RDR_5 to RAM |
| dmtend0_int       | DMAC_0 transfer end interrupt handler  
Disables SCI transmission and SCI transmission interrupts |
| dmtend1_int       | DMAC_1 transfer end interrupt handler  
Disables SCI reception and SCI reception interrupts |
| eri5_int          | Receive error interrupt handler  
In cases of error in reception, writes the contents of SSR_5 to RAM and then initializes SSR_5 |
| tei5_int          | Transmission end interrupt handler  
Disables TEI interrupt requests. Sets endflg to 1. |

Figure 9 Hierarchy Structure
### 5.2 Vector Table

**Table 8 Exception Handling Vector Table**

<table>
<thead>
<tr>
<th>Exception Handling Source</th>
<th>Vector Number</th>
<th>Address in Vector Table</th>
<th>Vector Table Address Handling Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>Reset</td>
<td>0</td>
<td>H'000000</td>
<td>Init</td>
</tr>
<tr>
<td>DMAC_0 DMTEND0</td>
<td>128</td>
<td>H'000200</td>
<td>dmtend0_int</td>
</tr>
<tr>
<td>DMAC_1 DMTEND1</td>
<td>129</td>
<td>H'000204</td>
<td>dmtend1_int</td>
</tr>
<tr>
<td>SCI_5 ERI5</td>
<td>222</td>
<td>H'000378</td>
<td>eri5_init</td>
</tr>
<tr>
<td>SCI_5 TEI5</td>
<td>223</td>
<td>H'00037C</td>
<td>tei5_init</td>
</tr>
</tbody>
</table>

### 5.3 RAM Usage

**Table 9 RAM Usage**

<table>
<thead>
<tr>
<th>Type</th>
<th>Variable Name</th>
<th>Contents</th>
<th>Used In</th>
</tr>
</thead>
<tbody>
<tr>
<td>unsigned char</td>
<td>endflg</td>
<td>Transmission end flag&lt;br&gt;0: Transmission in progress&lt;br&gt;1: Transmission ended</td>
<td>main, tei5_int</td>
</tr>
<tr>
<td>unsigned char</td>
<td>errbuf</td>
<td>Reception error buffer&lt;br&gt;The contents of SSR_5 are stored here when an overrun error, flaming error, or parity error occurs.</td>
<td>main, eri5_int</td>
</tr>
<tr>
<td>unsigned char</td>
<td>tcnt</td>
<td>Transmission counter</td>
<td>main, txi5_int</td>
</tr>
<tr>
<td>unsigned char</td>
<td>rcnt</td>
<td>Reception counter</td>
<td>main, rxi5_int</td>
</tr>
<tr>
<td>unsigned char</td>
<td>rcv_dt[128]</td>
<td>RAM area for storing received data</td>
<td>main, rxi5_int</td>
</tr>
</tbody>
</table>

### 5.4 Data Table

**Table 10 Data Table**

<table>
<thead>
<tr>
<th>Type</th>
<th>Variable Name</th>
<th>Contents</th>
<th>Used In</th>
</tr>
</thead>
<tbody>
<tr>
<td>unsigned char</td>
<td>trs_dt[128]</td>
<td>ROM area where data for transmission are stored.&lt;br&gt;128 bytes of data: H'00, H'01, ..., H'7F</td>
<td>main, txi5_int</td>
</tr>
</tbody>
</table>

### 5.5 Defined Macros

**Table 11 Defined Macros**

<table>
<thead>
<tr>
<th>Identifier</th>
<th>Contents</th>
<th>Used In</th>
</tr>
</thead>
<tbody>
<tr>
<td>MASTER</td>
<td>If this is defined, compilation generates the master-side program.</td>
<td>main</td>
</tr>
<tr>
<td>SLAVE</td>
<td>If this is defined, compilation generates the slave-side program.</td>
<td>main</td>
</tr>
</tbody>
</table>
5.6 Description of Functions

5.6.1 init Function

1. Overview
   Initialization routine. Takes the module out of module stopped mode, sets the clock, and calls the main function.

2. Arguments
   None

3. Return value
   None

4. Description of internal register usage
   Usage of internal registers in this task is described below. The given settings are those used in the task and differ from the initial settings.

- **System Clock Control Register (SCKCR)**
  Address: H'FFDC4

<table>
<thead>
<tr>
<th>Bit</th>
<th>Bit Name</th>
<th>Setting</th>
<th>R/W</th>
<th>Description</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>1</td>
<td>R/W</td>
<td>Selects the frequency of the CPU, DMAC, DTC module and system clock</td>
</tr>
<tr>
<td>8</td>
<td>ICK0</td>
<td>0</td>
<td>R/W</td>
<td>010: Input clock x 1</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>1</td>
<td>R/W</td>
<td>Selects the frequency of peripheral module clock</td>
</tr>
<tr>
<td>4</td>
<td>PCK0</td>
<td>0</td>
<td>R/W</td>
<td>010: Input clock x 1</td>
</tr>
<tr>
<td>2</td>
<td>BCK2</td>
<td>0</td>
<td>R/W</td>
<td>External bus clock (Bφ) selection</td>
</tr>
<tr>
<td>1</td>
<td>BCK1</td>
<td>1</td>
<td>R/W</td>
<td>Selects the frequency of the external bus clock</td>
</tr>
<tr>
<td>0</td>
<td>BCK0</td>
<td>0</td>
<td>R/W</td>
<td>010: Input clock x 1</td>
</tr>
</tbody>
</table>

- Registers MSTPCRA, MSTPCRB, and MSTPCRC control the module stop mode. Setting a bit to 1 makes the corresponding module enter the module stop mode, while clearing the bit to 0 takes the module out of stop mode.

- **Module Stop Control Register A (MSTPCRA)**
  Address: H'FFDC8

<table>
<thead>
<tr>
<th>Bit</th>
<th>Bit Name</th>
<th>Setting</th>
<th>R/W</th>
<th>Description</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>
<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>9</td>
<td>MSTPA9</td>
<td>1</td>
<td>R/W</td>
<td>8-bit timers (TMR_3, TMR_2)</td>
</tr>
<tr>
<td>8</td>
<td>MSTPA8</td>
<td>1</td>
<td>R/W</td>
<td>8-bit timers (TMR_1, TMR_0)</td>
</tr>
<tr>
<td>5</td>
<td>MSTPA5</td>
<td>1</td>
<td>R/W</td>
<td>D/A converter channels 1, 0</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>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) Address: H'FFDCA

<table>
<thead>
<tr>
<th>Bit</th>
<th>Bit Name</th>
<th>Setting</th>
<th>R/W</th>
<th>Description</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>1</td>
<td>R/W</td>
<td>Serial Communications Interface_4 (SCI_4)</td>
</tr>
<tr>
<td>10</td>
<td>MSTPB10</td>
<td>1</td>
<td>R/W</td>
<td>Serial Communications Interface_2 (SCI_2)</td>
</tr>
<tr>
<td>9</td>
<td>MSTPB9</td>
<td>1</td>
<td>R/W</td>
<td>Serial Communications Interface_1 (SCI_1)</td>
</tr>
<tr>
<td>8</td>
<td>MSTPB8</td>
<td>1</td>
<td>R/W</td>
<td>Serial Communications Interface_0 (SCI_0)</td>
</tr>
<tr>
<td>7</td>
<td>MSTPB7</td>
<td>1</td>
<td>R/W</td>
<td>I^C Bus Interface_1 (IIC_1)</td>
</tr>
<tr>
<td>6</td>
<td>MSTPB6</td>
<td>1</td>
<td>R/W</td>
<td>I^C Bus Interface_0 (IIC_0)</td>
</tr>
</tbody>
</table>

• Module Stop Control Register C (MSTPCRC) Address: H'FFDCC

<table>
<thead>
<tr>
<th>Bit</th>
<th>Bit Name</th>
<th>Setting</th>
<th>R/W</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>15</td>
<td>MSTPC15</td>
<td>0</td>
<td>R/W</td>
<td>Serial communications interface_5 (SCI_5) and (IrDA)</td>
</tr>
<tr>
<td>14</td>
<td>MSTPC14</td>
<td>1</td>
<td>R/W</td>
<td>Serial communications interface_6 (SCI_6)</td>
</tr>
<tr>
<td>13</td>
<td>MSTPC13</td>
<td>1</td>
<td>R/W</td>
<td>8-bit timers (TMR_4 and TMR_5)</td>
</tr>
<tr>
<td>12</td>
<td>MSTPC12</td>
<td>1</td>
<td>R/W</td>
<td>8-bit timers (TMR_6 and TMR_7)</td>
</tr>
<tr>
<td>11</td>
<td>MSTPC11</td>
<td>1</td>
<td>R/W</td>
<td>Universal Serial Bus interface (USB)</td>
</tr>
<tr>
<td>10</td>
<td>MSTPC10</td>
<td>1</td>
<td>R/W</td>
<td>Cyclic redundancy checker</td>
</tr>
<tr>
<td>4</td>
<td>MSTPC4</td>
<td>0</td>
<td>R/W</td>
<td>On-chip RAM_4 (H'FF2000 to H'FF3FFF)</td>
</tr>
<tr>
<td>3</td>
<td>MSTPC3</td>
<td>0</td>
<td>R/W</td>
<td>On-chip RAM_3 (H'FF4000 to H'FF5FFF)</td>
</tr>
<tr>
<td>2</td>
<td>MSTPC2</td>
<td>0</td>
<td>R/W</td>
<td>On-chip RAM_2 (H'FF6000 to H'FF7FFF)</td>
</tr>
<tr>
<td>1</td>
<td>MSTPC1</td>
<td>0</td>
<td>R/W</td>
<td>On-chip RAM_1 (H'FF8000 to H'FF9FFF)</td>
</tr>
<tr>
<td>0</td>
<td>MSTPC0</td>
<td>0</td>
<td>R/W</td>
<td>On-chip RAM_0 (H'FFA000 to H'FFBFFF)</td>
</tr>
</tbody>
</table>

5. Flowchart

```
init

CCR = H'80
Initialize the CCR by masking interrupts

SCKCR = H'0222
Clock (x1,x1,x1)

MSTPCRA = H'1FFF
MSTPCRB = H'FFFF
MSTPCRC = H'7F00
Release modules from the module stop mode

main()
Call the main routine
```
### 5.6.2 main Function

1. **Overview**  
   Main routine. Sets the average transfer rate in SCI, makes DMAC0_trs_init function and DMAC1_rcv_init function calls, transmits and receives a total of 256 bytes of data.

2. **Arguments**  
   None

3. **Return value**  
   None

4. **Description of internal register usage**
   Usage of internal registers in this task is described below. The given settings are those used in the task and differ from the initial settings.

   **• Port 1 Data Direction Register (P1DDR)**  
   Address: H'FFFB80

<table>
<thead>
<tr>
<th>Bit</th>
<th>Bit Name</th>
<th>Setting</th>
<th>R/W</th>
<th>Description</th>
</tr>
</thead>
</table>
   | 3   | P13DDR   | 1       | W   | 0: P13 pin is an input  
   |     |          |         |     | 1: P13 pin is an output |

   **• Port 1 Input Buffer Control Register (P1ICR)**  
   Address: H'FFFB90

<table>
<thead>
<tr>
<th>Bit</th>
<th>Bit Name</th>
<th>Setting</th>
<th>R/W</th>
<th>Description</th>
</tr>
</thead>
</table>
   | 5   | P15ICR   | 1       | R/W | 0: P15 pin input buffer is disabled. Input signal is fixed to the high level.  
   |     |          |         |     | 1: P15 pin input buffer is valid. The pin state reflects the peripheral modules. |
   | 3   | P13ICR   | 1       | R/W | 0: P13 pin input buffer is disabled. Input signal is fixed to the high level.  
   |     |          |         |     | 1: P13 pin input buffer is valid. The pin state reflects the peripheral modules. |

   **• Port Function Control Register (PFCRC)**  
   Address: H'FFFBCC

<table>
<thead>
<tr>
<th>Bit</th>
<th>Bit Name</th>
<th>Setting</th>
<th>R/W</th>
<th>Description</th>
</tr>
</thead>
</table>
   | 3   | ITS3     | 1       | R/W | IRQ3 Pin Select  
   |     |          |         |     | 0: Selects IRQ3-A input on pin P13  
   |     |          |         |     | 1: Selects IRQ3-B input on pin P53 |

   **• IRQ Sense Control Register L (ISCRL)**  
   Address: H'FFFD6A

<table>
<thead>
<tr>
<th>Bit</th>
<th>Bit Name</th>
<th>Setting</th>
<th>R/W</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>7</td>
<td>IRQ3SR</td>
<td>0</td>
<td>R/W</td>
<td>IRQ3 Sense Control Rise</td>
</tr>
</tbody>
</table>
   | 6   | IRQ3SF   | 1       | R/W | IRQ3 Sense Control Fall  
   |     |          |         |     | 01: Interrupt requests are sensed on falling edges of IRQ3 input |
### Serial Mode Register_5 (SMR_5) Address: H'FFF600

<table>
<thead>
<tr>
<th>Bit</th>
<th>Bit Name</th>
<th>Setting</th>
<th>R/W</th>
<th>Description</th>
</tr>
</thead>
</table>
| 7   | C/A      | 0       | R/W | Communications Mode  
0: Asynchronous mode  
1: Clock synchronous mode |
| 6   | CHR      | 0       | R/W | Character Length  
0: Selects 8 bits as the data length  
1: Selects 7 bits as the data length |
| 5   | PE       | 0       | R/W | Parity Enable  
0: No parity  
1: With parity |
| 3   | STOP     | 0       | R/W | Stop Bit Length  
Selects the length of the stop bit during transmission  
0: 1 stop bit  
1: 0 stop bit  
During reception, only the first bit of the stop bits is checked, and when the second bit is 0, it is regarded as the start bit of the next transmit frame. |

### Serial Control Register_5 (SCR_5) Address: H'FFFF602

<table>
<thead>
<tr>
<th>Bit</th>
<th>Bit Name</th>
<th>Setting</th>
<th>R/W</th>
<th>Description</th>
</tr>
</thead>
</table>
| 7   | TIE      | 0       | R/W | Transmit Interrupt Enable  
0: Disables TXI interrupts  
1: Enables TXI interrupts |
| 6   | RIE      | 0       | R/W | Receive Interrupt Enable  
0: Disables RXI, ERI interrupts  
1: Enables RXI, ERI interrupts |
| 5   | TE       | 0       | R/W | Transmit Enable  
0: Disables transmission  
1: Enables transmission |
| 4   | RE       | 0       | R/W | Receive Enable  
0: Disables reception  
1: Enables reception |
| 2   | TEIE     | 0       | R/W | Transmit End Interrupt Enable  
0: Disables TEI interrupts  
1: Enables TEI interrupts |
| 1   | CKE1     | 1       | R/W | Clock Enable 1, 0 |
| 0   | CKE0     | X       | R/W | Select the clock source.  
00: Internal baud rate generator  
1X: TMR clock input or average transfer rate generator |

**Legend**

X: Don't care.
### Serial Status Register_5 (SSR_5) Address: H'FFF604

<table>
<thead>
<tr>
<th>Bit</th>
<th>Bit Name</th>
<th>Setting</th>
<th>R/W</th>
<th>Description</th>
</tr>
</thead>
</table>
| 7   | TDRE     | Not fixed | R/(W)* | Transmit Data Register Empty  
Indicates whether TDR contains data for transmission  
[Setting conditions]  
• Clearing of the TE bit in SCR (to 0)  
• Transfer of data from TDR to TSR  
[Clearing conditions]  
• Writing a 0 to TDRE after having read TDRE = 1  
• Generation of a TXI interrupt request allowing DMAC to write data to TDR |
| 6   | RDRF     | 0        | R/(W)* | Receive Data Register Full  
Indicates whether RDR holds received data  
[Setting condition]  
• The normal end of serial reception and the transfer of received data from RSR to RDR  
[Clearing conditions]  
• Writing of 0 to RDRF after having read RDRF = 1  
• Generation of an RXI interrupt request allowing DMAC or DTC to read data from RDR  
The RDRF flag is not affected and retains its previous value when the RE bit in SCR is cleared to 0.  
Note that when the next serial reception is completed while the RDRF flag is being set to 1, an overrun error occurs and the received data are lost. |
| 5   | ORER     | 0        | R/(W)* | Overrun Error  
[Setting condition]  
• Occurrence of an overrun error during reception  
[Clearing condition]  
• Writing of 0 to ORER after having read ORER = 1 |
| 4   | FER      | 0        | R/(W)* | Framing Error  
[Setting condition]  
• Occurrence of a framing error during reception  
[Clearing condition]  
• Writing of 0 to FER after having read FER = 1 |
| 3   | PER      | 0        | R/(W)* | Parity Error  
[Setting condition]  
• Occurrence of a parity error during reception  
[Clearing condition]  
• Writing of 0 to PER after having read PER = 1 |
| 2   | TEND     | Not fixed | R    | Transmit End  
[Setting condition]  
• Clearing of the TE bit in SCR to 0  
• TDRE = 1 on transmission of the last bit of a character  
[Clearing conditions]  
• Writing of 0 to TDRE after having read TDRE = 1  
• Generation of a TXI interrupt request allowing DMAC to write data to TDR |

Note: * Only 0 can be written here, to clear the flag.
Using the Average Transfer Rate Generator in Transmission and Reception via the SCI (Serial Communications Interface): DMAC Volume

- **Smart Card Mode Register_5 (SCMR_5)**  
  Address: H'FFFF606

<table>
<thead>
<tr>
<th>Bit</th>
<th>Bit Name</th>
<th>Setting</th>
<th>R/W</th>
<th>Description</th>
</tr>
</thead>
</table>
| 0   | SMIF     | 0       | R/W | Smart Card Interface Mode Select  
  0: Operation is in the normal asynchronous or clock synchronous mode  
  1: Operation is in the smart card interface mode |

- **Serial Expansion Mode Register_5 (SEMR_5)**  
  Address: H'FFFF608

<table>
<thead>
<tr>
<th>Bit</th>
<th>Bit Name</th>
<th>Setting</th>
<th>R/W</th>
<th>Description</th>
</tr>
</thead>
</table>
| 4   | ABCS     | 0       | R/W | Asynchronous Mode Basic Clock Select (valid only in the asynchronous mode)  
  Selects the base clock for generating 1-bit periods (the transfer rate).  
  0: The transfer rate is 1/16 of the frequency of the base clock.  
  1: The transfer rate is 1/8 of the frequency for average-rate transfer. |
| 3   | ACS3     | 0       | R/W | Asynchronous Clock Source Select  
  Selects the clock source in the asynchronous mode. See table 12.  
  0011: Selects average transfer rate of 921.569 kbps specifically for P_0 = 16 MHz. |
| 2   | ACS2     | 0       | R/W |  
| 1   | ACS1     | 1       | R/W |  
| 0   | ACS0     | 1       | R/W | 0011: Selects average transfer rate of 921.569 kbps specifically for P_0 = 16 MHz.  
  Note 1: When the average transfer rate is selected, the base clock is automatically set regardless of the ABCS bit in the SEMR_5 register (asynchronous base clock selection).  
  Note 2: The setting has the desired effect only when bits ACS3 to ACS0 are in the asynchronous mode (C/Â bit of SMR register is 0), and the external clock input is selected (CKE1 bit of SCR register is 1). |
### Table 12 List of Setting for Asynchronous Clock Source Select

<table>
<thead>
<tr>
<th>ACS3 to 0</th>
<th>Transfer Rate</th>
<th>Pφ (MHz)</th>
<th>Functions</th>
</tr>
</thead>
<tbody>
<tr>
<td>0000</td>
<td>(Set by the ABCS bit)</td>
<td>-</td>
<td>The average rate transfer generator is not used.</td>
</tr>
<tr>
<td>0001</td>
<td>1/16&lt;sup&gt;th&lt;/sup&gt; of the base clock frequency for average-rate transfer</td>
<td>10.667</td>
<td>Average transfer rate 115.152 kbps</td>
</tr>
<tr>
<td>0010</td>
<td>1/8&lt;sup&gt;th&lt;/sup&gt; of the base clock frequency for average-rate transfer</td>
<td>10.667</td>
<td>Average transfer rate 460.606 kbps</td>
</tr>
<tr>
<td>0011</td>
<td>1/8&lt;sup&gt;th&lt;/sup&gt; of the base clock frequency for average-rate transfer</td>
<td>16</td>
<td>Average transfer rate 921.569 kbps</td>
</tr>
<tr>
<td></td>
<td></td>
<td>8</td>
<td>Average transfer rate 460.784 kbps</td>
</tr>
<tr>
<td>0100</td>
<td>(Set by the ABCS bits)</td>
<td>-</td>
<td>Selects TMR-clock input: compare-match output of TMR provides the base clock for transfer</td>
</tr>
<tr>
<td>0101</td>
<td>1/16&lt;sup&gt;th&lt;/sup&gt; of the base clock frequency for average-rate transfer</td>
<td>16</td>
<td>Average transfer rate 115.196 kbps</td>
</tr>
<tr>
<td>0110</td>
<td>1/16&lt;sup&gt;th&lt;/sup&gt; of the base clock frequency for average-rate transfer</td>
<td>16</td>
<td>Average transfer rate 460.784 kbps</td>
</tr>
<tr>
<td>0111</td>
<td>1/8&lt;sup&gt;th&lt;/sup&gt; of the base clock frequency for average-rate transfer</td>
<td>24</td>
<td>Average transfer rate 720 kbps</td>
</tr>
<tr>
<td>1000</td>
<td>1/16&lt;sup&gt;th&lt;/sup&gt; of the base clock frequency for average-rate transfer</td>
<td>24</td>
<td>Average transfer rate 115.132 kbps</td>
</tr>
<tr>
<td>1001</td>
<td>1/16&lt;sup&gt;th&lt;/sup&gt; of the base clock frequency for average-rate transfer</td>
<td>24</td>
<td>Average transfer rate 460.526 kbps</td>
</tr>
<tr>
<td></td>
<td></td>
<td>12</td>
<td>Average transfer rate 230.263 kbps</td>
</tr>
<tr>
<td>1010</td>
<td>1/8&lt;sup&gt;th&lt;/sup&gt; of the base clock frequency for average-rate transfer</td>
<td>24</td>
<td>Average transfer rate 720 kbps</td>
</tr>
<tr>
<td>1011</td>
<td>1/8&lt;sup&gt;th&lt;/sup&gt; of the base clock frequency for average-rate transfer</td>
<td>24</td>
<td>Average transfer rate 921.053 kbps</td>
</tr>
<tr>
<td></td>
<td></td>
<td>12</td>
<td>Average transfer rate 460.526 kbps</td>
</tr>
<tr>
<td>1100</td>
<td>1/16&lt;sup&gt;th&lt;/sup&gt; of the base clock frequency for average-rate transfer</td>
<td>32</td>
<td>Average transfer rate 720 kbps</td>
</tr>
</tbody>
</table>
5. Flowchart

```
main

i = 0

i < 128?

No

Yes

rcv_dt[i] = 0

i++

i < 128?

Yes

DMAC0_trs_init()
Initial setting of DMAC_0

DMAC1_rcv_init()
Initial setting of DMAC_1

Clear flags
endflg = 0
errbuf = 0
rcnt = 0
tcnt = 0

Clear TE and RE of SCR_5 to 0

Bit 5 of P1ICR = 1
Set pin P15 (RxD5) as an input

CKE1,0 of SCR_5 = B'1X
External clock input

SMR_5 = H'00
SCMR_5 &= H'F2
Communications format settings
  - Asynchronous mode
  - Data length: 8 bits
  - No parity
  - 1 stop bit

SEMR_5 = H'03
When \( P_{\phi} = 16 \text{ MHz} \), operations are at 921.569 kbps

i = 0

i++

Wait for 1-bit period

i < 20?

No

Yes

SSR_5 = H'87
Clear RDRF, ORER, PER, and FER of SSR_5 to 0

1
```
H8SX Family
Using the Average Transfer Rate Generator in Transmission and Reception via the SCI (Serial Communications Interface): DMAC Volume

1

Master side (MASTER)

P1DDR = H'08
P13 = 0
P13 = 1
Set pin P13 as an output and then output the low-level trigger on P13

Slave side (SLAVE)

P13CR = 1
PFRC3 = 0
IRQ3SC = 1
IRQ3 interrupt occurs on the falling edge of the IRQ3A signal

IRQ3F == 0?

Yes

I = 0
Enable interrupts

No

tmp = IRQ3F
IRQ3F = 0
Clear the IRQ3 flag

SCR_4 I = H'F4
Set TE, TIE, TEIE of SCR_4 to 1
Enable transmission, TIE and TEIE interrupts

DTE of DMAC_0 = 1
Enable transmission on DMAC_0

DTE of DMAC_1 = 1
Enable transmission on DMAC_1

endflg == 0?

Yes

No

tcnt < 128?

Yes

No

rcnt < 128?

Yes

No

End
5.6.3 DMAC0_trs_init Function

1. Overview
   DMAC_0 initial settings.

2. Arguments
   None

3. Return value
   None

4. Description of internal register usage
   Usage of internal registers in this task is described below. The given settings are those used in the task and differ from the initial settings.

   - **DMA source address register_0 (DSAR_0) Address: H'FFFFC00**
     Function: Specifies the source address for the transfer.
     Setting: &trs_dt

   - **DMA destination address register_0 (DDAR_0) Address: H'FFFFC04**
     Function: Specifies the destination address for the transfer.
     Setting: &TDR_5

   - **DMA transfer count register_0 (DTCR_0) Address: H'FFFFC0C**
     Function: Selects the amount of data to be transferred as 128 bytes.
     Setting: 128
### DMA mode control register_0 (DMDR_0) Address: H’FFFC14

<table>
<thead>
<tr>
<th>Bit</th>
<th>Bit Name</th>
<th>Setting</th>
<th>R/W</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>31</td>
<td>DTE</td>
<td>0</td>
<td>R/W</td>
<td>Data Transfer Enable</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0: Disables data transfer</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>1: Enables data transfer</td>
</tr>
<tr>
<td>26</td>
<td>NRD</td>
<td>0</td>
<td>R/W</td>
<td>Next Request Delay</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0: Starts accepting the next transfer request after</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>completion of the current transfer</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>1: Starts accepting the next transfer request one</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>cycle after completion of the current transfer.</td>
</tr>
<tr>
<td>17</td>
<td>ESIF</td>
<td>0</td>
<td>R/(W)*</td>
<td>Transfer Escape Interrupt Flag</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0: A transfer escape end interrupt has not been</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>requested.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>1: A transfer escape end interrupt has been requested.</td>
</tr>
<tr>
<td>16</td>
<td>DTIF</td>
<td>0</td>
<td>R/(W)*</td>
<td>Data Transfer Interrupt Flag</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0: A transfer end interrupt by the transfer counter</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>has not been requested.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>1: A transfer end interrupt by the transfer counter</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>has been requested.</td>
</tr>
<tr>
<td>15</td>
<td>DTSZ1</td>
<td>0</td>
<td>R/W</td>
<td>Data Access Size 1,0</td>
</tr>
<tr>
<td>14</td>
<td>DTSZ0</td>
<td>0</td>
<td>R/W</td>
<td>01: Data access size for transfer is bytes (8 bits)</td>
</tr>
<tr>
<td>13</td>
<td>MDS1</td>
<td>0</td>
<td>R/W</td>
<td>Transfer Mode Select 1, 0</td>
</tr>
<tr>
<td>12</td>
<td>MDS0</td>
<td>0</td>
<td>R/W</td>
<td>00: Normal transfer mode setting</td>
</tr>
<tr>
<td>9</td>
<td>ESIE</td>
<td>0</td>
<td>R/W</td>
<td>Transfer Escape Interrupt Enable</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0: Disables transfer escape interrupts.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>1: Enables transfer escape interrupts.</td>
</tr>
<tr>
<td>8</td>
<td>DTIE</td>
<td>0</td>
<td>R/W</td>
<td>Data Transfer End Interrupt Enable</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0: Disables transfer end interrupts</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>1: Enables transfer end interrupts</td>
</tr>
<tr>
<td>7</td>
<td>DTF1</td>
<td>1</td>
<td>R/W</td>
<td>Data Transfer Factor 1, 0</td>
</tr>
<tr>
<td>6</td>
<td>DTF0</td>
<td>0</td>
<td>R/W</td>
<td>10: DMAC activation source is an on-chip module</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>interrupt.</td>
</tr>
<tr>
<td>5</td>
<td>DTA</td>
<td>1</td>
<td>R/W</td>
<td>Data Transfer Acknowledge</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>When DTF1, 0 = H’10, the DTA bit is set to 1.</td>
</tr>
</tbody>
</table>

**Note**: Only 0 can be written here after having been read as 1, to clear the flag.

### DMA address control register_0 (DACR_0) Address: H’FFFC18

<table>
<thead>
<tr>
<th>Bit</th>
<th>Bit Name</th>
<th>Setting</th>
<th>R/W</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>31</td>
<td>AMS</td>
<td>0</td>
<td>R/W</td>
<td>Address Mode Select</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0: Dual address mode</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>1: Single address mode</td>
</tr>
<tr>
<td>21</td>
<td>SAT1</td>
<td>1</td>
<td>R/W</td>
<td>Source Address Update Mode 1, 0</td>
</tr>
<tr>
<td>20</td>
<td>SAT0</td>
<td>0</td>
<td>R/W</td>
<td>01: Increment the source address</td>
</tr>
<tr>
<td>17</td>
<td>DAT1</td>
<td>0</td>
<td>R/W</td>
<td>Destination Address Update Mode 1, 0</td>
</tr>
<tr>
<td>16</td>
<td>DAT0</td>
<td>0</td>
<td>R/W</td>
<td>01: Destination address is fixed</td>
</tr>
</tbody>
</table>

### DMA module request select register_0 (DMRSR_0) Address: H’FFFD20

Function: Specifies the source of on-chip module interrupts. The setting 221 corresponds to DMAC activation by SCI_5 transmission data empty interrupts.

Setting: 221
5. Flowchart

DMAC0_trs_init

tmp = DMDR_0
DMDR_0 = H'000001A0
-Enable transfer end interrupt requests
-DMAC activation source:
  On-chip module interrupt
-Enable clearing of the interrupt source flag
  selected by DMARSR

DSAR_0 = &trs_dt
Set the transfer data table start address as
the source address

DDAR_0 = &TDR_5
Set the address of TDR_5 as destination address

DTCR_0 = 128
Total amount for transfer: 128 bytes

DACR_0 = H'00200000
DSAR increase/decrease setting: Increase
DDAR increase/decrease setting: Fixed

DMRSR_0 = 221
DMAC activation source: SCI_5 transmit data
empty interrupts

End
5.6.4 DMAC1_rcv_init Function

1. Overview
   DMAC_1 initialization. Sets up the registers of DMAC channel for the transfer of received data from SCI_5.

2. Arguments
   None

3. Return value
   None

4. Description of internal register usage
   Usage of internal registers in this task is described below. The given settings are those used in the task and differ from the initial settings.

   • DMA source address register_1 (DSAR_1)  Address: H'FFFC20
     Function: Specifies the source address for the transfer.
     Setting: &RDR_5

   • DMA destination address register_1 (DDAR_1)  Address: H'FFFC24
     Function: Specifies the destination address for the transfer.
     Setting: &rcv_dt

   • DMA transfer count register_1 (DTCR_1)  Address: H'FFFC2C
     Function: Selects the amount of data to be transferred as 128 bytes.
     Setting: 128
### DMA mode control register_0 (DMDR_0)

<table>
<thead>
<tr>
<th>Bit</th>
<th>Bit Name</th>
<th>Setting</th>
<th>R/W</th>
<th>Description</th>
</tr>
</thead>
</table>
| 31  | DTE      | 0       | R/W | Data Transfer Enable  
0: Disables data transfer  
1: Enables data transfer |
| 26  | NRD      | 0       | R/W | Next Request Delay  
0: Starts accepting the next transfer request after completion of the current transfer  
1: Starts accepting the next transfer request one cycle after completion of the current transfer |
| 17  | ESIF     | 0       | R/(W)* | Transfer Escape Interrupt Flag  
0: A transfer escape end interrupt has not been requested.  
1: A transfer escape end interrupt has been requested. |
| 16  | DTIF     | 0       | R/(W)* | Data Transfer Interrupt Flag  
0: A transfer end interrupt by the transfer counter has not been requested.  
1: A transfer end interrupt by the transfer counter has been requested. |
| 15  | DTSZ1    | 0       | R/W | Data Access Size 1, 0  
00: Data access size for transfer is bytes (8 bits) |
| 14  | DTSZ0    | 0       | R/W | Transfer Mode Select 1, 0  
00: Normal transfer mode setting |
| 13  | MDS1     | 0       | R/W | Transfer Mode Select 1, 0  
00: Normal transfer mode setting |
| 12  | MDS0     | 0       | R/W | Transfer Mode Select 1, 0  
00: Normal transfer mode setting |
| 9   | ESIE     | 0       | R/W | Transfer Escape Interrupt Enable  
0: Disables transfer escape end interrupts  
1: Enables transfer escape end interrupts |
| 8   | DTIE     | 0       | R/W | Data Transfer Interrupt Enable  
0: Disables transfer end interrupts  
1: Enables transfer end interrupts |
| 7   | DTF1     | 1       | R/W | Data Transfer Factor 1, 0  
10: DMAC activation source is an on-chip module interrupt |
| 6   | DTF0     | 0       | R/W | Data Transfer Factor 1, 0  
10: DMAC activation source is an on-chip module interrupt |
| 5   | DTA      | 1       | R/W | Data Transfer Acknowledge  
When DTF1, 0 = H'10, the DTA bit is set to 1. |

Note: * Only 0 can be written here after having been read as 1, to clear the flag.

### DMA address control register_0 (DACR_0)

<table>
<thead>
<tr>
<th>Bit</th>
<th>Bit Name</th>
<th>Setting</th>
<th>R/W</th>
<th>Description</th>
</tr>
</thead>
</table>
| 31  | AMS      | 0       | R/W | Address Mode Selection  
0: Dual address mode  
1: Single address mode |
| 21  | SAT1     | 0       | R/W | Source Address Update Mode 1, 0 |
| 20  | SAT0     | 0       | R/W | 00: Source address is fixed. |
| 17  | DAT1     | 1       | R/W | Destination Address Update Mode 1, 0 |
| 16  | DAT0     | 0       | R/W | 10: Destination address is updated with an offset. |

### DMA module request select register_1 (DMRSR_1)

Function: Specifies the source of on-chip module interrupts. The setting 220 corresponds to DMAC activation by SCI_5 received data full interrupts.

Setting: 220
5. Flowchart

DMAC1_rcv_init

tmp = DMDR_1
DMDR_ = H'000001A0
- Enable transfer end interrupt requests
- DMAC activation source:
  On-chip module interrupt
- Enable clearing of the interrupt source flag selected by DMARSR

DSAR_1 = &RDR_5
Set the start address of RDR_5 register as the source address

DDAR_1 = &rcv_dt
Set the receive-data array start address as destination address

DTCR_1 = 128
Total amount for transfer: 128 bytes

DACR_1 = H'00200000
DSAR increase/decrease setting: Increase
DDAT increase/decrease setting: Fixed

DMRSR_1 = 220
DMAC activation source:
SCI_5 receive full interrupts

End
5.6.5  **dmtend0_init Function**

1. **Overview**
   Handler for the DMAC_0 transfer end interrupt. Stop the SCI transmission processing.

2. **Arguments**
   None

3. **Return value**
   None

4. **Description of internal register usage**
   Usage of internal registers in this task is described below. The given settings are those used in the task and differ from the initial settings.

   - **Serial Control Register_5 (SCR_5)**  Address: H'FFF602
     | Bit | Bit Name | Setting | R/W | Description                  |
     |-----|----------|---------|-----|------------------------------|
     | 7   | TIE      | 0       | R/W | Transmit Interrupt Enable   |
     |     |          |         |     | 0: Disables TXI interrupt requests |
     |     |          |         |     | 1: Enables TXI interrupt requests |

   - **DMA Mode Control Register_0 (DMDR_0)**  Address: H'FFFC14
     | Bit | Bit Name | Setting | R/W | Description                  |
     |-----|----------|---------|-----|------------------------------|
     | 8   | DTIE     | 0       | R/W | Data Transfer Interrupt Enable|
     |     |          |         |     | 0: Disables transfer end interrupt requests |
     |     |          |         |     | 1: Enables transfer end interrupt requests |

5. **Flowchart**

```
  dmtend0_init

  TIE of SCR_5 = 0
  Disable transmit end interrupts

  DTIE of DMDR_0 = 1
  Disable DMAC0 transfer end interrupts

  tcnt = 128
  Set 128 in transmission counter

  End
```
5.6.6  dmtend1_int Function

1. Overview
   Handler for the DMAC_1 transfer end interrupt. Stop the SCI reception processing.

2. Arguments
   None

3. Return value
   None

4. Description of internal register usage
   Usage of internal registers in this task is described below. The given settings are those used in the task and differ from the initial settings.

   • Serial Control Register_5 (SCR_5)  Address: H'FFF602
     | Bit | Bit Name | Setting | R/W | Description                      |
     |-----|----------|---------|-----|----------------------------------|
     | 6   | RIE      | 0       | R/W | Receive Interrupt Enable        |
     |     |          |         |     | 0: Disables RXI and ERI interrupt requests |
     |     |          |         |     | 1: Enables RXI and ERI interrupt requests |

   • DMA Mode Control Register_1 (DMDR_1)  Address: H'FFFC34
     | Bit | Bit Name | Setting | R/W | Description                      |
     |-----|----------|---------|-----|----------------------------------|
     | 8   | DTIE     | 0       | R/W | Data Transfer Interrupt Enable  |
     |     |          |         |     | 0: Disables transfer end interrupt requests |
     |     |          |         |     | 1: Enables transfer end interrupt requests |

5. Flowchart

```
  dmtend1_int
  RIE of SCR_5 = 0
  Disable receive end interrupts
  DTIE of DMDR_1 = 0
  Disable DMAC_1 transfer end interrupts
  rcnt = 128
  Set the transfer counter to 128
  End
```
5.6.7 eri5_int Function

1. Overview
   Handler for the receive error interrupts. Transfers one byte.

2. Arguments
   None

3. Return value
   None

4. Description of internal register usage
   Usage of internal registers in this task is described below. The given settings are those used in the task and differ from the initial settings.

   **Serial Status Register_5 (SSR_5)**  Address: H'FFF604

<table>
<thead>
<tr>
<th>Bit</th>
<th>Bit Name</th>
<th>Setting</th>
<th>R/W</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>ORER</td>
<td>0</td>
<td>R/(W)*</td>
<td>Overrun Error</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>[Setting condition]</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>• Occurrence of an overrun error during reception</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>[Clearing condition]</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>• Writing of 0 to ORER after having read ORER = 1</td>
</tr>
<tr>
<td>4</td>
<td>FER</td>
<td>0</td>
<td>R/(W)*</td>
<td>Framing Error</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>[Setting condition]</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>• Occurrence of a framing error during reception</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>[Clearing condition]</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>• Writing of 0 to FER after having read FER = 1</td>
</tr>
<tr>
<td>3</td>
<td>PER</td>
<td>0</td>
<td>R/(W)*</td>
<td>Parity Error</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>[Setting condition]</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>• Occurrence of a parity error during reception</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>[Clearing condition]</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>• Writing of 0 to PER after having read PER = 1</td>
</tr>
</tbody>
</table>

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

5. Flowchart

```
eri5_int
errbuf = SSR_5
SSR_5 &= H'87
Clear ORER, FER, and PER to 0
```
5.6.8 tei5_int Function

1. Overview
   Handler for the transmit end interrupt function. Transmits one byte.

2. Arguments
   None

3. Return value
   None

4. Description of internal register usage
   Usage of internal registers in this task is described below. The given settings are those used in the task and differ from the initial settings.

- **Serial Control Register_5 (SCR_5) Address: H'FFF602**

<table>
<thead>
<tr>
<th>Bit</th>
<th>Bit Name</th>
<th>Setting</th>
<th>R/W</th>
<th>Description</th>
</tr>
</thead>
</table>
| 5   | TE       | 0       | R/W | Transmit Enable  
 |     |          |        | 1: Disables transmission  
 |     |          |        | 0: Enables transmission  |
| 2   | TEIE     | 0       | R/W | Transmit End Interrupt Enable  
 |     |          |        | 0: Disables TEI interrupt requests  
 |     |          |        | 1: Enables TEI interrupt requests  |

5. Flowchart

```
  tei5_int
    |-----------------------------|
    | TE of SCR_5 = 0             |
    | Disable transmission        |
    |-----------------------------|
    | TEIE of SCR_5 = 0           |
    | Disable TEI interrupt requests|
    |-----------------------------|
    | endflg = 1                  |
    |-----------------------------|
    | End                         |
```
## Revision Record

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