To our customers,

Old Company Name in Catalogs and Other Documents

On April 1st, 2010, NEC Electronics Corporation merged with Renesas Technology Corporation, and Renesas Electronics Corporation took over all the business of both companies. Therefore, although the old company name remains in this document, it is a valid Renesas Electronics document. We appreciate your understanding.

Renesas Electronics website: http://www.renesas.com

April 1st, 2010
Renesas Electronics Corporation

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

Send any inquiries to http://www.renesas.com/inquiry.

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 from 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 depends 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.

SH7206 Group

Example of Setting the SCIF for Clocked Synchronous Serial Communication (Full-Duplex Communication)

Introduction

This application note presents an example of configuring the serial communication interface with FIFO (SCIF) for clocked synchronous communication.

Target Device

SH7206

Contents

1. Overview

1.1 Specifications

- Channel 3 of the SCIF is initialized as a transmitter/receiver module in clocked synchronous mode.
- Full-duplex communication is carried out on SCIF channel 3 using the transmit and receive interrupts. A transmit FIFO data empty interrupt and a receive FIFO data full interrupt initiate the corresponding interrupt processing for transmission and reception, respectively.

1.2 MCU Functions Used

- SCIF channel 3

1.3 Conditions for Application

• MCU:	SH7206 (R5S72060)	
• Operating frequency:	Internal clock:	200 MHz
	Bus clock:	66.67 MHz
	Peripheral clock:	33.33 MHz
• C compiler:	SuperH RISC engine Family C/C++ Compiler Package: Version 9.00 (from Renesas Technology Corp.)	
• Compiler options:	Default setting of HEW (-cpu=sh2a -debug -gbr=auto -global_volatile=0 -opt_range=all -infinite_loop=0 -del_vacant_loop=0 -struct_alloc=1)	

1.4 Related Application Note

The operation of the sample program in this application note was confirmed with the configuration specified in the application note "Example of SH7206 Initial Configuration". Please refer to that note in combination with this one.

2. Description of Sample Application

This sample application uses the serial communication interface with FIFO (SCIF).

2.1 Summary of MCU Functions Used

Data transmission/reception is synchronized with clock signal pulses in SCIF clocked synchronous mode. Either an internal clock or an external clock input from the SCK pin can be selected as the clock source. When an internal clock is selected, the serial clock is output via the SCK pin. When an external clock is selected, the serial clock is input from the SCK pin.

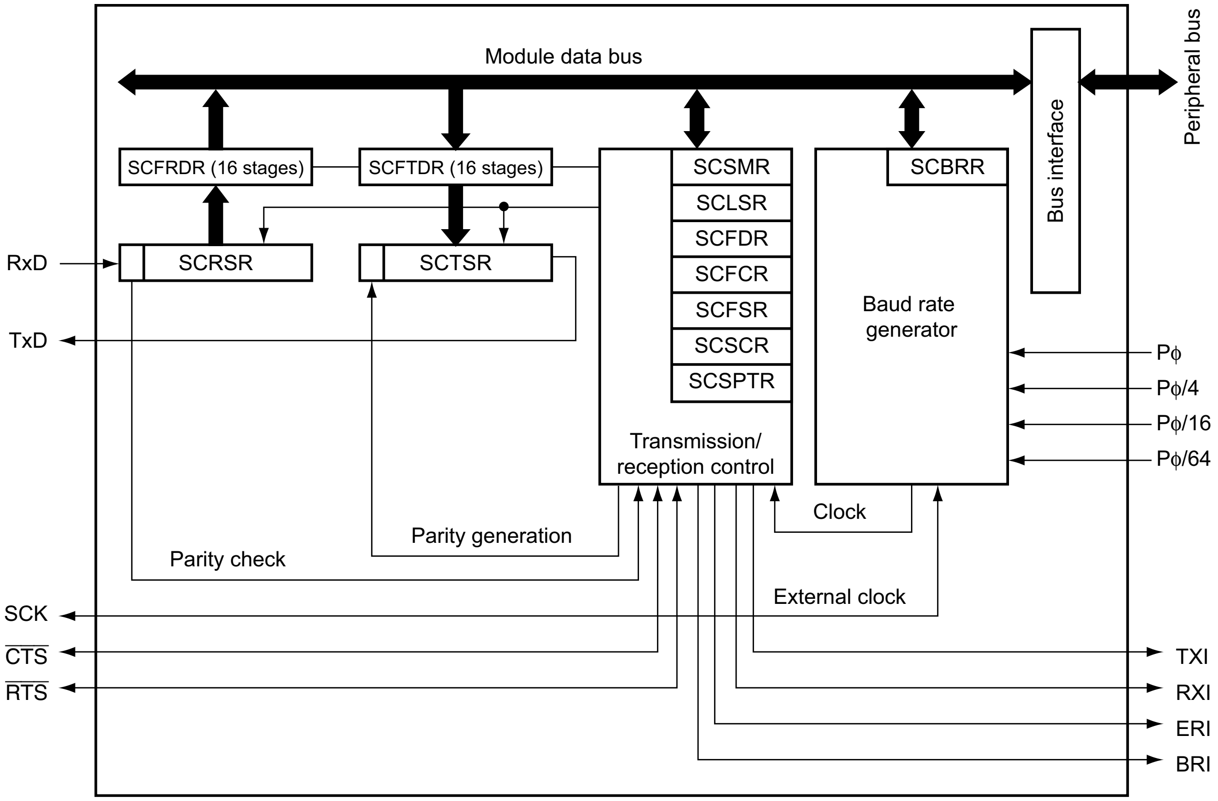
Communication data format is fixed at 8-bit length.

Table 1 summarizes the features of clocked synchronous mode. Figure 1 is a block diagram of the SCIF.

Table 1 Summary of SCIF (Clocked Synchronous Mode)

Item	Function
Number of channels	4 (SCIF0 to SCIF3)
Clock source	Internal clock: Pϕ, Pϕ/4, Pϕ/16, or Pϕ/64 (Pϕ: Peripheral clock) External clock: Clock input from the SCK0 to SCK3 pins
Data format	Transfer data length: 8 bits Transfer order: LSB first
Baud rate	Internal clock: 500 bps to 1000 kbps (Pϕ = 33 MHz) External clock: 5500 kbps at maximum (Pϕ = 33 MHz and externally input clock = 5.5 MHz)
Error detection	Overrun error
Interrupt requests	Transmit FIFO data empty interrupt (TXI), receive FIFO data full interrupt (RXI), and break interrupt (BRI), receive error (overrun error) interrupt (ERI)
Others	• When an internal clock is selected, the serial clock is output from the SCK pin. • Clock supply to unused channels can be stopped to save power. • The number of valid data bytes in the transmit and receive FIFO data registers and the number of receive errors of the receive data in the receive FIFO data register can be ascertained.

Note: * For details on the SCIF, refer to section 15, Serial Communication Interface with FIFO (SCIF), of the SH7206 Group Hardware Manual.



[Legend]

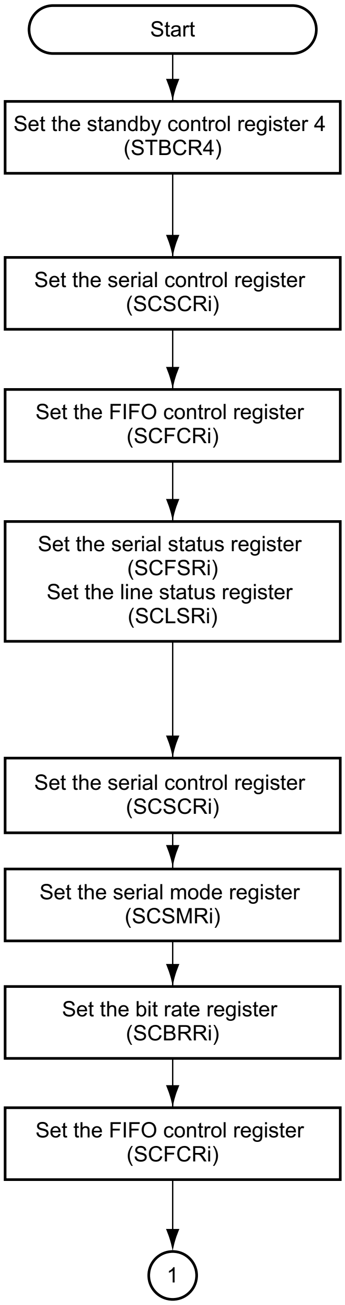
SCRSR:	Receive shift register	SCFSR:	Serial status register
SCFRDR:	Receive FIFO data register	SCBRR:	Bit rate register
SCTSR:	Transmit shift register	SCSPTR:	Serial port register
SCFTDR:	Transmit FIFO data register	SCFCR:	FIFO control register
SCSMR:	Serial mode register	SCFDR:	FIFO data count set register
SCSCR:	Serial control register	SCLSR:	Line status register

Figure 1 Block Diagram of SCIF

2.2 Procedure for Setting the MCU Modules

This section describes the basic setting procedures for SCIF clocked synchronous mode. Figures 2 and 3 show the example flow of initial settings for data transmission/reception in clocked synchronous mode. Figure 4 shows the example flow of transmit interrupt processing in clocked synchronous mode. Figure 5 shows the example flow of receive interrupt processing in clocked synchronous mode.

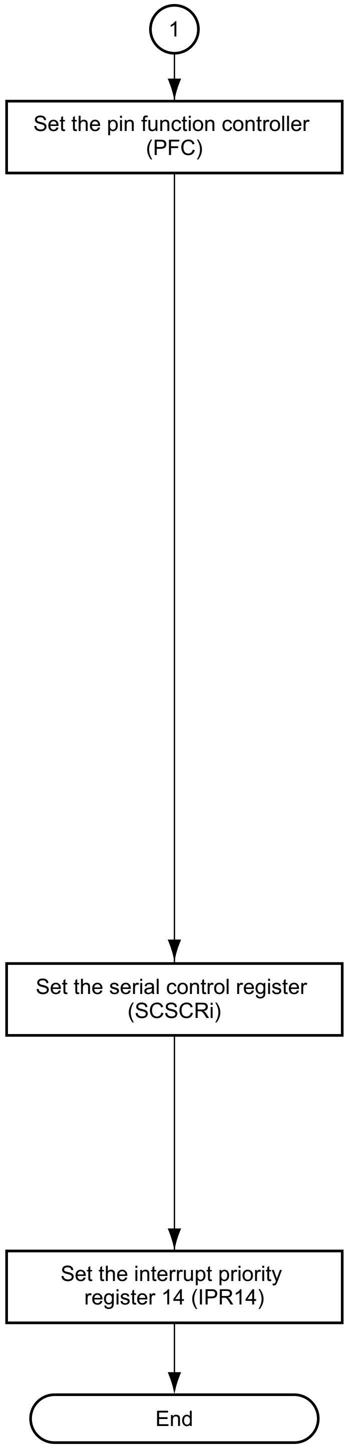
For details on the settings of individual registers, refer to the SH7206 Group Hardware Manual.


Start
Set the standby control register 4 (STBCR4)
Set the serial control register (SCSCRi)
Set the FIFO control register (SCFCRi)
Set the serial status register (SCFSRi)
Set the line status register (SCLSRi)
Set the serial control register (SCSCRi)
Set the serial mode register (SCSMRi)
Set the bit rate register (SCBRRi)
Set the FIFO control register (SCFCRi)
1

- Enabling clock supply to SCIFi (STBCR4)
  SCIF0: Clear the MSTP47 (module stop 47) bit to 0.
  SCIF1: Clear the MSTP46 (module stop 46) bit to 0.
  SCIF2: Clear the MSTP45 (module stop 45) bit to 0.
  SCIF3: Clear the MSTP44 (module stop 44) bit to 0.
  [Function] Enables clock supply to SCIFi.
- Disabling transmission/reception (SCSCRi)
  Clear the TE (transmit enable) bit to 0.
  [Function] Disables transmission.
  Clear the RE (receive enable) bit to 0.
  [Function] Disables reception.
- Invalidating data in the FIFOs (SCFCRi)
  Set the RFRST (receive FIFO data register reset) bit to 1.
  [Function] Invalidates the received data.
  Set the TFRST (transmit FIFO data register reset) bit to 1.
  [Function] Invalidates the transmit data.
- Clearing error status (SCFSRi, SCLSRi)
  Read each bit, and then write 0.
  Clear the ER (receive error) bit to 0.
  [Function] Clears the receive error flag.
  Clear the BRK (break detection) bit to 0.
  [Function] Clears the break detection flag.
  Clear the DR (receive data ready) bit to 0.
  [Function] Clears the receive data ready flag.
  Clear the ORER (overrun error) bit to 0.
  [Function] Clears the overrun error flag.
- Clock source selection and enabling/disabling clock output (SCSCRi)
  Set the CKE[1:0] (clock enable) bits.
  [Function] Selects the clock source.
- Setting serial communication format (SCSMRi)
  Set the C/$\overline{A}$ (communication mode) bit to 1.
  [Function] Selects clocked synchronous mode.
  Set the CKS[1:0] (clock select) bits.
  [Function] Selects a specific internal clock source.
- Setting communication rate (SCBRRi)
- Setting FIFO data trigger number and canceling FIFO reset state (SCFCRi)
  Set the RTRG[1:0] (receive FIFO data trigger) bits.
  [Function] Selects the receive trigger number.
  Set the TTRG[1:0] (transmit FIFO data trigger) bits.
  [Function] Selects the transmit trigger number.
  Clear the RFRST (receive FIFO data register reset) bit to 0.
  [Function] Validates the received data.
  Clear the TFRST (transmit FIFO data register reset) bit to 0.
  [Function] Validates the transmit data.
  Set the LOOP (loop-back test) bit.
  [Function] Enables or disables loop-back test.

Note: i = 0 to 3

Figure 2 Example Flow of Initial Settings for Data Transmission/Reception in Clocked Synchronous Mode (1)


1
Set the pin function controller (PFC)
Set the serial control register (SCSCRi)
Set the interrupt priority register 14 (IPR14)
End

- Setting the multiplexed pins as serial communication pins
  (PACRL1, PACRL2, PECRL2, PECRL3)
  SCIF0:
  Set the PA0MD (PA0 mode) bits to B'001.
  [Function] Selects RxD0 pin function.
  Set the PA1MD (PA1 mode) bits to B'001.
  [Function] Selects TxD0 pin function.
  Set the PA2MD (PA2 mode) bits to B'001.
  [Function] Selects SCK0 pin function.
  SCIF1:
  Set the PA3MD (PA3 mode) bits to B'001.
  [Function] Selects RxD1 pin function.
  Set the PA4MD (PA4 mode) bits to B'001.
  [Function] Selects TxD1 pin function.
  Set the PA5MD (PA5 mode) bits to B'001.
  [Function] Selects SCK1 pin function.
  SCIF2:
  Set the PE7MD (PE7 mode) bits to B'010.
  [Function] Selects RxD2 pin function.
  Set the PE10MD (PE10 mode) bits to B'10.
  [Function] Selects TxD2 pin function.
  Set the PE8MD (PE8 mode) bits to B'10.
  [Function] Selects SCK2 pin function.
  SCIF3:
  Set the PE4MD (PE4 mode) bits to B'010 or
  the PE11MD (PE11 mode) bits to B'011.
  [Function] Selects RxD3 pin function.
  Set the PE5MD (PE5 mode) bits to B'010 or
  the PE12MD (PE12 mode) bits to B'11.
  [Function] Selects TxD3 pin function.
  Set the PE6MD (PE6 mode) bits to B'010 or
  the PE9MD (PE9 mode) bits to B'011.
  [Function] Selects SCK3 pin function.

- Enabling transmission/reception and interrupts (SCSCRi)
  Set the TIE (transmit interrupt enable) bit to 1.
  [Function] Enables the transmit FIFO data empty interrupt (TXI).
  Set the RIE (receive interrupt enable) bit to 1.
  [Function] Enables receive FIFO data full interrupt (RXI).
  Set the TE (transmit enable) bit to 1.
  [Function] Enables transmission.
  Set the RE (receive enable) bit to 1.
  [Function] Enables reception.
  Set the REIE (receive error interrupt enable) bit to 1.
  [Function] Enables the receive error interrupt (ERI) and break interrupt (BRI).

- Enabling the TXI3 and RXI3 interrupts
  Set the interrupt priority level to 15.

Note: i = 0 to 3

Figure 3 Example Flow of Initial Settings for Data Transmission/Reception in Clocked Synchronous Mode (2)

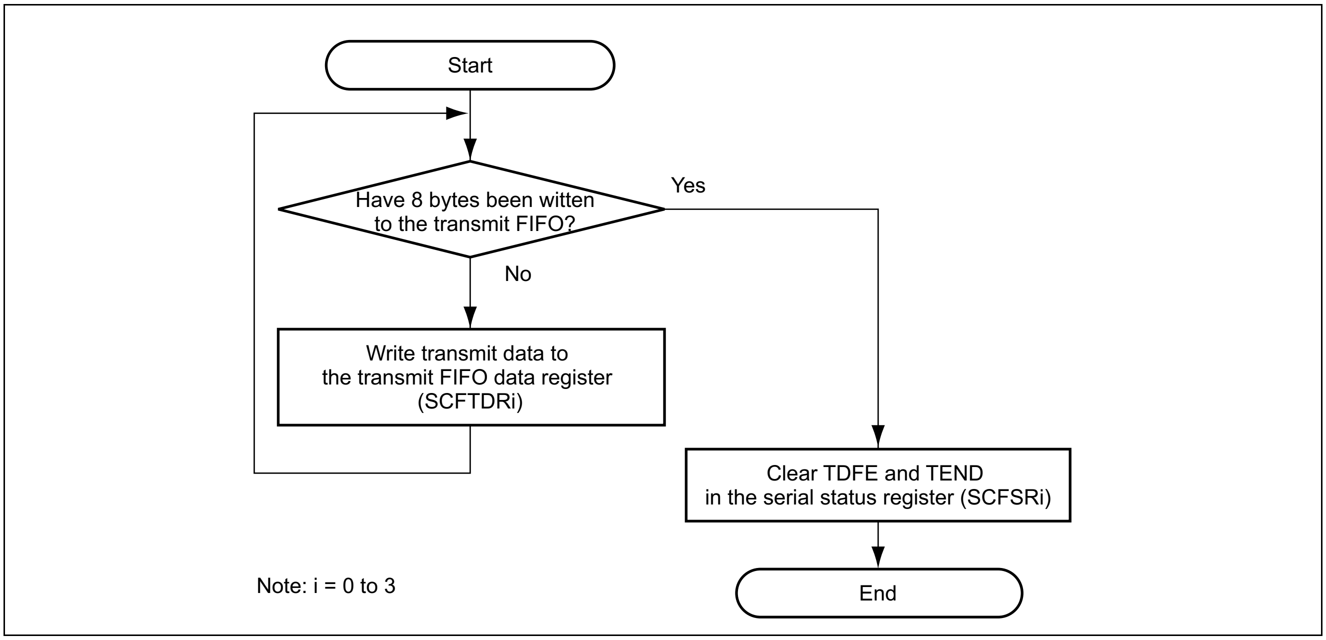

Start
Have 8 bytes been witten to the transmit FIFO?
Yes
No
Write transmit data to the transmit FIFO data register (SCFTDRi)
Clear TDFE and TEND in the serial status register (SCFSRi)
Note: i = 0 to 3
End

Figure 4 Example Flow of Transmit Interrupt Processing in Clocked Synchronous Mode

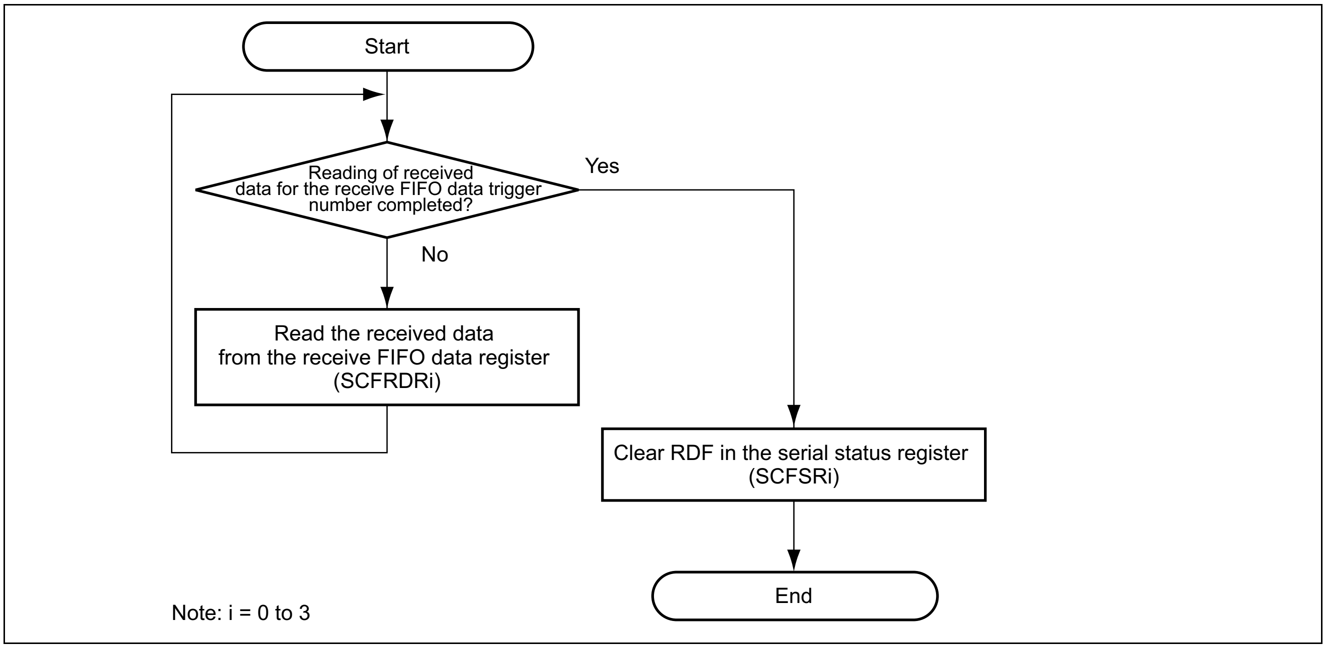

Start
Reading of received data for the receive FIFO data trigger number completed?
Yes
No
Read the received data from the receive FIFO data register (SCFRDRi)
Clear RDF in the serial status register (SCFSRi)
End
Note: i = 0 to 3

Figure 5 Example Flow of Receive Interrupt Processing in Clocked Synchronous Mode

2.3 Operation of Sample Program

The sample program uses the SCIF channel 3 as a data transmission/reception module in clocked synchronous mode. Full-duplex communication is carried out using the loop-back test function of SCIF channel 3, in which the serial transmit pin TxD3 and the serial receive pin RxD3 are internally connected. The loop-back test function is enabled by setting the loop-back test (LOOP) bit in the FIFO control register (SCFCR) to 1.

Table 2 shows the communication function settings of the sample program. Figure 6 illustrates the timing of the sample program operation.

Table 2 Communication Function Settings of Sample Program

Communication Format	Function Setting
Communication mode	Clocked synchronous mode
Channel used	Channel 3
Interrupts	Transmit FIFO data empty and receive FIFO data full interrupts
Communication rate	100 kbps
Data size	8 bits
Bit transfer order	LSB first
Serial clock	Serial clock output by SCIF3
FIFO data trigger number	Reception: 8; transmission: 8
Loop-back test function	Enabled (the TxD3 and RxD3 pins are internally connected)

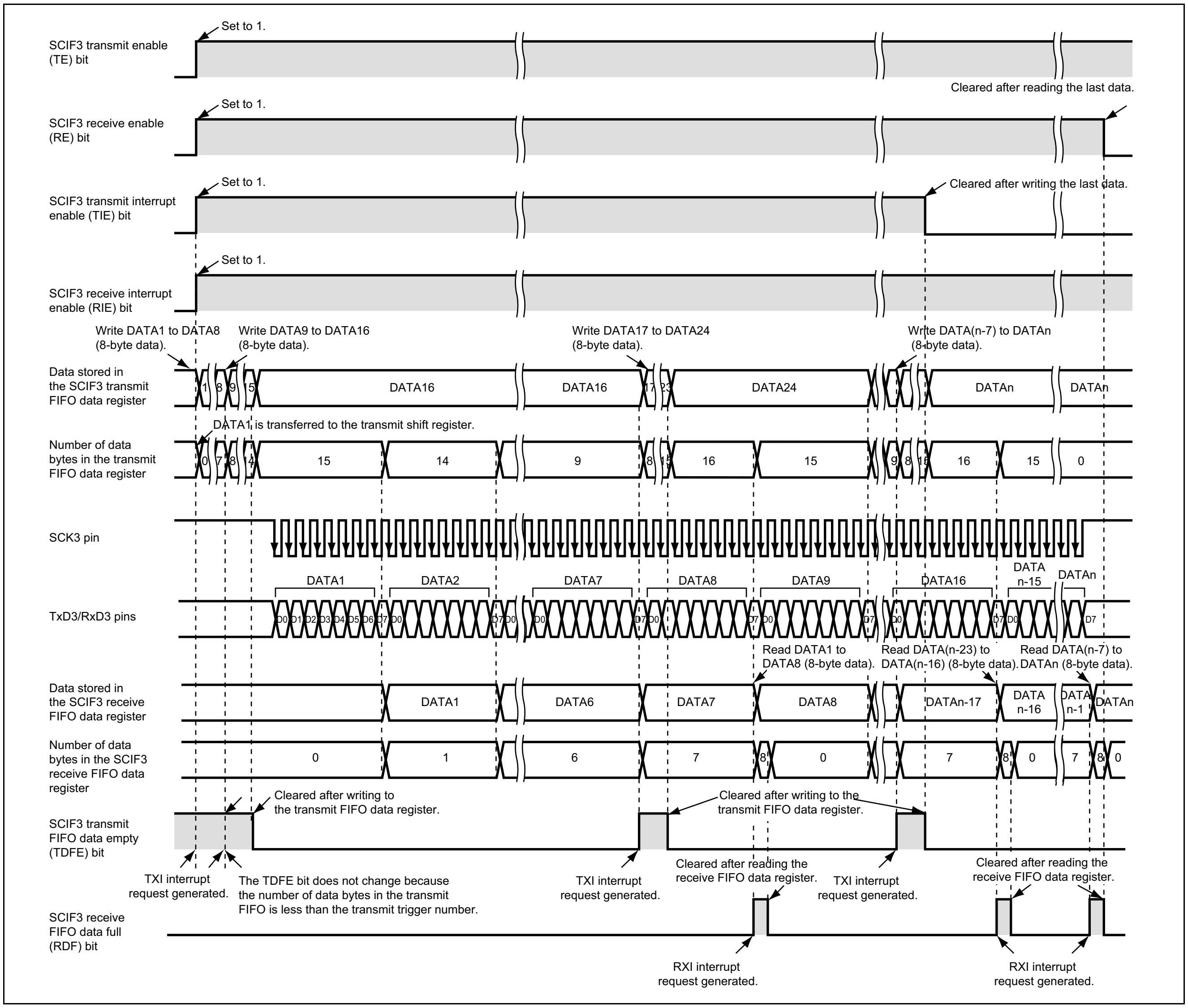

SCIF3 transmit enable (TE) bit
Set to 1.
Cleared after reading the last data.
SCIF3 receive enable (RE) bit
Set to 1.
SCIF3 transmit interrupt enable (TIE) bit
Set to 1.
Cleared after writing the last data.
SCIF3 receive interrupt enable (RIE) bit
Set to 1.
Write DATA1 to DATA8 (8-byte data).
Write DATA9 to DATA16 (8-byte data).
Write DATA17 to DATA24 (8-byte data).
Write DATA(n-7) to DATAn (8-byte data).
Data stored in the SCIF3 transmit FIFO data register
1
8
9
15
DATA16
DATA16
17
23
DATA24
DATAn
DATAn
DATA1 is transferred to the transmit shift register.
Number of data bytes in the transmit FIFO data register
0
7
8
14
15
14
9
8
15
16
15
9
8
16
16
15
0
SCK3 pin
DATA1
DATA2
DATA7
DATA8
DATA9
DATA16
DATA n-15
DATAn
TxD3/RxD3 pins
D0
D1
D2
D3
D4
D5
D6
D7
Read DATA1 to DATA8 (8-byte data).
Read DATA(n-23) to DATA(n-16) (8-byte data).
Read DATA(n-7) to DATAn (8-byte data).
Data stored in the SCIF3 receive FIFO data register
DATA1
DATA6
DATA7
DATA8
DATAn-17
DATA n-16
DATA n-1
DATAn
Number of data bytes in the SCIF3 receive FIFO data register
0
1
6
7
8
0
7
8
0
7
8
0
SCIF3 transmit FIFO data empty (TDFE) bit
Cleared after writing to the transmit FIFO data register.
Cleared after writing to the transmit FIFO data register.
TXI interrupt request generated.
The TDFE bit does not change because the number of data bytes in the transmit FIFO is less than the transmit trigger number.
TXI interrupt request generated.
Cleared after reading the receive FIFO data register.
TXI interrupt request generated.
Cleared after reading the receive FIFO data register.
SCIF3 receive FIFO data full (RDF) bit
RXI interrupt request generated.
RXI interrupt request generated.

Figure 6 Timing of Sample Program Operation

2.4 Register Settings and Processing Sequence of Sample Program

The sample program uses the SCIF channel 3 as a data transmission/reception module in clocked synchronous mode to perform full-duplex communication using the transmit and receive interrupts.

With the transmit FIFO trigger number being set to 8, the transmit interrupt processing routine writes 8 bytes of transmit data to the transmit FIFO data register (SCFTDR_3).

With the receive FIFO trigger number being set to 8, the receive interrupt processing routine reads 8 bytes of received data from the receive FIFO data register (SCFRDR_3).

The register settings of the sample program are shown in table 3, and processing flows of the sample program are shown in figures 7 to 9.

Table 3 Register Settings of Sample Program

Register Name	Address	Setting	Description
Standby control register 4 (STBCR4)	H'FFFE040C	H'E4	MSTP44 = 0: The SCIF3 runs (clock is supplied).
Port E control register L4 (PECRL4)	H'FFFE3A10	H'0003	PE12MD[1:0] = B'11: TxD3 output mode
Port E control register L3 (PECRL3)	H'FFFE3A12	H'3030	PE11MD[2:0] = B'011: RxD3 output mode PE9MD[2:0] = B'011: SCK3 input/output mode
Serial mode register_3 (SCSMR_3)	H'FFFE9800	H'0080	$C/\overline{A}$ = 1: Clocked synchronous mode CKS[1:0] = B'00: $P\phi$ clock

Serial control register_3 (SCSCR_3)	H'FFFE9808	H'0000	TE = 0: Transmission disabled
			RE = 0: Reception disabled
			CKE[1:0] = B'00: The SCK pin outputs an internal clock as the serial clock.

		H'00F8	TIE = 1: Transmit interrupt enabled
			RIE = 1: Receive interrupt enabled
			TE = 1: Transmission enabled
			RE = 1: Reception enabled
			REIE = 1: Break interrupt enabled

FIFO control register_3 (SCFCR_3)	H'FFFE9818	H'0006	TFRST = 1: Reset operation for the transmit FIFO data register is enabled. FRFST = 1: Reset operation for the receive FIFO data register is enabled.
		H'0081	RTRG[1:0] = B'10: Receive FIFO data trigger number is 8. *[1] TTRG[1:0] = B'00: Transmit FIFO data trigger number is 8. *[2] TFRST = 0: Reset operation for the transmit FIFO data register is disabled. RFRST = 0: Reset operation for the receive FIFO data register is disabled. LOOP = 1: Loop-back test is enabled.

Register Name	Address	Setting	Description
Bit rate register_3 (SCBRR_3)	H'FFFE9804	H'07	100 kbps
Serial status register_3 (SCFSR_3)	H'FFFE9810	H'FF6E*[3]	ER = 0: Receive error flag cleared BRK = 0: Break detection flag cleared DR = 0: Receive data ready flag cleared To clear these bits, read the bits while they are set and write 0 to them.
Line status register_3 (SCLSR_3)	H'FFFE9824	H'0000	ORER = 0: Overrun error flag cleared To clear this bit, read the bit while it is set and write 0 to it.

Notes: 1. Receive FIFO data trigger number is the number of data bytes in the receive FIFO which sets the RDF flag in the serial status register (SCFSR).

2. Transmit FIFO data trigger number is the number of remaining data bytes in the transmit FIFO which sets the transmit FIFO data empty (TDFE) flag in the serial status register (SCFSR).

3. The register value is ANDed with H'FF6E to clear the ER, BRK, and DR bits.

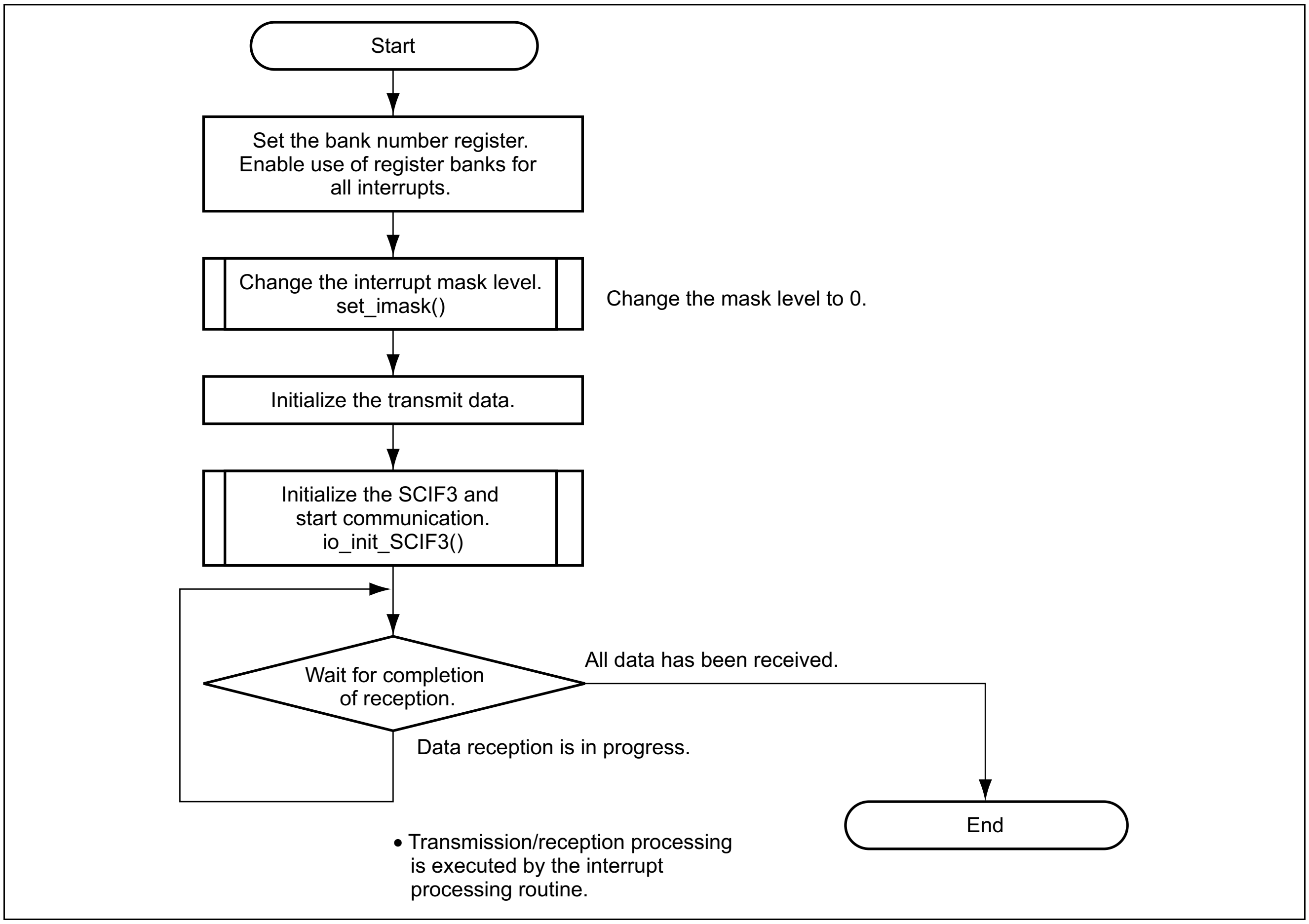

Start
Set the bank number register. Enable use of register banks for all interrupts.
Change the interrupt mask level. set_imask()
Change the mask level to 0.
Initialize the transmit data.
Initialize the SCIF3 and start communication. io_init_SCIF3()
Wait for completion of reception.
All data has been received.
Data reception is in progress.
End
• Transmission/reception processing is executed by the interrupt processing routine.

Figure 7 Main Processing Flow of Sample Program

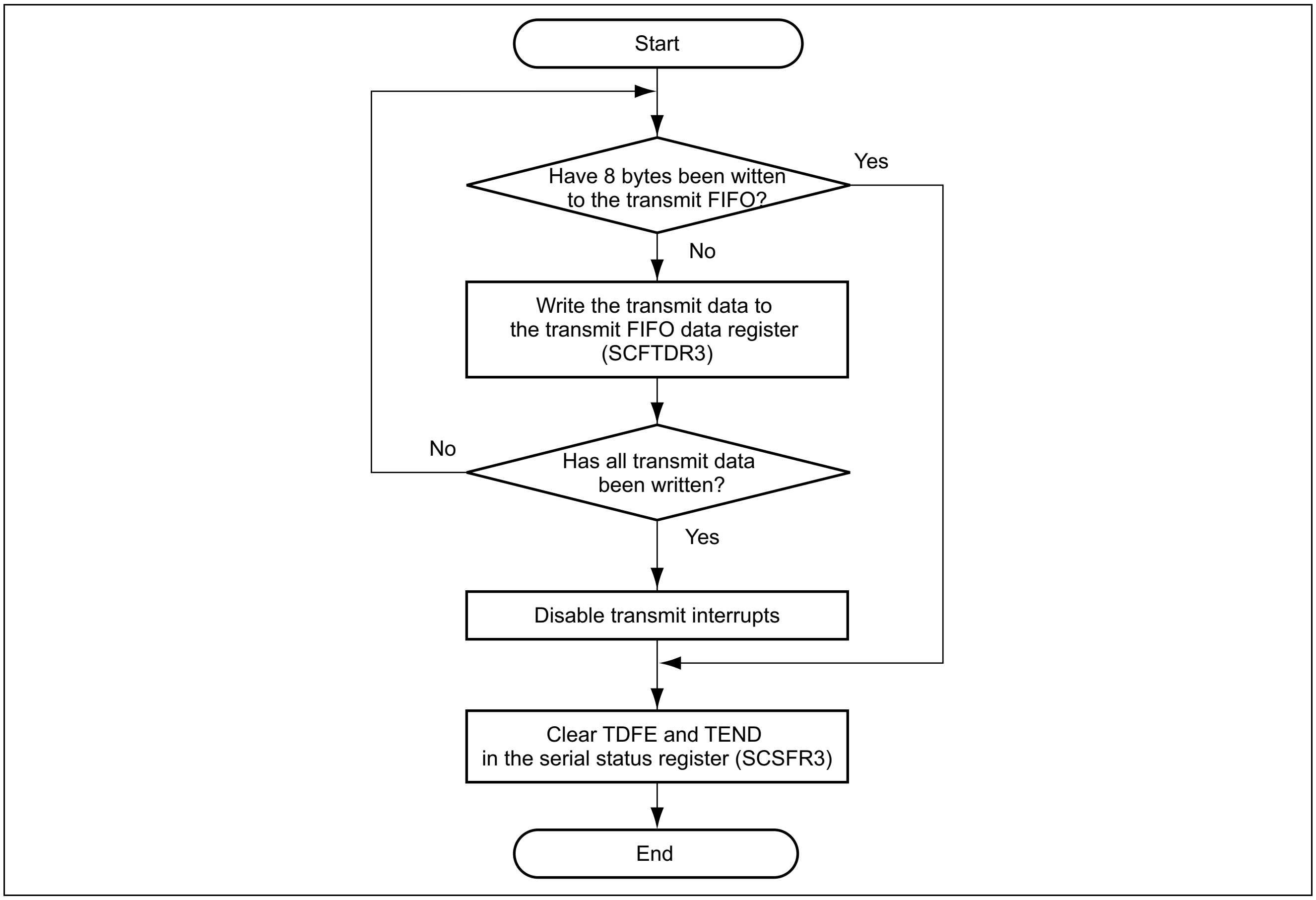

Start
Have 8 bytes been witten to the transmit FIFO?
Yes
No
Write the transmit data to the transmit FIFO data register (SCFTDR3)
No
Has all transmit data been written?
Yes
Disable transmit interrupts
Clear TDFE and TEND in the serial status register (SCSFR3)
End

Figure 8 Transmit Interrupt Processing Flow of Sample Program

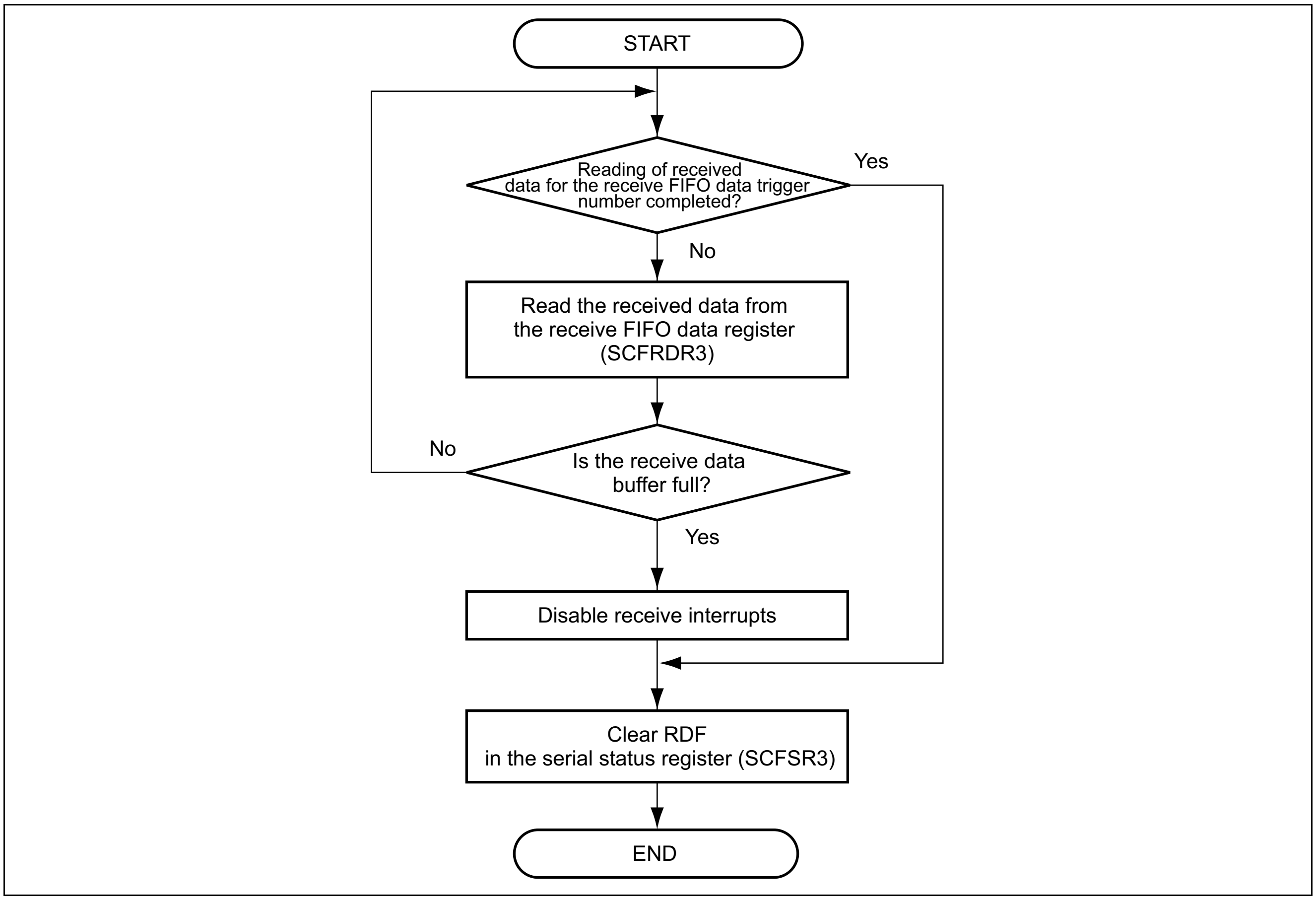

START
Reading of received data for the receive FIFO data trigger number completed?
Yes
No
Read the received data from the receive FIFO data register (SCFRDR3)
No
Is the receive data buffer full?
Yes
Disable receive interrupts
Clear RDF in the serial status register (SCFSR3)
END

Figure 9 Receive Interrupt Processing Flow of Sample Program

3. Sample Program Listing

- Sample Program Listing: main.c (1)

```
/*""FILE COMMENT""*****************************************************************************
 *
 *   System Name: SH7206 Sample Program
 *   File Name  : main.c
 *   Contents   : Sample program for clocked synchronous serial communication and
 *                interrupt processing by the serial communication interface with FIFO (SCIF)
 *   Version    : 1.00.00
 *   Model      : M3A-HS60
 *   CPU        : SH7206
 *   Compiler   : SHC9.0.00
 *
 *   note       : Sample program for clocked synchronous full-duplex communication by SCIF3
 *
 *                <Caution>
 *                Loop-back test is enabled to allow unit evaluation of the SH7206.
 *
 *                This sample program is for reference and its operation is not guaranteed.
 *                Customers should use this sample program for technical reference
 *                in software development.
 *
 *
 *    Copyright (C) 2004 Renesas Technology Corp. All Rights Reserved
 *    and Renesas Solutions Corp. All Rights Reserved
 *
 *    history   : 2004.10.27 ver.1.00.00
 *""FILE COMMENT END""*************************************************************************/
#include <machine.h>
#include "iodefine.h"                      /* iodefine.h is automatically created by HEW    */

/* ==== Prototype declaration ==== */
void main(void);
void io_init_SCIF3(int);

/* ==== Type declaration ==== */
/* SCIF baud rate setting */
typedef struct  {
    unsigned char scbrr;
    unsigned short scsmr;
} SH7206_BAUD_SET;

/* ---- Values for baud rate specification ---- */
enum{
    CBR_500,
    CBR_1K,
    CBR_2_5K,
    CBR_5K,
    CBR_10K,
    CBR_25K,
    CBR_50K,
    CBR_100K,
    CBR_250K,
    CBR_500K,
    CBR_1M
};

```

- Sample Program Listing: main.c (2)

```
/* ==== Bit rate setting value table ==== */
SH7206_BAUD_SET scif_baud[] = {
    {255, 3},     /*       500Hz  */
    {125, 3},     /*     1,000Hz  */
    {200, 2},     /*     2,500Hz  */
    {100, 2},     /*     5,000Hz  */
    {200, 1},     /*    10,000Hz  */
    { 80, 1},     /*    25,000Hz  */
    {160, 0},     /*    50,000Hz  */
    { 80, 0},     /*   100,000Hz  */
    { 31, 0},     /*   250,000Hz  */
    { 15, 0},     /*   500,000Hz  */
    {  7, 0}      /* 1,000,000Hz  */
};

#define DATA_NUM    0x100u
unsigned int DataNum = DATA_NUM;         /* Number of transmit data bytes                  */
unsigned int SndCnt;                     /* Transmit counter                               */
unsigned int RcvCnt;                     /* Receive counter                                */
unsigned char SndData[DATA_NUM];         /* Transmit data buffer                           */
unsigned char RcvData[DATA_NUM];         /* Receive data buffer                            */

/*""FUNC COMMENT""*****************************************************************************
* ID            :
* Module summary: Main function of the sample program
*               : (processing for clocked synchronous serial I/O communication)
*----------------------------------------------------------------------------------------------
* Include       : None
*----------------------------------------------------------------------------------------------
* Declaration   : void main(void)
*----------------------------------------------------------------------------------------------
* Functional description:
*               : Transmits and receives incremental counter values using
*               : the transmit FIFO data empty interrupt, receive FIFO data full interrupt,
*               : and FIFO of the SCIF3.
*----------------------------------------------------------------------------------------------
* Argument      : None
*----------------------------------------------------------------------------------------------
* Return value  : None
*----------------------------------------------------------------------------------------------
* Notes         :
*""FUNC COMMENT END""*************************************************************************/
void main(void)
{
    int i;

    /* ==== Set bank number register ==== */
    INTC.IBNR.BIT.BE = 0x01;             /* Enable use of register banks for all interrupts */

    /* ==== Change interrupt mask level ==== */
    set_imask(0);

    /* ==== Initialize transmit data ==== */
    for(i=0; i < DATA_NUM; i++){
        SndData[i] = (unsigned char)i;
    }

    /* ==== Initializeg the SCIF3 and starting communication ==== */
    io_init_SCIF3(CBR_100K);             /* Bit rate: 100 kbps */

    /* ==== Wait until reception is complete ==== */
    while(RcvCnt < DATA_NUM){
        /* Execute transmission/reception as interrupt processing */
    }

    while (1) {
        /* End of Program */
    }
}

```

- Sample Program Listing: main.c (3)

```
/*""FUNC COMMENT""****************************************************************************
 * ID              :
 * Module Summary: SCIF3 initialization
 *-------------------------------------------------------------------------------------------
 * Include         : None
 *-------------------------------------------------------------------------------------------
 * Declaration     : void io_init_SCIF3(int bps)
 *-------------------------------------------------------------------------------------------
 * Functional description: Initializes SCIF3 in clocked synchronous mode.
 *                 : Clocked synchronous mode/ 100 kbps/ 8 bits/ serial clock output
 *                 : Enables transmit and receive interrupts.
 *-------------------------------------------------------------------------------------------
 * Argument        : int bps : Value for bit rate setting
 *-------------------------------------------------------------------------------------------
 * Return value    : None
 *-------------------------------------------------------------------------------------------
 * Notes           : The baud rate setting values given in this program are those when the
 *                 : peripheral clock (Pf) frequency is 33 MHz.
 *                 : If a different clock is used, the baud rate setting values must be changed.
 *                 : Loop-back test is enabled to allow unit evaluation of the SH7206.
 *""FUNC COMMENT END""*********************************************************************/
void io_init_SCIF3(int bps)
{
    /* ==== Canceling power-down mode ====*/
    /* ---- Set standby control register 4 (STBCR4) ---- */
    CPG.STBCR4.BIT.MSTP44 = 0;           /* Start clock supply to SCIF3                    */

    /* ==== SCIF3 initialization  ==== */
    /* ---- Set serial control register (SCSCRi) ---- */
    SCIF3.SCSCR.WORD = 0x0000;           /* Disables transmission/reception by SCIF3       */

    /* ---- Set FIFO control register (SCFCRi)   ---- */
    SCIF3.SCFCR.WORD = 0x0006;           /* Reset receive/transmit FIFO data registers     */
    /* ---- Set serial status register (SCFSRi)  ---- */
    SCIF3.SCFSR.WORD &= 0xff6eu;         /* Clear the ER, BRK, and DR bits                 */

    /* ---- Set line status register (SCLSRi)    ---- */
    SCIF3.SCLSR.BIT.ORER = 0;            /* Clear the ORER bit                             */

    /* ---- Set serial mode register (SCSMRi)    ---- */
    SCIF3.SCSMR.WORD = scif_baud[bps].scsmr | 0x0080u ;
                                         /* Communication mode 1: Clocked synchronous mode  */
                                         /* Clock select        : Table value               */

    /* ---- Set serial control register (SCSCRi) ---- */
    SCIF3.SCSCR.BIT.CKE = 0x0;           /* B'00: Internal clock; serial clock output       */

    /* ---- Set bit rate register (SCBRRi)       ---- */
    SCIF3.SCBRR.BYTE = scif_baud[bps].scbrr;

```

- Sample Program Listing: main.c (4)

```
    /* ---- Setting FIFO control register (SCFCRi) ---- */
    SCIF3.SCFCR.WORD = 0x0081;           /* RTS output active trigger: Initial value        */
                                         /* Receive FIFO data trigger B'10: 8               */
                                         /* Transmit FIFO data trigger B'00: 8              */
                                         /* Modem control enable             : Disabled     */
                                         /* Receive FIFO data register reset : Disabled     */
                                         /* Transmit FIFO data register reset: Disabled     */
                                         /* Loop-back test                 1 : Enabled      */

    /* ==== Setting pin function controller (PFC) ==== */
    PORT.PECRL4.BIT.PE12MD = 3;          /* Set the PE12 pin for TxD3 output (PECRL4)       */
    PORT.PECRL3.BIT.PE11MD = 3;          /* Set the PE11 pin for RxD3 input (PECRL3)        */
    PORT.PECRL3.BIT.PE9MD  = 3;          /* Set the PE9 pin for SCK3 input/output (PECRL3)  */

    /* ---- Setting serial control register (SCSCRi) ---- */
    SCIF3.SCSCR.WORD |= 0x00f8u;    /* Enable SCIF3 transmission/reception and interrupts    */

    /* ==== Set interrupt priority register 14 (IPR14) ==== */
    INTC.IPR14.WORD = 0x000f;            /* Set the level to 15                             */

}
/* End of File */
```

- Sample Program Listing: intprg.c (1)

```
// 252 SCIF SCIF3 BRI3
/*""FUNC COMMENT""****************************************************************************
* ID            :
* Module summary: SCIF channel 3 receive error interrupt processing
*---------------------------------------------------------------------------------------------
* Include       : #include "vect.h"
*               : #include "iodefine.h"
*---------------------------------------------------------------------------------------------
* Declaration   : void INT_SCIF_SCIF3_BRI3(void)
*---------------------------------------------------------------------------------------------
* Functional description:
*               : Handles the overrun error generated in clocked synchronous mode.
*               : This sample program only clears the ORER bit.
*---------------------------------------------------------------------------------------------
* Argument      : None
*---------------------------------------------------------------------------------------------
* Return value  : None
*---------------------------------------------------------------------------------------------
* Note          : None
*""FUNC COMMENT END""**********************************************************************/
void INT_SCIF_SCIF3_BRI3(void)
{
    /* ==== Overrun error check ==== */
    if(SCIF3.SCLSR.BIT.ORER == 1) {
        /* Perform overrun error processing */
        SCIF3.SCLSR.BIT.ORER = 0;                /* Clear the ORER bit                      */
    }
}
```

- Sample Program Listing: intprg.c (2)

```
// 253 SCIF SCIF3 ERI3
void INT_SCIF_SCIF3_ERI3(void) {/* sleep(); */}
// 254 SCIF SCIF3 RXI3
/*""FUNC COMMENT""****************************************************************************
* ID                :
* Module summary    : SCIF channel 3 receive interrupt processing
*---------------------------------------------------------------------------------------------
* Include           : #include "vect.h"
*                   : #include "iodefine.h"
*---------------------------------------------------------------------------------------------
* Declaration       : void INT_SCIF_SCIF3_RXI3(void)
*---------------------------------------------------------------------------------------------
 Functional description:
*                   : This module is initiated by the SCIF3 receive FIFO data full interrupt.
*                   : In this sample application, the receive FIFO trigger number is set to 8
*                   : during initialization and data is read in 8-byte units.
*                   : Note that, in clocked synchronous mode, interrupts are not generated when
*                   : the number of data bytes that have been received is less than the receive
*                   : FIFO trigger number (when data transmission is interrupted).
*---------------------------------------------------------------------------------------------
* Argument          : None
*---------------------------------------------------------------------------------------------
* Return value      : None
*---------------------------------------------------------------------------------------------
* Note    : None
*""FUNC COMMENT END""************************************************************************/
void INT_SCIF_SCIF3_RXI3(void)
{
    unsigned short fifo_cnt;
    extern unsigned int  DataNum;                /* Number of transmit data bytes           */
    extern unsigned int  RcvCnt;                 /* Receive counter                         */
    extern unsigned char RcvData[];              /* Receive data buffer                     */


    /* ==== Set the number of bytes of data in FIFO to be acquired ==== */
    fifo_cnt = 8;

    /* ==== Repeat reception processing for the number of receive FIFO stages used ==== */
    while(fifo_cnt-- != 0 ){

        /* ==== Read received data ==== */
        RcvData[RcvCnt++] = SCIF3.SCFRDR.BYTE;   /* Store received data into the buffer     */

        /* Is the receive data buffer full? */
        if(RcvCnt == DataNum){
            SCIF3.SCSCR.BIT.RE = 0;              /* Disable reception                       */
            break;
        }
    }

    /* ==== Clear RDF in serial status register (SCFSR3) ==== */
    SCIF3.SCFSR.BIT.RDF = 0;                    /* Clear the RDF bit                        */
}
```

- Sample Program Listing: intprg.c (3)

```
// 255 SCIF SCIF3 TXI3
/*""FUNC COMMENT""****************************************************************************
* ID             :
* Module summary: SCIF3 transmit interrupt processing
*---------------------------------------------------------------------------------------------
* Include        : #include "vect.h"
*                : #include "iodefine.h"
*---------------------------------------------------------------------------------------------
* Declaration    : void INT_SCIF_SCIF3_TXI3(void)
*---------------------------------------------------------------------------------------------
* Functional description:
*                : This module is initiated by the SCIF3 transmit FIFO data empty interrupt.
*                : This sample program writes transmit data in 8-byte units in
*                : accordance with the number of data bytes received by the receiver.
*                : When there is no transmit data, the transmit FIFO data empty interrupt
*                : is disabled.
*---------------------------------------------------------------------------------------------
* Argument       : None
*---------------------------------------------------------------------------------------------
* Return value   : None
*---------------------------------------------------------------------------------------------
* Note           : After writing all transmit data, it can be confirmed that all transmit data
*                : in FIFO has been transmitted through the transmit end (TEND) bit in SCFSR.
*""FUNC COMMENT END""***********************************************************************/
void INT_SCIF_SCIF3_TXI3(void)
{
    unsigned short fifo_cnt;
    extern unsigned int DataNum;
    extern unsigned int SndCnt;
    extern unsigned char SndData[];

    /* ==== Set the number of data bytes to be written to FIFO ==== */
    fifo_cnt = 8;

    /* == Transmit 8 bytes of data from transmit FIFO == */
    while(fifo_cnt-- > 0){
        SCIF3.SCFTDR.BYTE = SndData[SndCnt++];

        /* Has all transmit data been written? */
        if(SndCnt == DataNum){
            /* Disable transmit interrupts */
            SCIF3.SCSCR.BIT.TIE = 0;
            break;
        }

    }

    /* ==== Clear TDEF and TEND bits in serial status register (SCFSR3) ==== */
    SCIF3.SCFSR.WORD &= ~0x0060u;
}

// Dummy
void Dummy(void){/* sleep(); */}

/* End of File */
```

- Sample Program Listing: vecttbl.c (1)

```
/*""FILE COMMENT""*******************************************************************************
*
*   System Name: SH7206 Sample Program
*   File Name  : vecttbl.c
*   Version    : 1.00.00
*   Contents   : Definition of exception handling vector table
*   Model      : M3A-HS60
*   CPU        : SH7206
*   Compiler   : SHC9.0.00
*   OS         : none
*
*   note       : This file was originally generated by Renesas Project Generator (Ver.3.1)
*              : and was modified for the application note.
*
*                <Caution>
*                 This sample program is for reference
*                 and its operation is not guaranteed.
*                 Customers should use this sample program for technical reference
*                 in software development.
*
*
*   This file is generated by Renesas Project Generator (Ver.3.1).
*
*   Copyright (C) 2004 Renesas Technology Corp. All Rights Reserved
*   AND Renesas Solutions Corp. All Rights Reserved
*
*   history    : 2004.10.14 ver.1.00.00
*""FILE COMMENT END""****************************************************************************/
#include "vect.h"

#pragma section VECTTBL

void (*RESET_Vectors[])(void) = {
//;<<VECTOR DATA START (POWER ON RESET)>>
//;0 Power On Reset PC
    PowerON_Reset_PC,
//;<<VECTOR DATA END (POWER ON RESET)>>
// 1 Power On Reset SP
    __secend("S"),
//;<<VECTOR DATA START (MANUAL RESET)>>
//;2 Manual Reset PC
    Manual_Reset_PC,
//;<<VECTOR DATA END (MANUAL RESET)>>
// 3 Manual Reset SP
    __secend("S")

};
#pragma section INTTBL
void (*INT_Vectors[])(void) = {
// 4 Illegal code

<<Omitted>>
// 252 SCIF SCIF3 BRI3
    INT_SCIF_SCIF3_BRI3,
// 253 SCIF SCIF3 ERI3
    INT_SCIF_SCIF3_ERI3,
// 254 SCIF SCIF3 RXI3
    INT_SCIF_SCIF3_RXI3,
// 255 SCIF SCIF3 TXI3
    INT_SCIF_SCIF3_TXI3,
// xx Reserved
    Dummy
};

/* End of File */
```

- Sample Program Listing: vect.h (1)

```
/*""FILE COMMENT""***************************************************************************
*
*   System Name: SH7206 Sample Program
*   File Name  : vect.h
*   Version    : 1.00.00
*   Contents   : Interrupt processing function prototype definition
*   Model      : M3A-HS60
*   CPU        : SH7206
*   Compiler   : SHC9.0.00
*   OS         : none
*
*   note       : This file was originally generated by Renesas Project Generator (Ver.3.1)
*              : and was modified for the application note.
*
*                 <Caution>
*                This sample program is for reference
*                and its operation is not guaranteed.
*                Customers should use this sample program for technical reference
*                in software development.
*
*
*   This file is generated by Renesas Project Generator (Ver.3.1).
*
*   Copyright (C) 2004 Renesas Technology Corp. All Rights Reserved
*   AND Renesas Solutions Corp. All Rights Reserved
*
*   history    : 2004.10.14 ver.1.00.00
*""FILE COMMENT END""***********************************************************************/
#ifndef VECT_H
#define VECT_H

//;<<VECTOR DATA START (POWER ON RESET)>>
//;0 Power On Reset PC
extern void PowerON_Reset_PC(void);

//;<<VECTOR DATA END (POWER ON RESET)>>
// 1 Power On Reset SP

<<Omitted>>
// 252 SCIF SCIF3 BRI3
#pragma interrupt INT_SCIF_SCIF3_BRI3(resbank)
extern void INT_SCIF_SCIF3_BRI3(void);

// 253 SCIF SCIF3 ERI3
#pragma interrupt INT_SCIF_SCIF3_ERI3(resbank)
extern void INT_SCIF_SCIF3_ERI3(void);

// 254 SCIF SCIF3 RXI3
#pragma interrupt INT_SCIF_SCIF3_RXI3(resbank)
extern void INT_SCIF_SCIF3_RXI3(void);

// 255 SCIF SCIF3 TXI3
#pragma interrupt INT_SCIF_SCIF3_TXI3(resbank)
extern void INT_SCIF_SCIF3_TXI3(void);

// Dummy
#pragma interrupt Dummy(resbank)
extern void Dummy(void);

#endif /* VECT_H */

/* End of File */
```

4. Reference Documents

- SH-2A SH2A-FPU Software Manual (Rev.3.00)
  (Download the latest edition from the website of Renesas Technology Corp.)

- SH7206 Group Hardware Manual (Rev. 1.00)
  (Download the latest edition from the website of Renesas Technology Corp.)

5. Website

- Website of Renesas Technology Corp.
  http://www.renesas.com/

Revision Record

Rev.	Date	Description	
		Page	Summary
1.00	Sep.14.05	—	First edition issued

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.