# SH7262/SH7264 Group

## Configuring Asynchronous Mode Using the Serial Communication Interface with FIFO and Direct Memory Access Controller

R01AN0215EJ0100
Rev. 1.00
Dec. 28, 2010

## Summary

This application note describes an example to transmit or receive data using the Serial Communication Interface with FIFO (SCIF) asynchronous mode, and store the data in the on-chip RAM using the Direct Memory Access Controller.

## Target Device

SH7262/SH7264 (In this document, SH7262/SH7264 are described as SH7264.)

## Contents

# 1. Introduction

## 1.1 Specifications

- Uses the Serial Communication Interface with FIFO (SCIF) channel 0
- Configures the SCIF in asynchronous mode to transmit or receive data
- Uses the Direct Memory Access Controller (DMAC) channel 4 to transfer the transmit data, and uses channel 5 to transfer the receive data

## 1.2 Modules Used

- Serial Communication Interface with FIFO (SCIF)
- Direct Memory Access Controller (DMAC)

## 1.3 Applicable Conditions

| | |
|---|---|
| MCU | SH7262/SH7264 |
| Operating Frequency | Internal clock: 144 MHz<br>Bus clock: 72 MHz<br>Peripheral clock: 36 MHz |
| Integrated Development Environment | Renesas Electronics Corporation<br>High-performance Embedded Workshop Ver.4.07.00 |
| C Compiler | Renesas Electronics SuperH RISC engine Family<br>C/C++ compiler package Ver.9.03 Release 00 |
| Compiler Options | Default setting in the High-performance Embedded Workshop<br>(-cpu=sh2afpu -fpu=single -debug -gbr=auto -global_volatile=0<br>-opt_range=all -infinite_loop=0 -del_vacant_loop=0 -struct_alloc=1) |

## 1.4 Related Application Notes

For more information, refer to the following application notes:

- SH7262/SH7264 Group Example of Initialization
- SH7262/SH7264 Group Serial Communication Interface with FIFO, Configuration to Transmit Strings in Asynchronous Mode
- SH7262/SH7264 Group Serial Communication Interface with FIFO, Configuration to Receive Strings in Asynchronous Mode
- SH7262/SH7264 Group Serial Communication Interface with FIFO, Configuring the Serial Communication in Clock Synchronous Mode (Full-duplex)

## 1.5 About Active-low Pins (Signals)

The symbol "#" suffixed to the pin (or signal) names indicates that the pins (or signals) are active-low.

## 1.6 Hardware Conditions

### 1.6.1 Platform

This demo code was test on M3A-HS64G50(CPU board), referenced in the source code, which uses SCIF channel 0. To run this on the SH7264 RSK, the code needs to be converted to use SCIF channel 3. This includes the pin configuration that is part of the io_init_scif0_dma() function and source for the DMAC triggers set in the io_init_dmac4() and io_init_dmac5() functions. Please refer to the notes in the flow chart in figure 3 for which STANDBY bit to clear when using SCIF3.

### 1.6.2 Serial Connections

The demo runs to completion only if it gets the required number of receive bytes. It will run to completion if a loopback is installed on the serial channel being used OR if the serial port is connected to a terminal and the correct number of bytes are typed in. Evaluate the hardware in use and connect TXD / RXD as necessary on the serial connector in use to fully exercise the demo code.

## 2. Applications

This application uses the Serial Communication Interface with FIFO (SCIF), and uses the Direct Memory Access Controller (DMAC) to transfer the transmit/receive data.

### 2.1 Overview of Modules

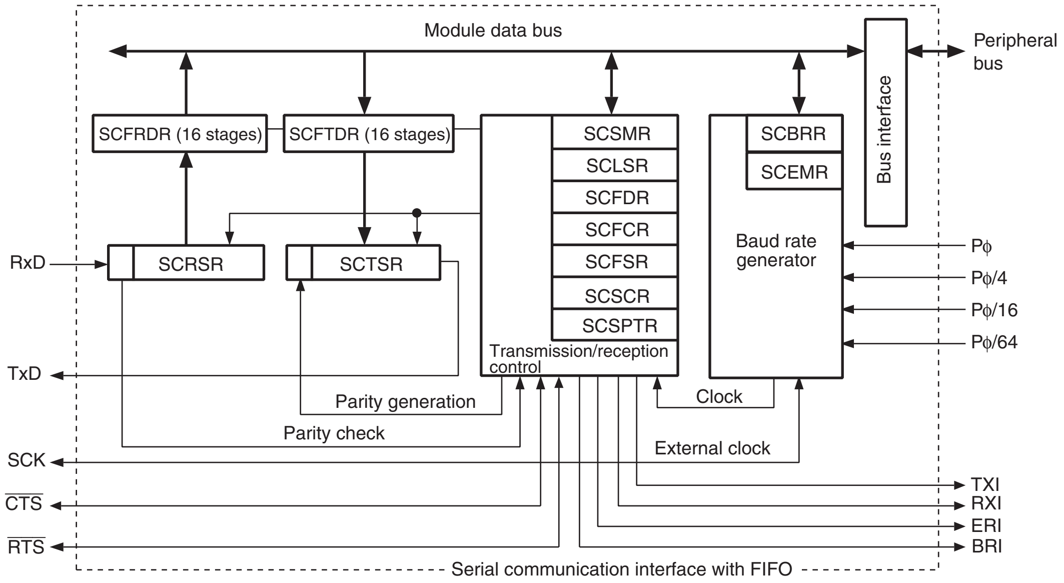
(1) Serial Communication Interface with FIFO (SCIF)

The SH7264 SCIF transmits or receives a "character", appending a start bit which indicates the initiation of the communication, and a stop bit which indicates the end of the communication to data. Then, the SH7264 SCIF handles communication in sync per character. The internal clock or external clock from the SCK pin can be specified as the clock source Transfer data format and baud rate can be set in the SCIF.

Table 1 lists the overview of asynchronous mode. Figure 1 shows the SCIF block diagram. For more information about the SCIF, refer to the Serial Communication Interface with FIFO chapter in the SH7262 Group, SH7264 Group Hardware Manual.

**Table 1 SCIF (Asynchronous Mode) Overview**

| Item | Description |
|---|---|
| Number of channels | 8 (SCIF0 to SCIF7) |
| Clock source | Internal clock: Pφ, Pφ/4, Pφ/16, Pφ/64 P (Pφ: internal peripheral clock)<br>External clock: SCK0 to SCK3 pin input clocks<br>(The pin input divided by 16 or 8 is selected as the SCIF operating clock.) |
| Data format | Transfer data length: 7-bit or 8-bit<br>Order of transfer: LSB first fixed<br>Start bit: 1-bit fixed<br>Stop bit: 1-bit or 2-bit<br>Parity bit: even parity, odd parity, or no parity |
| Baud rate | When specifying the internal clock: 68.66 bps to 4500 kbps (Pφ is at 36 MHz)<br>When specifying the external clock: up to 1125 kbps (Pφ is at 36 MHz, external clock is at 9 MHz) |
| Error detection | Parity error, framing error, overrun error |
| Interrupt request | Transmit-FIFO-data-empty interrupt (TXI) by the transmit FIFO data empty (TDFE)<br>Break interrupt (BRI) by the break (BRK) or overrun error (ORER)<br>Receive FIFO data full (RXI) by the Receive FIFO data full (RDF) or data ready (DR)<br>Receive-error interrupt (ERI) by the receive error (ER) |
| Other | • Break can be detected<br>• Supplying clock unused channels can be stopped to reduce power consumption<br>• Includes the modem control functions (RTS and CTS) (Only channels 1 and 3 for the SH7264, only channel 1 for the SH7262)<br>• The number of valid data stored in the Transmit and Receive FIFO data registers, and the number of receive errors stored in the Receive FIFO data register can be detected<br>• Time out error (DR) on reception can be detected<br>• Base clock frequency can be either 16 or 8 times the bit rate<br>• Double-speed mode can be specified for the baud rate generated (When not using the SCK pin) |



[Legend]

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

**Figure 1 SCIF Block Diagram**

(2) Direct Memory Access Controller (DMAC)

The DMAC transfers data among an external device with DACK (transfer request acknowledge signal), and external memory, internal memory, memory-mapped external device, and on-chip peripheral modules, instead of the CPU. It has two bus modes; cycle steal mode and burst mode.

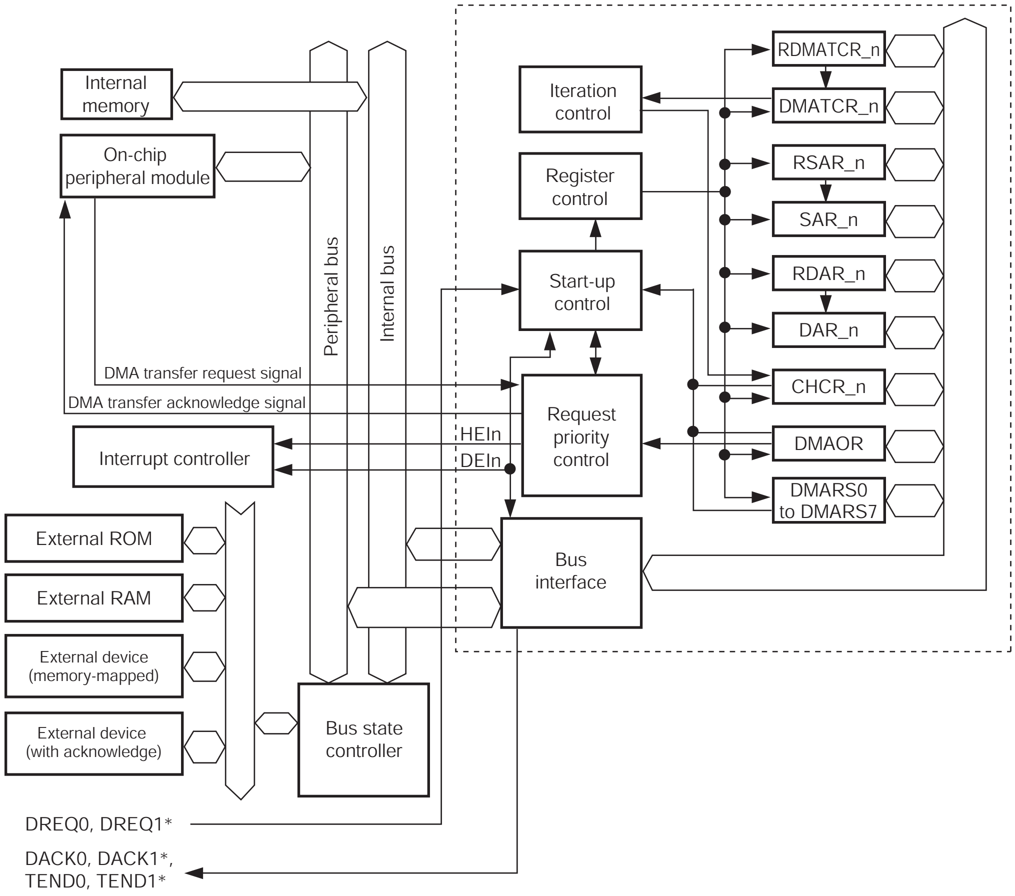
In cycle steal mode, the DMAC leaves the bus to the other masters when it finishes "a transmit" (in bytes, words, long words, or 16 bytes). When the DMAC receives another transfer request, it retrieves the bus again. Then, it transfers data in unit of a transfer, and leaves the bus again to the other bus. The DMAC repeats this operation until the transfer end conditions are satisfied.

This application transfers the transmit/receive data at SCIF channel 0 using cycle steal mode. Set the transfer source of the transmit data and destination of the receive data to the high-speed internal RAM.

Table 2 lists the features of the DMAC. Figure 2 shows its block diagram. For more information, refer to the Direct Memory Access Controller chapter in the SH7262 Group, SH7264 Group Hardware Manual.

**Table 2 DMAC Features**

| Item | Description |
|---|---|
| Number of channels | 16 |
| Address space | 4 GB physically |
| Transfer data length | Byte, word, long word, and 16 bytes |
| Number of transfers | 16,777,216 (24-bit) times |
| Address mode | Single address mode, dual address mode |
| Transfer request | External request, on-chip peripheral module request, auto-request |
| Bus mode | Cycle steal mode (normal mode and intermittent mode)<br>Burst mode |
| Interrupt source | One-half of the data transfer completed, a data transfer completed |
| Reload function | DMA transfer using the same setting as the current DMA transfer can be repeated automatically without specifying the setting again. Specify the reload register during the DMA transfer to execute the next DMA transfer with another setting. The reload function can be enabled or disabled per channel, and reload register. |



[Legend]

| | | | |
|---|---|---|---|
| RDMATCR: | DMA reload transfer count register | CHCR: | DMA channel control register |
| DMATCR: | DMA transfer count register | DMAOR: | DMA operation register |
| RSAR: | DMA reload source address register | DMARS0 to DMARS7: | DMA extension resource selectors 0 to 7 |
| SAR: | DMA source address register | HEIn: | DMA transfer half-end interrupt request to the CPU |
| RDAR: | DMA reload destination address register | DEIn: | DMA transfer end interrupt request to the CPU |
| DAR: | DMA destination address register | n = 0 to 15 | |

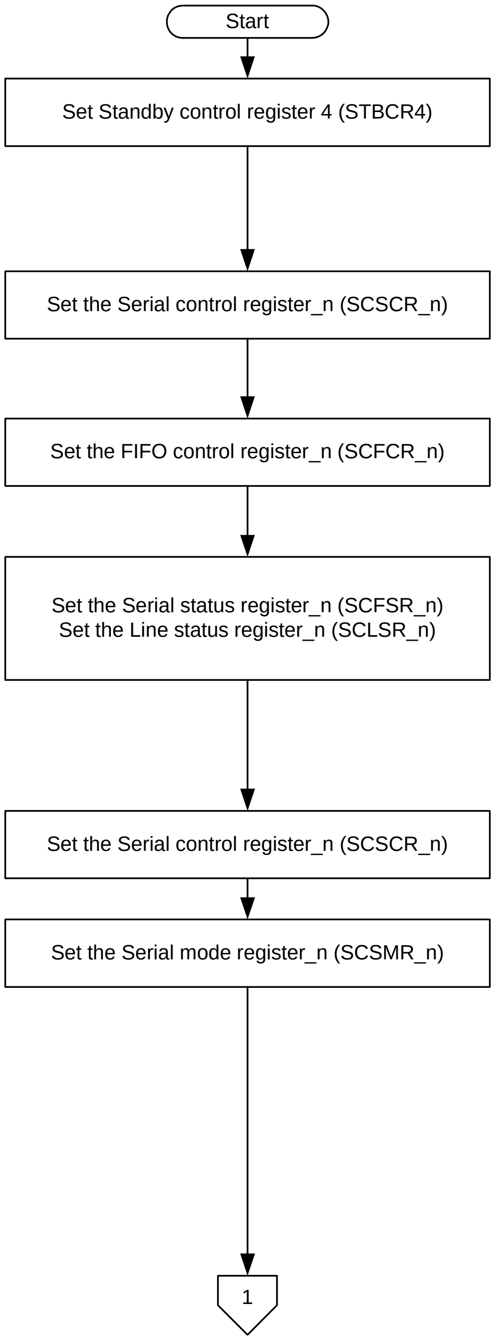
Note: * Pins in channel 1 can be used only in the SH7264 Group.

**Figure 2 DMAC Block Diagram**

## 2.2 Configuration Procedure

(1) Steps to configure the SCIF

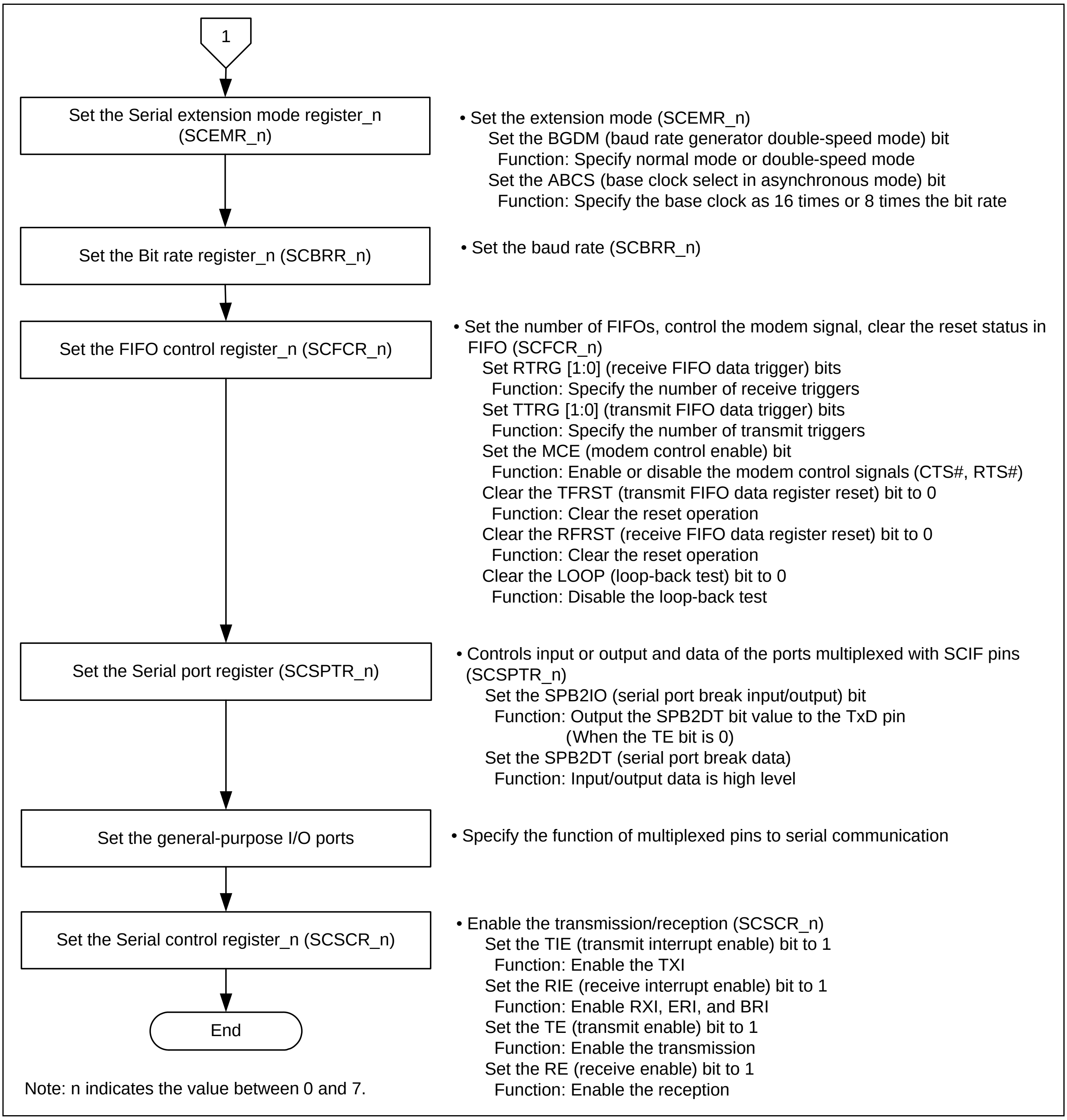
This section describes how to configure the communication in the SH7264 SCIF asynchronous mode. Figure 3 and Figure 4 show flow charts for configuring the transmission/reception in asynchronous mode.


Start
Set Standby control register 4 (STBCR4)
Set the Serial control register_n (SCSCR_n)
Set the FIFO control register_n (SCFCR_n)
Set the Serial status register_n (SCFSR_n)
Set the Line status register_n (SCLSR_n)
Set the Serial control register_n (SCSCR_n)
Set the Serial mode register_n (SCSMR_n)
1

- Enable supplying the clock to SCIFn (STBCR4)
  When using SCIF0: Clear MSTP47 bit to 0
  When using SCIF1: Clear MSTP46 bit to 0
  When using SCIF2: Clear MSTP45 bit to 0
  When using SCIF3: Clear MSTP44 bit to 0
  When using SCIF4: Clear MSTP43 bit to 0
  When using SCIF5: Clear MSTP42 bit to 0
  When using SCIF6: Clear MSTP41 bit to 0
  When using SCIF7: Cleart MSTP40 bit to 0
  Function: Supply the clock to SCIFn
- Disable the transmitter (SCSCR_n)
  Clear the TE (transmit enable) bit to 0
  Function: Disable the transmission
  Clear the RE (receive enable) bit to 0
  Function: Disable the reception
- Disable the data in FIFO (SCFCR_n)
  Set the TFRST (transmit FIFO data register reset) bit to 1
  Function: Disable the transmit data
  Set the RFRST (receive FIFO data register reset) bit to 1
  Function: Disable the receive data
- Clear the error status (SCFSR_n, SCLSR_n)
  Set 0 to each bit after reading the bit:
  Clear the ER (receive error) bit to 0
  Function: Clear the receive error flag
  Clear the DR (receive data ready) bit to 0
  Function: Clear the receive data ready flag
  Clear the BRK (break detection) bit to 0
  Function: Clear the break detection flag
  Clear the ORER (overrun error) bit to 0
  Function: Clear the overrun error flag
- Specify the clock source (SCSCR_n)
  Set CKE [1:0] (clock enable) bits
  Function: Specify the clock source
- Set the serial communication format (SCSMR_n)
  Set the C/A# (communication mode) bit to 0
  Function: Specify asynchronous mode
  Set the CHR (character length) bit
  Function: Specify the character length
  Set the PE (parity enable) bit
  Function: Specify whether to add and check the parity bit
  Set the O/E# (parity mode) bit
  Function: Specify the even parity or odd parity
  Set the STOP (stop bit length) bit
  Function: Specify the stop bit length
  Set CKS [1:0] (clock select) bits
  Function: Specify the internal clock source

Note: n indicates the value between 0 and 7.

**Figure 3 Flow Chart for Configuring the Transmission in Asynchronous Mode (1/2)**


1
Set the Serial extension mode register_n (SCEMR_n)
Set the Bit rate register_n (SCBRR_n)
Set the FIFO control register_n (SCFCR_n)
Set the Serial port register (SCSPTR_n)
Set the general-purpose I/O ports
Set the Serial control register_n (SCSCR_n)
End

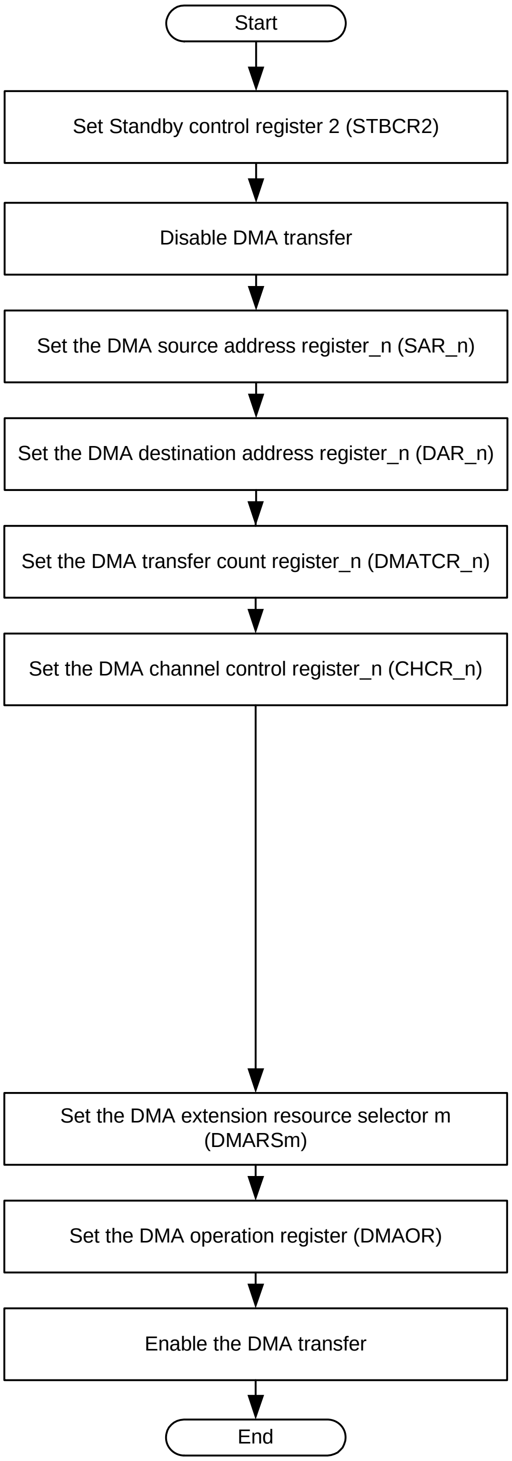
- Set the extension mode (SCEMR_n)
  Set the BGDM (baud rate generator double-speed mode) bit
  Function: Specify normal mode or double-speed mode
  Set the ABCS (base clock select in asynchronous mode) bit
  Function: Specify the base clock as 16 times or 8 times the bit rate
- Set the baud rate (SCBRR_n)
- Set the number of FIFOs, control the modem signal, clear the reset status in FIFO (SCFCR_n)
  Set RTRG [1:0] (receive FIFO data trigger) bits
  Function: Specify the number of receive triggers
  Set TTRG [1:0] (transmit FIFO data trigger) bits
  Function: Specify the number of transmit triggers
  Set the MCE (modem control enable) bit
  Function: Enable or disable the modem control signals (CTS#, RTS#)
  Clear the TFRST (transmit FIFO data register reset) bit to 0
  Function: Clear the reset operation
  Clear the RFRST (receive FIFO data register reset) bit to 0
  Function: Clear the reset operation
  Clear the LOOP (loop-back test) bit to 0
  Function: Disable the loop-back test
- Controls input or output and data of the ports multiplexed with SCIF pins (SCSPTR_n)
  Set the SPB2IO (serial port break input/output) bit
  Function: Output the SPB2DT bit value to the TxD pin
  (When the TE bit is 0)
  Set the SPB2DT (serial port break data)
  Function: Input/output data is high level
- Specify the function of multiplexed pins to serial communication
- Enable the transmission/reception (SCSCR_n)
  Set the TIE (transmit interrupt enable) bit to 1
  Function: Enable the TXI
  Set the RIE (receive interrupt enable) bit to 1
  Function: Enable RXI, ERI, and BRI
  Set the TE (transmit enable) bit to 1
  Function: Enable the transmission
  Set the RE (receive enable) bit to 1
  Function: Enable the reception

Note: n indicates the value between 0 and 7.

**Figure 4 Flow Chart for Configuring the Transmission in Asynchronous Mode (2/2)**

(2) Steps to configure the DMAC

When using the transmit FIFO data empty interrupt (TXI) or receive FIFO data full interrupt (RXI) of the SCIF as an interrupt source, only cycle steal mode can be specified. Figure 5 shows the flow chart for configuring the DMAC. For more information, refer to the SH7262 Group, SH7264 Group Hardware Manual.


Start
Set Standby control register 2 (STBCR2)
Disable DMA transfer
Set the DMA source address register_n (SAR_n)
Set the DMA destination address register_n (DAR_n)
Set the DMA transfer count register_n (DMATCR_n)
Set the DMA channel control register_n (CHCR_n)
Set the DMA extension resource selector m (DMARSm)
Set the DMA operation register (DMAOR)
Enable the DMA transfer
End

- Enable supplying the clock to the DMAC
  Set MSTP8 (module stop bit 8) bit to 0
  Function: Supply the clock to the DMAC

- Disable the DMA transfer
  Clear the DE (DMA enable) bit in the DMA channel control register_n (CHCR_n)
  Function: Disable the DMA transfer

- Set the DMA source address register_n (SAR_n)
  Set the transfer source start address

- Set the DMA destination address register_n （DAR_n)
  Set the transfer destination start address

- Set the DMA transfer count register_n (DMATCR_n)
  Set the number of transfers

- Set the DMA channel control register_n (CHCR_n)
  Set the TC (transfer count mode) bit to 0
  Function: Execute a transfer by a transfer request
  Set RLDSAR, RLDDAR (SAR, DAR reload function ON or OFF) bits
  Function: Enable or disable the reload function
  Set the DM (destination address mode) bit
  Function: Increment or decrement the destination address
  Set the SM (source address mode) bit
  Function: Increment or decrement the source address
  Set the RS (resource select) bit to "B'1000"
  Function: Set the transfer request source to the DMA extension resource selector
  Set the TB (transfer bus mode) bit to 0
  Function: Specify cycle steal mode
  Set the TS (transfer size) bit to "B'00"
  Function: Specify to transfer data in bytes
  Set the IE (interrupt enable) bit
  Function: Enable or disable the interrupt request

- Set the DMA extension resource selector m (DMARSm)
  Function: Specify the transfer request source as SCIF channel 0 transmit or receive

- Set the DMA operation register (DMAOR)
  Set the DME (DMA master enable) bit to 1
  Function: Enable the DMA transfer on all channels

- Enable the DMA transfer
  Set the DE (DMA enable) bit in the DMA channel control register_n (CHCR_n)
  Function: Enable the DMA transfer

Notes:
n indicates the value between 0 and 15. m indicates the value between 0 and 7.

**Figure 5 Flow Chart for Configuring the DMAC**

## 2.3 Sample Program Operation

This sample program uses SCIF channel 0 in asynchronous mode to transmit and receive data. It also uses DMAC channel 4 to transfer the transmit data, DMAC channel 5 to transfer the receive data.

Activate the DMAC by the transmit FIFO data empty interrupt (TXI) to transfer the transmit data from the high-speed internal RAM. Note that the interrupt does not occur to the CPU. The transmit FIFO data empty flag (TDFE) is automatically cleared by activating the DMAC. As the state of the transmit end flag (TEND) is undefined, it cannot be use as the transfer end flag.

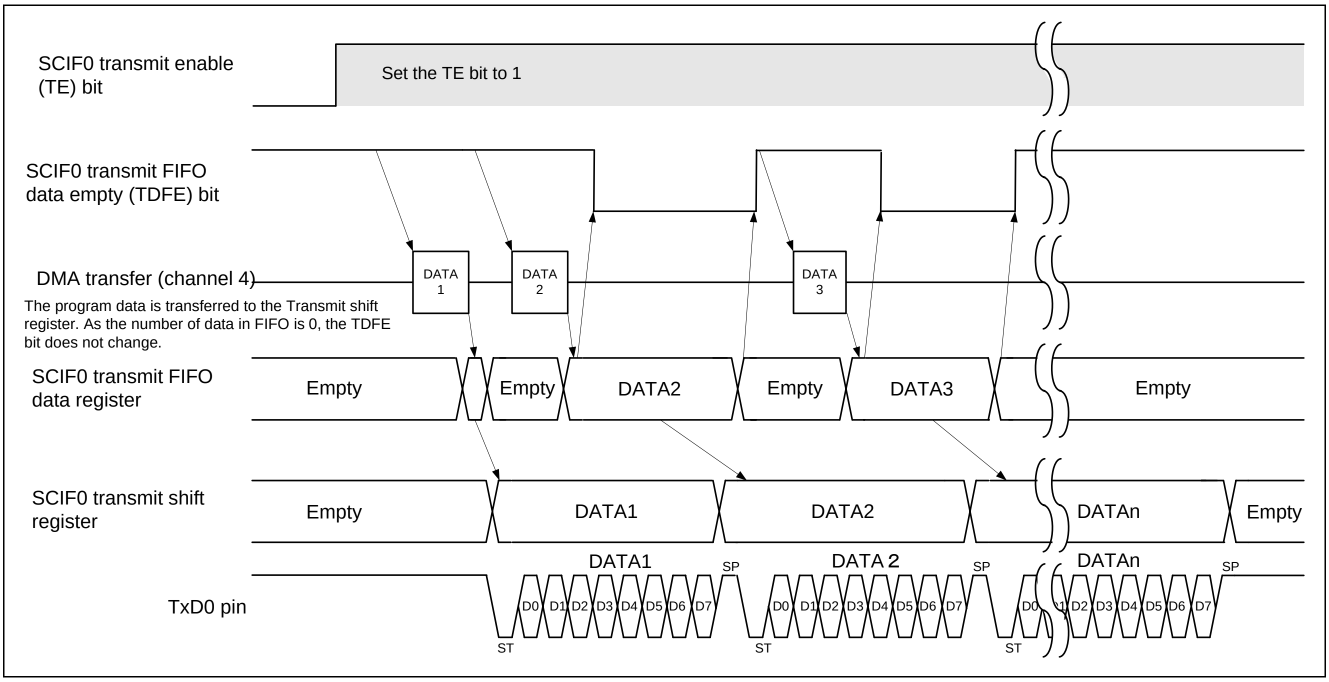
Activate the DMAC again by the receive FIFO data full or data ready interrupt (RXI) to transfer the receive data to the high-speed internal RAM. Note that the interrupt does not occur to the CPU. The receive FIFO data full flag (RDF) and receive data ready flag (DR) are automatically cleared by activating the DMAC.

When specifying the cache-enabled area as the transfer destination, use the software as appropriate to maintain the coherency between the cache and memory.

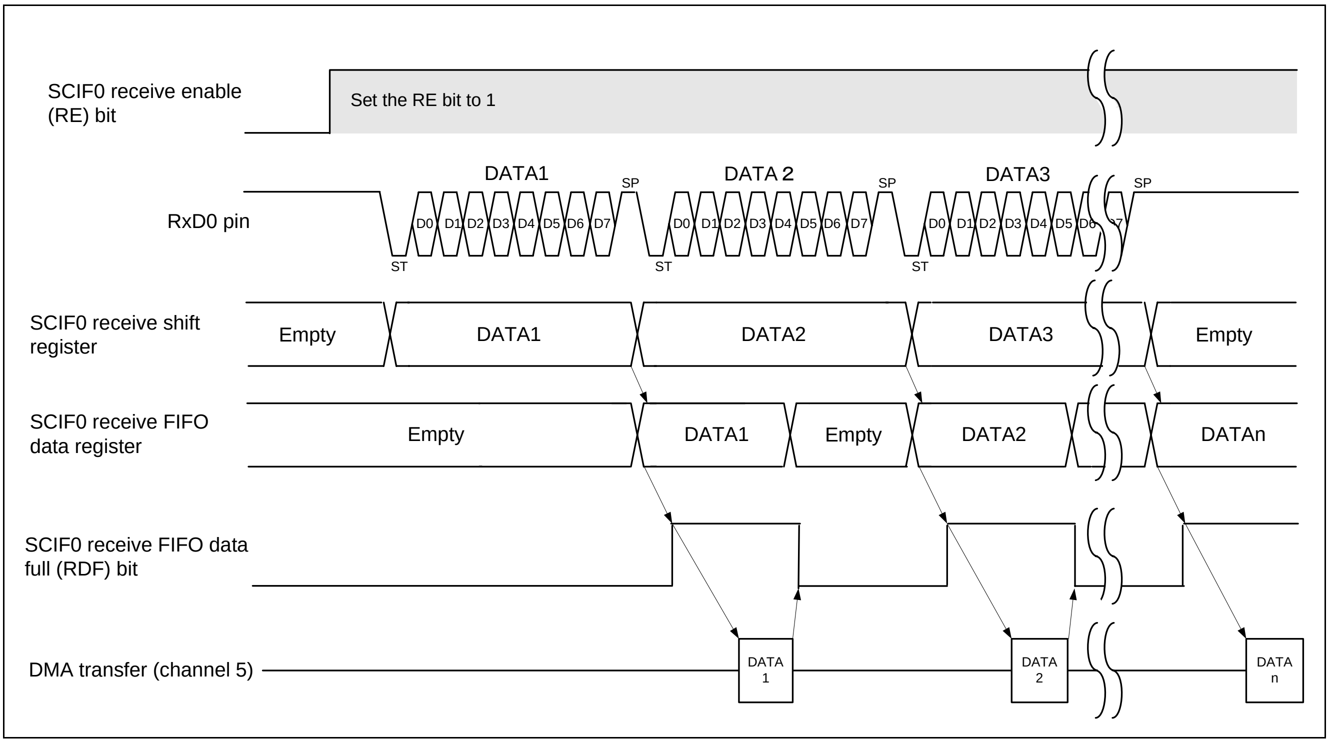
Table 3 lists the communication settings in the sample program. Figure 6 and Figure 7 show the operation timing of the sample program.

**Table 3 Communication Settings**

| Communication Format | Description |
|---|---|
| Communication mode | Asynchronous mode |
| Number of channel to use | Channel 0 |
| Interrupt | DMAC activated by the transmit FIFO data empty interrupt (TXI)<br>DMAC activated by the receive FIFO data full or data ready interrupt (RXI) |
| Baud rate | 19200 bps |
| Data length | 8-bit |
| Parity | No parity |
| Stop bit | 1 stop bit |
| Modem control | RTS/CTS functions are disabled |
| Bit order | LSB first |
| Number of FIFO data triggers | Transmit: 0 (Number of data not transmitted is 0 and the transmit FIFO data empty)<br>Receive: 1 (Number of data received is more than 1 and receive FIFO data full) |


SCIF0 transmit enable (TE) bit
Set the TE bit to 1
SCIF0 transmit FIFO data empty (TDFE) bit
DMA transfer (channel 4)
DATA 1
DATA 2
DATA 3
The program data is transferred to the Transmit shift register. As the number of data in FIFO is 0, the TDFE bit does not change.
SCIF0 transmit FIFO data register
Empty
Empty
DATA2
Empty
DATA3
Empty
SCIF0 transmit shift register
Empty
DATA1
DATA2
DATAn
Empty
TxD0 pin
DATA1
DATA 2
DATAn
SP
ST
D0 D1 D2 D3 D4 D5 D6 D7

## Figure 6 Transmission Timing in the Sample Program



## Figure 7 Reception Timing in the Sample Program

## 2.4 Sample Program Procedure

The sample program initializes SCIF channel 0, DMAC channels 4 and 5. After initialized, DMAC channel 4 transfers the transmit data in unit of 1-byte, and DMAC channel 5 transfers the receive data. Total number of bytes of transmit and receive data is 16 bytes.

Table 4 lists SCIF channel 0 register setting in the sample program. Table 5 and Table 6 list DMAC channels 4 and 5 register settings. Figure 8 shows the sample program flow chart.

**Table 4 SCIF Register Setting**

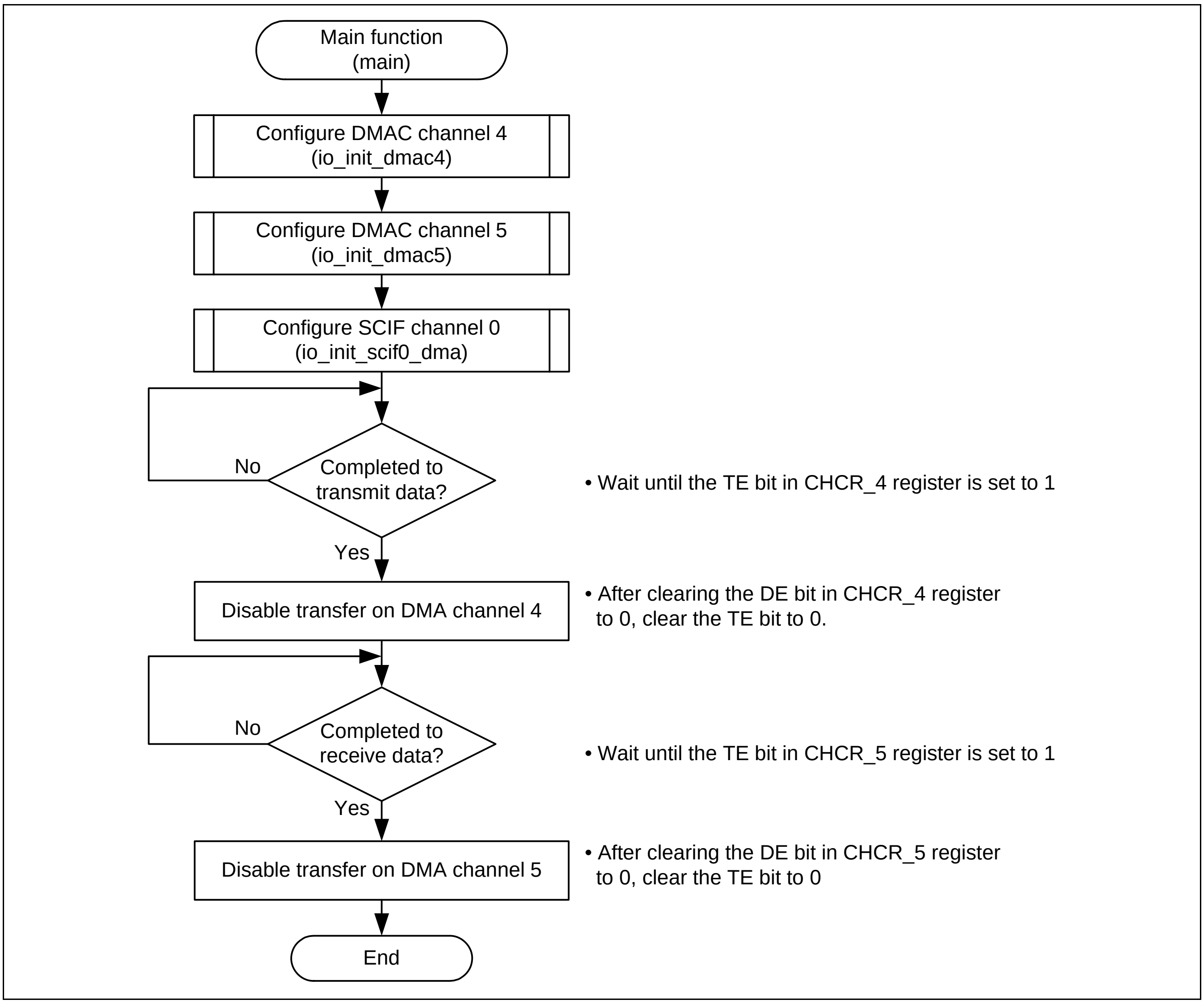
| Register Name | Address | Setting | Description |
|---|---|---|---|
| Standby control register 4 (STBCR4) | H'FFFE 040C | H'00 | • MSTP47 = "0": SCIF0 is operating (supplies the clock) |
| Port J control register 0 (PJCR0) | H'FFFE 390E | H'0044 | • PJ0MD = "B'100": TxD0 output (SCIF0)<br>• PJ1MD = "B'100": RxD0 output (SCIF0) |
| Serial control register_0 (SCSCR_0) | H'FFFE 8008 | H'0000 | • TE = "0": Disables the transmission<br>• RE = "0" Disables the reception<br>• CKE [1:0] = "B'00": Internal clock/SCK pin is an input pin |
| | | H'00F0 | • TIE = "1": Enables the TXI<br>• RIE = "1": Enables the RXI, ERI, and BRI<br>• TE = "1": Enables the transmission<br>• RE = "1": Enables the reception |
| FIFO control register_0 (SCFCR_0) | H'FFFE 8018 | H'0006 | • TFRST = "1": Enables to reset the transmit FIFO data register<br>• RFRST = "1": Enables to reset the receive FIFO data register |
| | | H'0030 | • RTRG [1:0] = "B'00": Sets the RDF flag when the number of data in the receive FIFO is equal to or more than 1<br>• TTRG [1:0] = "B'11": Sets the TDFE flag when the number of data in the transmit FIFO is equal to or less than 0<br>• MCE = "0": Disables the modem signal<br>• TFRST = "0": Disables to reset the transmit FIFO data register<br>• RFRST = "0": Disables to reset the receive FIFO data register<br>• LOOP = "0": Disables the loop-back test |
| Serial mode register_0 (SCSMR_0) | H'FFFE 8000 | H'0000 | • C/A# = "0": Asynchronous mode<br>• CHR = "0": 8-bit data<br>• PE = "0": Disables the parity bit<br>• STOP = "0": 1 stop bit<br>• CKS [1:0] = "B'00": Peripheral clock (no division) |
| Serial extension mode register_0 (SCEMR_0) | H'FFFE 8028 | H'0000 | • BGDM = "0": Normal mode<br>• ABCS = "0": Base clock frequency is 16 times the bit rate |
| Bit rate register_0 (SCBRR_0) | H'FFFE 8004 | H'3A | • Sets the bit rate to 19200 bps (Error: −0.69% when Pφ is 36 MHz) |
| Serial port register_0 (SCSPTR_0) | H'FFFE 8020 | H'0053 | • SPB2IO = "1": Outputs the SPB2DT bit value to the TxD pin<br>• SPB2DT = "1": Input/output data is at high level |

**Table 5 DMAC Channel 4 Register Setting**

| Register Name | Address | Setting | Description |
|---|---|---|---|
| Standby control register 2 (STBCR2) | H'FFFE 0018 | H'00 | • MSTP8 = "0": DMAC is operating |
| DMA source address register_4 (SAR_4) | H'FFFE 1040 | &snd_data[0] | • Transfer source start address: high-speed internal RAM |
| DMA destination address register_4 (DAR_4) | H'FFFE 1044 | H'FFFE 800C | • Transfer destination start address: transmit FIFO data register_0 |
| DMA transfer count register_4 (DMATCR_4) | H'FFFE 1048 | H'0000 0010 | • Number of transfers: 16 |
| DMA channel control register_4 (CHCR_4) | H'FFFE 104C | H'0000 0000 | • DE = "0": Disables the DMA transfer |
| | | H'0000 1800 | • TC = "0":  Executes a transfer by a DMA transfer request<br>• RLDSAR = "0": Disables the source address reload function<br>• RLDDAR = "0": Disables the destination address reload function<br>• DM = "B'00": Destination address is fixed<br>• SM = "B'01: Increments the source address<br>• RS = "B'1000": Specifies the extension resource selector as the transfer request source<br>• TB = "0": Specifies cycle steal mode<br>• TS = "B'00": Specifies to transfer data in bytes<br>• IE = "0": Disables the interrupt request |
| | | H'0000 1801 | • DE = "1": Enables the DMA transfer |
| DMA extension resource selector 2 (DMARS2) | H'FFFE 1308 | H'8281 | • Channel 4 transfer request source is the transmission at SCIF0 |
| DMA operation register (DMAOR) | H'FFFE 1200 | H'0001 | • DME = "1": Enables the DMA transfer on all channels |

**Table 6 DMAC Channel 5 Register Setting**

| Register Name | Address | Setting | Description |
|---|---|---|---|
| Standby control register 2 (STBCR2) | H'FFFE 0018 | H'00 | • MSTP8 = "0": DMAC is operating |
| DMA source address register_5 (SAR_5) | H'FFFE 1050 | H'FFFE 8014 | • Transfer source start address: receive FIFO data register_0 |
| DMA destination address register_5 (DAR_5) | H'FFFE 1054 | &rcv_data[0] | • Transfer destination start address: high-speed internal RAM |
| DMA transfer count register_5 (DMATCR_5) | H'FFFE 1058 | H'0000 0010 | • Number of transfers: 16 |
| DMA channel control register_5 (CHCR_5) | H'FFFE 105C | H'0000 0000 | • DE = "0": Disables the DMA transfer |
| | | H'0000 4800 | • TC = "0":  Executes a transfer by a DMA transfer request<br>• RLDSAR = "0": Disables the source address reload function<br>• RLDDAR = "0": Disables the destination address reload function<br>• DM = "B'01": Increments the destination address<br>• SM = "B'01: Source address is fixed<br>• RS = "B'1000": Specifies the extension resource selector as the transfer request source<br>• TB = "0": Specifies cycle steal mode<br>• TS = "B'00": Specifies to transfer data in bytes<br>• IE = "0": Disables the interrupt request |
| | | H'0000 4801 | • DE = "1": Enables the DMA transfer |
| DMA extension resource selector 2 (DMARS2) | H'FFFE 1308 | H'8281 | • Channel 5 transfer request source is the reception at SCIF0 |
| DMA operation register (DMAOR) | H'FFFE 1200 | H'0001 | • DME = "1": Enables the DMA transfer on all channels |



- Wait until the TE bit in CHCR_4 register is set to 1
- After clearing the DE bit in CHCR_4 register to 0, clear the TE bit to 0.
- Wait until the TE bit in CHCR_5 register is set to 1
- After clearing the DE bit in CHCR_5 register to 0, clear the TE bit to 0

**Figure 8 Sample Program Flow Chart**

# 3. Sample Program Listing

## 3.1 Supplement to the Sample Program

As the capacity of the SH7264 large-capacity internal RAM varies as 1 MB or 640 KB, depending on the MCU type, the section alignment and register setting must be partly altered. To support both MCU types, this application note provides two types of sample programs (workspaces) for 1-MB RAM and 640-KB RAM.

As the MCU with 640-KB RAM must be write-enabled before writing data in the data-retention RAM, the System control register 5 (SYSCR5) is set to write-enable the RAM in the sample program for 640-KB RAM.

Review your product and use the appropriate workspace.

## 3.2 Sample Program List "main.c" (1/9)

```
/********************************************************************************
*   DISCLAIMER
*
*   This software is supplied by Renesas Electronics Corporation and is only
*   intended for use with Renesas products.  No other uses are authorized.
*
*   This software is owned by Renesas Electronics Corporation and is protected under
*   all applicable laws, including copyright laws.
*
*   THIS SOFTWARE IS PROVIDED "AS IS" AND RENESAS MAKES NO WARRANTIES
*   REGARDING THIS SOFTWARE, WHETHER EXPRESS, IMPLIED OR STATUTORY,
*   INCLUDING BUT NOT LIMITED TO WARRANTIES OF MERCHANTABILITY, FITNESS FOR A
*   PARTICULAR PURPOSE AND NON-INFRINGEMENT.  ALL SUCH WARRANTIES ARE EXPRESSLY
*   DISCLAIMED.
*
*   TO THE MAXIMUM EXTENT PERMITTED NOT PROHIBITED BY LAW, NEITHER RENESAS
*   ELECTRONICS CORPORATION NOR ANY OF ITS AFFILIATED COMPANIES SHALL BE LIABLE
*   FOR ANY DIRECT, INDIRECT, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES
*   FOR ANY REASON RELATED TO THIS SOFTWARE, EVEN IF RENESAS OR ITS
*   AFFILIATES HAVE BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.
*
*   Renesas reserves the right, without notice, to make changes to this
*   software and to discontinue the availability of this software.
*   By using this software, you agree to the additional terms and
*   conditions found by accessing the following link:
*   http://www.renesas.com/disclaimer
*********************************************************************************
*   Copyright (C) 2010 Renesas Electronics Corporation. All rights reserved.
*""FILE COMMENT""*********** Technical reference data ***************************
*   System Name : SH7264 Sample Program
*   File Name   : main.c
*   Abstract    : Configuring Asynchronous Mode Using the Serial Communication
*               : Interface with FIFO and Direct Memory Access Controller
*   Version     : 1.00.00
*   Device      : SH7262/SH7264
*   Tool-Chain  : High-performance Embedded Workshop (Ver.4.07.00).
*               : C/C++ compiler package for the SuperH RISC engine family
*               :                             (Ver.9.03 Release00).
*   OS          : None
*   H/W Platform: M3A-HS64G50(CPU board)
*   Description :
*********************************************************************************
*   History     : Sep.06,2010 ver.1.00.00
*""FILE COMMENT END""***********************************************************/
#include "iodefine.h"       /* SH7264 iodefine  */

/* ==== Prototype declaration ==== */
void main(void);
void io_init_scif0_dma(int bps);
void io_init_dmac4(void *src, void *dst, int count);
void io_init_dmac5(void *src, void *dst, int count);

```

## 3.3 Sample Program List "main.c" (2/9)

```
/* ==== Type definition ==== */
/* SCIF baud rate setting */
typedef struct  {
  unsigned char scbrr;    /* SCBRR register setting */
  unsigned short scsmr;       /* SCSMR register setting */
} SH7264_BAUD_SET;

/* ---- Baud rate specified value ---- */
enum{
  CBR_1200,
  CBR_2400,
  CBR_4800,
  CBR_9600,
  CBR_19200,
  CBR_31250,
  CBR_38400,
  CBR_57600,
  CBR_115200
};

/* ==== Register setting table (P clock = 36 MHz) ==== */
static SH7264_BAUD_SET scif_baud[] = {
  {233, 1},    /*  1200 bps (error: 0.16%) */
  {116, 1},    /*  2400 bps (error: 0.16%) */
  {233, 0},    /*  4800 bps (error: 0.16%) */
  {116, 0},    /*  9600 bps (error: 0.16%) */
  { 58, 0},    /* 19200 bps (error: -0.69%) */
  { 35, 0},    /* 31250 bps (error: 0.00%) */
  { 28, 0},    /* 38400 bps (error: 1.02%) */
  { 19, 0},    /* 57600 bps (error: -2.34%) */
  {  9, 0}     /*115200 bps (error: -2.34%) */
};

/* ==== Data buffer ==== */
unsigned char snd_data[16] = "0123456789abcdef";
unsigned char rcv_data[16];

```

## 3.4 Sample Program List "main.c" (3/9)

```
/*""FUNC COMMENT""***************************************************************
 * ID            :
 * Outline       : Sample program main
 *-------------------------------------------------------------------------------
 * Include       :
 *-------------------------------------------------------------------------------
 * Declaration   : void main(void);
 *-------------------------------------------------------------------------------
 * Description   : Initializes SCIF0 in predefined communication format and
 *               : operating mode, and transmits/receives 16-byte data, using
 *               : DMA transfer.
 *-------------------------------------------------------------------------------
 * Argument      : void
 *-------------------------------------------------------------------------------
 * Return Value  : void
 *-------------------------------------------------------------------------------
 * Note          : None
 *""FUNC COMMENT END""***********************************************************/
void main(void)
{
  /* ==== Initializes the DMAC ==== */
  io_init_dmac4( snd_data, &SCIF0.SCFTDR.BYTE, 16);
  io_init_dmac5( &SCIF0.SCFRDR.BYTE, rcv_data, 16);

  /* ==== Initializes SCIF0 in asynchronous mode (enable transmission/reception) ==== */
  io_init_scif0_dma(CBR_19200);   /* Specifies the bit rate as 19200 bps */

  /* ==== Waits until transmission is completed ==== */
  while(DMAC.CHCR4.BIT.TE == 0) {
    /* wait */
  }
  DMAC.CHCR4.BIT.DE = 0;      /* Disables the transfer */
  DMAC.CHCR4.BIT.TE = 0;      /* Clears the TE flag */

  /* ==== Waits until reception is completed ==== */
  while(DMAC.CHCR5.BIT.TE == 0) {
    /* wait */
  }
  DMAC.CHCR5.BIT.DE = 0;      /* Disables the transfer */
  DMAC.CHCR5.BIT.TE = 0;      /* Clears the TE flag */

  while (1) {
    /* Program end */
  }
}

```

## 3.5 Sample Program List "main.c" (4/9)

```
/*""FUNC COMMENT""**************************************************************
 * ID            :
 * Outline       : SCIF0 configuration
 *------------------------------------------------------------------------------
 * Include       : iodefine.h
 *------------------------------------------------------------------------------
 * Declaration   : void io_init_scif0_dma(int bps);
 *------------------------------------------------------------------------------
 * Description   : Configures SCIF0 as the UART module.
 *               : Sets it in asynchronous mode (UART), 8-bit, no parity,
 *               : 1 stop bit, and RTS/CTS disabled.
 *               : Specifies the baud rate by the argument "bps".
 *------------------------------------------------------------------------------
 * Argument      : int bps ; I : Baud rate specified value (table index)
 *------------------------------------------------------------------------------
 * Return Value  : void
 *------------------------------------------------------------------------------
 * Note          : The above baud rate specified value is applicable when using
 *               : the peripheral clock (operating frequency for the peripheral
 *               : module using the internal clock) is 36 MHz. Alter the baud rate
 *               : setting when using other clocks.
 *""FUNC COMMENT END""*********************************************************/
void io_init_scif0_dma(int bps)
{
  /* ==== Wakes up the MCU from power-down mode ==== */
  /* ---- Sets Standby control register 4 (STBCR4) ---- */
  CPG.STBCR4.BIT.MSTP47 = 0;      /* Starts to supplying clock to SCIF0 */

  /* ==== Configures SCIF0 ==== */
  /* ---- Sets the Serial control register (SCSCRi) ---- */
  SCIF0.SCSCR.WORD = 0x0000;      /* SCIF0 stops transmission/reception */

  /* ---- Sets the FIFO control register (SCFCRi) ---- */
  SCIF0.SCFCR.BIT.TFRST = 1;      /* Resets the transmit FIFO */
  SCIF0.SCFCR.BIT.RFRST = 1;      /* Resets the receive FIFO */

  /* ---- Sets the Serial status register (SCFSRi) ---- */
  SCIF0.SCFSR.WORD &= 0xff6eu;/* Clears bits ER, BRK, and DR */

  /* ---- Sets the Line status register (SCLSRi) ---- */
  SCIF0.SCLSR.BIT.ORER = 0;       /* Clears the ORER bit */

  /* ---- Sets the Serial control register (SCSCRi) ---- */
  SCIF0.SCSCR.BIT.CKE = 0x0;      /* B'00: internal clock */

```

## 3.6 Sample Program List "main.c" (5/9)

```
  /* ---- Sets the Serial mode register (SCSMRi) ---- */
  SCIF0.SCSMR.WORD = scif_baud[bps].scsmr;
                            /*  Communication mode, 0: Asynchronous mode  */
                            /*  Character length, 0: 8-bit data   */
                            /*  Parity enable, */
                            /*     0: Disables to add and check parity */
                            /*  Parity mode, 0: Even parity */
                            /*  Stop bit length, 0: 1 stop bit */
                            /*  Clock select: Setting in table */

  /* ---- Sets the Serial extension mode register (SCEMRi) ---- */
  SCIF0.SCEMR.WORD = 0x0000;      /* Baud rate generator double-speed mode, */
                            /*    0: Normal mode */
                            /* Base clock select in asynchronous mode, */
                            /*    0: Base clock is 16 times the bit rate */

  /* ---- Sets the Bit rate register (SCBRRi) ---- */
  SCIF0.SCBRR.BYTE = scif_baud[bps].scbrr;

  /* ---- Sets the FIFO control register (SCFCRi) ---- */
  SCIF0.SCFCR.WORD = 0x0030;      /* RTS output active trigger: default */
                            /* Number of the receive FIFO data trigger: 1 */
                            /* Number of the transmit FIFO data trigger: 0 */
                            /* Modem control enable: Disabled */
                            /* Transmit FIFO data register reset: Disabled */
                            /* Receive FIFO data register reset: Disabled */
                            /* Loop-back test: Disabled */

  /* ---- Sets the Serial port register (SCSPTRi) ---- */
  SCIF0.SCSPTR.WORD = 0x0053;     /* Serial port break input/output, */
                            /*    1: Outputs the SPB2DT */
                            /* value to the TxD pin */
                            /* Serial port break data, */
                            /*    1: Input/output data is high level */

  /* ==== Sets the General-purpose I/O ports ==== */
  PORT.PJCR0.BIT.PJ0MD = 4;       /* Specifies TxD0 pin */
  PORT.PJCR0.BIT.PJ1MD = 4;       /* Specifies RxD0 pin */

  /* ---- Sets the Serial control register (SCSCRi) ---- */
  SCIF0.SCSCR.WORD |= 0x00F0;     /* Enables SCIF0 to transmit/receive data */
                            /* Enables the TXI request */
                            /* Enables RXI, ERI, BRI requests */
}
```

## 3.7 Sample Program List "main.c" (6/9)

```
/*""FUNC COMMENT""**************************************************************
 * ID              :
 * Outline         : DMAC setting
 *------------------------------------------------------------------------------
 * Include         : iodefine.h
 *------------------------------------------------------------------------------
 * Declaration     : void io_init_dmac4(void *src, void *dst, int count);
 *------------------------------------------------------------------------------
 * Description     : Configures the Direct Memory Access Controller (DMAC) channel 4.
 *                 : Uses the on-chip peripheral module request (TXI0) to transfer
 *                 : the transmit data to SCIF0. Reload function is not used.
 *------------------------------------------------------------------------------
 * Argument        : void *src ; I : Transfer source address
 *                 : void *dst ; O : Transfer destination address
 *                 : int count ; I : Number of transfers
 *------------------------------------------------------------------------------
 * Return Value    : void
 *------------------------------------------------------------------------------
 * Note            :
 *""FUNC COMMENT END""**********************************************************/
void io_init_dmac4(void *src, void *dst, int count)
{
  /* ==== Sets Standby control register 2 ==== */
  CPG.STBCR2.BIT.MSTP8 = 0;              /* Clears the DMAC module standby */

  /* ==== Disables transfer on DMA_channel 4 ==== */
  DMAC.CHCR4.BIT.DE = 0x0;           /* Disables the DMA transfer */

  /* ==== Sets DMA source address register_4 (SAR_4) ==== */
  DMAC.SAR4.LONG = (unsigned long)src;

  /* ==== Sets DMA destination address register_4 (DAR_4) ==== */
  DMAC.DAR4.LONG = (unsigned long)dst;

  /* ==== Sets DMA transfer count register_4 (DMATCR_4) ==== */
  DMAC.DMATCR4.LONG = count;

```

## 3.8 Sample Program List "main.c" (7/9)

```
  /* ==== Sets DMA channel control register_4 (CHCR_4) ==== */
  DMAC.CHCR4.LONG = 0x00001800;
    /*
        bit 31    : TC DMATCR transfer: 0-----   Executes a transfer by
                                             a DMA transfer request
        bit 30    : reserve 0
        bit 29    : RLDSAR ON : 0-------------   Disables the reload function (RSAR)
        bit 28    : RLDDAR ON : 0-------------   Disables the reload function (RDAR)
        bit 27    : reserve 0
        bit 26    : DAF : 0-------------------   Not used
        bit 25    : SAF : 0-------------------   Not used
        bit 24    : reserve 0
        bit 23    : DO over run0 : 0----------   Not used
        bit 22    : TL TEND low active : 0----   Not used
        bit 21    : reserve 0
        bit 20    : TEMASK :0-----------------   Not used
        bit 19    : HE :0---------------------   Not used
        bit 18    : HIE :0--------------------   Not used
        bit 17    : AM :0---------------------   Not used
        bit 16    : AL :0---------------------   Not used
        bits 15, 14: DM1:0 DM0:0--------------   Destination address is fixed
        bits 13, 12: SM1:0 SM0:1--------------   Increments the source address
        bits 11 to 8: RS: B'1000--------------   Transfer request is from
                                             the extension resource selector
        bit 7     : DL : DREQ level : 0 -------  Not used
        bit 6     : DS : DREQ select :0 Low level Not used
        bit 5     : TB :cycle :0---------------  Cycle steal mode
        bits 4, 3 : TS : transfer size: B'00---  Transfers data in bytes
        bit 2     : IE : interrupt enable: 0---  Disables interrupt
        bit 1     : TE : transfer end: 0-------  Clears the TE flag
                                               (Clear the flag to 0 after
                                               reading 1)
        bit 0     : DE : DMA enable bit: 0-----  Disables the DMA transfer
    */
  /* ==== Sets DMA extension resource selector 2 (DMARS2) ==== */
  DMAC.DMARS2.BYTE.CH4 = 0x81;        /* SCIF0 transmission */

  /* ---- Sets the DMA operation register (DMAOR)---- */
  DMAC.DMAOR.WORD |= 0x0007;              /* Sets the DME bit. To avoid clearing */
                                      /* address error and NMI flags, wrote 1 */
                                      /* to the AE bit and NMIF flag */
  /* ======= Enables transfer on DMA_channel 4 ======= */
  DMAC.CHCR4.BIT.DE = 0x1;
}
```

## 3.9 Sample Program List "main.c " (8/9)

```
/*""FUNC COMMENT""**************************************************************
 * ID               :
 * Outline          : DMAC setting
 *------------------------------------------------------------------------------
 * Include          : iodefine.h
 *------------------------------------------------------------------------------
 * Declaration      : void io_init_dmac5(void *src, void *dst, int count);
 *------------------------------------------------------------------------------
 * Description      : Configures the Direct Memory Access Controller (DMAC) channel 5.
 *                  : Uses the on-chip peripheral module request (RXI0) to transfer
 *                  : the receive data from SCIF0. Reload function is not used.
 *------------------------------------------------------------------------------
 * Argument         : void *src ; I : Transfer source address
 *                  : void *dst ; O : Transfer destination address
 *                  : int count ; I : Number of transfers
 *------------------------------------------------------------------------------
 * Return Value     : void
 *------------------------------------------------------------------------------
 * Note             :
 *""FUNC COMMENT END""*********************************************************/
void io_init_dmac5(void *src, void *dst, int count)
{
  /* ==== Sets Standby control register 2 ==== */
  CPG.STBCR2.BIT.MSTP8 = 0;               /* Clears the DMAC module standby */

  /* ==== Disables transfer on DMA_channel 5 ==== */
  DMAC.CHCR5.BIT.DE = 0x0;            /* Disables the DMA transfer */

  /* ==== Sets DMA source address register_5 (SAR_5) ==== */
  DMAC.SAR5.LONG = (unsigned long)src;

  /* ==== Sets DMA destination address register_5 (DAR_5) ==== */
  DMAC.DAR5.LONG = (unsigned long)dst;

  /* ==== Sets DMA transfer count register_5 (DMATCR_5) ==== */
  DMAC.DMATCR5.LONG = count;

```

## 3.10 Sample Program List "main.c" (9/9)

```
  /* ==== Sets DMA channel control register_5 (CHCR_5) ==== */
  DMAC.CHCR5.LONG = 0x00004800;
    /*
        bit 31    : TC DMATCR transfer:1------    Executes a transfer by
                                              a DMA transfer request
        bit 30    : reserve 0
        bit 29    : RLDSAR ON : 0-------------    Disables the reload function (RSAR)
        bit 28    : RLDDAR ON : 0-------------    Disables the reload function (RDAR)
        bit 27    : reserve 0
        bit 26    : DAF : 0-------------------    Not used
        bit 25    : SAF : 0-------------------    Not used
        bit 24    : reserve 0
        bit 23    : DO over run0 : 0----------    Not used
        bit 22    : TL TEND low active : 0----    Not used
        bit 21    : reserve 0
        bit 20    : TEMASK :0-----------------    Not used
        bit 19    : HE :0---------------------    Not used
        bit 18    : HIE :0--------------------    Not used
        bit 17    : AM :0---------------------    Not used
        bit 16    : AL :0---------------------    Not used
        bits 15, 14 : DM1:0 DM0:1-------------    Increments the destination address
        bits 13, 12 : SM1:0 SM0:0-------------    Source address is fixed
        bits 11 to 8: RS : B'1000-------------    Transfer request is from
                                              the extension resource selector
        bit 7     : DL : DREQ level : 0 ------    Not used
        bit 6     : DS : DREQ select :0 Low level Not used
        bit 5     : TB :cycle :0--------------    Cycle steal mode
        bits 4, 3 : TS : transfer size: B'00--    Transfers data in words
        bit 2     : IE : interrupt enable: 0--    Disables interrupt
        bit 1     : TE : transfer end: 0------    Clears the TE flag
                                                (Clear the flag to 0 after
                                                reading 1)
        bit0     : DE : DMA enable bit: 0-----    Disables the DMA transfer
    */
  /* ==== Sets DMA extension resource selector 2 (DMARS2) ==== */
  DMAC.DMARS2.BYTE.CH5 = 0x82;        /* SCIF0 reception */

  /* ---- Sets the DMA operation register (DMAOR)---- */
  DMAC.DMAOR.WORD |= 0x0007;              /* Sets the DME bit. To avoid clearing */
                                      /* address error and NMI flags, write 1 */
                                      /* to the AE bit and NMIF flag*/
  /* ======= Enables transfer on DMA_channel 5 ======= */
  DMAC.CHCR5.BIT.DE = 0x1;
}
/* End of File */
```

## 4. References

- Software Manual
  SH-2A/SH2A-FPU Software Manual Rev.3.00
  The latest version of the software manual can be downloaded from the Renesas Electronics website.

- Hardware Manual
  SH7262 Group, SH7264 Group Hardware manual Rev.2.00
  The latest version of the hardware manual can be downloaded from the Renesas Electronics website.

## Website and Support

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

Inquiries
  http://www.renesas.com/inquiry

## Revision Record

| Rev. | Date | Description | |
|---|---|---|---|
| | | Page | Summary |
| 1.00 | Dec.28.10 | — | First edition issued |

## General Precautions in the Handling of MPU/MCU Products

The following usage notes are applicable to all MPU/MCU products from Renesas. For detailed usage notes on the products covered by this manual, refer to the relevant sections of the manual. If the descriptions under General Precautions in the Handling of MPU/MCU Products and in the body of the manual differ from each other, the description in the body of the manual takes precedence.

1. Handling of Unused Pins
   Handle unused pins in accord with the directions given under Handling of Unused Pins in the manual.
   — The input pins of CMOS products are generally in the high-impedance state. In operation with an unused pin in the open-circuit state, extra electromagnetic noise is induced in the vicinity of LSI, an associated shoot-through current flows internally, and malfunctions occur due to the false recognition of the pin state as an input signal become possible. Unused pins should be handled as described under Handling of Unused Pins in the manual.
2. Processing at Power-on
   The state of the product is undefined at the moment when power is supplied.
   — The states of internal circuits in the LSI are indeterminate and the states of register settings and pins are undefined at the moment when power is supplied.
   In a finished product where the reset signal is applied to the external reset pin, the states of pins are not guaranteed from the moment when power is supplied until the reset process is completed.
   In a similar way, the states of pins in a product that is reset by an on-chip power-on reset function are not guaranteed from the moment when power is supplied until the power reaches the level at which resetting has been specified.
3. Prohibition of Access to Reserved Addresses
   Access to reserved addresses is prohibited.
   — The reserved addresses are provided for the possible future expansion of functions. Do not access these addresses; the correct operation of LSI is not guaranteed if they are accessed.
4. Clock Signals
   After applying a reset, only release the reset line after the operating clock signal has become stable. When switching the clock signal during program execution, wait until the target clock signal has stabilized.
   — When the clock signal is generated with an external resonator (or from an external oscillator) during a reset, ensure that the reset line is only released after full stabilization of the clock signal. Moreover, when switching to a clock signal produced with an external resonator (or by an external oscillator) while program execution is in progress, wait until the target clock signal is stable.
5. Differences between Products
   Before changing from one product to another, i.e. to one with a different type number, confirm that the change will not lead to problems.
   — The characteristics of MPU/MCU in the same group but having different type numbers may differ because of the differences in internal memory capacity and layout pattern. When changing to products of different type numbers, implement a system-evaluation test for each of the products.

# 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.

RENESAS

**SALES OFFICES** Renesas Electronics Corporation http://www.renesas.com

Refer to "http://www.renesas.com/" for the latest and detailed information.

**Renesas Electronics America Inc.**
2880 Scott Boulevard Santa Clara, CA 95050-2554, U.S.A.
Tel: +1-408-588-6000, Fax: +1-408-588-6130

**Renesas Electronics Canada Limited**
1101 Nicholson Road, Newmarket, Ontario L3Y 9C3, Canada
Tel: +1-905-898-5441, Fax: +1-905-898-3220

**Renesas Electronics Europe Limited**
Dukes Meadow, Millboard Road, Bourne End, Buckinghamshire, SL8 5FH, U.K
Tel: +44-1628-585-100, Fax: +44-1628-585-900

**Renesas Electronics Europe GmbH**
Arcadiastrasse 10, 40472 Düsseldorf, Germany
Tel: +49-211-6503-0, Fax: +49-211-6503-1327

**Renesas Electronics (China) Co., Ltd.**
7th Floor, Quantum Plaza, No.27 ZhiChunLu Haidian District, Beijing 100083, P.R.China
Tel: +86-10-8235-1155, Fax: +86-10-8235-7679

**Renesas Electronics (Shanghai) Co., Ltd.**
Unit 204, 205, AZIA Center, No.1233 Lujiazui Ring Rd., Pudong District, Shanghai 200120, China
Tel: +86-21-5877-1818, Fax: +86-21-6887-7858 / -7898

**Renesas Electronics Hong Kong Limited**
Unit 1601-1613, 16/F., Tower 2, Grand Century Place, 193 Prince Edward Road West, Mongkok, Kowloon, Hong Kong
Tel: +852-2886-9318, Fax: +852 2886-9022/9044

**Renesas Electronics Taiwan Co., Ltd.**
7F, No. 363 Fu Shing North Road Taipei, Taiwan, R.O.C.
Tel: +886-2-8175-9600, Fax: +886 2-8175-9670

**Renesas Electronics Singapore Pte. Ltd.**
1 harbourFront Avenue, #06-10, keppel Bay Tower, Singapore 098632
Tel: +65-6213-0200, Fax: +65-6278-8001

**Renesas Electronics Malaysia Sdn.Bhd.**
Unit 906, Block B, Menara Amcorp, Amcorp Trade Centre, No. 18, Jln Persiaran Barat, 46050 Petaling Jaya, Selangor Darul Ehsan, Malaysia
Tel: +60-3-7955-9390, Fax: +60-3-7955-9510

**Renesas Electronics Korea Co., Ltd.**
11F., Samik Lavied' or Bldg., 720-2 Yeoksam-Dong, Kangnam-Ku, Seoul 135-080, Korea
Tel: +82-2-558-3737, Fax: +82-2-558-5141

Colophon 1.0