SH7262/SH7264 Group

Serial Sound Interface in Master Transceiver Mode

R01AN0547EJ0101
Rev. 1.01
Feb. 23, 2011

Summary

This application note describes an example of setting the SH7262/SH7264 Microcomputers (MCUs) Serial Sound Interface (SSI) in master transceiver mode.

Target Device

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

Contents

1. Introduction 2
2. Applications 3
3. Sample Program Listing 15
4. References 28

1. Introduction

1.1 Specifications

Set the SH7264 Serial Sound Interface (SSI) in master transceiver mode to transmit/receive the PCM data in full-duplex mode. To transfer data to SSI, use the Direct Memory Access Controller (DMAC).

1.2 Modules Used

- Serial Sound Interface (SSI)
- Direct Memory Access Controller (DMAC)
- General-purpose I/O Ports
- Interrupt Controller

1.3 Applicable Conditions

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

1.4 Related Application Note

Refer to the related application notes as follows:

- SH7262/SH7264 Group Example of Initialization
- SH7262/SH7264 Group Serial Sound Interface in Slave Receiver Mode
- SH7262/SH7264 Group Serial Sound Interface in Master Transmitter Mode

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.

2. Applications

This application sets the sampling frequency of the SSI to 44.1 kHz to operate as the master transceiver.

2.1 SSI Operation

The SSI has the following features

- Number of channels: 4
- Operating mode: Non-compressed mode

Non-compressed mode supports the serial audio streams divided by channels.

- Operates both as the transmitter and the receiver
- Channel 0 supports full-duplex transmission/reception
- Supports the serial bus format
- Asynchronous transfer between the data buffer and the shift register
- Clock divide ratio used in the serial bus interface selectable
- Controls the data transmission/reception by the DMAC or interrupts
- Oversampling clock options as follows:
  AUDIO_CLK pin
  AUDIO_X1, AUDIO_X2 pins
- Eight deep FIFO buffer included both in the transmitter and receiver

Figure 1 shows the SSI block diagram.

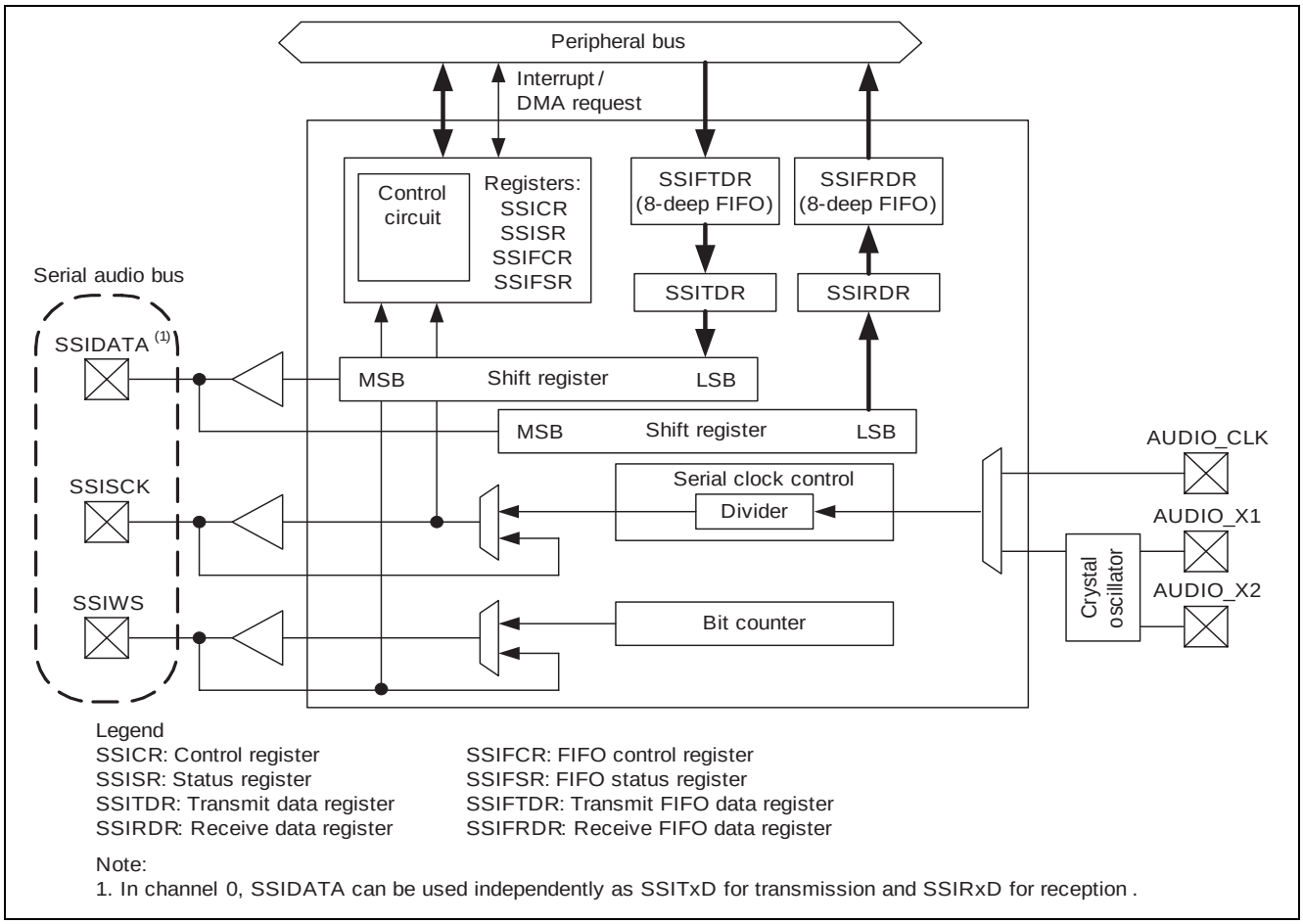

Figure 1 SSI Block Diagram

2.2 SSI Setting Procedure

Figure 2 shows the flow chart of setting the SSI. Figure 3 shows the flow chart of setting the DMAC.

Refer to the SH7262 Group, SH7264 Group Hardware Manual for details on registers.

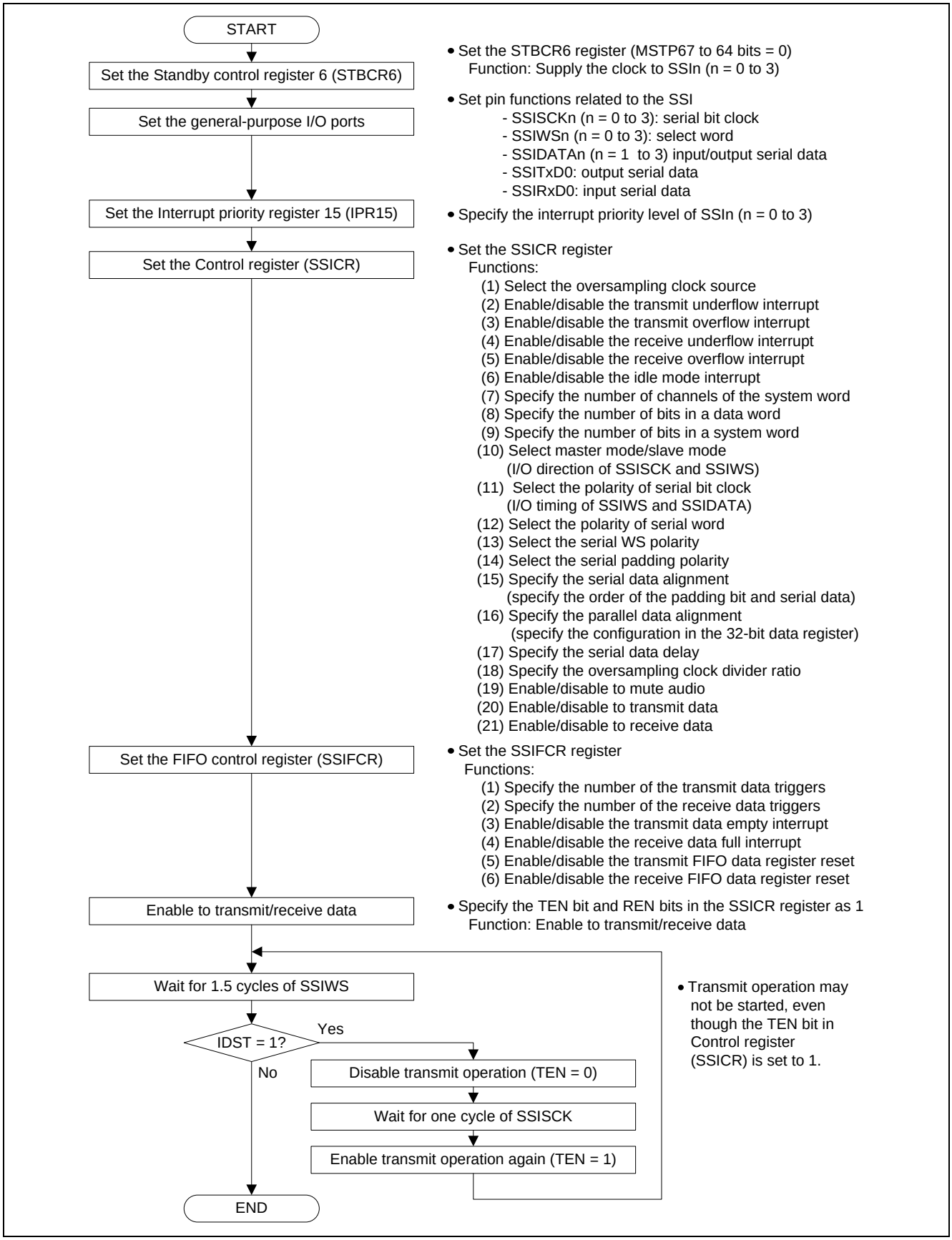

Figure 2 SSI Setup Flow Chart

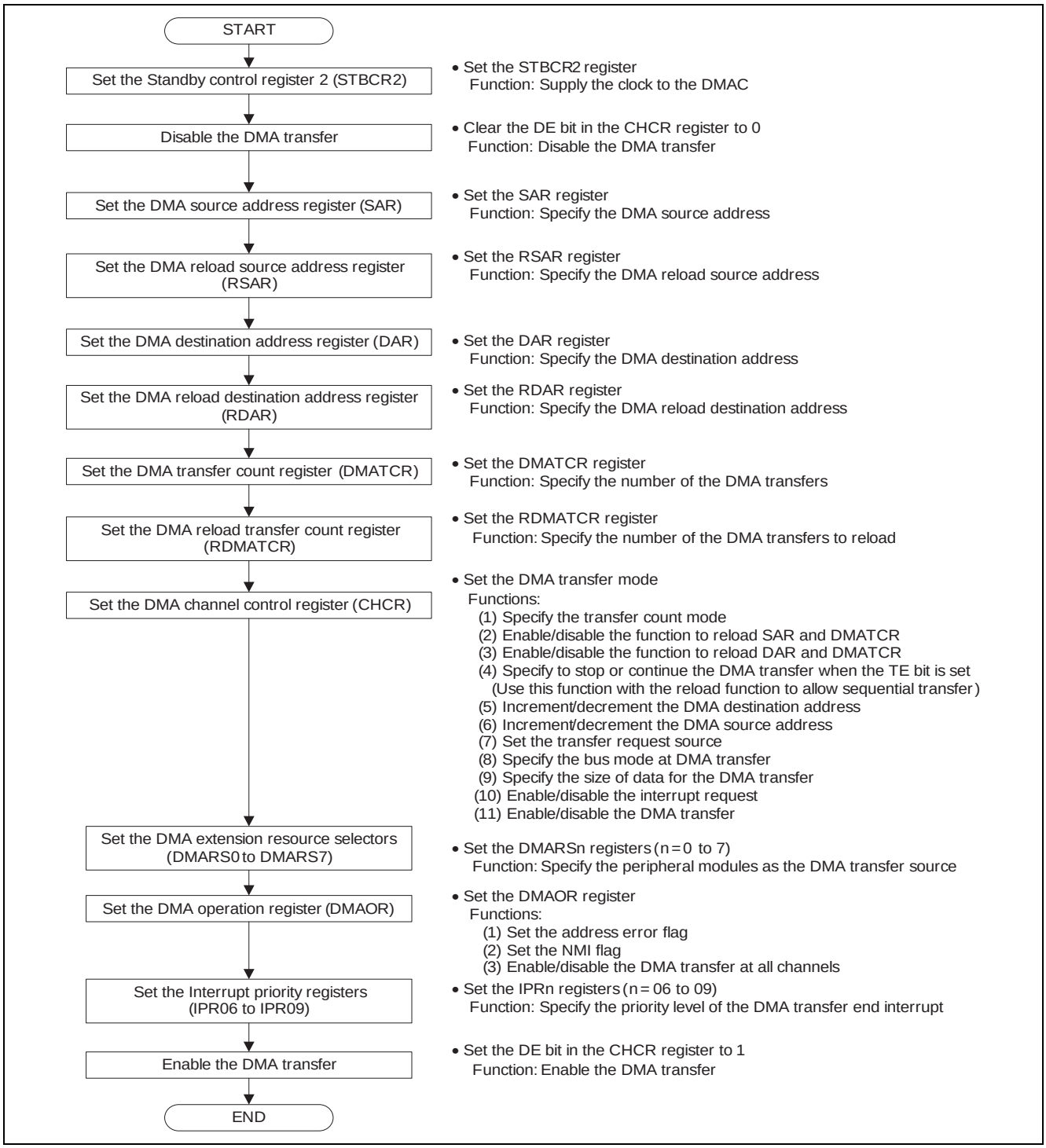

Figure 3 DMAC Setup Flow Chart

2.3 Sample Program Operation

The SSITxD0 and SSIRxD0 pins are directly connected to receive the transferred data in the sample program.

The SSI uses the DMA transfer request by the transmit data empty interrupt at the channel 0 of the SSI, and activates the channel 1 of the DMAC. The DMAC transfers data from an external memory to the Transmit FIFO data register (SSIFTDR) in the SSI channel 0. The SSITxD0 pin outputs the data in the SSIFTDR register via the Transmit data register (SSITDR).

The sample program continues to transfer ten samples (40 bytes) of PCM data for four times, and SSI is muted after the transfer is complete.

SSI setting in the sample program is as follows:

- Channel used: channel 0
- Operating mode: master transceiver
- Data transmission controlled by: DMAC (channel 1)
- Data reception controlled by: DMAC (channel 2)
- Oversampling clock: AUDIO_X1 input (11.2896 MHz)
- Serial oversampling clock frequency: One quarter the oversampling clock frequency (2.822 MHz)
- Data word length: 16 bits
- System word length: 32 bits
- Padding bit: Low level
- No delay between the SSIWS and SSIDATA signals
- Outputs the SSIWS and SSIDATA signals at the falling edge of the SSISCK signal
- Sampling frequency: 44.1 kHz (354 ns or 2.8224 MHz x 32 bits x 2)
- Outputs "H'FFFF" at the data word 1 (channel L) of the 1st channel, and "H'0000" at the data word 2 (channel R) of the 2nd channel.

Figure 4 shows the signal waveform in the sample program. Figure 5 shows the sample program block diagram.

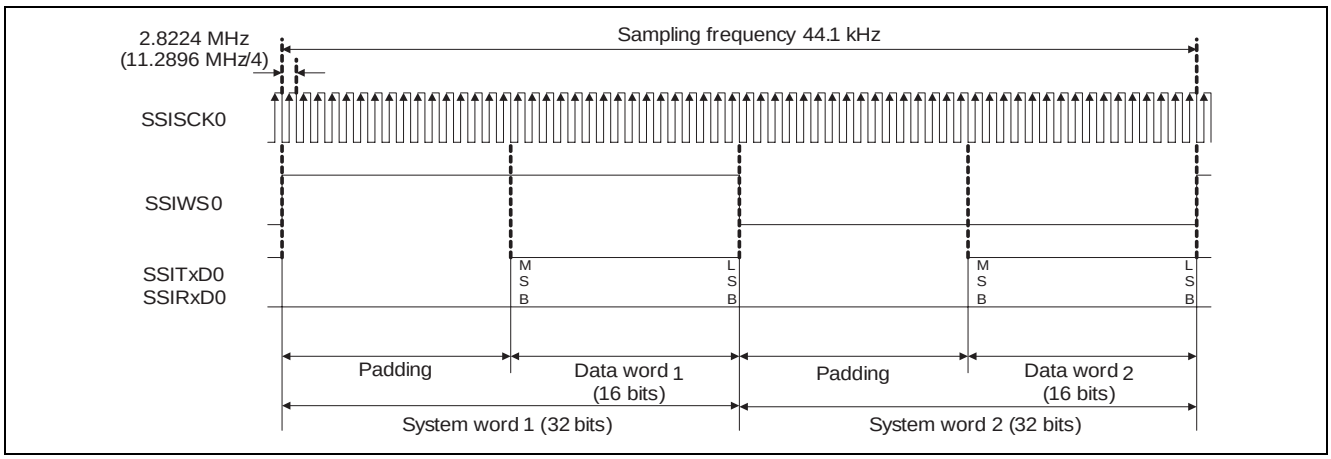

Figure 4 Signal Output Waveform in the Sample Program

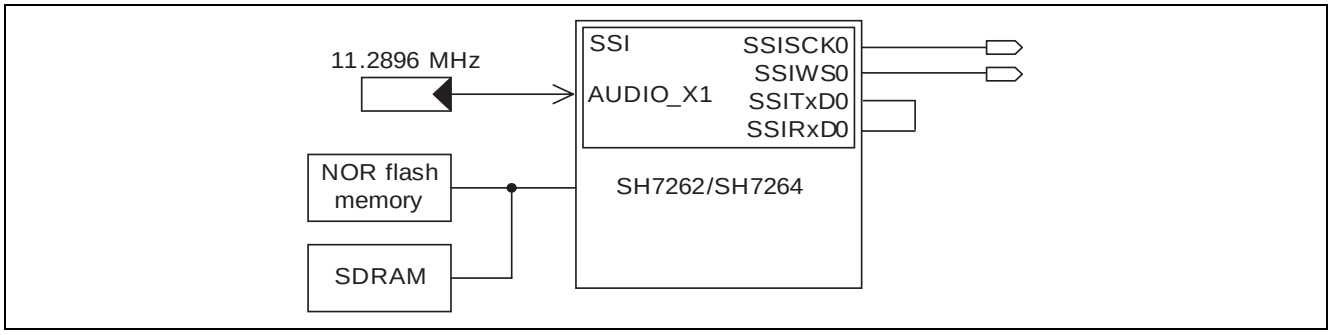

Figure 5 Sample Program Block Diagram

2.4 Sample Program Procedure

The table below lists the SSI registers setting in the sample program.

Table 2 and Table 3 list the DMAC registers setting in the sample program. Figure 6 to Figure 8 show the flow charts of the sample program.

Table 1 SSI Registers Setting

Register Name	Address	Value	Description
Control register 0 (SSICR_0)	H'FFFF 0000	H'3C0B D520	• CKS bit = B'0 (Specify the oversampling clock as AUDIO_X1 input) • TUIEN bit = B'1 (Enable the Transmit underflow interrupt) • TOIEN bit = B'1 (Enable the Transmit overflow interrupt) • RUIEN bit = B'0 (Disable the Receive underflow interrupt) • ROIEN bit = B'0 (Disable the Receive overflow interrupt) • IIEN bit = B'0 (Disable Idle mode interrupt) • CHNL [1:0] bits = B'00 (System words have one channel) • DWL [2:0] bits = B'001 (Data word length: 16 bits) • SWL [2:0] bits = B'011 (System word length: 32 bits) • SCKD bit = B'1 (Serial bit clock output, master mode) • SWSD bit = B'1 (Serial word select output, master mode) • SCKP bit = B'0 (Outputs SSIWS and SSIDATA signals at the falling edge of the SSISCK signal) • SWSP bit = B'1 (SSIWS is High for the 1st channel, and is Low for the 2nd channel) • SPDP bit = B'0 (Padding bits are low) • SDTA bit = B'1 (Transmits and receives in the order of padding bits, and serial data) • PDTA bit = B'0 (Lower bits of parallel data are transmitted and received) • DEL bit = B'1 (No delay between the SSIWS and SSIDATA) • CKDV bits [3:0]= B'0010 (Specify the oversampling clock as audio Φ4) • MUEN bit = B'0 (Not muted) • TEN bit = B'0 (Disable to transmit data) • REN bit = B'0 (Disable to receive data)
		H'3C0B D523	• TEN bit = B'1 (Enable to transmit data) • REN bit = B'1 (Enable to receive data)
FIFO control register 0 (SSIFCR_0)	H'FFFF 0010	H'0000 000C	• TTRG[1:0] bits = B'00 (Number of transmit data triggers: 7) • RTRG [1:0] bits = B'00 (Number of receive data triggers: 1) • TIE bit = B'1 (Enable the transmit data empty interrupt) • RIE bit = B'1 (Enable the receive data full interrupt) • TFRST bit = B'0 (Disable to reset the transmit FIFO) • RFRST bit = B'0 (Disable to reset the receive FIFO)

Table 2 DMAC Registers Setting (Channel 1)

Register Name	Address	Setting	Description
DMA channel control register 1 (CHCR_1)	H'FFFE 101C	H'0000 0000	• DE bit = B'0 (Disable the DMA transfer)
		H'2010 1814	• TC bit = B'0 (Transmit data once by one transfer request) • RLDSAR bit = B'1 (Disable the SAR reload function) • RLDDAR bit = B'0 (Disable the DAR reload function) • DAF bit, SAF bit = B'00 (not used) • DO bit = B'0 (Not used) • TL bit = B'0 (Not used) • TEMASK bit = B'1 (Continue the DMA transfer when the TE bit is set) • HE bit, HIE bit = B'00 (Not used) • AM bit, AL bit = B'00 (Not used) • DM [1:0] bits = B'00 (Destination address fixed) • SM [1:0] bits = B'01 (Source address incremented) • RS [3:0] bits = B'1000 (Specify the DMA extension resource) • DL bit, DS bit = B'00 (Not used) • TB bit = B'0 (Specify the cycle steal mode) • TS [1:0] bits = B'10 (Specify the longword transfer) • IE bit = B'1 (Enable an interrupt request) • DE bit = B'0 (Disable the DMA transfer)
		H'2010 1815	• DE bit = B'1 (Enable the DMA transfer)
DMA source address register 1 (SAR_1)	H'FFFE 1010	Internal RAM	• Specify the internal RAM as the DMA transfer source address
DMA reload source address register 1 (RSAR_1)	H'FFFE 1110	Internal RAM	• Specify the internal RAM as the DMA reload transfer source address
DMA destination address register 1 (DAR_1)	H'FFFE 1014	H'FFFF 0818	• Specify the SSIFTDR register 1 as the DMA transfer destination address
DMA transfer count register 1 (DMATCR_1)	H'FFFE 1018	H'0000 000A	• Number of transfers: 10
DMA reload transfer count register 1 (RDMATCR_1)	H'FFFE 1118	H'0000 000A	• Number of transfers: 10
DMA operation register (DMAOR)	H'FFFE 1200	H'0001	• CMS [1:0] bits = B'00 (Normal mode) • PR [1:0] bits = B'00 (Channel priority level: Fixed mode 1) • AE bit = B'0 (Clear the address error flag) • NMIF bit = B'0 (Clear the NMI interrupt) • DME bit = B'1 (Enable the DMA transfer on all channels)
DMA extension resource selector 0 (DMARS0)	H'FFFE 1300	H'2100	• Specify the transmit function of the SSI channel 0 as the transfer request source of the DMA channel 1

Table 3 DMAC Registers Setting (Channel 2)

Register Name	Address	Setting	Description
DMA channel control register 2 (CHCR_2)	H'FFFE 102C	H'0000 0000	• DE bit = B'0 (Disable the DMA transfer)
		H'1010 4814	• TC bit = B'0 (Transmit data once by one transfer request) • RLDSAR bit = B'0 (Disable the SAR reload function) • RLDDAR bit = B'1 (Disable the DAR reload function) • DAF bit, SAF bit = B'00 (Not used) • DO bit = B'0 (Not used) • TL bit = B'0 (Not used) • TEMASK bit = B'1 (Continue the DMA transfer when the TE bit is set) • HE bit, HIE bit = B'00 (Not used) • AM bit, AL bit = B'00 (Not used) • DM [1:0] bits = B'01 (Destination address incremented) • SM [1:0] bits = B'00 (Source address fixed) • RS [3:0] bits = B'1000 (Specify the DMA extension resource) • DL bit, DS bit = B'00 (Not used) • TB bit = B'0 (Specify the cycle steal mode) • TS [1:0] bits = B'10 (Specify the longword transfer) • IE bit = B'1 (Enable an interrupt request) • DE bit = B'0 (Disable the DMA transfer)
		H'1010 4815	• DE bit = B'1 (Enable the DMA transfer)
DMA source address register 2 (SAR_2)	H'FFFE 1020	H'FFFF 001C	• Specify the SSIFRDR register 0 as the DMA transfer source address
DMA destination address register 2 (DAR_2)	H'FFFE 1024	Internal RAM	• Specify the internal RAM as the DMA transfer destination address
DMA reload destination address register 2 (RDAR_2)	H'FFFE 1124	Internal RAM	• Specify the internal RAM as the DMA reload transfer destination address
DMA transfer count register 2 (DMATCR_2)	H'FFFE 1028	H'0000 000A	• Number of transfers: 10
DMA reload transfer count register 2 (RDMATCR_2)	H'FFFE 1128	H'0000 000A	• Number of transfers: 10
DMA operation register (DMAOR)	H'FFFE 1200	H'0001	• CMS [1:0] bits = B'00 (Normal mode) • PR [1:0] bits = B'00 (Channel priority level: Fixed mode 1) • AE bit = B'0 (Clear the address error flag) • NMIF bit = B'0 (Clear the NMI interrupt) • DME bit = B'1 (Enable the DMA transfer on all channels)
DMA extension resource selector 1 (DMARS1)	H'FFFE 1304	H'0022	• Specify the SSI channel 0 as the transfer request source of the DMA channel 2

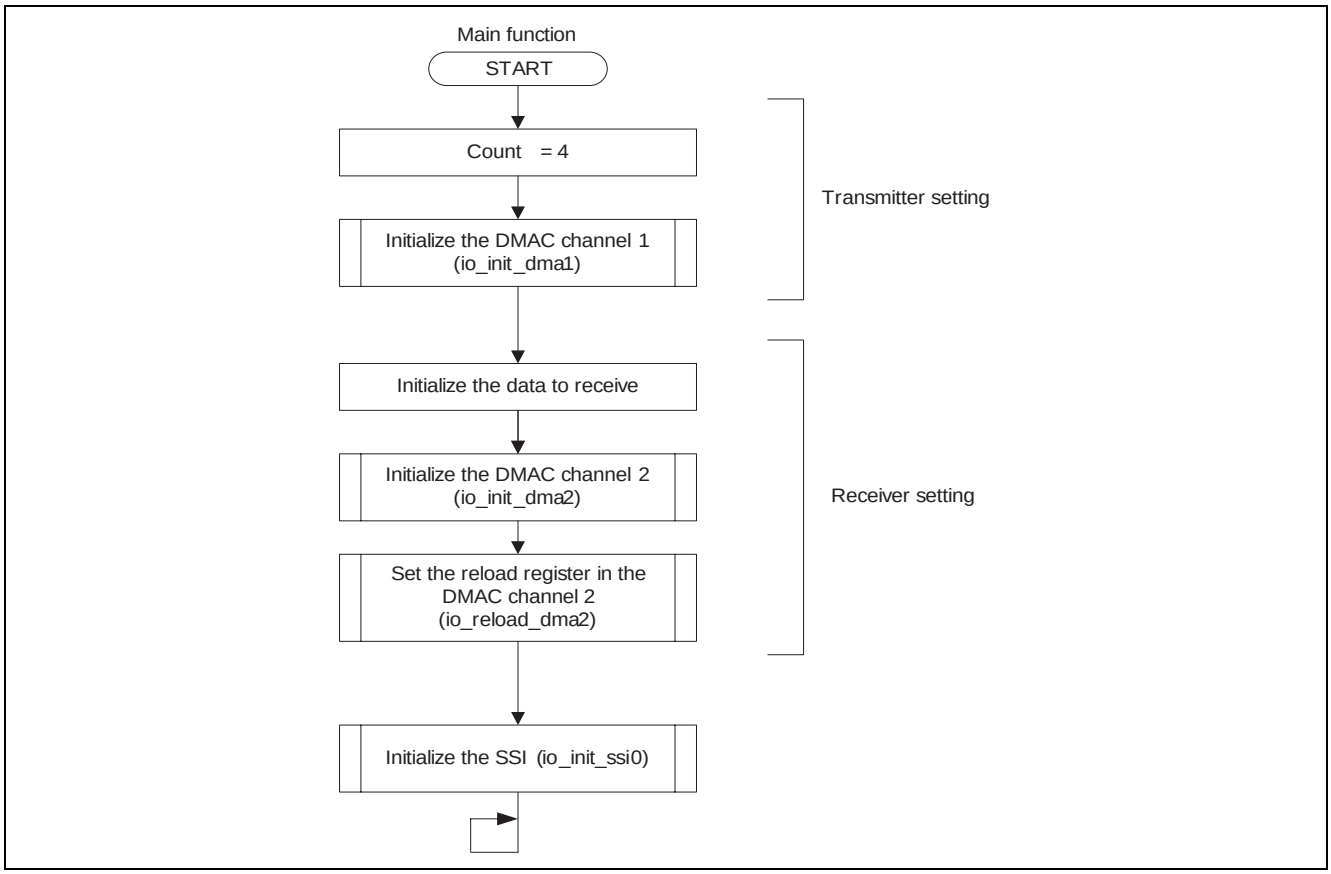

Figure 6 Sample Program Flow Chart (1/3)

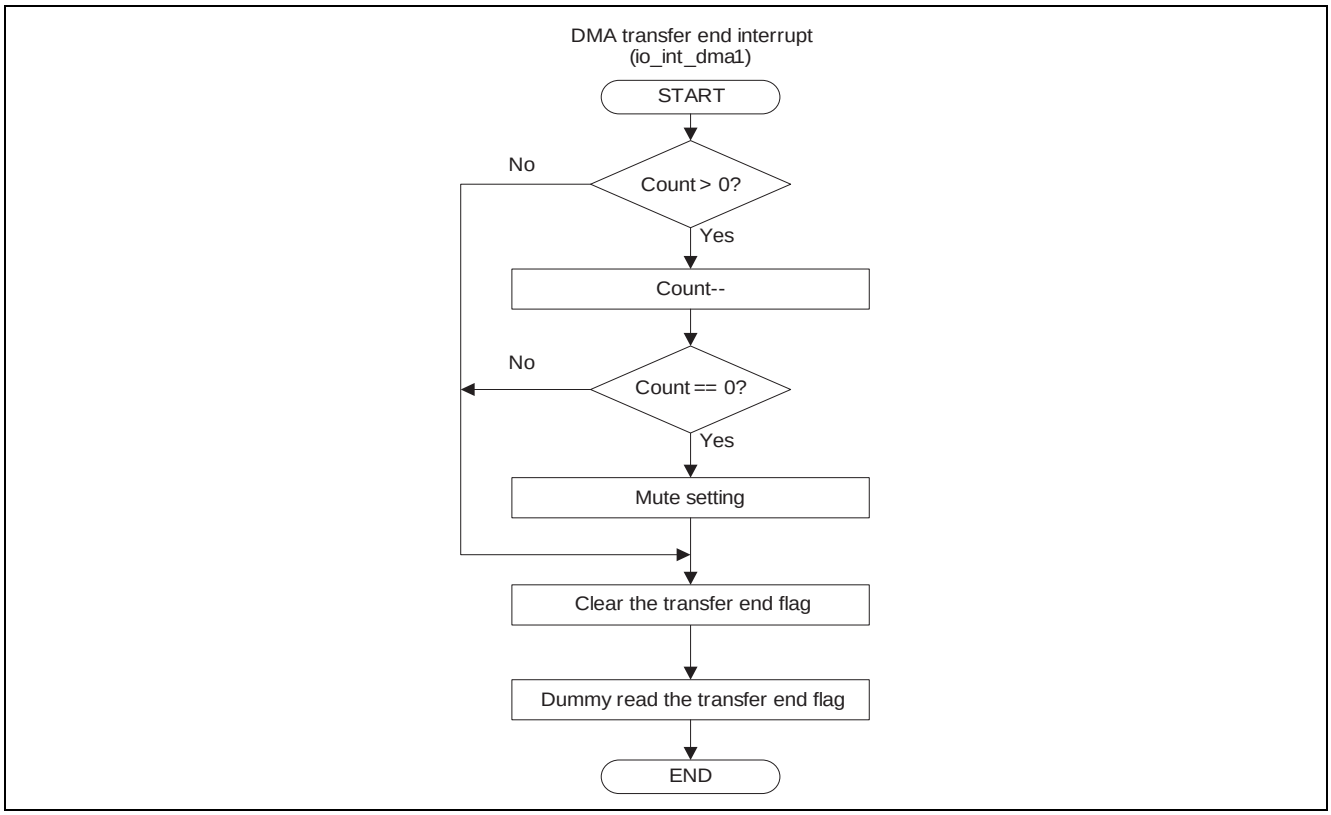

Figure 7 Sample Program Flow Chart (2/3)

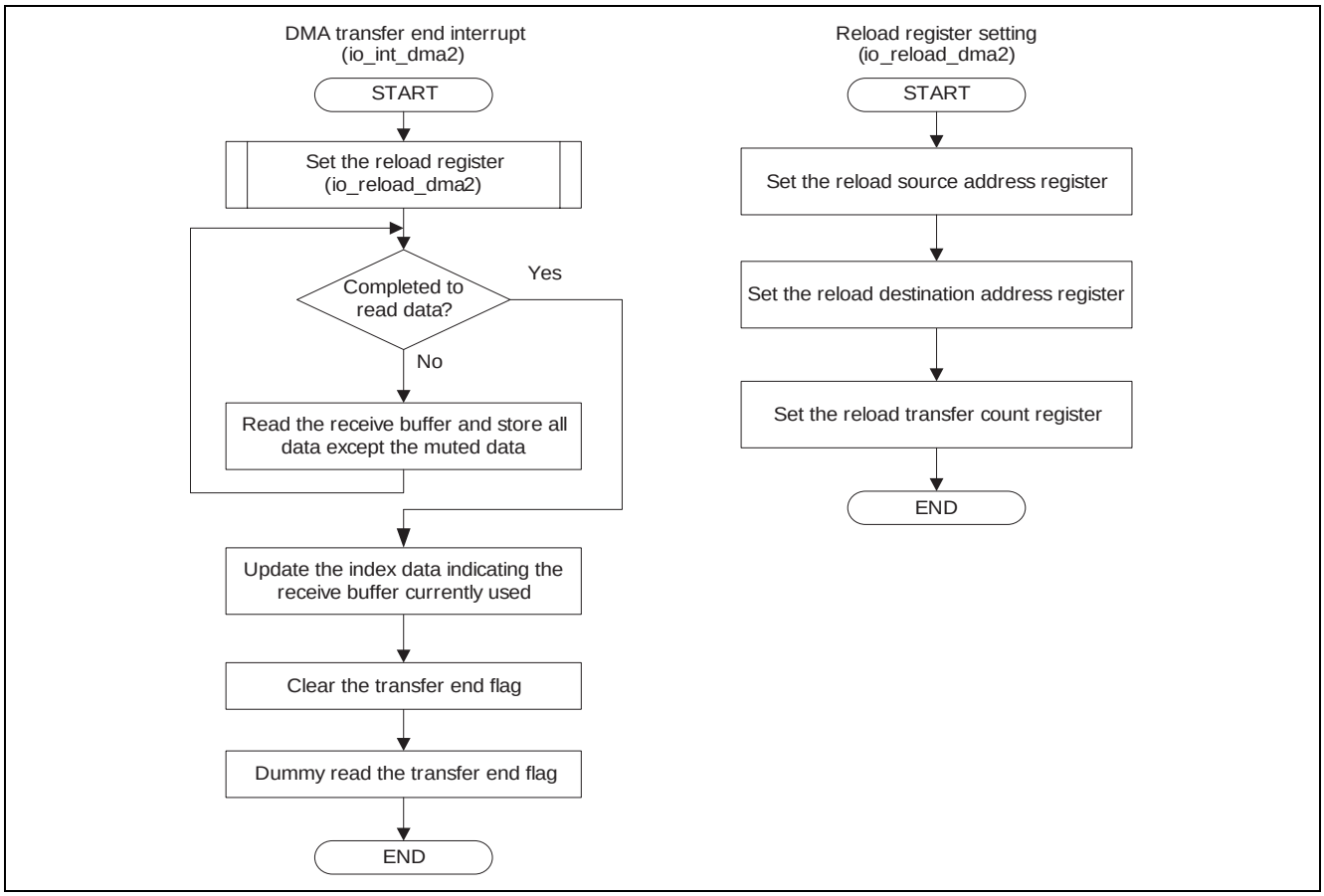

Figure 8 Sample Program Flow Chart (3/3)

3. Sample Program Listing

3.1 Sample Program Listing "main.c" (1/13)

```
1       /***********************************************************************
2       *   DISCLAIMER
3       *
4       *   This software is supplied by Renesas Electronics Corporation and is only
5       *   intended for use with Renesas products.  No other uses are authorized.
6       *
7       *   This software is owned by Renesas Electronics Corporation and is protected under
8       *   all applicable laws, including copyright laws.
9       *
10      *   THIS SOFTWARE IS PROVIDED "AS IS" AND RENESAS MAKES NO WARRANTIES
11      *   REGARDING THIS SOFTWARE, WHETHER EXPRESS, IMPLIED OR STATUTORY,
12      *   INCLUDING BUT NOT LIMITED TO WARRANTIES OF MERCHANTABILITY, FITNESS FOR A
13      *   PARTICULAR PURPOSE AND NON-INFRINGEMENT.  ALL SUCH WARRANTIES ARE EXPRESSLY
14      *   DISCLAIMED.
15      *
16      *   TO THE MAXIMUM EXTENT PERMITTED NOT PROHIBITED BY LAW, NEITHER RENESAS
17      *   ELECTRONICS CORPORATION NOR ANY OF ITS AFFILIATED COMPANIES SHALL BE LIABLE
18      *   FOR ANY DIRECT, INDIRECT, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES
19      *   FOR ANY REASON RELATED TO THIS SOFTWARE, EVEN IF RENESAS OR ITS
20      *   AFFILIATES HAVE BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.
21      *
22      *   Renesas reserves the right, without notice, to make changes to this
23      *   software and to discontinue the availability of this software.
24      *   By using this software, you agree to the additional terms and
25      *   conditions found by accessing the following link:
26      *   http://www.renesas.com/disclaimer
27      ***********************************************************************
28      *   Copyright (C) 2009(2011) Renesas Electronics Corporation. All rights reserved.
29      *""FILE COMMENT""*********** Technical reference data ************************
30      *   System Name : SH7264 Sample Program
31      *   File Name   : main.c
32      *   Abstract    : SSI in Master Transceiver Mode
33      *   Version     : 1.01.00
34      *   Device      : SH7262/SH7264
35      *   Tool-Chain  : High-performance Embedded Workshop (Ver.4.07.00).
36      *               : C/C++ compiler package for the SuperH RISC engine family
37      *               :                             (Ver.9.03 Release00).
38      *   OS          : None
39      *   H/W Platform: M3A-HS64G50(CPU board)
40      *   Description :
41      ***********************************************************************
42      *   History     : Feb.24,2009 Ver.1.00.00
43      *               : Feb.23,2011 Ver.1.01.00
44      *""FILE COMMENT END""*****************************************************/
45      #include <string.h>
46      #include "iodefine.h"     /* This file is automatically generated by the
47
```

3.2 Sample Program Listing "main.c" (2/13)

```
48
49    /* ==== Macro declaration ==== */
50    #define SSI_DATASIZE 40u
51    #define SSI_MUTEDATA 0x00000000ul
52
53    /* ==== Prototype declaration ==== */
54    void main(void);
55    void io_init_ssi0(void);
56    void io_init_dma1(void *src, void *dst, size_t size);
57    void io_init_dma2(void *src, void *dst, size_t size);
58    void io_reload_dma2(void *src, void *dst, size_t size);
59
60    /* ==== Variable declaration ==== */
61
62    /* ---- Transmitter data ---- */
63    unsigned long Data[SSI_DATASIZE/sizeof(unsigned long)] = {
64                    0x0000FFFFul,0x0000FFFFul,
65                    0x0000FFFFul,0x0000FFFFul,
66                    0x0000FFFFul,0x0000FFFFul,
67                    0x0000FFFFul,0x0000FFFFul,
68                    0x0000FFFFul,0x0000FFFFul};
69    unsigned int Count;
70
71    /* ---- Receiver data ---- */
72    unsigned long Buff[2][SSI_DATASIZE/sizeof(unsigned long)];  /* Receive buffer */
73    unsigned int BuffIdx;         /* Index data indicating the receive buffer currently used */
74    unsigned long RcvData[100];   /* Receive buffer to store all data except the muted data */
75    unsigned int RcvCnt;          /* Number of the receive data stored */
76
```

3.3 Sample Program Listing "main.c" (3/13)

```
77    /*""FUNC COMMENT""*****************************************************************
78     * ID          :
79     * Outline     : main
80     *-------------------------------------------------------------------------------
81     * Include     : #include "iodefine.h"
82     *-------------------------------------------------------------------------------
83     * Declaration : void main(void);
84     *-------------------------------------------------------------------------------
85     * Description : Initializes the SSI, and transmits data.
86     *-------------------------------------------------------------------------------
87     * Argument    : void
88     *-------------------------------------------------------------------------------
89     * Return Value: void
90     *""FUNC COMMENT END""*************************************************************/
91    void main(void)
92    {
93      /* ==== Transmitter setting ==== */
94      Count = 4u;                                       /* Number of the DMA transfers */
95      /* ---- Initializes the DMAC/enable the DMA transfer ---- */
96      io_init_dma1(    Data,                            /* Source address */
97                      (void *)&SSIF0.SSIFTDR,           /* Destination address */
98                       SSI_DATASIZE);                   /* Number of bytes */
99
100
101     /* ==== Receiver setting ==== */
102     RcvCnt = 0u;                                  /* Number of the receive data */
103     BuffIdx = 0u;                                     /* Receive buffer index */
104     /* ---- Initializes the DMAC/enable the DMA transfer ---- */
105     io_init_dma2(    (void *)&SSIF0.SSIFRDR,          /* Source address */
106                      Buff[BuffIdx],                   /* Destination address */
107                      SSI_DATASIZE);                   /* Number of bytes */
108     io_reload_dma2(  (void *)&SSIF0.SSIFRDR,
109                      Buff[BuffIdx^1u],
110                      SSI_DATASIZE);
111
112     /* ==== Initializes the SSI0 ==== */
113     io_init_ssi0();
114
115     while(1){
116       /* Program end */
117     }
118   }
```

3.4 Sample Program Listing "main.c" (4/13)

```
/*""FUNC COMMENT""*****************************************************************
 * ID          :
 * Outline     : SSI initialization
 *-----------------------------------------------------------------------------
 * Include     : #include "iodefine.h"
 *-----------------------------------------------------------------------------
 * Declaration : void io_init_ssi0(void);
 *-----------------------------------------------------------------------------
 * Description : Transfers data in master transceiver mode.
 *             : The sampling frequency is at 44.1 kHz.
 *-----------------------------------------------------------------------------
 * Argument    : void
 *-----------------------------------------------------------------------------
 * Return Value: void
 *""FUNC COMMENT END""*************************************************************/
void io_init_ssi0(void)
{
  /* ---- Supplies the clock to the SSI ---- */
  CPG.STBCR6.BIT.MSTP67 = 0u;         /* SSI0 */

  /* ----Selects the SSI pin functions ---- */
  PORT.PGCR2.BIT.PG8MD = 2u;          /* SSISCK0 */
  PORT.PGCR2.BIT.PG9MD = 2u;          /* SSIWS0 */
  PORT.PGCR2.BIT.PG10MD = 2u;         /* SSIRxD0 */
  PORT.PGCR2.BIT.PG11MD = 2u;         /* SSITxD0 */

  /* ---- Specifies the SSI interrupt level ---- */
  INTC.IPR15.BIT._SSI0 = 1u;          /* SSI0 */

  /* ---- Sets the Control register (SSICR) ---- */
  SSIF0.SSICR.LONG = 0x3C0BD520ul;
    /*
        bit31       : reserve 0
        bit30       : CKS : 0-----------------  AUDIO_X1 input
        bit29       : TUIEN : 1---------------  Enables the transmit underflow interrupt
        bit28       : TOIEN : 1---------------  Enables the transmit overflow interrupt
        bit27       : RUIEN : 1---------------  Enables the receive underflow interrupt
        bit26       : ROIEN : 1---------------  Enables the receive overflow interrupt
        bit25       : IIEN : 0----------------  Disables the idle mode interrupt
        bit24       : reserve 0
        bit23 to 22 : CHNL : B'00-------------  System words have one channel
        bit21 to 19 : DWL : B'001-------------  Data word length: 16 bits
        bit18 to 16 : SWL : B'011-------------  System word length: 32 bits
        bit15       : SCKD : 1----------------  Serial bit clock output, master mode
        bit14       : SWSD : 1----------------  Serial word WS output, master mode
        bit13       : SCKP : 0----------------  Outputs at the falling edge of the SSISCK,
                                                and latches at the rising edge of the SSISCK
        bit12       : SWSP : 1---------------   High level at 1st channel,
                                                low level at 2nd channel
        bit11       : SPDP : 0----------------  Padding bits are low level
        bit10       : SDTA : 1----------------  Transmits and receives in the order of
                                                padding bit, and serial data
```

3.5 Sample Program Listing "main.c" (5/13)

```
171           bit9        : PDTA : 0-------------------   Transmits and receives lower bits of
172                                                     parallel data
173           bit8        : DEL : 1--------------------   No delay between the SSIWS and SSIDATA
174           bit7 to 4   : CKDV : B'0010--------------   Specifies the oversampling clock as
175                                                     the AUDIO clock/4 (44.1kHz)
176           bit3        : MUEN : 0-------------------   Not muted
177           bit2        : reserve 0
178           bit1        : TEN : 0--------------------   Disables to transmit data
179           bit0        : REN : 0--------------------   Disables to receive data
180       */
181     /* ---- Sets the FIFO control register (SSIFCR) ---- */
182     SSIF0.SSIFCR.LONG = 0x0000000Cul;
183       /*
184           bit31 to 8: reserve 0
185           bit7 to 6 : TTRG : B'00----------------  Number of transmit data triggers: 7
186           bit5 to 4 : RTRG : B'00----------------  Number of receive data triggers: 1
187           bit3      : TIE : 1--------------------  Enables the transmit data empty interrupt
188           bit2      : RIE : 1--------------------  Enables the receive data full interrupt
189           bit1      : TFRST : 0------------------  Disables to reset the transmit FIFO
190                                                   data register
191           bit0      : RFRST : 0------------------  Disables to reset the receive FIFO
192                                                   receive register
193       */
194     /* ---- Enables to receive data ---- */
195     SSIF0.SSICR.BIT.REN = 1u;
196     /* ---- Enables to transmit data ---- */
197     SSIF0.SSICR.BIT.TEN = 1u;
198
199     /* ---- Checks the transmission begins ---- */
200     while(1){
201       /* Wait for 1.5 cycles of SSIWS */
202       for( w = 16000 ; w > 0 ; w--){/* 1.1 ms = (1/44.1 kHz) * 32 bit * 1.5 cyc */
203           /* wait */
204       }
205       /* If the serial bus is running */
206       if( SSIF0.SSISR.BIT.IDST == 0 ){
207           break;
208       }
209       /* Disables to transmit data */
210       SSIF0.SSICR.BIT.TEN = 0u;
211
212       /* Wait for one cycle of SSISCK */
213       for( w = 400 ; w > 0 ; w--){      /* 23 us = 1/44.1 kHz */
214           /* wait */
215       }
216       /* Enables to transmit data */
217       SSIF0.SSICR.BIT.TEN = 1u;
218     }
219
220   }
```

3.6 Sample Program Listing "main.c" (6/13)

```
221   /*""FUNC COMMENT""*****************************************************************
222    * ID          :
223    * Outline     : SSI interrupt
224    *-------------------------------------------------------------------------------
225    * Include     : #include "iodefine.h"
226    *-------------------------------------------------------------------------------
227    * Declaration : void io_int_ssi0(void);
228    *-------------------------------------------------------------------------------
229    * Description : Handles the SSI interrupt.
230    *-------------------------------------------------------------------------------
231    * Argument    : void
232    *-------------------------------------------------------------------------------
233    * Return Value: void
234    *""FUNC COMMENT END""*************************************************************/
235   void io_int_ssi0(void)
236   {
237     volatile int w;
238
239     /* Transmit underflow error */
240     if(SSIF0.SSISR.BIT.TUIRQ == 1u){
241       SSIF0.SSISR.BIT.TUIRQ = 0u;
242       while(1){
243           /* dead loop */
244       }
245     }
246     /* Transmit overflow error */
247     if(SSIF0.SSISR.BIT.TOIRQ == 1u){
248       SSIF0.SSISR.BIT.TOIRQ = 0u;
249       while(1){
250           /* dead loop */
251       }
252     }
253     /* Receive underflow error */
254     if(SSIF0.SSISR.BIT.RUIRQ == 1u){
255       SSIF0.SSISR.BIT.RUIRQ = 0u;
256       while(1){
257           /* dead loop */
258       }
259     }
260     /* Receive overflow error */
261     if(SSIF0.SSISR.BIT.ROIRQ == 1u){
262       SSIF0.SSISR.BIT.ROIRQ = 0u;
263       while(1){
264           /* dead loop */
265       }
266     }
267   }
268
```

3.7 Sample Program Listing "main.c" (7/13)

```
/*""FUNC COMMENT""*************************************************************
 * ID          :
 * Outline     : DMA transfer initialization
 *-----------------------------------------------------------------------------
 * Include     : #include "iodefine.h"
 *-----------------------------------------------------------------------------
 * Declaration : void io_init_dma1(void *src, void *dst, size_t size);
 *-----------------------------------------------------------------------------
 * Description : Transfers the "size" bytes of data from the source address "src" to
 *             : the destination address "dst" by the DMAC.
 *             : When the DMA transfer is complete, it continues to transmit the
 *             : same data. Enables the DMA transfer end interrupt.
 *             : Specifies the transfer size in units of longword, and the SSI1 as
 *             : transfer source.
 *             : When the transfer size, and source or destination address alignment
 *             : does not match, the operation will not be guaranteed.
 *-----------------------------------------------------------------------------
 * Argument    : void *src   : source address
 *             : void *dst   : destination address
 *             : size_t size : transfer size (in bytes).
 *-----------------------------------------------------------------------------
 * Return Value: void
 *""FUNC COMMENT END""*************************************************************/
void io_init_dma1(void *src, void *dst, size_t size)
{
  /* ---- Sets the Standby control register 2 ---- */
  CPG.STBCR2.BIT.MSTP8 = 0u;             /* DMAC operates */

  /* ---- Disables the DMA transfer ---- */
  DMAC.CHCR1.BIT.DE = 0u;

  /* ---- Sets the DMA source address register ---- */
  /* ---- Sets the DMA reload source address register ---- */
  DMAC.SAR1.LONG = (unsigned long)src;
  DMAC.RSAR1.LONG= (unsigned long)src;

  /* ---- Sets the DMA destination address register ---- */
  DMAC.DAR1.LONG = (unsigned long)dst;

  /* ---- Sets the DMA transfer count register ---- */
  /* ---- Sets the DMA reload transfer count register ---- */
  DMAC.DMATCR1.LONG = size >> 2u;
  DMAC.RDMATCR1.LONG= size >> 2u;

```

3.8 Sample Program Listing "main.c" (8/13)

```
313     /* ---- Sets the DMA channel control register ---- */
314     DMAC.CHCR1.LONG = 0x20101814ul;
315       /*
316            bit31      : TC : 0-------------------- Transmits data once by one request
317            bit30      : reserve 0
318            bit29      : RLDSAR : 1----------------- Enables the SAR reload function
319            bit28      : RLDDAR : 0----------------- Disables the DAR reload function
320            bit27      : reserve 0
321            bit26      : DAF : 0-------------------- Not used
322            bit25      : SAF : 0-------------------- Not used
323            bit24      : reserve 0
324            bit23      : DO : 0--------------------- Not used
325            bit22      : TL : 0--------------------- Not used
326            bit21      : reserve 0
327            bit20      : TEMASK : 1----------------- Continues the DMA transfer when
328                                                     the TE bit is set
329            bit19      : HE : 0--------------------- Not used
330            bit18      : HIE : 0-------------------- Not used
331            bit17      : AM : 0--------------------- Not used
332            bit16      : AL : 0--------------------- Not used
333            bit15 to 14: DM[1:0] : B'00------------- Destination address fixed
334            bit13 to 12: SM[1:0] : B'01------------- Increments the source address
335            bit11 to 8 : RS[3:0] : B'1000----------- Specifies the DMA extension
336                                                     resource selector
337            bit7       : DL : 0--------------------- Not used
338            bit6       : DS : 0--------------------- Not used
339            bit5       : TB : 0--------------------- Specifies the cycle steal mode
340            bit4 to 3  : TS : B'10------------------ Specifies the longword transfer
341            bit2       : IE : 1--------------------- Enables an interrupt request
342            bit1       : TE : 0--------------------- Transfer end flag
343            bit0       : DE : 0--------------------- Disables the DMA transfer
344       */
345     /* ----Sets the DMA extension resource selector 0 ---- */
346     DMAC.DMARS0.BIT.CH1MID = 0x08u;         /* MID = SSI0 */
347     DMAC.DMARS0.BIT.CH1RID = 0x01u;         /* RID = TxD */
348
349     /* ----Sets the DMA operation register---- */
350     DMAC.DMAOR.WORD &= 0xFFF9u;             /* Clears the AE, NMIF bits */
351     DMAC.DMAOR.BIT.DME = 1u;                /* Enables the DMA transfer on all channels */
352
353     /* ---- Sets the interrupt priority level register ---- */
354     INTC.IPR06.BIT._DMAC1 = 1u;
355
356     /* ---- Enables the DMA transfer ---- */
357     DMAC.CHCR1.BIT.DE = 1ul;
358   }
```

3.9 Sample Program Listing "main.c" (9/13)

```
359  /*""FUNC COMMENT""*****************************************************************
360   * ID          :
361   * Outline     : DMA transfer initialization
362   *-------------------------------------------------------------------------------
363   * Include     : #include "iodefine.h"
364   *-------------------------------------------------------------------------------
365   * Declaration : void io_init_dma2(void *src, void *dst, size_t size);
366   *-------------------------------------------------------------------------------
367   * Description : Transfers the "size" bytes of data from the source address "src" to
368   *             : the destination address "dst" by the DMAC.
369   *             : As it continues to transfer data after the DMA transfer is complete,
370   *             : specify the reload register separately.
371   *             : Enables the DMA transfer end interrupt. Specifies the transfer size
372   *             : in units of longword, and the SSI2 as the transfer source.
373   *             : When the transfer size, and source or destination address alignment does
374   *             : not match, the operation will not be guaranteed.
375   *-------------------------------------------------------------------------------
376   * Argument    : void *src   : source address
377   *             : void *dst   : destination address
378   *             : size_t size : transfer size (in bytes).
379   *-------------------------------------------------------------------------------
380   * Return Value: void
381   *""FUNC COMMENT END""*************************************************************/
382  void io_init_dma2(void *src, void *dst, size_t size)
383  {
384    /* ---- Sets the Standby control register 2 ---- */
385    CPG.STBCR2.BIT.MSTP8 = 0u;             /* DMAC operates */
386
387    /* ---- Disables the DMA transfer ---- */
388    DMAC.CHCR2.BIT.DE = 0u;
389
390    /* ---- Sets the DMA source address register ---- */
391    DMAC.SAR2.LONG = (unsigned long)src;
392
393    /* ---- Sets the DMA destination address register ---- */
394    DMAC.DAR2.LONG = (unsigned long)dst;
395
396    /* ---- Sets the DMA transfer count register ---- */
397    DMAC.DMATCR2.LONG = size >> 2u;
398
```

3.10 Sample Program Listing "main.c" (10/13)

```
399     /* ---- Sets the DMA channel control register ---- */
400     DMAC.CHCR2.LONG = 0x10104814ul;
401       /*
402           bit31      : TC : 0-------------------- Transmits data once by one request
403           bit30      : reserve 0
404           bit29      : RLDSAR : 0----------------- Disables the SAR reload function
405           bit28      : RLDDAR : 1----------------- Enables the DAR reload function
406           bit27      : reserve 0
407           bit26      : DAF : 0-------------------- Not used
408           bit25      : SAF : 0-------------------- Not used
409           bit24      : reserve 0
410           bit23      :   DO : 0----------------------  Not used
411           bit22      :   TL : 0----------------------  Not used
412           bit21      : reserve 0
413           bit20      : TEMASK : 1----------------- Continues the DMA transfer when
414                                                     the TE bit is set
415           bit19      : HE : 0-------------------- Not used
416           bit18      : HIE : 0-------------------- Not used
417           bit17      : AM : 0--------------------- Not used
418           bit16      : AL : 0--------------------- Not used
419           bit15 to 14: DM[1:0] : B'01------------- Increments the destination address
420           bit13 to 12: SM[1:0] : B'00------------- Source address fixed
421           bit11 to 8 : RS[3:0] : B'1000----------- Specifies the DMA extension selector
422           bit7       : DL : 0--------------------- Not used
423           bit6       : DS : 0--------------------- Not used
424           bit5       : TB : 0--------------------- Specifies the cycle steal mode
425           bit4 to 3  : TS : B'10------------------ Specifies the longword transfer
426           bit2       : IE : 1--------------------- Enables an interrupt request
427           bit1       : TE : 0--------------------- Transfer end flag
428           bit0       : DE : 0--------------------- Disables the DMA transfer
429       */
430     /* ----Sets the DMA extension resource selector 0 ---- */
431     DMAC.DMARS1.BIT.CH2MID = 0x08u;         /* MID = SSI0 */
432     DMAC.DMARS1.BIT.CH2RID = 0x02u;         /* RID = RxD */
433
434     /* ----Sets the DMA operation register---- */
435     DMAC.DMAOR.WORD &= 0xFFF9u;             /* Clears the AE, NMIF bits */
436     DMAC.DMAOR.BIT.DME = 1u;                /* Enables the DMA transfer on all channels */
437
438     /* ---- Sets the interrupt priority level register ---- */
439     INTC.IPR06.BIT._DMAC2 = 1u;
440     /* ---- Enables the DMA transfer ---- */
441
442     DMAC.CHCR2.BIT.DE = 1ul;                /* Enables the DMA transfer */
443   }
444
```

3.11 Sample Program Listing "main.c" (11/13)

```
/*""FUNC COMMENT""*************************************************************
 * ID          :
 * Outline     : DMA transfer reload setting
 *-----------------------------------------------------------------------------
 * Include     : #include "iodefine.h"
 *-----------------------------------------------------------------------------
 * Declaration : void io_reload_dma2(void *src, void *dst, size_t size);
 *-----------------------------------------------------------------------------
 * Description : Specifies values in the reload source address register, reload destination
 *             : address register, and reload transfer count register.
 *             : Specify the transfer size in units of longword.
 *             : When the transfer size, and source or destination address alignment
 *             : does not match, the operation will not be guaranteed.
 *-----------------------------------------------------------------------------
 * Argument    : void *src   : source address.
 *             : void *dst   : destination address
 *             : size_t size : transfer size (in bytes).
 *-----------------------------------------------------------------------------
 * Return Value: void
 *""FUNC COMMENT END""*********************************************************/
void io_reload_dma2(void *src, void *dst, size_t size)
{
  /* ---- Sets the DMA reload source address register ---- */
  DMAC.RSAR2.LONG= (unsigned long)src;

  /* ---- Sets the DMA reload destination address register ---- */
  DMAC.RDAR2.LONG= (unsigned long)dst;

  /* ---- Sets the DMA reload transfer count register ---- */
  DMAC.RDMATCR2.LONG= size >> 2u;

}
```

3.12 Sample Program Listing "main.c" (12/13)

```
478  /*""FUNC COMMENT""*****************************************************************
479   * ID          :
480   * Outline     : DMA transfer end interrupt
481   *-------------------------------------------------------------------------------
482   * Include     : #include "iodefine.h"
483   *-------------------------------------------------------------------------------
484   * Declaration : void io_int_dma1(void);
485   *-------------------------------------------------------------------------------
486   * Description : When the specified number of the DMA transfers are complete,
487   *             : the SSI transitions to the mute status.
488   *-------------------------------------------------------------------------------
489   * Argument    : void
490   *-------------------------------------------------------------------------------
491   * Return Value: void
492   *""FUNC COMMENT END""*************************************************************/
493  void io_int_dma1(void)
494  {
495    volatile unsigned long dummy;
496
497    if( Count > 0 ){
498      /* ---- Counts the number of transfers ---- */
499      Count--;
500
501      if( Count == 0 ){
502          /* ---- When the specified number of transfers are complete,
503                  it transitions to the mute status (Continues the DMA transfer) ---- */
504          SSIF0.SSICR.BIT.MUEN = 1u;     /* Starts to mute */
505                                         /*  Replaces the data in FIFO with the muted data */
506      }
507    }
508    /* ---- Clears the transfer end flag ---- */
509    DMAC.CHCR1.BIT.TE = 0u;
510    dummy = DMAC.CHCR1.BIT.TE;      /* Dummy read */
511  }
```

3.13 Sample Program Listing "main.c" (13/13)

```
512  /*""FUNC COMMENT""*****************************************************************
513   * ID          :
514   * Outline     : DMA transfer end interrupt
515   *-------------------------------------------------------------------------------
516   * Include     : #include "iodefine.h"
517   *-------------------------------------------------------------------------------
518   * Declaration : void io_int_dma2(void);
519   *-------------------------------------------------------------------------------
520   * Description : Reads all the receive data except the muted data from the receive buffer.
521   *-------------------------------------------------------------------------------
522   * Argument    : void
523   *-------------------------------------------------------------------------------
524   * Return Value: void
525   *""FUNC COMMENT END""*************************************************************/
526  void io_int_dma2(void)
527  {
528    volatile unsigned long dummy;
529    unsigned long rdata;
530    int i;
531
532    /* ---- Updates the reload register ---- */
533    io_reload_dma2((void *)&SSIF2.SSIFRDR, Buff[BuffIdx], SSI_DATASIZE);
534
535    /* ---- Reads the receive data ---- */
536    for(i=0; i<SSI_DATASIZE/sizeof(unsigned long); i++){
537      rdata = Buff[BuffIdx][i];
538
539      if(SSI_MUTEDATA != rdata){
540          RcvData[RcvCnt++] = rdata;
541          if( RcvCnt >= (sizeof(RcvData)/sizeof(unsigned long)) ){
542              RcvCnt = 0u;
543          }
544      }
545    }
546    /* ---- Updates the index of the receive buffer ---- */
547    BuffIdx ^= 1u;
548
549    /* ---- Clears the transfer end flag ---- */
550    DMAC.CHCR2.BIT.TE = 0u;
551    dummy = DMAC.CHCR2.BIT.TE;      /* Dummy read */
552  }
553
554  /* End of File */
555
```

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	Apr.14.09	—	First edition issued
1.01	Feb.23.11	5,19	Corresponds to Technical Update (TN-SH7-A799A/E)

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