To our customers,

# Old Company Name in Catalogs and Other Documents

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

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

April 1st, 2010
Renesas Electronics Corporation

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

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

# Notice

1. All information included in this document is current as of the date this document is issued. Such information, however, is subject to change without any prior notice. Before purchasing or using any Renesas Electronics products listed herein, please confirm the latest product information with a Renesas Electronics sales office. Also, please pay regular and careful attention to additional and different information to be disclosed by Renesas Electronics such as that disclosed through our website.
2. Renesas Electronics does not assume any liability for infringement of patents, copyrights, or other intellectual property rights of third parties by or arising from the use of Renesas Electronics products or technical information described in this document. No license, express, implied or otherwise, is granted hereby under any patents, copyrights or other intellectual property rights of Renesas Electronics or others.
3. You should not alter, modify, copy, or otherwise misappropriate any Renesas Electronics product, whether in whole or in part.
4. Descriptions of circuits, software and other related information in this document are provided only to illustrate the operation of semiconductor products and application examples. You are fully responsible for the incorporation of these circuits, software, and information in the design of your equipment. Renesas Electronics assumes no responsibility for any losses incurred by you or third parties arising from the use of these circuits, software, or information.
5. When exporting the products or technology described in this document, you should comply with the applicable export control laws and regulations and follow the procedures required by such laws and regulations. You should not use Renesas Electronics products or the technology described in this document for any purpose relating to military applications or use by the military, including but not limited to the development of weapons of mass destruction. Renesas Electronics products and technology may not be used for or incorporated into any products or systems whose manufacture, use, or sale is prohibited under any applicable domestic or foreign laws or regulations.
6. Renesas Electronics has used reasonable care in preparing the information included in this document, but Renesas Electronics does not warrant that such information is error free. Renesas Electronics assumes no liability whatsoever for any damages incurred by you resulting from errors in or omissions from the information included herein.
7. Renesas Electronics products are classified according to the following three quality grades: "Standard", "High Quality", and "Specific". The recommended applications for each Renesas Electronics product depends on the product's quality grade, as indicated below. You must check the quality grade of each Renesas Electronics product before using it in a particular application. You may not use any Renesas Electronics product for any application categorized as "Specific" without the prior written consent of Renesas Electronics. Further, you may not use any Renesas Electronics product for any application for which it is not intended without the prior written consent of Renesas Electronics. Renesas Electronics shall not be in any way liable for any damages or losses incurred by you or third parties arising from the use of any Renesas Electronics product for an application categorized as "Specific" or for which the product is not intended where you have failed to obtain the prior written consent of Renesas Electronics. The quality grade of each Renesas Electronics product is "Standard" unless otherwise expressly specified in a Renesas Electronics data sheets or data books, etc.

   "Standard": Computers; office equipment; communications equipment; test and measurement equipment; audio and visual equipment; home electronic appliances; machine tools; personal electronic equipment; and industrial robots.

   "High Quality": Transportation equipment (automobiles, trains, ships, etc.); traffic control systems; anti-disaster systems; anti-crime systems; safety equipment; and medical equipment not specifically designed for life support.

   "Specific": Aircraft; aerospace equipment; submersible repeaters; nuclear reactor control systems; medical equipment or systems for life support (e.g. artificial life support devices or systems), surgical implantations, or healthcare intervention (e.g. excision, etc.), and any other applications or purposes that pose a direct threat to human life.
8. You should use the Renesas Electronics products described in this document within the range specified by Renesas Electronics, especially with respect to the maximum rating, operating supply voltage range, movement power voltage range, heat radiation characteristics, installation and other product characteristics. Renesas Electronics shall have no liability for malfunctions or damages arising out of the use of Renesas Electronics products beyond such specified ranges.
9. Although Renesas Electronics endeavors to improve the quality and reliability of its products, semiconductor products have specific characteristics such as the occurrence of failure at a certain rate and malfunctions under certain use conditions. Further, Renesas Electronics products are not subject to radiation resistance design. Please be sure to implement safety measures to guard them against the possibility of physical injury, and injury or damage caused by fire in the event of the failure of a Renesas Electronics product, such as safety design for hardware and software including but not limited to redundancy, fire control and malfunction prevention, appropriate treatment for aging degradation or any other appropriate measures. Because the evaluation of microcomputer software alone is very difficult, please evaluate the safety of the final products or system manufactured by you.
10. Please contact a Renesas Electronics sales office for details as to environmental matters such as the environmental compatibility of each Renesas Electronics product. Please use Renesas Electronics products in compliance with all applicable laws and regulations that regulate the inclusion or use of controlled substances, including without limitation, the EU RoHS Directive. Renesas Electronics assumes no liability for damages or losses occurring as a result of your noncompliance with applicable laws and regulations.
11. This document may not be reproduced or duplicated, in any form, in whole or in part, without prior written consent of Renesas Electronics.
12. Please contact a Renesas Electronics sales office if you have any questions regarding the information contained in this document or Renesas Electronics products, or if you have any other inquiries.

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

(Note 2) "Renesas Electronics product(s)" means any product developed or manufactured by or for Renesas Electronics.

# SH7206 Group

## Example of Transferring Data to an On-Chip Module Using the DMAC

### Introduction

This application note describes an example of data transfer to an on-chip module using the direct memory access controller (DMAC) in the SH7206.

### Target Device

SH7206

### Contents

# 1. Overview

## 1.1 Specifications

- Using the DMAC channel 1, data is transferred from external memory to the transmit FIFO data register (SCFTDR) of the serial communication interface with FIFO (SCIF channel 0), and thus character strings are sent.
- The transmit-FIFO-data-empty transfer request (a request from on-chip peripheral module) of the SCIF is used as the DMA transfer request.

## 1.2 MCU Functions Used

- Direct memory access controller (DMAC channel 1)
- Serial communication interface with FIFO (SCIF channel 0)

## 1.3 Conditions for Application

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

## 1.4 Related Application Note

- The operation of the sample program in this application note was confirmed with the configuration specified in the application note "Example of SH7206 Initial Configuration". Please refer to that note in combination with this one.
- Details on SCIF UART transmission are described in the SH7206 application note "Example Settings for UART Transmission by the SCIF".

## 2. Description of Sample Application

This sample application applies the direct memory access controller (DMAC) to transfer of data from external memory to the SCIF in response to an on-chip peripheral module request.

### 2.1 Summary of MCU Functions Used

When DMA transfer requests are generated, the DMAC initiates transfer in accordance with the priority levels assigned to its channels, and terminates the transfer when the transfer-end conditions are satisfied. There are three transfer request modes: auto request, external request and on-chip peripheral module request. Either burst mode or cycle-steal mode can be selected as the bus mode.

Table 1 summarizes the features of the DMAC. Figure 1 is a schematic view of the DMAC.

**Table 1 Summary of the DMAC**

| Item | Function |
|---|---|
| Number of channels | 8 (CH0 to CH7)<br>Of these, four channels CH0 to CH3 can receive external requests. |
| Address space | 4 Gbytes |
| Transfer data sizes | Byte, word (2 bytes), longword (4 bytes), and 16 bytes (longword × 4) |
| Maximum transfer count | 16,777,216 (24 bits) |
| Address modes | Single-address mode and dual-address mode |
| Transfer requests | Auto request, external request, and on-chip peripheral module request<br>(SCIF: 8 sources, IIC3: 2 sources, ADC: 2 sources, MTU2: 5 sources, CMT: 2 sources) |
| Bus modes | Cycle-steal mode and burst mode |
| Channel priority | Fixed mode and round-robin mode |
| Interrupt request | An interrupt is requested to the CPU upon completion of half- or full-data transfer. |
| External request detection | Detection of low or high level of the DREQ input, or rising or falling edge of the DREQ input |
| Transfer request acknowledge signal/transfer end signal | Selectable active levels of DACK and TEND signals |

Note: For details on the DMAC, refer to section 9, Direct Memory Access Controller, of the SH7206 Group Hardware Manual.

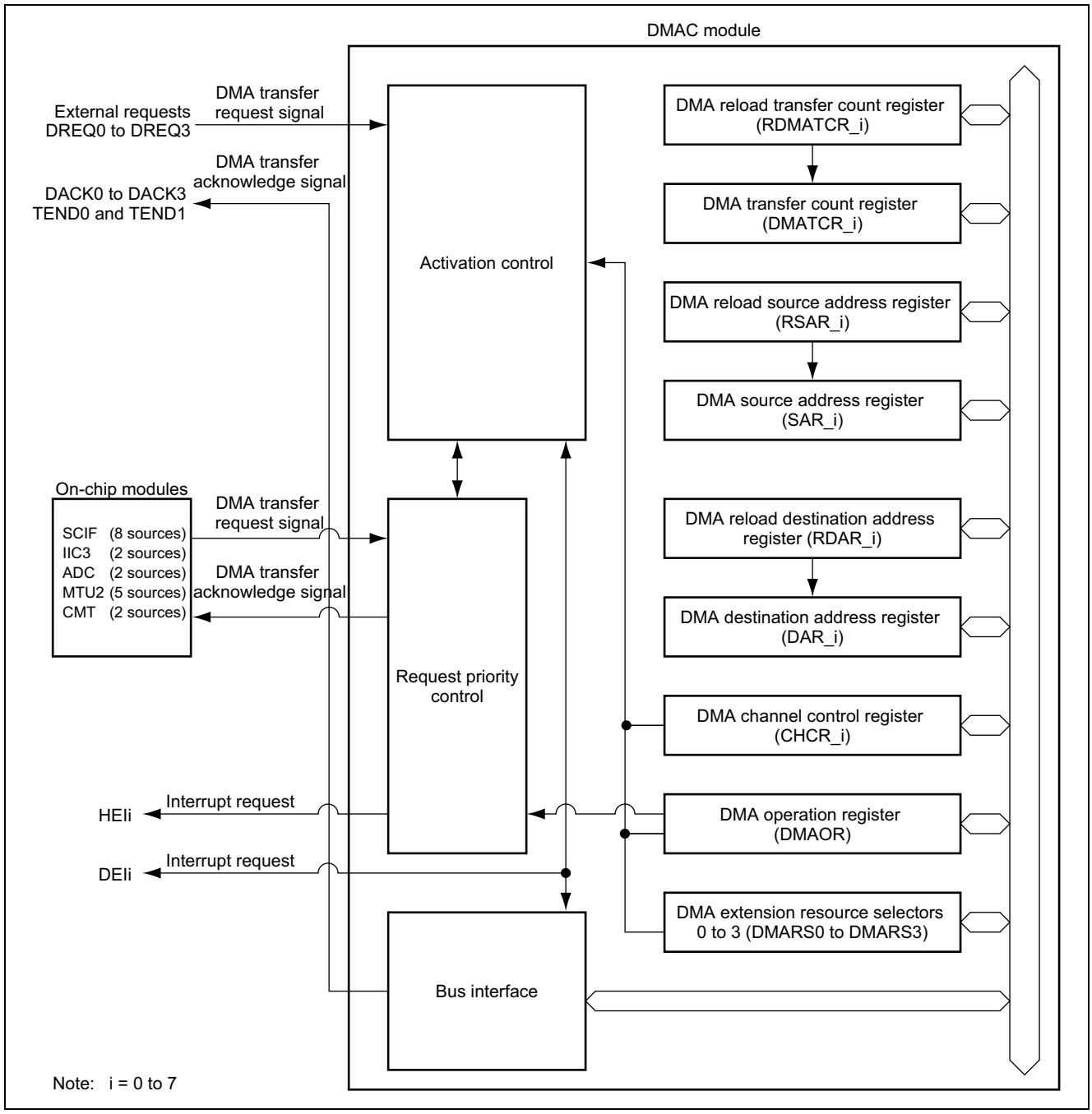


**Figure 1 Schematic View of the DMAC**

## 2.2 Procedure for Setting the MCU Modules

This section describes the initial setting procedure for data transfer from memory to the on-chip peripheral module by the DMAC in response to a transfer request form an on-chip peripheral module. Figure 2 shows an example flow of making initial settings of the DMAC. For details on the settings of individual registers, refer to the SH7206 Group Hardware Manual.

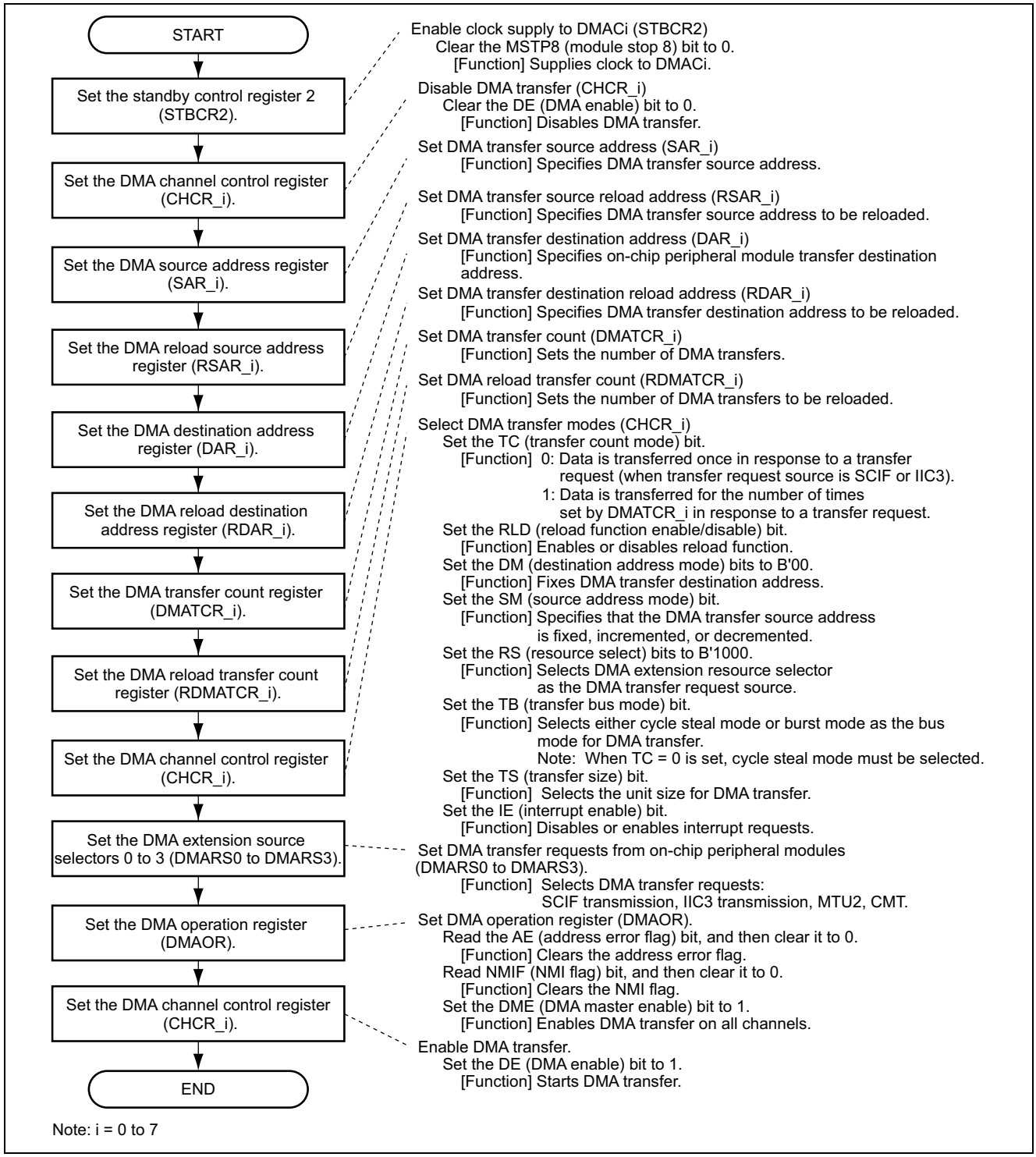


**Figure 2 Example Flow for Initial Settings of the DMAC**

## 2.3 Operation of Sample Program

In the sample program, channel 1 of the DMAC is activated in response to a transmit-FIFO-data-empty transfer request (on-chip peripheral module request) from the SCIF, and data are transferred from external memory to the transmit FIFO data register (SCFTDR) of the SCIF channel 0. The data written to the SCFTDR register is transmitted in UART mode.

Figure 3 illustrates the timing of the sample program operation.

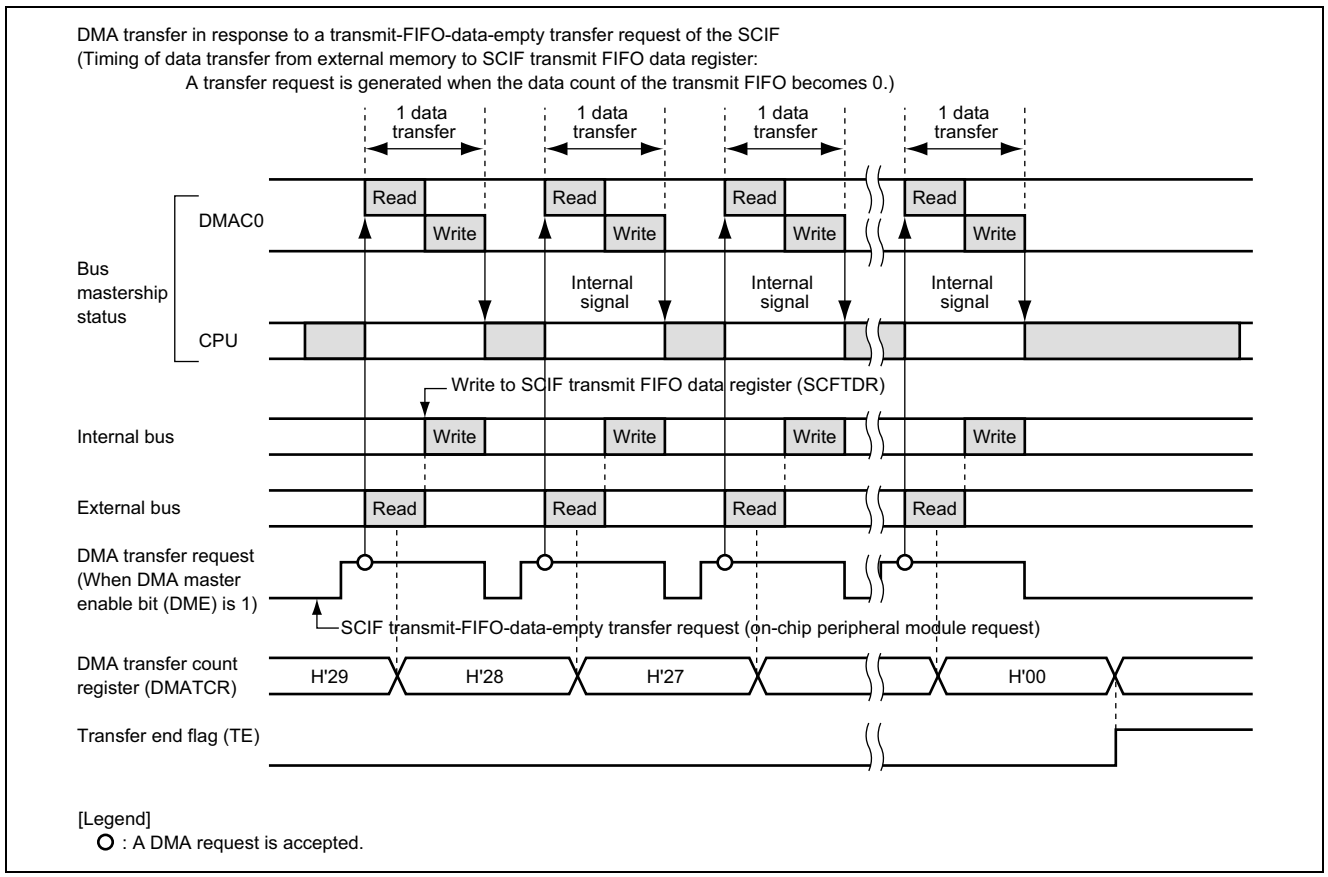


**Figure 3 Timing of Sample Program Operation**

## 2.4 Register Settings and Processing Sequence of Sample Program

In the sample program, character string data stored in external memory is transferred to the transmit FIFO data register (SCFTDR) of the SCIF channel 0 by DMA transfer. The data is then transmitted in UART mode.

The register settings of the sample program are shown in table 2, macro definitions used in the sample program is listed in table 3, and processing flow of the sample program is shown in figure 4.

**Table 2 Register Settings in the Sample Program**

| Register Name | Address | Setting | Description |
|---|---|---|---|
| Standby control register 2 (STBCR2) | H'FFFE0018 | H'00 | MSTP8 = 0: The DMAC runs. |
| DMA channel control register_1 (CHCR_1) | H'FFFE101C | H'00000000 | DE = 0: DMA transfer is disabled. |
| | | H'00001800 | TC = 1: Transfer once in response to each DMA request.<br>RLD = 0: Reload function is disabled.<br>DM = B'00: Destination address is fixed.<br>SM = B'01: Source address is incremented.<br>RS = B'1000: Extension resource selector<br>TB = 0: Cycle-steal mode<br>TS = B'00: Byte transfer<br>IE = 0: Interrupt requests are disabled. |
| | | H'00001801 | DE = 1: DMA transfer is enabled |
| DMA source address register_1 (SAR_1) | H'FFFE1010 | Address of character string data | Transfer source start address: Start address of character strings stored in external memory. |
| DMA destination address register_1 (DAR_1) | H'FFFE1014 | H'FFFE800C | Transfer destination start address: Address of the SCIF transmit FIFO data register_1 (SCFTDR_1). |
| DMA transfer count register_1 (DMATCR_1) | H'FFFE1014 | Number of bytes in character string data | Transfer count: Number of bytes in the character string data |
| DMA extension resource selector (DMARS0) | H'FFFE1300 | H'0081 | MID = B'100000<br>RID = B'01: A transmit-FIFO-data-empty transfer request of SCIF_0 is specified. |
| DMA operation register (DMAOR) | H'FFFE1200 | H'0001 | DME = 1: DMA transfer is enabled on all channels. |

**Table 3 Macro Definitions Used in the Sample Program**

| Macro Definition | Setting | Description |
|---|---|---|
| DMA_SIZE_BYTE | H'0000 | Byte transfer |
| DMA_SIZE_WORD | H'0001 | Word transfer |
| DMA_SIZE_LONG | H'0002 | Longword transfer |
| DMA_SIZE_LONGx4 | H'0003 | 16-byte transfer |
| DMA_INT_DISABLE | H'0000 | DMA transfer end interrupt is not used |
| DMA_INT_ENABLE | H'0010 | DMA transfer end interrupt is used |

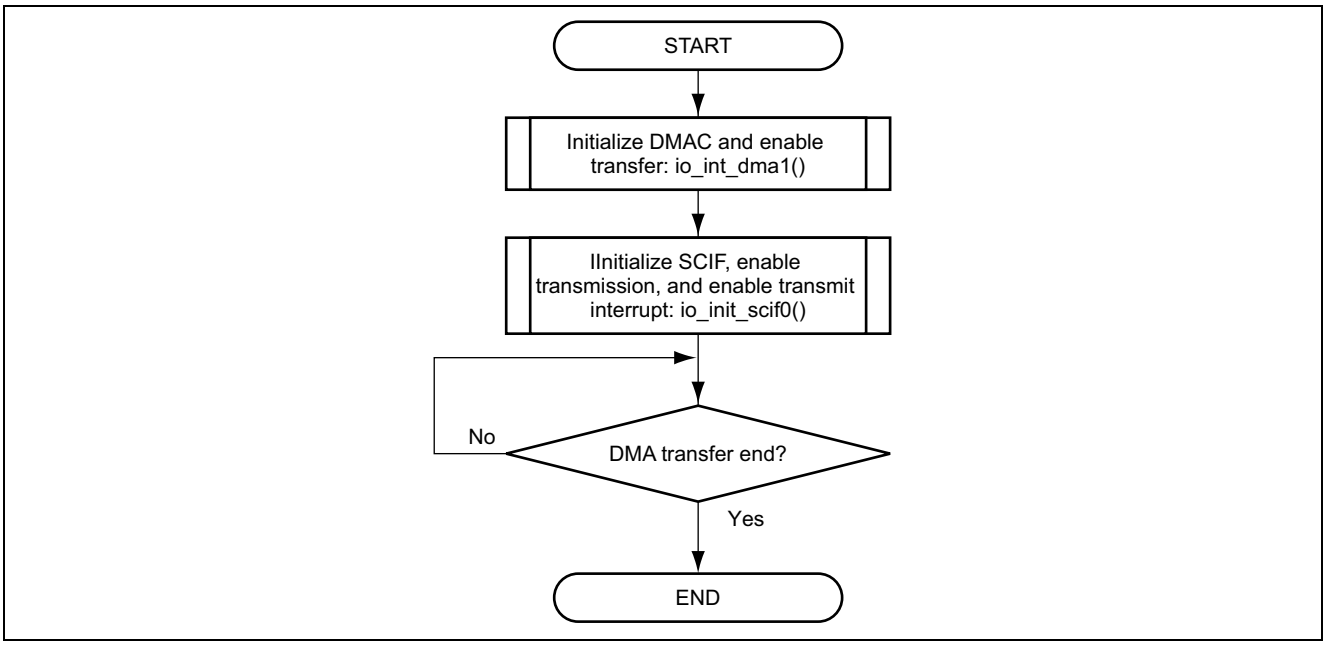


**Figure 4 Processing Flow of the Sample Program**

## 3. Sample Program Listing

1. Sample Program Listing: main.c (1)

```
/*""FILE COMMENT""*********************************************************************
*
*   System Name: SH7206 Sample Program
*   File Name  : main.c
*   Version    : 1.00.00
*   Contents   : DMAC sample program
*   Model      : M3A-HS60
*   CPU        : SH7206
*   Compiler   : SHC9.0.00
*
*   Note       : Sample program for transferring data from the SCIF by DMAC1
*
*               <Caution>
*               This sample program is for reference
*               and its operation is not guaranteed.
*               Customers should use this sample program for technical reference
*               in software development.
*
*   COPYRIGHT (C) 2004 RENESAS TECHNOLOGY CORP. ALL RIGHTS RESERVED
*   AND RENESAS SOLUTIONS CORP. ALL RIGHTS RESERVED
*
*   history   : 2004.10.28 ver.1.00.00
*""FILE COMMENT END""*****************************************************************/
#include <string.h>
#include "iodefine.h"                   /* iodefine.h is automatically created by HEW      */

/* ==== Macro declarations ==== */
/* ==== DMAC setting ==== */
#define DMA_SIZE_BYTE    0x0000u
#define DMA_SIZE_WORD    0x0001u
#define DMA_SIZE_LONG    0x0002u
#define DMA_SIZE_LONGx4  0x0003u
#define DMA_INT_DISABLE  0x0000u
#define DMA_INT_ENABLE   0x0010u
#define DMA_INT          (DMA_INT_ENABLE >> 4u)

/* ==== Prototype declarations ==== */
void main(void);
void io_init_dma1 (void *src, void *dst, size_t size, unsigned int mode);
void io_dma1_stop (void);
void io_init_scif0(int);

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

/* ---- Values for baud rate specification ---- */
enum{
    CBR_1200,
    CBR_2400,
    CBR_4800,
    CBR_9600,
    CBR_19200,
    CBR_31250,
    CBR_38400,
    CBR_57600,
    CBR_115200
};

```

2. Sample Program Listing: main.c (2)

```
 63  /* ==== Table of register setting values ==== */
 64  static SH7206_BAUD_SET scif_baud[] = {
 65      {214, 1},        /*  1200bps (-0.07%) */
 66      {106, 1},        /*  2400bps ( 0.39%) */
 67      {214, 0},        /*  4800bps (-0.07%) */
 68      {106, 0},        /*  9600bps ( 0.39%) */
 69      { 53, 0},        /* 19200bps (-0.54%) */
 70      { 32, 0},        /* 31250bps ( 0.00%) */
 71      { 26, 0},        /* 38400bps (-0.54%) */
 72      { 17, 0},        /* 57600bps (-0.54%) */
 73      {  8, 0}         /*115200bps (-0.54%) */
 74  };
 75
 76  /* Character string to be transmitted */
 77  const signed char data[] = "SCIF request DMAC Sample Software SH7206.\r\n";
 78
 79  /*""FUNC COMMENT""*****************************************************************************
 80  * ID           :
 81  * Module summary: Main function of the sample program (UART transmission with use of DMAC)
 82  *-----------------------------------------------------------------------------------------------
 83  * Include      : #include <string.h>
 84  *-----------------------------------------------------------------------------------------------
 85  * Declaration  : void main(void)
 86  *-----------------------------------------------------------------------------------------------
 87  * Functional description:
 88  *              : The character string data stored in external memory is DMA transferred
 89  *              : to the SCIF transmit FIFO data register. The DMAC is activated
 90  *              : by an SCIF transmit interrupt request.
 91  *-----------------------------------------------------------------------------------------------
 92  * Argument     : None
 93  *-----------------------------------------------------------------------------------------------
 94  * Return value : None
 95  *-----------------------------------------------------------------------------------------------
 96  * Notes        :
 97  *""FUNC COMMENT END""*************************************************************************/
 98  void main(void)
 99  {
100      /* ==== Enabling DMAC initialization/transfer ==== */
101      io_init_dma1(data, (void *)&SCIF0.SCFTDR.BYTE ,sizeof(data),
102      DMA_SIZE_BYTE | DMA_INT_DISABLE);
103                   /* On-chip peripheral module request (SCIF transmit interrupt request) */
104                   /* Data transfer from external memory to SCIF transmit                 */
105                   /* FIFO data register                                                  */
106
107      /* ==== Initialization of SCIF0/enabling transmission ==== */
108      io_init_scif0(CBR_115200);
109                   /* Communication mode: UART mode                                       */
110                   /* Bit rate: 115.2 kbps                                                */
111                   /* TXI interrupt is generated when data in transmit FIFO is one byte   */
112
113      /* ==== Disabling DMA transfer ==== */
114      io_dma1_stop();
115
116      while(1){
117          /* End of program */
118      }
119
120  }
```

3. Sample Program Listing: main.c (3)

```
/*""FUNC COMMENT""*****************************************************************************
* ID            :
* Module summary: Initial settings for DMA transfer
*-----------------------------------------------------------------------------------------------
* Include       : #include "iodefine.h"
*-----------------------------------------------------------------------------------------------
* Declaration   : io_init_dma1(void *src, void *dst, size_t size, int mode)
*-----------------------------------------------------------------------------------------------
* Functional description:
*               : Transfers 'size'-byte data from source address 'src' to destination
*               : address 'dst' by the DMAC. The transfer is triggered by a request from
*               : SCIF0. Transfer size and use of interrupt are specified in 'mode'.
*-----------------------------------------------------------------------------------------------
* Arguments     : void *src         : Source address
*               : void *dst         : Destination address
*               : size_t size       : Transfer size (byte)
*               : unsigned int mode : Transfer mode; following modes are specified by
*                                     logical OR.
*               :         DMA_SIZE_BYTE(0x0000) Byte transfer
*               :         DMA_SIZE_WORD(0x0001) Word transfer
*               :         DMA_SIZE_LONG(0x0002) Longword transfer
*               :         DMA_SIZE_LONGx4(0x0003) 16-byte transfer
*               :         DMA_INT_DISABLE(0x0000) DMA transfer-end interrupt is not used.
*               :         DMA_INT_ENABLE(0x0010)  DMA transfer-end interrupt is used.
*-----------------------------------------------------------------------------------------------
* Return valu   : None
*-----------------------------------------------------------------------------------------------
* Note          : Correct operation cannot be guaranteed if the transfer data size
*                : does not agree with alignment of source and destination addresses.
*                : When an interrupt is used, the corresponding interrupt routine must be
prepared.
*""FUNC COMMENT END""*************************************************************************/
void io_init_dma1(void *src, void *dst, size_t size, unsigned int mode)
{
    unsigned int ts;
    unsigned long ie;

    ts = mode & 0x3u;
    ie = (mode & 0x00f0u ) >> 4u;

    /* ==== Setting standby control register 2 (STBCR2) ==== */
    CPG.STBCR2.BIT.MSTP8 = 0x0;              /* Cancel DMAC module stop mode           */

    /* ---- Setting DMA channel control register ---- */
    DMAC.CHCR1.BIT.DE = 0ul;                 /* Disable DMA transfer                   */

    /* ---- Setting DMA source address register ---- */
    DMAC.SAR1.LONG = (unsigned long)src;

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

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

    /* ---- Setting DMA reload destination address register ---- */
    DMAC.RDAR1.LONG = (unsigned long)dst;
```

5. Sample Program Listing: main.c (4)

```
178      /* ---- Setting DMA transfer count register ---- */
179      /* ---- Setting DMA reload transfer count register ---- */
180
181      if(ts != DMA_SIZE_LONGx4){
182          DMAC.DMATCR1.LONG = size >> ts;        /* Set the number of transfers           */
183          DMAC.RDMATCR1.LONG = size >> ts;
184      }
185      else{
186          DMAC.DMATCR1.LONG = size >> 4u;        /* Set the number of transfers (1/16)    */
187          DMAC.RDMATCR1.LONG = size >> 4u;
188      }
189
190      /* ---- Setting DMA channel control register ---- */
191      DMAC.CHCR1.LONG = 0x00001800ul |  (mode << 3u) | (ie << 2u) ;
192          /*
193              bit31   : TC DMATCR transfer: 0--------- Transfer once
194              bit30-29: reserve 0
195              bit28   : RLD OFF : 0------------------- Disable reload function
196              bit27-24: reserve 0
197              bit23   : DO over run0 : 0-------------- Unused
198              bit22   : TL TEND low active : 0-------- Unused
199              bit21-20: reserve 0
200              bit19   : HE  :0------------------------ Unused
201              bit18   : HIE :0------------------------ Unused
202              bit17   : AM  :0------------------------ Unused
203              bit16   : AL  :0------------------------ Unused
204              bit15-14: DM1 :0 DM0:0------------------ Fix destination address
205              bit13-12: SM1 :0 SM0:1------------------ Increment source address
206              bit11-8 : RS  : auto request : B'1000--- DMA extension resource selector
207              bit7    : DL : DREQ level : 0 ---------- Unused
208              bit6    : DS : DREQ select :0 Low level- Unused
209              bit5    : TB :cycle :0------------------ Cycle steal mode
210              bit4-3  : TS : transfer size: B'00------ Byte transfer
211              bit2    : IE : interrupt enable: 0------ Disable interrupt
212              bit1    : TE : transfer end: 0
213              bit0    : DE : DMA enable bit: 0-------- Disable DMA transfer
214          */
215      /* ---- Setting DMA extension resource selector 0 ---- */
216      DMAC.DMARS0.BIT.CH1MID = 0x20;             /* MID = SCIF0 */
217      DMAC.DMARS0.BIT.CH1RID = 0x01;             /* RID = Transmission                    */
218
219      /* ---- Setting DMA operation register ---- */
220      DMAC.DMAOR.WORD &= 0xfff9u;                /* Clear AE, NMIF bits                   */
221
222      if(DMAC.DMAOR.BIT.DME == 0ul){             /* Enable DMA transfer on all channels   */
223          DMAC.DMAOR.BIT.DME = 1ul;
224      }
225
226      /* ---- DMA transfer execution ---- */
227      DMAC.CHCR1.BIT.DE = 1ul;                   /* Enable DMA transfer                   */
228
229  }
230
```

5. Sample Program Listing: main.c (5)

```
231  /*""FUNC COMMENT""*****************************************************************************
232  * ID            :
233  * Module summary: Stopping DMAC
234  *-----------------------------------------------------------------------------------------------
235  * Include       : #include "iodefine.h"
236  *-----------------------------------------------------------------------------------------------
237  * Declaration   : void io_dma1_stop(void)
238  *-----------------------------------------------------------------------------------------------
239  * Functional description: Detects the end of DMA transfer and disables DMA transfer.
240  *-----------------------------------------------------------------------------------------------
241  * Argument      : None
242  *-----------------------------------------------------------------------------------------------
243  * Return value  : None
244  *-----------------------------------------------------------------------------------------------
245  * Notes         :
246  *""FUNC COMMENT END""*************************************************************************/
247  void io_dma1_stop(void)
248  {
249      /* Detecting end of transfer */
250      while(DMAC.CHCR1.BIT.TE == 0ul){
251                                              /* Wait until the TE bit is set              */
252      }
253
254      /* ---- Stopping DMA transfer ---- */
255      DMAC.CHCR1.BIT.DE = 0ul;                /* Disable DMA1 transfer                     */
256  }
257
```

6. Sample Program Listing: main.c (6)

```
/*""FUNC COMMENT""*****************************************************************************
* ID            :
* Module summary: Initial setting of SCIF0 as an asynchronous (UART) transmit module
*-------------------------------------------------------------------------------------------
* Include       : #include "iodefine.h"
*-------------------------------------------------------------------------------------------
* Declaration   : void io_init_scif0(int bps)
*-------------------------------------------------------------------------------------------
* Functional description: Initializes SCIF0
*               : Asynchronous (UART)/ 8 bits / No parity/ 1 stop bit/ RTS/CTS disabled
*               : Baud rate is specified by argument bps
*-------------------------------------------------------------------------------------------
* Argument      : int bps : Value for baud rate specification
*-------------------------------------------------------------------------------------------
* Return value  : None
*-------------------------------------------------------------------------------------------
* Notes         : The baud rate setting values given in this program are those when
*               : the peripheral module clock (Pf) frequency is 33 MHz. If a different
*               : clock is used, the baud rate setting values must be changed.
*""FUNC COMMENT END""*************************************************************************/
void io_init_scif0(int bps)
{
    /* ==== Power-down mode cancellation ==== */
    /* ---- Setting Standby control register 4 (STBCR4) ---- */
    CPG.STBCR4.BIT.MSTP47 = 0;                /* Start clock supply to SCIF0             */

    /* ==== SCIF0 initialization ==== */
    /* ---- Setting serial control register (SCSCRi) ---- */
    SCIF0.SCSCR.WORD = 0x0000;                /* Stop transmission/reception by SCIF0    */

    /* ---- Setting FIFO control register (SCFCRi) ---- */
    SCIF0.SCFCR.BIT.TFRST = 1;                /* Reset transmit FIFO                     */

    /* ---- Setting serial control register (SCSCRi) ---- */
    SCIF0.SCSCR.BIT.CKE = 0x0;                /* B'00: Internal clock                    */

    /* ---- Setting 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: Disable addition and check */
                                    /* Parity mode         0: Even parity                */
                                    /* Stop bit length     0: 1 stop bit                 */
                                    /* Clock select         : Table value                */

    /* ---- Setting bit rate register (SCBRRi) ---- */
    SCIF0.SCBRR.BYTE = scif_baud[bps].scbrr;

    /* ---- Setting FIFO control register (SCFCRi) ---- */
    SCIF0.SCFCR.WORD = 0x0030;      /* Transmit FIFO data count trigger:                 */
                                    /*                       Number of data bytes = 0    */
                                    /* Modem control enable             : Disabled       */
                                    /* Transmit FIFO data register reset : Disabled      */
                                    /* Loopback test                     : Disabled      */

    /* ==== Setting pin function controller (PFC) ==== */
    PORT.PACRL1.BIT.PA1MD = 1;      /* Enable TxD0 pin (PACRL1)                          */

    /* ---- Setting serial control register (SCSCRi) ---- */
    SCIF0.SCSCR.BIT.TIE = 1;        /* Enable SCIF0 transmit interrupt                   */
    SCIF0.SCSCR.BIT.TE = 1;         /* Enable SCIF0 transmission                         */

}
/* End of File */
```

## 4. Reference Documents

- SH-2A SH2A-FPU Software Manual (Rev.3.00)
  (Download the latest edition from the website of Renesas Technology Corp.)
- SH7206 Group Hardware Manual (Rev. 1.00)
  (Download the latest edition from the website of Renesas Technology Corp.)

## 5. Website

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

## Revision Record

| | | Description | |
|---|---|---|---|
| Rev. | Date | Page | Summary |
| 1.00 | Sep.05.05 | — | First edition issued |
| | | | |
| | | | |
| | | | |
| | | | |

## Keep safety first in your circuit designs!

1. Renesas Technology Corp. puts the maximum effort into making semiconductor products better and more reliable, but there is always the possibility that trouble may occur with them. Trouble with semiconductors may lead to personal injury, fire or property damage.
   Remember to give due consideration to safety when making your circuit designs, with appropriate measures such as (i) placement of substitutive, auxiliary circuits, (ii) use of nonflammable material or (iii) prevention against any malfunction or mishap.

## Notes regarding these materials

1. These materials are intended as a reference to assist our customers in the selection of the Renesas Technology Corp. product best suited to the customer's application; they do not convey any license under any intellectual property rights, or any other rights, belonging to Renesas Technology Corp. or a third party.
2. Renesas Technology Corp. assumes no responsibility for any damage, or infringement of any third-party's rights, originating in the use of any product data, diagrams, charts, programs, algorithms, or circuit application examples contained in these materials.
3. All information contained in these materials, including product data, diagrams, charts, programs and algorithms represents information on products at the time of publication of these materials, and are subject to change by Renesas Technology Corp. without notice due to product improvements or other reasons. It is therefore recommended that customers contact Renesas Technology Corp. or an authorized Renesas Technology Corp. product distributor for the latest product information before purchasing a product listed herein.
   The information described here may contain technical inaccuracies or typographical errors.
   Renesas Technology Corp. assumes no responsibility for any damage, liability, or other loss rising from these inaccuracies or errors.
   Please also pay attention to information published by Renesas Technology Corp. by various means, including the Renesas Technology Corp. Semiconductor home page (http://www.renesas.com).
4. When using any or all of the information contained in these materials, including product data, diagrams, charts, programs, and algorithms, please be sure to evaluate all information as a total system before making a final decision on the applicability of the information and products. Renesas Technology Corp. assumes no responsibility for any damage, liability or other loss resulting from the information contained herein.
5. Renesas Technology Corp. semiconductors are not designed or manufactured for use in a device or system that is used under circumstances in which human life is potentially at stake. Please contact Renesas Technology Corp. or an authorized Renesas Technology Corp. product distributor when considering the use of a product contained herein for any specific purposes, such as apparatus or systems for transportation, vehicular, medical, aerospace, nuclear, or undersea repeater use.
6. The prior written approval of Renesas Technology Corp. is necessary to reprint or reproduce in whole or in part these materials.
7. If these products or technologies are subject to the Japanese export control restrictions, they must be exported under a license from the Japanese government and cannot be imported into a country other than the approved destination.
   Any diversion or reexport contrary to the export control laws and regulations of Japan and/or the country of destination is prohibited.
8. Please contact Renesas Technology Corp. for further details on these materials or the products contained therein.