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 Memory-to-Memory Transfer by the DMAC

Introduction

This application note describes an example of data transfer between memories using the direct memory access controller (DMAC) in the SH7206.

Target Device

SH7206

Contents

1. Overview

1.1 Specifications

- Using the DMAC channel 0, data in on-chip RAM are transferred to external memory in cycle-steal mode.
- Auto-request mode (transfer request by software) is used to request DMA transfers.

1.2 MCU Functions Used

- Direct memory access controller (DMAC 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.

2. Description of Sample Application

This sample program applies the direct memory access controller (DMAC) to transfer of data from on-chip RAM to external memory.

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) 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 (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.

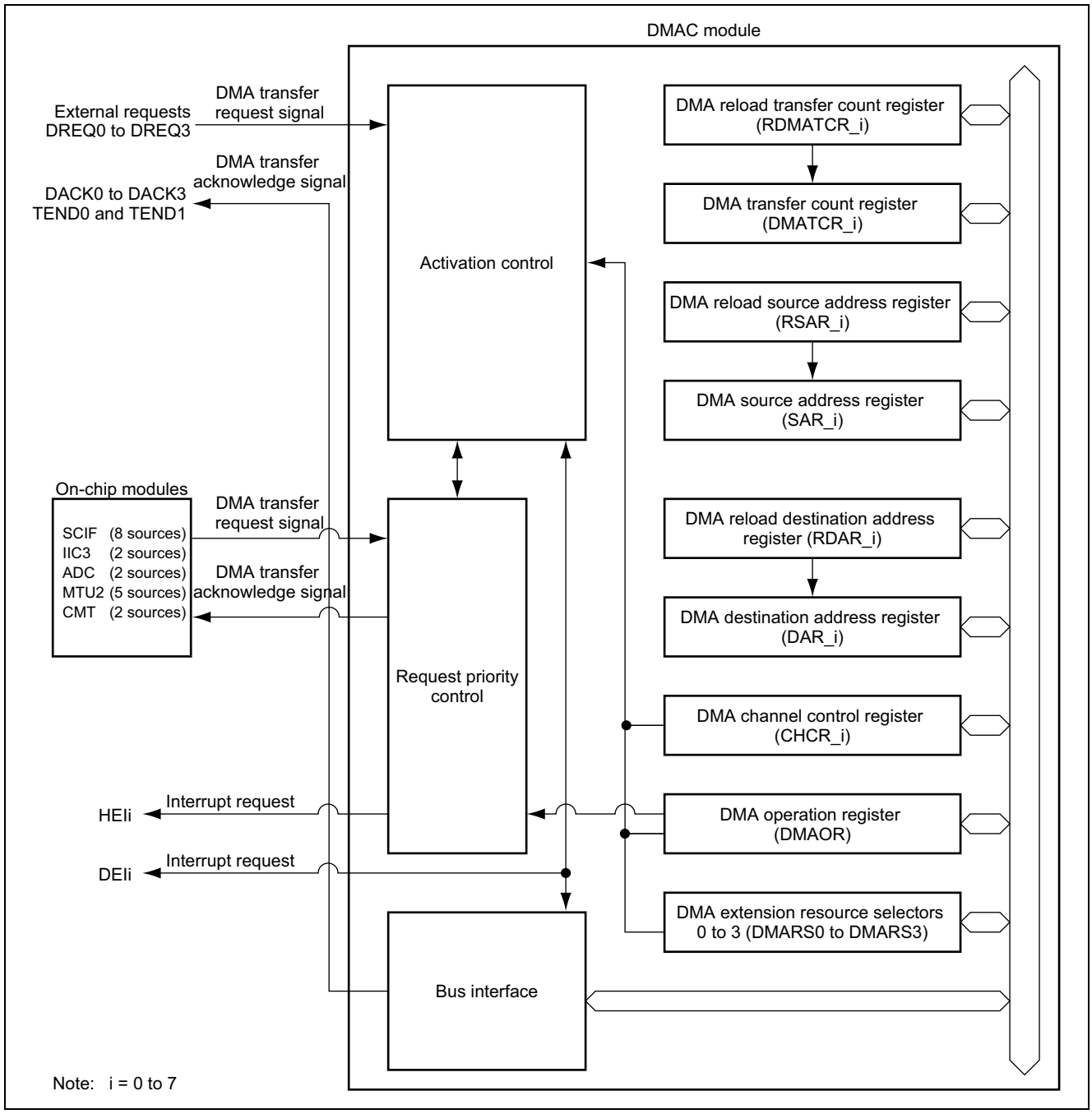

Figure 1 Schematic View of the DMAC

2.2 Procedure for Setting the MCU Modules

This section describes the initial setting procedure for memory-to-memory transfer by the DMAC in response to an auto request. 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.

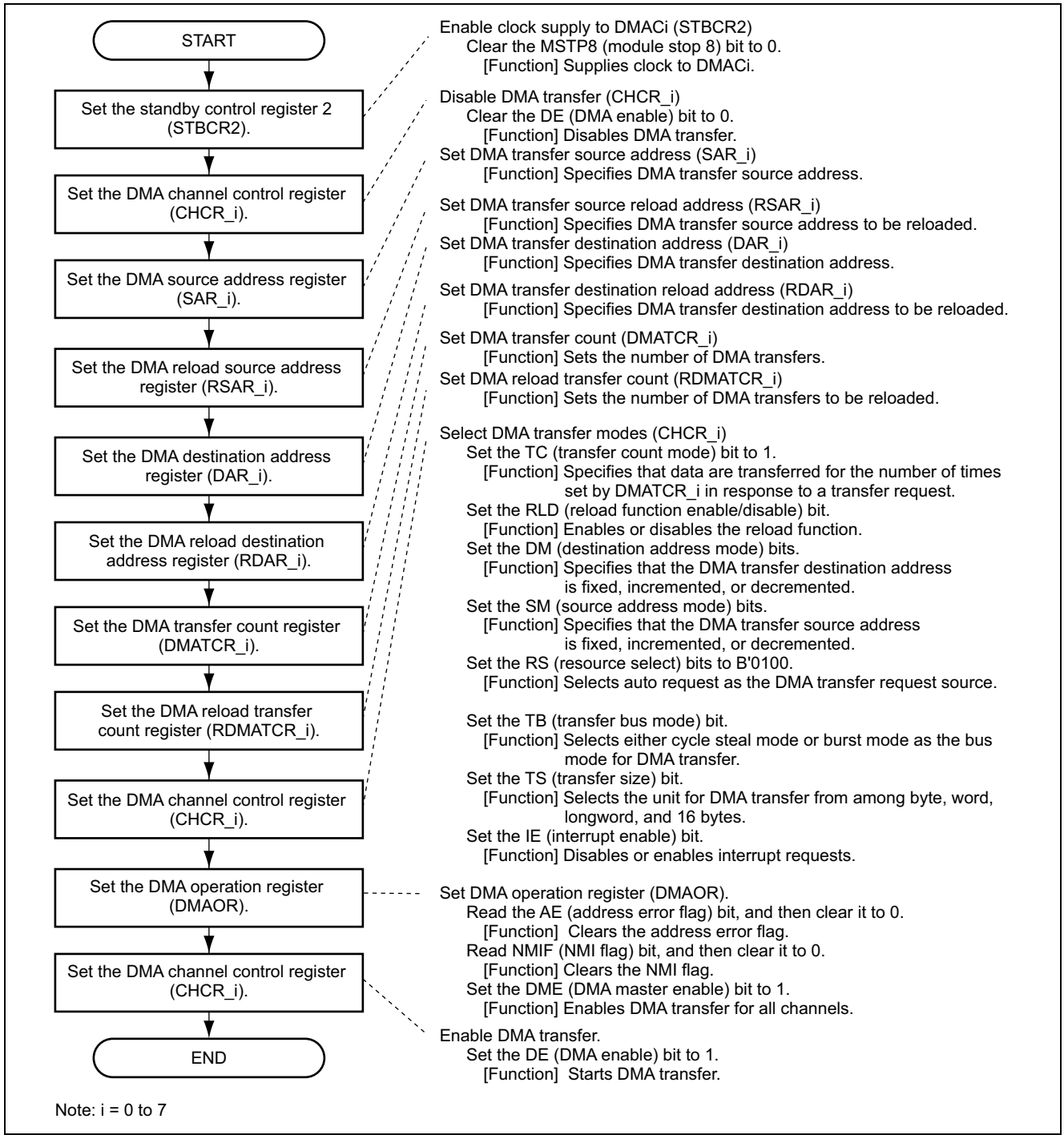

Figure 2 Example Flow for Initial Settings of the DMAC

2.3 Operation of Sample Program

In the sample program, channel 0 of the DMAC is activated in response to the auto request, and data are transferred from on-chip RAM to external memory in cycle-steal mode. In cycle-steal mode, the DMAC releases the bus mastership to the CPU after each data transfer.

Figure 3 illustrates the timing of the sample program operation.

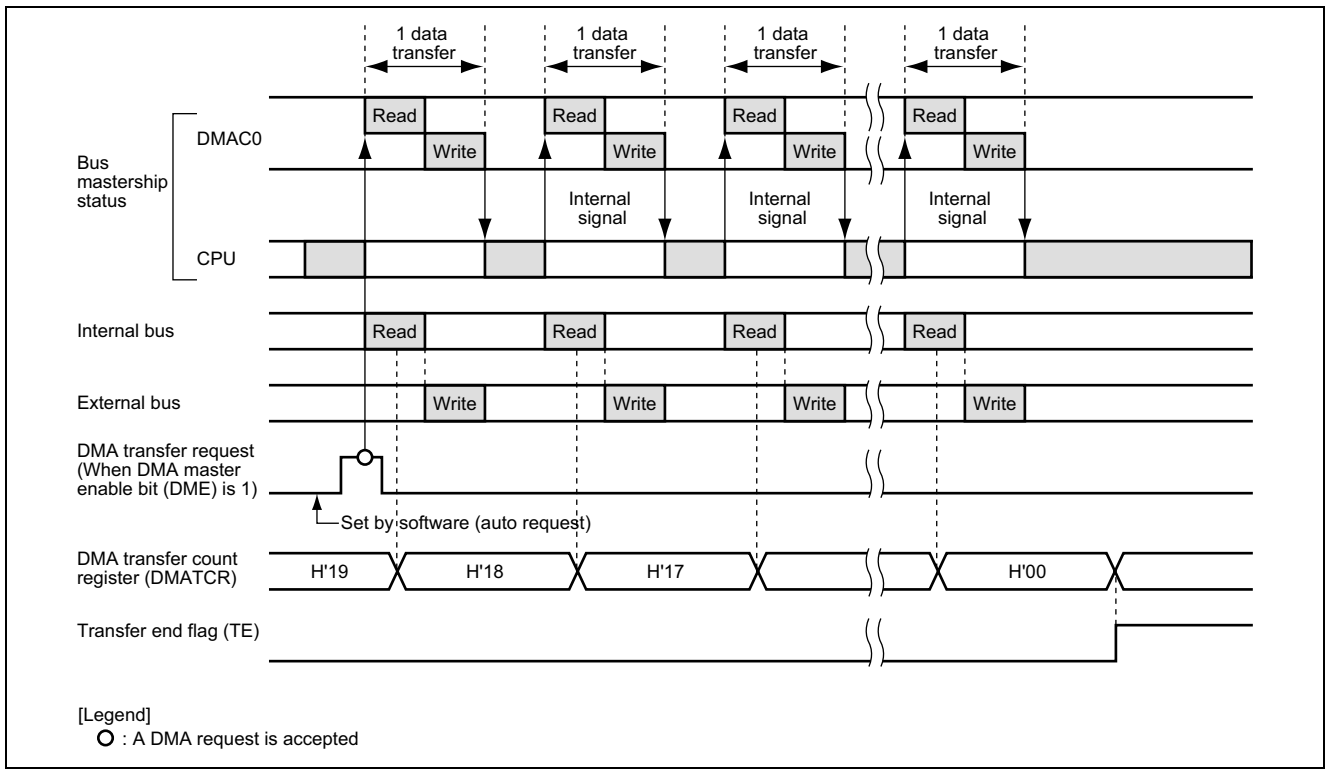

Figure 3 Timing of Sample Program Operation

2.4 Notes on Sample Program Usage

- In the sample program, absolute addresses are used to clarify the start addresses of the data transfer source and destination. When allocating memory areas by absolute addresses, be careful so that they do not overlap with the sections used by other programs.
- In DMA transfer with the operand cache enabled, coherency must be kept by disabling or writing back the cache. In the sample program, coherency is kept because non-cacheable space is accessed from the CPU.

2.5 Register Settings and Processing Sequence of Sample Program

In the sample program, the 100 bytes of data stored in on-chip RAM are transferred to external memory by DMA transfer. Completion of DMA transfer is detected by checking the transfer-end flag (TE bit).

The register settings of the sample program are shown in table 2, macro definitions used in the sample program are 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_0 (CHCR_0)	H'FFFE100C	H'00000000	DE = 0: DMA transfer is disabled
		H'80005410	TC = 1: Transfer by the number of times set in DMATCR0 in response to a DMA request. RLD = 0: Reload function is disabled. DM = B'01: Destination address is incremented. SM = B'01: Source address is incremented. RS = B'0100: Auto request TB = 0: Cycle-steal mode TS = B'10: Longword transfer IE = 0: Interrupt requests are disabled.
		H'80005411	DE = 1: DMA transfer is enabled.
DMA source address register_0 (SAR_0)	H'FFFE1000	H'FFF90000	Transfer source start address: Address in on-chip RAM area is specified.
DMA destination address register_0 (DAR_0)	H'FFFE1004	H'0C000000	Transfer destination start address: Address in external memory area* is specified.
DMA transfer count register_0 (DMATCR_0)	H'FFFE1008	H'64	Number of transfers: 100 (H'64)
DMA operation register (DMAOR)	H'FFFE1200	H'0001	DME = 1: DMA transfer is enabled on all channels.

Note: * The address of the external memory area depends on the target board to be used.

Table 3 Macro Definitions in the Sample Program

Macro Definition	Setting	Description
SDRAM_DST_ADR	H'0C00 0000	SDRAM start address
SRAM_SRC_ADR	H'FFF9 0000	On-chip RAM start address
SIZE	H'64	Transfer count
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

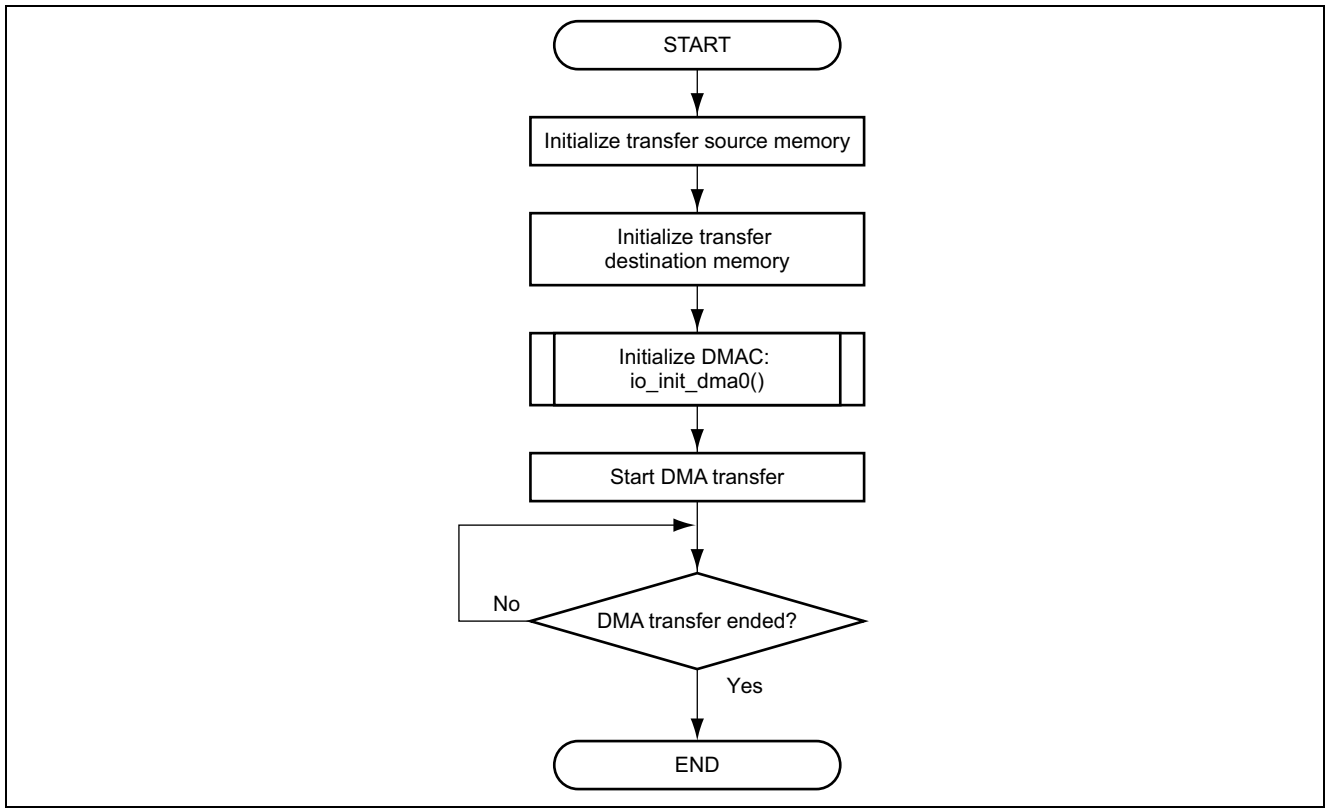

Figure 4 Processing Flow of the Sample Program

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 by DMAC0
*              : Triggered by software, the DMAC transfers 100 bytes of data from on-chip
*              : SRAM to external SDRAM.
*
*                <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 "iodefine.h"                       /* iodefine.h is automatically created by HEW */



/* ==== Macro declarations ==== */
#define SDRAM_DST_ADR ((void *)0x2c000000)   /* External SDRAM start address               */
#define SRAM_SRC_ADR  ((void *)0xfff90000)   /* On-chip SRAM start address                 */
#define SIZE          100                    /* 100 bytes of data are transferred          */


#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_dma0 (void *src, void *dst, unsigned int size, unsigned int mode);
void io_dma0_trans(void);
void io_dma0_stop (void);

```

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

```
/*""FUNC COMMENT""*************************************************************************
* ID            :
* Module summary: Main function of the sample program
*-------------------------------------------------------------------------------------------
* Include       :
*-------------------------------------------------------------------------------------------
* Declaration   : void main(void)
*-------------------------------------------------------------------------------------------
* Functional description:
*       Sample program for transferring 100-byte data from on-chip SRAM to external SDRAM.
*       Completion of DMA transfer is detected through the DMA transfer-end flag.
*       When DMA transfer ends, the processing enters infinite loop.
*-------------------------------------------------------------------------------------------
* Argument      : None
*-------------------------------------------------------------------------------------------
* Return value  : None
*-------------------------------------------------------------------------------------------
* Notes         : In the sample program, absolute addresses are used to clarify
*               : the start addresses of the data transfer source and destination.
*               : When allocating memory areas by absolute addresses, be careful so that
*               : they do not overlap with the sections used by user programs.
*               : In DMA transfer with the operand cache enabled,
*               : coherency must be kept by disabling or writing back the cache.
*               : In the sample program, coherency is kept because non-cacheable space is
*               : accessed from the CPU.
*""FUNC COMMENT END""*********************************************************************/
void main(void)
{
    int i;
    unsigned char *ptr;

    /* ==== Transfer source memory initialization ==== */
    ptr = SRAM_SRC_ADR;
    for(i=0; i < SIZE; i++){
        *ptr++ = 0x55;                      /* Fill the transfer source memory with 0x55 */
    }

    /* ==== Transfer destination memory initialization  ==== */
    ptr = SDRAM_DST_ADR;
    for(i=0; i < SIZE; i++){
        *ptr++ = 0;                         /* Clear transfer destination memory to all 0 */
    }

    /* ==== DMAC initialization ==== */
    io_init_dma0(SRAM_SRC_ADR, SDRAM_DST_ADR, SIZE , DMA_SIZE_LONG | DMA_INT_DISABLE);

    /* ---- Start DMA transfer ---- */
    io_dma0_trans();

    /* ---- Stop DMA transfer ---- */
    io_dma0_stop();

    while(1){
        /* End of program */
    }
}
```

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

```
/*""FUNC COMMENT""*****************************************************************************
* ID            :
* Module summary: Initial settings for memory-to-memory transfer by the DMAC
*-----------------------------------------------------------------------------------------------
* Include       : #include "iodefine.h"
*-----------------------------------------------------------------------------------------------
* Declaration   : io_init_dma0(void *src, void *dst, size_t size, unsigned int mode)
*-----------------------------------------------------------------------------------------------
* Functional description:
*      Transfers 'size'-byte data from source address 'src' to destination address 'dst'
*      by the DMAC. The transfer is executed by auto request.
*      Transfer size and use of interrupts 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 value    : None
*-----------------------------------------------------------------------------------------------
*  Note   : Correct operation cannot be guaranteed if the transfer data size does not
*         : agree with alignment of source and destination addresses. When interrupts
*         : are used, the corresponding interrupt routine must be prepared.
*""FUNC COMMENT END""*************************************************************************/
void io_init_dma0(void *src, void *dst,  unsigned int size, unsigned int mode)
{
    unsigned int ts;
    unsigned long ie;

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

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

    /* ---- Set DMA channel control register ---- */
    DMAC.CHCR0.BIT.DE = 0ul;              /* Disable DMA transfer                       */

    /* ---- Set DMA source address register ---- */
    DMAC.SAR0.LONG = (unsigned long)src;

    /* ---- Set DMA reload source address register ---- */
    DMAC.RSAR0.LONG = (unsigned long)src;

    /* ---- Set DMA destination address register ---- */
    DMAC.DAR0.LONG = (unsigned long)dst;

    /* ---- Set DMA reload destination address register ---- */
    DMAC.RDAR0.LONG = (unsigned long)dst;

```

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

```
164       /* ---- Set DMA transfer count register ---- */
165       /* ---- Set DMA reload transfer count register ---- */
166
167       if(ts != DMA_SIZE_LONGx4){
168           DMAC.DMATCR0.LONG = size >> ts;      /* Set transfer count                        */
169           DMAC.RDMATCR0.LONG = size >> ts;
170       }
171       else{
172           DMAC.DMATCR0.LONG = size >> 4u;      /* Set transfer count (1/16)                 */
173           DMAC.RDMATCR0.LONG = size >> 4u;
174       }
175
176       /* ---- Set DMA channel control register ---- */
177       DMAC.CHCR0.LONG = 0x80005400ul |  (mode << 3u) | (ie << 2u) ;
178           /*
179               bit31   : TC DMATCR transfer: 1--------- DMA transfer count specified in DMATCR
180               bit30-29: reserve 0
181               bit28   : RLD OFF : 0------------------- Disable reload function
182               bit27-24: reserve 0
183               bit23   : DO over run0 : 0-------------- Unused
184               bit22   : TL TEND active low : 0-------- Unused
185               bit21-20: reserve 0
186               bit19   : HE :0------------------------- Unused
187               bit18   : HIE :0------------------------ Unused
188               bit17   : AM :0------------------------- Unused
189               bit16   : AL :0------------------------- Unused
190               bit15-14: DM1:0 DM0:1------------------- Increment destination address
191               bit13-12: SM1:0 SM0:1------------------- Increment source address
192               bit11-8 : RS : auto request : B'0100---- Auto request
193               bit7    : DL : DREQ level : 0 ---------- Unused
194               bit6    : DS : DREQ select: 0 Low level- Unused
195               bit5    : TB :cycle :0------------------ Cycle-steal mode
196               bit4-3  : TS : transfer size :B'10------ Longword transfer
197               bit2    : IE : interrupt enable: 0------ Disable interrupt
198               bit1    : TE : transfer end: 0
199               bit0    : DE : DMA enable bit: 0-------- Disable DMA transfer
200           */
201
202       /* ---- Set DMA operation register ---- */
203       DMAC.DMAOR.WORD &= 0xfff9u;              /* Clear AE, NMIF bits                       */
204
205       if(DMAC.DMAOR.BIT.DME == 0){             /* Enable DMA transfer on all channels       */
206           DMAC.DMAOR.BIT.DME = 1;
207       }
208
209   }
210
```

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

```
211  /*""FUNC COMMENT""*****************************************************************************
212  * ID            :
213  * Module summary: Activating DMAC
214  *-----------------------------------------------------------------------------------------------
215  * Include       : #include "iodefine.h"
216  *-----------------------------------------------------------------------------------------------
217  * Declaration   : void io_dma0_trans(void)
218  *-----------------------------------------------------------------------------------------------
219  * Functional description: Executes DMA transfer and detects the end of transfer.
220  *-----------------------------------------------------------------------------------------------
221  * Argument      : None
222  *-----------------------------------------------------------------------------------------------
223  * Return value  : None
224  *-----------------------------------------------------------------------------------------------
225  * Notes         :
226  *""FUNC COMMENT END""*************************************************************************/
227  void io_dma0_trans(void)
228  {
229      /* ---- Execute DMA transfer ---- */
230      DMAC.CHCR0.BIT.DE = 1ul;                /* Enable DMA transfer                    */
231
232      /* Detect the end of transfer */;
233      while(DMAC.CHCR0.BIT.TE == 0ul){        /* Wait until the TE bit is set           */
234      }
235  }
236
237  /*""FUNC COMMENT""*****************************************************************************
238  * ID            :
239  * Module summary: Stopping DMAC
240  *-----------------------------------------------------------------------------------------------
241  * Include       : #include "iodefine.h"
242  *-----------------------------------------------------------------------------------------------
243  * Declaration   : void io_dma0_stop(void)
244  *-----------------------------------------------------------------------------------------------
245  * Functional description: Stops DMA transfer.
246  *-----------------------------------------------------------------------------------------------
247  * Argument     : None
248  *-----------------------------------------------------------------------------------------------
249  * Return value : None
250  *-----------------------------------------------------------------------------------------------
251  * Notes        :
252  *""FUNC COMMENT END""*************************************************************************/
253  void io_dma0_stop(void)
254  {
255      /* ---- Stop DMA transfer ---- */
256      DMAC.CHCR0.BIT.DE = 0ul;                /* Disable transfer by DMA0               */
257  }
258
259  /* End of File */
260
```

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.