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.

SH7780 Group

Example of Memory-to-Memory Transfer by the DMAC (LRAM to DDR SDRAM)

Introduction

This application note gives an example of how to set products of the SH7780 Group for inter-memory (LRAM to DDR SDRAM) data transfer and describes a sample application for this purpose.

Target Device

SH7780 (MS7780SE03 Solution Engine by Hitachi ULSI Systems)

Contents

1. Preface

1.1 Specifications

DMA channel 0 is used to transfer all data on pages 0 and 1 (16 KB) of the L memory to DDR-SDRAM. Checking to see if the transfer is complete proceeds every 5 ms; if it is complete, DMA transfer is started again.

Cycle-stealing mode is selected as the bus mode and DMA transfers are requested in auto request mode (transfer requests by software).

1.2 Module Used

- DMAC channel 0

1.3 Applicable Conditions

• MCU	SH7780	
• Operating frequency	Internal clock	: 400 MHz
	SuperHyway clock	: 200 MHz
	Peripheral clock	: 33 MHz
	DDR clock	: 160 MHz
	External clock	: 33 MHz
	PCI bus clock	: 33 MHz
• Clock operating mode	Mode 3 (MODE7 = low, MODE2 = low, MODE1 = high, MODE0 = high)	
• Data alignment	Little endian	
• Addressing mode	29-bit	
• C compiler	SuperHRISC Engine Family C/C++ Compiler Package Ver.9.1.0 (manufactured by Renesas Technology)	

1.4 Related Application Notes

The operation of the reference program for this document was confirmed with the setting conditions described in the following application notes.

- SH7780 Initialization Example: REJ06B0712-0100
- Example of TMU Operation (Generating 1 Second by Fixed Interval Timer): REJ06B0718-0100
  Note that the timer unit (TMC channel 0) is used as a fixed-interval timer to measure 5-ms periods in this application note.

Please refer to these application notes in combination with this one.

2. Description of the Sample Application

This sample program uses channel 0 of the direct memory access controller (DMAC) to perform block transfer from the on-chip RAM (LRAM) to external memory (DDR-SDRAM). After the first transfer starts, the completion of transfer is checked every 5 ms; if it is, the program restarts the transfer of next block.

When one-block transfer is complete, the program disables transfer and waits for 5 ms. It waits after the completion of transfer in the same way when an address error has occurred during transfer.

2.1 Description of the Sample Program

This sample program consists of the following four source files.

(1) dmac.c
(2) main.c
(3) intprg.c
(4) vecttbl.src

Since the code for *Example of TMU Operation (Generating 1 Second by Fixed Interval Timer)* is used to count 5-ms intervals, tmu.c from the corresponding sample application should be added to this project.

(1) dmac.c describes the functions used in this program to set up DMAC operation and to restart DMAC transfer when an address error is encountered.
This program code is not included in the application for *SH7780 Initialization Example*, which is used as its basis.

(2) main.c sets the status register (SR) and calls the function to set up DMAC operation.
Change main.c that was included with *SH7780 Initialization Example* as required to match main.c for this sample program.

(3) intprg.c describes the interrupt program called from the exception/interrupt handler.
Change intprg.c that was included with *SH7780 Initialization Example* as required to match intprg.c for this sample program. Since counting of 5-ms intervals is handled in the same way as in *Example of TMU Operation (Generating 1 Second by Fixed Interval Timer)*, intprg.c should also be modified to match that sample program.

(4) vecttbl.src describes the exception/interrupt vector table (including vector-table entries), and interrupt mask table. To ensure that interrupts that have been accepted are not accepted again while they are being processed, the interrupt mask levels to be set in the IMASK bits of the status register should be described in the interrupt mask table.
Change vecttbl.src that was included with *SH7780 Initialization Example* as required to match vecttbl.src for this sample program. As with intprg.c, to use 5-ms counting as in *Example of TMU Operation (Generating 1 Second by Fixed Interval Timer)*, make modifications to include interrupt mask levels with reference to *Example of TMU Operation (Generating 1 Second by Fixed Interval Timer)*.

2.2 Operational Overview of Module Used

When the DMAC has a DMA transfer request, it initiates transfer in accord with the selected order of priority for its channels, and ends the transfer when the transfer-end condition is satisfied. There are three activation methods for DMA transfer: auto request, external request and peripheral module request. The bus mode can be selected as either burst mode or cycle-stealing mode.

Table 1 gives an overview of the DMAC.

Table 1 DMAC Overview

Item	Overview
No. of channels available	12 (channels 0 to 3 can accept external requests)
Address space	4 Gbytes
Data transfer unit	Byte, word (2 bytes), longword (4 bytes), 16 bytes, and 32 bytes
Maximum transfer count	16777216
Address mode	Dual address mode
Transfer requests	Auto request, external request, and peripheral module request
Bus modes	Cycle-stealing mode and burst mode
Channel priority	Fixed mode and round-robin mode
Interrupt request	An interrupt request is issued to the CPU after a transfer is half completed or completed, and when an address error occurs.
External request detection	The following four senses for DREQ input detection are available: low or high level, or rising or falling edge.
Transfer request acknowledge signal/transfer end signal	Active levels for DACK and DRAK can be set independently.

2.3 Procedure for Setting Module Used

This section describes the initialization procedure for inter-memory transfer by the DMAC. Transfer requests are in auto request mode.

In this program, the following initial settings are made at the beginning of the main function on the assumption that this is based on the program for *SH7780 Initialization Example*. Since operation in privileged mode is a precondition for the program doing this, take care with regard to the processing mode when you adapt this code for use with other programs etc.

Figure 1 shows an example of the initialization sequence for data transfer between the LRAM and DDR SDRAM memory areas. For details on the settings of individual registers, see the *SH7780 Group Hardware Manual*.

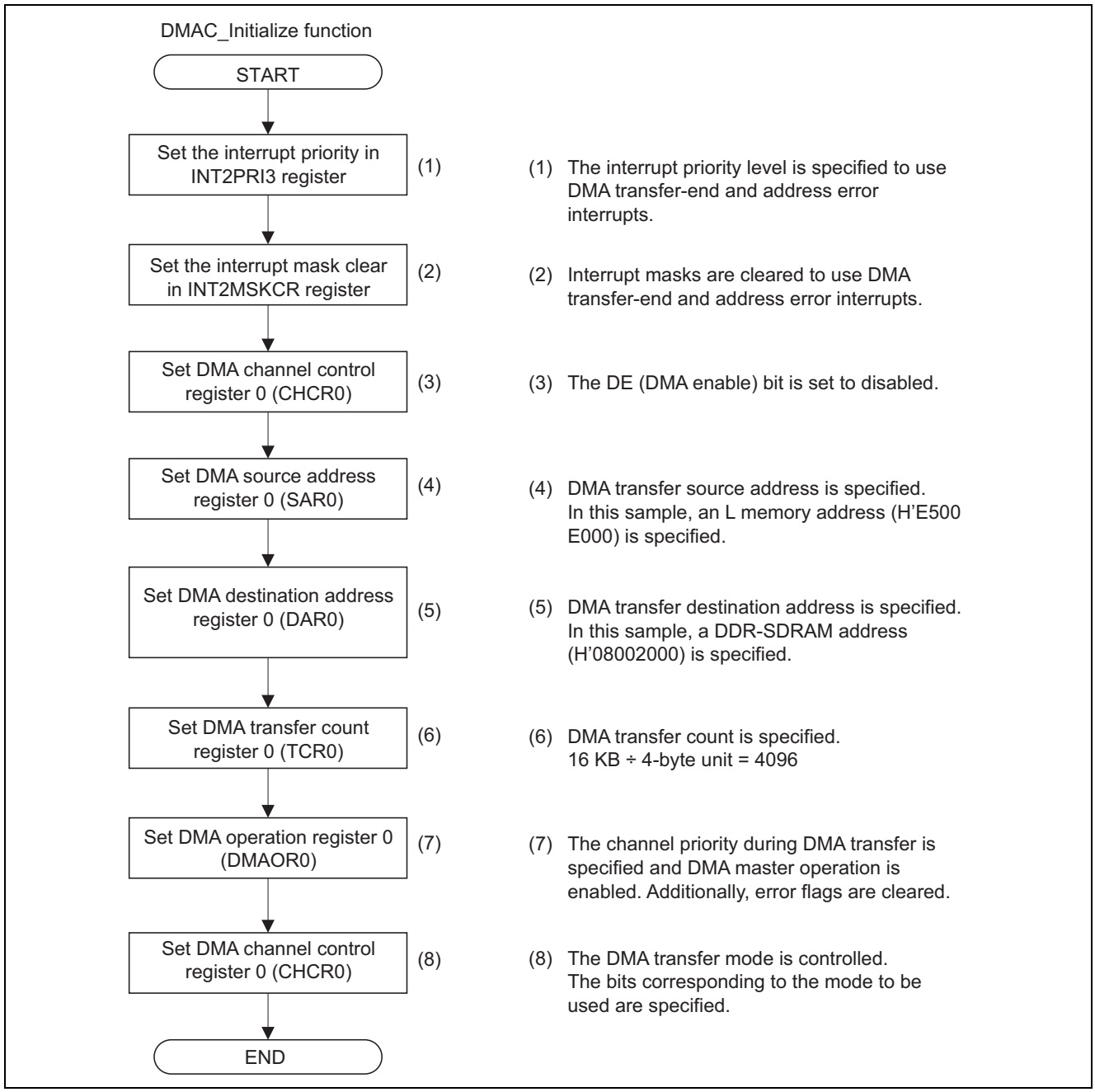


Figure 1 Flow of Setting the DMAC for Inter-Memory Transfer

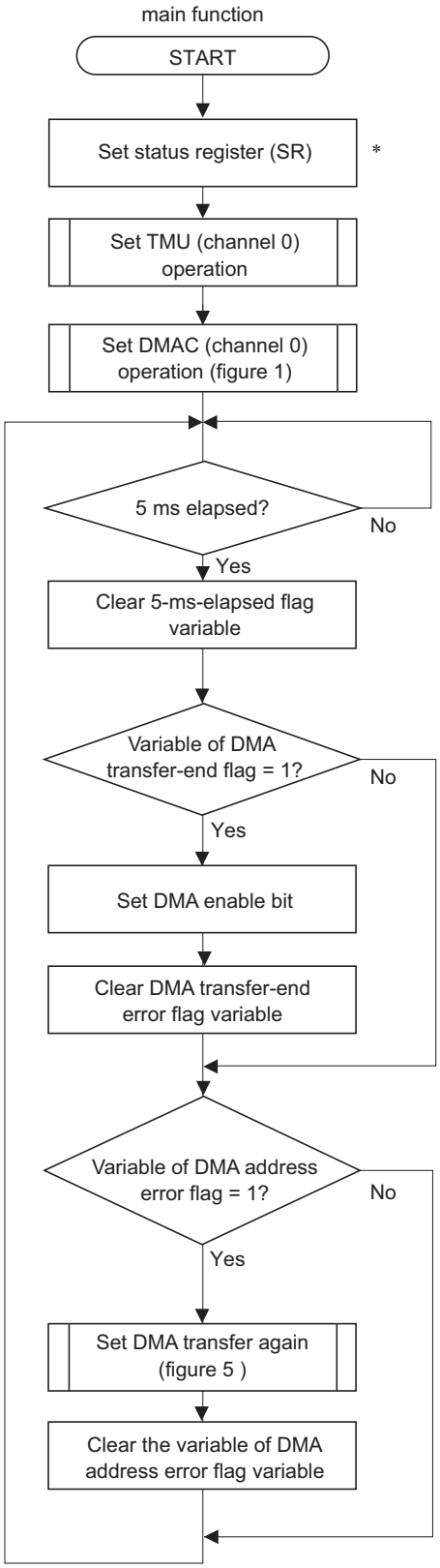
2.4 Procedure for processing the Sample Program

Table 2 gives examples for setting DMAC-related registers.

The following pages provide sample flowcharts of the main function, transfer-end interrupt handling, address-error interrupt handling, and restarting transfers.

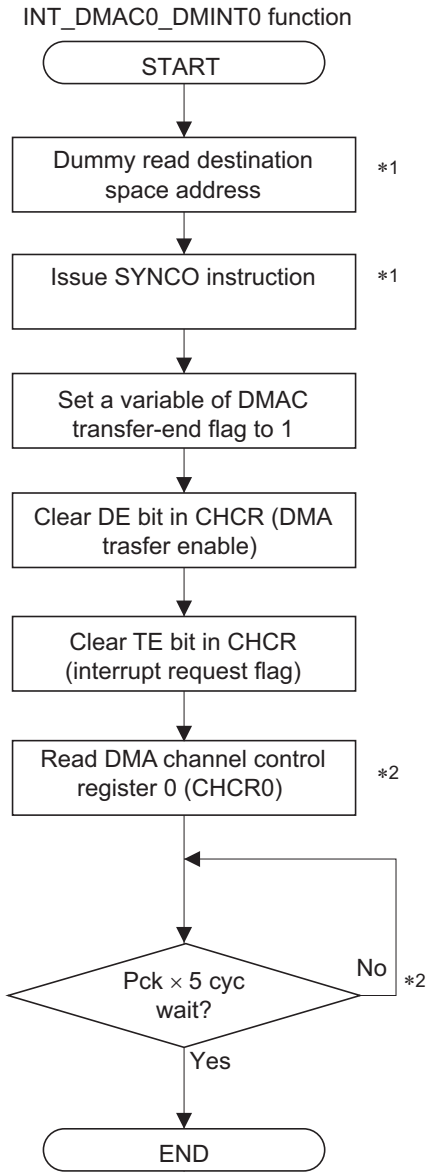
Table 2 DMAC Setting

Register Name	Address	Setting	Description
Interrupt priority register 3 (INT2PRI3)	H'FFD4 000C	H'001F 0000	DMAC (0) interrupt priority level: 31
Interrupt mask clear register (INT2MSKCR)	H'FFD4 003C	H'0000 0100	DMAC (0) interrupt mask clear
DMA source address register 0 (SAR0)	H'FC80 8020	H'E500E000	(Transfer source address)
DMA destination address register 0 (DAR0)	H'FC80 8024	H'08002000	(Transfer destination address)
DMA transfer count register 0 (TCR0)	H'FC80 8028	H' 1000	16 Kbytes ÷ 4 bytes = 4096
DMA channel control register 0 (CHCR0)	H'FC80 802C	H' 42005414	Repeat mode (SAR/DAR/TCR) Transfer in 4-byte (longword) units Transfer destination address is incremented. Transfer source address is incremented. Auto request mode Cycle-steal mode DMA ch0 interrupt requests are enabled.
DMA operation register 0 (DMAOR0)	H'FC80 8060	H'3001	Intermittent mode 64 DMA master enable



Note: * Since this program directly changes registers for peripheral functions, the MD bit (processing mode bit) is set to 1 (privileged mode).
If this bit is changed to "0" (user mode) and the function for setting operations is called, access to the registers for peripheral functions leads to an address error.

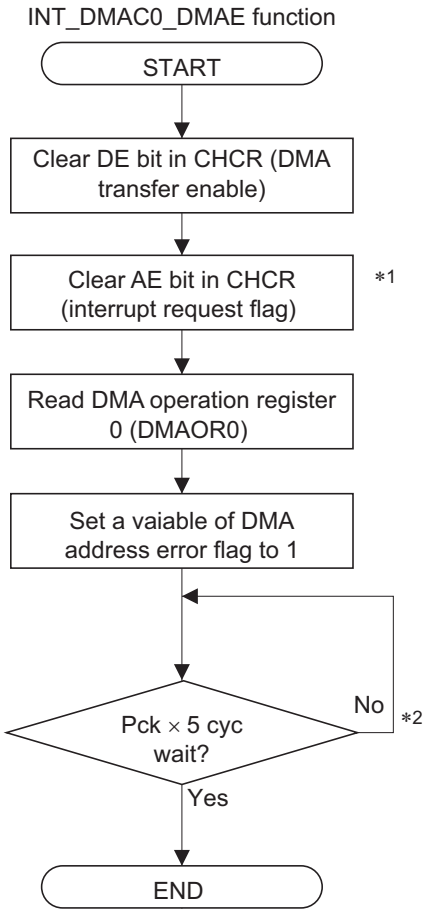
Figure 2 Flow of Processing by the Main Function of the Sample Program (Memory-to-Memory Transfer by the DMAC)



Notes: 1. When a DMA transfer-end interrupt is used, completion of the last transfer must be confirmed by a dummy-read from the destination space followed by issuing of a SYNCO instruction.

2. To prevent the erroneous acceptance of interrupts from sources that should have been updated, wait for the priority determination time (Pck × 5 cycles) after reading the on-chip module register that contains the given flag and before setting the BL bit to 0.

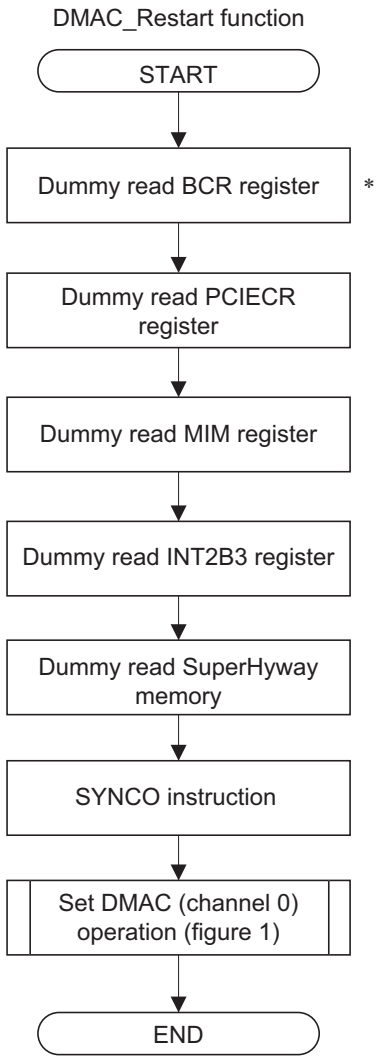
Figure 3 Flow of DMINT0 (DMA Transfer-End) Interrupt Handling



Notes: 1. To prevent the erroneous acceptance of interrupts from sources that should have been updated, wait for the priority determination time (Pck × 5 cycles) after reading the on-chip module register that contains the given flag and before setting the BL bit to 0.

2. When an address error occurs, its occurrence is recorded and a transition to the wait state is made.

Figure 4 Flow of Address Error Interrupt Handling



Note: * When a DMA address error occurs, the following steps should be performed before restarting transfer.

1. Dummy read to the following modules:
   - LBSC: BCR register
   - PCIC: PCIECR register
   - DDRIF: MIM register
   - INTC: LINT2B3 register
   - SuperHyway memory
2. Issue a SYNCO instruction.
3. Reconfigure all channels corresponding to the DMAOR where an address error occurred.

Figure 5 Flow of Processing to Restart DMA Transfer

3. Listing of Sample Program

1. Sample Program Listing: "dmac.c" (1)

```
1   /*""FILE COMMENT""*********************************************************************
2   *     System Name :   SH7780 Sample Program
3   *     File Name   :   dmac.c
4   *     Version     :   1.00.00
5   *     Contents    :   SH7780 DMAC transmit Program
6   *     Model       :   Hitachi_ULSI_Systems SolutionEngine MS7780SE03
7   *     CPU         :   SH7780
8   *     Compiler    :   SHC.9.1.00
9   *     OS          :   none
10  *
11  *     note        :   < Caution >
12  *                     This sample program is provided simply as a reference and
13  *                     its operation is not guaranteed.
14  *                     Use this sample program as a technical reference when
15  *                     developing software.
16  *
17  * Copyright (C) 2007 Renesas Technology Corp. All Rights Reserved
18  *
19  *     History     :   2007/12/26 ver 1.00.00
20  *
21  ***************************************************************************************/
22  #include <machine.h>
23
24  /* --- Function Definition(internal) --- */
25  void DMAC_Initialize(void);
26  void DMAC_Restart(void);
27
28  /* --- Symbol Definition --- */
29  struct st_dmac0{                                    /* struct DMAC0 */
30              void           *SAR;                    /*      SAR     */
31              void           *DAR;                    /*      DAR     */
32              unsigned int   TCR;                     /*      TCR     */
33              union{                                  /*      CHCR    */
34                  unsigned int LONG;                  /* Long Word Access */
35                  struct{                             /* Bit Access  */
36                          unsigned int   :1;          /*              */
37                          unsigned int LCKN :1;       /*      LCKN    */
38                          unsigned int   :2;          /*              */
39                          unsigned int RPT :3;        /*      RPT     */
40                          unsigned int   :1;          /*              */
41                          unsigned int DO :1;         /*      DO      */
42                          unsigned int RL :1;         /*      RL      */
43                          unsigned int   :1;          /*              */
44                          unsigned int TS2 :1;        /*      TS2     */
45                          unsigned int HE :1;         /*      HE      */
46                          unsigned int HIE :1;        /*      HIE     */
47                          unsigned int AM :1;         /*      AM      */
48                          unsigned int AL :1;         /*      AL      */
49                          unsigned int DM :2;         /*      DM      */
50                          unsigned int SM :2;         /*      SM      */
51                          unsigned int RS :4;         /*      RS      */
52                          unsigned int DL :2;         /*      DL      */
53                          unsigned int TB :1;         /*      TB      */
54                          unsigned int TS :2;         /*      TS      */
```

2. Sample Program Listing: "dmac.c" (2)

```
55                          unsigned int IE:1;          /*      IE      */
56                          unsigned int TE:1;          /*      TE      */
57                          unsigned int DE:1;          /*      DE      */
58                      }   BIT;
59                  }   CHCR;
60  };
61
62  union st_dmac{                                      /* struct DMAOR */
63                  unsigned short WORD;                /* Word Access */
64                  struct{                             /* Bit Access  */
65                      unsigned short     :2;          /*              */
66                      unsigned short CMS :2;          /*      CMS     */
67                      unsigned short     :2;          /*              */
68                      unsigned short PR  :2;          /*      PR      */
69                      unsigned short     :5;          /*              */
70                      unsigned short AE  :1;          /*      AE      */
71                      unsigned short NMIF:1;          /*      NMIF    */
72                      unsigned short DME :1;          /*      BIT     */
73                  }   BIT;
74  };
75
76  #define DMAC0         (*(volatile struct st_dmac0   *)0xFC808020)  /* DMAC ch0 register top
                          Address */
77  #define DMAOR0        (*(volatile union st_dmac  *)0xFC808060)  /* DMAOR0 Address */
78
79  #define BCR           (*(volatile unsigned int   *)0xFF801000)
80  #define PCIECR        (*(volatile unsigned int   *)0xFE000008)
81  #define MIM_L         (*(volatile unsigned int   *)0xFE80000C)
82  #define INT2B3        (*(volatile unsigned int   *)0xFFD4004C)
83  #define SHMEM0_TOP_ADD   ((void *)0xFE410000)
84
85  #define INT2PRI3 (*(volatile unsigned int   *)0xFFD4000C)
86  #define INT2MSKCR     (*(volatile unsigned int   *)0xFFD4003C)  /* INT2MSKCR Address */
87
88  #define  LRAM_SRC_ADR       ((void  *)0xE500E000)  /* DMA source Address */
89  #define  DDR_DST_ADR        ((void  *)0x08002000)  /* DMA destination Address */
90
91  /***** Function Comment ***************************************
92  * Outline     : DMAC_Initialize
93  *-------------------------------------------------------------
94  * Declaration     : void DMAC_Initialize(void)
95  *-------------------------------------------------------------
96  * Functional description:
97  *               Initialize DMAC controller and start transmit
98  *-------------------------------------------------------------
99  * Return Value    : -
100 * Argument        : -
101 *-------------------------------------------------------------
102 * Input           : -
103 * Output      : -
104 *-------------------------------------------------------------
105 * Notes           : -
106 ****** Function Comment End *********************************/
107 void DMAC_Initialize(void)
108 {
109   INT2PRI3 |= 0x001f0000;              /* DMAC ch0 interrupt level 31 */
```

3. Sample Program Listing: "dmac.c" (3)

```
110   INT2MSKCR = 0x00000100;           /* DMAC ch0 interrupt mask clear */
111
112   DMAC0.CHCR.BIT.DE = 0;            /* DMAC ch0 transmit disable */
113   DMAC0.SAR = LRAM_SRC_ADR;         /* DMAC ch0 transmit source Address */
114   DMAC0.DAR = DDR_DST_ADR;          /* DMAC ch0 transmit destination Address */
115   DMAC0.TCR = 0x00001000;           /* DMAC ch0 transmit count */
116   DMAC0.CHCR.LONG = 0x42005414;     /* DMAC ch0 CHCR set */
117   DMAOR0.BIT.CMS = 0x03;            /* Intermittent mode 64 */
118   DMAOR0.BIT.AE &= 0;               /* clear Address Error flag */
119   DMAOR0.BIT.NMIF &= 0;             /* clear NMI interrupt flag */
120   DMAOR0.BIT.DME = 1;               /* enable DMAC ch0 DMA master */
121   DMAC0.CHCR.BIT.DE = 1;            /* enable DMAC ch0 transmite */
122 }
123
124 /***** Function Comment ***************************************
125 * Outline    : DMAC_ReStart
126 *-------------------------------------------------------------
127 * Declaration    : void DMAC_Restart(void)
128 *-------------------------------------------------------------
129 * Functional description:
130 *              restart DMA Controller transmit
131 *-------------------------------------------------------------
132 * Return Value   : -
133 * Argument       : -
134 *-------------------------------------------------------------
135 * Input          : -
136 * Output         : -
137 *-------------------------------------------------------------
138 * Notes          : -
139 ****** Function Comment End *********************************/
140 void DMAC_Restart(void)
141 {
142   volatile unsigned int dummy;
143
144   dummy = BCR;                      /* dummy read BCR */
145   dummy = PCIECR;                   /* dummy read PCIECR */
146   dummy = MIM_L;                    /* dummy read MIM */
147   dummy = INT2B3;                   /* dummy read INT2B3 */
148   dummy = (*(unsigned int *)SHMEM0_TOP_ADD);    /* dummy read SuperHyway RAM */
149
150   synco();                          /* SYNCO */
151
152   DMAC_Initialize();
153 }
```

4. Sample Program Listing: "intprg.c" (1)

```
/*""FILE COMMENT""*********************************************************************
*     System Name :   SH7780 Sample Program
*     File Name   :   intprg.c
*     Version     :   1.00.00
*     Contents    :   SH7780 Initialize Program
*     Model       :   Hitachi_ULSI_Systems SolutionEngine MS7780SE03
*     CPU         :   SH7780
*     Compiler    :   SHC.9.1.00
*     OS          :   none
*
*     note        :   < Caution >
*                     This sample program is provided simply as a reference and
*                     its operation is not guaranteed.
*                     Use this sample program as a technical reference when
*                     developing software.
*
* Copyright (C) 2007 Renesas Technology Corp. All Rights Reserved
*
*     History     :   2007/12/26 ver 1.00.00
*
*********************************************************************************************/
#include <machine.h>

/* --- Function Definition(internal) --- */
static void int_responstime_wait(unsigned int wait_time);

/* --- Symbol Definition --- */
struct st_tmu{                                          /* struct TMU0  */
               unsigned int     TCOR;                   /*   TCOR       */
               unsigned int     TCNT;                   /*    TCNT      */
               union {                                 /*    TCR       */
                     unsigned short WORD;              /*  Word Access */
                     struct {                          /*  Bit  Access */
                            unsigned short     :7;      /*              */
                            unsigned short UNF :1;      /*    UNF       */
                            unsigned short     :2;      /*              */
                            unsigned short UNIE:1;      /*    UNIE      */
                            unsigned short CKEG:2;      /*    CKEG      */
                            unsigned short TPSC:3;      /*    TPSC      */
                            }       BIT;                /*              */
                     }          TCR;                    /*              */
};

union st_int2b3{
  unsigned int       LONG;          /* Long Word Access    */
  struct{                           /* Bit Access  */
      unsigned int           :18;   /*             */
      unsigned int   DMAE1   :1;    /*   DMAE1     */
      unsigned int   DMAE0   :1;    /*   DMAE0     */
      unsigned int   DMINT11 :1;    /*   DMINT11   */
      unsigned int   DMINT10 :1;    /*   DMINT10   */
      unsigned int   DMINT9  :1;    /*   DMINT9    */
      unsigned int   DMINT8  :1;    /*   DMINT8    */
      unsigned int   DMINT7  :1;    /*   DMINT7    */
      unsigned int   DMINT6  :1;    /*   DMINT6    */
      unsigned int   DMINT5  :1;    /*   DMINT5    */
```

5. Sample Program Listing: "intprg.c" (2)

```
57        unsigned int    DMINT4  :1;     /*   DMINT4     */
58        unsigned int    DMINT3  :1;     /*   DMINT3     */
59        unsigned int    DMINT2  :1;     /*   DMINT2     */
60        unsigned int    DMINT1  :1;     /*   DMINT1     */
61        unsigned int    DMINT0  :1;     /*   DMINT0     */
62     }BIT;
63  };
64
65  struct st_dmac0{                                    /* struct DMAC0 */
66                void            *SAR;                 /*      SAR     */
67                void            *DAR;                 /*      DAR     */
68                unsigned int    TCR;                  /*      TCR     */
69                union{                                /*      CHCR    */
70                    unsigned int LONG;                /* Long Word Access */
71                    struct{                           /* Bit Access  */
72                            unsigned int    :1;       /*              */
73                            unsigned int LCKN :1;     /*      LCKN    */
74                            unsigned int    :2;       /*              */
75                            unsigned int RPT :3;      /*      RPT     */
76                            unsigned int    :1;       /*              */
77                            unsigned int DO :1;       /*      DO      */
78                            unsigned int RL :1;       /*      RL      */
79                            unsigned int    :1;       /*              */
80                            unsigned int TS2 :1;      /*      TS2     */
81                            unsigned int HE :1;       /*      HE      */
82                            unsigned int HIE :1;      /*      HIE     */
83                            unsigned int AM :1;       /*      AM      */
84                            unsigned int AL :1;       /*      AL      */
85                            unsigned int DM :2;       /*      DM      */
86                            unsigned int SM :2;       /*      SM      */
87                            unsigned int RS :4;       /*      RS      */
88                            unsigned int DL :2;       /*      DL      */
89                            unsigned int TB :1;       /*      TB      */
90                            unsigned int TS :2;       /*      TS      */
91                            unsigned int IE:1;        /*      IE      */
92                            unsigned int TE:1;        /*      TE      */
93                            unsigned int DE:1;        /*      DE      */
94                    }   BIT;
95                }   CHCR;
96  };
97
98  union st_dmac{                                      /* struct DMAOR */
99                unsigned short WORD;                  /* Word Access */
100               struct{                               /* Bit Access  */
101                   unsigned short     :2;           /*              */
102                   unsigned short CMS :2;           /*      CMS     */
103                   unsigned short     :2;           /*              */
104                   unsigned short PR  :2;           /*      PR      */
105                   unsigned short     :5;           /*              */
106                   unsigned short AE  :1;           /*      AE      */
107                   unsigned short NMIF:1;           /*      NMIF    */
108                   unsigned short DME :1;           /*      BIT     */
109               }   BIT;
110 };
111
```

6. Sample Program Listing: "intprg.c" (3)

```
112 #define DMAC0        (*(volatile struct st_dmac0    *)0xFC808020)  /* DMAC ch0 register top
                    Address */
113 #define DMAOR0       (*(volatile union st_dmac  *)0xFC808060)  /* DMAOR0 Address */
114
115 #define INT2B3   (*(volatile union st_int2b3 *)0xFFD4004C) /* INT2B3 Address */
116
117 #define TMU0 (*(volatile struct st_tmu  *)0xFFD80008)  /* TMU0  Address */
118 #define  DDR_TOP_ADR     (*(volatile unsigned int *)0x08000000)
119
120 #define INTC_RESPONSEWAIT    (0x00000014)       /* INT response wait Pck 5cycle
121                                                 H'14 = (1/Pck*5cyc) / (1/Ick*3cyc) */
122
123 /* --- RAM allocation variable declaration --- */
124 extern unsigned char u1Flg5ms;
125 extern unsigned char u1FlgEndDmac0;
126 extern unsigned char u1FlgAddErrDMAC0;
127 #pragma section IntPRG

293 /* H'580 TMU ch-0 underflow interrupt */
294 void INT_TMU0_TUNI0(void)
295 {
296   volatile unsigned short dummy;
297
298   TMU0.TCR.BIT.UNF = 0;      /* TMU ch0 UNF flag clear */
299   u1Flg5ms = 1;              /* set 5ms flag */
300   dummy = TMU0.TCR.WORD; /* dummy read */
301
302   int_responstime_wait(INTC_RESPONSEWAIT);  /* 5cyc(Pck=33MHz) wait */
303 }

321 /* H'640 ch-0 DMA transmit end or halfend interrupt */
322 void INT_DMAC0_DMINT0(void)
323 {                                /* (start of additional part from Initialize program) */
324   volatile unsigned int dummy;
325
326   dummy = DDR_TOP_ADR;              /* destinate address dummy read */
327   synco();                          /* SYNCO */
328
329   u1FlgEndDmac0 = 1;                /* set"DMAC ch0 transmit end flag" */
330
331   DMAC0.CHCR.BIT.DE = 0;            /* DMAC ch0 transmite disable */
332   DMAC0.CHCR.BIT.TE &= 0;           /* DMAC ch0 TEbit clear */
333
334   dummy = DMAC0.CHCR.LONG;          /* dummy read */
335
336   int_responstime_wait(INTC_RESPONSEWAIT);  /* 5cyc(Pck=33MHz) wait */
337 }                                /* (end of additional part from Initialize program) */
338 /* H'660 ch-1 DMA transmit end or halfend interrupt */
339 void INT_DMAC0_DMINT1(void)
340 {
341 }
342 /* H'680 ch-2 DMA transmit end or halfend interrupt */
343 void INT_DMAC0_DMINT2(void)
344 {
345 }
346
```

7. Sample Program Listing: "intprg.c" (4)

```
347 /* H'6A0 ch-3 DMA transmit end or halfend interrupt */
348 void INT_DMAC0_DMINT3(void)
349 {
350 }
351 /* H'6C0 ch0-5,ch6-11 DMA address error interrupt */
352 void INT_DMAC0_DMAE(void)
353 {                               /* (start of additional part from Initialize program) */
354   volatile unsigned int dummy;
355
356   DMAC0.CHCR.BIT.DE = 0;        /* DMAC ch0 transmit disable */
357   DMAOR0.BIT.AE &= 0;           /* clear address error flag */
358
359   dummy = DMAOR0.WORD;          /* dummy read */
360
360   u1FlgAddErrDMAC0 = 1;         /* set "DMAC ch0-5 address error flag" */
361
362   int_responstime_wait(INTC_RESPONSEWAIT);  /* 5cyc(Pck=33MHz) wait */
363 }                               /* (end of additional part from Initialize program) */

605 #pragma inline_asm(int_responstime_wait)
606 static void int_responstime_wait(unsigned int wait_time)
607 {
608 ?0001:
609             DT          R4
610             BF          ?0001
611             NOP
612 }
```

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

```
/*""FILE COMMENT""*********************************************************************
*     System Name :   SH7780 Sample Program
*     File Name   :   main.c
*     Version     :   1.00.00
*     Contents    :   SH7780 Initialize Program
*     Model       :   Hitachi_ULSI_Systems SolutionEngine MS7780SE03
*     CPU         :   SH7780
*     Compiler    :   SHC.9.1.00
*     OS          :   none
*
*     note        :   < Caution >
*                     This sample program is provided simply as a reference and
*                     its operation is not guaranteed.
*                     Use this sample program as a technical reference when
*                     developing software.
*
* Copyright (C) 2007 Renesas Technology Corp. All Rights Reserved
*
*     History     :   2007/12/26 ver 1.00.00
*
*****************************************************************************************/
#include <machine.h>

/* --- Function Definition(internal) --- */

/* --- Symbol Definition --- */
struct st_dmac0{                                    /* struct DMAC0 */
              void           *SAR;                  /*      SAR     */
              void           *DAR;                  /*      DAR     */
              unsigned int   TCR;                   /*      TCR     */
              union{                                /*      CHCR    */
                  unsigned int LONG;                /* Long Word Access */
                  struct{                           /* Bit Access  */
                          unsigned int   :1;        /*              */
                          unsigned int LCKN :1;     /*      LCKN    */
                          unsigned int   :2;        /*              */
                          unsigned int RPT :3;      /*      RPT     */
                          unsigned int   :1;        /*              */
                          unsigned int DO :1;       /*      DO      */
                          unsigned int RL :1;       /*      RL      */
                          unsigned int   :1;        /*              */
                          unsigned int TS2 :1;      /*      TS2     */
                          unsigned int HE :1;       /*      HE      */
                          unsigned int HIE :1;      /*      HIE     */
                          unsigned int AM :1;       /*      AM      */
                          unsigned int AL :1;       /*      AL      */
                          unsigned int DM :2;       /*      DM      */
                          unsigned int SM :2;       /*      SM      */
                          unsigned int RS :4;       /*      RS      */
                          unsigned int DL :2;       /*      DL      */
                          unsigned int TB :1;       /*      TB      */
                          unsigned int TS :2;       /*      TS      */
                          unsigned int IE:1;        /*      IE      */
                          unsigned int TE:1;        /*      TE      */
                          unsigned int DE:1;        /*      DE      */
                  }   BIT;
```

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

```
57              }   CHCR;
58  };
59
60  union st_dmac{                                  /* struct DMAOR */
61              unsigned short WORD;                /* Word Access */
62              struct{                             /* Bit Access  */
63                  unsigned short     :2;          /*             */
64                  unsigned short CMS :2;          /*      CMS    */
65                  unsigned short     :2;          /*             */
66                  unsigned short PR  :2;          /*      PR     */
67                  unsigned short     :5;          /*             */
68                  unsigned short AE  :1;          /*      AE     */
69                  unsigned short NMIF:1;          /*      NMIF   */
70                  unsigned short DME :1;          /*      BIT    */
71              }   BIT;
72  };
73
74  #define DMAC0        (*(volatile struct st_dmac0    *)0xFC808020)  /* DMAC ch0 register top
                 Address */
75
76  #define  SR_Init        0x400000e0         /* Privileged mode, IMASK level 14 */
77
78  /* --- RAM allocation variable declaration --- */
79  volatile unsigned char u1Flg5ms;
80  volatile unsigned char u1Cnt1s;
81  volatile unsigned char u1FlgEndDmac0;
82  volatile unsigned char u1FlgAddErrDMAC0;
83
84  /***** Function Comment ***************************************
85  * Outline    : main
86  *-------------------------------------------------------------
87  * Declaration    : void main(void)
88  *-------------------------------------------------------------
89  * Functional description:
90  *              main function
91  *-------------------------------------------------------------
92  * Return Value   : -
93  * Argument       : -
94  *-------------------------------------------------------------
95  * Input          : -
96  * Output     : -
97  *-------------------------------------------------------------
98  * Notes          : -
99  ****** Function Comment End *********************************/
100 void main(void)
101 {
102   set_cr(SR_Init);          /* Set SR "Privileged mode, IMASK level 14" */
103
104   TMU0_Initialize();        /* TMU0 Initialize (additional part from Initialize program)*/
105   DMAC_Initialize();        /* DMAC Initialize (additional part from Initialize program)*/
106
107   while(1)
108   {                         /* (start of additional part from Initialize program) */
109       while(u1Flg5ms == 0x00)
110       {
111           nop();
```

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

```
112       }
113       u1Flg5ms = 0;              /* clear 5ms flag */
114       u1Cnt1s++;                 /* 1s count +1 */
115
116       if(u1FlgEndDmac0 == 1)     /* if DMAC-ch0 transmit end */
117       {
118           DMAC0.CHCR.BIT.DE = 1; /* DMAC ch0 transmit enable */
119           u1FlgEndDmac0 = 0;     /* clear "DMAC ch0 transmit end flag" */
120       }
121       if(u1FlgAddErrDMAC0 == 1)
122       {
123           DMAC_Restart();        /* DMAC-ch0 transmit restart */
124           u1FlgAddErrDMAC0 = 0;
125       }
126   }                              /* (end of additional part from Initialize program) */
127 }
```

11. Sample Program Listing: "vecttbl.src"

```
1   ;*""FILE COMMENT""*********************************************************************
2   ;     System Name :   SH7780 Sample Program
3   ;     File Name   :   vecttbl.src
4   ;     Version     :   1.00.00
5   ;     Contents    :   SH7780 Initialize Program
6   ;     Model       :   Hitachi_ULSI_Systems SolutionEngine MS7780SE03
7   ;     CPU         :   SH7780
8   ;     Compiler    :   SHC.9.1.00
9   ;     OS          :   none
10  ;
11  ;     note        :   < Caution >
12  ;                     This sample program is provided simply as a reference and
13  ;                     its operation is not guaranteed.
14  ;                     Use this sample program as a technical reference when
15  ;                     developing software.
16  ;
17  ; Copyright (C) 2007 Renesas Technology Corp. All Rights Reserved
18  ;
19  ;     History     :   2007/12/26 ver 1.00.00
20  ;
21  ;*****************************************************************************************/
22
23                .include    "vect.inc"
24
25                .section    VECTTBL,data
26                .export     _RESET_VECTORS

305 ;TMU-ch0
306               ;H'580      TMU_TUNI0
307               .data.b      H'F0

320 ;DMAC(0)
321               ;H'640      DMINT0
322                .data.b     H'F0        /* (change part from Initialize program) */
323               ;H'660      DMINT1
324                .data.b     H'00
325               ;H'680      DMINT2
326                .data.b     H'00
327               ;H'6A0      DMINT3
328                .data.b     H'00
329               ;H'6C0      DMAE
330                .data.b     H'F0        /* (change part from Initialize program) */
331               ;H'6E0      reserve
332                .data.b     H'00
```

4. Documents for Reference

- Hardware Manual
  SH7780 Hardware Manual
  The most up-to-date version of this document is available on the Renesas Technology Website.
- Software Manual
  SH-4A Software Manual
  The most up-to-date version of this document is available on the Renesas Technology Website.

Website and Support

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

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

Revision Record

		Description	
Rev.	Date	Page	Summary
1.00	Mar.21.08	—	First edition issued

Notes regarding these materials

1. This document is provided for reference purposes only so that Renesas customers may select the appropriate Renesas products for their use. Renesas neither makes warranties or representations with respect to the accuracy or completeness of the information contained in this document nor grants any license to any intellectual property rights or any other rights of Renesas or any third party with respect to the information in this document.
2. Renesas shall have no liability for damages or infringement of any intellectual property or other rights arising out of the use of any information in this document, including, but not limited to, product data, diagrams, charts, programs, algorithms, and application circuit examples.
3. You should not use the products or the technology described in this document for the purpose of military applications such as the development of weapons of mass destruction or for the purpose of any other military use. When exporting the products or technology described herein, you should follow the applicable export control laws and regulations, and procedures required by such laws and regulations.
4. All information included in this document such as product data, diagrams, charts, programs, algorithms, and application circuit examples, 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 products listed in this document, please confirm the latest product information with a Renesas sales office. Also, please pay regular and careful attention to additional and different information to be disclosed by Renesas such as that disclosed through our website. (http://www.renesas.com)
5. Renesas has used reasonable care in compiling the information included in this document, but Renesas assumes no liability whatsoever for any damages incurred as a result of errors or omissions in the information included in this document.
6. When using or otherwise relying on the information in this document, you should evaluate the information in light of the total system before deciding about the applicability of such information to the intended application. Renesas makes no representations, warranties or guaranties regarding the suitability of its products for any particular application and specifically disclaims any liability arising out of the application and use of the information in this document or Renesas products.
7. With the exception of products specified by Renesas as suitable for automobile applications, Renesas products are not designed, manufactured or tested for applications or otherwise in systems the failure or malfunction of which may cause a direct threat to human life or create a risk of human injury or which require especially high quality and reliability such as safety systems, or equipment or systems for transportation and traffic, healthcare, combustion control, aerospace and aeronautics, nuclear power, or undersea communication transmission. If you are considering the use of our products for such purposes, please contact a Renesas sales office beforehand. Renesas shall have no liability for damages arising out of the uses set forth above.
8. Notwithstanding the preceding paragraph, you should not use Renesas products for the purposes listed below:
   (1) artificial life support devices or systems
   (2) surgical implantations
   (3) healthcare intervention (e.g., excision, administration of medication, etc.)
   (4) any other purposes that pose a direct threat to human life

   Renesas shall have no liability for damages arising out of the uses set forth in the above and purchasers who elect to use Renesas products in any of the foregoing applications shall indemnify and hold harmless Renesas Technology Corp., its affiliated companies and their officers, directors, and employees against any and all damages arising out of such applications.
9. You should use the products described herein within the range specified by Renesas, especially with respect to the maximum rating, operating supply voltage range, movement power voltage range, heat radiation characteristics, installation and other product characteristics. Renesas shall have no liability for malfunctions or damages arising out of the use of Renesas products beyond such specified ranges.
10. Although Renesas endeavors to improve the quality and reliability of its products, IC products have specific characteristics such as the occurrence of failure at a certain rate and malfunctions under certain use conditions. Please be sure to implement safety measures to guard against the possibility of physical injury, and injury or damage caused by fire in the event of the failure of a Renesas 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 applicable measures. Among others, since the evaluation of microcomputer software alone is very difficult, please evaluate the safety of the final products or system manufactured by you.
11. In case Renesas products listed in this document are detached from the products to which the Renesas products are attached or affixed, the risk of accident such as swallowing by infants and small children is very high. You should implement safety measures so that Renesas products may not be easily detached from your products. Renesas shall have no liability for damages arising out of such detachment.
12. This document may not be reproduced or duplicated, in any form, in whole or in part, without prior written approval from Renesas.
13. Please contact a Renesas sales office if you have any questions regarding the information contained in this document, Renesas semiconductor products, or if you have any other inquiries.