SH7216 Group

REJ06B0979-0100
Rev. 1.00
Jun 04, 2010

Configuration to Receive Data Frames Using the Controller Area Network and Data Transfer Controller

Summary

This application note describes the configuration example of the SH7216 microcomputers (MCUs) to receive data frames using the Controller Area Network, and to store data frames in on-chip RAM using the Data Transfer Controller.

Target Device

SH7216 MCU

Contents

1. Introduction

1.1 Specifications

When the SH7216 Controller Area Network receives a data frame, this application activates the SH7216 Data Transfer Controller and stores the data in on-chip RAM, and this application repeats the set of such behaviors for five times. After transferring data for five times is completed, this application generates an interrupt, sets the source to activate the Data Transfer Controller again in the interrupt processing, and waits for receiving the next data frame.

1.2 Modules Used

- Controller Area Network
  - Transmission speed: 1 Mbps
  - Receive mailbox: Mailbox 0
  - Mailbox 0 setting: Identifier: H'00A, standard format

- Data Transfer Controller
  - Transfer mode: Block transfer mode
  - Transfer data size: words
  - Source to activate: Controller Area Network Mailbox 0 data frame received interrupt
  - Transfer source: Controller Area Network Mailbox 0
  - Transfer destination: SH7216 on-chip RAM (H'FFF8 C000 to H'FFF8 C059, H'FFF8 C100 to H'FFF8 C159)
  - Block area: Transfer destination
  - Block size: 9 words
  - Number of transfers: 5
  - Transfer information table start address: H'FFF8 8800

1.3 Applicable Conditions

MCU	SH7216
Operating Frequencies	Internal clock: 200 MHz Bus clock: 50 MHz Peripheral clock: 50 MHz
Integrated Development Environment	Renesas Electronics Corporation High-performance Embedded Workshop Ver.4.05.01
C Compiler	Renesas Electronics SuperH RISC engine Family C/C++ Compiler Package Ver.9.03 Release 00
Compiler Options	-cpu=sh2afpu -fpu=single -include="$(WORKSPDIR)\inc" -object="$(CONFIGDIR)\$(FILELEAF).obj" -debug -gbr=auto -chgincpath -errorpath -global_volatile=0 -opt_range=all -infinite_loop=0 -del_vacant_loop=0 -struct_alloc=1 -nologo

1.4 Related Application Notes

For more information, refer to the following application note:

- SH7216 Group Example of Initialization

2. Applications

This application receives a data frame by the Controller Area Network, and stores the data frame in on-chip RAM by the Data Transfer Controller.

2.1 Overview of Modules

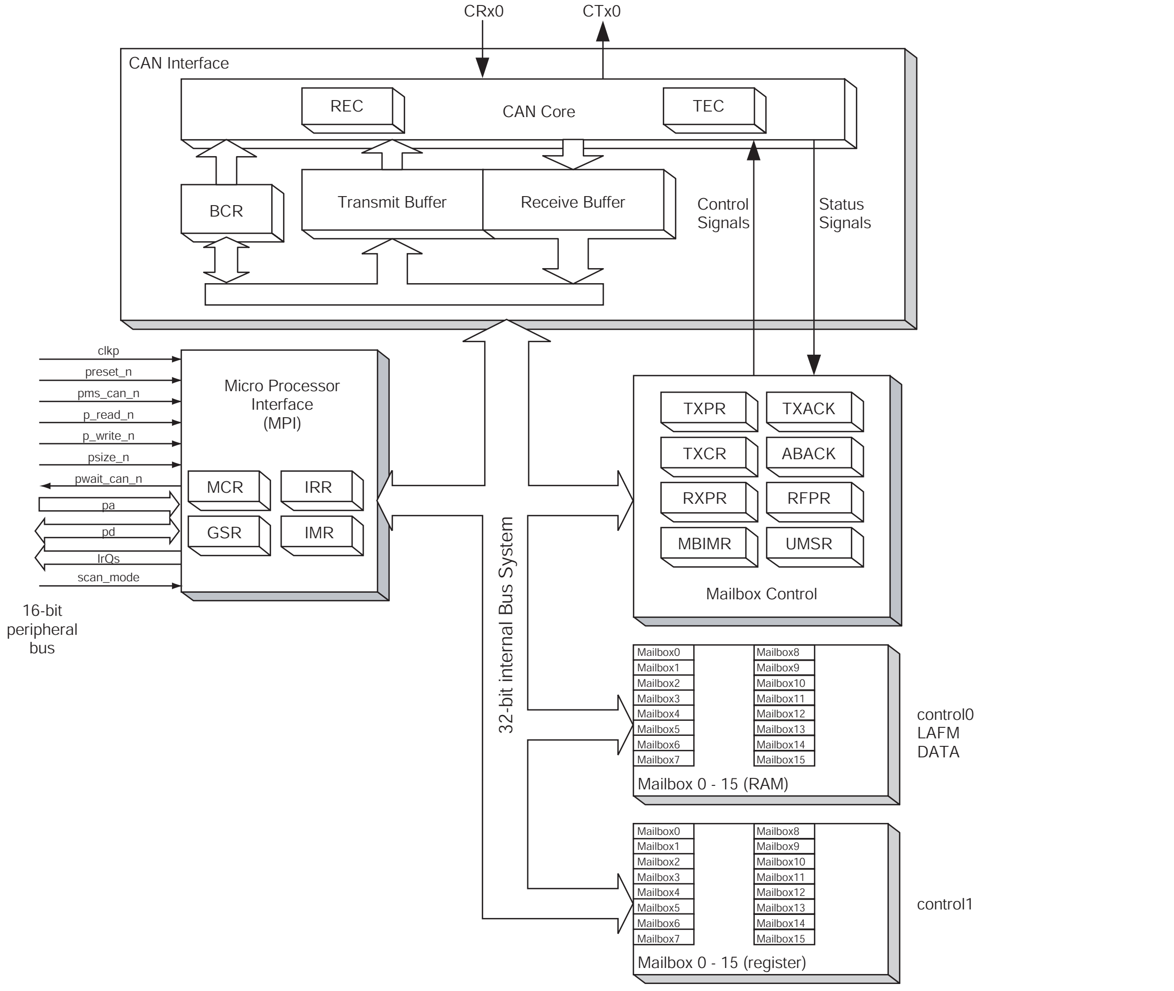
(1) Controller Area Network

The SH7216 includes a Controller Area Network module which is compliant with the CAN protocol, version 2.0B active, and ISO 11898.

The Controller Area Network module has 15 programmable mailboxes for transmission/reception, one mailbox for reception, and one programmable receive filtering mask to provide flexible communication procedure. Table 1 lists the features of the Controller Area Network. Figure 1 shows its block diagram. Table 2 lists interrupt sources. Sources to activate the Data Transfer Controller are mailbox 0 data frame received interrupt, and remote frame received frame only. For more information, refer to the Controller Area Network chapter in the SH7216 Group Hardware Manual.

Table 1 Controller Area Network Features

Item	Description
Protocol	CAN protocol, version 2.0B. Bit timing is compliant to ISO 11898
Number of mailboxes	16 (15 programmable transmit/receive mailbox, and one receive mailbox)
Transfer speed	Up to 1 Mbps
Number of interrupt sources	12
Test mode	Includes listen-only mode, and error passive mode



Note: RCAN-ET core is designed based on the 32-bit bus system, however, 16-bit bus interface applies to CPU on the RCAN-ET (including the MPI to CPU). Therefore, execute the longword (32-bit) access using two neighboring words (16-bit). The longword access explained in the Hardware Manual means to access by such two neighboring words.

Figure 1 Controller Area Network Block Diagram

Table 2 Controller Area Network Interrupt Sources

Interrupt	Source	Interrupt flag	Activating the Data Transfer Controller/Direct Memory Access Controller
ERS_0	Error passive	IRR5	Not allowed
	Bus off/Bus off recovery	IRR6	
	Error warning (TEC ≥ 96)	IRR3	
	Error warning (REC ≥ 96)	IRR4	
OVR_0	Message error detection	IRR13 [1]	
	Transition to Reset/halt/CAN sleep	IRR0	
	Overload frame transmission	IRR7	
	Unread message overwrite (overrun)	IRR9	
	Detection of CAN bus operation in CAN sleep mode	IRR1	
RM0_0 [2] RM1_0 [2]	Data frame reception	IRR1 [3]	Allowed [4]
	Remote frame reception	IRR2 [3]	
SLE_0	Message transmission/transmission disabled	IRR8	Not allowed

Notes: 1. Available only in test mode.

2. RM0_0 is an interrupt generated by the Remote frame pending flag for mailbox 0 (RFPR0 [0]) on the Data frame pending flag for mailbox 0 (RXPR0 [0]). RM1_0 is an interrupt generated by the Remote frame pending flag for mailbox n (RFPR0 [n]) or the Data frame pending flag for mailbox n (RXPR0 [n]) (n = 1 to 15).

3. IRR1 is a data frame received interrupt flag for mailboxes 0 to 15, and IRR2 is a remote frame request interrupt flag for mailboxes 0 to 15.

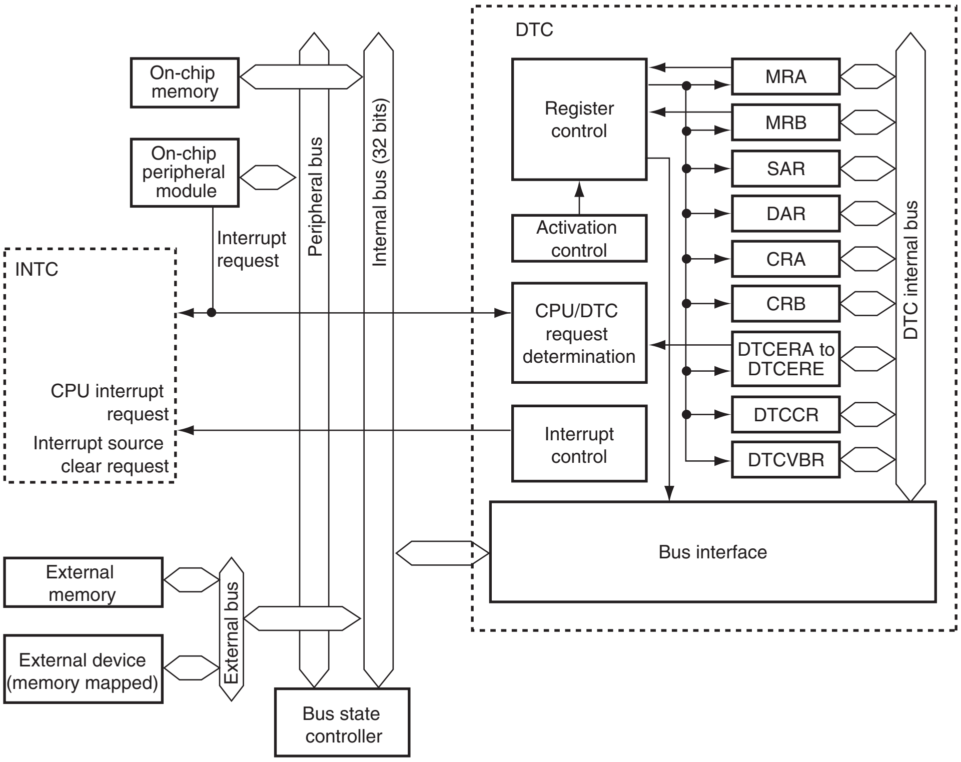
4. The Data Transfer Controller and the Direct Memory Access Controller can be activated only by the RM0_0 interrupt.

(2) Data Transfer Controller

The Data Transfer Controller is activated by an interrupt request, and transfers data. When it is activated, it reads the transfer information from the data area, transfers data, and writes back the transfer information after the transfer is completed. Table 3 lists the features of the Data Transfer Controller. Figure 2 shows its block diagram. For more information, refer to the Data Transfer Controller chapter in the SH7216 Group Hardware Manual.

Table 3 Data Transfer Controller Features

Item	Description
Transfer mode	Normal mode, repeat mode, and block transfer mode
Data size	Bytes, words, and long words
Chain transfer	Supported (Chain transfer is to transfer data for multiple times by an activation source)
Interrupt	A CPU interrupt using the DTC can be requested A CPU interrupt can be requested after "a transfer" is completed A CPU interrupt can be requested after transferring the specified data is completed
Read skip the transfer information	Supported
Short address mode	Supported



[Legend]

MRA, MRB:	DTC mode registers A, B
SAR:	DTC source address register
DAR:	DTC destination address register
CRA, CRB:	DTC transfer count registers A, B
DTCERA to DTCERE:	DTC enable registers A to E
DTCCR:	DTC control register
DTCVBR:	DTC vector base register

Figure 2 Data Transfer Controller Block Diagram

2.2 Data Transfer Controller Transfer Information

The Data Transfer Controller transfers data according to the transfer information.

Allocate the transfer information on read- and write-enabled RAM. Figure 3 shows an example to allocate the transfer information. The start address of the transfer information is indicated by the DTC vector table. Figure 4 shows the relationship between the DTC vector table and transfer information.

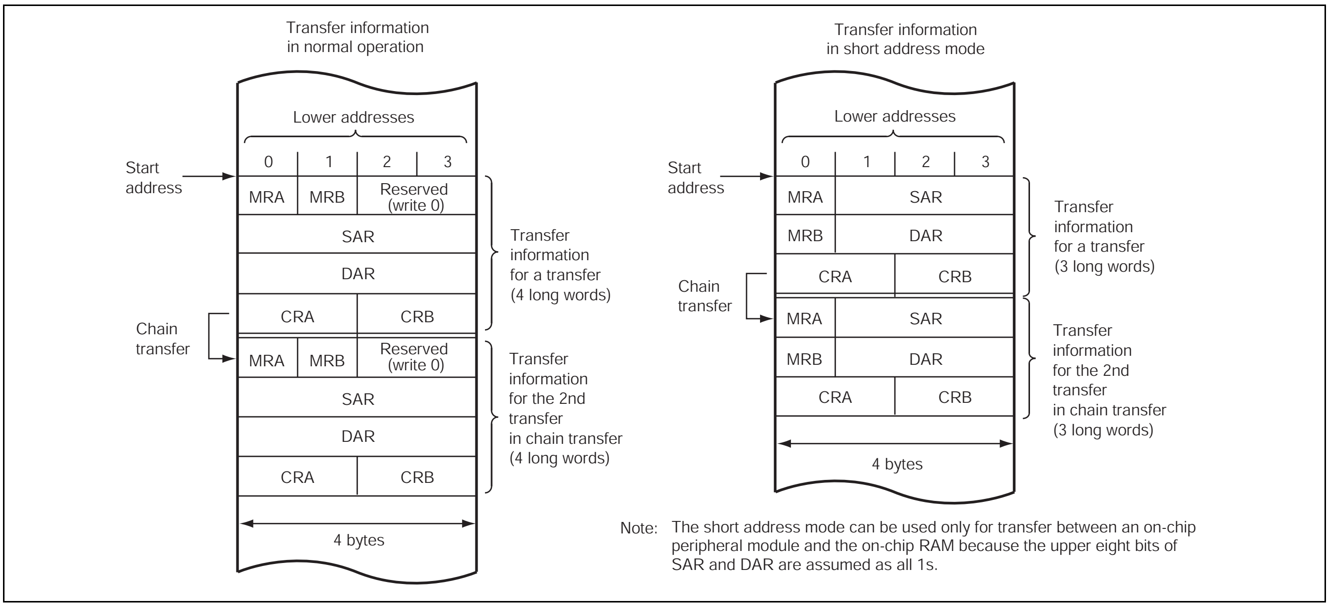
The Data Transfer Controller reads the start address of the transfer information from the DTC vector table, by source. Then, it reads the transfer information from the start address.

Specify the start address of the transfer information in multiples of four in the DTC vector table. When specifying the value other than in multiples of four, the Data Transfer Controller ignores lower two bits and reads the transfer information.

The Data Transfer Controller specifies the transfer source address in the DTC source address register (SAR), the transfer destination address in the DTC destination address register (DAR), and the number of transfers in the DTC transfer count register A (CRA) as the transfer information.

When a transfer is completed, the Data Transfer Controller increments, decrements or keeps values in the SAR and DAR depending on values in the DTC mode register A (MRA), and DTC mode register B (MRB). The Data Transfer Controller decrements the CRA value when a transfer is completed.

When a transfer is completed, the Data Transfer Controller writes back these updated register information to the transfer information.



Note: The short address mode can be used only for transfer between an on-chip peripheral module and the on-chip RAM because the upper eight bits of SAR and DAR are assumed as all 1s.

Figure 3 Allocating the Transfer Information on the Data Area

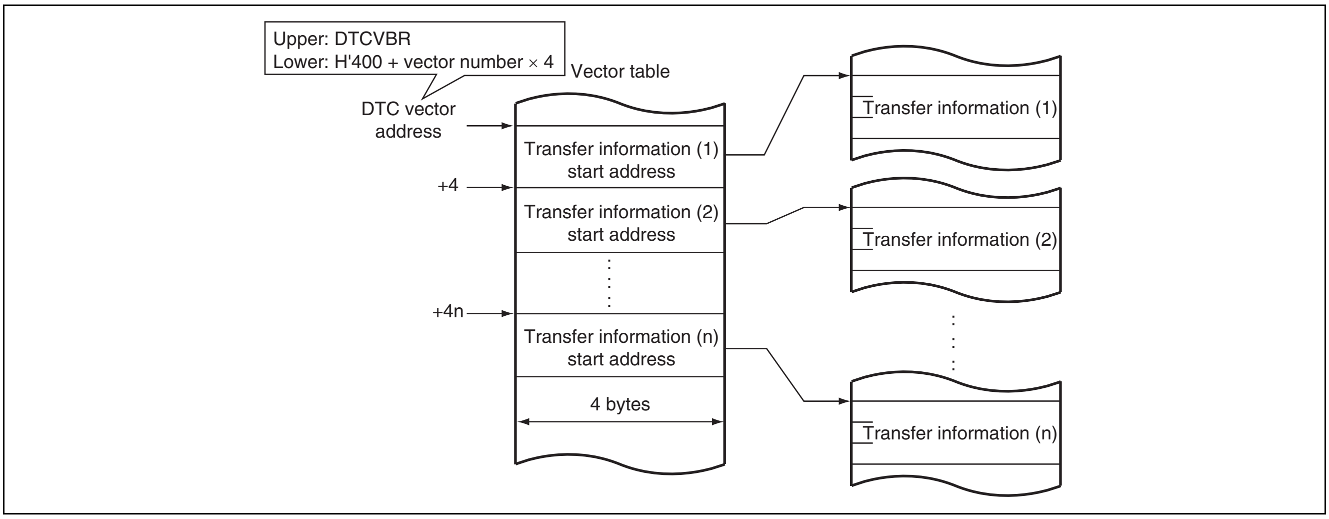

Upper: DTCVBR
Lower: H'400 + vector number × 4
Vector table
DTC vector address
+4
+4n
Transfer information (1) start address
Transfer information (2) start address
Transfer information (n) start address
4 bytes
Transfer information (1)
Transfer information (2)
Transfer information (n)

Figure 4 Relationship between the DTC Vector Table and Transfer Information

2.3 Configuration Procedure

(1) Steps to configure the Controller Area Network

Configure the Controller Area Network in reset mode (configuration mode). After configuration is completed, clear the reset mode to join the CAN bus activity. To activate the Data Transfer Controller by the Controller Area Network data frame received interrupt (RM0_0), set bit 1 in the Interrupt mask register, and bit 0 in the Mailbox interrupt mask register to enable interrupts.

Figure 5 and Figure 6 show flow charts for configuring the Controller Area Network. For details on register settings, refer to the Controller Area Network chapter in the SH7216 Group Hardware Manual.

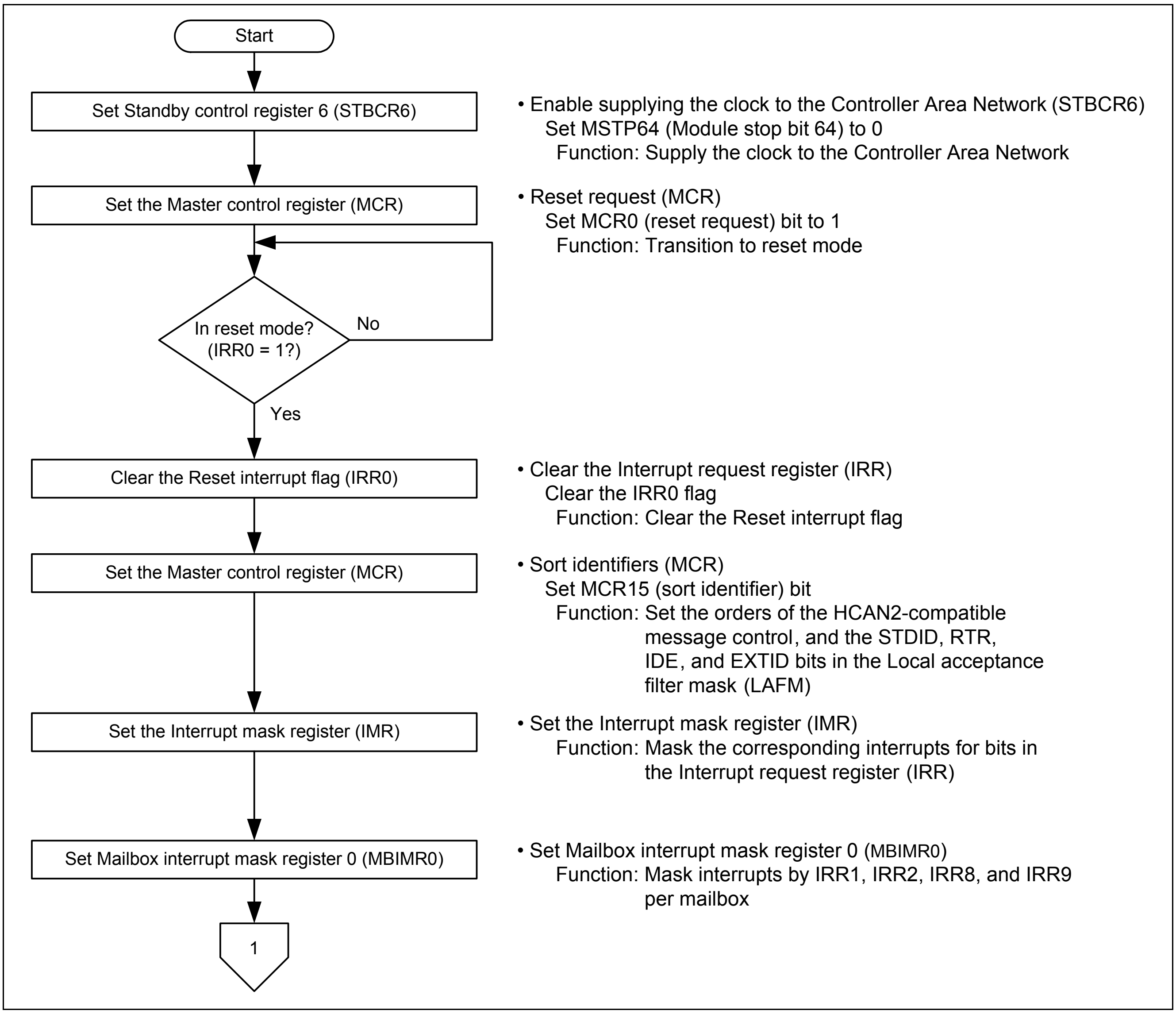
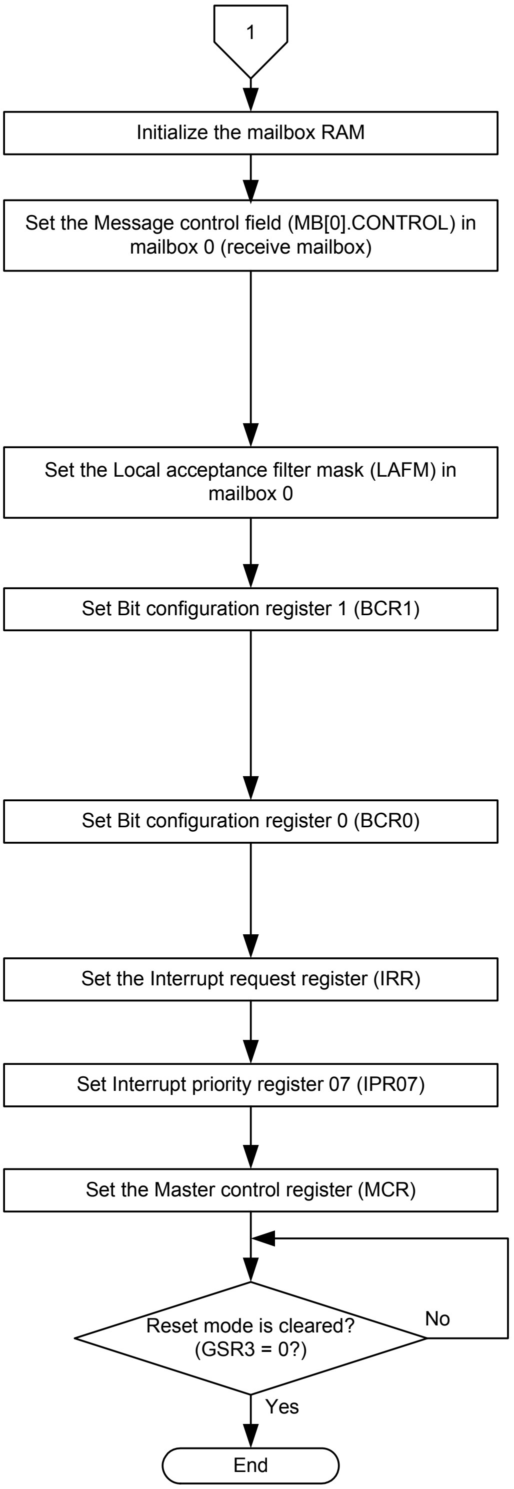

Start
Set Standby control register 6 (STBCR6)
• Enable supplying the clock to the Controller Area Network (STBCR6)
Set MSTP64 (Module stop bit 64) to 0
Function: Supply the clock to the Controller Area Network
Set the Master control register (MCR)
• Reset request (MCR)
Set MCR0 (reset request) bit to 1
Function: Transition to reset mode
In reset mode? (IRR0 = 1?)
No
Yes
Clear the Reset interrupt flag (IRR0)
• Clear the Interrupt request register (IRR)
Clear the IRR0 flag
Function: Clear the Reset interrupt flag
Set the Master control register (MCR)
• Sort identifiers (MCR)
Set MCR15 (sort identifier) bit
Function: Set the orders of the HCAN2-compatible message control, and the STDID, RTR, IDE, and EXTID bits in the Local acceptance filter mask (LAFM)
Set the Interrupt mask register (IMR)
• Set the Interrupt mask register (IMR)
Function: Mask the corresponding interrupts for bits in the Interrupt request register (IRR)
Set Mailbox interrupt mask register 0 (MBIMR0)
• Set Mailbox interrupt mask register 0 (MBIMR0)
Function: Mask interrupts by IRR1, IRR2, IRR8, and IRR9 per mailbox
1

Figure 5 Flow Chart for Configuring the Controller Area Network (1/2)


1
Initialize the mailbox RAM
Set the Message control field (MB[0].CONTROL) in mailbox 0 (receive mailbox)
Set the Local acceptance filter mask (LAFM) in mailbox 0
Set Bit configuration register 1 (BCR1)
Set Bit configuration register 0 (BCR0)
Set the Interrupt request register (IRR)
Set Interrupt priority register 07 (IPR07)
Set the Master control register (MCR)
Reset mode is cleared? (GSR3 = 0?)
No
Yes
End

- Clear the RAM area in mailbox

- Sets the Message control register (MB[0].CONTROL)
  Set MBC [2:0] (Mailbox configuration) bits to "B'010"
  Function: Receiving a remote frame or a data frame is enabled
  Set the IDE (Identifier extension) to 0
  Function: Specify the standard format
  Set the RTR (Remote transmission request) to 0
  Function: Specify a data frame
  Set STDID [10:0] (Standard identifier) bits to "H'00A"
  Function: Specify the identifier to H'00A

- Set the Local acceptance filter mask (LAFM)
  Set the LAFM to "H'0000 0000"
  Function: Mask is not specified

- Set Bit configuration register 1 (BCR1)
  Specify "B'1111" (16 time quanta) in TSG1 [3:0] bits (Time segment 1)
  Specify "B'111" (8 time quanta) in TSG2 [2:0] bits (Time segment 2)
  Specify "B'00" (1 time quantum) in SJW [1:0] bits (Resynchronization jump width)
  Function: Avoid moving the sample point substantially
  Specify the BSP (Bit sample point)
  Function: Specify to sample at one point or to sample at three points and decide by majority of them

- Set Bit configuration register 0 (BCR0)
  Specify "B'0000 0000" in BRP [7:0] bits (Baud rate prescaler)
  Function: Specify 1 time quantum as 2 × peripheral bus clock
  Transfer speed is 1 Mbps by TSG1, TSG2, and baud rate prescaler
  (Refer to 2.3 Bit configuration and transfer speed)

- Clear the Interrupt request register (IRR)
  Clear IRR0 to IRR15 bits
  Function: Clear interrupt flags

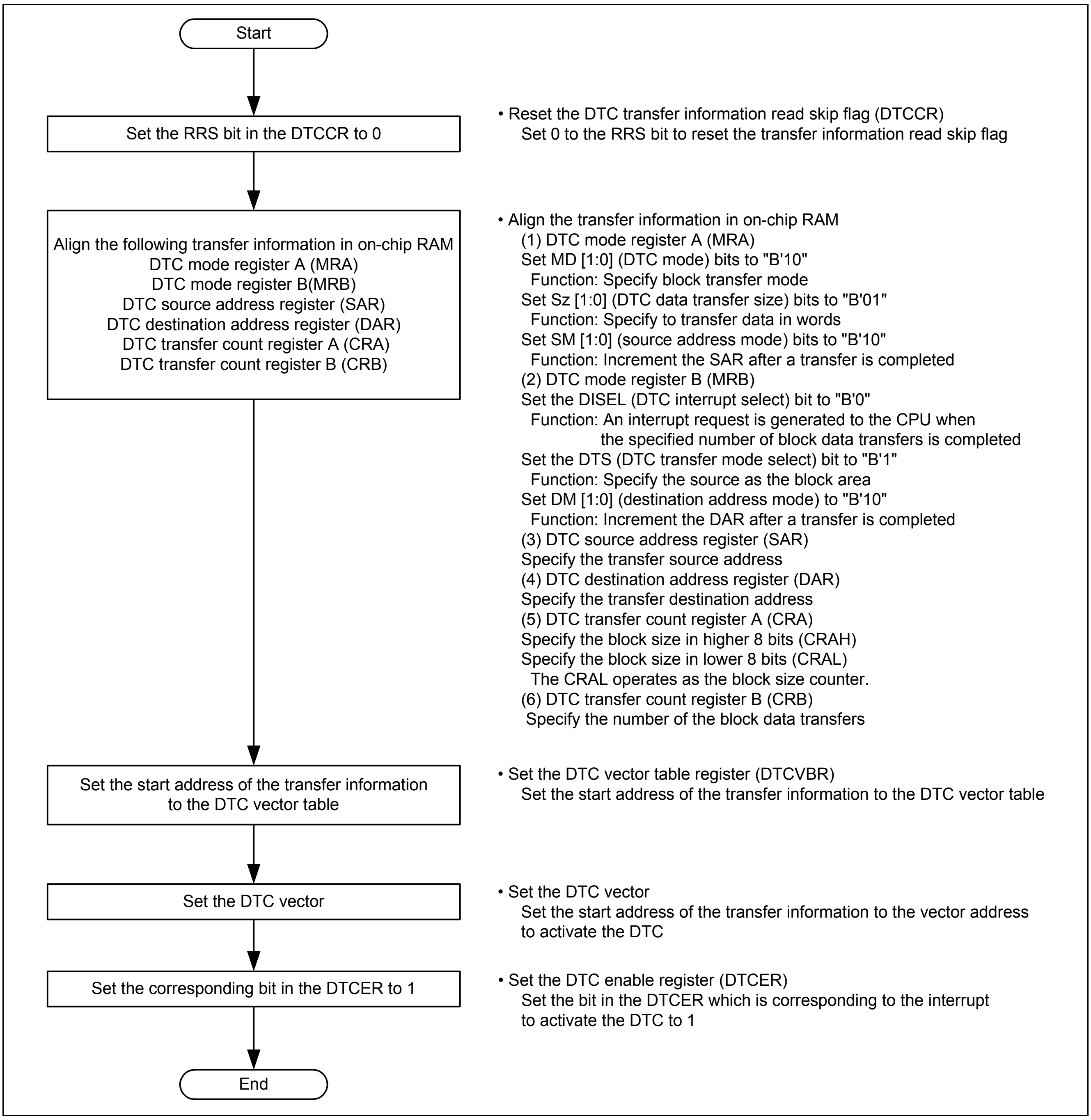
- Set Interrupt priority register 07 (IPR07)
  Specify "H'A" in IPR07
  Function: Specify the interrupt priority as 10

- Set the Master control register (MCR)
  Clear MCR0 (reset request)
  Function: Clear the reset mode transition request
  Clear MCR1 (halt request)
  Function: Clear the halt mode transition request

Figure 6 Flow Chart for Configuring the Controller Area Network (2/2)

(2) Steps to configure the Data Transfer Controller

Figure 7 shows the flow chart for configuring the Data Transfer Controller. For more information on register settings, refer to the Data Transfer Controller chapter in the SH7216 Group Hardware Manual.


Start
Set the RRS bit in the DTCCR to 0
Align the following transfer information in on-chip RAM
DTC mode register A (MRA)
DTC mode register B(MRB)
DTC source address register (SAR)
DTC destination address register (DAR)
DTC transfer count register A (CRA)
DTC transfer count register B (CRB)
Set the start address of the transfer information to the DTC vector table
Set the DTC vector
Set the corresponding bit in the DTCER to 1
End

- Reset the DTC transfer information read skip flag (DTCCR)
  Set 0 to the RRS bit to reset the transfer information read skip flag
- Align the transfer information in on-chip RAM
  (1) DTC mode register A (MRA)
  Set MD [1:0] (DTC mode) bits to "B'10"
  Function: Specify block transfer mode
  Set Sz [1:0] (DTC data transfer size) bits to "B'01"
  Function: Specify to transfer data in words
  Set SM [1:0] (source address mode) bits to "B'10"
  Function: Increment the SAR after a transfer is completed
  (2) DTC mode register B (MRB)
  Set the DISEL (DTC interrupt select) bit to "B'0"
  Function: An interrupt request is generated to the CPU when the specified number of block data transfers is completed
  Set the DTS (DTC transfer mode select) bit to "B'1"
  Function: Specify the source as the block area
  Set DM [1:0] (destination address mode) to "B'10"
  Function: Increment the DAR after a transfer is completed
  (3) DTC source address register (SAR)
  Specify the transfer source address
  (4) DTC destination address register (DAR)
  Specify the transfer destination address
  (5) DTC transfer count register A (CRA)
  Specify the block size in higher 8 bits (CRAH)
  Specify the block size in lower 8 bits (CRAL)
  The CRAL operates as the block size counter.
  (6) DTC transfer count register B (CRB)
  Specify the number of the block data transfers
- Set the DTC vector table register (DTCVBR)
  Set the start address of the transfer information to the DTC vector table
- Set the DTC vector
  Set the start address of the transfer information to the vector address to activate the DTC
- Set the DTC enable register (DTCER)
  Set the bit in the DTCER which is corresponding to the interrupt to activate the DTC to 1

Figure 7 Flow Chart for Configuring the Data Transfer Controller

2.4 Bit Configuration and Transmission Speed

One bit in the Controller Area Network consists of the following four segments:

1. Synchronization segment (SS)
2. Propagation time segment (PRSEG)
3. Phase buffer segment 1 (PHSEG1)
4. Phase buffer segment 2 (PHSEG2)

Each segment is composed of the reference time Tq (time quanta). Figure 8 shows the bit configuration example when SS = 1 Tq, PRSEG = 8 Tq, PHSEG1 = 8 Tq, and PHSEG2 = 8 Tq.

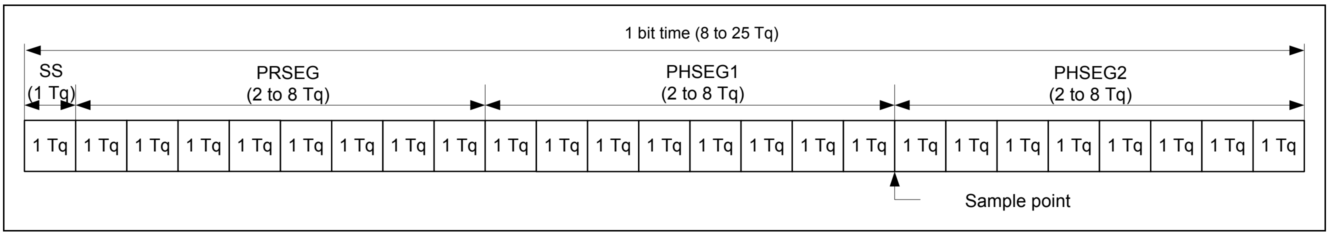


Figure 8 Bit Configuration

CAN defines $1\ \text{Tq} = \dfrac{2 \times (\text{BRP}\,[7{:}0] + 1)}{\text{Peripheral bus clock}}$ By this formula, the transmission speed is calculated as follows:

$$\text{Transmission speed} = \frac{\text{Peripheral bus clock}}{2 \times (\text{BRP}\,[7{:}0] + 1) \times (\text{the number of Tqs/bit})}$$

$$= \frac{\text{Peripheral Bus Clock}}{2 \times (\text{BRP}\,[7{:}0] + 1) \times \{(\text{TSG1}[3{:}0]{+}1) + (\text{TSG2}\,[2{:}0] + 1) + 1\}}$$

The Controller Area Network sets the number of Tqs of PRSEG + PHSEG1 to bits TSG1 [3:0] in the Bit configuration register 1 (BCR1), and the number of Tqs of PSEG2 to bits TSG2 [2:0] in BCR1 register (Value + 1 is the number of Tqs). Also, the number of peripheral bus clocks for 1 Tq is set to bits BRP [7:0] in Bit configuration register 0 (BCR0).

In the following description, bits BRP [7:0], TSG1 [3:0], and TSG2 [2:0] are register values, and bits BRP, TSEG1. TSEG2, and SJW are the corresponding values for the register values. For the corresponding values for register values, refer to the Controller Area Network chapter in the SH7216 Group Hardware Manual.

Following is the restriction on setting the bit configuration register.

TSEG1 (Min.) > TSEG2 ≥ SJW (Max.) (SJW = 1 to 4)

SJW is the resynchronization jump width. It is a segment that lengthens phase buffer segment 1 or shortens phase buffer segment 2 to correct the phase difference.

8 ≤ TSEG1 + TSEG2 + 1 ≤ 25 time quanta
TSEG2 ≥ 2

As this sample program specifies the peripheral bus clock as 50 MHz, BRP [7:0] = 0, TSG1 [3:0] = 15, and TSEG2 [2:0] = 7, the transmission speed is calculated as follows:

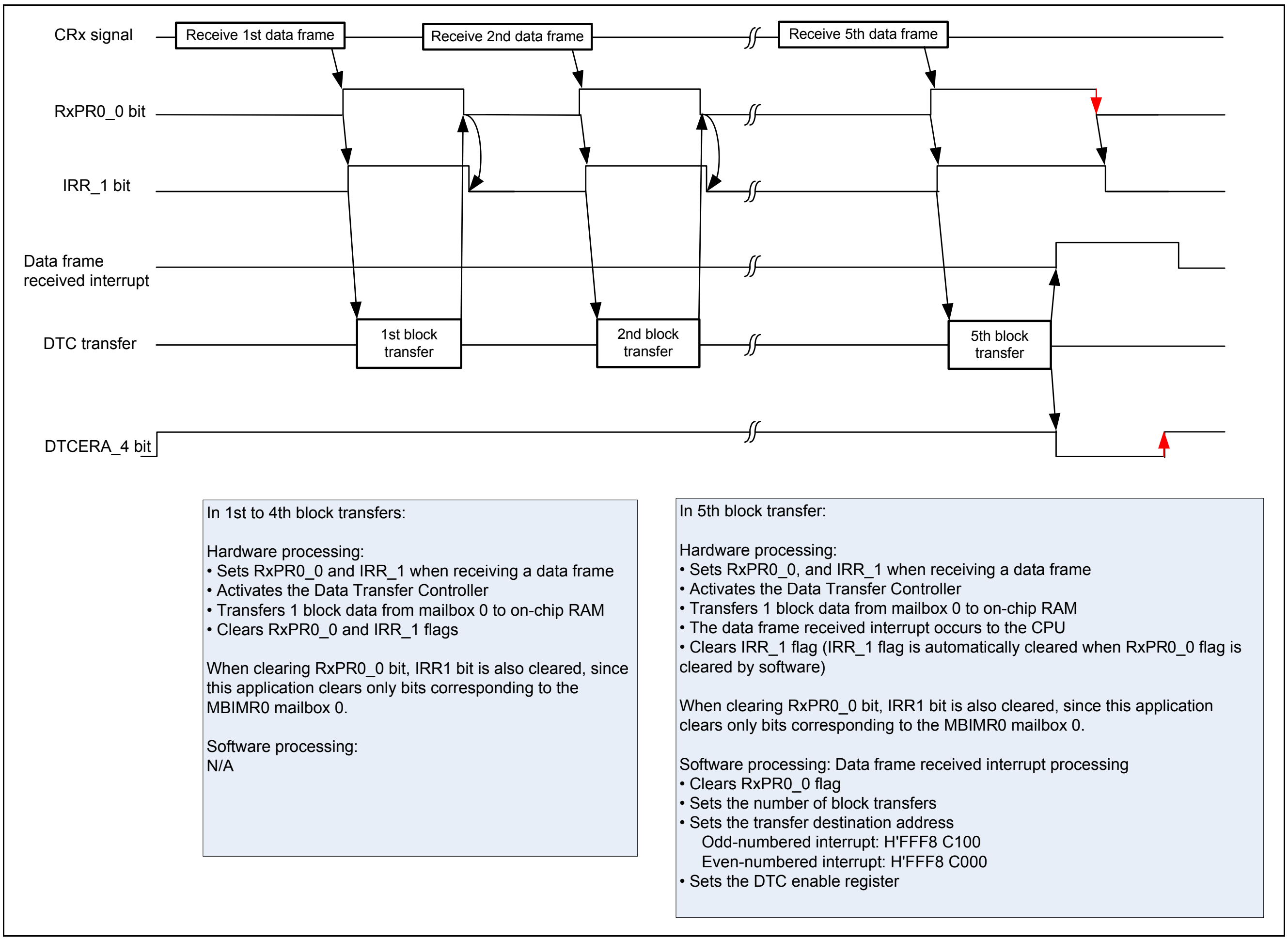
$$\text{Transmission speed} = \frac{50\text{M}}{2 \times (0 + 1) \times \{(15 + 1) + (7 + 1) + 1\}} = 1\text{M....1 Mbps}$$

2.5 Sample Program Operation

This sample program receives a standard CAN data frame with identifier H'00A in mailbox 0, and generates an interrupt. Then, it uses the data frame received interrupt (RM0_0) to activate the Data Transfer Controller and transfers data in mailbox 0 to on-chip RAM in blocks. When the specified number of block transfers is completed, it generates an interrupt to the CPU, and resets the Data Transfer Controller in the interrupt processing. The sample program has two transfer destination on-chip RAM areas to switch between these areas per interrupt processing.

Note: When the Data Transfer Controller transfers data in mailbox 0 using the data frame received interrupt (RM0_0) as an interrupt source, data from Message control field 0 (CONTROL0) to Message control field 1 (CONTROL1) in mailbox 0 must be included.

Figure 9 shows the sample program operation (overview). Figure 10 shows the sample program transfer area.



In 1st to 4th block transfers:

Hardware processing:
- Sets RxPR0_0 and IRR_1 when receiving a data frame
- Activates the Data Transfer Controller
- Transfers 1 block data from mailbox 0 to on-chip RAM
- Clears RxPR0_0 and IRR_1 flags

When clearing RxPR0_0 bit, IRR1 bit is also cleared, since this application clears only bits corresponding to the MBIMR0 mailbox 0.

Software processing:
N/A

In 5th block transfer:

Hardware processing:
- Sets RxPR0_0, and IRR_1 when receiving a data frame
- Activates the Data Transfer Controller
- Transfers 1 block data from mailbox 0 to on-chip RAM
- The data frame received interrupt occurs to the CPU
- Clears IRR_1 flag (IRR_1 flag is automatically cleared when RxPR0_0 flag is cleared by software)

When clearing RxPR0_0 bit, IRR1 bit is also cleared, since this application clears only bits corresponding to the MBIMR0 mailbox 0.

Software processing: Data frame received interrupt processing
- Clears RxPR0_0 flag
- Sets the number of block transfers
- Sets the transfer destination address
  Odd-numbered interrupt: H'FFF8 C100
  Even-numbered interrupt: H'FFF8 C000
- Sets the DTC enable register

Figure 9 Sample Program Operation (Overview)

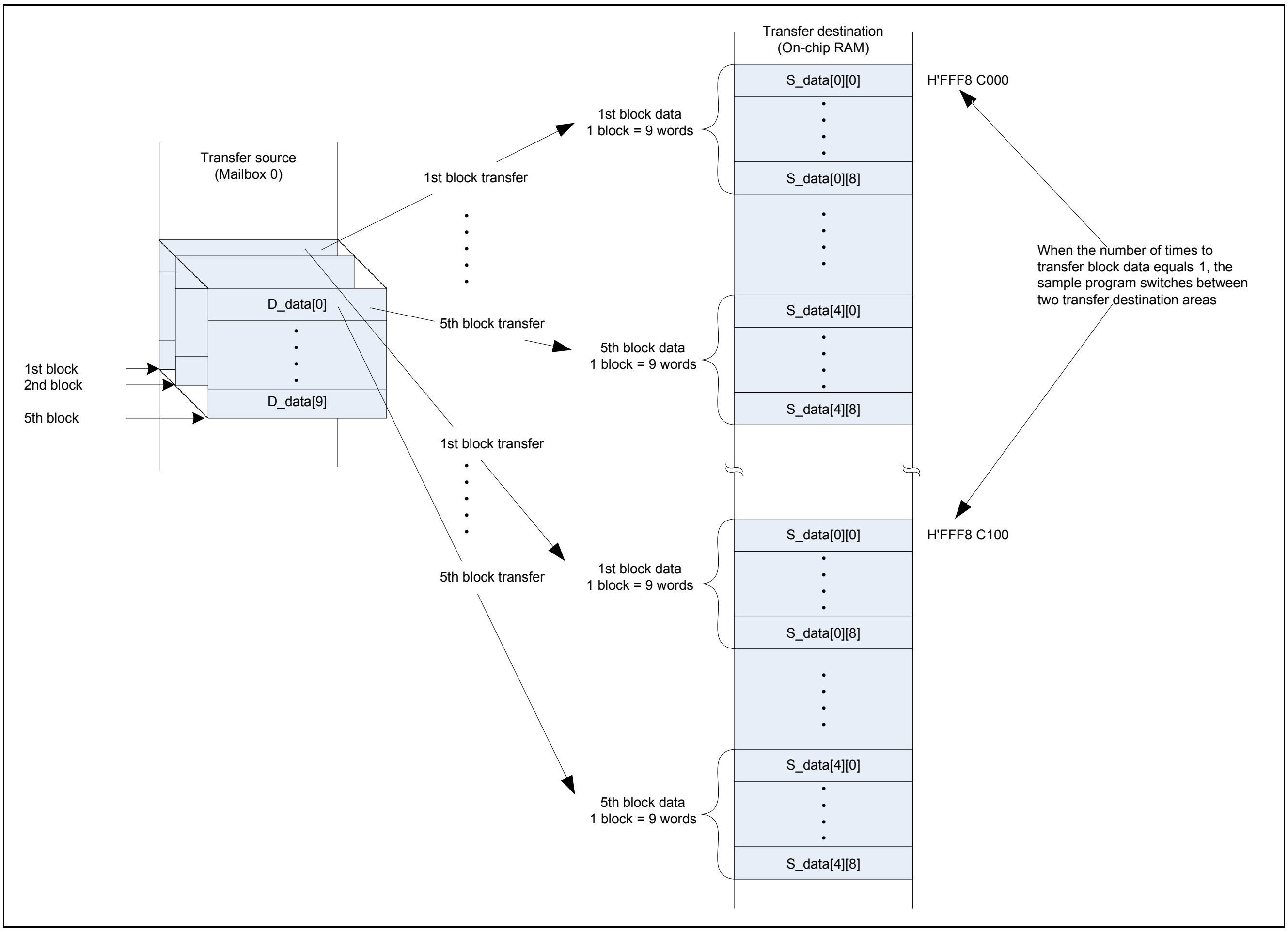


Figure 10 Sample Program Transfer Area

2.6 Sample Program Flow Chart

Table 4 lists setting examples of the Controller Area Network. Table 5 lists setting examples of the Data Transfer Controller. Figure 11 shows the flow chart of the sample program.

Table 4 Controller Area Network Settings

Register Name	Address	Setting	Description
Standby control register 6 (STBCR6)	H'FFFE 041C	H'8F	MSTP64 = "0": Controller Area Network is running
Master control register (MCR)	H'FFFF D000	H'0001	MCR0 = "1": Reset mode transition request
		H'8001	MCR15 = "1": The order of the RCAN-ET message and of the HCAN2 message are different
		H'8000	MCR0 = "0": Clears reset mode
Mailbox interrupt mask register 0 (MBIMR0)	H'FFFF D052	H'FFFE	MBIMR00 = "0": Enables the receive interrupt in mailbox 0
Interrupt mask register (IMR)	H'FFFF D00A	H'FFFD	Enables the data frame received interrupt
Bit configuration register 1 (BCR1)	H'FFFF D004	H'F700	TSG1 [3:0] = "B'1111": PRSEG + PHSEG1 = 16 Tq TSG2 [2:0] = "B'111": PHSEG2 = 8 Tq SJW = "0": SJW = 1 Tq BSP = "0": Samples at one point
Bit configuration register 0 (BCR0)	H'FFFF D006	H'0000	BRP [7:0] = "0": 1 Tq = 2 × Pφ
Message control field 1 in mailbox 0 (MB[0].CONTROL1)	H'FFFF D110	H'0200	MBC [2:0] = "B'010": Receiving a data frame and remote frame is enabled

Message control field 0 in mailbox 1 (MB[1].CONTROL0)	H'FFFF D120	H'0028 0000	IDE = "0": Standard format
			RTR = "0": Data frame
			STDID [10:0] = "H'00A": Standard identifier is H'00A

Mailbox 0 local acceptance filter mask (MB[0].LAFM)	H'FFFF D104	H'0000 0000	Clear: Mask is not specified

Table 5 Data Transfer Controller (DTC) Settings

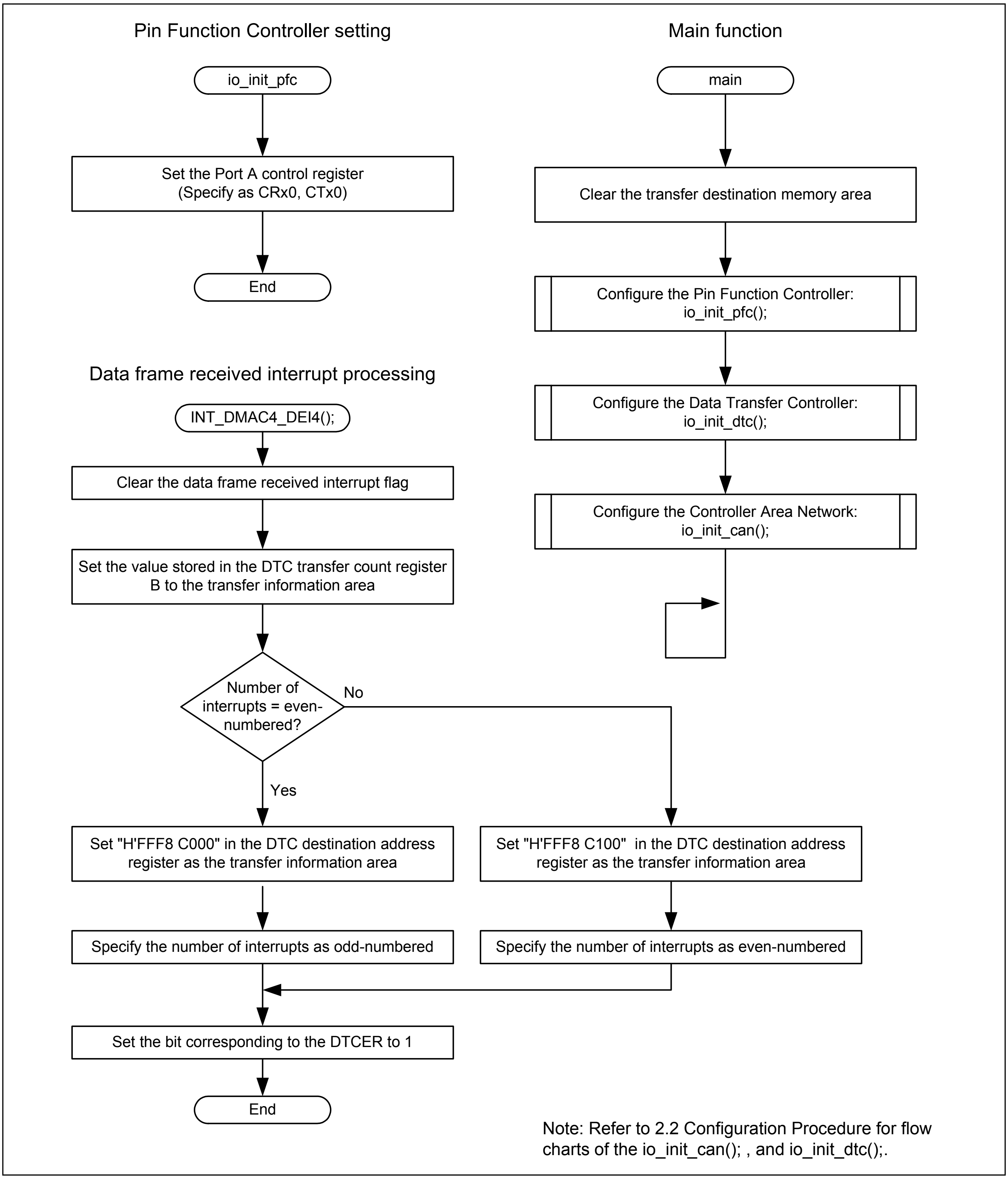
Register Name	Address	Setting	Description
DTC control register (DTCCR)	H'FFFE 6010	H'00	RRS = "0": Disables the DTC transfer information read skip flag
DTC vector base register (DTCVBR)	H'FFFE 6014	H'FFF8 8000	DTC vector table start address: H'FFFF A000
DTC enable register A (DTCERA)	H'FFFE 6000	H'0010	RM0_0 = "1": Sets the DTC activation source as RM0_0

Table 6 Transfer Information Settings

Register Name	Setting	Description
DTC mode register (MRA)	H'98	• MD = "B'10": Block transfer mode • Sz = "B'01": Transfer in words • SM = "B'10": Increments the SAR after a transfer is completed
DTC mode register (MRB)	H'18	• CHNE = "0": Disables the chain transfer • DISEL = "0": Interrupt occurs when the specified number of transfers is completed • DTS = "B'1": Specifies the source as the block area • DM = "B'10": Increments the DAR after a transfer is completed
DTC source address register (SAR)	H'FFFF D100	Transfer source start address
DTC destination address register (DAR)	H'FFF8 C000 H'FFF8 C100	Transfer destination start address
DTC transfer count register A (CRA)	H'0909	Block size: 9
DTC transfer count register B (CRB)	H'0005	Number of block transfers: 5

Table 7 Variables to Use

Variable Name	Type	Description	Module Name
SrcData [9]	unsigned char	Stores the DTC transfer source data	main function
DstData_even [5] [9]	unsigned short	Stores the DTC transfer destination data on even-numbered block transfer	main function
DstData_odd [5] [9]	unsigned short	Stores the DTC transfer destination data on odd-numbered block transfer	main function


Pin Function Controller setting
io_init_pfc
Set the Port A control register (Specify as CRx0, CTx0)
End
Main function
main
Clear the transfer destination memory area
Configure the Pin Function Controller: io_init_pfc();
Configure the Data Transfer Controller: io_init_dtc();
Configure the Controller Area Network: io_init_can();
Data frame received interrupt processing
INT_DMAC4_DEI4();
Clear the data frame received interrupt flag
Set the value stored in the DTC transfer count register B to the transfer information area
Number of interrupts = even-numbered?
No
Yes
Set "H'FFF8 C000" in the DTC destination address register as the transfer information area
Set "H'FFF8 C100" in the DTC destination address register as the transfer information area
Specify the number of interrupts as odd-numbered
Specify the number of interrupts as even-numbered
Set the bit corresponding to the DTCER to 1
End

Note: Refer to 2.2 Configuration Procedure for flow charts of the io_init_can(); , and io_init_dtc();.

Figure 11 Sample Program Flow Chart

3. Sample Program Listing

3.1 Sample Program List "main.c" (1/6)

```
/*******************************************************************************
*   DISCLAIMER
*
*   This software is supplied by Renesas Electronics Corp. and is only
*   intended for use with Renesas products.  No other uses are authorized.
*
*   This software is owned by Renesas Electronics Corp. and is protected under
*   all applicable laws, including copyright laws.
*
*   THIS SOFTWARE IS PROVIDED "AS IS" AND RENESAS MAKES NO WARRANTIES
*   REGARDING THIS SOFTWARE, WHETHER EXPRESS, IMPLIED OR STATUTORY,
*   INCLUDING BUT NOT LIMITED TO WARRANTIES OF MERCHANTABILITY, FITNESS FOR A
*   PARTICULAR PURPOSE AND NON-INFRINGEMENT.  ALL SUCH WARRANTIES ARE EXPRESSLY
*   DISCLAIMED.
*
*   TO THE MAXIMUM EXTENT PERMITTED NOT PROHIBITED BY LAW, NEITHER RENESAS
*   ELECTRONICS CORP. NOR ANY OF ITS AFFILIATED COMPANIES SHALL BE LIABLE
*   FOR ANY DIRECT, INDIRECT, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES
*   FOR ANY REASON RELATED TO THIS SOFTWARE, EVEN IF RENESAS OR ITS
*   AFFILIATES HAVE BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.
*
*   Renesas reserves the right, without notice, to make changes to this
*   software and to discontinue the availability of this software.
*   By using this software, you agree to the additional terms and
*   conditions found by accessing the following link:
*   http://www.renesas.com/disclaimer
********************************************************************************
*   (C) 2010 Renesas Electronics Corporation. All rights reserved.
*""FILE COMMENT""*********** Technical reference data **************************
*   System Name : SH7216 Sample Program
*   File Name   : main.c
*   Abstract    : DTC+CAN Module Application (Data Frame Receive)
*   Version     : 1.00.00
*   Device      : SH7216
*   Tool-Chain  : High-performance Embedded Workshop (Ver.4.07.00).
*               : C/C++ compiler package for the SuperH RISC engine family
*               :                             (Ver.9.03 Release00).
*   OS          : None
*   H/W Platform: R0K572167 (CPU board)
*   Description :
********************************************************************************
*   History     : Apr.28,2010 Ver.1.00.00
*""FILE COMMENT END""**********************************************************/
#include "iodefine.h"
#include "dtc.h"

```

3.2 Sample Program List "main.c" (2/6)

```
/* ---- prototype declaration ---- */
void main(void);
void io_init_pfc(void);
void io_init_dtc(unsigned long sar, unsigned long dar, unsigned char block_size, unsigned short count);
void io_init_can(void);

/* ---- Global variable ---- */
#pragma section DTC_VECT_TABLE
unsigned long dtc_vect[512];  /* Vector table area */

#pragma section D_DATA_O
unsigned short DstData_odd[DTC_COUNT][DTC_BLOCK_LENG];   /* Transfer destination */
                                                 /* memory area */

#pragma section D_DATA_E
unsigned short DstData_even[DTC_COUNT][DTC_BLOCK_LENG];  /* Transfer destination */
                                                 /* memory area */

#pragma section
int Int_Flag_EvenOdd = 1;             /* Interrupt occur even- and odd-numbered flag */

/*""FUNC COMMENT""**************************************************************
 * ID           :
 * Outline      : Sample program main
 *------------------------------------------------------------------------------
 * Include      : "iodefine.h"
 *------------------------------------------------------------------------------
 * Declaration  : void main(void);
 *------------------------------------------------------------------------------
 * Description  : Configures the PFC, DTC,  and CAN.
 *------------------------------------------------------------------------------
 * Argument     : void
 *------------------------------------------------------------------------------
 * Return Value : void
 *------------------------------------------------------------------------------
 * Note         : None
 *""FUNC COMMENT END""**********************************************************/
void main(void)
{
  int i,j;

  /* ==== Clears the transfer destination memory area ==== */
  for(i = 0; i < DTC_COUNT; i++){
    for(j = 0; j < DTC_BLOCK_LENG; j++){
        DstData_odd[i][j] = 0x0000;
        DstData_even[i][j] = 0x0000;
    }
  }

```

3.3 Sample Program List "main.c" (3/6)

```
  /* ==== Clears the vector table transfer information area ==== */
  for(i = 0; i < 512; i++){
    dtc_vect[i] = 0x00000000;
  }

  /* ==== Configures the PFC ==== */
  io_init_pfc();

  /* ==== Configures the DTC ==== */
  io_init_dtc((unsigned long)RCANET.MB, (unsigned long)DstData_odd, DTC_BLOCK_LENG, DTC_COUNT);

  /* ==== Configures the CAN ==== */
  io_init_can();

  while(1){
    if(Int_Flag_EvenOdd == 0){
        /* ---- DTC end interrupt completed at odd-numbered times ---- */
        while(Int_Flag_EvenOdd == 0){
        }
    }
    else{
        /* ---- DTC end interrupt completed at even-numbered times ---- */
        while(Int_Flag_EvenOdd == 1){
        }
    }
  }
}

/*""FUNC COMMENT""***************************************************************
 * ID           :
 * Outline      : PFC configuration
 *-------------------------------------------------------------------------------
 * Include      : "iodefine.h"
 *-------------------------------------------------------------------------------
 * Declaration  : void io_init_pfc(void);
 *-------------------------------------------------------------------------------
 * Description  : Configures the pin functions (CRx0 input and CTx0 output).
 *-------------------------------------------------------------------------------
 * Argument     : void
 *-------------------------------------------------------------------------------
 * Return Value : void
 *-------------------------------------------------------------------------------
 * Note         : None
 *""FUNC COMMENT END""**********************************************************/
void io_init_pfc(void)
{
  /* ==== Configures the PFC ==== */
  PFC.PACRL1.BIT.PA0MD = 0x5;            /* Set CRx0 */
  PFC.PACRL1.BIT.PA1MD = 0x5;            /* Set CTx0 */
}

```

3.4 Sample Program List "main.c" (4/6)

```
/*""FUNC COMMENT""**************************************************************
 * ID             :
 * Outline        : Data Transfer Controller Configuration
 *------------------------------------------------------------------------------
 * Include        : "iodefine.h"
 *------------------------------------------------------------------------------
 * Declaration    : void io_init_dtc(void);
 *------------------------------------------------------------------------------
 * Description    : Configures the Data Transfer Controller.
 *                : Sets the DTC to transfer mailbox 0 (block size is 9) to on-chip
 *                : RAM for five times in block transfer mode.
 *------------------------------------------------------------------------------
 * Arguments      : unsigned long sar    : Transfer source address
 *                : unsigned long dar    : Transfer destination address
 *                : unsigned short num   : Block size
 *                : unsigned short num2  : Number of transfers
 *------------------------------------------------------------------------------
 * Return Value   : void
 *------------------------------------------------------------------------------
 * Note           : None
 *""FUNC COMMENT END""**********************************************************/
void io_init_dtc(unsigned long sar, unsigned long dar, unsigned char block_size, unsigned
short count)
{
  /* ---- Sets the DTC control register ---- */
  DTC.DTCCR.BYTE = 0x00;          /* No read skip for transfer information */

  /* ==== Sets the DTC transfer information ==== */
  /* ---- Sets the DTC mode register A ---- */
  DTC_REG.MRA = 0x98;             /* Block transfer mode     */
                              /* Transfer data in words  */
                              /* Increment the SAR after a transfer is */
                              /* completed */
  /* ---- Sets the DTC mode register B ---- */
  DTC_REG.MRB = 0x18;             /* Disable the chain transfer  */
                              /* Interrupt when the specified number of */
                              /* transfers is completed  */
                              /* Specify the source as the block area   */
                              /* Increment the DAR after a transfer is  */
                              /* completed  */
  /* ---- Sets the DTC source address register ---- */
  DTC_REG.SAR = (unsigned long)sar;
  /* ---- Sets the DTC destination address register ---- */
  DTC_REG.DAR = (unsigned long)dar;
  /* ---- Sets the DTC transfer count register ---- */
  DTC_REG.CRA.BYTE.H = block_size;        /* Block size          */
  DTC_REG.CRA.BYTE.L = block_size;        /* Block size counter  */
  DTC_REG.CRB = count;                /* Number of transfers */
  /* ---- Sets the DTC vector base register ---- */
  DTC.DTCVBR = DTC_VECT_BASE;
```

3.5 Sample Program List "main.c" (5/6)

```
  /* ---- Sets the DTC vector ---- */
  dtc_vect[0x5a8/sizeof(unsigned long)] = (unsigned long)&DTC_REG;

  /* ---- Sets the DTC enable register A ---- */
  DTC.DTCERA.BIT.RM0 = 1;         /* interrupt source RM0 */
}

/*""FUNC COMMENT""***************************************************************
 * ID            :
 * Outline       : Controller Area Network setting
 *-------------------------------------------------------------------------------
 * Include       : "iodefine.h"
 *-------------------------------------------------------------------------------
 * Declaration   : void io_init_can(void);
 *-------------------------------------------------------------------------------
 * Description   : Configures the Controller Area Network.
 *               : Transfer speed is set as 1 Mbps, and sets mailbox 0.
 *               : Enables mailbox 0 to receive data frames.
 *-------------------------------------------------------------------------------
 * Argument      : void
 *-------------------------------------------------------------------------------
 * Return Value : void
 *-------------------------------------------------------------------------------
 * Note          : None
 *""FUNC COMMENT END""***********************************************************/
void io_init_can(void)
{
  int i,j;

  /* ==== Sets the Standby control register 6 ==== */
  STB.CR6.BYTE = 0x8f;                /* Clears RCAN module standby */

  /* ==== Sets the Master control register ==== */
  RCANET.MCR.WORD = 0x0001;               /* Sets reset mode */
  while((RCANET.IRR.WORD & CAN_IRR0) != CAN_IRR0){
    /* Waits for completing transition to reset mode */
  }
  /* ==== IRR = 1, GSR = 1 (sets automatically) ==== */

  /* ---- Clears the reset interrupt flag ---- */
  RCANET.IRR.WORD = 0x0001;

  /* ---- Sets the Master control register ---- */
  RCANET.MCR.WORD |= 0x8000;              /* RCAN-ET is not same as HCAN2 */

  /* ---- Sets the Interrupt mask register ---- */
  RCANET.IMR.WORD = 0xfffd;

  /* ---- Sets Mailbox interrupt mask register 0 ---- */
  RCANET.MBIMR0.WORD = 0xfffe;
```

3.6 Sample Program List "main.c" (6/6)

```

  /* ----Clears mailbox RAM ---- */
  for(i = 0; i < 16; i++){
    RCANET.MB[i].CTRL0.LONG = 0x00000000;
    RCANET.MB[i].LAFM.LONG = 0x00000000;
    for(j = 0; j < 8; j++){
        RCANET.MB[i].MSG_DATA[j] = 0x00;
    }
  }

  /* ---- Sets mailbox 0 ---- */
  RCANET.MB[0].CTRL1.WORD = 0x0200;       /* MBC = 2, dlc = 0  */
  RCANET.MB[0].CTRL0.LONG = 0x00280000;   /* Standard data frame, id = 0x00a */
  RCANET.MB[0].LAFM.LONG = 0x00000000;
  for(i = 0; i < 8; i++){                 /* Clears data */
    RCANET.MB[0].MSG_DATA[i] = 0x00;
  }

  /* ---- Sets the Bit configuration register ---- */
  RCANET.BCR1.WORD = 0xf700;              /* tsg1 = 15(16-bit), tsg2 = 7(8-bit), */
                                      /* sjw =0 (1-bit), bsp=0 */
  RCANET.BCR0.WORD = 0x0000;              /* 1 Mbps */

  /* ---- Sets the Interrupt request register ---- */
  RCANET.IRR.WORD = 0xffff;

  /* =======Sets Interrupt priority level register 07 (IPR07) ======= */
  INTC.IPR18.BIT._RCAN = 0xa;

  /* ---- Sets the Master control register ---- */
  RCANET.MCR.WORD &= 0xf8fc;              /* Clears MCR0 and MCR1 */
  while( (RCANET.GSR.WORD & 0x0008) != 0x0000 ){
    /* Waits until reset end is end */
  }
}

/* End of File */
```

3.7 Sample Program List "intprg.c"

```
==== preceding information deleted ====


  void INT_RCANET0_RM01_0(void)
  {
    /* ---- Clears the data frame received interrupt flag ---- */
    RCANET.RXPR0.BIT.MB0 = 0x1;

    DTC_REG.CRB = DTC_COUNT;    /* transfer counter     */

    if(Int_Flag_EvenOdd == 0){
      /* ---- DTC destination address register ---- */
      DTC_REG.DAR = (unsigned long)DstData_odd;
      Int_Flag_EvenOdd = 1;
    }
    else{
      /* ---- DTC destination address register ---- */
      DTC_REG.DAR = (unsigned long)DstData_even;
      Int_Flag_EvenOdd = 0;
    }

    DTC.DTCERA.BIT.RM0 = 1;         /* interrupt source RM0 */
  }


  ==== additional information deleted ====
```

3.8 Sample Program List "dtc.h" (1/2)

```
/********************************************************************************
*   DISCLAIMER
*
*   This software is supplied by Renesas Electronics Corp. and is only
*   intended for use with Renesas products.  No other uses are authorized.
*
*   This software is owned by Renesas Electronics Corp. and is protected under
*   all applicable laws, including copyright laws.
*
*   THIS SOFTWARE IS PROVIDED "AS IS" AND RENESAS MAKES NO WARRANTIES
*   REGARDING THIS SOFTWARE, WHETHER EXPRESS, IMPLIED OR STATUTORY,
*   INCLUDING BUT NOT LIMITED TO WARRANTIES OF MERCHANTABILITY, FITNESS FOR A
*   PARTICULAR PURPOSE AND NON-INFRINGEMENT.  ALL SUCH WARRANTIES ARE EXPRESSLY
*   DISCLAIMED.
*
*   TO THE MAXIMUM EXTENT PERMITTED NOT PROHIBITED BY LAW, NEITHER RENESAS
*   ELECTRONICS CORP. NOR ANY OF ITS AFFILIATED COMPANIES SHALL BE LIABLE
*   FOR ANY DIRECT, INDIRECT, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES
*   FOR ANY REASON RELATED TO THIS SOFTWARE, EVEN IF RENESAS OR ITS
*   AFFILIATES HAVE BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.
*
*   Renesas reserves the right, without notice, to make changes to this
*   software and to discontinue the availability of this software.
*   By using this software, you agree to the additional terms and
*   conditions found by accessing the following link:
*   http://www.renesas.com/disclaimer
*********************************************************************************
*   (C) 2010 Renesas Electronics Corporation. All rights reserved.
*""FILE COMMENT""*********** Technical reference data ***************************
*   System Name : SH7216 Sample Program
*   File Name   : dtc.h
*   Abstract    : DMAC+CAN Module Application (Data Frame Receive)
*   Version     : 1.00.00
*   Device      : SH7216
*   Tool-Chain  : High-performance Embedded Workshop (Ver.4.07.00).
*               : C/C++ compiler package for the SuperH RISC engine family
*               :                             (Ver.9.03 Release00).
*   OS          : None
*   H/W Platform: R0K572167 (CPU board)
*   Description :
*********************************************************************************
*   History     : Apr.28,2010 Ver.1.00.00
*""FILE COMMENT END""**********************************************************/

```

3.9 Sample Program List "dtc.h" (2/2)

```
/* ---- structure definition ---- */
struct st_dtc_info{
  unsigned char MRA;              /* DTC mode register A                 */
  unsigned char MRB;              /* DTC mode register B                 */
  unsigned char dummy1;           /* reserved                            */
  unsigned char dummy2;           /* reserved                            */
  unsigned long SAR;              /* DTC source address register         */
  unsigned long DAR;              /* DTC destination address register    */
  union{
    unsigned short WORD;
    struct {
        unsigned char H;
        unsigned char L;
    } BYTE;
  } CRA;                          /* DTC transfer count register A       */
  unsigned short CRB;             /* DTC transfer count register B       */
};

/* ---- symbol definition ---- */
#define CAN_IRR0   0x0001
#define DTC_COUNT  5              /* DTC transfer count      */
#define DTC_BLOCK_LENG 0x09       /* DTC block size          */

#define DTC_REG(*(volatile struct st_dtc_info*)0xfff88800)  /* DTC transfer information */
#define DTC_VECT_BASE  0xfff88000 /* DTC vector base address */

/* End of File */
```

4. References

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

- Hardware Manual
  SH7216 Group Hardware Manual Rev. 1.01
  The latest version of the hardware manual can be downloaded from the Renesas Electronics website.

Website and Support

Renesas Electronics Website

http://www.renesas.com/

Inquiries

http://www.renesas.com/inquiry

Revision Record

		Description	
Rev.	Date	Page	Summary
1.00	Jun.04.10	—	First edition issued

General Precautions in the Handling of MPU/MCU Products

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

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

Notice

1. All information included in this document is current as of the date this document is issued. Such information, however, is subject to change without any prior notice. Before purchasing or using any Renesas Electronics products listed herein, please confirm the latest product information with a Renesas Electronics sales office. Also, please pay regular and careful attention to additional and different information to be disclosed by Renesas Electronics such as that disclosed through our website.
2. Renesas Electronics does not assume any liability for infringement of patents, copyrights, or other intellectual property rights of third parties by or arising from the use of Renesas Electronics products or technical information described in this document. No license, express, implied or otherwise, is granted hereby under any patents, copyrights or other intellectual property rights of Renesas Electronics or others.
3. You should not alter, modify, copy, or otherwise misappropriate any Renesas Electronics product, whether in whole or in part.
4. Descriptions of circuits, software and other related information in this document are provided only to illustrate the operation of semiconductor products and application examples. You are fully responsible for the incorporation of these circuits, software, and information in the design of your equipment. Renesas Electronics assumes no responsibility for any losses incurred by you or third parties arising from the use of these circuits, software, or information.
5. When exporting the products or technology described in this document, you should comply with the applicable export control laws and regulations and follow the procedures required by such laws and regulations. You should not use Renesas Electronics products or the technology described in this document for any purpose relating to military applications or use by the military, including but not limited to the development of weapons of mass destruction. Renesas Electronics products and technology may not be used for or incorporated into any products or systems whose manufacture, use, or sale is prohibited under any applicable domestic or foreign laws or regulations.
6. Renesas Electronics has used reasonable care in preparing the information included in this document, but Renesas Electronics does not warrant that such information is error free. Renesas Electronics assumes no liability whatsoever for any damages incurred by you resulting from errors in or omissions from the information included herein.
7. Renesas Electronics products are classified according to the following three quality grades: "Standard", "High Quality", and "Specific". The recommended applications for each Renesas Electronics product depends on the product's quality grade, as indicated below. You must check the quality grade of each Renesas Electronics product before using it in a particular application. You may not use any Renesas Electronics product for any application categorized as "Specific" without the prior written consent of Renesas Electronics. Further, you may not use any Renesas Electronics product for any application for which it is not intended without the prior written consent of Renesas Electronics. Renesas Electronics shall not be in any way liable for any damages or losses incurred by you or third parties arising from the use of any Renesas Electronics product for an application categorized as "Specific" or for which the product is not intended where you have failed to obtain the prior written consent of Renesas Electronics. The quality grade of each Renesas Electronics product is "Standard" unless otherwise expressly specified in a Renesas Electronics data sheets or data books, etc.
   "Standard": Computers; office equipment; communications equipment; test and measurement equipment; audio and visual equipment; home electronic appliances; machine tools; personal electronic equipment; and industrial robots.
   "High Quality": Transportation equipment (automobiles, trains, ships, etc.); traffic control systems; anti-disaster systems; anti-crime systems; safety equipment; and medical equipment not specifically designed for life support.
   "Specific": Aircraft; aerospace equipment; submersible repeaters; nuclear reactor control systems; medical equipment or systems for life support (e.g. artificial life support devices or systems), surgical implantations, or healthcare intervention (e.g. excision, etc.), and any other applications or purposes that pose a direct threat to human life.
8. You should use the Renesas Electronics products described in this document within the range specified by Renesas Electronics, especially with respect to the maximum rating, operating supply voltage range, movement power voltage range, heat radiation characteristics, installation and other product characteristics. Renesas Electronics shall have no liability for malfunctions or damages arising out of the use of Renesas Electronics products beyond such specified ranges.
9. Although Renesas Electronics endeavors to improve the quality and reliability of its products, semiconductor products have specific characteristics such as the occurrence of failure at a certain rate and malfunctions under certain use conditions. Further, Renesas Electronics products are not subject to radiation resistance design. Please be sure to implement safety measures to guard them against the possibility of physical injury, and injury or damage caused by fire in the event of the failure of a Renesas Electronics product, such as safety design for hardware and software including but not limited to redundancy, fire control and malfunction prevention, appropriate treatment for aging degradation or any other appropriate measures. Because the evaluation of microcomputer software alone is very difficult, please evaluate the safety of the final products or system manufactured by you.
10. Please contact a Renesas Electronics sales office for details as to environmental matters such as the environmental compatibility of each Renesas Electronics product. Please use Renesas Electronics products in compliance with all applicable laws and regulations that regulate the inclusion or use of controlled substances, including without limitation, the EU RoHS Directive. Renesas Electronics assumes no liability for damages or losses occurring as a result of your noncompliance with applicable laws and regulations.
11. This document may not be reproduced or duplicated, in any form, in whole or in part, without prior written consent of Renesas Electronics.
12. Please contact a Renesas Electronics sales office if you have any questions regarding the information contained in this document or Renesas Electronics products, or if you have any other inquiries.

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

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

Renesas Electronics Corporation

http://www.renesas.com

SALES OFFICES

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

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

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

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

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

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

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

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

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

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

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

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

Colophon 1.0