SH7266/SH7267 Group

Boot from the Serial Flash Memory with Multiple I/Os

R01AN0653EJ0100
Rev. 1.00
Sep. 27, 2011

Summary

This application note describes how to boot the SH7266/SH7267 Microcomputers (MCUs) from the on-chip serial flash memory that is applicable to multiple I/Os.

Target Device

SH7267

Contents

1. Introduction

1.1 Specifications

The SH7267 boots from the serial flash memory in boot mode 1 or boot mode 3 (hereinafter called serial flash boot). This application note describes the loader program and the application program examples when using the serial flash boot in high speed. The boot time is shortened by using multiple I/O for transferring the application programs.

This application note also describes the downloader to write the loader program and the application program to serial flash memory.

1.2 Modules Used

- Boot mode (serial flash boot)
- Renesas Serial Peripheral Interface (RSPI)
- Renesas Quad Serial Peripheral Interface (RQSPI)

1.3 Applicable Conditions

MCU	SH7266/SH7267
Operating Frequency	Internal clock: 144MHz Bus clock: 72MHz Peripheral clock: 36MHz
Integrated Development Environment	Renesas Electronics Corporation High-performance Embedded Workshop Ver.4.07.00
C Compiler	Renesas Electronics SuperH RISC engine Family C/C++ compiler package Ver.9.03 Release 02
Compiler Options	Default setting in the High-performance Embedded Workshop (-cpu=sh2afpu -fpu=single -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)
Serial Flash Memory	S25FL032P (Spansion)

1.4 Related Application Notes

For more information, refer to the following application notes:

- SH7266/SH7267 Group Interfacing Serial Flash Memory Using the Renesas Quad Serial Peripheral Interface
- SH7266/SH7267 Groups Renesas Quad Serial Peripheral Interface High Speed Read Processing on Serial Flash Memory
- SH7266/SH7267 Group Boot From the Serial Flash Memory

1.5 About Active-low Pins (Signals)

The symbol "#" suffixed to the pin (or signal) names indicates that the pins (or signals) are active-low.

2. Overview of Serial Flash Boot Function

This chapter describes an overview of the serial flash boot function.

2.1 Glossary of Terms

Table 1 lists the terms used in this application note to describe the serial flash boot.

Table 1 Glossary

Item	Description
On-chip ROM program to boot	Transfers the loader program stored in the beginning of the serial flash memory to the high-speed on-chip RAM, and jumps to the loader program when the MCU is booted in boot mode 1 or 3. As this program is already stored in the on-chip ROM to boot in CPU, user is not required to create it.
Loader program	Transfers the application program from the serial flash memory to the on-chip RAM, and jumps to the entry function in the application program. The size of the loader program is fixed to 8KB. Create it according to the user's system.
Application program	Is created by user according to the system
Downloader	To write the loader program and application program to the serial flash memory. Create it according to the user's system.

2.2 Serial Flash Boot Operation

Table 2 lists the external pins (MD_BOOT1 to MD_BOOT0) to decide the boot mode.

Table 2 Relationship between the External Pin Setting and Serial Flash Boot Mode

MD_BOOT1	MD_BOOT0	Boot Mode	Description
0	1	Boot mode 1	Boots the MCU from serial flash memory connected to Renesas Serial Peripheral Interface channel 0 in low-speed communication. Low-speed communication: 1/4 of the bus clock speed (Bφ)
1	1	Boot mode 3	Boots the MCU from serial flash memory connected to Renesas Serial Peripheral Interface channel 0 in high-speed communication. High-speed communication: 1/2 of the bus clock speed (Bφ)

When in boot mode 1 or 3, the on-chip ROM program for booting transfers the loader program from the serial flash memory connected to the Renesas Serial Peripheral Interface channel 0 (RSPI0) to the high-speed on-chip RAM after the power-on reset is canceled. After transferring, the program jumps to the head of the loader program. Figure 1 shows the operation image of the on-chip ROM program for booting. These processing is automatically performed.

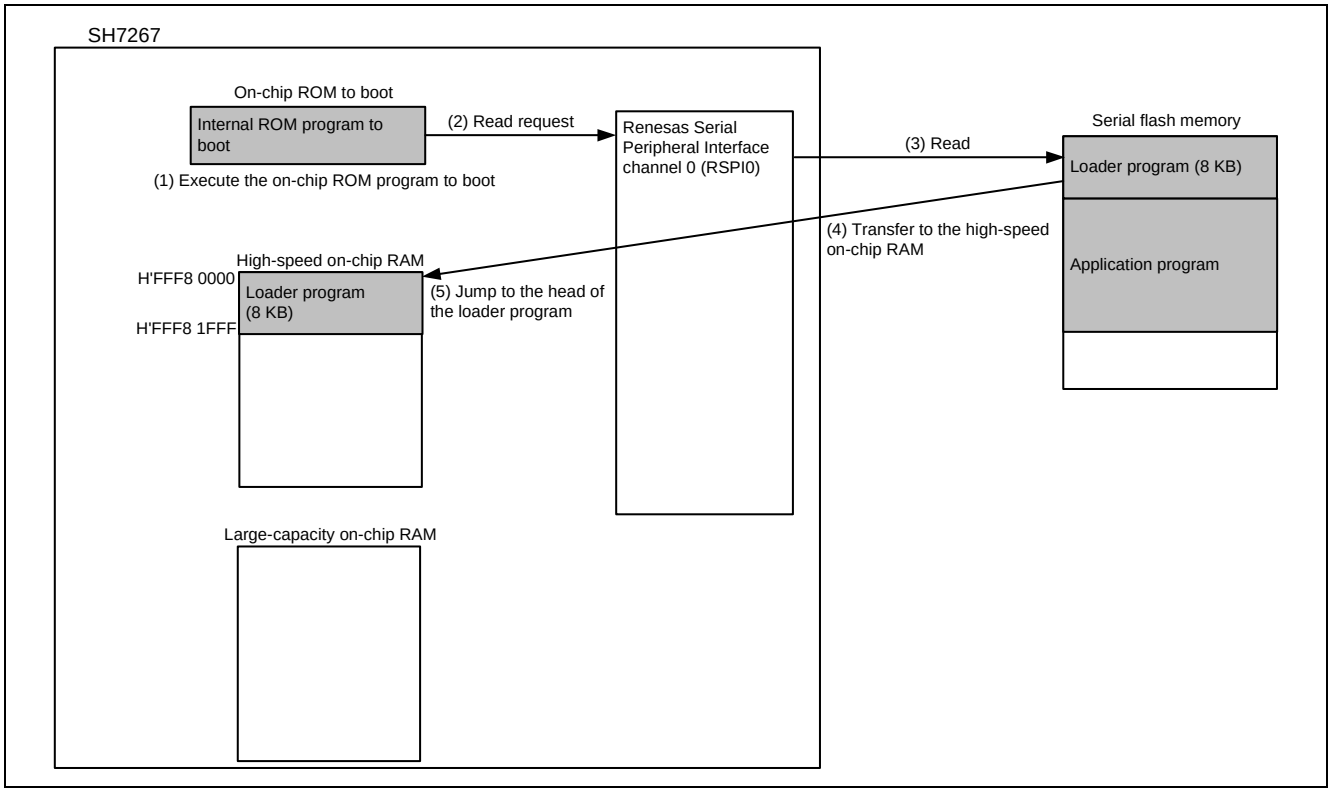

Figure 1 Operation Image of the on-chip ROM Program for Booting

The loader program transfers the application program to the large capacity on-chip RAM. In this application, The Renesas Quad Serial Peripheral Interface (RQSPI) is used for the purpose of saving the transfer time. Figure 2 shows the operation image in the loader program.

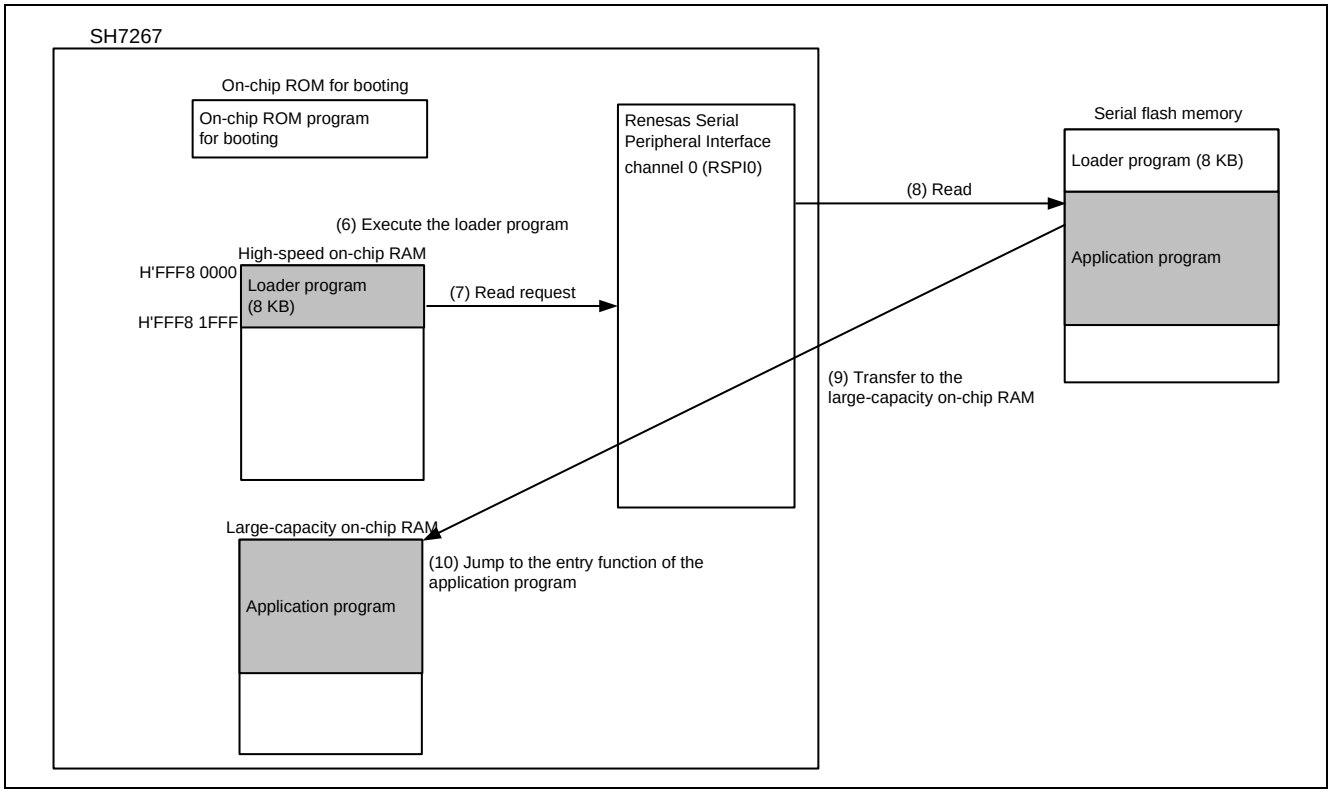

Figure 2 Operation Image of the Loader Program

Note: Application program can be transferred to an external RAM such as SDRAM by modifying the loader program.

2.3 Downloader Operation

The downloader writes the loader program on the high-speed on-chip RAM and the application program on the RAM to the serial flash memory. Figure 3 shows the operation image of the downloader.

For more information, refer to "3.3 Downloader".

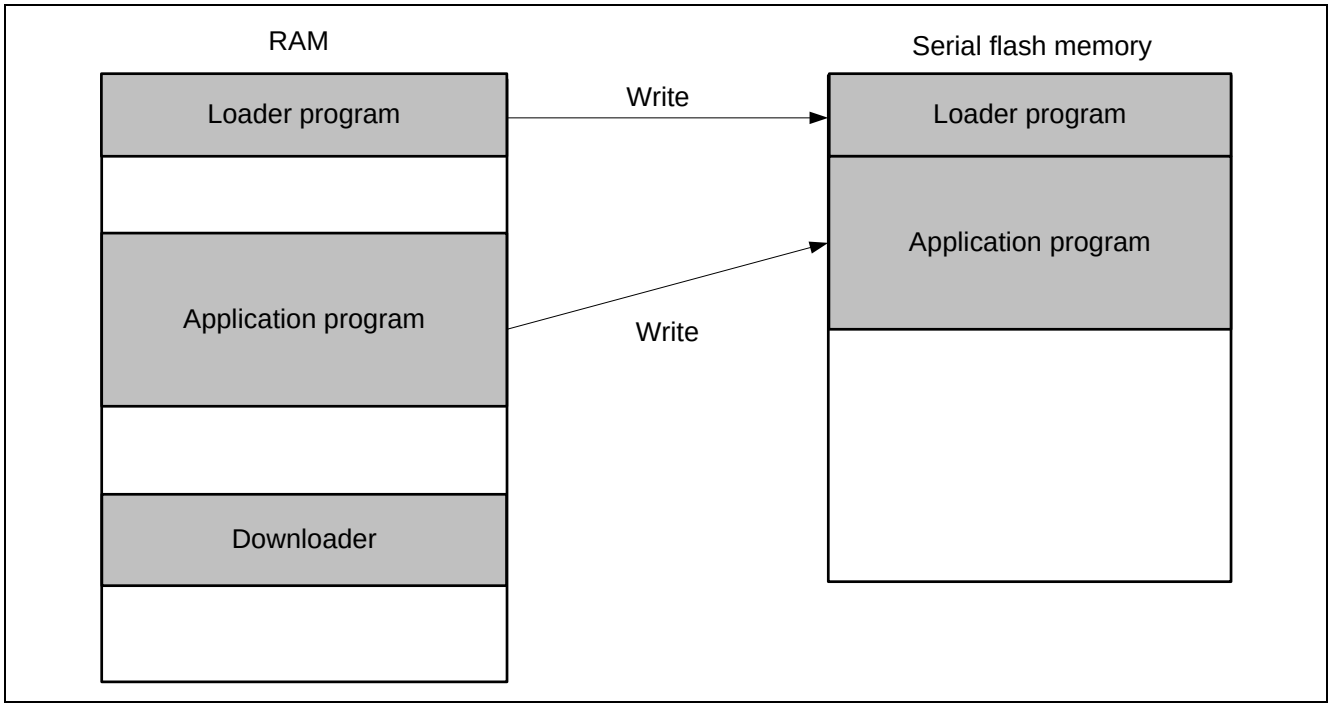

Figure 3 Operation Image of the Downloader

2.4 Serial Flash Memory Connection

Figure 4 shows an example of a connection circuit when using the serial flash boot function. When using the serial flash boot function, the serial flash memory in the SPI interface should be connected to the Renesas serial peripheral interface channel 0 (RSPI0). This application uses the Renesas Quad Serial Peripheral Interface (RQSPI) to transfer the application program. Therefore, the serial flash memory with multiple I/Os should be connected as shown in Figure 4.

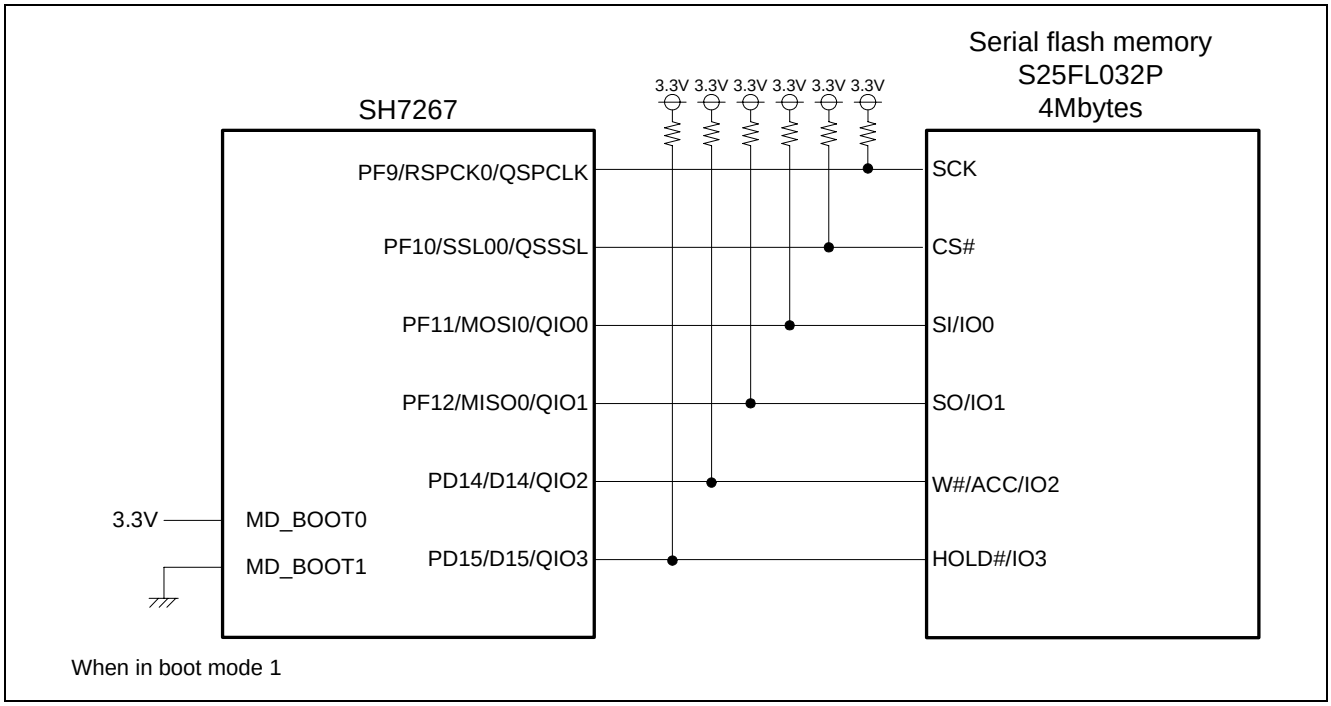

Figure 4 Serial Flash Memory Circuit

Note: The SH7266/SH7267 uses the RSPI clock at 1/4 of the bus clock rate (Bφ) in boot mode 1, and at 1/2 of the bus clock rate in boot mode 3. Select the boot mode to satisfy the AC characteristics of the serial flash memory and the RSPI.

3. Applications

This chapter describes the loader program, the application program, and the downloader.

3.1 Loader Program Specifications

The loader program transfers the application program from the serial flash memory to the large-capacity on-chip RAM, and jumps to the entry function of the application program.

3.1.1 Memory Map

Figure 5 shows the memory map of the loader program.

1. The loader program (the program area) is allocated to the address from H'FFF8 0000 to H'FFF8 1AFF.
2. The tentative exception handling vector table is allocated to the address from H'FFF8 1B00 to H'FFF8 1B4F (For more information, refer to 3.1.5).
3. The loader program stack area is allocated to the address from H'FFF8 1C00 to H'FFF8 1FFF (For more information, refer to 3.1.3).

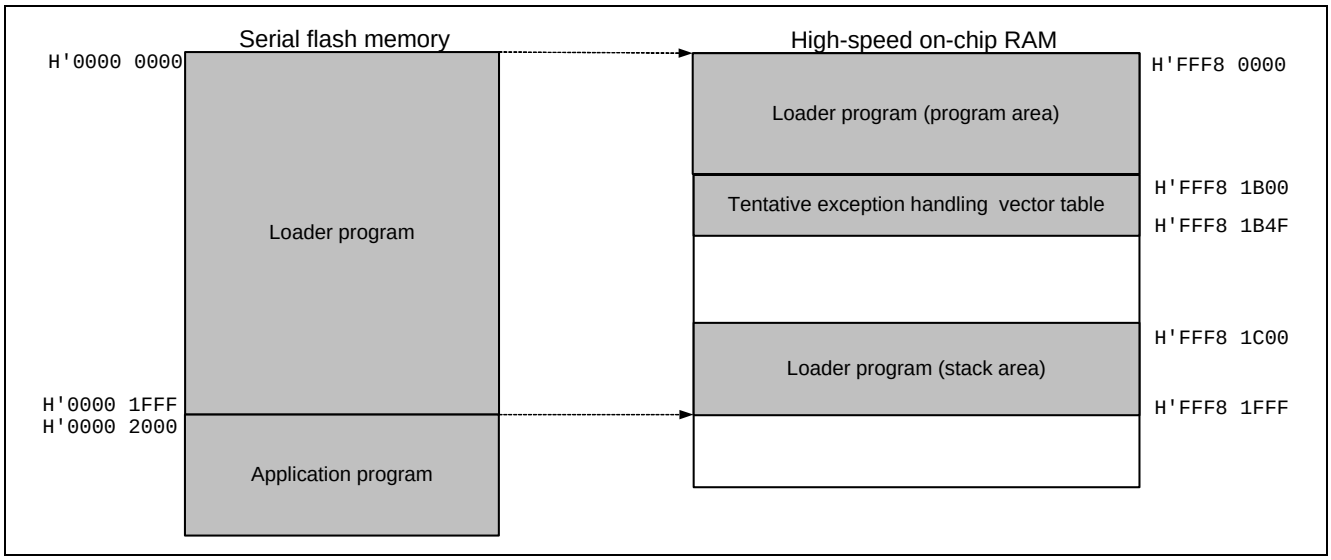

Figure 5 Loader Program Memory Map

3.1.2 Loader Program Flow Chart

Figure 6 shows the flow chart of the loader program. For more information, refer to sections 3.1.3 to 3.1.11.

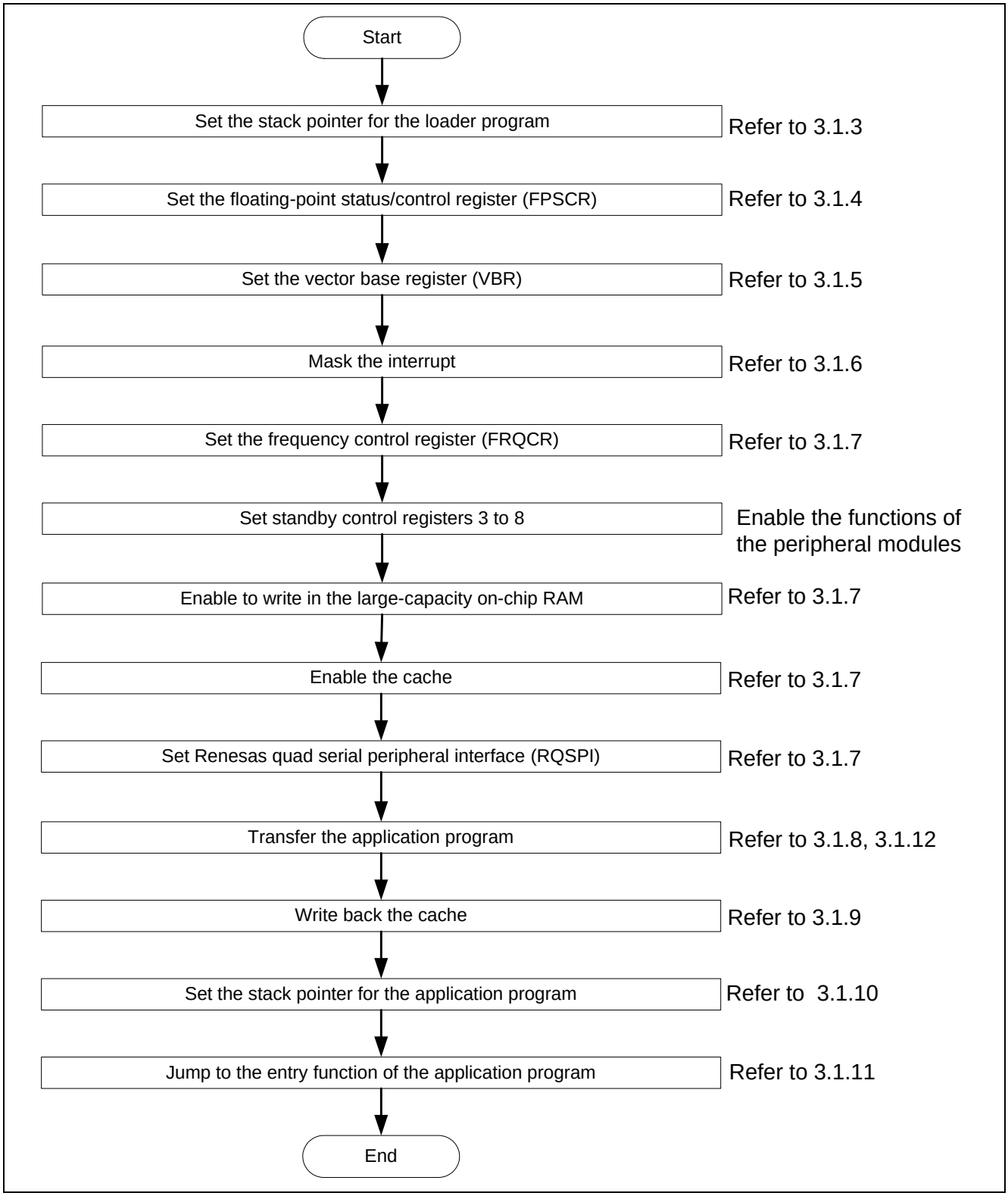

Figure 6 Loader Program Flow Chart

3.1.3 Stack Pointer Setting

Set the stack pointer (R15) to the address H'FFF8 2000. Allocate the loader program processing at the address H'FFF8 0000, and use the assembly language to avoid the loader program using the undefined stack pointer. Then, the loader program jumps to the entry function of the loader program. Once the stack pointer is configured, C can be used.

3.1.4 Floating-Point Status Setting/Control Register (FPSCR) Setting

Specify the FPSCR at the address H'0004 0001 (single-precision operation, round to zero).

3.1.5 Vector Base Register (VBR) Setting

The loader program sets the tentative exception handling vector table in VBR to support the exception handling the loader program is operating. Do not generate exceptions or interrupts before setting the VBR, as the exception handling vector table is undefined. As the loader program does not use interrupts, only vector numbers 0 to 18 are defined in the tentative exception handling vector table. To embed the exception handling such as the external interrupt during the loader program is operating, extend the tentative exception handling vector table.

Note: Store the exception handling vector table in the flash memory to allow the CPU to access the flash memory before executing exception handling. For more information, refer to the section 6.9.4 "Note before Exception Handling Begins Running" in the SH7266 Group, SH7267 Group User's Manual: Hardware.

3.1.6 Interrupt Mask

Specify B'1111 in the interrupt mask level bit in the status register (SR) as the loader program does not support interrupts during it is operating.

3.1.7 Configuration

Configure the peripheral functions to read the application program from the serial flash memory.

3.1.8 Application Program Transfer

The loader program refers to the application program transfer information (appinfo) in the serial flash memory, and transfers the application program to the large-capacity on-chip RAM. Table 3 lists the application program transfer information (appinfo) in detail. Allocate the application program transfer information (appinfo) at the address H'0000 2000 in the serial flash memory. The loader program handles the information at H'0000 2000 to H'0000 2007 in the serial flash memory as the application program transfer information.

Table 3 Application Program Transfer Information (appinfo)

Item	Address	Size
Destination start address	H'0000 2000	4
Destination end address	H'0000 2004	4

Figure 7 shows the transfer image using the application program transfer information. Refer to3.2.7 for the procedures to generate the application program transfer information,

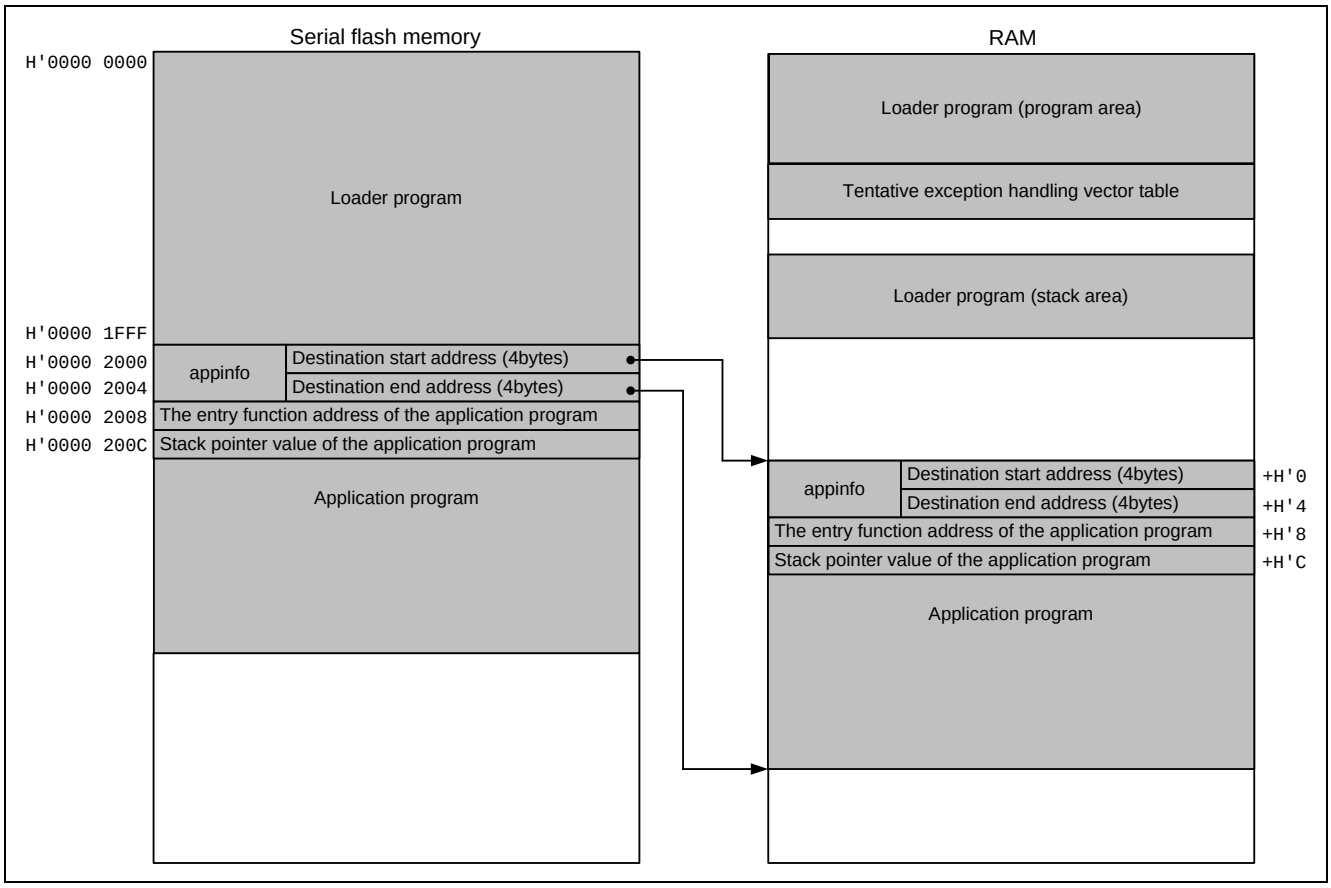

Figure 7 Application Program Transfer Image

3.1.9 Cache Write Back Processing

After transferring the application program to the large-capacity on-chip RAM, the loader program writes back the cache to guarantee the coherency between the cache and the large-capacity on-chip RAM.

3.1.10 Application Program Stack Pointer Setting

The loader program specifies the value stored in the first 12 to 15bytes in the application program in the stack pointer (R15).

3.1.11 Application Program Jump to the Entry Function Address

The loader program jumps to the entry function address stored in the first 8 to 11bytes in the application program.

3.1.12 Serial Flash Memory Commands

A set of commands are used to access the serial flash memory. The loader program uses the Quad Output Read command in the serial flash memory to read the application program from the serial flash memory, and transfers the program to the large-capacity on-chip RAM. Table 4 lists the serial flash memory command used in the loader program.

Table 4 Serial Flash Memory Command

Command Name	Opcode	Function
Quad Output Read	H'6B	Reads the quad data

Note: This application refers to the commands of the SPNSION S25FL032P. The serial flash memory command specifications vary according to the manufacturers. Refer to the datasheet provided by the serial flash memory manufacturer.

3.1.13 Register Status After Executing the Loader Program

Table 5 lists the register status after executing the loader program. Registers excluded in the tables are set as default value given in the SH7266 Group, SH7267 Group User's Manual: Hardware.

Table 5 Register Status (1/2)

Register Name	Abbreviation	Value	Remarks
General registers	R0 to R14	Undefined	
Program counter	PC	Depends on the setting	Application program entry function address
Stack pointer	SP (R15)	Depends on the setting	Stack pointer value in the application program
Status register	SR	Undefined	IMASK bit is B'1111
Vector base register	VBR	H'FFF8 1B00	
Floating-point status/control register	FPSCR	H'0004 0001	Single precision operation, round to 0
Frequency control register	FRQCR	H'1103	
Standby control register 3	STBCR3	H'02	
Standby control register 4	STBCR4	H'00	
Standby control register 5	STBCR5	H'10	
Standby control register 6	STBCR6	H'00	
Standby control register 7	STBCR7	H'2A	
Standby control register 8	STBCR8	H'7A	
System control register 5	SYSCR5	H'0F	Write-enable to the large-capacity on-chip RAM
Cache control register1	CCR1	H'0000 0101	Instruction cache and operand cache are enabled
DMA source address register 4	SAR_4	H'FFFFE004	
DMA destination address register 40	DAR_4	H'1C001EE0	
DMA channel control register 40	CHCR_4	H'02084818	
DMA operation register	DMAOR	H'0001	
DMA expansion resource selector 2	DMARS2	H'00A2	
Port F control register 3	PFCR3	H'0006	pin QMI/QIO1
Port F control register 2	PFCR2	H'6660	pin QMO/QIO0, QSSL, QSPCLK
Port D control register 3	PDCR3	H'3300	pin QIO2, QIO3
Control register	SPCR	H'08	
Pin control register	SPPCR	H'26	
Status register	SPSR	H'60	
Sequence control register	SPSCR	H'02	
Slave select negation delay register	SSLND	H'01	
Succeeding access delay register	SPND	H'01	
Command register 0	SPCMD0	H'E287	

Table 6 Register Status (2/2)

Register Name	Abbreviation	Value	Remarks
Command register 1	SPCMD1	H'E2D1	
Command register 2	SPCMD2	H'E251	
Command register 3	SPCMD3	H'79C8	
Buffer control register	SPBFCR	H'25	
Transfer data length multiplier setting register 0	SPBMUL0	H'01	
Transfer data length multiplier setting register 1	SPBMUL1	H'01	
Transfer data length multiplier setting register 2	SPBMUL2	Undefined	
Transfer data length multiplier setting register 3	SPBMUL3	Undefined	

3.2 Application Program Example

As the loader program transfers the application program from the serial flash memory to the large-capacity on-chip RAM, the memory map of the application program must be allocated to allow the loader program to read. Also, the application program must include the address information to be referred to by the loader program.

This section describes how to create the application program unique to the serial flash boot.

3.2.1 Section Alignment

This section describes the section alignment in the application program.

1. As an application program is executed on the RAM, sections of the application program are located on the large-capacity on-chip RAM in this example.

2. As the loader program uses the start address and end address of the application program to transfer the application program from the serial flash memory to the large-capacity on-chip RAM, allocate the program area, the constant area and the initialized data area of the application program to the physically contiguous areas. Uninitialized data area and stack area can be allocated at a desired address.

3. Allocate the application program transfer information (appinfo), the application program entry function address, and the stack pointer value at fixed address. Place the application program transfer information (appinfo) in the section DAPPINFO, and the application program entry function address and the stack pointer value in he section DVECTTBL. Allocate the section DAPPINFO at the start on the RAM, and then allocate the section DVECTTBL next to it.

4. As the loader program uses the address from H'FFF8 0000 to H'FF8 1FFF on the high-speed on-chip RAM, do not allocate the program area, the constant area, and the initialized data area of the application program at that address.

5. Allocate the reset vector table RESET_Vectors in the start address of section DVECTTBL.

6. As the cache operation program is executed on the cache-disabled space, re-allocate section PCACHE to section RPCACHE on the high-speed on-chip RAM, and execute the section.

Figure 8 shows an example of the section alignment on the RAM.

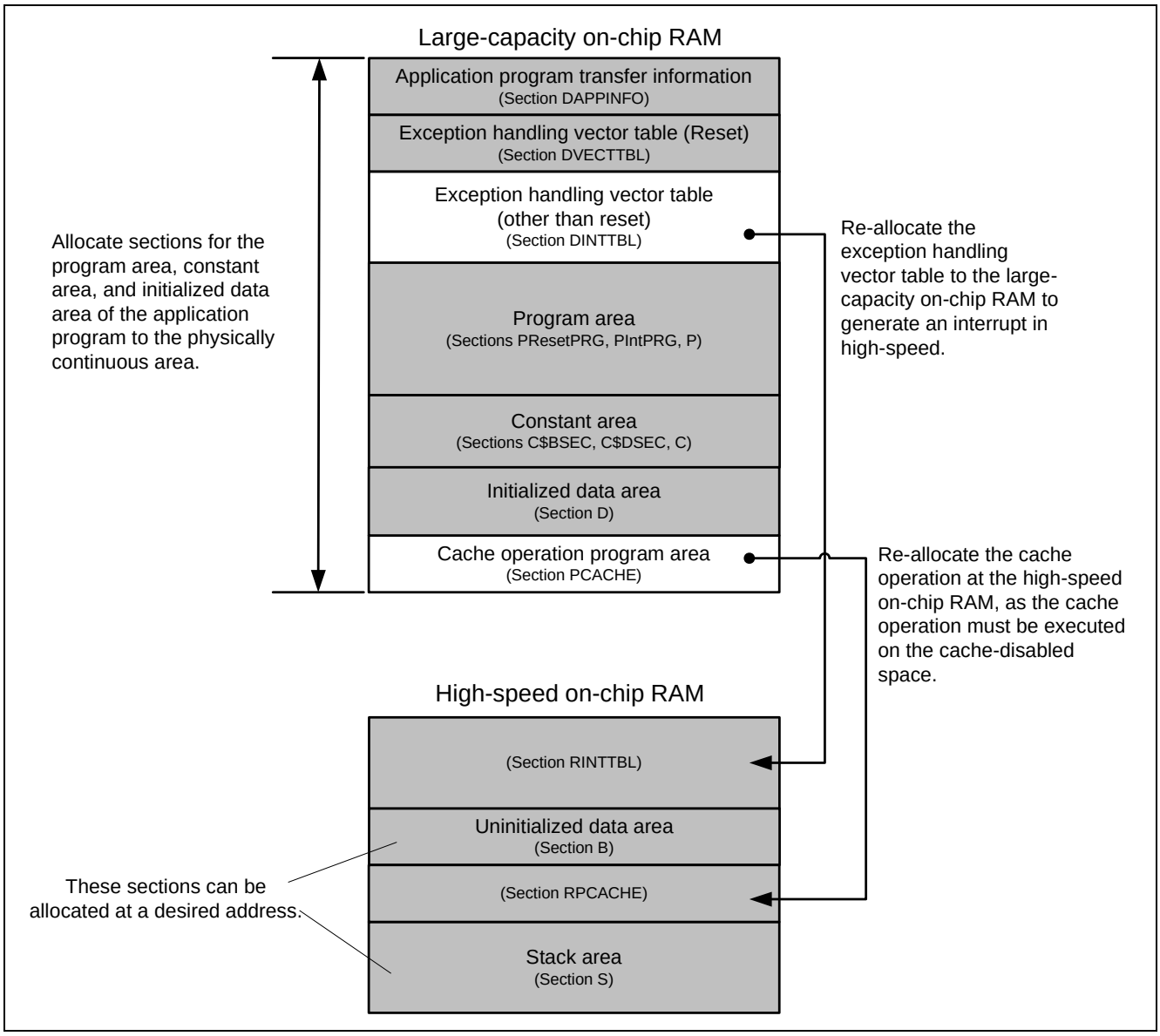

Figure 8 Application Program Section Alignment

3.2.2 Flow Chart

The application program in this application transmits the strings of characters to channel 0 of the Serial Communication Interface with FIFO (SCIF0). Figure 9 shows the flow chart of the application program.

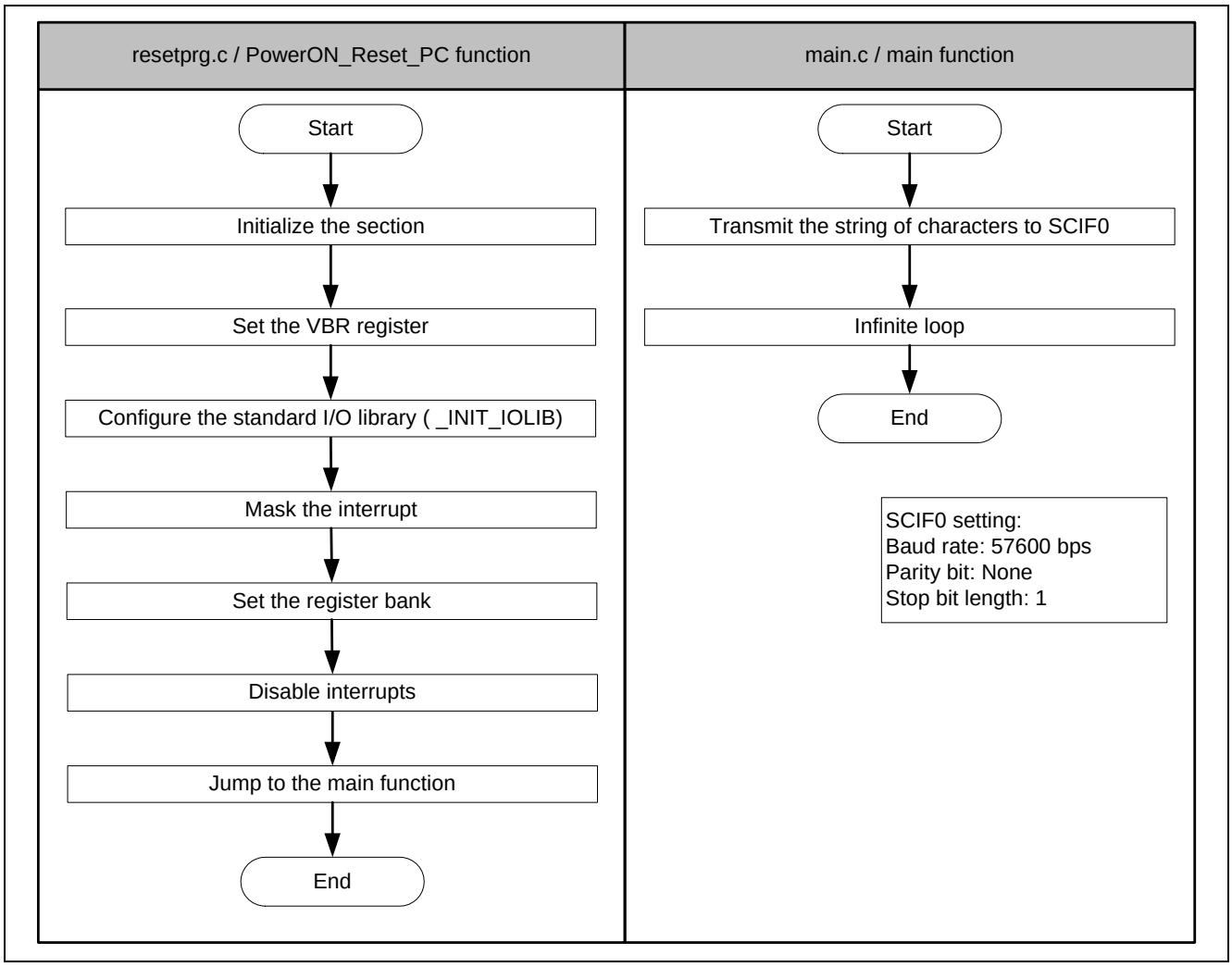

Figure 9 Application Program Flow Chart

3.2.3 Entry Function Setting

Set the entry function address of the application program to table number 0 of the reset vector table RESET_Vectors. Table 7 shows the address setting.

Table 7 Entry Function Address Setting

Item	Description
File Name	vecttbl.c
Name of the section to place	DVECTTBL
Table name	RESET_Vectors
Table number	0
Default	PowerON_Reset_PC

Note: PowerON_Reset_PC is an entry function in the application program.

3.2.4 Stack Pointer Setting

Set the stack pointer of the application program to table number 1 of the reset vector table RESET_Vectors. Table 8 shows the stack pointer setting value.

Table 8 Stack Pointer Setting

Item	Description
File name	vecttbl.c
Name of the section to place	DVECTTBL
Table name	RESET_Vectors
Table number	1
Default	_secend ("S")

3.2.5 Initializing the Section

Initialize the section by executing the section initialization routine (_INITSCT function). To execute the _INITSCT function, use values stored in section initialization tables (DTBL and BTBL) described in the file dbsct.c. After executing the _INITSCT function, write back the cache to guarantee the coherency between the cache and the large-capacity on-chip RAM.

3.2.6 Vector Base Register (VBR) Setting

Set the exception handling vector table of the application program to the VBR.

3.2.7 Generating the Application Program Transfer Information (appinfo)

Table 9 shows the structure to generate the application program transfer information (appinfo). Retrieve the start and end address of the application program by section address operators (_sectop, _secend). Allocate the following structure in section DAPPINFO. Register the start address of the application program (program area, constant area, and initialized data area) in the app_top, and the end address of the application program in the app_end.

Table 9 Application Program Transfer Information (appinfo)

Item	Description		
File name	appinfo.c		
Structure name	appinfo		
Structure member	**Member Name**	**Value**	**Description**
	void *app_top	_sectop ("DAPPINFO")	Start address of the application program
	Void *app_end	_secend ("PCACHE")	End address of the application program +1
Name of the section to place	DAPPINFO		

Note: The amount of the size of the loader program (8 KB) and application program must not exceed the capacity of the serial flash memory.

Figure 10 shows the procedure to generate the application program transfer information (appinfo).

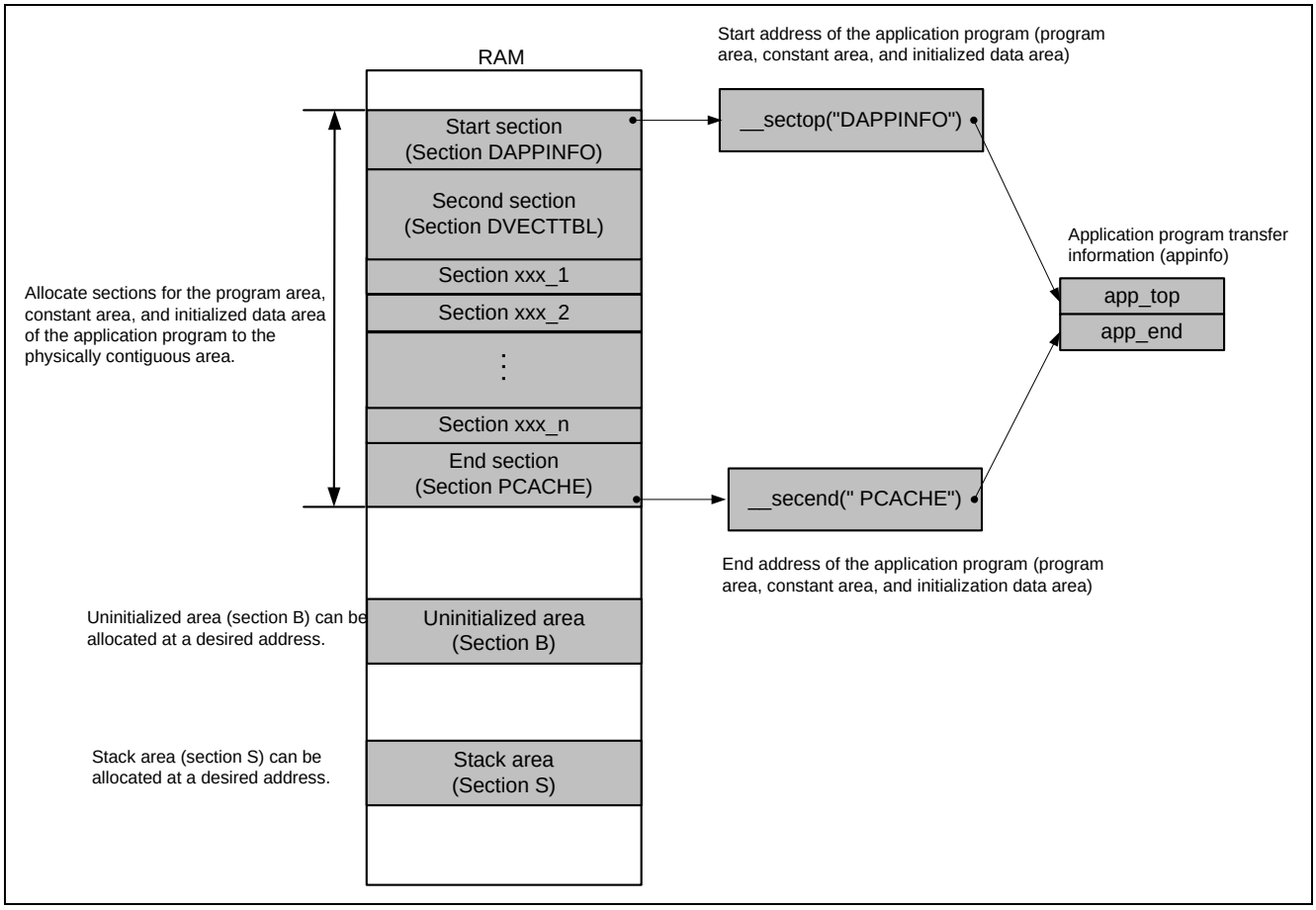

Figure 10 Application Program Transfer Information Generated Image

3.3 Downloader

This section describes the downloader in this application.

3.3.1 Operation Overview

Before executing the downloader, transfer the downloader and the loader program from the development environment to the high-speed internal RAM on the system by the debugger, and the application program to the large-capacity on-chip RAM. Figure 11 shows an operation image of the downloader.

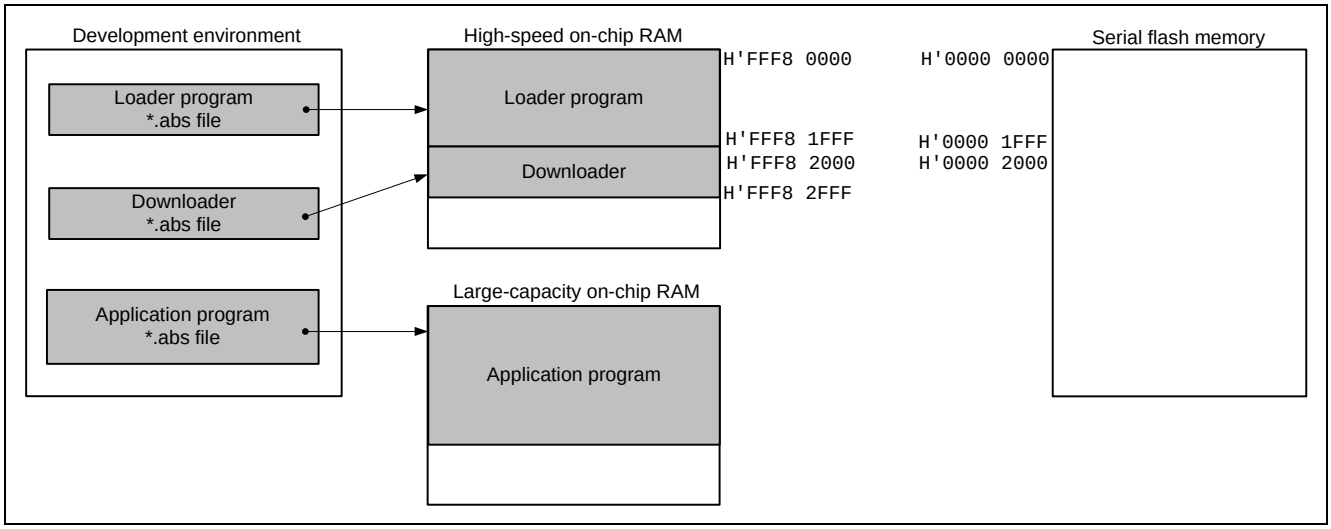

Figure 11 Downloader Operation Image (1)

Execute the downloader to write the loader program and the application program in the serial flash memory. The downloader writes the loader program in the address at H'0000 0000 to H'0000 1FFF, and the application program at H'0000 2000 or later in the serial flash memory. Figure 12 shows an operation image of the downloader.

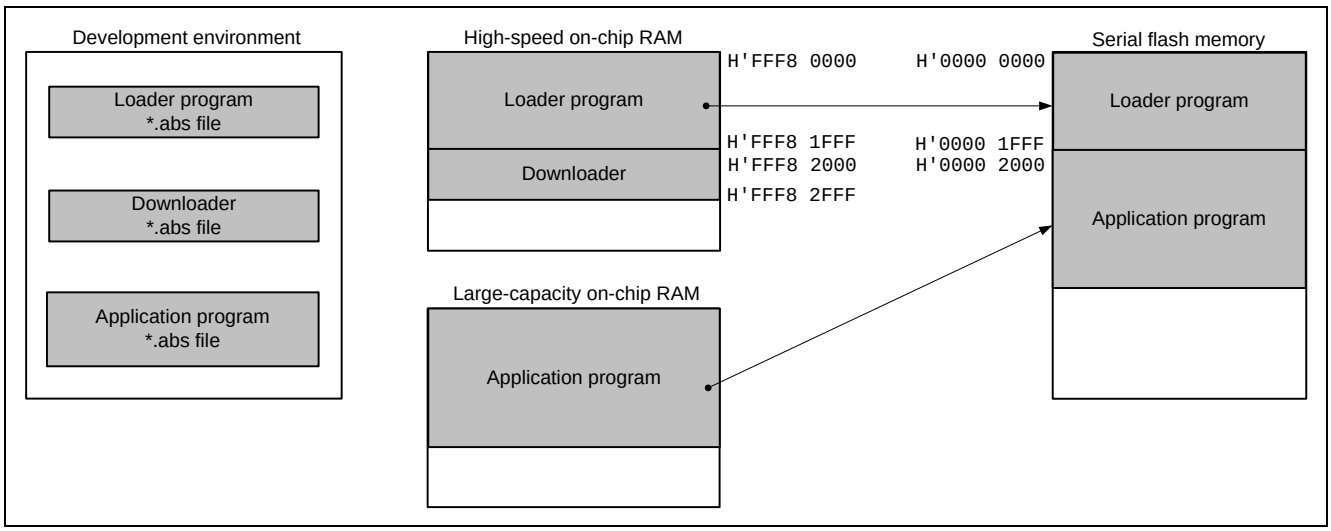

Figure 12 Downloader Operation Image (2)

3.3.2 Area Used by the Downloader

The downloader occupies the address from H'FFF8 2000 to H'FFF8 2FFF. When the loader program, application program and downloader compete on the same section, programs do not operate properly.

3.3.3 Flow Chart

Figure 13 shows the flow chart of the downloader. Executing the downloader placed on the high-speed internal RAM enables to write the loader program and application program in the serial flash memory. For more information, refer to sections 3.3.4 to 3.3.8.

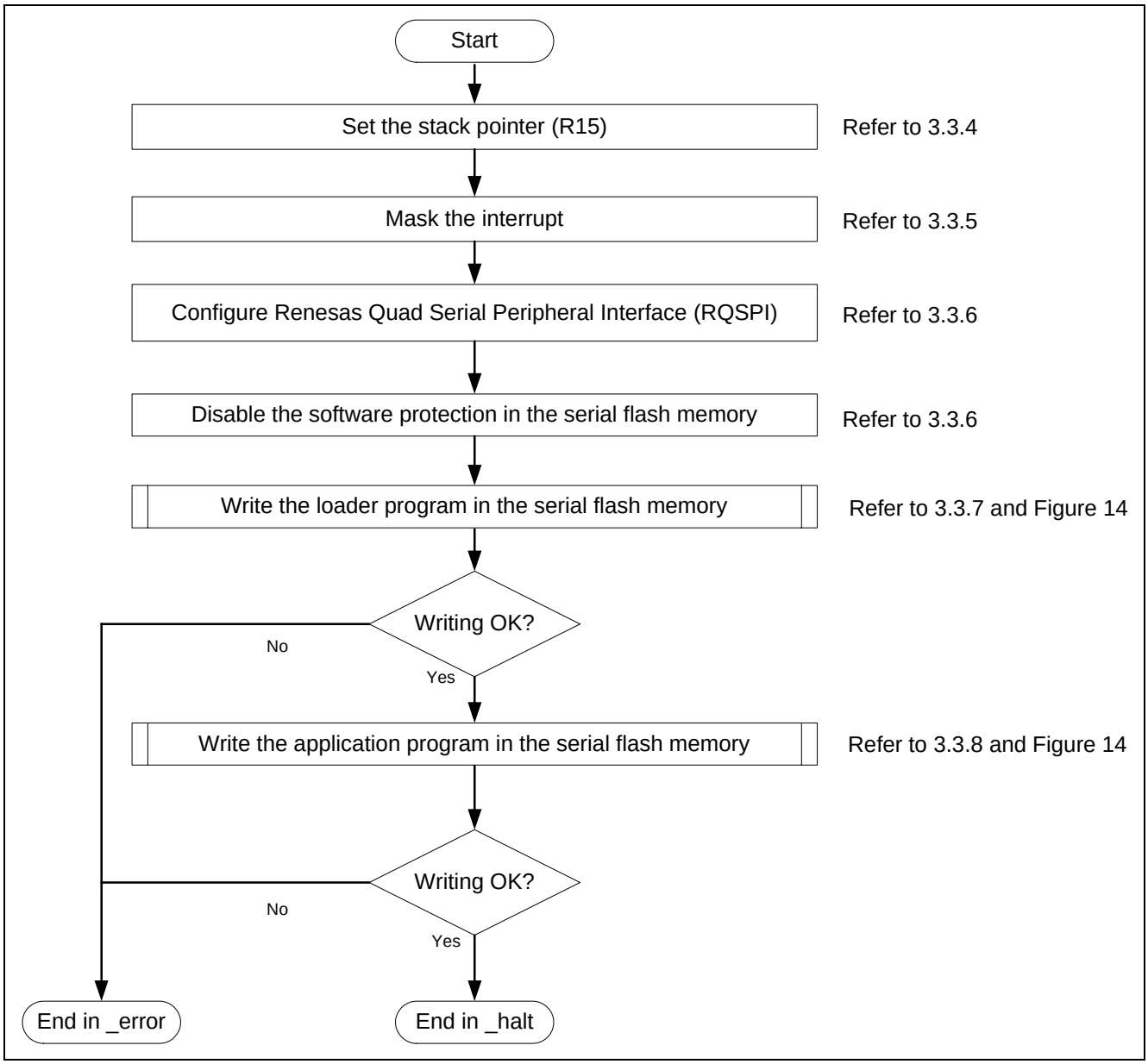

Figure 13 Downloader Flow Chart

Figure 14 shows the flow chart of writing the loader program and the application program.

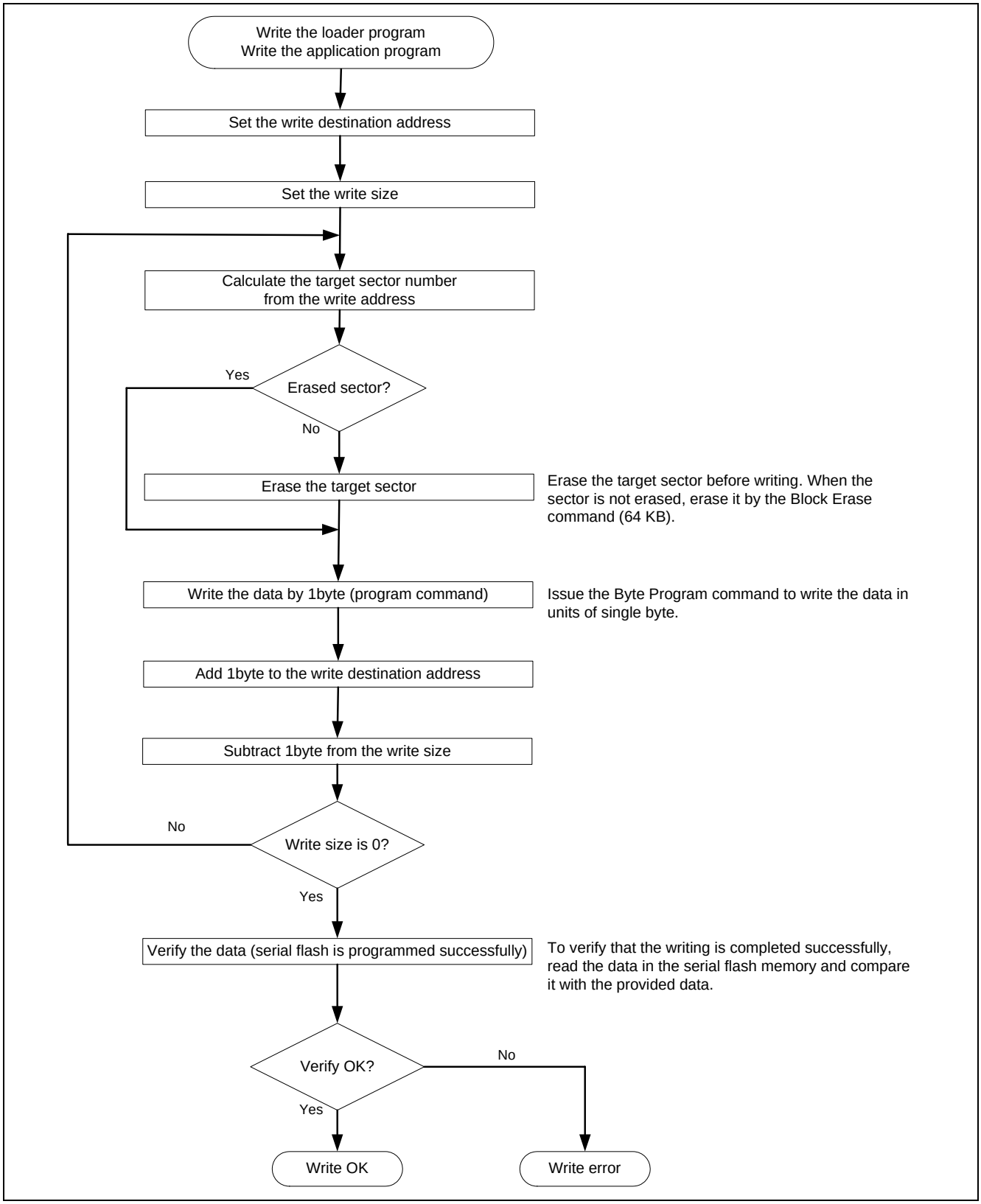

Figure 14 Flow Chart of Writing

3.3.4 Stack Pointer Setting

Specify the address at H'FFF8 3000 to the stack pointer (R15). Allocate this processing at the address H'FFF8 2000, and use the assembly language to avoid the downloader using the undefined stack pointer. After setting the stack pointer, the processing can be described in C (the downloader entry function).

3.3.5 Interrupt Mask

Specify B'1111 in the interrupt mask level bit of the status register (SR) as the downloader does not support interrupts during it is operating.

3.3.6 Initialization

Initialize the serial flash memory before accessing.

1. Configure the RSPI0
2. Issue the Write Status Register command to the serial flash memory to cancel the software protection.

3.3.7 Writing the Loader Program

The downloader reads the loader program that has been transferred at the address from H'FFF8 0000 to H'FFF8 1FFF in the high-speed on-chip RAM, and writes the loader program at the address from H'0000 0000 to H'0000 1FFF in the serial flash memory. Table 10 lists the loader program writing.

Table 10 Loader Program Writing

Item	Description
Loader program transfer source address (high-speed on-chip RAM)	H'FFF8 0000 (fixed)
Loader program transfer destination address (serial flash memory)	H'0000 0000 (fixed)
Transfer size	8KB (fixed)
Writing procedures	1. Check if the destination address is already erased 2. Erase the data when the address is not erased 3. Issue the program command to write the loader program in units of single byte

3.3.8 Writing the Application Program

The downloader writes the application program in the large-capacity on-chip RAM at the address from H'0000 2000. Table 11 lists the application program writing processing.

Table 11 Application Program Writing

Item	Description
Application program transfer source address (large-capacity on-chip RAM)	Retrieve from the appinfo in the application program (Application program dependent)
Application program transfer destination address (serial flash memory)	H'0000 2000 (fixed)
Transfer size	Extracts from the appinfo in the application program (Application program dependent)
Writing procedures	1. Check if the destination address is already erased 2. Erase the data when the address is not erased 3. Issue the program command to write the application program in units of single byte

3.3.9 Serial Flash Memory Commands

Table 12 lists the serial flash memory commands used in the downloader. Issue these commands via Renesas Serial Peripheral Interface channel 0 (RSPI0) to operate the serial flash memory.

Table 12 Serial Flash Memory Commands

Command Name	Opcode	Function
Quad Output Read	H'6B	Reads the quad data
Write Enable	H'06	Enables the commands, program(write),/erase/ write status register
Write Disable	H'04	Disables the commands, program(write)/ erase/ write status register, and so on
Read Status Register	H'05	Reads the status register
Write Status Register	H'01	Writes the status register (cancel the software protection)
64Kbyte Block Erase	H'D8	Erases the data in blocks (64KB)
Byte Program	H'02	Writes data (1byte)

Notes: 1. This application refers to the commands of the S25FL032P by SPANSION. As the serial flash memory commands depend on the type of the serial flash memory, refer to the datasheet provided by the serial flash memory manufacturer.
2. Erase the data in the destination address in the serial flash memory before writing in it.

3.3.10 Batch File

The loader program and the downloader must be transferred to the high-speed on-chip RAM, and the application program must be transferred to the large-capacity on-chip RAM to write the loader program and the application program in the serial flash memory.

This application note uses the command batch file in the High-performance Embedded Workshop to execute the series of above described processing automatically.

Figure 15 shows the flow chart of operating the command batch file. The command batch file is used to transfer programs to the high-speed on-chip RAM and the large-capacity on-chip RAM, and write programs in the serial flash memory.

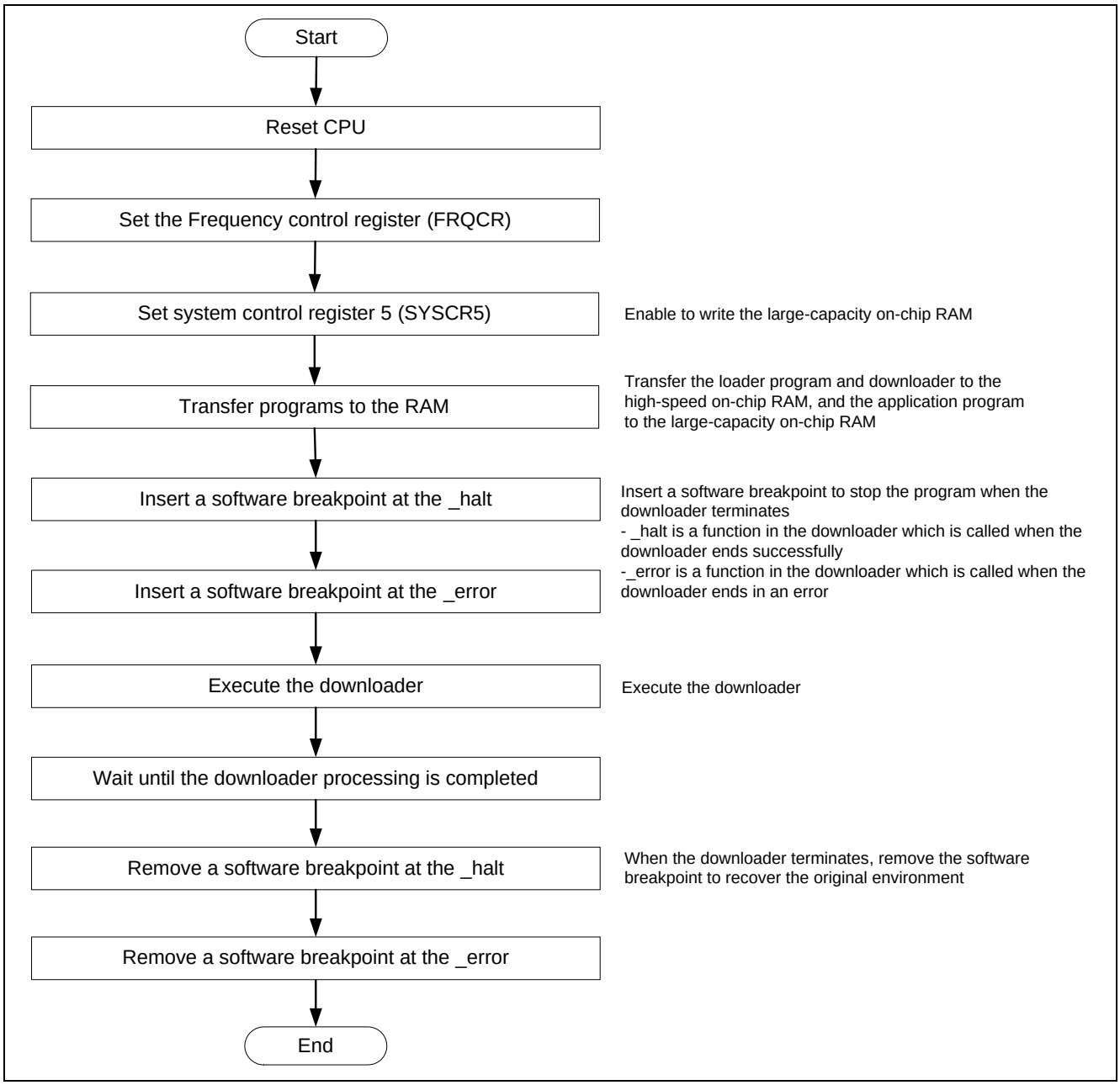

Figure 15 Command Batch File Flow Chart

4. Sample Program

4.1 Loader Program

4.1.1 Loader Program List "loader.src" (1)

```
1     ;/***********************************************************************
2     ;*   DISCLAIMER
3     ;*
4     ;*   This software is supplied by Renesas Electronics Corporation and is only
5     ;*   intended for use with Renesas products. No other uses are authorized.
6     ;*
7     ;*   This software is owned by Renesas Electronics Corporation and is protected under
8     ;*   all applicable laws, including copyright laws.
9     ;*
10    ;*   THIS SOFTWARE IS PROVIDED "AS IS" AND RENESAS MAKES NO WARRANTIES
11    ;*   REGARDING THIS SOFTWARE, WHETHER EXPRESS, IMPLIED OR STATUTORY,
12    ;*   INCLUDING BUT NOT LIMITED TO WARRANTIES OF MERCHANTABILITY, FITNESS FOR A
13    ;*   PARTICULAR PURPOSE AND NON-INFRINGEMENT.  ALL SUCH WARRANTIES ARE EXPRESSLY
14    ;*   DISCLAIMED.
15    ;*
16    ;*   TO THE MAXIMUM EXTENT PERMITTED NOT PROHIBITED BY LAW, NEITHER RENESAS
17    ;*   ELECTRONICS CORPORATION NOR ANY OF ITS AFFILIATED COMPANIES SHALL BE LIABLE
18    ;*   FOR ANY DIRECT, INDIRECT, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES
19    ;*   FOR ANY REASON RELATED TO THIS SOFTWARE, EVEN IF RENESAS OR ITS
20    ;*   AFFILIATES HAVE BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.
21    ;*
22    ;*   Renesas reserves the right, without notice, to make changes to this
23    ;*   software and to discontinue the availability of this software.
24    ;*   By using this software, you agree to the additional terms and
25    ;*   conditions found by accessing the following link:
26    ;*   http://www.renesas.com/disclaimer
27    ;***********************************************************************
28    ;*   Copyright (C) 2010 Renesas Electronics Corporation. All rights reserved.
29    ;*""FILE COMMENT""*********** Technical reference data *************************
30    ;*   System Name : SH7266/SH7267 Sample Program
31    ;*   File Name   : loader.src
32    ;*   Abstract    : Loader program preprocessing/jump processing to the application
33    ;*   Version     : 1.00.00
34    ;*   Device      : SH7266/SH7267
35    ;*   Tool-Chain  : High-performance Embedded Workshop (Ver.4.07.00).
36    ;*               : C/C++ compiler package for the SuperH RISC engine family
37    ;*               :                               (Ver.9.03 Release02).
38    ;*   OS          : None
39    ;*   H/W Platform: R0K57267(CPU board)
40    ;*   Description :
41    ;***********************************************************************
42    ;*   History     : Aug.17,2010 Ver.1.00.00 First Release
43    ;*""FILE COMMENT END""*****************************************************/
44      .SECTION LOADER_ENTRY,CODE,ALIGN = 4
45      .IMPORT _main
46      .EXPORT _jmp_app_prog
47
```

4.1.2 Loader Program List "loader.src" (2)

```
48    _loader_prog:
49      MOV.L L2,R15 ; Sets the stack pointer
50      MOV.L L1,R0      ; Retrieves the entry function of the loader program
51      JMP @R0          ; Jumps to the entry function of the loader program
52      NOP
53
54
55    ;/*""FUNC COMMENT""*****************************************************************
56    ; * ID          :
57    ; * Outline     : Jump to the application program
58    ; *-------------------------------------------------------------------------------
59    ; * Include     :
60    ; *-------------------------------------------------------------------------------
61    ; * Declaration : _jmp_app_prog
62    ; *-------------------------------------------------------------------------------
63    ; * Description : 1. Retrieves the stack pointer value stored in the first 12 to
64    ; *             :    15 bytes in the application program.
65    ; *             : 2. Specifies the stack pointer (R15).
66    ; *             : 3. Retrieves the entry function address stored in the first 8 to
67    ; *             :    11 bytes in the application program.
68    ; *             : 4. Jumps to the entry function.
69    ; *-------------------------------------------------------------------------------
70    ; * Argument    : R4  ; I : Start address of the application program
71    ; *-------------------------------------------------------------------------------
72    ; * Return Value: none
73    ; *""FUNC COMMENT END""*************************************************************/
74    _jmp_app_prog:
75
76      MOV.L R4,R0      ; Substitutes the start address of the application program for R0
77      ADD #12,R0       ; Calculates the address storing the stack pointer value and
78                       ; substitutes the address for R0
79      MOV.L @R0,R15    ; Sets the stack pointer
80
81      MOV.L R4,R0      ; Substitutes the start address of the application program for R0
82      ADD #8,R0        ; Calculates the address storing the entry function of the application
83                       ; program and substitutes the address for R0
84      MOV.L @R0,R0     ; Substitutes the entry function address of the application
85                       ; program for R0
86      JMP @R0          ; Jumps to the entry function of the application program
87      NOP
88
89
90      .ALIGN 4
91    L1:
92      .DATA.L _main               ; Entry function address of the loader program
93
94    L2:
95      .DATA.L H'FFF82000          ; Stack pointer (R15) value of the loader program
96
97      .pool
98      .end
99
100   ;/* End of File */
```

4.1.3 Loader Program List "ld_main.c" (1)

```
1     /***********************************************************************
2     *   DISCLAIMER
3     *
4     *   This software is supplied by Renesas Electronics Corporation and is only
5     *   intended for use with Renesas products. No other uses are authorized.
6     *
7     *   This software is owned by Renesas Electronics Corporation and is protected under
8     *   all applicable laws, including copyright laws.
9     *
10    *   THIS SOFTWARE IS PROVIDED "AS IS" AND RENESAS MAKES NO WARRANTIES
11    *   REGARDING THIS SOFTWARE, WHETHER EXPRESS, IMPLIED OR STATUTORY,
12    *   INCLUDING BUT NOT LIMITED TO WARRANTIES OF MERCHANTABILITY, FITNESS FOR A
13    *   PARTICULAR PURPOSE AND NON-INFRINGEMENT.  ALL SUCH WARRANTIES ARE EXPRESSLY
14    *   DISCLAIMED.
15    *
16    *   TO THE MAXIMUM EXTENT PERMITTED NOT PROHIBITED BY LAW, NEITHER RENESAS
17    *   ELECTRONICS CORPORATION NOR ANY OF ITS AFFILIATED COMPANIES SHALL BE LIABLE
18    *   FOR ANY DIRECT, INDIRECT, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES
19    *   FOR ANY REASON RELATED TO THIS SOFTWARE, EVEN IF RENESAS OR ITS
20    *   AFFILIATES HAVE BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.
21    *
22    *   Renesas reserves the right, without notice, to make changes to this
23    *   software and to discontinue the availability of this software.
24    *   By using this software, you agree to the additional terms and
25    *   conditions found by accessing the following link:
26    *   http://www.renesas.com/disclaimer
27    ***********************************************************************
28    * Copyright (C) 2010(2011) Renesas Electronics Corporation. All rights reserved.
29    *""FILE COMMENT""*********** Technical reference data *************************
30    *   System Name : SH7266/SH7267 Sample Program
31    *   File Name   : ld_main.c
32    *   Abstract    : Loader program
33    *   Version     : 1.01.00
34    *   Device      : SH7266/SH7267
35    *   Tool-Chain  : High-performance Embedded Workshop (Ver.4.07.00).
36    *               : C/C++ compiler package for the SuperH RISC engine family
37    *               :                             (Ver.9.03 Release02).
38    *   OS          : None
39    *   H/W Platform: R0K57267(CPU board)
40    *   Description :
41    ***********************************************************************
42    *   History     : Aug.17,2010 Ver.1.00.00 First Release
43    *               : Apr.14,2011 Ver.1.01.00 Modified for RQSPI
44    *""FILE COMMENT END""*****************************************************/
45    #include <stdio.h>
46    #include <string.h>
47    #include <machine.h>
48    #include "iodefine.h"
49    #include "ld_qserial_flash.h"
50
```

4.1.4 Loader Program List "ld_main.c" (2)

```
51  /* ==== macro defined ==== */
52  #define FPSCR_INIT     0x00040001         /* Value to set in the FPSCR register */
53  #define INT_MASK       0x000000F0         /* Value to set in the SR register
54                                            (for masking the interrupt) */
55
56  #define APROG_TOP_SFLASH   0x00002000     /* Start address of the application program */
57                                            /* (serial flash memory) */
58
59  #define APPINFO_TOP    APROG_TOP_SFLASH   /* Address the appinfo.app_top is located */
60  #define APPINFO_END    (APROG_TOP_SFLASH + 4) /* Address the appinfo.app_end is located */
61
62
63  /* ==== prototype declaration ==== */
64  void main(void);
65  void get_appinfo( unsigned long *app_top_addr,unsigned long *app_end_addr);
66  void app_prog_transfer(unsigned long app_top_addr,unsigned long app_end_addr);
67  void system_down(void);
68
69  extern void jmp_app_prog(unsigned long app_top_addr);
70  extern void io_set_cpg(void);
71  extern void io_init_cache(void);
72  extern int io_cache_writeback(void);
73
74
75  /* ==== external data ==== */
76  extern unsigned long DUMMY_Vectors;
77
78
```

4.1.5 Loader Program List "ld_main.c" (3)

```
79    /*""FUNC COMMENT""*****************************************************************
80     * ID          :
81     * Outline     : Loader program main
82     *-------------------------------------------------------------------------------
83     * Include     : #include "serial_flash.h"
84     *-------------------------------------------------------------------------------
85     * Declaration : void main(void);
86     *-------------------------------------------------------------------------------
87     * Description : Refers the data in the appinfo to transfer the application program
88     *             : to the large-capacity internal RAM, and jumps to the entry function
89     *             : of the application program.
90     *-------------------------------------------------------------------------------
91     * Argument    : void
92     *-------------------------------------------------------------------------------
93     * Return Value: void
94     *""FUNC COMMENT END""*************************************************************/
95    void main(void)
96    {
97      unsigned long app_top,app_end;
98
99      /* Sets the FPSCR */
100     set_fpscr(FPSCR_INIT);
101
102     /* Sets the tentative VBR */
103     set_vbr((void *)(&DUMMY_Vectors));
104
105     /* Masks the interrupt */
106     set_cr(INT_MASK);
107
108     /* Sets the CPG */
109     io_set_cpg();
110
111     /* Enables the cache */
112     io_init_cache();
113
114     /* Initializes the serial flash memory */
115     sf_init_serial_flash();
116
117     /* Retrieves the appinfo */
118     get_appinfo(&app_top,&app_end);
119
120     /* Transfers the application program to the large-capacity internal RAM */
121     app_prog_transfer(app_top, app_end);
122
123     /* Writes back the cache */
124     io_cache_writeback();
125
126     /* Jumps to the application program */
127     jmp_app_prog(app_top);
```

4.1.6 Loader Program List "ld_main.c" (4)

```
128
129     while(1){
130       /* LOOP */
131     }
132   }
133
134   /*""FUNC COMMENT""*****************************************************************
135    * ID          :
136    * Outline     : Retrieve the appinfo
137    *-------------------------------------------------------------------------------
138    * Include     : #include "serial_flash.h"
139    *-------------------------------------------------------------------------------
140    * Declaration : void get_appinfo (unsigned long *app_top_addr,
141    *             :                   unsigned long *app_end_addr);
142    *-------------------------------------------------------------------------------
143    * Description : Retrieves the appinfo.
144    *             : Retrieves the appinfo.top from H'2000 to H'2003 in serial flash
145    *             : memory, and stores it in the address specified by the first
146    *             : argument. This function also retrieves the appinfo.end from
147    *             : H'2004 to H'2007 in serial flash memory, and stores it in the
148    *             : address specified by the second argument.
149    *-------------------------------------------------------------------------------
150    * Argument    : unsigned long app_top_addr  ; O : Start address of the application
151    *             :                                   program at destination
152    *             : unsigned long app_end_addr  ; O : End address of the application
153    *             :                                   program at destination
154    *-------------------------------------------------------------------------------
155    * Return Value: void
156    *""FUNC COMMENT END""*************************************************************/
157   void get_appinfo( unsigned long *app_top_addr,unsigned long *app_end_addr)
158   {
159     /* Retrieves the appinfo.top */
160     sf_byte_read(APPINFO_TOP, (unsigned char *)app_top_addr, 4);
161
162     /* Retrieves the appinfo.end */
163     sf_byte_read(APPINFO_END, (unsigned char *)app_end_addr, 4);
164   }
165
```

4.1.7 Loader Program List "ld_main.c" (5)

```
/*""FUNC COMMENT""*****************************************************************
 * ID          :
 * Outline     : Transfer the application program
 *-----------------------------------------------------------------------------
 * Include     : #include "serial_flash.h"
 *-----------------------------------------------------------------------------
 * Declaration : void app_prog_transfer(unsigned long app_top_addr,
 *             :                        unsigned long app_end_addr);
 *-----------------------------------------------------------------------------
 * Description : Calculates the size of the application program, and transfers
 *             : the application program from serial flash memory to the
 *             : large-capacity internal RAM. (Rounds up the allocation size of the
 *             : application program to multiples of 4 to transfer in longword.)
 *-----------------------------------------------------------------------------
 * Argument    : unsigned long app_top_addr  ; I : Start address of the application
 *             :                                   program at destination
 *             : unsigned long app_end_addr  ; I : End address of the application
 *             :                                   at destination
 *-----------------------------------------------------------------------------
 * Return Value: void
 *""FUNC COMMENT END""*************************************************************/
void app_prog_transfer(unsigned long app_top_addr,unsigned long app_end_addr)
{
  unsigned long app_prog_size;

  /* Calculates the size of the application program */
  app_prog_size = app_end_addr - app_top_addr;
  if( ( app_prog_size & 0x0000000F ) != 0 ){
    app_prog_size &= 0xFFFFFFF0;
    app_prog_size += 16;      /* Rounds up the allocation size of the application
                              program to multiples of 16. */
  }

  /* Loads the application program in the large-capacity internal RAM */
  sf_byte_read(APROG_TOP_SFLASH, (unsigned char *)app_top_addr, app_prog_size);
}

```

4.1.8 Loader Program List "ld_main.c" (6)

```
203   /*""FUNC COMMENT""*****************************************************************
204    * ID          :
205    * Outline     : Terminate the system
206    *-------------------------------------------------------------------------------
207    * Include     :
208    *-------------------------------------------------------------------------------
209    * Declaration : void system_down(void);
210    *-------------------------------------------------------------------------------
211    * Description : This function contains the infinite loop.
212    *             : As this is registered in the DUMMY_Vectors table, this is
213    *             : called when an exception occurs while the loader program
214    *             : is operating.
215    *-------------------------------------------------------------------------------
216    * Argument    : void
217    *-------------------------------------------------------------------------------
218    * Return Value: void
219    *""FUNC COMMENT END""*************************************************************/
220   void system_down(void)
221   {
222     while(1){
223       /* System error */
224     }
225   }
226
227   /* End of File */
```

4.2 Application Program

4.2.1 Application Program List "main.c" (1)

```
1     /***********************************************************************
2     *   DISCLAIMER
3     *
4     *   This software is supplied by Renesas Electronics Corporation and is only
5     *   intended for use with Renesas products. No other uses are authorized.
6     *
7     *   This software is owned by Renesas Electronics Corporation and is protected under
8     *   all applicable laws, including copyright laws.
9     *
10    *   THIS SOFTWARE IS PROVIDED "AS IS" AND RENESAS MAKES NO WARRANTIES
11    *   REGARDING THIS SOFTWARE, WHETHER EXPRESS, IMPLIED OR STATUTORY,
12    *   INCLUDING BUT NOT LIMITED TO WARRANTIES OF MERCHANTABILITY, FITNESS FOR A
13    *   PARTICULAR PURPOSE AND NON-INFRINGEMENT.  ALL SUCH WARRANTIES ARE EXPRESSLY
14    *   DISCLAIMED.
15    *
16    *   TO THE MAXIMUM EXTENT PERMITTED NOT PROHIBITED BY LAW, NEITHER RENESAS
17    *   ELECTRONICS CORPORATION NOR ANY OF ITS AFFILIATED COMPANIES SHALL BE LIABLE
18    *   FOR ANY DIRECT, INDIRECT, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES
19    *   FOR ANY REASON RELATED TO THIS SOFTWARE, EVEN IF RENESAS OR ITS
20    *   AFFILIATES HAVE BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.
21    *
22    *   Renesas reserves the right, without notice, to make changes to this
23    *   software and to discontinue the availability of this software.
24    *   By using this software, you agree to the additional terms and
25    *   conditions found by accessing the following link:
26    *   http://www.renesas.com/disclaimer
27    ************************************************************************
28    * Copyright (C) 2010(2011) Renesas Electronics Corporation. All rights reserved.
29    *""FILE COMMENT""*********** Technical reference data *********************
30    *   System Name : SH7266/SH7267 Sample Program
31    *   File Name   : main.c
32    *   Abstract    : Application program example
33    *   Version     : 1.01.00
34    *   Device      : SH7266/SH7267
35    *   Tool-Chain  : High-performance Embedded Workshop (Ver.4.07.00).
36    *               : C/C++ compiler package for the SuperH RISC engine family
37    *               :                             (Ver.9.03 Release02).
38    *   OS          : None
39    *   H/W Platform: R0K57267(CPU board)
40    *   Description :
41    ************************************************************************
42    *   History     : Aug.17,2010 Ver.1.00.00 First Release
43    *               : Apr.14,2011 Ver.1.01.00 Modified for RQSPI
44    *""FILE COMMENT END""*****************************************************/
45    #include <stdio.h>
46
47    /* ==== prototype declaration ==== */
48    void main(void);
49
```

4.2.2 Application Program List "main.c" (2)

```
50    /*""FUNC COMMENT""*****************************************************************
51     * ID          :
52     * Outline     : Application program main function
53     *-------------------------------------------------------------------------------
54     * Include     :
55     *-------------------------------------------------------------------------------
56     * Declaration : void main(void);
57     *-------------------------------------------------------------------------------
58     * Description : Transmits the strings of characters to the SCIF0.
59     *             : (Baud rate: 57600 bps, no parity, stop bit length: 1).
60     *-------------------------------------------------------------------------------
61     * Argument    : void
62     *-------------------------------------------------------------------------------
63     * Return Value: void
64     *""FUNC COMMENT END""*************************************************************/
65    void main(void)
66    {
67      puts("\nSH7267 CPU Board Sample Program. Ver.1.01.00");
68        puts("Copyright (C) 2010(2011) Renesas Electronics Corporation. All rights
69    reserved.\n");
70      puts("SH7266/SH7267 Serial-flash boot done.\n");
71      fflush(stdout);
72
73      while(1){
74        /* loop */
75      }
76    }
77    /* End of File */
```

4.2.3 Application Program List "appinfo.c" (1)

```
1     /***********************************************************************
2     *   DISCLAIMER
3     *
4     *   This software is supplied by Renesas Electronics Corporation and is only
5     *   intended for use with Renesas products. No other uses are authorized.
6     *
7     *   This software is owned by Renesas Electronics Corporation and is protected under
8     *   all applicable laws, including copyright laws.
9     *
10    *   THIS SOFTWARE IS PROVIDED "AS IS" AND RENESAS MAKES NO WARRANTIES
11    *   REGARDING THIS SOFTWARE, WHETHER EXPRESS, IMPLIED OR STATUTORY,
12    *   INCLUDING BUT NOT LIMITED TO WARRANTIES OF MERCHANTABILITY, FITNESS FOR A
13    *   PARTICULAR PURPOSE AND NON-INFRINGEMENT.  ALL SUCH WARRANTIES ARE EXPRESSLY
14    *   DISCLAIMED.
15    *
16    *   TO THE MAXIMUM EXTENT PERMITTED NOT PROHIBITED BY LAW, NEITHER RENESAS
17    *   ELECTRONICS CORPORATION NOR ANY OF ITS AFFILIATED COMPANIES SHALL BE LIABLE
18    *   FOR ANY DIRECT, INDIRECT, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES
19    *   FOR ANY REASON RELATED TO THIS SOFTWARE, EVEN IF RENESAS OR ITS
20    *   AFFILIATES HAVE BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.
21    *
22    *   Renesas reserves the right, without notice, to make changes to this
23    *   software and to discontinue the availability of this software.
24    *   By using this software, you agree to the additional terms and
25    *   conditions found by accessing the following link:
26    *   http://www.renesas.com/disclaimer
27    ***********************************************************************
28    *   Copyright (C) 2010 Renesas Electronics Corporation. All rights reserved.
29    *""FILE COMMENT""*********** Technical reference data *************************
30    *   System Name : SH7266/SH7267 Sample Program
31    *   File Name   : appinfo.c
32    *   Abstract    : Generate the application program transfer information (appinfo).
33    *   Version     : 1.00.00
34    *   Device      : SH7266/SH7267
35    *   Tool-Chain  : High-performance Embedded Workshop (Ver.4.07.00).
36    *               : C/C++ compiler package for the SuperH RISC engine family
37    *               :                              (Ver.9.03 Release02).
38    *   OS          : None
39    *   H/W Platform: R0K57267(CPU board)
40    *   Description :
41    ***********************************************************************
42    *   History     : Aug.17,2010 Ver.1.00.00 First Release
43    *""FILE COMMENT END""*****************************************************/
```

4.2.4 Application Program List "appinfo.c" (2)

```
44    #include "appinfo.h"
45
46    #pragma section APPINFO
47
48    static APPINFO appinfo = {
49      __sectop("DAPPINFO"),    /* Start address in the start section of the application */
50                               /* program (program area, constant area, and initialized */
51                               /* data area). */
52
53      __secend("PCACHE")       /* End address in the end section of the application */
54                               /* program (program area, constant area, and initialized */
55                               /* data area) */
56    };
57
58    /* End of File */
59
```

4.2.5 Application Program Listing "appinfo.h"

```
/***********************************************************************
*   DISCLAIMER
*
*   This software is supplied by Renesas Electronics Corporation and is only
*   intended for use with Renesas products. No other uses are authorized.
*
*   This software is owned by Renesas Electronics Corporation 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 CORPORATION 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
***********************************************************************
*   Copyright (C) 2010 Renesas Electronics Corporation. All rights reserved.
*""FILE COMMENT""*********** Technical reference data *************************
*   System Name : SH7266/SH7267 Sample Program
*   File Name    : appinfo.h
*   Abstract     : Header file of the application program transfer information (appinfo).
*   Version      : 1.00.00
*   Device       : SH7266/SH7267
*   Tool-Chain  : High-performance Embedded Workshop (Ver.4.07.00).
*                 : C/C++ compiler package for the SuperH RISC engine family
*                 :                              (Ver.9.03 Release02).
*   OS           : None
*   H/W Platform: R0K57267(CPU board)
*   Description :
***********************************************************************
*   History      : Aug.17,2010 Ver.1.00.00 First Release
*""FILE COMMENT END""*****************************************************/
#ifndef __APPINFO_H__
#define __APPINFO_H__

typedef struct appinfo_t {
  void *app_top;          /* Start address of the application program */
  void *app_end;          /* End address of the application program */
} APPINFO;

#endif  /* __APPINFO_H__ */

/* End of File */
```

4.3 Downloader

4.3.1 Downloader Program List "downloader.hdc" (1)

```
1     #/*************************************************************************
2     #*   DISCLAIMER
3     #*
4     #*   This software is supplied by Renesas Electronics Corporation and is only
5     #*   intended for use with Renesas products. No other uses are authorized.
6     #*
7     #*   This software is owned by Renesas Electronics Corporation and is protected under
8     #*   all applicable laws, including copyright laws.
9     #*
10    #*   THIS SOFTWARE IS PROVIDED "AS IS" AND RENESAS MAKES NO WARRANTIES
11    #*   REGARDING THIS SOFTWARE, WHETHER EXPRESS, IMPLIED OR STATUTORY,
12    #*   INCLUDING BUT NOT LIMITED TO WARRANTIES OF MERCHANTABILITY, FITNESS FOR A
13    #*   PARTICULAR PURPOSE AND NON-INFRINGEMENT.  ALL SUCH WARRANTIES ARE EXPRESSLY
14    #*   DISCLAIMED.
15    #*
16    #*   TO THE MAXIMUM EXTENT PERMITTED NOT PROHIBITED BY LAW, NEITHER RENESAS
17    #*   ELECTRONICS CORPORATION NOR ANY OF ITS AFFILIATED COMPANIES SHALL BE LIABLE
18    #*   FOR ANY DIRECT, INDIRECT, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES
19    #*   FOR ANY REASON RELATED TO THIS SOFTWARE, EVEN IF RENESAS OR ITS
20    #*   AFFILIATES HAVE BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.
21    #*
22    #*   Renesas reserves the right, without notice, to make changes to this
23    #*   software and to discontinue the availability of this software.
24    #*   By using this software, you agree to the additional terms and
25    #*   conditions found by accessing the following link:
26    #*   http://www.renesas.com/disclaimer
27    #*************************************************************************
28    #*   Copyright (C) 2010 Renesas Electronics Corporation. All rights reserved.
29    #*""FILE COMMENT""*********** Technical reference data *************************
30    #*   System Name : SH7266/SH7267 Sample Program
31    #*   File Name   : downloader.hdc
32    #*   Abstract    : Batch File for the Downloader
33    #*   Version     : 1.00.00
34    #*   Device      : SH7266/SH7267
35    #*   Tool-Chain  : High-performance Embedded Workshop (Ver.4.07.00).
36    #*                 : C/C++ compiler package for the SuperH RISC engine family
37    #*                 :                              (Ver.9.03 Release02).
38    #*   OS            : None
39    #*   H/W Platform: R0K57267(CPU board)
40    #*   Description :
41    #*************************************************************************
42    #*   History     : Aug.17,2010 Ver.1.00.00 First Release
43    #*""FILE COMMENT END""*****************************************************/
44
45
46    tcl enable
47
48
```

4.3.2 Downloader Program List "downloader.hdc" (2)

```
49    #Macro downloader -Start
50    proc init_hardware {} {
51
52      # Set the CPG
53      # FRQCR I=144MHz/B=72MHz/P=36MHz/CLK MODE0
54      MF H'FFFE0010 H'FFFE0011 H'1103 WORD
55
56      # On-Chip Large-Capacity RAM write enable
57      # CPG.SYSCR5.BYTE = 0x0fu;
58      MF H'FFFE0428 H'FFFE0428 H'0F BYTE
59    }
60
61
62    proc downloader {} {
63      # Reset CPU
64      reset
65
66      # Calls the init_hardware routine
67      init_hardware
68
69      # Downloads all modules registered in the High-performance Embedded Workshop
70      file_load_all
71
72      # Enables the user stack (to use the software breakpoint)
73      sh2a_sbstk enable
74
75      # Inserts a software breakpoint at the _halt (refer to main.c)
76      set_disassembly_soft_break _halt set
77
78      # Inserts a software breakpoint at the _error (refer to main.c)
79      set_disassembly_soft_break _error set
80
81      # Executes the _downloader (refer to downloader.src) to wait until it terminates
82      go wait _downloader
83
84      # Removes a software breakpoint at the _halt
85      set_disassembly_soft_break _halt clear
86
87      # Removes a software breakpoint at the _error
88      set_disassembly_soft_break _error clear
89
90    }
91
92    downloader
93    #Macro downloader -End
94
95
96
97    # Note: "tcl", "reset", "file_load", "sh2a_sbstk", "set_disassembly_soft_break",
98    # and "go" are commands used in the High-performance Embedded Workshop and the
99    # E10A-USB emulator. For details, refer to manuals.
100
101   # /* End of File */
```

4.3.3 Downloader Program List "downloader.src" (1)

```
1     ;/***********************************************************************
2     ;*   DISCLAIMER
3     ;*
4     ;*   This software is supplied by Renesas Electronics Corporation and is only
5     ;*   intended for use with Renesas products. No other uses are authorized.
6     ;*
7     ;*   This software is owned by Renesas Electronics Corporation and is protected under
8     ;*   all applicable laws, including copyright laws.
9     ;*
10    ;*   THIS SOFTWARE IS PROVIDED "AS IS" AND RENESAS MAKES NO WARRANTIES
11    ;*   REGARDING THIS SOFTWARE, WHETHER EXPRESS, IMPLIED OR STATUTORY,
12    ;*   INCLUDING BUT NOT LIMITED TO WARRANTIES OF MERCHANTABILITY, FITNESS FOR A
13    ;*   PARTICULAR PURPOSE AND NON-INFRINGEMENT.  ALL SUCH WARRANTIES ARE EXPRESSLY
14    ;*   DISCLAIMED.
15    ;*
16    ;*   TO THE MAXIMUM EXTENT PERMITTED NOT PROHIBITED BY LAW, NEITHER RENESAS
17    ;*   ELECTRONICS CORPORATION NOR ANY OF ITS AFFILIATED COMPANIES SHALL BE LIABLE
18    ;*   FOR ANY DIRECT, INDIRECT, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES
19    ;*   FOR ANY REASON RELATED TO THIS SOFTWARE, EVEN IF RENESAS OR ITS
20    ;*   AFFILIATES HAVE BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.
21    ;*
22    ;*   Renesas reserves the right, without notice, to make changes to this
23    ;*   software and to discontinue the availability of this software.
24    ;*   By using this software, you agree to the additional terms and
25    ;*   conditions found by accessing the following link:
26    ;*   http://www.renesas.com/disclaimer
27    ;***********************************************************************
28    ;*   Copyright (C) 2010 Renesas Electronics Corporation. All rights reserved.
29    ;*""FILE COMMENT""*********** Technical reference data *************************
30    ;*   System Name : SH7266/SH7267 Sample Program
31    ;*   File Name   : downloader.src
32    ;*   Abstract     : Downloader
33    ;*   Version      : 1.00.00
34    ;*   Device       : SH7266/SH7267
35    ;*   Tool-Chain  : High-performance Embedded Workshop (Ver.4.07.00).
36    ;*                 : C/C++ compiler package for the SuperH RISC engine family
37    ;*                 :                              (Ver.9.03 Release02).
38    ;*   OS            : None
39    ;*   H/W Platform: R0K57267(CPU board)
40    ;*   Description :
41    ;***********************************************************************
42    ;*   History     : Aug.17,2010 Ver.1.00.00 First Release
43    ;*""FILE COMMENT END""*****************************************************/
```

4.3.4 Downloader Program List "downloader.src" (2)

```
44      .SECTION DOWNLOADER_ENTRY,CODE,ALIGN = 4
45      .IMPORT _main
46
47    _downloader:
48      MOV.L L2,R15        ; Sets the stack pointer
49      MOV.L L1,R0         ; Retrieves the entry function of the downloader
50      JMP @R0             ; Jumps to the entry function of the downloader
51      NOP
52
53      .ALIGN 4
54    L1:
55      .DATA.L _main       ; Entry function address of the downloader
56
57    L2:
58      .DATA.L H'FFF83000  ; Stack pointer (R15) value of the downloader
59
60      .pool
61      .end
```

4.3.5 Downloader Program List "dl_main.c" (1)

```
/***********************************************************************
*   DISCLAIMER
*
*   This software is supplied by Renesas Electronics Corporation and is only
*   intended for use with Renesas products. No other uses are authorized.
*
*   This software is owned by Renesas Electronics Corporation 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 CORPORATION 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
***********************************************************************
* Copyright (C) 2010 Renesas Electronics Corporation. All rights reserved.
*""FILE COMMENT""*********** Technical reference data *************************
*   System Name : SH7266/SH7267 Sample Program
*   File Name   : dl_main.c
*   Abstract    : Downloader
*   Version     : 1.00.00
*   Device      : SH7266/SH7267
*   Tool-Chain  : High-performance Embedded Workshop (Ver.4.07.00).
*               : C/C++ compiler package for the SuperH RISC engine family
*               :                             (Ver.9.03 Release02).
*   OS          : None
*   H/W Platform: R0K57267(CPU board), Spansion[S25FL032P](serial flash)
*   Description :
***********************************************************************
*   History     : Aug.17,2010 Ver.1.00.00 First Release
*""FILE COMMENT END""*****************************************************/
#include <stdio.h>
#include <string.h>
#include <machine.h>
#include "iodefine.h"
#include "dl_qserial_flash.h"

```

4.3.6 Downloader Program List "dl_main.c" (2)

```
50   /* ==== macro defined ==== */
51   #define INT_MASK     0x000000F0         /* Value to set in the SR register
52                                           (for masking the interrupt) */
53
54   #define SECTOR_SIZE  0x10000            /* Sector size: 64 KB */
55   #define SECTOR_NUM   32                 /* Total number of sectors in the device */
56   #define DEVICE_SIZE  (SECTOR_SIZE * SECTOR_NUM) /* Device size */
57
58   #define L_PROG_SIZE  8192               /* Loader program size */
59   #define L_PROG_SRC   0xFFF80000         /* Source address of the loader program */
60   #define L_PROG_DST   0x00000000         /* Destination address of the loader program */
61
62   #define APROG_TOP_SFLASH    0x00002000 /* Start address of the application program */
63   #define APROG_TOP_RAM       0x1C000000 /* Start address of the application program */
64                                           /* When changing the start section of the */
65                                           /* application program, change this definition */
66
67   #define APPINFO_TOP  APROG_TOP_RAM      /* Address the appinfo.app_top is located */
68   #define APPINFO_END  (APROG_TOP_RAM + 4)/* Address the appinfo.app_end is located */
69
70   /* ==== prototype declaration ==== */
71   /*** User API ****/
72   void main(void);
73
74   static void halt(void);
75   static void error(void);
76   static void init_erase_flag(void);
77   static int Is_erased_sector(unsigned long sector_no);
78   static void set_erase_flag(unsigned long sector_no);
79   static int write_prog_data(unsigned char *program_data, unsigned long sflash_addr, unsigned
       long size);
80
81   /*** data ***/
82   static unsigned char sflash_erase_flag[SECTOR_NUM]={0}; /* 0: sector not erased, 1: sector
       erased */
83
```

4.3.7 Downloader Program List "dl_main.c" (3)

```
84      /*""FUNC COMMENT""*****************************************************************
85       * ID          :
86       * Outline     : Downloader main
87       *-------------------------------------------------------------------------------
88       * Include     :
89       *-------------------------------------------------------------------------------
90       * Declaration : void main(void);
91       *-------------------------------------------------------------------------------
92       * Description : Writes the loader program and application program in serial
93       *             : flash memory as the following procedures.
94       *             : 1. Mask the interrupt while the downloader is operating.
95       *             : 2. Initialize the RSPI0.
96       *             : 3. Disable the software protection in serial flash memory.
97       *             : 4. Write the loader program in serial flash memory.
98       *             : 5. Write the application program in serial flash memory.
99       *-------------------------------------------------------------------------------
100      * Argument    : void
101      *-------------------------------------------------------------------------------
102      * Return Value: void
103      *""FUNC COMMENT END""*************************************************************/
104     void main(void)
105     {
106       unsigned long app_top_addr,app_end_addr,app_prog_size;
107
108       /* Masks the interrupt */
109       set_cr(INT_MASK);
110
111       /* Initializes the erase flag */
112       init_erase_flag();
113
114       /* Initializes the RSPI0 */
115       sf_init_serial_flash();
116
117       /* Disables the software protection in serial flash memory */
118       sf_protect_ctrl(SF_REQ_UNPROTECT);
119
120       /* Writes the loader program */
121       if( write_prog_data( (unsigned char *)L_PROG_SRC, L_PROG_DST, L_PROG_SIZE) < 0 ){
122         error();
123       }
124
125       /* Retrieves the start address and end address from the application program
126         transfer information (appinfo) */
127       app_top_addr = *(volatile unsigned long *)APPINFO_TOP;
128       app_end_addr = *(volatile unsigned long *)APPINFO_END;
129
130       /* Calculates the size of the application program */
131       app_prog_size = app_end_addr - app_top_addr;
132
```

4.3.8 Downloader Program List "dl_main.c" (4)

```
133    /* Writes the application program */
134    if( write_prog_data((unsigned char *)app_top_addr, APROG_TOP_SFLASH, app_prog_size) < 0 ){
135      error();
136    }
137
138    /* Enables the software protection in serial flash memory */
139    sf_protect_ctrl(SF_REQ_PROTECT);
140
141    /* Exits the downloader */
142    halt();
143  }
144
145  /*""FUNC COMMENT""*****************************************************************
146   * ID          :
147   * Outline     : Write the program data
148   *-------------------------------------------------------------------------------
149   * Include     :
150   *-------------------------------------------------------------------------------
151   * Declaration : int write_prog_data(unsigned char *program_data,
152   *             :                     unsigned long sflash_addr, unsigned long size);
153   *-------------------------------------------------------------------------------
154   * Description : Writes the program data as the following procedures.
155   *             : 1. Erase the target sector when it is not erased.
156   *             : 2. Write the program data in serial flash memory.
157   *             : 3. Reads the data in serial flash memory and compare it with the
158   *             :    provided data.
159   *-------------------------------------------------------------------------------
160   * Argument    : unsigned char *program_data ; I : Start address of the program data
161   *             : unsigned long sflash_addr   ; I : Start address at the destination in
162   *                                                 serial flash memory
163   *             : unsigned long size          ; I : Write size
164   *-------------------------------------------------------------------------------
165   * Return Value: Equal or bigger than 0: Success
166   *             : Less than 0: Error
167   *""FUNC COMMENT END""*************************************************************/
```

4.3.9 Downloader Program List "dl_main.c" (5)

```
168    int write_prog_data(unsigned char *program_data, unsigned long sflash_addr, unsigned long
       size)
169    {
170      unsigned long sector_no;
171      unsigned long saddr;
172      unsigned long sz;
173      unsigned char read_data;
174      unsigned char *w_p;
175
176      /* ==== Copies the value from the argument to the local variable ==== */
177      saddr = sflash_addr;
178      sz = size;
179      w_p = program_data;
180
181      /* ==== Writes data in serial flash memory ==== */
182      while( sz > 0){
183        sector_no = saddr / SECTOR_SIZE;
184        if( Is_erased_sector(sector_no) == 0 ){   /* When it is not erased */
185            sf_sector_erase(sector_no);           /* Erase */
186            set_erase_flag(sector_no);            /* When it is erased, set the erase flag */
187        }
188
189        sf_byte_program(saddr, w_p, 1);           /* Writes data in units of single byte */
190        w_p++;
191        saddr++;
192        sz--;
193      }
194
195      /* ==== Verifies data (serial flash memory is programmed successfully) ==== */
196      saddr = sflash_addr;
197      sz = size;
198      w_p = program_data;
199
200      while( sz > 0){
201        sf_byte_read(saddr,&read_data, 1);/* Reads the data written in serial flash memory */
202
203        if( *w_p != read_data ){
204            return -1;                    /* Returns an error when the data unmatched */
205        }
206
207        w_p++;
208        saddr++;
209        sz--;
210      }
211
212      return 0;
213    }
214
```

4.3.10 Downloader Program List "dl_main.c" (6)

```
215    /*""FUNC COMMENT""*****************************************************************
216     * ID          :
217     * Outline     : Initialize the Erase Flag
218     *-------------------------------------------------------------------------------
219     * Include     :
220     *-------------------------------------------------------------------------------
221     * Declaration : static void init_erase_flag(void);
222     *-------------------------------------------------------------------------------
223     * Description : Initializes the table sflash_erase_flag[].
224     *-------------------------------------------------------------------------------
225     * Argument    : void
226     *-------------------------------------------------------------------------------
227     * Return Value: void
228     *""FUNC COMMENT END""*************************************************************/
229    static void init_erase_flag(void)
230    {
231      int i;
232
233      for( i=0; i < SECTOR_NUM ;i++){
234        sflash_erase_flag[i] = 0;
235      }
236    }
237
238    /*""FUNC COMMENT""*****************************************************************
239     * ID          :
240     * Outline     : Retrieve the Sector Erase Status
241     *-------------------------------------------------------------------------------
242     * Include     :
243     *-------------------------------------------------------------------------------
244     * Declaration : static int Is_erased_sector(unsigned long sector_no);
245     *-------------------------------------------------------------------------------
246     * Description : Returns the information (not erased or eraser) of the
247     *             : sector specified by the sector number.
248     *-------------------------------------------------------------------------------
249     * Argument    : unsigned long sector_no   ; I : Sector number
250     *-------------------------------------------------------------------------------
251     * Return Value: 1 : Sector in the specified address is already erased
252     *             : 0 : Sector in the specified address is not erased
253     *""FUNC COMMENT END""*************************************************************/
254    static int Is_erased_sector(unsigned long sector_no)
255    {
256      return sflash_erase_flag[sector_no];
257    }
258
```

4.3.11 Downloader Program List "dl_main.c" (7)

```
/*""FUNC COMMENT""*****************************************************************
 * ID          :
 * Outline     : Set the Erase Flag
 *-----------------------------------------------------------------------------
 * Include     :
 *-----------------------------------------------------------------------------
 * Declaration : static void set_erase_flag(unsigned long sector_no);
 *-----------------------------------------------------------------------------
 * Description : Sets the erase flag to modify the information of the specified
 *             : sector as erased.
 *-----------------------------------------------------------------------------
 * Argument    : unsigned long sector_no   ; I : Sector number
 *-----------------------------------------------------------------------------
 * Return Value: void
 *""FUNC COMMENT END""*************************************************************/
static void set_erase_flag(unsigned long sector_no)
{
  sflash_erase_flag[sector_no] = 1;
}

/*""FUNC COMMENT""*****************************************************************
 * ID          :
 * Outline     : Program stops (successful).
 *-----------------------------------------------------------------------------
 * Include     :
 *-----------------------------------------------------------------------------
 * Declaration : static void halt(void);
 *-----------------------------------------------------------------------------
 * Description : When the downloader ends successfully, this function is called
 *             : to stop the program.
 *-----------------------------------------------------------------------------
 * Argument    : void
 *-----------------------------------------------------------------------------
 * Return Value: void
 *""FUNC COMMENT END""*************************************************************/
static void halt(void)
{
    while(1){
        /* When the downloader ends successfully, this function stops the program. */
    }
}

```

4.3.12 Downloader Program List "dl_main.c" (8)

```
301    /*""FUNC COMMENT""*****************************************************************
302     * ID          :
303     * Outline     : Program stops (error).
304     *-------------------------------------------------------------------------------
305     * Include     :
306     *-------------------------------------------------------------------------------
307     * Declaration : static void error(void);
308     *-------------------------------------------------------------------------------
309     * Description : When the downloader ends in error, this function is called
310     *             : to stop the program.
311     *-------------------------------------------------------------------------------
312     * Argument    : void
313     *-------------------------------------------------------------------------------
314     * Return Value: void
315     *""FUNC COMMENT END""*************************************************************/
316    static void error(void)
317    {
318        while(1){
319        /* When the downloader ends in error, this function stops the program */
320        }
321    }
322
323    /* End of File */
324
```

5. Using the Downloader

The downloader in this application is designed to operate with the combination of the High-performance Embedded Workshop and the E10A-USB emulator. When using the downloader with other development tools, alter the program according to the tool.

Programs cannot be written in the serial flash memory by selecting the downloader module in the **Debug Settings** dialog box on the Debug menu. This section explains the procedures to write programs in the serial flash memory using the downloader.

5.1 Sample Program Configuration

The sample program consists of three workspaces as listed in Table 13.

Table 13 Sample Program Configuration

Workspace Name	Description
sh7267_sflash_downloader	Build the downloader in the project of this workspace
sh7267_sflash_loader_prg	Build the loader program in the project of this workspace
sh7267_sflash_app	Build the application program in the project of this workspace. The downloader which is created in the [sh7267_sflash_downloader] workspace, a batch file to boot the downloader, and the loader program which is created in the [sh7267_sflash_loader_prog] workspace are registered in the project of this workspace. Use these items to write the loader program and application program in the serial flash memory. When modifying the downloader and the loader program, overwrite their abs files in the sfhash_boot folder.

5.2 Writing Programs in the Serial Flash Memory

This section describes how to write the loader program and application program in the serial flash memory using the [sh7267_sflash_app] workspace.

5.2.1 Registering the Download Module and the Batch File

Figure 16 shows the directory configuration of the [sh7267_sflash_app] workspace. Download modules (A, B, and D) and a batch file (C) in the figure are registered in the project.

```
¥sh7267_sflash_app                  : Workspace directory
  |-sh7267_sflash_app               : Project directory
  |    |-debug                      :
  |        |-sh7267_sflash_app.abs  : Application program execute file---------- A
  |                                 :
  |-inc                             : Directory to store the common include files
  |-src                             : Directory to store the source files
  |-sflash_boot                     : Directory to store the downloader and loader program
    |-sh7267_sflash_downloader.abs  : Downloader execute file---------- B
    |-downloader.hdc                : Batch file to boot the downloader ---------- C
    |-sh7267_sflash_loader_prog.abs : Loader program execute file ---------- D
```

Figure 16 [sh7267_sflash_app] Workspace Directory Configuration

1. Changing the download module
   Change the download module registered in the project in the **Debug Settings** dialog box. On the **Debug** menu in the High-performance Embedded Workshop, click **Debug Settings**, and the dialog box appears.
   For registering the download modules, refer to the High-performance Embedded Workshop User's Manual.

2. Changing the batch file
   Change the batch file registered in the project in the **Set Batch File** dialog box. Following the next procedure, the **Set Batch File** dialog box will open. First, on the View menu in the High-performance Embedded Workshop, click the **Command Line** command to show the **Command Line** window. Then, open the **Set Batch File** dialog box from the **Batch File** pop-up menu on the **Command Line** window.
   For registering the batch file, refer to the High-performance Embedded Workshop User's Manual.

5.2.2 Procedures for Writing Programs

This section describes how to write the loader program and application program in the serial flash memory using the [sh7267_sflash_app] workspace.

1. Copy the [sh7267_sflash_app] workspace directory in C:\Workspace.
2. Double-click the [sh7267_sflash_app].hws in the workspace directory to activate the High-performance Embedded Workshop.
3. On the **Build** menu, select the **Build All** command to build the project. The application program is generated.
4. On the **Debug** menu, select the **Go** command to connect with the target device.
5. After the connection is established, select the **Command Line** command to show the **Command Line** window as shown in Figure 17.
6. Click the **Run Batch** button on the **Command Line** window to execute the registered batch file [downloader.hdc].

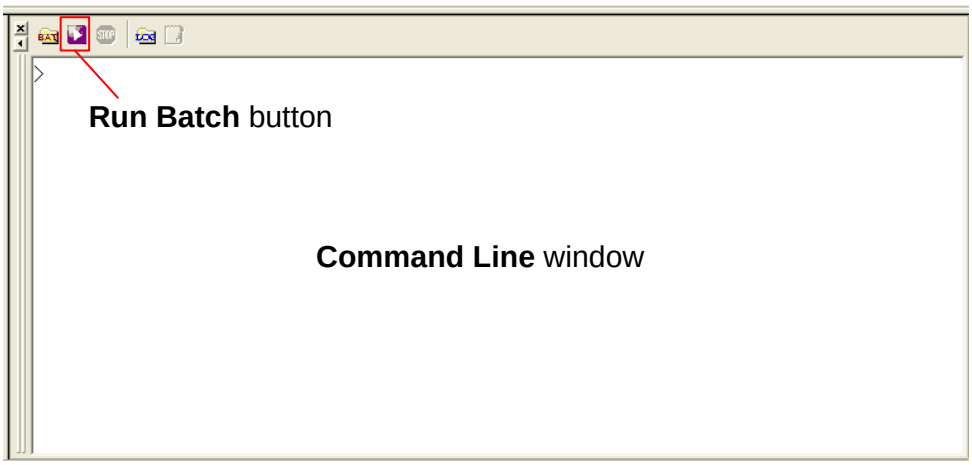

Figure 17 Command Line Window and Run Batch Button

7. When the batch file [downloader.hdc] is executed, all of the download modules registered in the workspace (loader program, application program, and downloader) are transferred to RAM to execute the downloader. As show in Figure 18, the program counter stops at the _halt when the downloader ends normally. The program counter stops at the _error when the downloader ends in an error. A source file may appear when the [sh7267_sflash_downloader] workspace directory is copied in C:\Workspace.
8. When writing is completed successfully, the loader program and application program can be executed after **Reset Go**.

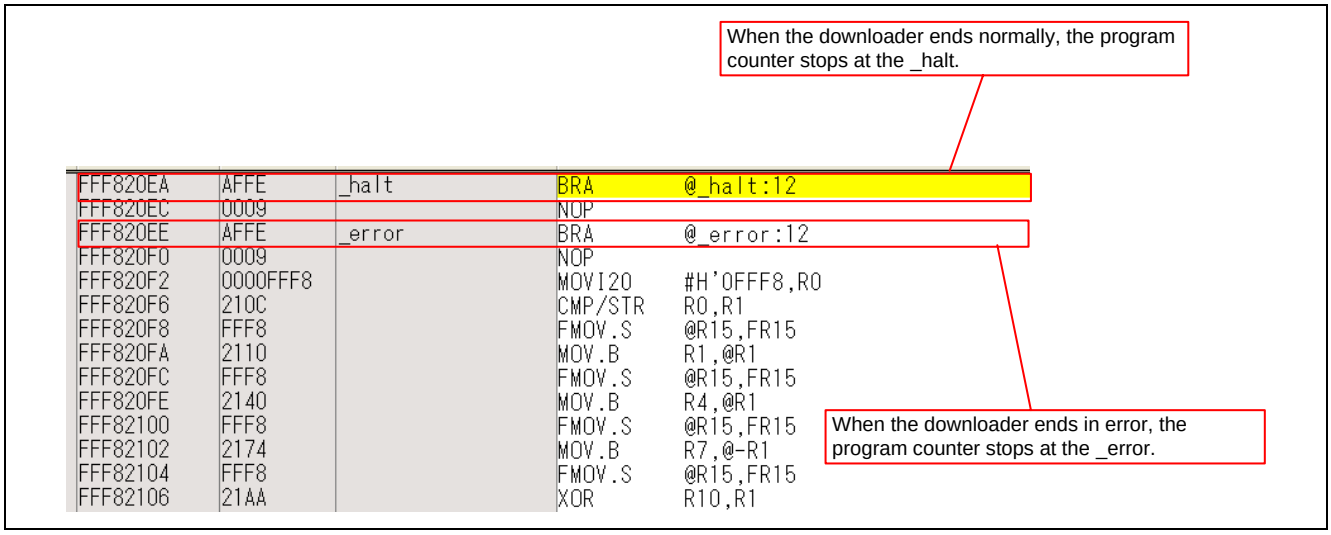

Figure 18 High-performance Embedded Workshop Window When the Downloader Ends

6. References

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

- Hardware Manual
  SH7266 Group, SH7267 Group User's Manual: Hardware Rev. 1.00
  The latest version can be downloaded from the Renesas Electronics website.

Website and Support

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

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

All trademarks and registered trademarks are the property of their respective owners.

Revision Record

Rev.	Date	Description	
		Page	Summary
1.00	Sep. 27, 11	—	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

SALES OFFICES

Renesas Electronics Corporation

http://www.renesas.com

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

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

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

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

Renesas Electronics Europe GmbH
Arcadiastrasse 10, 40472 Düsseldorf, Germany
Tel: +49-211-65030, 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.
13F, No. 363, Fu Shing North Road, Taipei, Taiwan
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

© 2011 Renesas Electronics Corporation. All rights reserved.

Colophon 1.1