SH7268/SH7269 Groups

E10A-USB Flash Memory Download Function (Download to the Serial Flash Memory)

R01AN0883EJ0100
Rev.1.00
Nov. 30, 2011

Abstract

E10A-USB emulator has the function to download a load module to the flash memory which requires a download program (hereinafter called FMTOOL) to access the flash memory.

This document describes how to download a load module to the serial flash memory applying the FMTOOL.

Target Device

SH7268/SH7269 Groups (hereinafter called as “SH7269”)

When using this application note with other Renesas MCUs, careful evaluation is recommended after making modifications to comply with the alternate MCU.

Contents

1. Specifications

Download the load module allocated in the SPI multi I/O bus space to the serial flash memory using the FMTOOL that support the serial flash memory. The FMTOOL uses the SPI multi I/O bus controller and allows the serial flash memory accessed when the flash memory is compliant with the multi I/O bus controller whose data bus width is 4 bits.

Table 1.1 lists the peripheral functions and their applications. Figure 1.1 shows the procedure of download using the FMTOOL.

Table 1.1 Peripheral Functions and Their Applications

Peripheral Function	Application
SPI multi I/O bus controller	Download to the serial flash memory
H-UDI	Connects the E10A-USB emulator

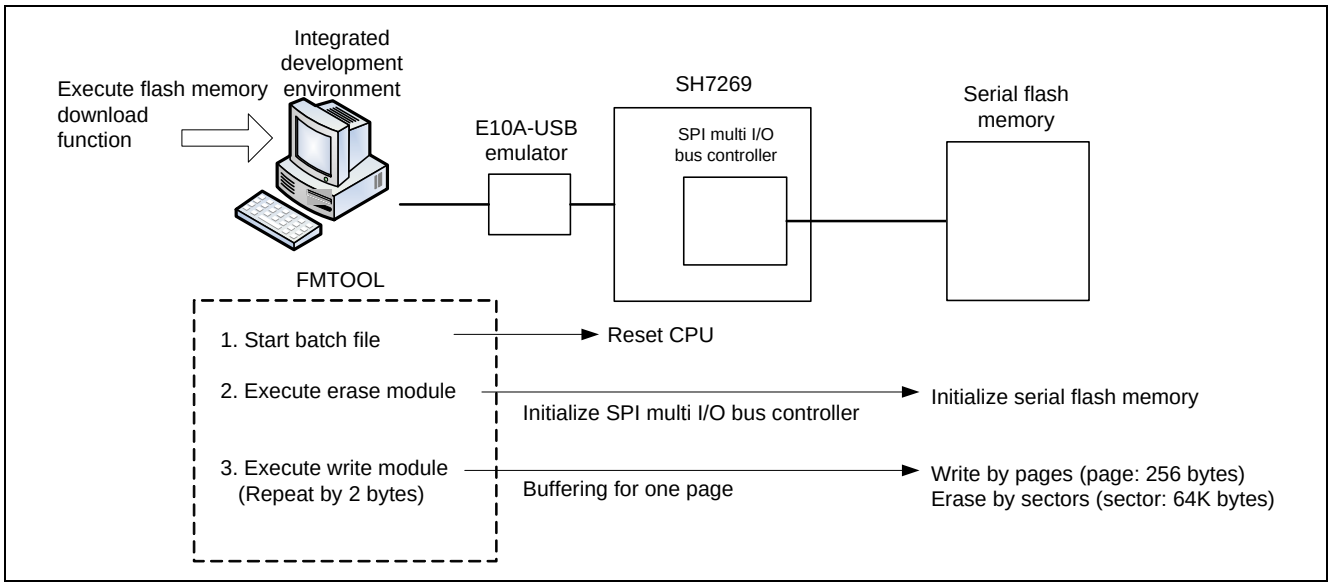


Figure 1.1 Procedure of Download Using FMTOOL

2. Operation Confirmation Conditions

The sample code accompanying this application note has been run and confirmed under the conditions below.

Table 2.1 Operation Confirmation Conditions

Item	Contents
MCU used	SH7269
Device used	Serial flash memory applicable to multi I/O bus manufacturer: Spansion Inc. model: S25FL032P0XMFI01
Operating frequency	CPU internal clock (Iφ): 266.67MHz Internal clock (Bφ): 133.33MHz Peripheral clock 1 (P1φ): 66.67MHz Peripheral clock 0 (P0φ): 33.33MHz
Operating voltage	Source power (I/O): 3.3V Source power (internal): 1.25V
Integrated development environment	Renesas Electronics Corporation High-performance Embedded Workshop Ver.4.07.00
C compiler	Renesas Electronics Corporation SuperH RISC engine FamilyC/C++ Compiler Package Ver.9.03 Release02
	Complier option -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 (with default setting in the integrated development environment)
Board used	R0K572690C000BR

3. Reference Application Note(s)

For additional information associated with this document, refer to the following application note(s).

- SH7268/SH7269 Group Boot From the Serial Flash Memory Using SPI Multi I/O Bus Controller (document No.: R01AN0663EJ)
- SH7268/SH7269 Groups SPI Multi I/O Bus Controller Serial Flash Memory Connection Sample Program (document No.: R01AN0671EJ)
- Flash Memory Download Program for the E10A-USB Emulator Application Note (document No.:REJ10J1221)

4. Peripheral Functions

This chapter provides supplementary information on the SPI multi I/O bus controller. Refer to the SH7268/SH7269 Groups: User's Manual: Hardware for basic information.

SPI multi I/O bus controller has two modes; the SPI operating mode, and external address space read mode. The former is used for erasing or writing the serial flash memory, and the latter is used for a fetch of the program written in the serial flash memory.

For details on the SPI operating mode setting procedure, refer to the application note, "SH7268/SH7269 Group SPI Multi I/O Bus Controller Serial Flash Memory Connection Sample Program (document No. R01AN0671EJ)".

5. Hardware

5.1 Hardware Configuration

Table 5.1 shows the used pins and their functions.

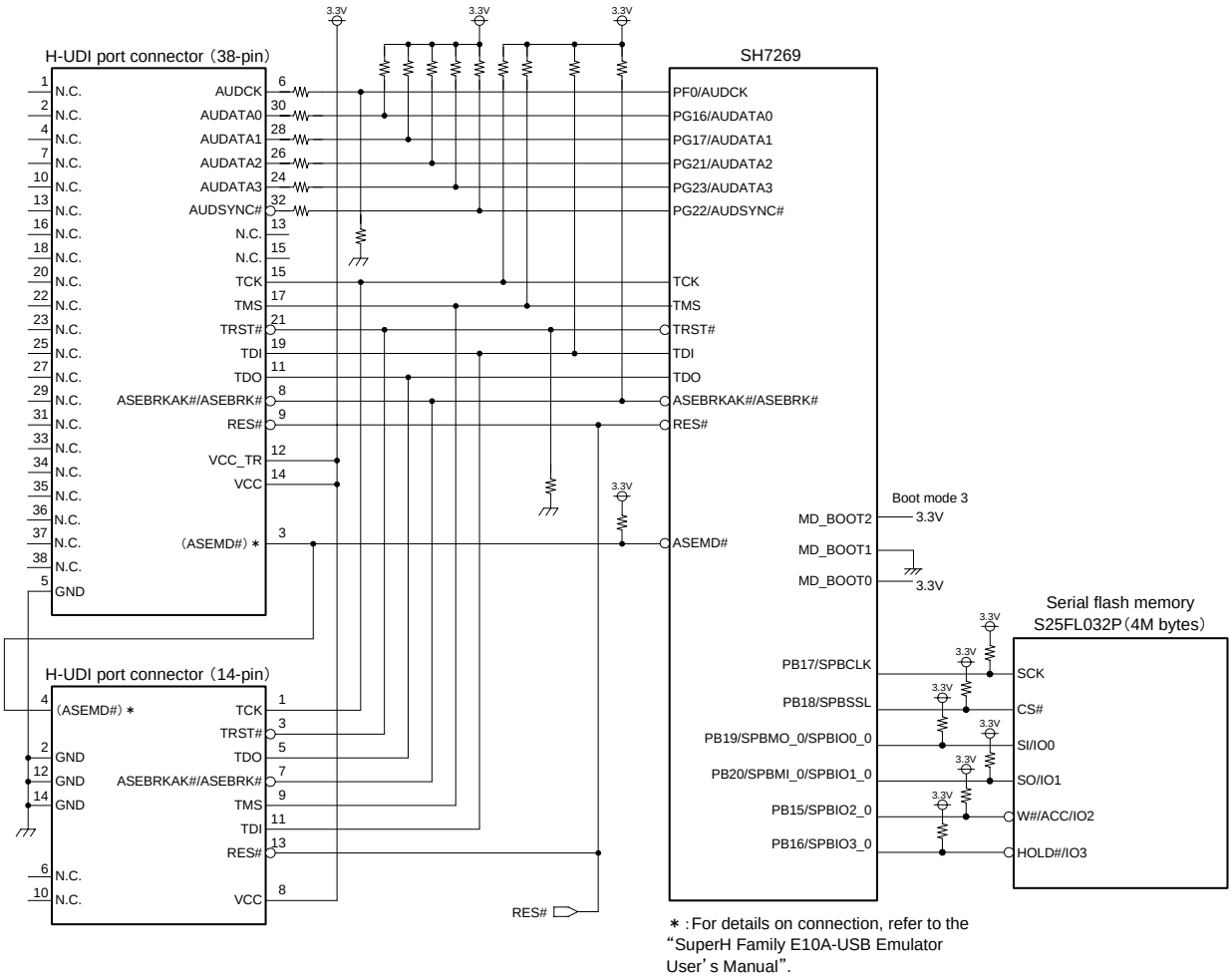
Table 5.1 Used Pins and Their Functions

Pin name	Input/output	Function
SPBCLK	Output	Clock output to the serial flash memory
SPBSSL	Output	Output device selection signal to the serial flash memory
SPBIO0_0	Output/output	Data input/output to/from the serial flash memory (bit 0)
SPBIO1_0	Output/output	Data input/output to/from the serial flash memory (bit 1)
SPBIO2_0	Output/output	Data input/output to/from the serial flash memory (bit 2)
SPBIO3_0	Output/output	Data input/output to/from the serial flash memory (bit 3)
MD_BOOT0	Input	Selection of boot mode (bit 0)
MD_BOOT1	Input	Selection of boot mode (bit 1)
MD_BOOT2	Input	Selection of boot mode (bit 2)
AUDCK	Output	Clock output to the E10A-USB emulator (38-pin)
AUDATA0	Output	Address output to the E10A-USB emulator (38-pin) (bit 0)
AUDATA1	Output	Address output to the E10A-USB emulator (38-pin) (bit 1)
AUDATA2	Output	Address output to the E10A-USB emulator (38-pin) (bit 2)
AUDATA3	Output	Address output to the E10A-USB emulator (38-pin) (bit 3)
AUDSYNC#	Output	Synchronous signal output to the E10A-USB emulator (38-pin)
TCK	Input	Clock input from the E10A-USB emulator
TMS	Input	Mode selection from the E10A-USB emulator
TRST#	Input	Reset input from the E10A-USB emulator
TDI	Input	Data input from the E10A-USB emulator
TDO	Output	Data output to the E10A-USB emulator
ASEBRKAK#/ASEBRK#	Output/output	Break request and response
RES#	Input	System reset signal
ASEMD#	Input	Selection of ASE mode

Note: “#” indicates a negative-true logic or an active low.

5.2 Reference Circuit

Figure 5.1 shows the connection with the serial flash memory.



Note: l "#" indicates a negative-true logic or an active low.

Figure 5.1 Connection Example

6. Software

6.1 Operation Overview

The FMTOOL consists of two programs; the erase module and the write module. The E10A-USB emulator writes data in the flash memory using them. For details on the erase module and the write module, refer to the section “6.22 Downloading to the Flash Memory Area” in the “Super H™ Family E10A-USB Emulator User’s Manual”.

6.1.1 Batch File

Before executing the FMTOOL, initialize the SH7269 by the reset command. For details on the reset command, refer to the manual listed in the integrated development environment on our web site.

6.1.2 Erase Module

Figure 6.1 shows the outline of the erase module which is executed in the high-speed on-chip RAM. Unlike the typical processing that assigns the chip erase processing to the flash memory in the erase module, in the sample code offered in this application note, the initial setting processing is assigned in the erase module. The initial setting processing initializes the SPI multi I/O bus controller, sets the serial flash memory mode, and cancels the protect setting.

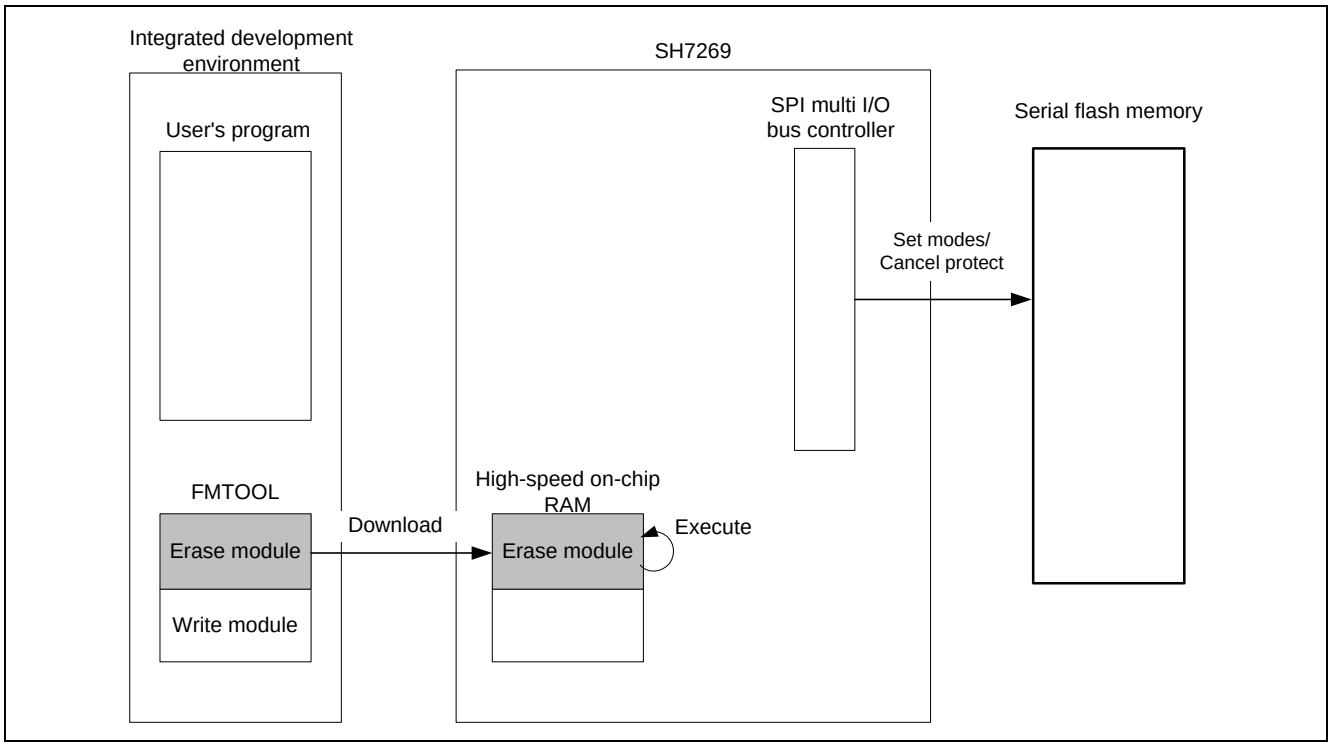


Figure 6.1 Erase Module Outline

6.1.3 Write Module

Figure 6.2 shows the outline of the write module which is executed in the high-speed on-chip RAM, and receives the word data as the argument from user program. The write module buffers the data and writes to the serial flash memory at the timing of page transit. When the write destination address specified by the argument is in the undeleted sector, writes after erasing the sector.

The write module converts the start address in the SPI multi I/O bus space (address H'1800 0000) to correspond to the start address in the serial flash memory (address H'0000 0000).

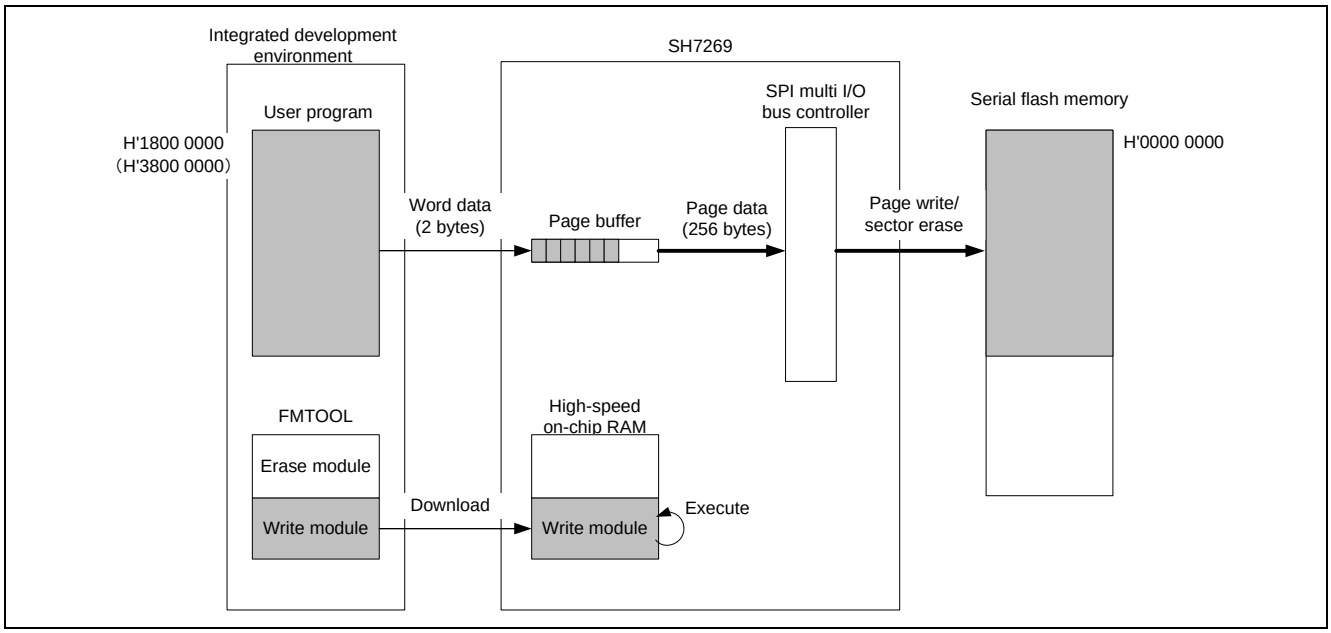


Figure 6.2 Write Module Outline

6.2 File Composition

Table 6.1 lists the file composition. Files generated by the integrated development environment should not be listed in this table.

Table 6.1 File Composition

File Name	Outline	Remarks
fmtool_entry.src	Entry module of FMTOOL	Entry of erase module and write module
fmtool_main.c	Main module of FMTOOL	
fmtool_cpg.c	Initialization for CPG	
fmtool_cpg.h	I/F definition of fmtool_cpg.c	
r_sf_spibsc.c	Serial flash memory processing	Supports multi I/O
r_sf_spibsc.h	I/F definition of r_sf_spibsc.c	
io_spibsc.c	SPI multi I/O bus controller control	
io_spibsc.h	I/F definition of io_spibsc.c	
sh7269_spibsc_fmtool.hdc	Batch file	Registers in the integral development environment
fmtool_debug.c	Debag code for FMTOOL	Used only during debugging

6.3 Constants

Figure 6.2 lists the constants used in the sample code.

Table 6.2 Constants Used in the Sample Code

Constant Name	Setting Value	Contents
SFLASH_DUAL	0	Does not allow a dual connection with serial flash memories
SPI_QUAD	1	Multi-I/O-bus-capable serial flash memory
SF_PAGE_SIZE	256	Page size (256 bytes)
PAGE_SIZE	SF_PAGE_SIZE	ditto
SF_SECTOR_SIZE	(64*1024)	Sector size (64 bytes)
SECTOR_SIZE	SF_SECTOR_SIZE	ditto
SR_Init	0x000000F0	Status register initial value
DEFAULT_VALUE	0xFFFFFFFF	Initial value of management data used by FMTOOL
SFLASH_ADDRESS_MASK	0xFC000000	Mask setting value to convert the SPI multi I/O bus space address to the serial flash memory address
TYPE_BYTE	0x4220	R5 parameter of write module (data access size : byte-size)
TYPE_WORD	0x5720	R5 parameter of write module (data access size: word-size)
TYPE_LONG	0x4C20	R5 parameter of write module (data access size: long-size)
ENABLE_DEBUG_MODE	1	FMTOOL debug code is valid

6.4 Structure/Union List

Figure 6.3 shows the structure/union used in the sample code.

```
/* ==== Structure for SPI multi I/O bus controller transfer control ==== */
typedef struct{
       /* ---- Setting value for SPI mode enable setting register (SMENR) ---- */
       uint32_t       cdb   :2;    /* command bit width */
       uint32_t       ocdb  :2;    /* optional command bit width*/
       uint32_t       adb   :2;    /* address bit width*/
       uint32_t       opdb  :2;    /* optional data bit width */
       uint32_t       spidb :2;    /* transfer data bit width */
       uint32_t       cde   :1;    /* command enable */
       uint32_t       ocde  :1;    /* optional command enable */
       uint32_t       ade   :4;    /* address enable */
       uint32_t       opde  :4;    /* optional data enable */
       uint32_t       spide :4;    /* transfer data enable */

       /* ---- setting value for SPI mode control register (SMCR) ---- */
       uint32_t       sslkp :1;    /* retains the SPBSSL signal level */
       uint32_t       spire :1;    /* data read enable */
       uint32_t       spiwe :1;    /* data write enable */
       uint32_t             :5;

       /* ---- setting value for SPI mode command setting register (SMCMR) ---- */
       uint8_t        cmd;         /* command */
       uint8_t        ocmd;        /* optional command */

       /* ---- setting value for SPI mode address setting register (SMADR) ---- */
       uint32_t       addr;

       /* ---- setting value for SPI mode optional setting register (SMOPR) ---- */
       uint8_t        opd[4];      /* optional data 0 to 3 */

       /* ---- setting value for SPI mode read data register (SMRDR0,SMRDR1) ---- */
       uint32_t       smrdr[2];

       /* ---- setting value for SPI mode write data register (SMWDR0,SMWDR1) ---- */
       uint32_t       smwdr[2];

} st_spibsc_spimd_reg_t;
```

Figure 6.3 Structure/Union Used in the Sample Code

6.5 Variables

Table 6.3 lists the global variables. Table 6.4 lists the static variables

Table 6.3 Global Variables

Type	Variable Name	Contents	Function Used
st_spibsc_spimd_reg_t	g_spibsc_spimd_reg	Setting data for SPI multi I/O bus controller	R_SF_SPIBSC_EraseChip R_SF_SPIBSC_EraseSector R_SF_SPIBSC_ByteProgram R_SF_SPIBSC_ByteRead read_data_quad read_status read_config write_enable write_status

Table 6.4 Static Variables

Type	Variable Name	Contents	Function Used
uint32_t	fmtool_pre_erase_sctno	Management information of erased sectors	fmtool_init, fmtool_write
uint32_t	fmtool_cur_page	Start address of buffering pages	fmtool_init, fmtool_write
uint16_t	fmtool_page_bufr[PAGE_SIZE / sizeof(uint16_t)]	Page buffer	fmtool_write

6.6 Functions

Table 6.5 lists the functions.

Table 6.5 Functions

Function Name	Outline
_ERASE_ENTRY	Entry processing for erase module
_WRITE_ENTRY	Entry processing for write module
fmtool_init	Main processing for erase module (initialization)
fmtool_write	Main processing for write module (erases/writes)
R_SF_SPIBSC_GetBsz	Serial flash memory operating function (detects data bus width)
R_SF_SPIBSC_SetBsz	Serial flash memory operating function (sets data bus width)
R_SF_SPIBSC_AllocateExspace	Serial flash memory operating function (sets the external address space read mode)
R_SF_SPIBSC_Init	Serial flash memory operating function (initializes SPI multi I/O bus controller and sets mode for serial flash memory)
R_SF_SPIBSC_CtrlProtect	Serial flash memory operating function (protect control)
R_SF_SPIBSC_EraseChip	Serial flash memory operating function (chip erase processing)
R_SF_SPIBSC_EraseSector	Serial flash memory operating function (sector erase processing)
R_SF_SPIBSC_ByteProgram	Serial flash memory operating function (writes)
R_SF_SPIBSC_ByteRead	Serial flash memory operating function (reads using SPI operation mode)
fmtool_debug	Debug function for FMTOOL

6.7 Function Specifications

The following tables list the sample code function specifications.

_ERASE_ENTRY

Outline	Entry processing for the erase module
Header	None
Declaration	_ERASE_ENTRY:
Description	Allocate this function in the address H'FFF8 2000 in the entry section of the erase module which is activated by the E10A-USB flash memory download function. This module executes fmtool_init function after setting the stack pointer.
Argument	R4 register : Access size (byte: H'4220, word: H'5720, long: H'4C00
Returned value	None
Remarks	Written in the assembly language

_WRITE_ENTRY

Outline	Entry processing for the write module
Header	None
Declaration	_WRITE_ENTRY:
Description	Allocate this function in the address H'FFF8 2100 in the entry section of the write module which is activated by the E10A-USB flash memory download function. This module executes fmtool_write function after setting the stack pointer.
Argument	R4 register : The address where write data are allocated R5 register : Access size (byte: H'4220, word: H'5720, long: H'4C00) R6 register : Write data R7 register : Verify flag (0: not verify, 1: verify)
Returned value	R0 register is 0: normal end R0 register is 1: error end
Remarks	Written in the assembly language

fmtool_init

Outline	Main processing for the erase module as initialization
Header	None
Declaration	void fmtool_init(void);
Description	Initializes the SPI multi I/O bus controller and the serial flash memory. This function is executed from the entry point of the FMTOOL (_ERASE_ENTRY).
Argument	None
Returned value	None
Remarks	

fmtool_write

Outline	Main processing for the write module (erases/writes)	
Header	None	
Declaration	int32_t fmtool_write(uint32_t addr, int32_t access_size, uint32_t write_data, int32_t v_flag);	
Description	Erases and writes in the serial flash memory which is accessed by sectors for erasing and by pages for writing. This function is executed from the entry point of the FMTOOL (_WRITE_ENTRY).	
Argument	First argument: addr	: The address where write data are allocated
	Second argument: size	: Access size (byte: H'4220, word: H'5720, long: H'4C00)
	Third argument: write_data	: Write data
	Forth argument: v_flag	: Verify flag (0: not verify, 1: verify) * unused
Returned value	0: normal end negative value: error end	
Remarks	Only word size is available for the access size.	

R_SF_SPIBSC_GetBsz

Outline	Serial flash memory operating function (detects data bus width)
Header	“r_sf_spibsc.h”
Declaration	int32_t R_SF_SPIBSC_GetBsz(void);
Description	Returns the data bus width (number of devices connected) to the SPI multi I/O bus controller.
Argument	None
Returned value	1: data bus width is 4 bits (device x 1) 2: data bus width is 8 bits (device x 2)
Remarks	

R_SF_SPIBSC_SetBsz

Outline	Serial flash memory operating function (sets data bus width)	
Header	“r_sf_spibsc.h”	
Declaration	void R_SF_SPIBSC_SetBsz(int32_t bsz);	
Description	Sets the data bus width (number of devices connected) to the SPI multi I/O bus controller.	
Argument	First argument: bsz	: data bus width (number of devices connected)
Returned value	None	
Remarks		

R_SF_SPIBSC_AllocateExspace

Outline	Serial flash memory operating function (sets the external address space read mode)
Header	“r_sf_spibsc.h”
Declaration	void R_SF_SPIBSC_AllocateExspace(void);
Description	Sets the external address space read mode to the SPI multi I/O bus controller.
Argument	None
Returned value	None
Remarks	

R_SF_SPIBSC_Init

Outline	Serial flash memory operating function (initializes the SPI multi I/O bus controller and sets the serial flash memory mode)
Header	“r_sf_spibsc.h”
Declaration	void R_SF_SPIBSC_Init(void);
Description	Initializes the basic part of the SPI multi I/O bus controller. Sets the serial flash memory on Quad operation mode.
Argument	None
Returned value	None
Remarks	

R_SF_SPIBSC_CtrlProtect

Outline	Serial flash memory operating function (protect control)
Header	“r_sf_spibsc.h”
Declaration	void R_SF_SPIBSC_CtrlProtect(en_sf_req_t req);
Description	Sets/cancels protect in the serial flash memory.
Argument	First argument: req : protect request (SF_REQ_PROTECT: sets protect SF_REQ_UNPROTECT: cancels protect)
Returned value	None
Remarks	

R_SF_SPIBSC_EraseChip

Outline	Serial flash memory operating function (chip erase)
Header	“r_sf_spibsc.h”
Declaration	void R_SF_SPIBSC_EraseChip(void);
Description	Carries a chip erase in the serial flash memory.
Argument	None
Returned value	None
Remarks	

R_SF_SPIBSC_EraseSector

Outline	Serial flash memory operating function (sector erase)	
Header	“r_sf_spibsc.h”	
Declaration	void R_SF_SPIBSC_EraseSector(int32_t sector_no);	
Description	Carries a sector erase in the serial flash memory.	
Argument	First argument: sector_no	: sector number to be erased
Returned value	None	
Remarks		

R_SF_SPIBSC_ByteProgram

Outline	Serial flash memory operating function (writes)	
Header	“r_sf_spibsc.h”	
Declaration	void R_SF_SPIBSC_ByteProgram(uint32_t addr, uint8_t * buf, int32_t size);	
Description	Writes data specified by the argument to the serial flash memory. When specifying 1 by the macro SPI_QUAD, uses the page program command (H'32) which supports Quad, and when specifying 0, uses the page program command (H'02) which supports Single.	
Argument	First argument: addr	: write address (the address in the serial flash memory)
	Second argument: buf	: write data (start address in the buffer)
	Third argument: size	: data byte count
Returned value	None	
Remarks		

R_SF_SPIBSC_ByteRead

Outline	Serial flash memory operating function (read)	
Header	“r_sf_spibsc.h”	
Declaration	void R_SF_SPIBSC_ByteRead(uint32_t addr, uint8_t * buf, int32_t size);	
Description	Reads the area in the serial flash memory specified by the argument to store in the buffer. When specified 1 by SPI_QUAD macro, uses the read command (H'6B) which is Quad processable, and when specified 0, uses the read command (H'0B) which is Single processable.	
Argument	First argument: addr	: read address (the address in the serial flash memory)
	Second argument: buf	: start address in the read buffer
	Third argument: size	: data byte count
Returned value	None	
Remarks		

fmtool_debug

Outline	Debug function for the FMTOOL
Header	None
Declaration	void fmtool_debug(void);
Description	Executes the erase module and write module to debug the FMTOOL. This function is activated by the integrated development environment. Set fmtool_debug to PC, and H'FFF90000 to SP in the register window. This function writes serial numbers in the initial two sectors in the address H'1800 0000 to H'1801 FFFF, but does not in the last page. The last page is allocated as the dummy data area unavailable for a write. The external address space read mode is set after writing in the serial flash memory so to check the written value in the memory window. SPI multi I/O bus space is inaccessible before setting the external address space read mode. Open the memory window after setting the external address space read mode.
Argument	None
Returned value	None
Remarks	This function executes the main processing as the erase module and write module alter the value of the stack pointer.

6.8 Flowchart

This section describes the procedure of major functions used in the sample code. For the serial flash memory operating functions, refer to the “SH7268/SH7269 Group SPI Multi I/O Bus Controller Serial Flash Memory Connection Sample Program (doc No. R01AN0671EJ)”

6.8.1 Erase Module

Figure 6.4 shows the procedure of the erase module.

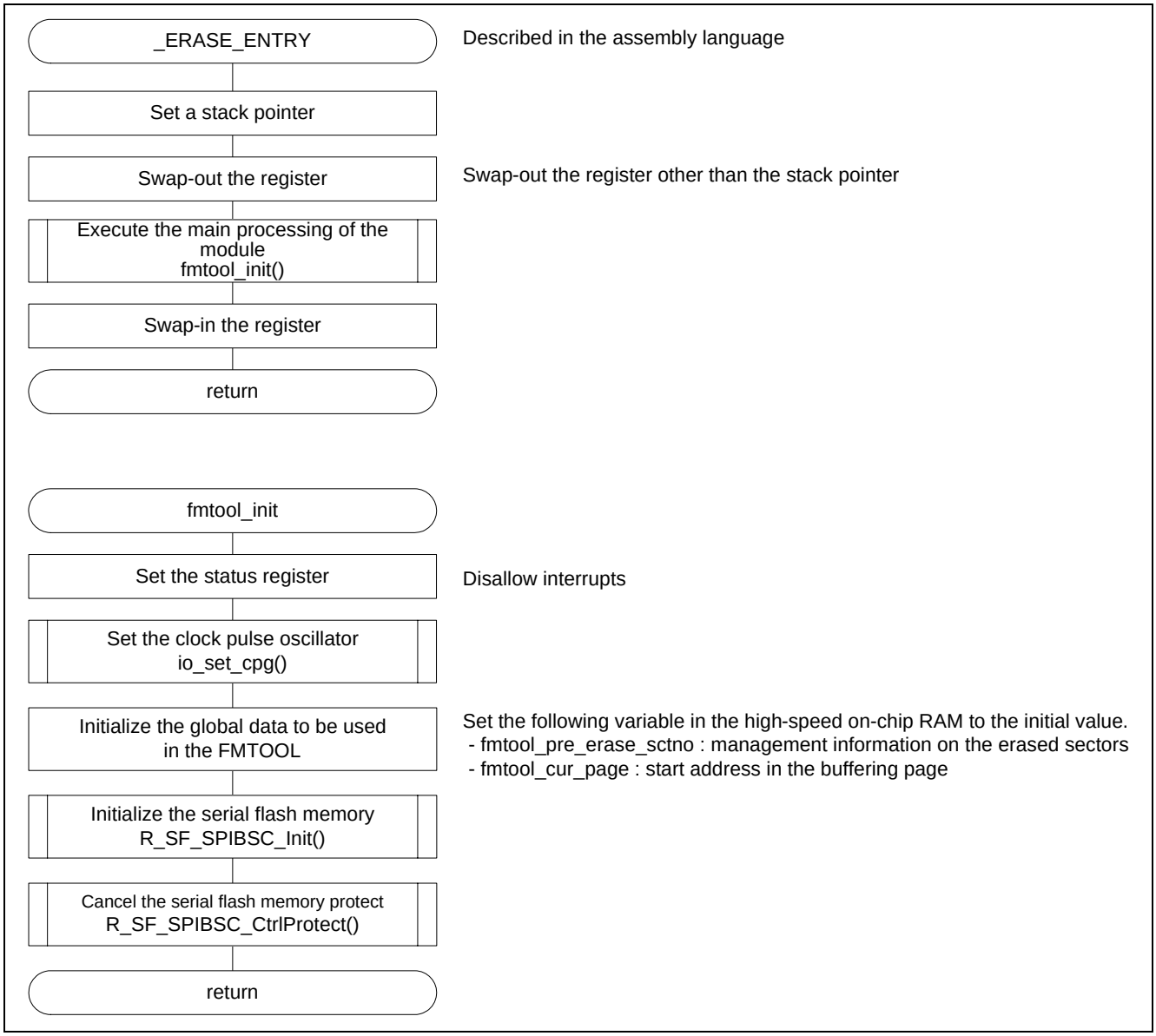


Figure 6.4 Erase Module

6.8.2 Write Module

Figure 6.5 shows the procedure of the write module.

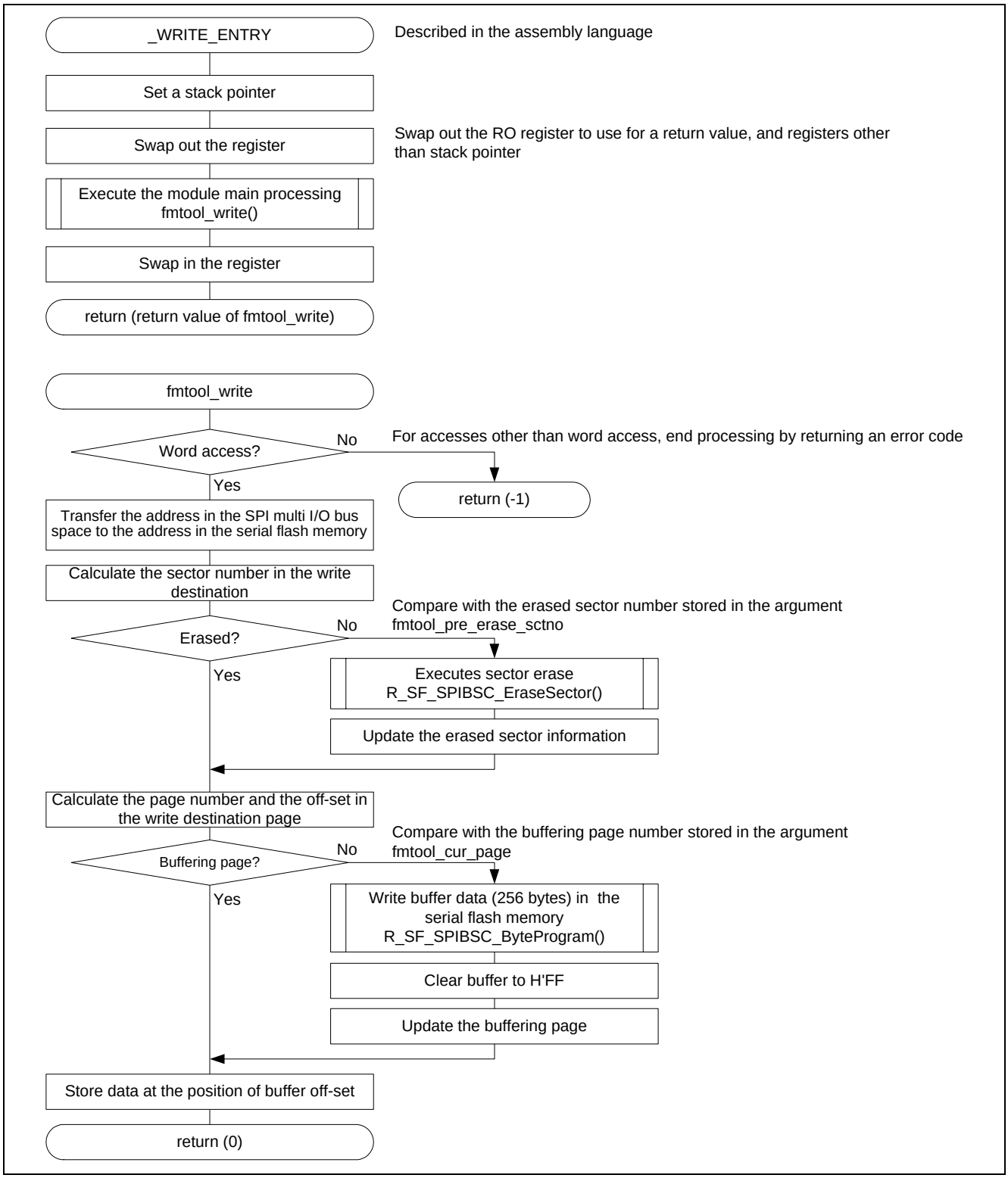


Figure 6.5 Write Module

6.9 Basic Precautions

6.9.1 Adding Dummy Data to the Load Module

The FMTOOL writes data by pages with buffering for the purpose of accelerating the write speed to the serial flash memory. A write to the serial flash memory is carried in the timing of specifying the address in the page different from the page under buffering. Therefore it is possible that the data for the last page may be remained in the buffer and not be written in the serial flash memory. Assign dummy data in the last page of the load module to avoid leaving the valid data in the buffer.

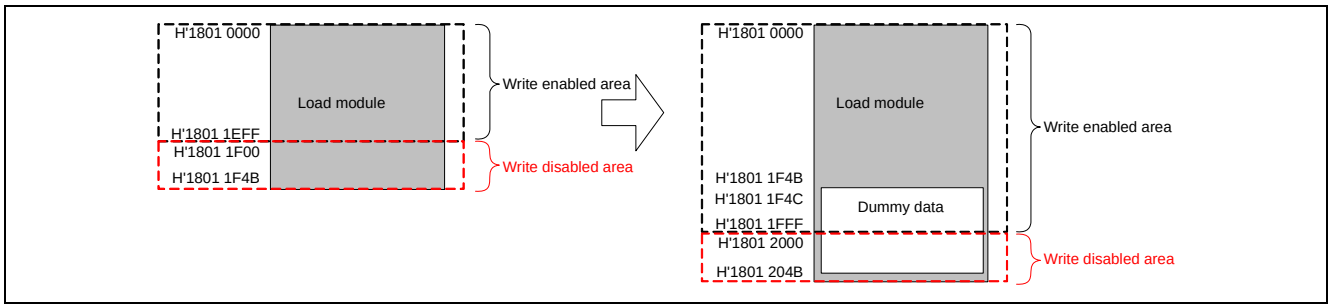


Figure 6.6 Write Disabled Area in Load Module

Figure 6.7 shows an example for adding dummy data to the section. Define the constant data of 256 bytes in the provided dummy section (CDUMMY_MODULE_END) and allocate it at the end of the ROM area.

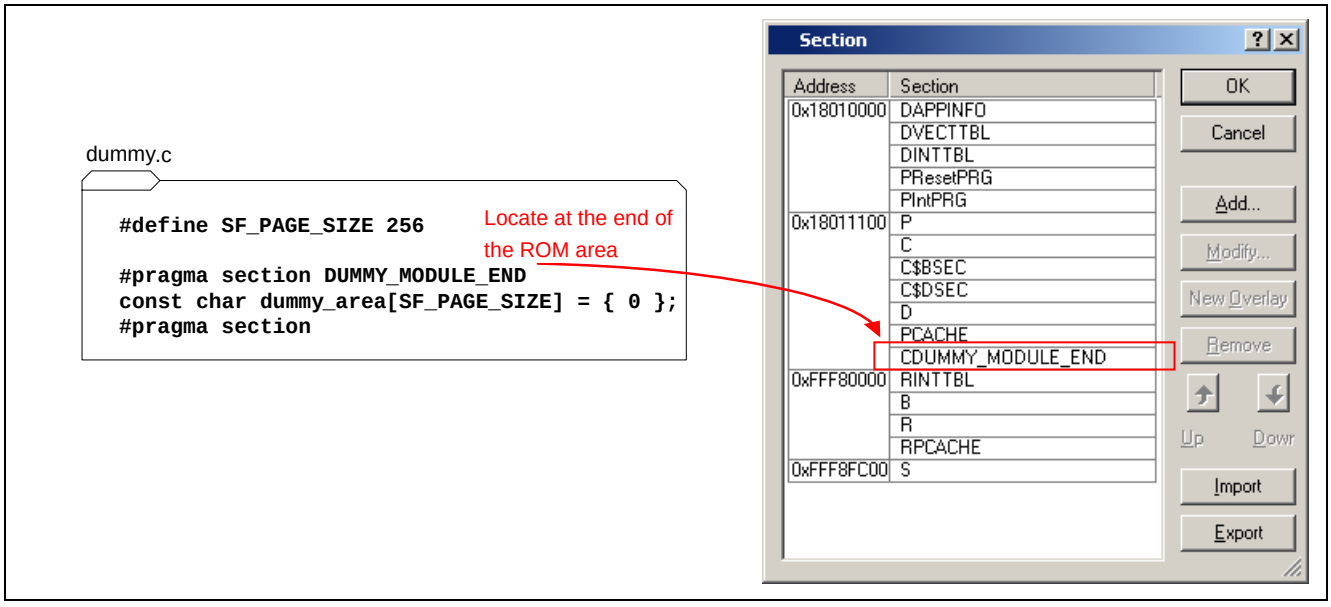


Figure 6.7 Example of Adding Dummy Data

6.9.2 Forbidding Sharing Sectors between the Load Modules

Figure 6.8 shows the operation under the assumption that two load module share one sector. Composing a user program to be downloaded by the FMTOOL of multiple load modules is enabled, although sharing one sector between the load modules is disabled. When downloading multiple data in one sector, the earlier downloaded data is deleted that may be followed by a false operation.

The mentioned load module area includes the dummy data area described in the section 6.9.1.

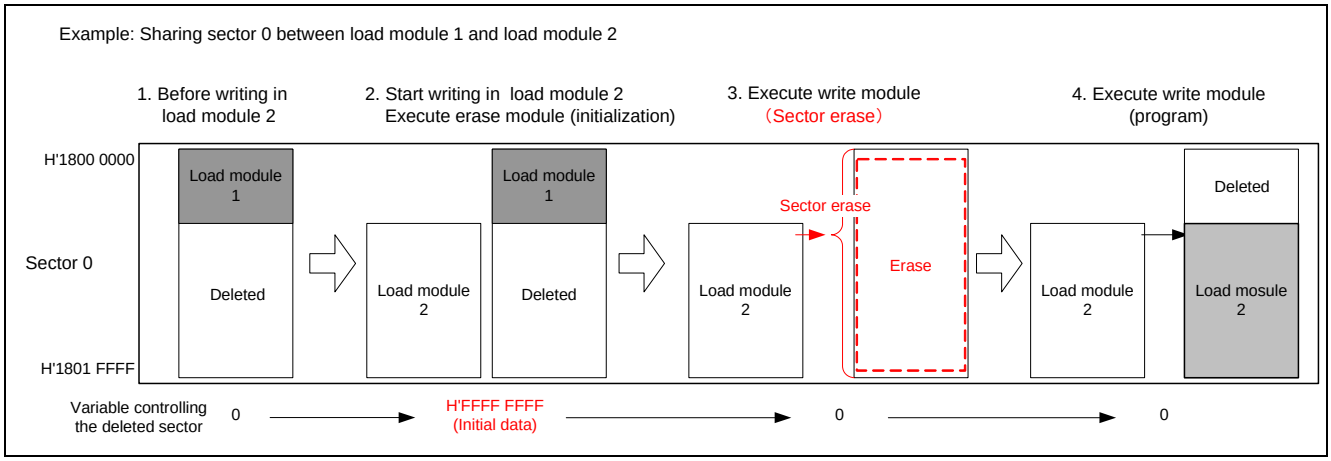


Figure 6.8 Operation when Sharing A Sector between Load Modules

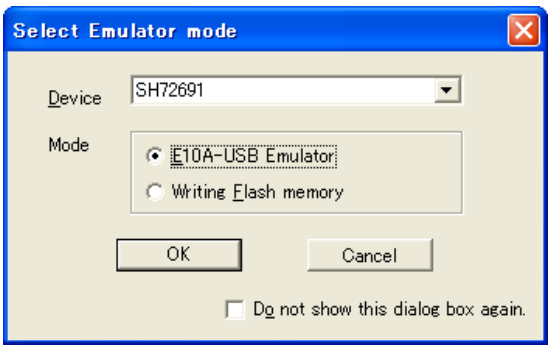
7. Application Example

7.1 Procedure of User Program Download

This section describes the procedure of downloading user programs to the serial flash memory using the created FMTOOL (sh7269_spibsc_fmtool.mot).

7.1.1 Prepare for the Download Environment

1. Connect user's system with the E10A-USB emulator conned to PC.
2. Start the High-performance Embedded Workshop to open the work space for user programs.
3. The Select Emulator mode box is open as shown in Figure 7.1.
   Select the CPU in use in the drop-down listbox for Device. Click the OK button.



Note: The shown window is an example agopting the SH72691

Figure 7.1 Device Select Dialog Box

4. The Connecting box is displayed and emulator connection starts.
   The reset signal request dialog box shown in Figure 7.2 is displayed.

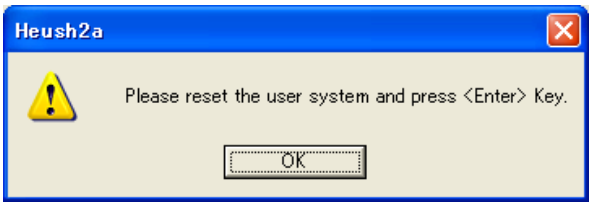


Figure 7.2 RESET Signal Request Dialog Box

5. Turn on the user's system.
   Having received the RESET signal from the user's system, click the OK button.
   When "connected" is displayed on the Output Window in the High –performance Embedded Workshop, the E10A-USB emulator successfully started.

7.1.2 Registering A Batch File

1. Select in the menu; [Debug] → [Debug Settings]
2. The window shown in Figure 7.3 opens.
3. Select "Before download modules" in the pull-down menu for the "Command batch file load timing".
4. Click the "Add" at "Command line batch processing" to add a batch file.
5. Click the OK button, and registration is completed.

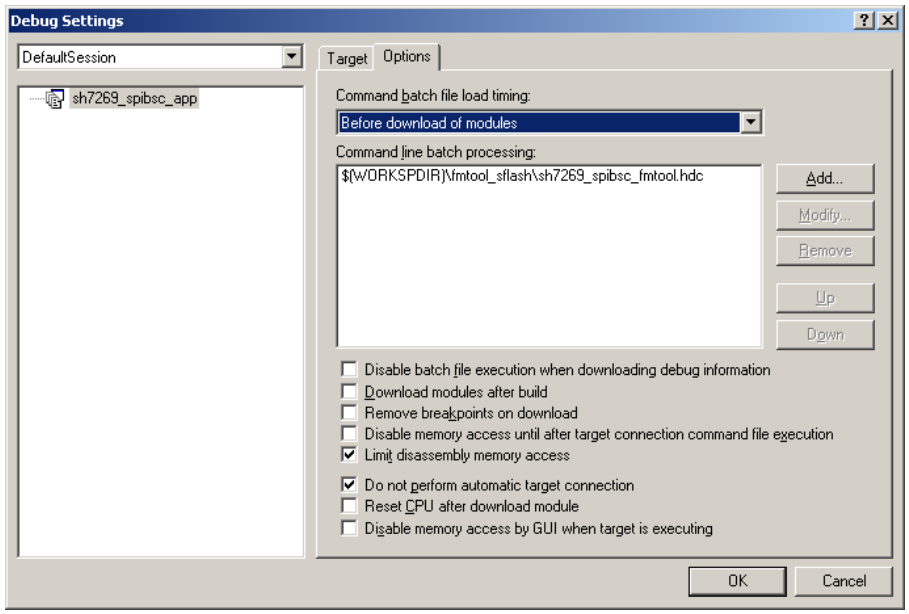


Figure 7.3 Window for Debug Setting

7.1.3 Setting Configuration Dialog Box

Figure 7.4 shows the “Configuration” dialog box for setting to download a user program to the external flash memory using the E10A-USB emulator.

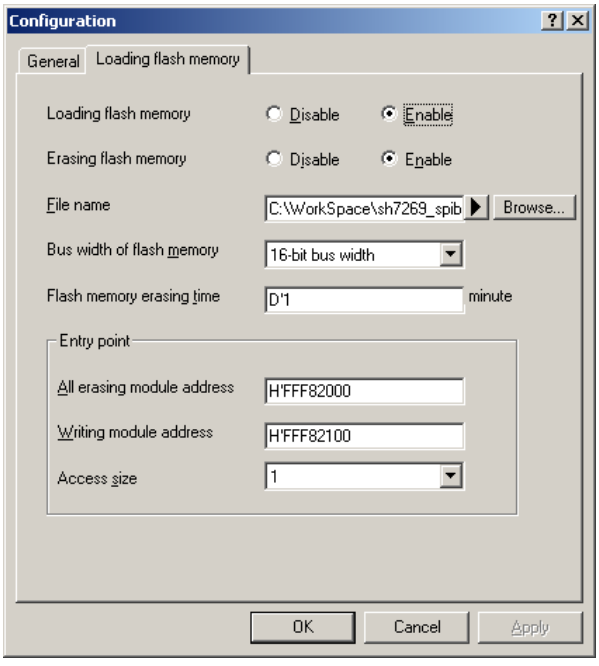


Figure 7.4 Configuration Dialog Box (in the page of Loading flash memory)

Table 7.1 lists the setting in the items in the configuration dialog box. Finish setting and click the OK button, configuration is completed.

Table 7.1 Setting Value in the Configuration Dialog Box

Item	Setting Value
Loading flash memory	Enable
Erasing flash memory	Enable
File Name	sh7269_spibsc_fmtool.mot
Bus width of flash memory	16-bit bus width
All erasing module address	Specify the start address of erase module
Writing module address	Specify the start address of write module

7.1.4 Adding User Program to Download Module

Open the debug setting window from the debug menu. Click "Add". In the download module window shown in Figure 7.5, add user programs to be loaded in the serial flash memory to the download module.

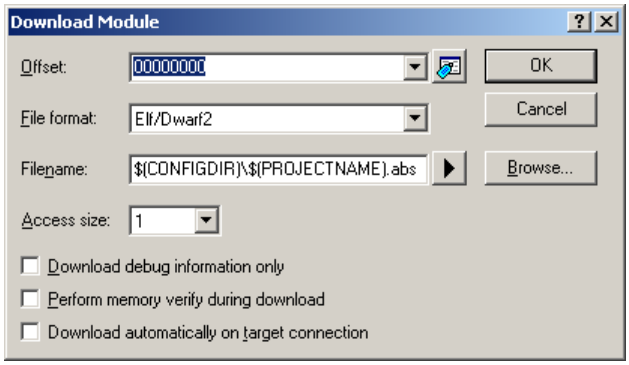


Figure 7.5 Download Module Window

7.1.5 Downloading User Programs

Using the download function shown in Figure 7.6, download the user programs.

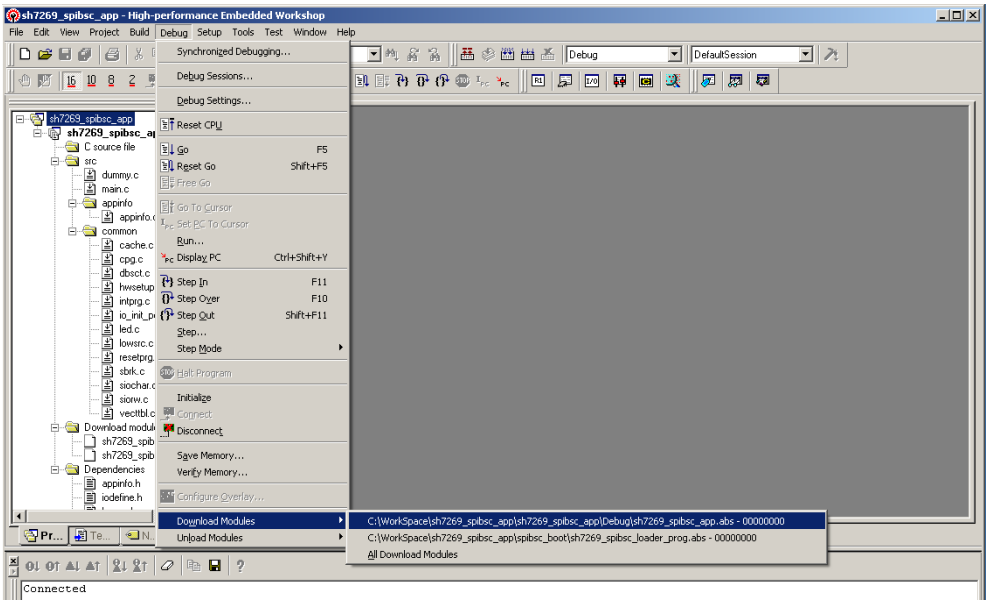


Figure 7.6 Downloading User Programs

7.2 Application to Serial Flash Boot

In this application note, the function to boot from the serial flash memory is called the “serial flash boot”. For details on the serial flash boot, refer to “SH7268/SH7269 Group Boot From the Serial Flash Memory Using SPI Multi I/O Bus Controller (document No.:R01AN0663EJ)”.

In this section, introduced the revised items for replacing the downloader as the flash write tool with the FMTOOL in the above mentioned application note.

7.2.1 Section Assignment

Figure 7.7 shows the section assignment when using the FMTOOL. Assign a loader program and application program caring the following points.

- Place sections in the SPI multi I/O bus space.
- Do not share one sector between different load modules. Example: placing the application program in the address H'1801 0000 or others not in H'1800 2000.
- Map the loader program in the address H'FFF8 0000 by using the optimizing linkage editor option (the section for mapping from ROM to RAM).

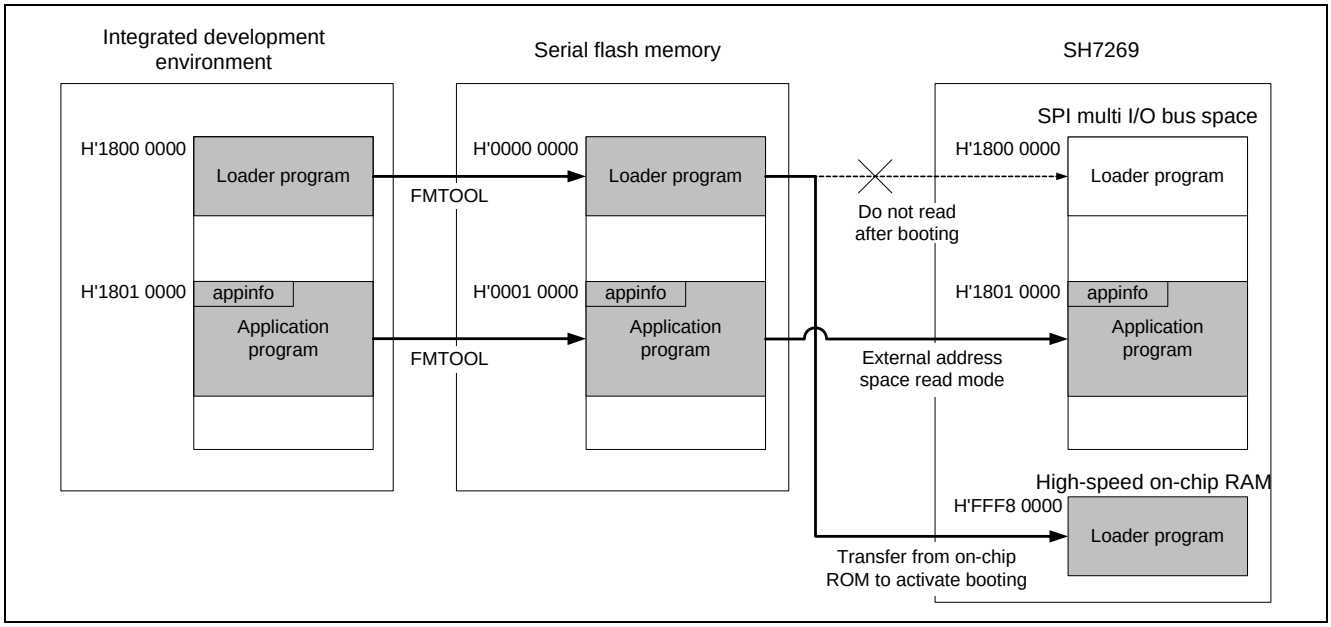


Figure 7.7 Section Assignment with FMTOOL

7.2.2 Adding Dummy Data

Make sure to add dummy data to the loader program and the application program as described in the section “6.9.1 Adding Dummy Data to the Load Module”.

7.2.3 Downloading the Load Module

The integrated development environment is also changed for downloading the load module. For the procedure to download, refer to the section “7.1 Procedure of User Program Download”.

7.3 Customizing FMTOOL

The sample code is dependent on the specification of the device in the serial flash memory. Customization of the program may be necessary when altering the device.

7.3.1 Device Specification Capable for Sample Code

Table 7.2 and Table 7.3 list the specification of the used device and the commands used in the sample code respectively.

Table 7.2 Specification of the Used Device

Item	Description
Manufacturer	Spansion Inc.
Model	S25FL032P0XMFI01
Capacity	4M bytes
Interface	SPI multi I/O bus (Single/Dual/Quad mode)
Access time	104MHz（Single mode）, 80MHz（Dual/Quad mode）
Sector structure	Uniform
Sector size	64K bytes
Page size	256 bytes

Table 7.3 Commands Used in the Sample Code

Item	Description
Erase command	H'D8 (64KB sector erase)
Program command	H'32（Quad page programming, H'02（Single page programming）

7.3.2 Contents of Customization

Table 7.4 lists the necessary customizations and their contents.

Table 7.4 Necessary Customization and the Contents

Item	Content
Inoperable Quad mode (for operation in Single)	Alter the macro SPI_QUAD setting value to 0
Improper sector size (not suitable for 64K-byte sector erase)	For the Uniform type sector structure, alter the setting value of macro SF_SECTOR_SIZE to the new sector size. Change the sector erase command used in R_SF_SPIBSC_EraseChip function to the command that supports the new sector size. For the Top or Bottom type structure, the algorithm to discriminate sector number in fmtool_write function should also be altered.
Different procedure for device initialization	Customization is needed for the serial flash memory operation function and the SPI multi I/O bus controller control function. For details, refer to the sample code.
The command in Table 7.3 is unusable	
Different electric characteristics	

Note: The FMTOOL is flash memory specification dependent. Therefore the items in Table 7.4 do not cover all the cases. Check the data sheet and modify the FMTOOL according to the specification in it.

7.4 Procedure of FMTOOL Debug

For the FMTOOL, the common procedure of debug is not applicable as it is booted from the integral development environment. In this section, the procedure of debug for the FMTOOL is described.

7.4.1 Implementing Debug Code

Carry a build with ENABLE_DEBUG_MODE set to 1. The debug function *fmtool_debug* in the load module of the FMTOOL is implemented. For details on the specification of he debug function, refer to the section “6.7 Function Specifications”.

7.4.2 Preparing for the Debug Environment

Connect the E10A-USB emulator to the user’s system in the same procedure shown in the section “7.1.1 Prepare for the Download Environment”.

7.4.3 Downloading FMTOOL

1. Reset the CPU
   Choose [Debug] → [Reset CPU] in the menu.
2. Download the FMTOOL in the high-speed on-chip RAM.
   Follow the procedure shown in the section “7.1.5 Downloading User Programs” to download sh7269_spibsc_fmtool.abs in the high-speed on-chip RAM. Download the ELF_DWARF file (.abs), not the S format file (.mot) to download the debug information together.

7.4.4 Debug of FMTOOL

1 Set the program counter at the beginning of the debug program.
   Double-click the “PC” in the register window to input “fmtool_debug”.
2 Set a stack pointer.
   Double-click the “SP” in the register window to input “H'FFF90000”.

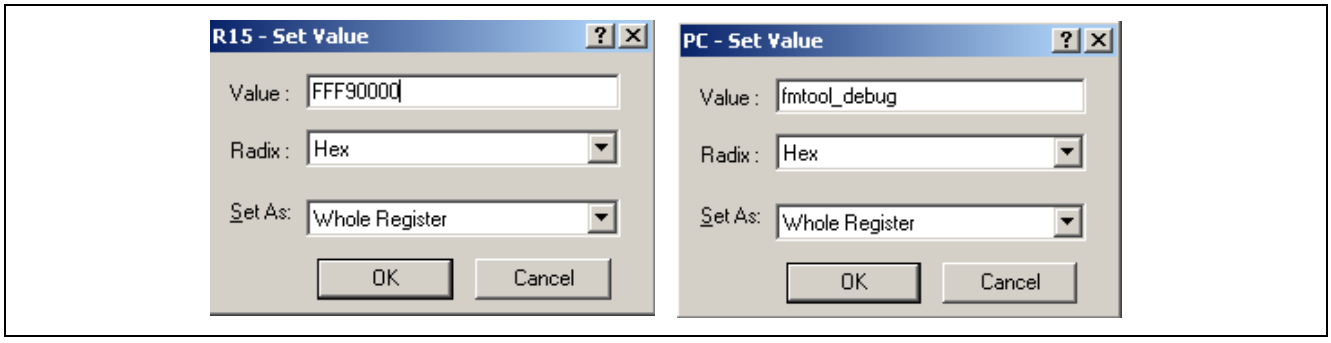


Figure 7.8 Initialization for PC and SP

3 Execute the debug code.
   Execute the program by choosing [Debug] → [Go] in the menu. Debug the program by stepwise execution or using break point as appropriate.
4 Check the operation result.
   The debug code writes the serial numbers in the beginning two sectors (address H'1800 0000 to H'1801 FFFF) except the final page which is reserved as the unwritable dummy data area.
   The external address space read mode is set after transferring data in the serial flash memory so that the written value can be checked in the memory window. Note that before setting the external address space read mode, the SPI multi I/O bus space is inaccessible. Open the memory window after setting the external address space read mode.

8. Sample Code

Sample code can be downloaded from the Renesas Electronics website.

9. Reference Documents

Hardware Manual

SH7268 Group, SH7269 Group User's Manual: Hardware Rev.1.00
The latest version can be downloaded from the Renesas Electronics website.

Technical Update/Technical News

The latest information can be downloaded from the Renesas Electronics website.

Development Tool Manual

SuperH RISC Engine Family C/C++ Compiler, Assembler, Optimizing Linkage Editor
Complier Package V.9.04 User's Manual Rev.1.01
The latest version can be downloaded from the Renesas Electronics website.

SuperH Family E10A-USB Emulator User's Manual Rev.9.00
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/contact/

Revision History	SH7268/SH7269 Group Application Note for E10A-USB Flash Memory Download Function (Download to Serial Flash Memory)

Rev.	Date	Description	
		Page	Summary
1.00	Nov. 30, 2011	—	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

Colophon 1.1