RZ/T1 Group

Serial Flash Sample Program (SPIBSC)

R01AN3010EJ0140
Rev.1.40
Jun. 07, 2018

Summary

This application note describes how to make settings for the SPI multi-I/O bus controller of an RZ/T1 group microcontroller.

Restrictions

The following restriction applies to the sample program.

(1) Since this program is for the area in the serial flash memory, the other areas cannot be programmed or erased by this program.

Applicable Devices

RZ/T1 Group

When applying the program covered in this application note to another microcontroller, modify the program according to the specifications for the target microcontroller and extensively evaluate the modified program.

Table of Contents

1. Specifications

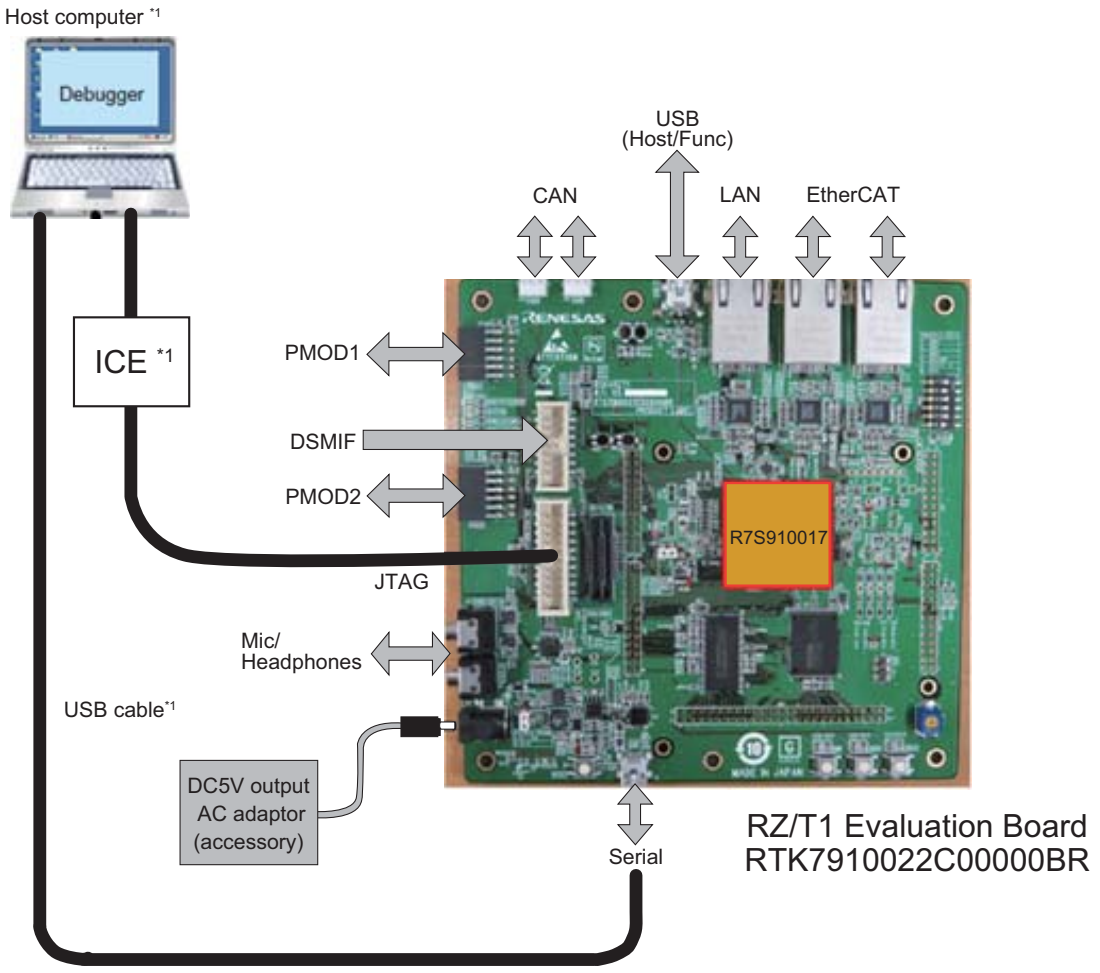
The SPI multi-I/O bus controller (SPIBSC) is a bus controller for the output of control signals to a serial flash memory connected to the SPI multi-I/O bus space (SPIBSC space). Making settings for the SPIBSC allows direct reading of the serial flash memory connected to the SPIBSC space and data transfer in SPI mode.

This application note describes how to make settings for the SPIBSC to read data directly from the SPIBSC space and how, in SPI mode, to make the settings for the registers of the serial flash memory which are required for access to the serial flash memory with a width of 4 bits.

Table 1.1 lists the peripheral modules used and their applications.

Table 1.1 Peripheral Modules and Their Applications

Peripheral Module	Application
SPI multi-I/O bus controller (SPIBSC)	This is used to generate signals for use in access to the serial flash memory connected to the external address space (SPI multi-I/O bus space).



Note 1. Indicates the device that the user needs to prepare.

Figure 1.1 Operating Environment

2. Operating Environment

The sample program covered in this application note is for the environment below.

Table 2.1 Operating Environment

Item	Description
MCU used	RZ/T1 Group
Operating frequency	CPU clock (CPUCLK): 450 MHz
Operating voltage	Power supply voltage (I/O): 3.3 V
Integrated development environment	• Embedded Workbench® for Arm Version 8.20.2 from IAR Systems • Arm® Integrated Development Environment Arm Development Studio 5 (DS-5™) Version 5.26.2 • Renesas e2studio Version: 6.1.0
Operating mode	SPI boot mode (serial flash memory) 16-bit bus boot mode (NOR flash memory)
Board used	RZ/T1 evaluation board (RTK7910022C00000BR)
Devices used (functions to be used on the board)	Serial flash memory allocated to the SPI multi-I/O bus space (1- or 4-bit bus width) • Manufacturer: Macronix International Co., Ltd. • Product type name: MX25L51245G

3. Related Application Notes

The application note related to the descriptions in this application note is listed below. Also consult the following document along with this application note.

- RZ/T1 Group Initial Settings (R01AN2554EJ)

Note: For registers not covered in this application note, the values set in the RZ/T1 Group Initial Settings application note are used without change.

4. Description of Hardware

4.1 List of Pins

Table 4.1 lists the pins used and their functions.

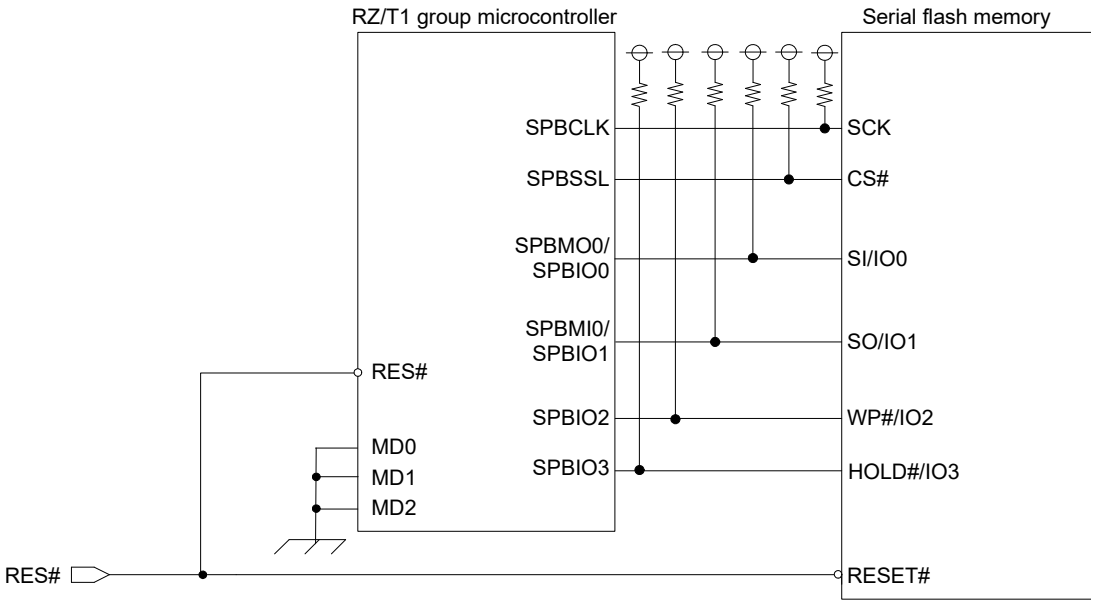
Table 4.1 Pins Used and Their Functions

Pin Name	I/O	Description
SPBCLK	Output	Clock output
SPBSSL	Output	Slave select
SPBMO/SPBIO0	I/O	Master send data: data 0
SPBMI/SPBIO1	I/O	Master input data: data 1
SPBIO2	I/O	Data 2
SPBIO3	I/O	Data 3
MD2, MD1, MD0	Input	Selection of boot mode (set to SPI boot mode) MD_BOOT2: “L” MD_BOOT1: “L” MD_BOOT0: “L”
RES#	Input	System reset signal

Note: The symbol # represents negative logic (or active low).

4.2 Reference Circuit

Figure 4.1 is a connection example.



Note: The symbol # represents negative logic (or active low).

Figure 4.1 Connection Example

5. Description of Software

5.1 Outline of Operation

This section presents an outline of operation of the sample program.

When an RZ/T1 group microcontroller is booted up in SPI boot mode, it transfers the loader program in the serial flash memory connected to the SPIBSC space to the “B” tightly-coupled memory (BTCM) of the RZ/T1 group microcontroller and branches to the transferred loader program. Once execution branches to the loader program, the microcontroller can read data directly from the SPIBSC space. However, since the settings made for the SPIBSC are common to serial flash memory in general, you must make optimum settings for the SPIBSC to suit the given serial flash memory. For example, to read data at a high speed, you will need to make settings such as the setting for the read command to use 4 data pins. This sample program makes SPIBSC settings optimized for use with a Macronix serial flash memory (product type name: MX25L51245G).9

5.1.1 Project Settings

Project settings for use in the EWARM, DS-5, or e2studio as the development environment are described in the RZ/T1 Group Initial Settings application note.

Note: Since the sample program of the SPI boot version for the EWARM environment (RZ_T1_serialflash_serial_boot.eww) does not update the contents of rodata section when the serial flash memory is modified, the rodata section is placed in the “A” tightly-coupled memory (ATCM). When placing, "RZ_T1_init_serial_boot.icf" is changed as shown below (bold indicates added text, whereas double-strikethrough indicates deleted text).

```
define block USER_PRG_RBLOCK { ro code, section .rodata_init };
define block USER_PRG_WBLOCK { rw code, section .rodata };
        :
initialize manually { ro code object loader_init.o,
                      ro code object loader_init2.o,
                      ro code object r_atcm_init.o,
                      ro code object r_cpg.o,
                      ro code object r_ram_init.o,
                      ro code object r_mpc.o,
                      ro code object bus_init_serial_boot.o,
                      ro code object r_reset.o,
                      ro code object vector.o,
                      ro code object spibsc_flash_api.o,
                      ro code object spibsc_flash_userdef.o,
                      ro code object spibsc_ioset_api.o,
                      ro code object spibsc_ioset_drv.o,
                      ro code object spibsc_ioset_userdef.o,
                      ro code,
                      section .rodata
                    };
        :
place in ROM_region { block USER_PRG_RBLOCK~~, section .rodata~~ };
```

5.1.2 Preparing to Run the Program

This sample program handles processing for transfer to and from the PC and the following describes the preparation for running the program.

(1) Start the terminal software on the host PC and make settings for the serial port as follows (the following is the case for Tera Term on COM3).

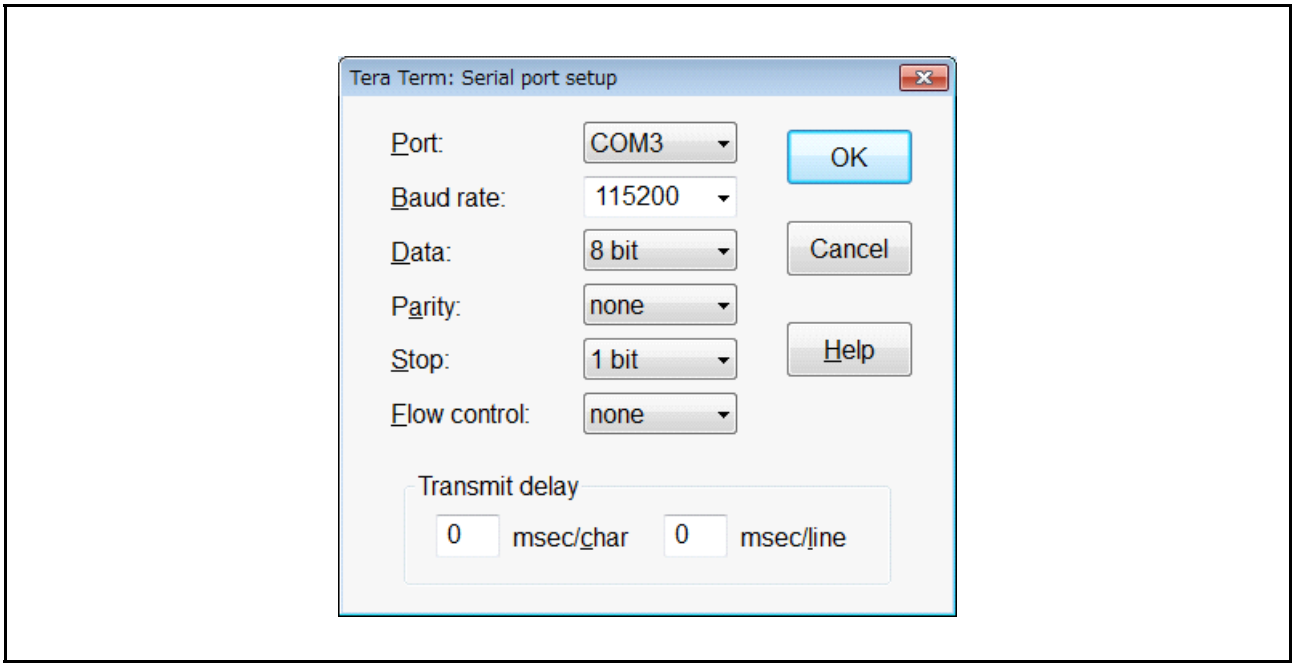


Figure 5.1 Settings for the Serial Port

(2) When the sample program is run and ready to handle transfer, the data received form the sample program are displayed in the terminal software as shown below.

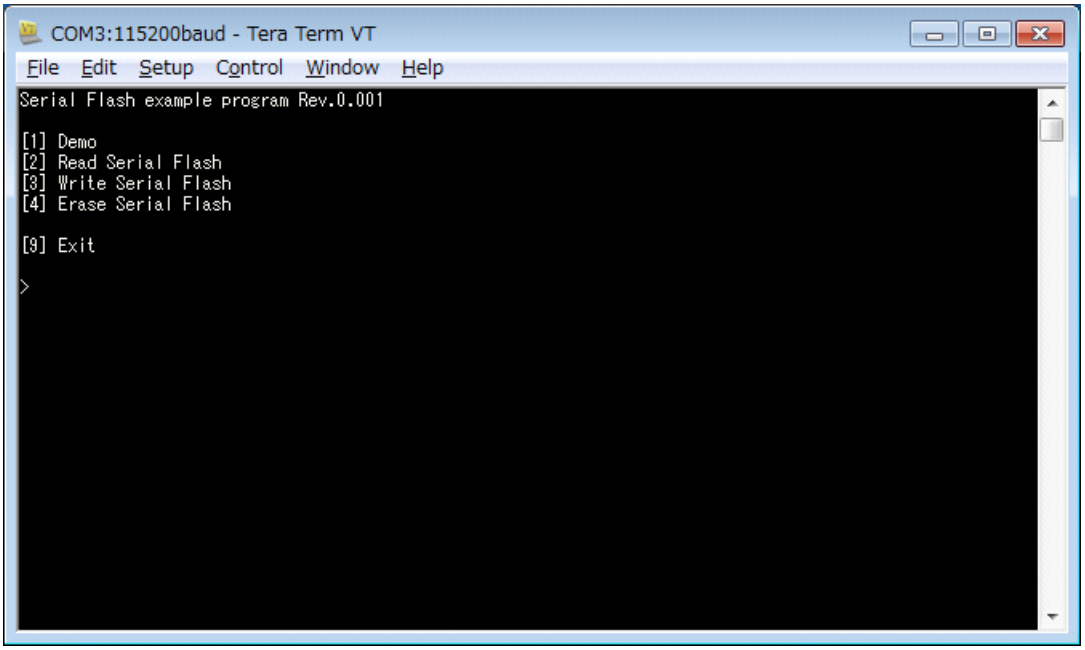


Figure 5.2 Display in the Terminal Software after the Sample Program is Run

5.2 Fixed-Width Integers

Table 5.1 lists fixed-width integers used in the sample program.

Table 5.1 Fixed-Width Integers Used in the Sample Program

Symbol	Description
char_t	8-bit signed integer
int_t	Signed integer
int32_t	32-bit signed integer
uint8_t	8-bit unsigned integer
uint16_t	16-bit unsigned integer
uint32_t	32-bit unsigned integer

5.3 Structures and Unions

Table 5.2 to Table 5.8 list the structures used in the sample program.

Table 5.2 Structure of SPIBSC External Address Read Settings (st_spibsc_cfg_t) (1)

Member		Description
uint8_t	udef_cmd	Read command • Sets the read command output to the serial flash memory when converting read operations for the SPI multi-I/O bus space to SPI communications. • The value of this member is set in the CMD[7:0] bits of the data read command setting register (DRCMR).
uint8_t	udef_cmd_width	Read command bit width • Sets the bit width for issuing read commands. • Available settings: SPIBSC_1BIT: 1-bit width SPIBSC_4BIT: 4-bit width • The value of this member is set in the CDB[1:0] bits of the data read enable register (DRENR).
uint8_t uint8_t uint8_t uint8_t	udef_opd3 udef_opd2 udef_opd1 udef_opd0	Optional data • Set the optional data output to the serial flash memory when converting read operations for the SPI multi-I/O bus space to SPI communications. • The values of these members are set in the OPD3[7:0], OPD2[7:0], OPD1[7:0], and OPD0[7:0] bits of the data read option setting register (DROPR).
uint8_t	udef_opd_enable	Optional data enable • Selects whether the optional data is to be issued. • Available settings: SPIBSC_OUTPUT_DISABLE: No data are output. SPIBSC_OUTPUT_OPD_3: OPD3 is output. SPIBSC_OUTPUT_OPD_32: OPD3 and OPD2 are output. SPIBSC_OUTPUT_OPD_321: OPD3, OPD2, and OPD1 are output. SPIBSC_OUTPUT_OPD_3210: OPD3, OPD2, OPD1, and OPD0 are output. • The value of this member is set in the OPDE[3:0] bits of the data read enable register (DRENR).
uint8_t	udef_opd_width	Optional data bit width • Sets the bit width for issuing the optional data. • Available settings: SPIBSC_1BIT: 1-bit width SPIBSC_4BIT: 4-bit width • The value of this member is set in the OPDB[1:0] bits of the data read enable register (DRENR).

Table 5.3 Structure of SPIBSC External Address Read Settings (st_spibsc_cfg_t) (2)

Member	Description
uint8_t udef_dmycyc_num	Number of dummy cycles • Sets the number of dummy cycles output to the serial flash memory when converting read operations for the SPI multi-I/O bus space to SPI communications. • Available settings: SPIBSC_DUMMY_1CYC: 1 cycle SPIBSC_DUMMY_2CYC: 2 cycles SPIBSC_DUMMY_3CYC: 3 cycles SPIBSC_DUMMY_4CYC: 4 cycles SPIBSC_DUMMY_5CYC: 5 cycles SPIBSC_DUMMY_6CYC: 6 cycles SPIBSC_DUMMY_7CYC: 7 cycles SPIBSC_DUMMY_8CYC: 8 cycles • The value of this member is set in the DMCYC[2:0] bits of the data read dummy cycle setting register (DRDMCR).
uint8_t udef_dmycyc_enable	Dummy cycle enable • Selects whether dummy cycles are to be inserted. • Available settings: SPIBSC_DUMMY_CYC_DISABLE: Dummy cycles are not inserted. SPIBSC_DUMMY_CYC_ENABLE: Dummy cycles are inserted. • The value of this member is set in the DME bit of the data read enable register (DRENR).
uint8_t udef_dmycyc_width	Dummy cycle bit width • Sets the bit width for issuing dummy cycles. • Available settings: SPIBSC_1BIT: 1-bit width SPIBSC_4BIT: 4-bit width • The value of this member is set in the DMDB[1:0] bits of the data read dummy cycle setting register (DRDMCR).
uint8_t udef_data_width	Data read bit width • Sets the bit width for reading data from the serial flash memory when converting read operations for the SPI multi-I/O bus space to SPI communications. • Available settings: SPIBSC_1BIT: 1-bit width SPIBSC_4BIT: 4-bit width • The value of this member is set in the DRDB[1:0] bits of the data read enable register (DRENR).

Table 5.4 Structure of SPIBSC External Address Read Settings (st_spibsc_cfg_t) (3)

Member	Description
uint8_t udef_spbr	Bit rate • Sets the bit rate of the serial clock (SPBCLK) output to the serial flash memory when converting read operations for the SPI multi-I/O bus space to SPI communications. • Available settings: Make settings to match the bit-rate division setting (udef_brdv). • The value of this member is set in the SPBR[7:0] bits of the bit rate setting register (SPBCR).
uint8_t udef_brdv	Bit-rate division setting • Sets the bit rate of the serial clock (SPBCLK) output to the serial flash memory when converting read operations for the SPI multi-I/O bus space to SPI communications. • Available settings: Make settings to match the bit-rate division setting (udef_brdv). • The value of this member is set in the BRDV[1:0] bits of the bit-rate setting register (SPBCR).
uint8_t udef_addr_width	Address bit width • Sets the width in bits of the address line or lines for output to the serial flash memory when converting read operations for the SPI multi-I/O bus space to SPI communications. • Available settings: SPIBSC_1BIT: 1-bit width SPIBSC_4BIT: 4-bit width • The value of this member is set in the ADB[1:0] bits of the data read enable register (DRENR).
uint8_t udef_addr_mode	Address enable • Sets the address for output to the serial flash memory when converting read operations for the SPI multi-I/O bus space to SPI communications. • Available settings: SPIBSC_OUTPUT_ADDR_24: 24-bit address output SPIBSC_OUTPUT_ADDR_32: 32-bit address output • The value of this member is set in the ADE[3:0] bits of the data read enable register (DRENR).

Table 5.5 Structure of SPIBSC SPI Mode Settings (st_spibsc_spimd_reg_t) (1)

Member	Description
uint32_t cdb	Command bit width • Sets the command bit width in SPI operation mode. • Available settings: SPIBSC_1BIT: 1-bit width SPIBSC_4BIT: 4-bit width • The value of this member is set in the CDB[1:0 bits of the SPI mode enable register (SMENR).
uint32_t ocdb	Optional command bit width • Specifies the optional command bit width in SPI operation mode. • Available settings: SPIBSC_1BIT: 1-bit width SPIBSC_4BIT: 4-bit width • The value of this member is set in the OCDB[1:0] bits of the SPI mode enable register (SMENR).
uint32_t adb	Address bit width • Specifies the width in bits of the address line or lines in SPI operation mode. • Available settings: SPIBSC_1BIT: 1-bit width SPIBSC_4BIT: 4-bit width • The value of this member is set in the ADB[1:0] bits of the SPI mode enable register (SMENR).
uint32_t opdb	Optional data bit width • Specifies the optional data bit width in SPI operation mode. • Available settings: SPIBSC_1BIT: 1-bit width SPIBSC_4BIT: 4-bit width • The value of this member is set in the OPDB[1:0] bits of the SPI mode enable register (SMENR).
uint32_t spidb	Transfer data bit width • Specifies the transfer data bit width in SPI operation mode. • Available settings: SPIBSC_1BIT: 1-bit width SPIBSC_4BIT: 4-bit width • The value of this member is set in the SPIDB[1:0] bits of the SPI mode enable register (SMENR).
uint32_t cde	Sets whether commands are to be output in SPI operation mode. • Available settings: SPIBSC_OUTPUT_DISABLE: Output is disabled. SPIBSC_OUTPUT_ENABLE: Output is enabled. • The value of this member is set in the CDE bit of the SPI mode enable register (SMENR).

Table 5.6 Structure of SPIBSC SPI Mode Settings (st_spibsc_spimd_reg_t) (2)

Member	Description
uint32_t ocde	Optional command enable • Sets whether the optional command is to be output in SPI operation mode. • Available settings: SPIBSC_OUTPUT_DISABLE: Output is disabled. SPIBSC_OUTPUT_ENABLE: Output is enabled. • The value of this member is set in the OCDE bit of the SPI mode enable register (SMENR).
uint32_t ade	Address enable • Sets whether the address is to be output in SPI operation mode. • Available settings: SPIBSC_OUTPUT_DISABLE: Output is disabled. SPIBSC_OUTPUT_ADDR_24: ADR[23:0] are output. SPIBSC_OUTPUT_ADDR_32: ADR[31:0] are output. • The value of this member is set in the ADE[3:0] bits of the SPI mode enable register (SMENR).
uint32_t opde	Optional data enable • Sets whether the optional data is to be issued in SPI operation mode. • Available settings: SPIBSC_OUTPUT_DISABLE: Output is disabled. SPIBSC_OUTPUT_OPD_3: OPD3 is output. SPIBSC_OUTPUT_OPD_32: OPD3 and OPD2 are output. SPIBSC_OUTPUT_OPD_321: OPD3, OPD2, and OPD1 are output. SPIBSC_OUTPUT_OPD_3210: OPD3, OPD2, OPD1, and OPD0 are output. • The value of this member is set in the OPDE[3:0] bits of the SPI mode enable setting register (SMENR).
uint32_t spide	Transfer data enable • Sets whether data transfer is to proceed in SPI operation mode. • Available settings: SPIBSC_OUTPUT_DISABLE: Output is disabled. SPIBSC_OUTPUT_SPID_8: 8- (or 16-) bit transfer SPIBSC_OUTPUT_SPID_16: 16- (or 32-) bit transfer SPIBSC_OUTPUT_SPID_32: 32- (or 64-) bit transfer • The value of this member is set in the SPIDE[3:0] bits of the SPI mode enable register (SMENR).
uint32_t sslkp	SPBSSL signal level retention • Sets the state of the SPBSSL signal after the end of transfer in SPI operation mode. • Available settings: SPIBSC_SPISSL_NEGATE: The signal is negated at the end of transfer. SPIBSC_SPISSL_KEEP: The level of the SPBSSL signal is retained from the end of transfer to the start of next access. • The value of this member is set in the SSLKP bit of the SPI mode control register (SMCR).

Table 5.7 Structure of SPIBSC SPI Mode Settings (st_spibsc_spimd_reg_t) (3)

Member	Description
uint32_t spire	Data read enable • Enables or disables reading of data in SPI operation mode. • Available settings: SPIBSC_SPIDATA_DISABLE: Reading of data is disabled. SPIBSC_SPIDATA_ENABLE: Reading of data is enabled. • The value of this member is set in the SPIRE bit of the SPI mode control register (SMCR).
uint32_t spiwe	Data write enable • Enables or disables writing of data in SPI operation mode. • Available settings: SPIBSC_SPIDATA_DISABLE: Writing of data is disabled. SPIBSC_SPIDATA_ENABLE: Writing of data is enabled. • The value of this member is set in the SPIWE bit of the SPI mode control register (SMCR).
uint32_t dme	Dummy cycle enable • Sets whether dummy cycles are to be inserted in SPI operation mode. • Available settings: SPIBSC_DUMMY_CYC_DISABLE: Dummy cycles are not inserted. SPIBSC_DUMMY_CYC_ENABLE: Dummy cycles are inserted. • The value of this member is set in the DME bit of the SPI mode enable register (SMENR).
uint8_t dmdb	Dummy cycle bit width • Sets the bit width of dummy cycles in SPI operation mode. • Available settings: SPIBSC_1BIT: 1-bit width SPIBSC_4BIT: 4-bit width • The value of this member is set in the DMDB[1:0] bits of the SPI mode dummy cycle setting register (SMDMCR).
uint8_t dmcyc	Number of dummy cycles • Sets the number of dummy cycles in SPI operation mode. • Available settings: SPIBSC_DUMMY_1CYC: 1 cycle SPIBSC_DUMMY_2CYC: 2 cycles SPIBSC_DUMMY_3CYC: 3 cycles SPIBSC_DUMMY_4CYC: 4 cycles SPIBSC_DUMMY_5CYC: 5 cycles SPIBSC_DUMMY_6CYC: 6 cycles SPIBSC_DUMMY_7CYC: 7 cycles SPIBSC_DUMMY_8CYC: 8 cycles • The value of this member is set in the DMCYC[2:0] bits of the SPI mode dummy cycle setting register (SMDMCR).

Table 5.8 Structure of SPIBSC SPI Mode Settings (st_spibsc_spimd_reg_t) (4)

Member	Description
uint8_t cmd	Command • Sets the command for output in SPI operation mode. • The value of this member is set in the CMD[7:0] bits of the SPI mode command setting register (SMCMR).
uint8_t ocmd	Optional command • Sets the optional command for output in SPI operation mode. • The value of this member is set in the OCMD[7:0] bits of the SPI mode command setting register (SMCMR).
uint32_t addr	Address • Sets the address for output in SPI operation mode. • The value of this member is set in the ADR[23:0] bits of the SPI mode address setting register (SMADR). Set the ADRE[7:0] bits for output in 32 bits. This setting is effective when the ADE[3] bit of SMENR is 1.
uint8_t opd[4]	Optional data • Sets the optional data for output in SPI operation mode. • The value of this member is set in the OPDn[7:0] bits of the SPI mode option setting register (SMOPR) as follows. OPD3[7:0] ← opd[0] OPD2[7:0] ← opd[1] OPD1[7:0] ← opd[2] OPD0[7:0] ← opd[3]
uint32_t smrdr	Read data storage buffer • Holds the data read in SPI operation mode (the value of the SPI mode read data register 0 (SMRDR0)) as follows. SMRDR0 → smrdr
uint32_t smwdr	Write data storage buffer • Holds the data for writing in SPI operation mode (the value of the SPI mode write data register 0 (SMWDR0)) as follows. SMWDR0 ← swrdr

5.4 Constants

Table 5.9 to Table 5.12 list the constants used in the sample program.

Table 5.9 Constants Used in the Sample Program (1)

Constant	Setting	Description
SFLASHCMD_SECTOR_ERASE	(0xD8)	Sector erase (3-byte address) command*1
SFLASHCMD_BYTE_PROGRAM	(0x02)	Page programming (3-byte address) command
SFLASHCMD_FAST_READ	(0x0B)	Read fast (3-byte address) command
SFLASHCMD_QUAD_FAST_READ	(0x6B)	Quad read fast (3-byte address) command
SFLASHCMD_QUAD_IO_READ	(0xEB)	Quad I/O read (3-byte address) command
SFLASHCMD_WRITE_ENABLE	(0x06)	Write enable command
SFLASHCMD_READ_STATUS	(0x05)	Read status register-1 command
SFLASHCMD_READ_CONFIG	(0x15)	Read configuration register-1 command
SFLASHCMD_WRITE_STATUS	(0x01)	Write register (status-1, configuration-1) command
SFLASHCMD_SECTOR_ERASE_4B	(0xDC)	Sector erase (4-byte address) command*1
SFLASHCMD_BYTE_PROGRAM_4B	(0x12)	Page programming (4-byte address) command
SFLASHCMD_FAST_READ_4B	(0x0C)	Read fast (4-byte address) command
SFLASHCMD_QUAD_FAST_READ_4B	(0x6C)	Quad read fast (4-byte address) command
SFLASHCMD_QUAD_IO_READ_4B	(0xEC)	Quad I/O read (4-byte address) command

Note 1. A Macronix serial flash memory (product type name: MX25L51245G) for use by this sample program is erased in 64-Kbyte block units.

Table 5.10 Constants Used in the Sample Program (2)

Constant	Setting	Description
STREG_SRWD_BIT	(0x80)	Status register/SRWD bit mask value
STREG_QUAD_BIT	(0x40)	Status register/QUAD bit mask value
STREG_BPROTECT_BIT	(0x3C)	Status register/block protection bit mask value
STREG_WEL_BIT	(0x02)	Status register/write enable latch bit mask value
STREG_WIP_BIT	(0x01)	Status register/write in progress bit mask value
CFREG_LC_BIT	(0xC0)	Configuration register/latency code bit mask value
CFREG_4BYTE_BIT	(0x20)	Configuration register/4-byte bit mask value

Table 5.11 Constants Used in the Sample Program (3)

Constant	Setting	Description
SPIBSC_CMNCR_BSZ_SINGLE	(0)	One serial flash memory is connected to the SPIBSC data bus.
SPIBSC_CMNCR_MD_EXTRD	(0)	Sets mode of the SPIBSC to external address space read mode.
SPIBSC_CMNCR_MD_SPI	(1)	Sets mode of the SPIBSC to SPI operation mode.
SPIBSC_OUTPUT_LOW	(0)	Sets the pin output value to 0.
SPIBSC_OUTPUT_HIGH	(1)	Sets the pin output value to 1.
SPIBSC_OUTPUT_LAST	(2)	Sets the pin output value as the value of the last bit from the previous transfer.
SPIBSC_OUTPUT_HiZ	(3)	Sets the pin output value to HI-Z.
SPIBSC_CMNCR_CPHAT_EVEN	(0)	Sets the edge of the SPBCLK signal for received data to odd edges.
SPIBSC_CMNCR_CPHAT_ODD	(1)	Sets the edge of the SPBCLK signal for received data to even edges.
SPIBSC_CMNCR_CPHAR_ODD	(0)	Sets the edge of the SPBCLK signal for received data to even edges.
SPIBSC_CMNCR_CPHAR_EVEN	(1)	Sets the edge of the SPBCLK signal for received data to odd edges.
SPIBSC_CMNCR_SSLP_LOW	(0)	Sets the sense of the SPBSSL signal to active low.
SPIBSC_CMNCR_SSLP_HIGH	(1)	Sets the sense of the SPBSSL signal to active high.
SPIBSC_CMNCR_CPOL_LOW	(0)	Sets the output level of the SPBCLK pin for the period while the SPBSSL signal is inactive to 0.
SPIBSC_CMNCR_CPOL_HIGH	(1)	Sets the output level of the SPBCLK pin for the period while the SPBSSL signal is inactive to 1.
SPIBSC_DELAY_1SPBCLK	(0)	Sets B’000 to each symbol of the SSL delay register.
SPIBSC_DELAY_2SPBCLK	(1)	Sets B’001 to each symbol of the SSL delay register.
SPIBSC_DELAY_3SPBCLK	(2)	Sets B’010 to each symbol of the SSL delay register.
SPIBSC_DELAY_4SPBCLK	(3)	Sets B’011 to each symbol of the SSL delay register.
SPIBSC_DELAY_5SPBCLK	(4)	Sets B’100 to each symbol of the SSL delay register.
SPIBSC_DELAY_6SPBCLK	(5)	Sets B’101 to each symbol of the SSL delay register.
SPIBSC_DELAY_7SPBCLK	(6)	Sets B’110 to each symbol of the SSL delay register.
SPIBSC_DELAY_8SPBCLK	(7)	Sets B’111 to each symbol of the SSL delay register.
SPIBSC_BURST_1 to SPIBSC_BURST_16	(0x00) to (0x0f)	Burst length for reading: 1 to 16 data-length bits
SPIBSC_BURST_DISABLE	(0)	Disables the read cache.
SPIBSC_BURST_ENABLE	(1)	Enables the read cache.
SPIBSC_DRCR_RCF_EXE	(1)	Clears all entries in the read cache.
SPIBSC_SSL_NEGATE	(0)	The SPBSSL pin is inactive.
SPIBSC_TRANS_END	(1)	Indicates the completion of data transfer.
SPIBSC_1BIT	(0)	Sets the bit width for issuing read commands to 1 bit.
SPIBSC_4BIT	(2)	Sets the bit width for issuing read commands to 4 bits.
SPIBSC_OUTPUT_DISABLE	(0)	Specifies that no command is output when a read command is issued.
SPIBSC_OUTPUT_ENABLE	(1)	Specifies that a command is output when a read command is issued.
SPIBSC_OUTPUT_ADDR_24	(0x07)	Outputs 24-bit addresses.
SPIBSC_OUTPUT_ADDR_32	(0x0f)	Outputs 32-bit addresses.
SPIBSC_OUTPUT_OPD_3	(0x08)	Outputs the optional data OPD3 when a read command is issued.
SPIBSC_OUTPUT_OPD_32	(0x0c)	Outputs the optional data OPD3 and OPD2 when a read command is issued.
SPIBSC_OUTPUT_OPD_321	(0x0e)	Outputs the optional data OPD3, OPD2, and OPD1 when a read command is issued.
SPIBSC_OUTPUT_OPD_3210	(0x0f)	Outputs the optional data OPD3, OPD2, OPD1, and OPD0 when a read command is issued.
SPIBSC_OUTPUT_SPID_8	(0x08)	Enables 8- (or 16-) bit transfer in SPI operation mode.

Table 5.11 Constants Used in the Sample Program (3)

Constant	Setting	Description
SPIBSC_OUTPUT_SPID_16	(0x0c)	Enables 16- (or 32-) bit transfer in SPI operation mode.
SPIBSC_OUTPUT_SPID_32	(0x0f)	Enables 32- (or 64-) bit transfer in SPI operation mode.
SPIBSC_SPISSL_NEGATE	(0)	Sets the state of SPBSSL signal after the end of transfer to the negated state in SPI operation mode.
SPIBSC_SPISSL_KEEP	(1)	Specifies that the level of the SPBSSL signal is retained from the end of transfer to the start of next access in SPI operation mode.
SPIBSC_SPIDATA_DISABLE	(0)	Disables reading of data in SPI operation mode.
SPIBSC_SPIDATA_ENABLE	(1)	Enables reading of data in SPI operation mode.
SPIBSC_SPI_ENABLE	(1)	Starts transfer of SPI data.
SPIBSC_DUMMY_CYC_DISABLE	(0)	Disables insertion of dummy cycles.
SPIBSC_DUMMY_CYC_ENABLE	(1)	Enables insertion of dummy cycles.
SPIBSC_DUMMY_1CYC	(0)	Sets the number of dummy cycles for output to the serial flash memory when converting read operations for the SPI multi-I/O bus space to SPI communications to 1.
SPIBSC_DUMMY_2CYC	(1)	Sets the number of dummy cycles for output to the serial flash memory when converting read operations for the SPI multi-I/O bus space to SPI communications to 2.
SPIBSC_DUMMY_3CYC	(2)	Sets the number of dummy cycles for output to the serial flash memory when converting read operations for the SPI multi-I/O bus space to SPI communications to 3.
SPIBSC_DUMMY_4CYC	(3)	Sets the number of dummy cycles for output to the serial flash memory when converting read operations for the SPI multi-I/O bus space to SPI communications to 4.
SPIBSC_DUMMY_5CYC	(4)	Sets the number of dummy cycles for output to the serial flash memory when converting read operations for the SPI multi-I/O bus space to SPI communications to 5.
SPIBSC_DUMMY_6CYC	(5)	Sets the number of dummy cycles for output to the serial flash memory when converting read operations for the SPI multi-I/O bus space to SPI communications to 6.
SPIBSC_DUMMY_7CYC	(6)	Sets the number of dummy cycles for output to the serial flash memory when converting read operations for the SPI multi-I/O bus space to SPI communications to 7.
SPIBSC_DUMMY_8CYC	(7)	Sets the number of dummy cycles for output to the serial flash memory when converting read operations for the SPI multi-I/O bus space to SPI communications to 8.

Table 5.12 Constants Used in the Sample Program (4)

Constant	Setting	Description
R_SERIAL_FLASH_TOP	(0x10000000U)	Address where the installed serial flash memory starts
R_SERIAL_FLASH_END	(0x13FFFFFCU)	Address where the installed serial flash memory ends
R_SERIAL_FLASH_ADDR_LIMIT	0x14000000U	Boundary value of the address of the installed serial flash memory
R_SERIAL_FLASH_READ_SIZE	(64U)	Maximum value of the amount of data to be read
R_SERIAL_FLASH_CNT_ADDR	(0x13FF0000U)	Counter variable area
R_SERIAL_FLASH_DATA_WIDTH	SPIBSC_4BIT (2)	Data read bit width
R_SERIAL_FLASH_ADDR_MODE	SPIBSC_OUTPUT_ADDR_32 (0x0f)	Number of address bytes
R_SERIAL_FLASH_WRITE_BYTE	(4)	Amount of bytes to be written
R_SERIAL_FLASH_READ_BYTE	(4)	Amount of bytes to be read

5.5 Variables

Table 5.13 and Table 5.14 list the static and global constants, respectively.

Table 5.13 Static Variables

Type	Variable Name	Description
static char	gbuff[16]	Data received from the terminal software
st_spibsc_cfg_t	spibsc_cfg	SPIBSC external address read settings storage variable Stores the SPIBSC external address read settings.

Table 5.14 Global Variables

Type	Variable Name	Description
st_spibsc_spimd_reg_t	g_spibsc_spimd_reg	SPIBSC SPI mode operation settings storage variable • Stores the SPIBSC settings when the SPIBSC is used in SPI mode. In the sample program, these settings are also used as arguments when running serial flash control functions within the API functions and user-defined functions.

5.6 Functions

The sample code consists of the following functions: the interface functions for using peripheral modules (API functions); the user-defined functions that must be prepared by the user to suit the application of the user system (function called by the API functions); and the sample functions required to operate the sample code.

Table 5.15 to Table 5.17 list the functions to be used.

Table 5.15 List of Functions (1)

Function	Description
spibsc_init	SPIBSC initial settings function Makes settings optimized for use with a Macronix flash memory (product type name: MX25L51245G) within this function.
serial_flash_demo	Function for executing the demonstration program
serial_flash_read	Function for executing the serial flash read program
serial_flash_write	Function for executing the serial flash write program
serial_flash_erase	Function for executing the serial flash erasure program
main	Main function of the sample program

Table 5.16 List of Functions (2)

Function	Description
R_SPIBSC_ExmodeSetting	SPIBSC initial settings function Makes initial settings required for controlling the serial flash memory and for using the SPIBSC in external address read mode. This function also makes register settings in the flash memory to suit the initial settings. After the initial settings, it places the SPIBSC in external address read mode.
R_SPIBSC_WaitTend	SPIBSC data transfer end wait function Waits for the completion of data transfer from the SPIBSC.
R_SPIBSC_Exmode	SPIBSC external address mode setting function Places the SPIBSC in external address read mode.
R_SPIBSC_SetConfig	SPIBSC external address read settings function Makes initial settings required for using the SPIBSC in external address read mode.
R_SPIBSC_Spimode	SPIBSC SPI mode setting function Places the SPIBSC in SPI mode.
R_SPIBSC_ExmodeInit	SPIBSC external address mode initial settings function Makes initial settings for using the SPIBSC in external address read mode. After the initial settings, it places the SPIBSC in external address read mode.
R_SPIBSC_EraseSector	Serial flash memory erasure function Uses SPI mode of the SPIBSC to erase the serial flash memory.
R_SPIBSC_ByteProgram	Serial flash memory programming function Uses SPI mode of the SPIBSC to write data to the serial flash memory.
R_SPIBSC_ByteRead	Serial flash memory read function Uses SPI mode of the SPIBSC to read data from the serial flash memory.
R_SPIBSC_SpibscTransfer	Serial flash memory control function Issues commands to the serial flash memory according to arguments.

Table 5.17 List of Functions (3)

Function	Description
userdef_spibsc_set_config	SPIBSC external address read settings function Determines the SPIBSC external address read mode settings to suit the serial flash memory in use. The sample program makes initial settings required for using the SPIBSC in external address read mode on the basis of the settings made by this function. The sample program makes initial settings for the SPIBSC for use with a Macronix serial flash memory (product type name: MX25L51245G).
userdef_sflash_set_mode	Serial flash memory internal register settings function Makes settings for the registers in the serial flash memory required when using the SPIBSC in external address read mode, to suit the serial flash memory in use. The sample program makes initial settings for the registers in the Macronix serial flash memory (product type name: MX25L51245G).
userdef_sflash_write_enable	Serial flash memory write enable function Makes settings for the registers in the serial flash memory to enable writing, to suit the serial flash memory in use. The sample program makes settings for the registers in the Macronix serial flash memory (product type name: MX25L51245G).
userdef_sflash_busy_wait	Serial flash memory ready wait function Reads the registers in the serial flash memory and waits for the serial flash memory to enter the ready state, over a period that suits the serial flash memory in use. The sample program waits for the Macronix serial flash memory (product type name: MX25L51245G) to enter the ready state.

5.7 Details of Functions

The following tables list the details of the functions.

spibsc_init	
Synopsis	SPIBSC initial settings function
Declaration	static void spibsc_init(void);
Description	This function makes settings optimized for use with a Macronix serial flash memory (product type name: MX25L51245G) within this function.
Arguments	None
Return value	None
Note	This function is used when the MCU is booted in 16-bit bus boot mode. This function is not used when the MCU is booted in SPI boot mode, since the initial settings for the SPIBSC are made within the settings for bus connection during booting up. For details, check with the RZ/T1 Group Initial Settings application note (R01AN2554EJ).

serial_flash_demo	
Synopsis	Function for executing the demonstration program
Declaration	static void serial_flash_demo (void)
Description	This function executes the demonstration program by handling the issuing of user commands by key input.
Arguments	None
Return value	None

serial_flash_read	
Synopsis	Function for executing the serial flash read program
Declaration	static void serial_flash_read (void)
Description	This function executes the serial flash read program by handling the issuing of user commands by key input.
Arguments	None
Return value	None

serial_flash_write	
Synopsis	Function for executing the serial flash write program
Declaration	static void serial_flash_write (void)
Description	This function executes the serial flash write program by handling the issuing of user commands by key input.
Arguments	None
Return value	None

serial_flash_erase	
Synopsis	Function for executing the serial flash erasure program
Declaration	static void serial_flash_erase (void)
Description	This function executes the serial flash erasure program by handling the issuing of user commands by key input.
Arguments	None
Return value	None

main	
Synopsis	Main function of the sample program
Declaration	void main (void)
Description	Main processing of the sample program
Arguments	None
Return value	None

R_SPIBSC_ExmodeSetting		
Synopsis	SPIBSC initial settings function	
Declaration	int32_t R_SPIBSC_ExmodeSetting (st_spibsc_cfg_t *spibsccfg);	
Description	Makes initial settings required for controlling the serial flash memory and for using the SPIBSC in external address read mode. This function also makes register settings in the flash memory to suit the initial settings. After the initial settings, it places the SPIBSC in external address read mode. The SPIBSC external address mode initial settings function (R_SPIBSC_ExmodeInit) is executed from within this function.	
Arguments	st_spibsc_cfg_t *spibsccfg	SPIBSC external address read settings For details of the settings, see Table 5.2 to Table 5.4.
Return value	0: Normal termination -1: Error	

R_SPIBSC_SetConfig		
Synopsis	SPIBSC external address read settings function	
Declaration	int32_t R_SPIBSC_SetConfig (st_spibsc_cfg_t *spibsccfg);	
Description	Determines the settings for using the SPIBSC in external address read mode to suit the serial flash memory in use. The user-defined function (SPIBSC external address read settings function: userdef_spibsc_set_config) is executed from within this function.	
Arguments	st_spibsc_cfg_t *spibsccfg	SPIBSC external address read settings For details of the settings, see Table 5.2 to Table 5.4.
Return value	0: Normal termination -1: Error	

R_SPIBSC_WaitTend	
Synopsis	SPIBSC data transfer end wait function
Declaration	void R_SPIBSC_WaitTend(void);
Description	Waits for the completion of data transfer from the SPIBSC.
Arguments	None
Return value	None

R_SPIBSC_Exmode	
Synopsis	SPIBSC external address mode setting function
Declaration	int32_t R_SPIBSC_Exmode(void);
Description	This function places the SPIBSC in external address read mode.
Arguments	None
Return value	0: Setting has succeeded.

R_SPIBSC_Spimode	
Synopsis	SPIBSC SPI mode setting function
Declaration	int32_t R_SPIBSC_Spimode(void);
Description	This function places the SPIBSC in SPI mode.
Arguments	None
Return value	0: Setting has succeeded.

R_SPIBSC_ExmodeInit		
Synopsis	SPIBSC external address mode initial settings function	
Declaration	int32_t R_SPIBSC_ExmodeInit(st_spibsc_cfg_t *spibsccfg)	
Description	This function makes initial settings required for using the SPIBSC in external address read mode. After the initial settings, it places the SPIBSC in external address read mode.	
Arguments	st_spibsc_cfg_t *spibsccfg	SPIBSC external address read settings For details of the settings, see Table 5.2 to Table 5.4.
Return value	0: Normal termination -1: Error	

R_SPIBSC_EraseSector		
Synopsis	Serial flash memory erase function	
Declaration	int32_t R_SPIBSC_EraseSector(uint32_t addr, uint32_t data_width, uint32_t addr_mode);	
Description	This function erases the serial flash memory using SPI mode of the SPIBSC. It erases the serial flash memory in 64-Kbyte block units.	
Arguments	uint32_t addr	Address to be erased in serial flash memory
	uint32_t data_width	Data read bit width Bit width for reading data from the serial flash memory when converting read operations for the SPI multi-I/O bus space to SPI communications. SPIBSC_1BIT: 1-bit width SPIBSC_4BIT: 4-bit width
	uint32_t addr_mode	Address mode setting Sets the address for output to the serial flash memory when converting read operations for the SPI multi-I/O bus space to SPI communications. SPIBSC_OUTPUT_ADDR_24: 24-bit address output SPIBSC_OUTPUT_ADDR_32: 32-bit address output
Return value	0: Setting has succeeded. -1: Setting has failed.	

R_SPIBSC_ByteProgram		
Synopsis	Flash memory write function	
Declaration	int32_t R_SPIBSC_ByteProgram(uint32_t addr, uint8_t *buf, int32_t size, uint32_t data_width, uint32_t addr_mode);	
Description	This function writes data to the serial flash memory using the SPI mode of the SPIBSC.	
Arguments	uint32_t addr	Address to be written to in serial flash memory The specified address must be on a 4-byte boundary.
	uint8_t *buf	Write data storage buffer
	int32_t size	Amount of data to be written (in bytes) A multiple of 4 must be specified as the amount of data.
	uint32_t data_width	Data read bit width Bit width for reading data from the serial flash memory when converting read operations for the SPI multi-I/O bus space to SPI communications. SPIBSC_1BIT: 1-bit width SPIBSC_4BIT: 4-bit width
	uint32_t addr_mode	Address mode setting Sets the address for output to the serial flash memory when converting read operations for the SPI multi-I/O bus space to SPI communications. SPIBSC_OUTPUT_ADDR_24: 24-bit address output SPIBSC_OUTPUT_ADDR_32: 32-bit address output
Return value	0: Setting has succeeded. -1: Setting has failed.	

R_SPIBSC_ByteRead		
Synopsis	Serial flash memory read function	
Declaration	int32_t R_SPIBSC_ByteRead (uint32_t addr, uint8_t *buf, int32_t size, uint32_t data_width, uint32_t addr_mode);	
Description	This function reads data to the serial flash memory using the SPI mode of the SPIBSC.	
Arguments	uint32_t addr	Address to be written to in serial flash memory If the bit width of the data is 4, the specified address must be on a 4-byte boundary.
	uint8_t *buf	Write data storage buffer
	int32_t size	Amount of data to be written (in bytes)
	uint32_t data_width	Data read bit width Bit width for reading data from the serial flash memory when converting read operations for the SPI multi-I/O bus space to SPI communications. SPIBSC_1BIT: 1-bit width SPIBSC_4BIT: 4-bit width
	uint32_t addr_mode	Address mode setting Sets the address for output to the serial flash memory when converting read operations for the SPI multi-I/O bus space to SPI communications. SPIBSC_OUTPUT_ADDR_24: 24-bit address output SPIBSC_OUTPUT_ADDR_32: 32-bit address output
Return value	0: Setting has succeeded. -1: Setting has failed.	

R_SPIBSC_SpibscTransfer		
Synopsis	Serial flash memory control function	
Declaration	int32_t R_SPIBSC_SpibscTransfer(st_spibsc_spimd_reg_t *regset);	
Description	This function accesses the serial flash memory using SPI mode of the SPIBSC.	
Arguments	st_spibsc_spimd_reg_t * regset	SPIBSC SPI mode setting For details of the settings, see Table 5.5 to Table 5.8.
Return value	0: Setting has succeeded. -1: Setting has failed.	

userdef_spibsc_set_config		
Synopsis	SPIBSC external address read settings function	
Declaration	void userdef_spibsc_set_config (st_spibsc_cfg_t *spibsccfg);	
Description	Determines the SPIBSC external address read mode settings to suit the serial flash memory in use. This function makes initial settings required for using the SPIBSC in the area specified by argument spibsccfg in external address read mode. For details of the settings as argument spibsccfg, see Table 5.2 to Table 5.4.	
Arguments	st_spibsc_cfg_t *spibsccfg	SPIBSC external address read settings For details of the settings, see Table 5.2 to Table 5.4.
Return value	None	
Note	The sample program makes initial settings for the SPIBSC for use with a Macronix serial flash memory (product type name: MX25L51245G).	

userdef_sflash_set_mode		
Synopsis	Serial flash memory internal register settings function	
Declaration	int32_t userdef_spibsc_set_mode (uint32_t data_width, uint32_t addr_mode);	
Description	Within this function, implement processing for required setting of the registers in the serial flash memory for use with the SPIBSC in external address read mode, to suit the serial flash memory to be used.	
Arguments	uint32_t data_width	Data read bit width Bit width for reading data from the serial flash memory when converting read operations for the SPI multi-I/O bus space to SPI communications. SPIBSC_1BIT: 1-bit width SPIBSC_4BIT: 4-bit width
	uint32_t addr_mode	Address mode setting Sets the address for output to the serial flash memory when converting read operations for the SPI multi-I/O bus space to SPI communications. SPIBSC_OUTPUT_ADDR_24: 24-bit address output SPIBSC_OUTPUT_ADDR_32: 32-bit address output
Return value	0: Setting has succeeded. -1: Setting has failed.	
Note	The sample program makes initial settings for the SPIBSC for use with a Macronix serial flash memory (product type name: MX25L51245G).	

userdef_sflash_write_enable		
Synopsis	Serial flash memory write enable function	
Declaration	int32_t userdef_spibsc_write_enable (void);	
Description	Within this function, implement processing for setting of the registers in the serial flash memory to enable writing, to suit the serial flash memory to be used.	
Arguments	None	
Return value	0: Setting has succeeded. -1: Setting has failed.	
Note	The sample program makes initial settings for the SPIBSC for use with a Macronix serial flash memory (product type name: MX25L51245G).	

userdef_sflash_busy_wait		
Synopsis	Serial flash memory ready wait function	
Declaration	int32_t userdef_spibsc_busy_wait (uint32_t data_width);	
Description	Within this function, implement processing for reading the registers in the serial flash memory for the transition of the serial flash memory to the ready state, to suit the serial flash memory to be used.	
Arguments	uint32_t data_width	Data read bit width Bit width for reading data from the serial flash memory when converting read operations for the SPI multi-I/O bus space to SPI communications. SPIBSC_1BIT: 1-bit width SPIBSC_4BIT: 4-bit width
Return value	None	
Note	The sample program makes initial settings for the SPIBSC for use with a Macronix serial flash memory (product type name: MX25L51245G).	

5.8 Flowcharts of the Sample Program Functions

5.8.1 SPIBSC Initial Settings Function

Figure 5.3 shows the flow of the SPIBSC initial settings function.

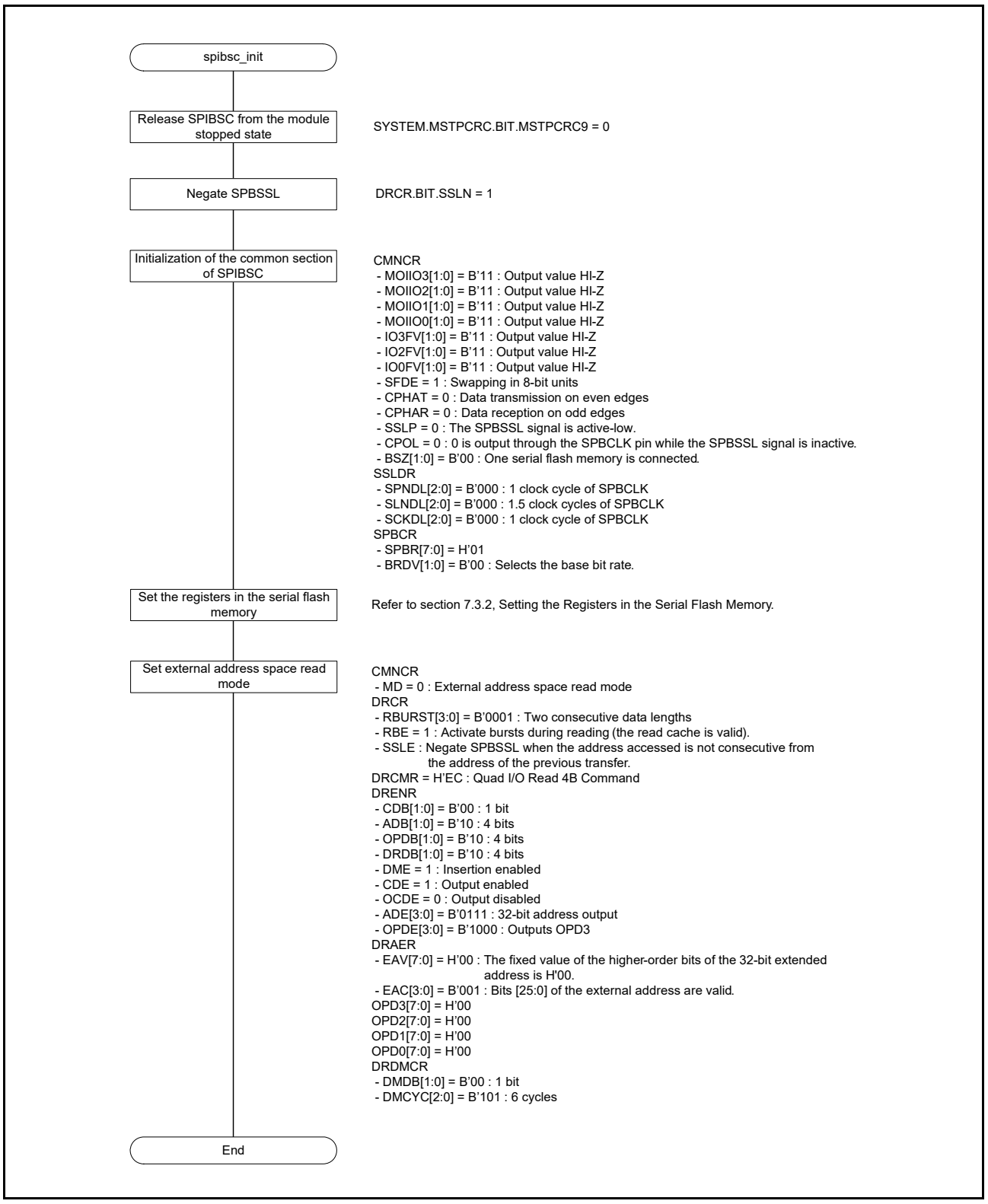


Figure 5.3 Flow of the SPIBSC Initial Settings Function

5.8.2 Main Function

Figure 5.4 shows the flow of the main function.

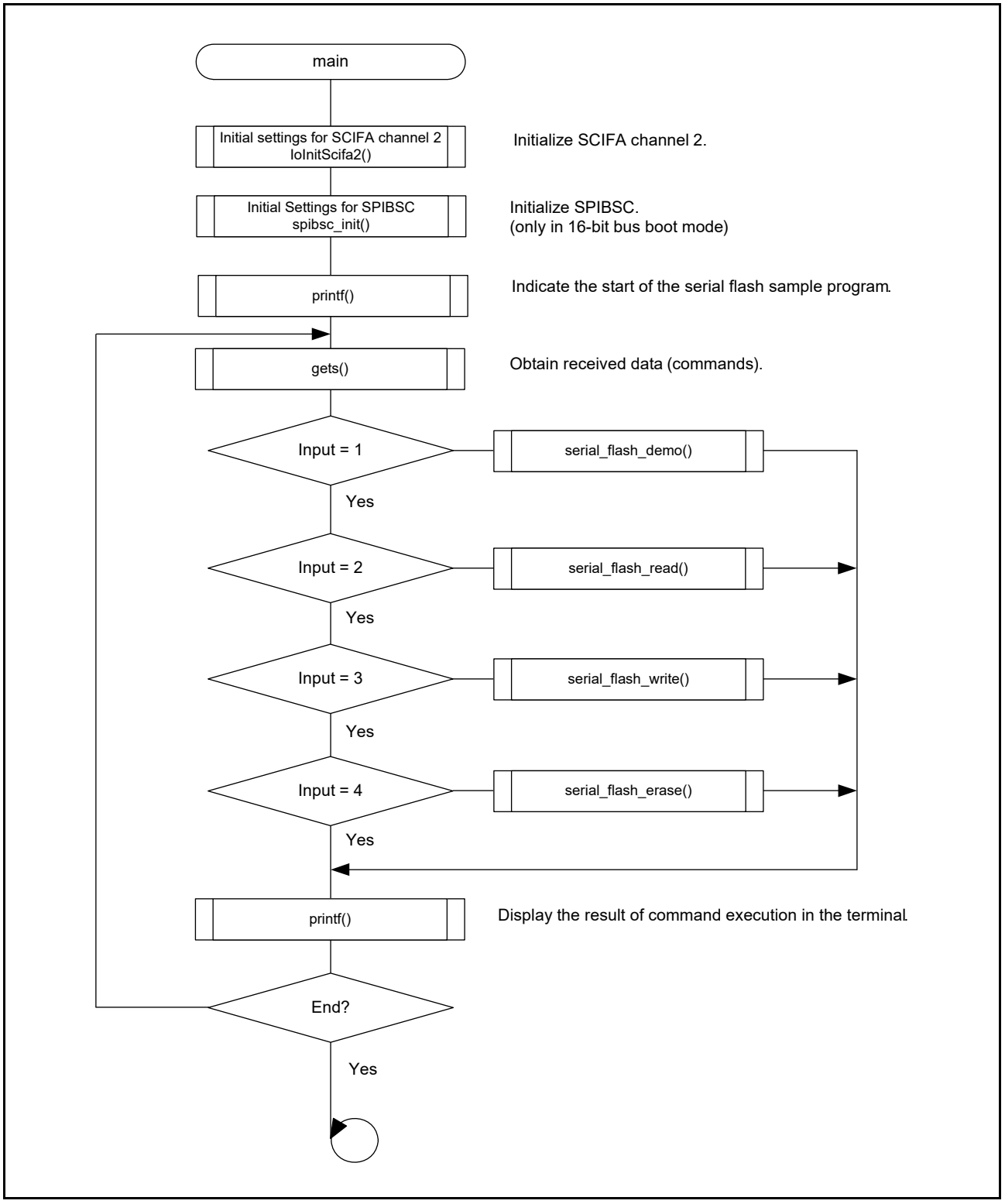


Figure 5.4 Flow of the Main Function

5.9 Operation of the Sample Program

The following shows operation of the sample program in the terminal software on the PC.

Note that a value input by the user in the sample program will be 0 if any character other than a number 0 to 9 or letter A to F is input.

The following menu is displayed when the power of the evaluation board is switched on or following a reset.

```
Serial Flash example program

[1] Demo
[2] Read Serial Flash
[3] Write Serial Flash
[4] Erase Serial Flash

[9] Exit

>
```

5.9.1 Demonstration Program

Switch the mode of SPIBSC for reading and programming of the serial flash memory.

When [1] is selected in the menu, the counter variable set at the end of serial flash memory in external address space read mode is read and displayed.

```
>1[Enter]

OutAddrMode  13FF0000  FFFFFFFF
```

In this sample program, the address of the counter variable is set to 0x13FF0000.

Switch the current mode to SPI operation mode and the value of the counter variable is again read and displayed.

At this time, check that the values of the counter variable read in the two modes are identical.

```
>1[Enter]

OutAddrMode  13FF0000  FFFFFFFF
SPIMode      13FF0000  FFFFFFFF
```

After the value is read in SPI operation mode, the value of the counter variable plus 1 is written.

If you boot the MCU and select [1], you can confirm that the values of the counter variable in the two modes have been added.

```
>1[Enter]

OutAddrMode  13FF0000  00000000
SPIMode      13FF0000  00000000
```

5.9.2 Reading the Serial Flash Memory

Read the data of the specified address in SPI operation mode. The amount of data to be read is selectable.

Select [2] in the menu, then input the address and size.

Here, 11000000 and 20 are input as the address and size, respectively.

```
>2[Enter]

Input the top address (10000000 - 13FFFFFC)

>11000000[Enter]

>Input the number of blocks (1 - 64: 4-byte blocks)

>20[Enter]

>
```

Twenty long-words of data are displayed as shown below.

```
>20[Enter]

            00       04       08       0C
11000000 00000000 00070707 00000003 00000000
11000010 00000000 3000004C 00006000 00802000
11000020 00000000 00000000 00000000 00000000
11000030 00000000 00000000 00000000 00000000
11000040 00000000 00000000 0000B7DD F1020011

[1] Demo
[2] Read Serial Flash
[3] Write Serial Flash
[4] Erase Serial Flash

[9] Exit
```

5.9.3 Programming the Serial Flash Memory

Write data to the specified address in SPI operation mode.

Select [3] in the menu and input the address where the writing of data is to start.

```
>3[Enter]

Input the top address (10000000 - 13FFFFFC)

>13000000[Enter]

Input the data (00000000 - FFFFFFFF)
Input just [Enter] without data to exit

13000000 FFFFFFFF >
```

The start address and the current value of the data are displayed. Input the data.

If the current value is not “not set” (FFFFFFFF), erase the data in the serial flash memory according to the procedure in Section 5.9.4, Erasing the Serial Flash Memory, before programming.

After the completion of data programming, the next long-word address and current value are displayed.

```
13000000 FFFFFFFF > 1234[Enter]
13000004 FFFFFFFF >
```

To end programming, input only [Enter].

```
13000000 FFFFFFFF > 00001234[Enter]
13000004 FFFFFFFF > 00000000[Enter]
13000008 FFFFFFFF > 00001234[Enter]
1300000C FFFFFFFF > 00000000[Enter]
13000010 FFFFFFFF > [Enter]
```

5.9.4 Erasing the Serial Flash Memory

Erase the sector which includes the specified address in SPI operation mode.

In the menu, select [4] and input an address within the sector you want to erase.

```
>4[Enter]

Input the top address (10000000 - 13FFFFFC)

>13001000[Enter]
```

The data of the sector which includes the input address (in the example, the range from 13000000 to 1300FFFF) will be erased.

6. Procedure for Switching SPIBSC Mode

6.1 Problems which May Arise Due to Branch Prediction and Speculative Execution

6.1.1 Access to ROM/RAM in Cortex-R4

If the memory type of the given address range is set to “normal” by the MPU, branch prediction and speculative execution may lead to unintentional access. Accordingly, if the memory type of the area is set to “normal”, be sure to make it accessible beforehand.

6.1.2 Modes of the SPI Multi-I/O Bus Controller

The SPI multi-I/O bus controller has two modes: external address space read mode and SPI operation mode.

In external address space read mode, the SPI multi-I/O bus space can be accessed directly by the CPU.

Accordingly, the memory type of the SPI multi-I/O bus space can be set to “normal” in this mode.

On the other hand, the SPI multi-I/O bus space cannot be accessed directly by the CPU in SPI operation mode.

Accordingly, the memory type of the SPI multi-I/O bus space must be set to a type other than “normal”, or to “access-prohibited” if it is left in this mode.

6.2 Outline of Mode Switching

6.2.1 Switching between Modes

When switching external address space read mode and SPI operation mode, change the settings for the MPU appropriately.

6.2.2 Outline of Procedure for Making Changes

Modify the conversion table of the MPU while in eternal address space read mode.

Be sure to take the above step first, since if the conversion table of the MPU is modified after switching to SPI operation mode, unintentional access to the I/O bus space may occur before the table has reflected changes.

If processing for changing the register settings in the SPI multi-I/O bus controller is executed from the SPI multi-I/O bus space (serial flash memory), the subsequent operation is not guaranteed. Be sure to execute this processing from the internal RAM.

7. Application Examples

This section describes how to change the sample program to suit the flash memory used by the customer, as an example of the practical application of the sample program.

7.1 Conditions for the Sample Program

The sample program makes settings optimized for use with a Macronix serial flash memory (product type name: MX25L51245G) under the conditions listed in Table 7.1.

How to change the sample program when these conditions are to be changed is described below.

Table 7.1 Conditions for the Sample Program

Condition	Settings	Remark
Serial flash memory	Macronix serial flash memory (product type name: MX25L51245G)	—
Data bus width	4 bits	Bit width for reading data
Number of address bytes	4 bytes	Number of bytes to be issued when specifying addresses

7.2 Changing the Sample Program when the Serial Flash Memory is Not to be Changed

Table 7.2 lists how to change the sample program when the serial flash memory in use is not to be changed.

Table 7.2 How to Change the Sample Program when the Serial Flash Memory is Not to be Changed

Condition	Changes	How to Make Changes
Data bus width for reading data	1 bit	Define (1) in the macro definition (SPIBSC_BUS_WITDH)*1.
	4 bits	Define (4) in the macro definition (SPIBSC_BUS_WITDH).
Number of address bytes	3 bytes	Define (SPIBSC_OUTPUT_ADDR_24) in the macro definition (SPIBSC_OUTPUT_ADDR)*2.
	4 bytes	Define (SPIBSC_OUTPUT_ADDR_32) in the macro definition (SPIBSC_OUTPUT_ADDR).

Note 1. The macro definition (SPIBSC_BUS_WITDH) is defined in the spibsc_ioset_userdef.c file.*3
Note 2. The macro definition (SPIBSC_OUTPUT_ADDR) is defined in the spibsc_ioset_userdef.c file.*3
Note 3. Refer to the above macro definitions in the serial_flash_ioset_userdef.c file in 16-bit bus boot mode.

7.3 Changing the Sample Program when the Serial Flash Memory is to be Changed

When changing the serial flash memory, the sample program must be changed to suit the specifications of the flash memory in use.

Table 7.3 lists the points to be changed in the sample program.

Table 7.3 Points to be Changed in the Sample Program

Points to be Changed	Description
Read command waveform	In external address space read mode, change the signal output to the serial flash memory when converting read operations for the SPI multi-I/O bus space to SPI communications to match the read command of the serial flash memory you are using.
Register settings in the serial flash memory	Make settings for the registers in the serial flash memory required when using the SPIBSC in external address read mode, to suit the serial flash memory in use.
Enabling writing to the serial flash memory	Set the registers in the serial flash memory to enable writing, to suit the serial flash memory in use.*1
Waiting for the serial flash memory to be ready	Read the registers in the serial flash memory and wait for the serial flash memory to enter the ready state, to suit the flash memory in use.
Releasing the serial flash memory from protection	Make settings for the registers in the serial flash memory to release it from protection, to suit the flash memory in use.*2

Note 1. Setting the registers in the serial flash memory may require enabling of writing depending on the serial flash memory in use.
Note 2. Setting the registers in the serial flash memory may require releasing it from protection depending on the serial flash memory in use.

7.3.1 Changing the Read Command Waveforms

In external address space read mode, change the signal output to the serial flash memory when converting read operations for the SPI multi-I/O bus space to SPI communications to match the read command of the serial flash memory you are using.

The signal output to the serial flash memory in external address space read mode is changed by a setting in an SPIBSC control register.

In the sample program, the values set in the SPIBSC control register can be changed by a global variable (SPIBSC external address read setting storage variable: spibsc_cfg). A user-defined function (SPIBSC external address read settings function: userdef_spibsc_set_config) makes settings for spibsc_cfg.

Figure 7.1 shows the relationship between the SPIBSC control register settings and the waveforms output to the serial flash memory while the SPIBSC is reading from an external address, and Table 7.4 lists the settings of the SPIBSC control register in the sample program.

Refer to these example settings to make settings for spibsc_cfg to match the read command of the serial flash memory in use.

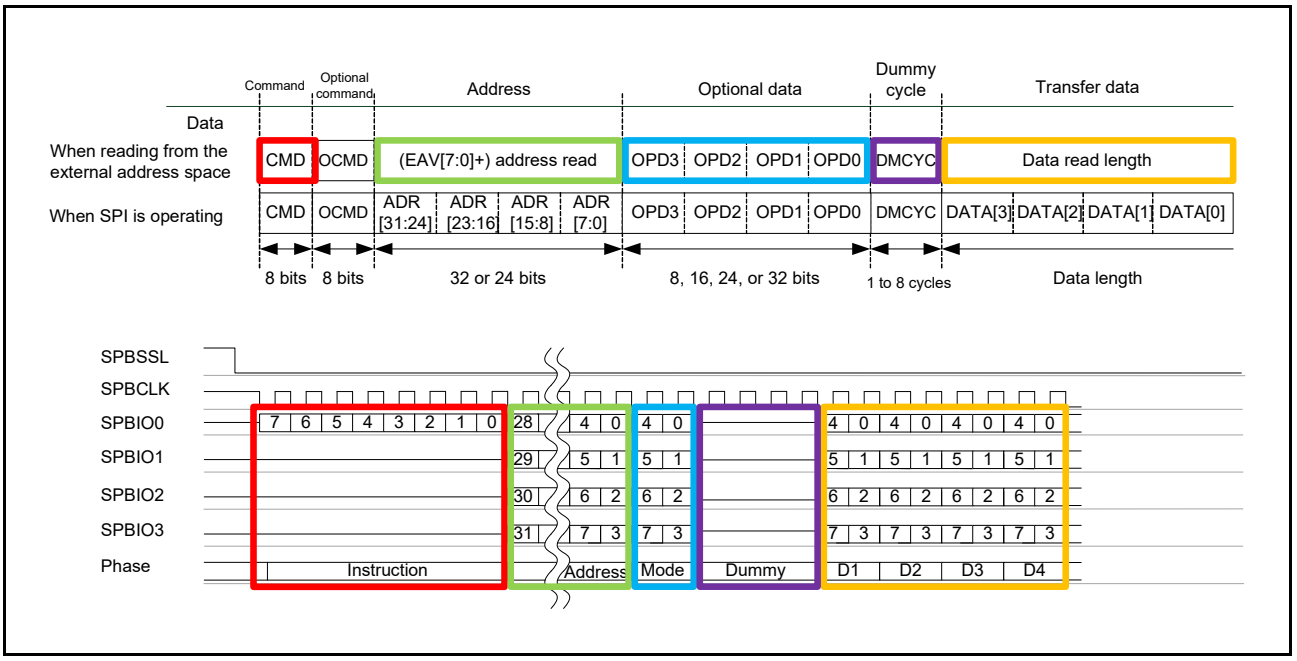


Figure 7.1 Relationship between the SPIBSC Control Register Settings and the Waveforms Output to the Serial Flash Memory while the SPIBSC is Reading from an External Address

If the MCU is to be booted up in SPI boot mode, please change the user-defined function Userdef_SPIBSC_Set_Config for boot processing to reflect the read command of the serial flash memory you are using.

Table 7.4 SPIBSC Control Register Settings in the Sample Program

SPIBSC Registers		Setting	Remark
DRCMR	CMD[7:0]	H'EB	Quad I/O read command
	OCMD[7:0]	H'00	—
DROPR	OPD3[7:0]	H'00	—
	OPD2[7:0]	H'00	—
	OPD1[7:0]	H'00	—
	OPD0[7:0]	H'00	—
DRENR	CDB[1:0]	B'00	Command bit width: 1-bit width
	OCDB[3:0]	B'0000	—
	ADB[1:0]	B'10	Address bit width: 4-bit width
	OPDB[1:0]	B'10	Optional data bit width: 4-bit width
	DRDB[1:0]	B'10	Data read bit width: 4-bit width
	DME	B'1	Dummy cycles: Inserted
	CDE	B'1	Commands: Issued
	OCDE	B'0	Optional command: Not issued
	ADE[3:0]	B'0111	Address enable: 24-bit address output
	OPDE[3:0]	B'1000	Optional data: OPD3 is output
DRDMCR	DMDB[1:0]	B'00	Bit width of dummy cycles: 1-bit width
	DMCYC[2:0]	B'101	Number of dummy cycles: 6 cycles
SPBCR	SPBR[7:0]	H'01	Bit rate: PCLKA/2
	BRDV[1:0]	B'00	

7.3.2 Setting Registers in the Serial Flash Memory

The registers in the serial flash memory must be set when reading from the serial flash memory in Section 7.3.1, Changing the Read Command Waveforms.

In the sample program, the user-defined function (serial flash memory internal register settings function: userdef_sflash_set_mode) handles the processing to set the QUAD bit in the status register of the Macronix serial flash memory (product type name: MX25L51245G) to 1 (= quad), the DC1 (dummy cycle 1) bit in the configuration register to 1, and the DC0 (dummy cycle 0) bit to 0.

When setting the registers in the serial flash memory, SPI mode of the SPIBSC is used. To set the registers in the Macronix serial flash memory (product type name: MX25L51245G), the write enable latch (WEL) bit must be set to 1 by issuing the write enable command (WREN). This must be done before the status register and configuration registers can be set. In the sample program, the user-defined function (serial flash memory write enable function: userdef_sflash_write_enable) handles the processing to issue the write enable command (WREN).

Figure 7.2 shows the flow of setting the registers in the serial flash memory in the sample program.

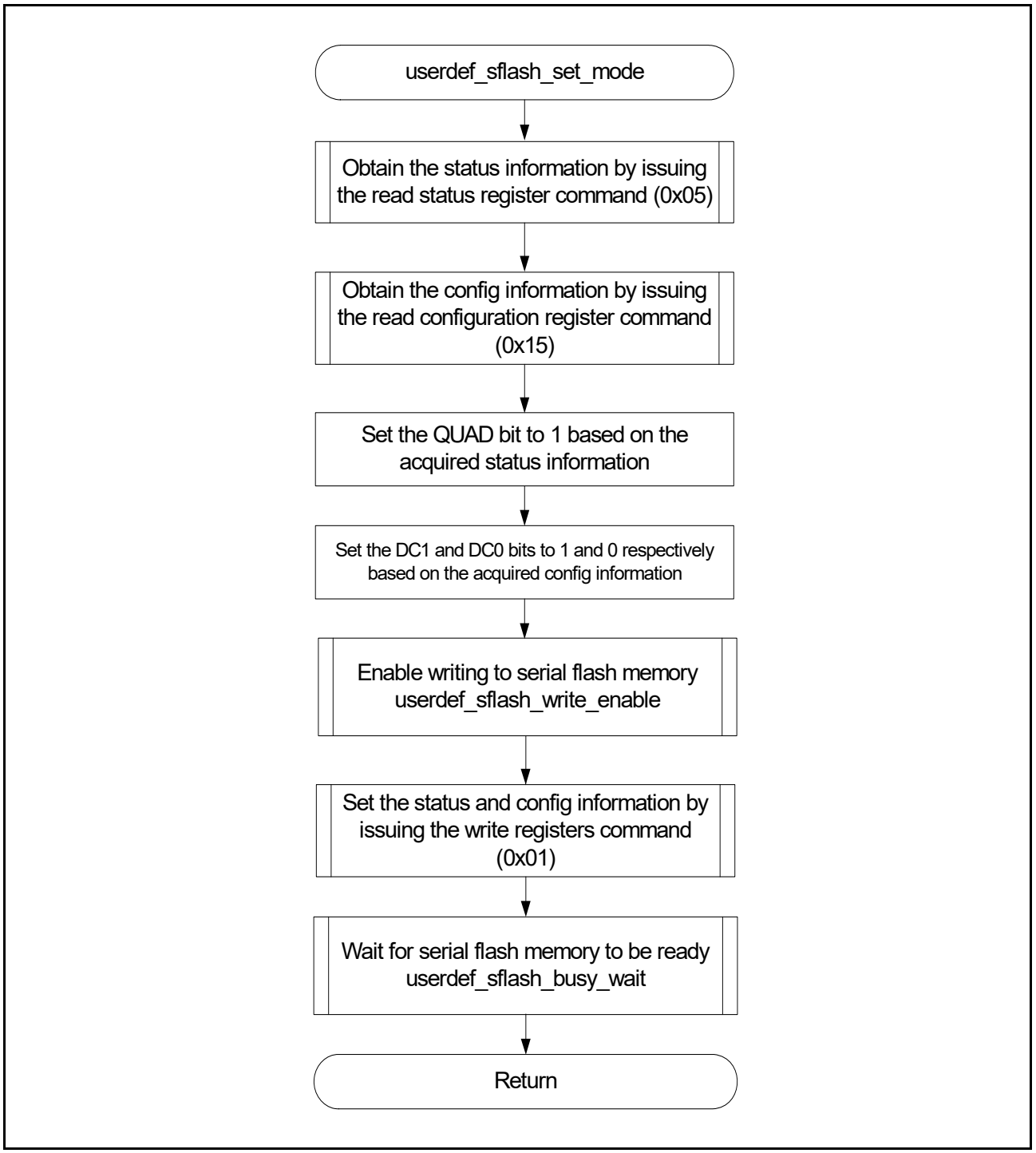


Figure 7.2 Flow of Setting the Registers in the Serial Flash Memory

If the MCU is to be booted up in SPI boot mode, please change the user-defined function Userdef_SPIBSC_Set_Mode for boot processing to reflect the register settings of the serial flash memory you are using.

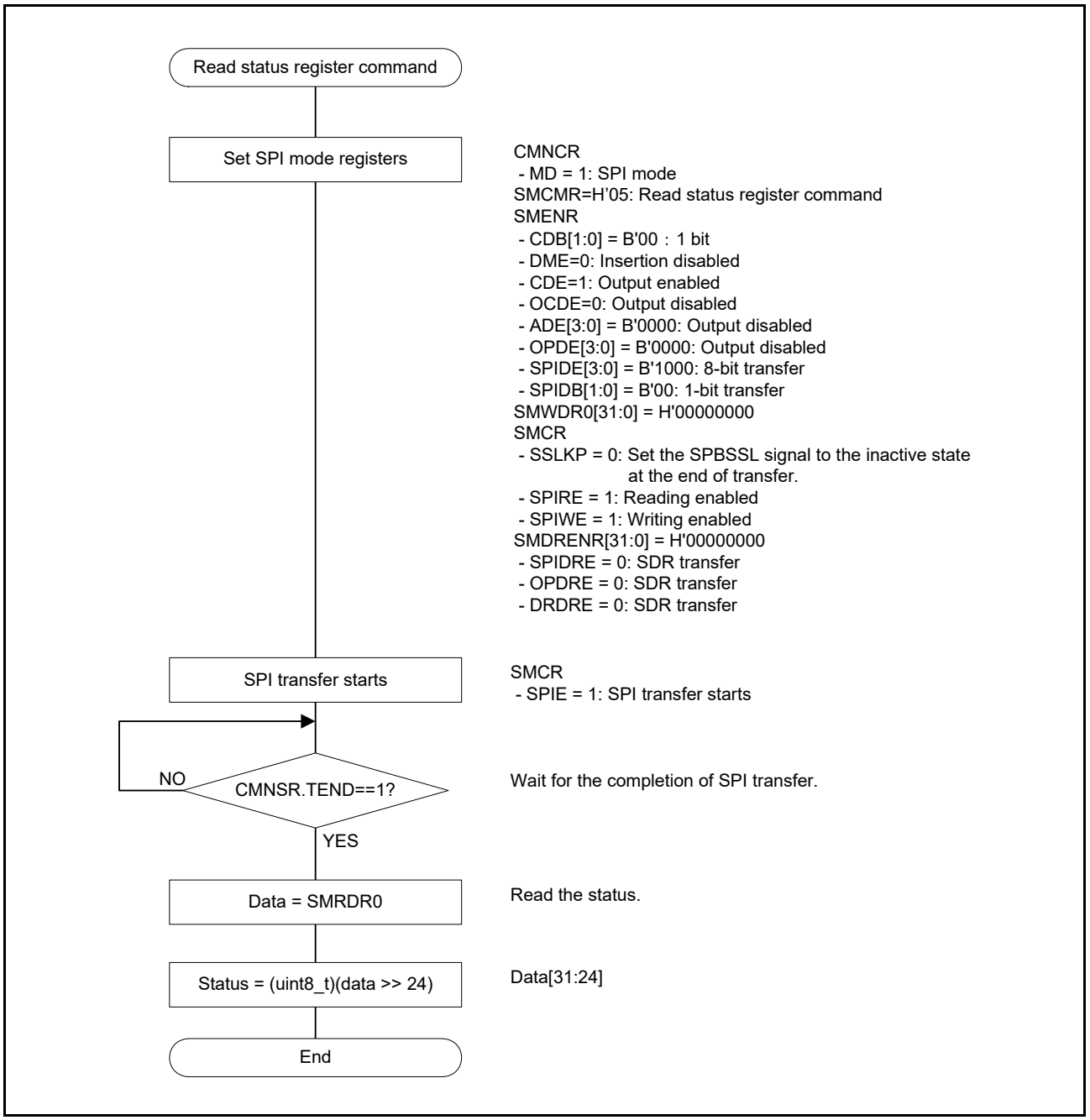


Figure 7.3 Read Status Register Command Flow

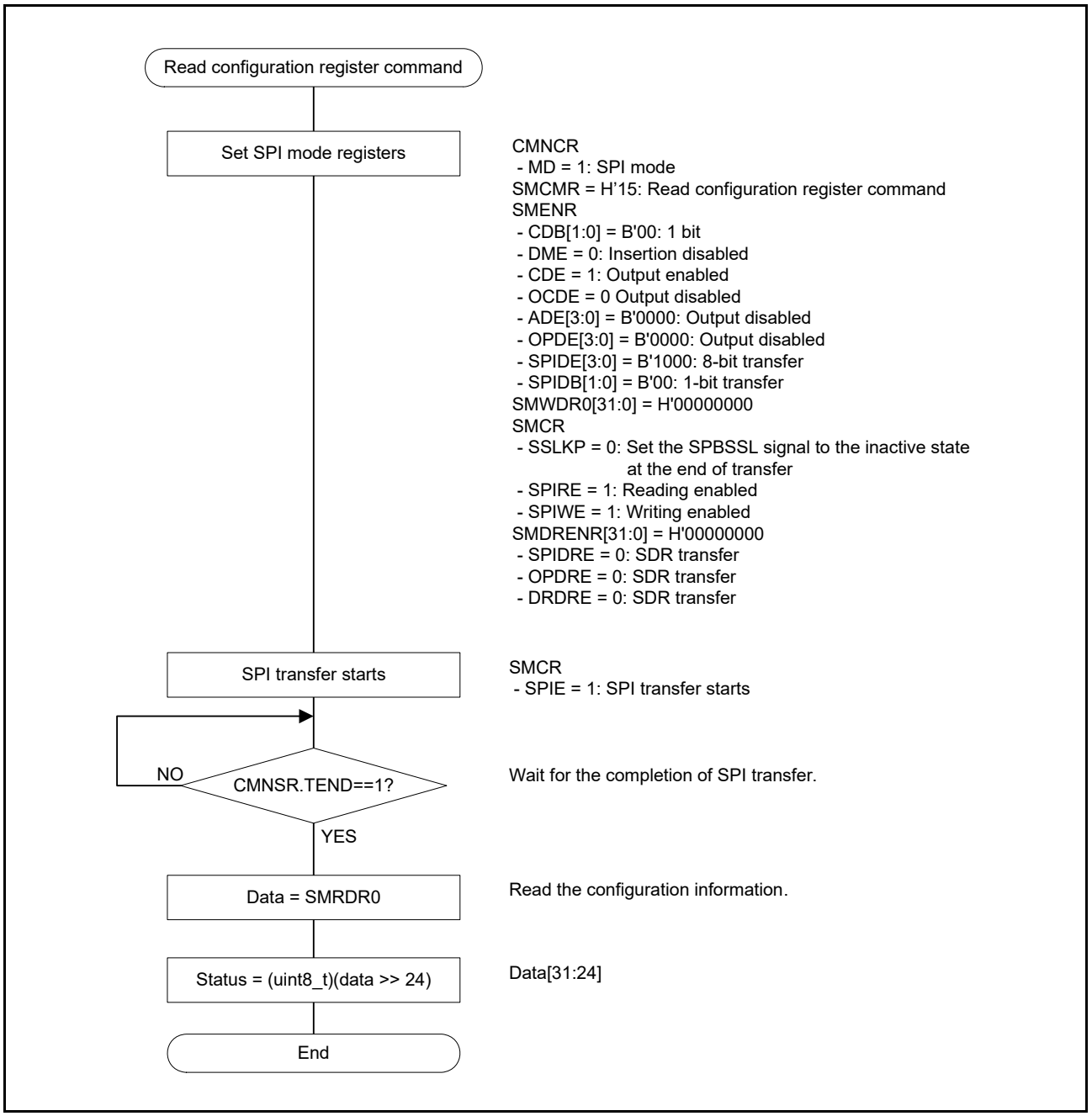


Figure 7.4 Read Configuration Register Command Flow

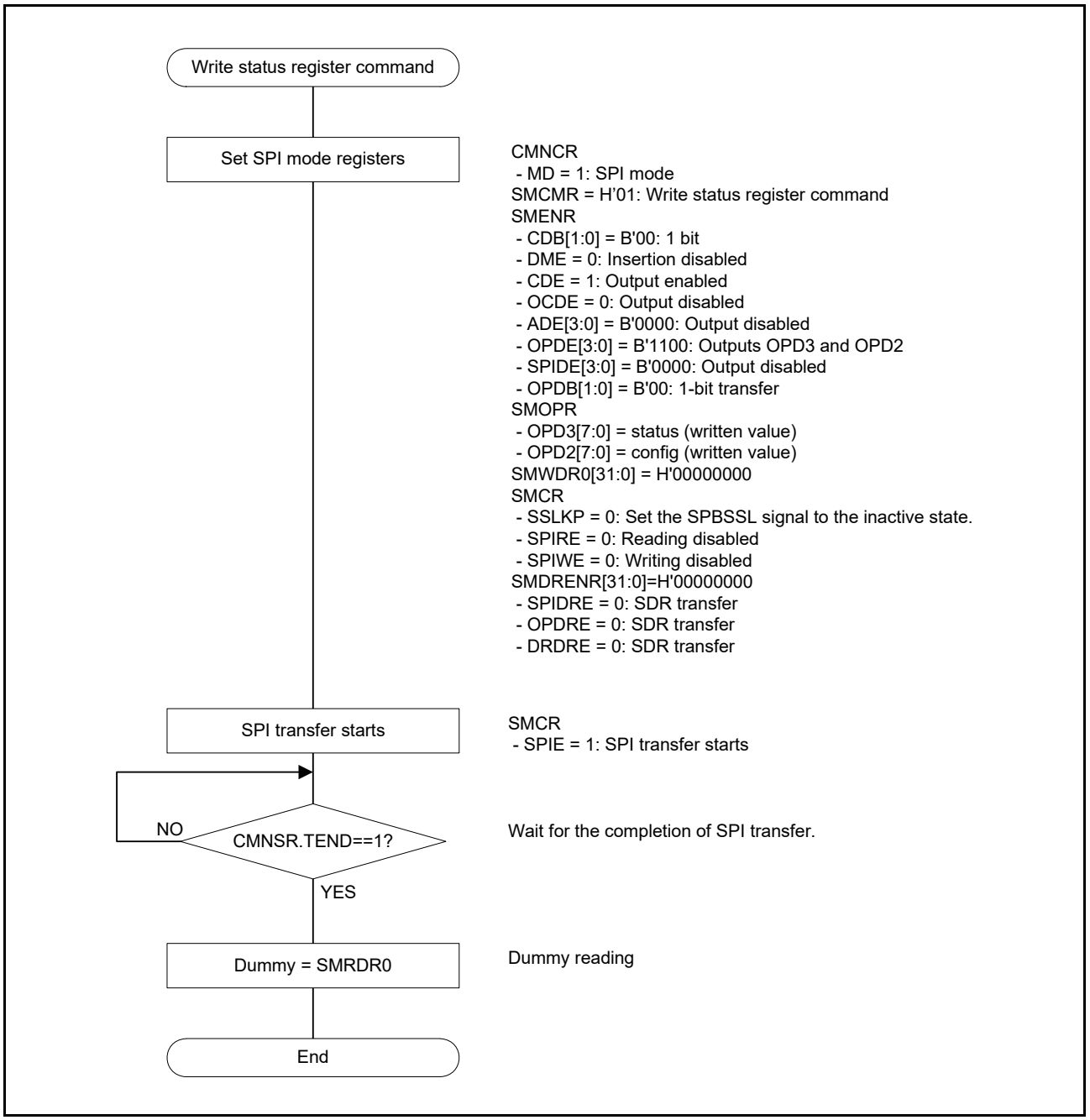


Figure 7.5 Write Status Register Command Flow

7.3.3 Enabling Writing to the Serial Flash Memory

Write the code to enabling writing, which is required for setting the registers in the serial flash memory, as described in Section 7.3.2, Setting Registers in the Serial Flash Memory, in a user-defined function (serial flash memory write enable function: userdef_sflash_write_enable).

Figure 7.6 shows the flow of enabling writing to the serial flash memory in the sample program.

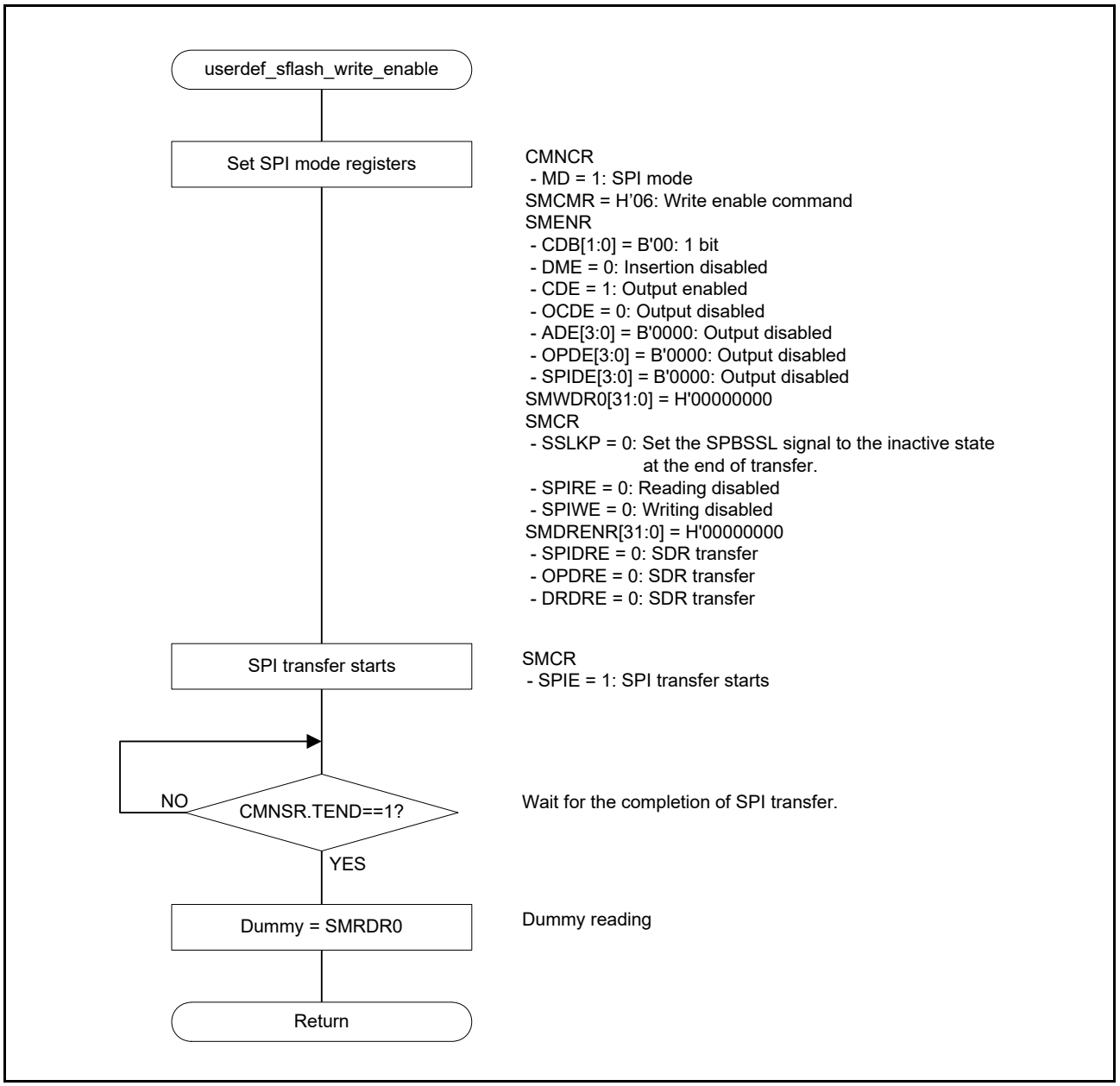


Figure 7.6 Flow of Enabling Writing to the Serial Flash Memory

If the MCU is to be booted up in SPI boot mode, please change the user-defined function Userdef_SPIBSC_Write_Enable for boot processing to reflect the write enable processing which the serial flash memory you are using requires.

7.3.4 Waiting for the Serial Flash Memory to be Ready

Issuing a programming command (page programming) or erase command (sector erase) to the serial flash memory leads to the serial flash memory being placed in the busy state. Write the code to wait for the transition from the busy state to the ready state in a user-defined function (serial flash memory ready wait function: userdef_sflash_busy_wait).

With the Macronix serial flash memory (product type name: MX25L51245G), you can check for a transition to the ready state by reading a register in the serial flash memory.

Figure 7.7 shows the flow of waiting for the serial flash memory to be ready in the sample program.

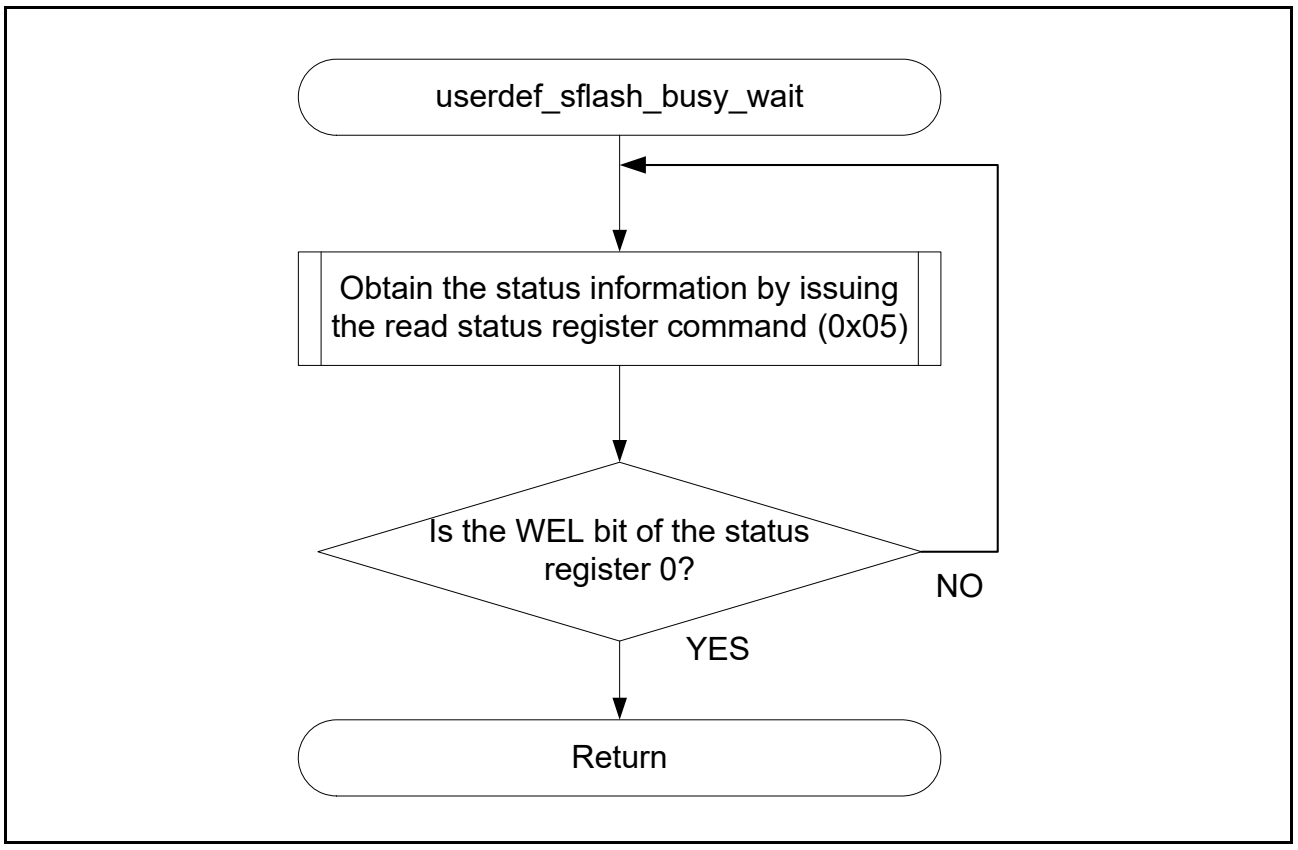


Figure 7.7 Flow of Waiting for the Serial Flash Memory to be Ready

If the MCU is to be booted up in SPI boot mode, please change the user-defined function Userdef_SFLASH_Busy_Wait for boot processing to reflect the ready-wait processing which the serial flash memory you are using requires.

7.3.5 Reference: Releasing the Serial Flash Memory from Protection

When changing the data in the serial flash memory in accord with the specifications of the serial flash memory, it must be released from protection by writing to registers in the serial flash memory. If processing to handle release from protection is required, please include it with reference to the user-defined function Userdef_SFLASH_Ctrl_Protect for SPI boot processing.

With the Macronix serial flash memory (product type name: MX25L51245G), programming or erasure of the serial flash memory cannot proceed if it is in the protected state. To release it from protection, the block protection (BP0, BP1, BP2, and BP3) bits in the status register must be set to 0.

Figure 7.8 shows the flow of releasing the serial flash memory from protection.

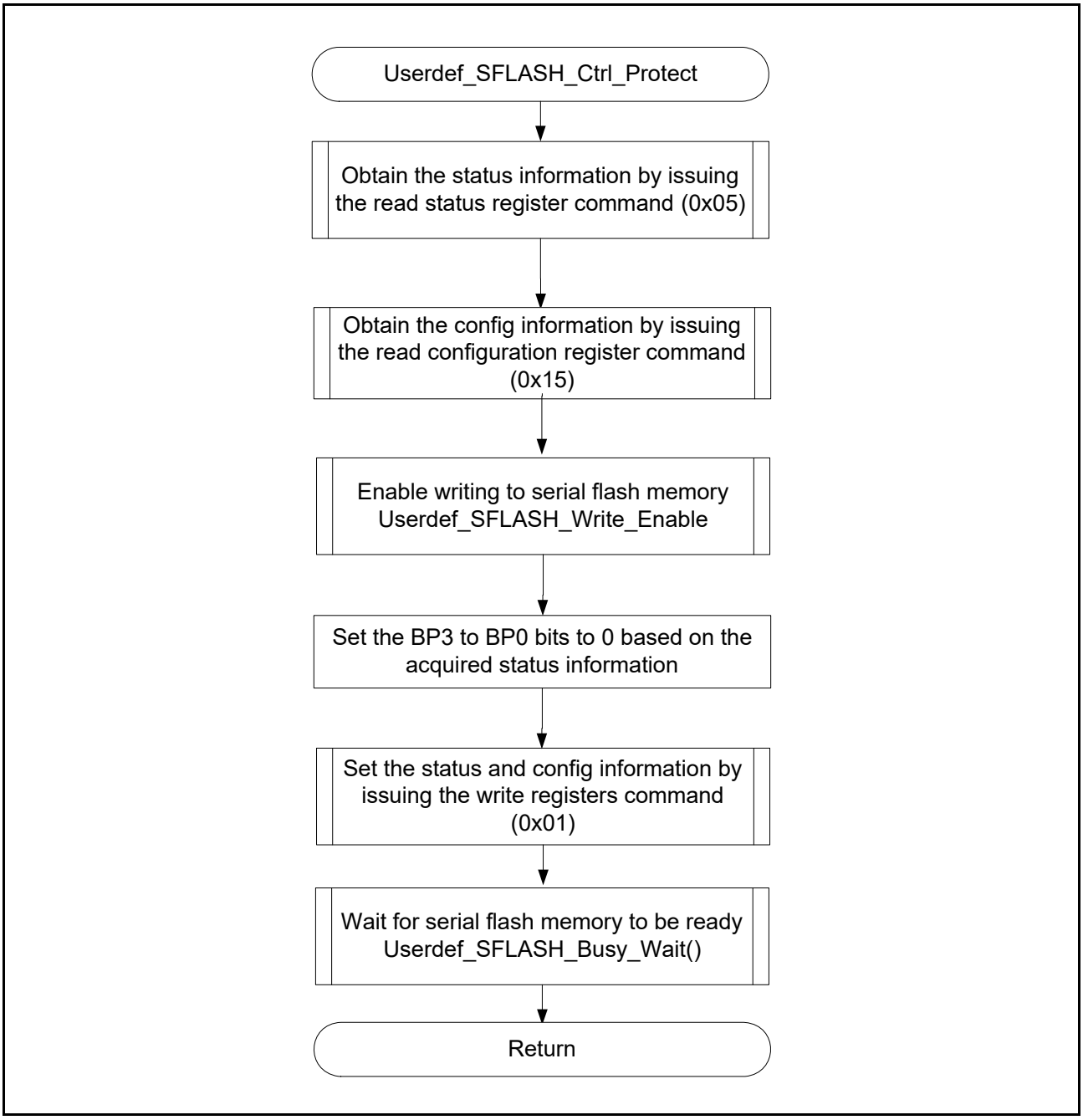


Figure 7.8 Flow of Releasing the Serial Flash Memory from Protection

8. Sample Program

The sample program is available from the Renesas Electronics website.

9. Documents for Reference

- User's Manual: Hardware
  RZ/T1 Group User's Manual: Hardware
  (Download the latest version from the Renesas Electronics website.)

  RZ/T1 Evaluation Board RTK7910022C00000BR User's Manual
  (Download the latest version from the Renesas Electronics website.)

- Technical Update and Technical News
  (Download the latest version from the Renesas Electronics website.)

- User's Manual: Development Environment
  For the IAR integrated development environment (IAR Embedded Workbench® for Arm), visit the IAR Systems website.
  (Download the latest version from the IAR Systems website.)

  For the Arm software development tools (Arm Compiler toolchain, Arm DS-5, etc.), visit the Arm ltd. website.
  (Download the latest version from the Arm ltd. website.)

  For the Renesas Electronics software development tools (e2studio, etc.), visit the Renesas Electronics website.
  (Download the latest version from the Renesas Electronics website.)

Website and Support

Renesas Electronics website

http://www.renesas.com/

Inquiries

http://www.renesas.com/inquiry

Revision History	Application Note: Serial Flash Sample Program (SPIBSC)

Rev.	Date	Description	
		Page	Summary
1.00	Dec. 22, 2015	—	First Edition issued
1.10	Jun. 14, 2016	Front page, header of each page	
		1	Document title changed, restrictions modified
		2. Operating Environment	
		5	Table 2.1 Operating Environment Integrated development environment and Emulator: Version number modified, entries added
		5.1.1 Project Settings	
		10	Section placement information that is changed from the initial setting, added
		5.4 Constants	
		22	Table 5.12 Constants Used in the Sample Program (4), modified, entries added
		5.7 Details of Functions	
		29	R_SPIBSC_ByteRead, Description, modified
		5.9 Operation of the Sample Program	
		34	A point to note added
		5.9.2 Reading the Serial Flash Memory	
		35	Usage example of the input address, partially changed
1.20	Oct. 14, 2016	Header of each page	
		—	Document number, revision number, date of issue modified
		2. Operating Environment	
		5	Table 2.1 Operating Environment, modified
		5.4 Constants	
		20	Table 5.9 Constants Used in the Sample Program (1), size in the description column for SFLASHCMD_SECTOR_ERASE and SFLASHCMD_SECTOR_ERASE_4B modified, Note 1 added
		5.7 Details of Functions	
		28	R_SPIBSC_EraseSector function, description added
1.30	Sep. 14, 2017	2. Operating Environment	
		5	Description modified
		5	Table 2.1 Operating Environment, modified
		4. Description of Hardware	
		8	Figure 4.1 Connection Example: Pin names modified
		5.7 Details of Functions	
		28	R_SPIBSC_ByteProgram: Description added to argument uint32_t addr
		29	R_SPIBSC_ByteRead: Description added to argument uint32_t addr
		29	R_SPIBSC_SpibscTransfer: Tables for reference to the settings for argument st_spibsc_spimd_reg_t * regset corrected
		7.3.1 Changing the Read Command Waveforms	
		40	Name of the user-defined function modified; description to note added
		7.3.2 Setting Registers in the Serial Flash Memory	
		42	Names of the user-defined functions modified
		43	Figure 7.2 Flow of Setting the Registers in the Serial Flash Memory: Names of the user-defined functions modified
		43	Description to note added
		7.3.3 Enabling Writing to the Serial Flash Memory	
		47	Name of the user-defined function modified; description to note added
		47	Figure 7.6 Flow of Enabling Writing to the Serial Flash Memory: Name of the user-defined function modified

Rev.	Date	Description	
		Page	Summary
1.30	Sep.14, 2017	7.3.4 Waiting for the Serial Flash Memory to be Ready	
		48	Name of the user-defined function modified; description to note added
		48	Figure 7.7 Flow of Waiting for the Serial Flash Memory to be Ready: Name of the user-defined function modified
		7.3.5 Reference: Releasing the Serial Flash Memory from Protection	
		49	Description added
1.40	Jun. 07, 2018	2. Operating Environment	
		5	Table 2.1 Operating Environment: The description on the integrated development environment, modified
		9. Documents for Reference	
		51	"ARM" changed to "Arm"

General Precautions in the Handling of Microprocessing Unit and Microcontroller Unit Products

The following usage notes are applicable to all Microprocessing unit and Microcontroller unit products from Renesas. For detailed usage notes on the products covered by this document, refer to the relevant sections of the document as well as any technical updates that have been issued for the products.

1. Handling of Unused Pins

   Handle unused pins in accordance 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 a product with a different part number, confirm that the change will not lead to problems.

   — The characteristics of Microprocessing unit or Microcontroller unit products in the same group but having a different part number may differ in terms of the internal memory capacity, layout pattern, and other factors, which can affect the ranges of electrical characteristics, such as characteristic values, operating margins, immunity to noise, and amount of radiated noise. When changing to a product with a different part number, implement a system-evaluation test for the given product.

Notice

1. 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 or any other use of the circuits, software, and information in the design of your product or system. Renesas Electronics disclaims any and all liability for any losses and damages incurred by you or third parties arising from the use of these circuits, software, or information.
2. Renesas Electronics hereby expressly disclaims any warranties against and liability for infringement or any other claims involving 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, including but not limited to, the product data, drawings, charts, programs, algorithms, and application examples.
3. No license, express, implied or otherwise, is granted hereby under any patents, copyrights or other intellectual property rights of Renesas Electronics or others.
4. You shall not alter, modify, copy, or reverse engineer any Renesas Electronics product, whether in whole or in part. Renesas Electronics disclaims any and all liability for any losses or damages incurred by you or third parties arising from such alteration, modification, copying or reverse engineering.
5. Renesas Electronics products are classified according to the following two quality grades: “Standard” and “High Quality”. The intended applications for each Renesas Electronics product depends on the product’s quality grade, as indicated below.

   "Standard": Computers; office equipment; communications equipment; test and measurement equipment; audio and visual equipment; home electronic appliances; machine tools; personal electronic equipment; industrial robots; etc.

   "High Quality": Transportation equipment (automobiles, trains, ships, etc.); traffic control (traffic lights); large-scale communication equipment; key financial terminal systems; safety control equipment; etc.

   Unless expressly designated as a high reliability product or a product for harsh environments in a Renesas Electronics data sheet or other Renesas Electronics document, Renesas Electronics products are not intended or authorized for use in products or systems that may pose a direct threat to human life or bodily injury (artificial life support devices or systems; surgical implantations; etc.), or may cause serious property damage (space system; undersea repeaters; nuclear power control systems; aircraft control systems; key plant systems; military equipment; etc.). Renesas Electronics disclaims any and all liability for any damages or losses incurred by you or any third parties arising from the use of any Renesas Electronics product that is inconsistent with any Renesas Electronics data sheet, user’s manual or other Renesas Electronics document.
6. When using Renesas Electronics products, refer to the latest product information (data sheets, user’s manuals, application notes, “General Notes for Handling and Using Semiconductor Devices” in the reliability handbook, etc.), and ensure that usage conditions are within the ranges specified by Renesas Electronics with respect to maximum ratings, operating power supply voltage range, heat dissipation characteristics, installation, etc. Renesas Electronics disclaims any and all liability for any malfunctions, failure or accident arising out of the use of Renesas Electronics products outside of such specified ranges.
7. Although Renesas Electronics endeavors to improve the quality and reliability of Renesas Electronics products, semiconductor products have specific characteristics, such as the occurrence of failure at a certain rate and malfunctions under certain use conditions. Unless designated as a high reliability product or a product for harsh environments in a Renesas Electronics data sheet or other Renesas Electronics document, Renesas Electronics products are not subject to radiation resistance design. You are responsible for implementing safety measures to guard against the possibility of bodily injury, injury or damage caused by fire, and/or danger to the public in the event of a failure or malfunction of Renesas Electronics products, 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 and impractical, you are responsible for evaluating the safety of the final products or systems manufactured by you.
8. Please contact a Renesas Electronics sales office for details as to environmental matters such as the environmental compatibility of each Renesas Electronics product. You are responsible for carefully and sufficiently investigating applicable laws and regulations that regulate the inclusion or use of controlled substances, including without limitation, the EU RoHS Directive, and using Renesas Electronics products in compliance with all these applicable laws and regulations. Renesas Electronics disclaims any and all liability for damages or losses occurring as a result of your noncompliance with applicable laws and regulations.
9. Renesas Electronics products and technologies shall 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. You shall comply with any applicable export control laws and regulations promulgated and administered by the governments of any countries asserting jurisdiction over the parties or transactions.
10. It is the responsibility of the buyer or distributor of Renesas Electronics products, or any other party who distributes, disposes of, or otherwise sells or transfers the product to a third party, to notify such third party in advance of the contents and conditions set forth in this document.
11. This document shall not be reprinted, 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.

(Note 1) “Renesas Electronics” as used in this document means Renesas Electronics Corporation and also includes its directly or indirectly controlled subsidiaries.

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

(Rev.4.0-1 November 2017)

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.
1001 Murphy Ranch Road, Milpitas, CA 95035, U.S.A.
Tel: +1-408-432-8888, Fax: +1-408-434-5351

Renesas Electronics Canada Limited
9251 Yonge Street, Suite 8309 Richmond Hill, Ontario Canada L4C 9T3
Tel: +1-905-237-2004

Renesas Electronics Europe Limited
Dukes Meadow, Millboard Road, Bourne End, Buckinghamshire, SL8 5FH, U.K
Tel: +44-1628-651-700, Fax: +44-1628-651-804

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

Renesas Electronics (China) Co., Ltd.
Room 1709 Quantum Plaza, No.27 ZhichunLu, Haidian District, Beijing, 100191 P. R. China
Tel: +86-10-8235-1155, Fax: +86-10-8235-7679

Renesas Electronics (Shanghai) Co., Ltd.
Unit 301, Tower A, Central Towers, 555 Langao Road, Putuo District, Shanghai, 200333 P. R. China
Tel: +86-21-2226-0888, Fax: +86-21-2226-0999

Renesas Electronics Hong Kong Limited
Unit 1601-1611, 16/F., Tower 2, Grand Century Place, 193 Prince Edward Road West, Mongkok, Kowloon, Hong Kong
Tel: +852-2265-6688, Fax: +852 2886-9022

Renesas Electronics Taiwan Co., Ltd.
13F, No. 363, Fu Shing North Road, Taipei 10543, Taiwan
Tel: +886-2-8175-9600, Fax: +886 2-8175-9670

Renesas Electronics Singapore Pte. Ltd.
80 Bendemeer Road, Unit #06-02 Hyflux Innovation Centre, Singapore 339949
Tel: +65-6213-0200, Fax: +65-6213-0300

Renesas Electronics Malaysia Sdn.Bhd.
Unit 1207, 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 India Pvt. Ltd.
No.777C, 100 Feet Road, HAL 2nd Stage, Indiranagar, Bangalore 560 038, India
Tel: +91-80-67208700, Fax: +91-80-67208777

Renesas Electronics Korea Co., Ltd.
17F, KAMCO Yangjae Tower, 262, Gangnam-daero, Gangnam-gu, Seoul, 06265 Korea
Tel: +82-2-558-3737, Fax: +82-2-558-5338

Colophon 7.0