FUJITSU SEMICONDUCTOR
CONTROLLER MANUAL

CM91-10102-2E

FM3

32-BIT MICROCONTROLLER

MB9BF500 Series FLASH PROGRAMMING MANUAL

For the information for microcontroller supports, see the following web site.

http://edevice.fujitsu.com/micom/en-support/

FUJITSU SEMICONDUCTOR LIMITED

ARM™

FUJITSU

Preface

■ Purpose of this manual and intended readers

This manual explains the functions, operations and serial programming of the flash memory of the MB9BF500 series.
This manual is intended for engineers engaged in the actual development of products using the MB9BF500 series.

■ Trademark

ARM and Cortex-M3 are the trademarks of ARM Limited in the EU and other countries.

The company names and brand names herein are the trademarks or registered trademarks of their respective owners.

■ Organization of this Manual

This manual consists of the following 3 chapters.

CHAPTER 1 Flash Memory

This chapter gives an overview of, and explains the structure, operation, and registers of the flash memory.

CHAPTER 2 Flash Security

The flash security feature provides possibilities to protect the content of the flash memory.
This section describes the overview and operations of the flash security.

CHAPTER 3 Serial Programming Connection

This chapter explains the basic configuration for serial write to flash memory by using the Fujitsu Semiconductor Serial Programmer.

■ Sample programs and development environment

Fujitsu Semiconductor offers sample programs free of charge for using the peripheral functions of the FM3 family. Fujitsu Semiconductor also makes available descriptions of the development environment required for the MB9BFxxx series. Feel free to use them to verify the operational specifications and usage of this Fujitsu Semiconductor microcontroller.

- Microcontroller support information:
  http://edevice.fujitsu.com/micom/en-support/

* : Note that the sample programs are subject to change without notice. Since they are offered as a way to demonstrate standard operations and usage, evaluate them sufficiently before running them on your system.
Fujitsu Semiconductor assumes no responsibility for any damage that may occur as a result of using a sample program.

- The contents of this document are subject to change without notice.
  Customers are advised to consult with sales representatives before ordering.
- The information, such as descriptions of function and application circuit examples, in this document are presented solely for the purpose of reference to show examples of operations and uses of FUJITSU SEMICONDUCTOR device; FUJITSU SEMICONDUCTOR does not warrant proper operation of the device with respect to use based on such information. When you develop equipment incorporating the device based on such information, you must assume any responsibility arising out of such use of the information. FUJITSU SEMICONDUCTOR assumes no liability for any damages whatsoever arising out of the use of the information.
- Any information in this document, including descriptions of function and schematic diagrams, shall not be construed as license of the use or exercise of any intellectual property right, such as patent right or copyright, or any other right of FUJITSU SEMICONDUCTOR or any third party or does FUJITSU SEMICONDUCTOR warrant non-infringement of any third-party's intellectual property right or other right by using such information. FUJITSU SEMICONDUCTOR assumes no liability for any infringement of the intellectual property rights or other rights of third parties which would result from the use of information contained herein.
- The products described in this document are designed, developed and manufactured as contemplated for general use, including without limitation, ordinary industrial use, general office use, personal use, and household use, but are not designed, developed and manufactured as contemplated (1) for use accompanying fatal risks or dangers that, unless extremely high safety is secured, could have a serious effect to the public, and could lead directly to death, personal injury, severe physical damage or other loss (i.e., nuclear reaction control in nuclear facility, aircraft flight control, air traffic control, mass transport control, medical life support system, missile launch control in weapon system), or (2) for use requiring extremely high reliability (i.e., submersible repeater and artificial satellite).
  Please note that FUJITSU SEMICONDUCTOR will not be liable against you and/or any third party for any claims or damages arising in connection with above-mentioned uses of the products.
- Any semiconductor devices have an inherent chance of failure. You must protect against injury, damage or loss from such failures by incorporating safety design measures into your facility and equipment such as redundancy, fire protection, and prevention of over-current levels and other abnormal operating conditions.
- Exportation/release of any products described in this document may require necessary procedures in accordance with the regulations of the Foreign Exchange and Foreign Trade Control Law of Japan and/or US export control laws.
- The company names and brand names herein are the trademarks or registered trademarks of their respective owners.

How to Use This Manual

■ Finding a function

The following methods can be used to search for the explanation of a desired function in this manual:

- Search from the table of the contents
  The table of the contents lists the manual contents in the order of description.

- Search from the register
  The address where each register is located is not described in the text. To verify the address of a register, see "APPENDIX Register Map" of "Peripheral Manual".

■ Terminology

This manual uses the following terminology.

Term	Explanation
Word	Indicates access in units of 32 bits.
Half word	Indicates access in units of 16 bits.
Byte	Indicates access in units of 8 bits.

■ Notations

- The notations in bit configuration of the register explanation of this manual are written as follows.
  - bit : bit number
  - Field : bit field name
  - Attribute : Attributes for read and write of each bit
    - R : Read only
    - W : Write only
    - R/W : Readable/Writable
    - - : Undefined
  - Initial value : Initial value of the register after reset
    - 0 : Initial value is "0"
    - 1 : Initial value is "1"
    - X : Initial value is undefined

- The multiple bits are written as follows in this manual.
  Example : bit7:0 indicates the bits from bit7 to bit0

- The values such as for addresses are written as follows in this manual.
  - Hexadecimal number : "0x" is attached in the beginning of a value as a prefix (example : 0xFFFF)
  - Binary number : "0b" is attached in the beginning of a value as a prefix (example: 0b1111)
  - Decimal number : Written using numbers only (example : 1000)

CONTENTS

MAIN CHANGES IN THIS EDITION

Page	Section	Change Results
-	-	· Revised the corporate names · fm3 → FM3
33-36	CHAPTER 3 Serial Programming Connection	Revised the corporate name in this section.

CHAPTER: Flash Memory

This chapter gives an overview of, and explains the structure, operation, and registers of the flash memory. The MB9BF500 has built-in flash memory with a capacity of 256 KBytes that supports erasing by all-sector batch erase and per-sector erase, and that can be programmed by the CPU.

1. Overview

The MB9BF500 is equipped with 256 KBytes of built-in flash memory.

The built-in flash memory can be erased sector-by-sector, all-sector batch erased, and programmed in units of half words (16 bits) by the Cortex-M3 CPU.

This flash memory also has built-in ECC (Error Correction Code) functionality.

■ Flash Memory Features

- Usable capacity:
  MB9BF500: 256 Kbytes
  Note: Because this model stores ECC codes, it is equipped with additional flash memory of 6 bits for every 4 bytes of memory described above.

- High-speed flash:
  At 80 MHz: 2-wait (During address continuous access: 0-wait)
  At 60 MHz: 0-wait

- Operating mode:
  1. CPU-ROM mode
     This mode only allows reading of flash memory data. Word access is available. However, in this mode, it is not possible to activate the automatic programming algorithm[*1] to perform writing or erasing.
  2. CPU programming mode
     This mode allows reading, writing, and erasing of flash memory (automatic programming algorithm[*1]). Because word access is not available, programs that are contained in the flash memory cannot be executed while operating in this mode. Half-word access is available.
  3. ROM writer mode
     This mode allows reading, writing, and erasing of flash memory from a ROM writer (automatic programming algorithm[*1]).

- Built-in flash security function
  (Prevents reading of the content of flash memory by a third party)
  See "CHAPTER Flash Security" for details on the flash security function.

- Equipped with an Error Correction Code (ECC) function that can correct up to 1 bit of errors in each word.
  (The device is not equipped with a function to detect 2-bit errors.) Errors are automatically corrected when memory is read.
  Furthermore, ECC codes are automatically added upon writing to flash memory. Because there are no read cycle penalties as a result of error correction, there is no need to give any consideration to read correction penalties during software development.

<Note>

This document explains flash memory in the case where it is being used in CPU mode.

For details on accessing the flash memory from a ROM writer, see the instruction manual of the ROM writer that is being used.

*1: Automatic programming algorithm=Embedded Algorithm

2. Configuration

This model consists of a 256-Kbyte flash memory region, a security code region, and a built-in CR trimming data region.

Figure 2-1 shows the address and sector structure of the flash memory built into the MB9BF500, Figure 2-2 shows the address of security/CR trimming data.

Note: See "CHAPTER Flash Security" for details on the security.
Note: See "CHAPTER Clock" of the Peripheral Manual for details on the Built-in High-Speed CR trimming.

Figure 2-1 Memory map of flash memory

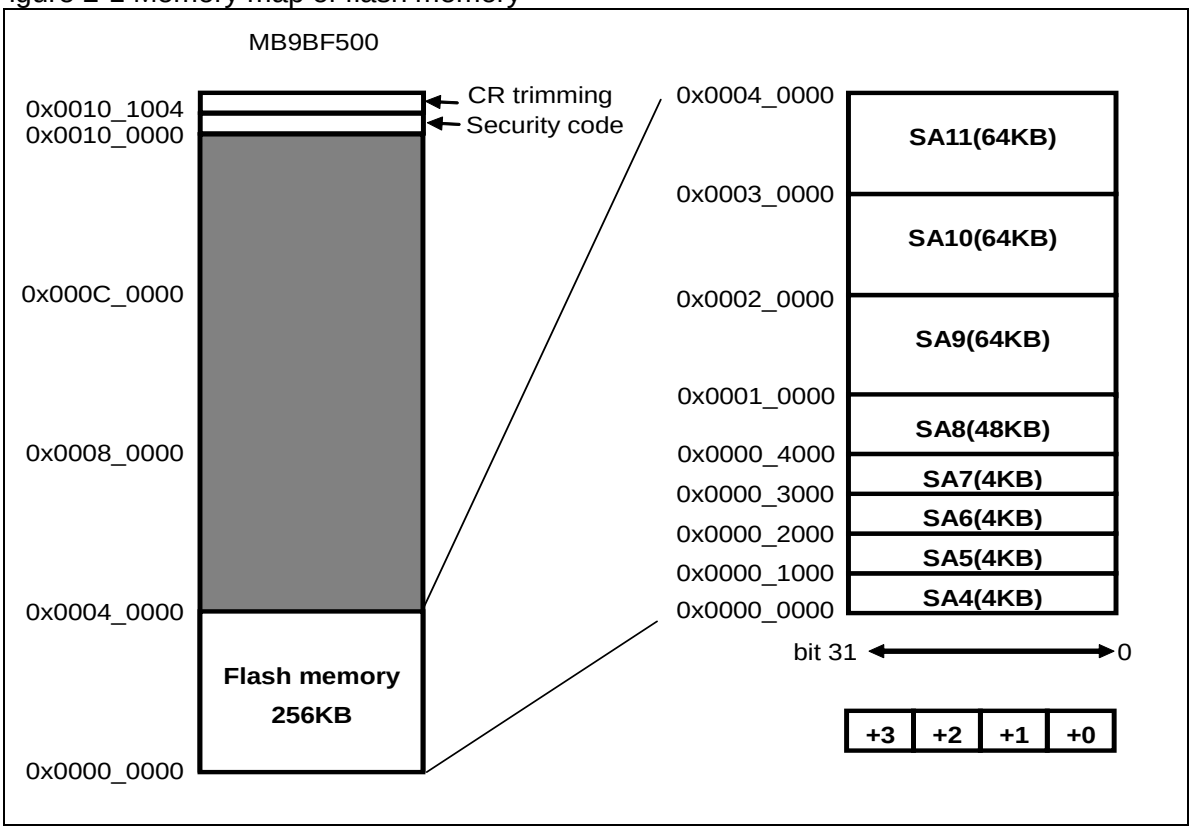
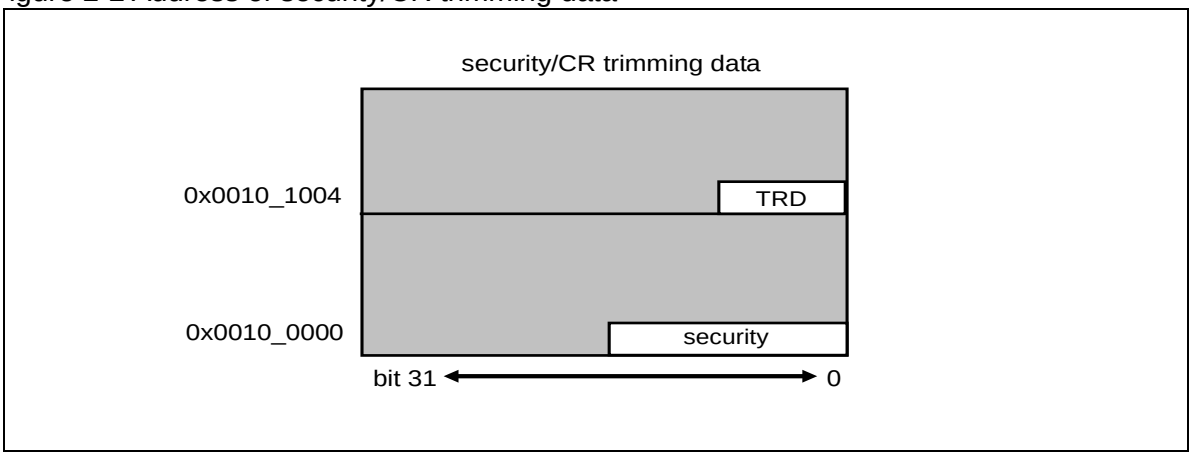


Figure 2-2 Address of security/CR trimming data



3. Operating Description

This section explains the operating description.

3.1. Flash Memory Access Modes

The following two access modes are available for accessing flash memory from the CPU.

· CPU ROM mode
· CPU programming mode

These modes are selected by the flash access size bits (FASZR: ASZ).

■ CPU ROM Mode

This mode only allows reading of flash memory data.

This mode is entered by setting the flash access size bits (FASZR: ASZ) to "10" (32-bit read), and enables word access.

However, in this mode, it is not possible to execute commands, to activate the automatic programming algorithm or to write or erase data.

The flash memory always enters this mode after reset is released.

■ CPU Programming Mode

This mode allows reading, writing, and erasing of data.

This mode is entered by setting the flash access size bits (FASZR: ASZ) to "01" (16-bit read/write), and enables flash programming.

Because word access is not possible in this mode, programs that are contained in the flash memory cannot be executed. The operation while in this mode is as follows.

· During reading
  Flash memory is accessed in half-words, with data read out in blocks of 16 bits.

· During writing commands
  The automatic programming algorithm can be activated to write or erase data. See Section "3.2 Automatic Programming Algorithm" for details on the automatic programming algorithm.

Table 3-1 Access modes of Flash memory

Access Mode	Access Size	Automatic programming algorithm	Instruction execution in the Flash Memory
CPU ROM mode	32bit	disable	enable
CPU programming mode	16bit	enable	Prohibited

<Note>

The flash memory is always set to CPU ROM mode when a reset is released. Therefore, if a reset occurs after entering CPU programming mode, the flash access size bits (FASZR: ASZ) are set to "10" and the flash memory returns to CPU ROM mode.

3.2. Automatic Programming Algorithm

When CPU programming mode is used, writing to and erasing flash memory is performed by activating the automatic programming algorithm.

This section explains the automatic programming algorithm.

3.2.1. Command Sequences

The automatic programming algorithm is activated by sequentially writing half-word (16-bit) data to the flash memory one to six times in a row. This is called a command. Table 3-2 shows the command sequences.

Table 3-2 Command sequence chart

Command	Number of writes	1st write		2nd write		3rd write		4th write		5th write		6th write	
		Address	Data	Address	Data	Address	Data	Address	Data	Address	Data	Address	Data
Read/ Reset	1	0xXXX	0xF0	RA	RD	--	--	--	--	--	--	--	--
Read/ Reset	4	0xAA8	0xAA	0x554	0x55	0xAA8	0xF0	RA	RD	--	--	--	--
Write	4	0xAA8	0xAA	0x554	0x55	0xAA8	0xA0	PA	PD	--	--	--	--
Chip erase	6	0xAA8	0xAA	0x554	0x55	0xAA8	0x80	0xAA8	0xAA	0x554	0x55	0xAA8	0x10
Sector erase	6	0xAA8	0xAA	0x554	0x55	0xAA8	0x80	0xAA8	0xAA	0x554	0x55	SA	0x30

X: Any value
RA: Read address
PA: Write address
SA: Sector address (Specify any address within the address range of the sector to erase)
RD: Read data
PD: Write data

<Notes>

- The data notation in the table only shows the lower 8 bits. The upper 8 bits can be set to any value.
- Write commands as half-words at any time.
- The address notation in the table only shows the lower 12 bits. The upper 20 bits should be set to any address within the address range of the target flash memory. When the address outside the flash address range is specified, the command sequence doesn't move correctly because the flash memory cannot recognize the command

3.2.2. Command Operating Explanations

This section explains the command operating.

■ Read/Reset Command

The flash memory can be read and reset by sending the read/reset command to the target sector in sequence.

Although there is a read/reset command with only one write cycle, and a read/reset command with four write cycles, these are essentially the same.

When a read/reset command is issued, the flash memory maintains the read state until another command is issued.

When the execution of the automatic programming algorithm exceeds the time limit, the flash memory is returned to the read/reset state by issuing the read/reset command. Read data from the flash memory during the read cycle.

See Section "3.3.1 Read/Reset Operation" for details on the actual operation.

■ Program (Write) Command

The automatic programming algorithm can be activated and the data is written to the flash memory by sending the write command to the target sector in four consecutive writes. Data writes can be performed in any order of addresses, and may also cross sector boundaries.

In CPU programming mode, data is written in half-words.

Once the fourth sequential write has finished, the automatic programming algorithm is activated and the automatic write to the flash memory starts. After executing the automatic write algorithm command sequence, there is no need to control the flash memory externally.

See Section "3.3.2 Write Operation" for details on the actual operation.

<Notes>

- The write is not performed properly if the fourth write command (write data cycle) is written to an odd address. Always write to an even address.
- Only a single half-word of data can be written for each write command sequence.
  To write multiple pieces of data, issue one write command sequence for each piece of data.

■ Chip Erase Command

All of the sectors in flash memory can be batch-erased by sending the chip erase command to the target sector in six consecutive writes. Once the sixth sequential write has finished, the automatic programming algorithm is activated and the chip erase operation starts.

Because the flash memory writes "0" to all cells within the chip and verifies the margin (preprogramming) before erasing the entire chip when the automatic erase algorithm is activated there is no need to write to the flash memory before erasing the chip.

Furthermore, there is no need to control the flash memory externally during the margin verification operation.

See Section "3.3.3 Chip Erase Operation" for details on the actual operation.

■ Sector Erase Command

A single sector of flash memory can be erased by sending the sector erase command to the target sector in six consecutive writes. Once the sixth sequential write has finished and 40 µs has elapsed (timeout interval), the automatic programming algorithm is activated and the sector erase operation begins.

To erase multiple sectors, write the erase code (0x30) to the address of the sector to erase within 40 µs (timeout interval). If the next sector is not input within the timeout interval, the sector erase command may become inactive.

When the automatic erase algorithm is activated, the flash memory writes "0" to all cells within the sector and verifies the margin (preprogramming) before erasing the sector. Therefore, there is no need to write to the flash memory before erasing the sector.

Furthermore, there is no need to control the flash memory externally during the margin verification operation.

See Section "3.3.4 Sector Erase Operation" for details on the actual operation.

3.2.3. Automatic Programming Algorithm Run States

Because writing and erasing of flash memory is performed by the automatic programming algorithm, whether or not the automatic programming algorithm is currently executing can be checked using the flash ready bit (FSTR: RDY) and the operating status can be checked using the hardware sequence flags.

■ Hardware Sequence Flags

These flags indicate the status of the automatic programming algorithm. When the flash ready bit (FSTR: RDY) is "0", the operating status can be checked by reading any address in flash memory.

Figure 3-1 shows the bit structure of the hardware sequence flags.

Figure 3-1 Bit structure of the hardware sequence flags

In the event of half-word access

bit	15	14	13	12	11	10	9	8
	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined

bit	7	6	5	4	3	2	1	0
	DPOL	TOGG	TLOV	Undefined	SETI	Undefined	Undefined	Undefined

In the event of byte access

bit	7	6	5	4	3	2	1	0
	DPOL	TOGG	TLOV	Undefined	SETI	Undefined	Undefined	Undefined

<Notes>

- These flags cannot be read using word access. When in CPU programming mode, always read using half-word or byte access.
- In CPU ROM mode, the hardware sequence flags cannot be read no matter which address is read.
- Because the correct value might not be read out after issuing a command, ignore the first value of the hardware sequence flags that is read after issuing a command.

● Status of each bit and flash memory

Table 3-3 shows the correspondence between each bit of the hardware sequence flags and the status of the flash memory.

Table 3-3 List of hardware sequence flag states

State				DPOL	TOGG	TLOV	SETI
Running	Write operation			Inverted data (*1)	Toggle	0	0
	Erase operation	Chip erase		0	Toggle	0	1
		Sector erase	timeout interval	1	Toggle	0	0
			erase	0	Toggle	0	1
Time limit exceeded	Write operation			Inverted data (*1)	Toggle	1	0
	Sector/chip erase			0	Toggle	1	1

*1 See "● Bit Descriptions" for the values that can be read.

● Bit Descriptions

[bit15:8] Undefined bits

[bit7] DPOL: Data polling flag bit

When the hardware sequence flags are read, by specifying an arbitrary address, this bit uses a data polling function to indicate whether or not the automatic programming algorithm is currently running.

The value that is read out varies depending on the operating state.

- During writing
  - While write is in progress:
    Reads out the opposite value (inverse data) of the value of bit 7 of the data that was last written.
    This does not access the address that was specified for reading the hardware sequence flags.
  - After write finishes:
    Reads out the value of bit 7 of the address specified for reading the hardware sequence flags.
- During sector erase
  - While sector erase is executing:
    Reads out "0" from the sector that is currently being erased.
  - After sector erase finishes:
    Always reads out "1".
- During chip erase
  - While chip erase is executing: Always reads out "0".
  - After chip erase: Always reads out "1".

<Note>

The data for a specified address cannot be read while the automatic programming algorithm is running. Confirm that the automatic programming algorithm has finished running by using this bit before reading data.

[bit6] TOGG: Toggle Flag Bit

When the hardware sequence flags are read by specifying an arbitrary address, this bit indicates whether or not the automatic programming algorithm is currently running.

The value that is read out varies depending on the operating state.

- During write, sector erase, or chip erase
  - While write, sector erase, or chip erase is in progress:
    When this bit is read out continuously, it alternatingly returns "1" and "0" (toggles). The address that was specified for reading the hardware sequence flags is not accessed.
  - After write, sector erase, or chip erase has finished:
    Reads out the value of bit 6 of the address specified for reading the hardware sequence flags.

[bit5] TLOV: Timing Limit Exceeded Flag Bit

When the hardware sequence flags are read by specifying an arbitrary address, this bit indicates whether or not the execution time of the automatic programming algorithm has exceeded the rated time defined internally within the flash memory (number of internal pulses).

The value that is read out varies depending on the operating state.

- During write, sector erase, or chip erase
  The following values are read out.

  0: Within the rated time
  1: Rated time exceeded

  When this bit is "1", if the DPOLL bit and TOGG bit indicate that the automatic programming algorithm is currently executing, that means a failure occurred during the write or erase.

  For example, because data that has been written to "0" cannot be overwritten to "1" in flash memory, if "1" is written to an address that has been written to "0", the flash memory is locked and the automatic programming algorithm does not finish. In this case, the value of the DPOLL bit remains invalid, and "1" and "0" are continuously read out alternatingly from the TOGG bit.

  Once the rated time is exceeded while still in this state, this bit changes to "1". If this bit changes to "1", issue the reset command.

<Note>

If this bit is "1", it indicates that the flash memory was not used correctly. This is not a malfunction of the flash memory.

Perform the appropriate processing after issuing the reset command.

[bit4] Undefined bit

[bit3] SETI: Sector Erase Timer Flag Bit

When a sector is erased, a timeout interval of 40 μs is required from when the sector erase command is issued until the sector erase actually begins.

When the hardware sequence flags are read by specifying an arbitrary address, this bit indicates whether or not the flash memory is currently in the sector erase command timeout interval.

The value that is read out varies depending on the operating state.

- During sector erase:
  When sectors are being erasing, it can be checked whether or not the following sector erase code can be accepted by checking this bit before inputting the following sector erase code.

  The following values are read out without accessing the address specified in order to read the hardware sequence flags.

  0: Within sector erase wait interval
  The following sector erase code (0x30) can be accepted.

  1: Sector erase wait interval exceeded
  In this case, if the DPOLL bit and TOGG bit indicate that the automatic programming algorithm is currently executing, the erase operation has started internally within the flash memory. In this case, commands other than the sector erase code (0x30) are ignored until the internal flash memory erase operation has finished.

[bit2:0] Undefined bits

3.3. Explanation of Flash Memory Operation

The operation of the flash memory is explained for each command.

3.3.1. Read/Reset Operation

This section explains the read/reset operation.

The flash memory can be put into the read/reset state by sending the read/reset command to the target sector sequentially.

Because the read/reset state is the default state of the flash memory, the flash memory always returns to this state when the power is turned on or when a command finishes successfully. When the power is turned on, there is no need to issue a data read command. Furthermore, because data can be read by normal read access and programs can be accessed by the CPU while in the read/reset state, there is no need to issue read/reset commands.

3.3.2. Write Operation

This section explains the write operation.

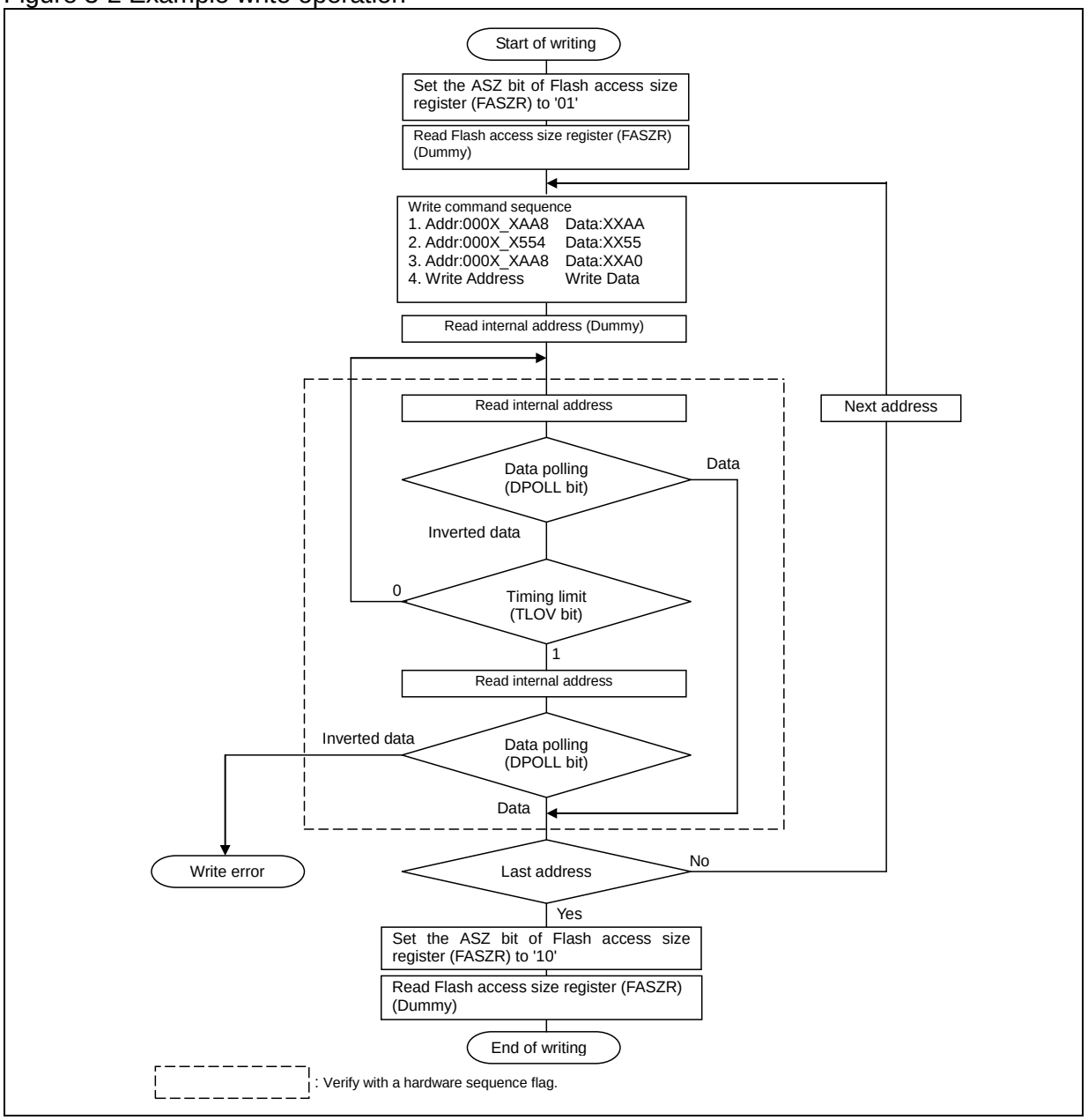
Writes are performed according to the following procedure.

1. The write command is sent to the target sector sequentially
   The automatic programming algorithm activates and the data is written to the flash memory.
   After the write command is issued, there is no need to control the flash memory externally.

2. Perform read access on the address that was written
   The data that is read is the hardware sequence flags. Therefore, once bit 7 (the DPOLL bit) of the read data matches the value that was written, the write to the flash memory has finished. If the write has not finished, the reverse value (inverted data) of the value of bit 7 of the data that was written last is read out.

Figure 3-2 shows an example of a write operation to the flash memory.

Figure 3-2 Example write operation



<Notes>

- Because the flash memory returns to read mode once the write has finished, write addresses are no longer accepted.
- See Section "3.2 Automatic Programming Algorithm" for details on the write command.
- Because the value of the DPOLL bit of the hardware sequence flags changes at the same time as the TLOV bit, the value needs to be checked again even if the TLOV bit is "1".
- The toggle operation stops at the same time as the TOGG bit and TLOV bit of the hardware sequence flags change to "1". Therefore, even if the TLOV bit is "1", the TOGG bit needs to be checked again.
- Although the flash memory can be written in any sequence of addresses regardless of crossing sector boundaries, only a single half-word of data can be written with each write command sequence. To write multiple pieces of data, issue one write command sequence for each piece of data.
- All commands written to the flash memory during the write operation are ignored.
- If the device is reset while the write is in progress, the data that is written is not guaranteed.
- Because ECC bits are added in this model, writes are always required to be performed in units of 32 bits by using two 16-bit writes. See Section "3.4 Writing to Flash Memory in Products Equipped with ECC" for details on the procedure.

3.3.3. Chip Erase Operation

This section explains the chip erase operation.

All sectors in flash memory can be erased in one batch. Erasing all of the sectors in one batch is called chip erase.

The automatic programming algorithm can be activated and all of the sectors can be erased in one batch by sending the chip erase command sequentially to the target sector.

See Section "3.2 Automatic Programming Algorithm" for details on the chip erase command.

1. Send the chip erase command sequentially to the target sector
   The automatic programming algorithm is activated and the data is written to the flash memory.

2. Perform read access to an arbitrary address
   The data that is read is the hardware sequence flag. Therefore, if the value of bit 7 (the DPOLL bit) of the data that was read is "1", that means t the chip erase has finished.

The time required to erase the chip is "sector erase time x total number of sectors + chip write time (preprogramming)".

Once the chip erase operation has finished, the flash memory returns to read/reset mode.

<Note>

Because the flash memory writes "0" to all of the cells in the chip and verifies the margin (preprogramming) before erasing the entire chip when the automatic erase algorithm is activated, there is no need to write to the flash memory before erasing the chip.

Furthermore, there is no need to control the flash memory externally during the margin verification.

3.3.4. Sector Erase Operation

This section explains the sector erase operation.

Sectors in the flash memory can be selected and the data of only the selected sectors can be erased. Multiple sectors can be specified at the same time.

Sectors are erased according to the following sequence.

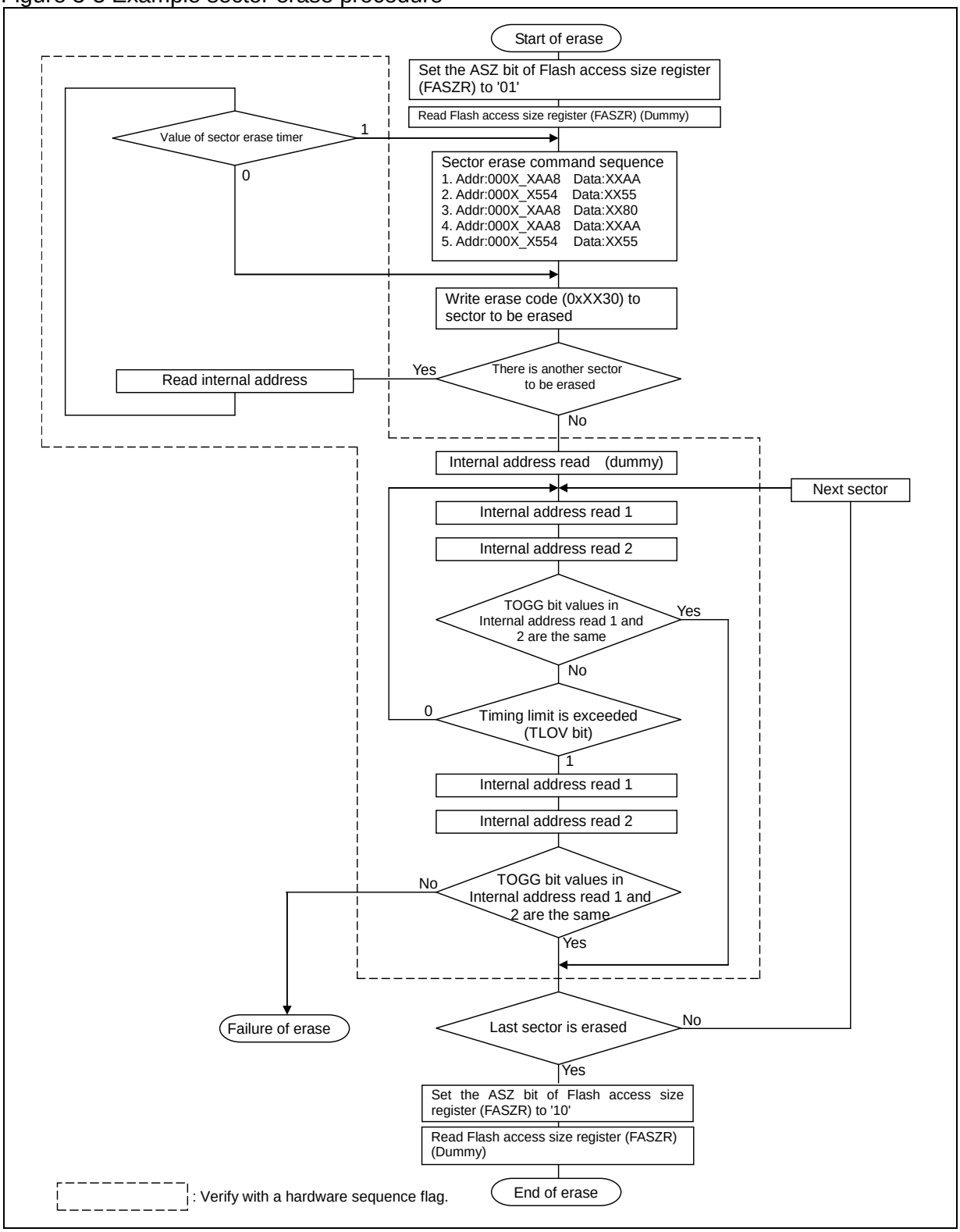
1. Send the sector erase command sequentially to the target sector
   Once 40 μs has elapsed (the timeout interval), the automatic programming algorithm activates and the sector erase operation begins.

   To erase multiple sectors, write the erase code (0x30) to an address in the sector to erase within 40 μs (the timeout interval). If the code is written after the timeout interval has elapsed, the sector erase command may be invalid.

2. Perform read access to an arbitrary address
   The data that is read is the hardware sequence flags. Therefore, if the value of bit 7 (the DPOLL bit) of the data that was read is "1", that means the sector erase has finished.

   Furthermore, it can be checked whether or not the sector erase has finished by using the TOGG bit. Figure 3-3 shows an example of the sector erase procedure for the case of using the TOGG bit for confirmation.

Figure 3-3 Example sector erase procedure



The time required to erase a sector is "(sector erase time + sector write time (preprogramming)) × number of sectors".

Once the sector erase operation has finished, the flash memory returns to read/reset mode.

<Notes>

- See Section "3.2 Automatic Programming Algorithm" for details on the sector erase command.
- Because the value of the DPOLL bit of the hardware sequence flags changes at the same time as the TLOV bit, the value needs to be checked again even if the TLOV bit is "1".
- The toggle operation stops at the same time as the TOGG bit and TLOV bit of the hardware sequence flags change to "1". Therefore, even if the TLOV bit is "1", the TOGG bit needs to be checked again.
- If a command other than a sector erase command or erase pause command is issued during the sector erase, including the timeout interval, the flash memory returns to the read/reset state.
  In this event, because the flash memory is reset, the one or more sector erase commands that were issued before this command are invalid.
  To erase the sectors, issue the sector erase commands from the beginning again.
- Because the flash memory writes "0" to all of the cells being erased and verifies the margin (preprogramming) before erasing the sectors when the automatic erase algorithm activates, there is no need to write to the flash memory before erasing the sectors.
  Furthermore, there is no need to control the flash memory externally during the margin verification.

3.4. Writing to Flash Memory in Products Equipped with ECC

This section explains the writing to flash memory in products equipped with ECC.

Because ECC (Error Correction Codes) are attached to each word in this device, writes need to be performed in blocks of words. Write the data one word at a time by writing two half-words consecutively using the following procedure. If this procedure is not followed, the data is written to the flash memory without calculating the ECC, and the written data will not be read correctly.

1. Set the flash access size setting to 16 bits. (FASZR: ASZ="01")
   Perform a dummy read, after setting the FASZR register.
2. Issue a write command. Write address = PA Write data = PD[15:0]
   See Section "3.3.2 Write Operation" for details on the write command.
3. Read the hardware sequence flags once. Because the correct value might not be read out immediately after issuing a command, this read value should be ignored.
4. Read the hardware sequence flags until the write has finished.
   See Section "3.2.3 Automatic Programming Algorithm Run States" for details on reading the hardware sequence flags.
5. Issue a write command. Write address = PA+2 Write data = PD[31:16]
   At this time, the hardware automatically calculates the ECC codes together with PD[15:0] from step 2, and also automatically writes the ECC codes at the same time.
6. Read the hardware sequence flags once. Because the correct value might not be read out immediately after issuing a command, this read value should be ignored.
7. Read the hardware sequence flags until the write has finished.
8. If there is more write data, return to step 2. Once finished writing all of the data, proceed to step 9.
9. Switch to CPU-ROM mode. Set the flash access size setting to 32 bits.
   (FASZR: ASZ="10")
10. Read the value that was written, and check that the correct value can be read. Furthermore, even if the correct value was read, check the flash error bits (FSTR: ERR) to ensure that there have been no ECC corrections. If an ECC correction has occurred, erase the flash memory and start again from the beginning.

PA: Write address (word-aligned)
PD[31:0]: Write data
PD[31:16]: Upper 16 bits of the write data
PD[15:0]: Lower 16 bits of the write data

3.5. Cautions When Using Flash Memory

This section explains the cautions when using flash memory.

- If this device is reset during the write, the data that is written cannot be guaranteed.
  Moreover, it is necessary to note that reset not expected like Watch Dog Timer doesn't work during the write and delete.
- If the CPU programming mode is configured (ASZ=01) in the ASZ[1:0] bits of the flash access size register (FASZR), do not execute any programs in the flash memory. The correct values will not be retrieved and the program will run out of control.
- If the CPU programming mode is configured (ASZ=01) in the ASZ[1:0] bits of the flash access size register (FASZR) and the interrupt vector table is in the flash memory, ensure that no interrupt sources occur. The correct values will not be retrieved and the program will run out of control.
- If the CPU programming mode is configured (ASZ=01) in the ASZ[1:0] bits of the flash access size register (FASZR), do not transition to sub run mode or low power consumption mode.
- If the CPU ROM mode is configured (ASZ=10) in the ASZ[1:0] bits of the flash access size register (FASZR), do not write to the flash memory.
- If the CPU programming mode is configured (ASZ=01) in the ASZ[1:0] bits of the flash access size register (FASZR), always write to the flash memory in half-words. Do not write in bytes.
- After writing to the flash memory, always perform a dummy read before reading the data that is actually wanted. If data is read immediately after a write, the read value cannot be guaranteed.
- Because ECC bits are added in this model, writes are always required to be performed in units of 32 bits by using two 16-bit writes. See Section "3.4 Writing to Flash Memory in Products Equipped with ECC" for details on the procedure.

4. Registers

This section explains the registers.

■ List of Registers

Abbreviated Register Name	Register Name	Reference
FASZR	Flash Access Size Register	4.1
FRWTR	Flash Read Wait Register	4.2
FSTR	Flash Status Register	4.3
FSYNDN	Flash Sync Down Register	4.4

4.1. FASZR (Flash Access Size Register)

This section explains the FASZR.

This register configures the access size for flash memory. After reset is released, ASZ is set to "10" (32-bit read), and the flash memory enters CPU-ROM mode. To put the flash memory into CPU programming mode, set ASZ to "01".

bit	7	6	5	4	3	2	1	0
Field	Reserved						ASZ	
Access							RW	RW
Init							1	0

Field	Bit	Description
ASZ	1:0	Flash Access Size Specifies the access size of the flash memory. 00: Setting prohibited 01: 16-bit read/write (CPU programming mode) 10: 32-bit read (CPU ROM mode: Default value) 11: Setting prohibited

<Notes>

- When ASZ is set to "01", always perform writes to flash using half-word access (16-bit access).
- Do not change this register using an instruction that is contained in the flash memory. Overwrite this register from a program in any other region except for flash memory.
- Perform a dummy read to this register, after changing this register.

4.2. FRWTR (Flash Read Wait Register)

This section explains the FAWTR.

This register is meaningful when ASZ="10" (32-bit read mode).
It configures the minimum wait cycles for flash.

bit	7	6	5	4	3	2	1	0
Field	Reserved						RWT	
Access							RW	R
Init							1	0

Field	Bit	Description
RWT	1:0	Read Wait Cycle Specifies the minimum read cycle for flash memory. Bit "0" is locked at the hardware level. 00: 0 cyc. wait This setting can be used when the CPU clock is 60 MHz or less. 10: 2 cyc. wait (default value) This setting can be used when the CPU clock is 80 MHz or less.

<Note>

Do not set RWT to "00" if the CPU frequency exceeds 60 MHz.
Operation cannot be guaranteed if this setting is made.

4.3. FSTR (Flash Status Register)

This section explains the FSTR.

This is a status register of flash macro.

bit	7	6	5	4	3	2	1	0
Field	Reserved					EER	HNG	RDY
Access						RW	R	R
Init						0	0	X

[bit7:3] Reserved bits

The read values are undefined. Ignored on write.

[bit2] ERR: Flash ECC Error

This bit is set to "1" if ECC error correction occurs.

Field	Bit	Description
EER	2	Flash ECC Error On read: 0: Correction due to an ECC error has not occurred. 1: Correction due to an ECC error has occurred. On write: 0: Clears this bit. 1: Ignored.

[bit1] HNG: Flash Hang

Indicates whether the flash memory is in the HANG state. Flash memory enters the HANG state if the timing is exceeded (See "[bit5] TLOV: Timing Limit Exceeded Bit"). If this bit becomes "1", issue a reset command (See Section "3.2.1 Command Sequences").

Because the correct value might not be read out immediately after issuing an automatic programming algorithm command, ignore the value of this bit as read out the first time after a command is issued.

Field	Bit	Description
HNG	1	Flash Hang 0: The flash memory HANG state has not been detected. 1: The flash memory HANG state has been detected.

[bit0] RDY: Flash Rdy

Indicates whether a flash memory write or erase operation using the automatic programming algorithm is in progress or finished. While an operation is in progress, data cannot be written and the flash memory cannot be erased.

Field	Bit	Description
RDY	0	Flash Rdy 0: Operation in progress (cannot write or erase) 1: Operation finished (can write or erase)

Because the correct value might not be read immediately after an automatic programming algorithm command is issued, ignore the value of this bit as read the first time after a command is issued.

4.4. FSYNDN (Flash Sync Down Register)

This section explains the FSYNDN.

The wait cycle is inserted in the read access to the flash memory at the CPU ROM mode. Power consumption can be reduced by decreasing the access clock frequency of the flash memory.

bit	7	6	5	4	3	2	1	0
Field	Reserved					SD		
Access						RW	RW	RW
Init						0	0	0

Field	Bit	Description
SD	2:0	0b000: 0 (default value) 0b001: +1 wait 0b010: Setting prohibited 0b011: +3 wait 0b100: Setting prohibited 0b101: +5 wait 0b110: Setting prohibited 0b111: +7 wait

The number of wait set by this bit is added to the RWT bit of the flash read wait register (FRWTR).

Example)
in case of RWT=0b01(0cycle wait) and SD=0b011, 0+3=3 wait
in case of RWT=0b10(2cycle wait) and SD=0b011, 2+3=5 wait

CHAPTER: Flash Security

The flash security feature provides possibilities to protect the content of the flash memory.

This section describes the overview and operations of the flash security.

Administration code: 9BF500SECURITY-E01.1

1. Overview

This section explains the overview of the flash security.

If the protection code of 0x0001 is written in the security code area of flash memory, access to the flash memory is restricted. Once the flash memory is protected, performing the chip erase operation only can unlock the function otherwise read/write access to the flash memory from any external pins is not possible.

This function is suitable for applications requiring security of self-containing program and data stored in the flash memory.

Table 1-1 shows the address and the protection code of the security code.

Table 1-1 Address and protection code of security code

Address	Protection Code
0x0010_0000	0x0001

2. Operation Explanation

This section explains the operation of the flash security.

■ Setting Security

Write the protection code 0x0001 in address of the security code. The security is enabled and set after all the reset factors are generated or after turning on the power again.

■ Releasing Security

The security is released by all the reset factors or turning on the power again after performing the chip erase.

■ Operation with Security Enabled

The operations with security enabled vary depending on each mode.

Table 2-1 shows the security operations in each mode.

Table 2-1 Flash Operation with Security Enabled

Mode	Mode pin MD[1:0]	Access to flash			Access from JTAG pins
		Chip erase	Other commands	Read	
User mode	"00"	Enabled	Enabled	Valid data	Disabled
Serial writer mode	"01"	Enabled	Disabled	Invalid data	Disabled

<Notes>

- Writing the protection code is generally recommended to take place at the end of the flash programming. This is to avoid unnecessary protection during the programming.
- In user mode, there is no limit to flash memory even during security is enabled. However, JTAG pins are fixed not to access internally from these pins during security is enabled. To release security, perform the chip erase operation using a serial writer because the security cannot be released through JTAG pins.
- When security enabled, the obstruction analysis of the flash memory cannot be performed.

CHAPTER: Serial Programming Connection

MB9BF500 supports serial onboard write (Fujitsu Semiconductor standard) to flash memory. This chapter explains the basic configuration for serial write to flash memory by using the Fujitsu Semiconductor Serial Programmer.

1. Serial Programmer

1. Serial Programmer

Fujitsu Semiconductor Serial Programmer (software) is an onboard programming tool for all microcontrollers with built-in flash memory.

Two types of Serial Programmer are available according to the PC interface (RS-232C or USB) used. Choose the type according to your environment.

Onboard write is possible with the product which USB function is installed by connecting the PC and microcontroller directly without performing USB-serial conversion.

1.1. Basic Configuration

This section explains the basic configuration.

■ Basic Configuration of FUJITSU SEMICONDUCTOR MCU Programmer (Clock Asynchronous Serial Write)

FUJITSU SEMICONDUCTOR MCU Programmer writes data, through clock asynchronous serial communication, to built-in flash memory of a microcontroller installed in the user system when the PC and the user system are connected through RS-232C cable.

Figure 1-1 shows the basic configuration of FUJITSU SEMICONDUCTOR MCU Programmer, and Table 1-1 lists the system configuration.

Figure 1-1 Basic Configuration of FUJITSU SEMICONDUCTOR MCU Programmer

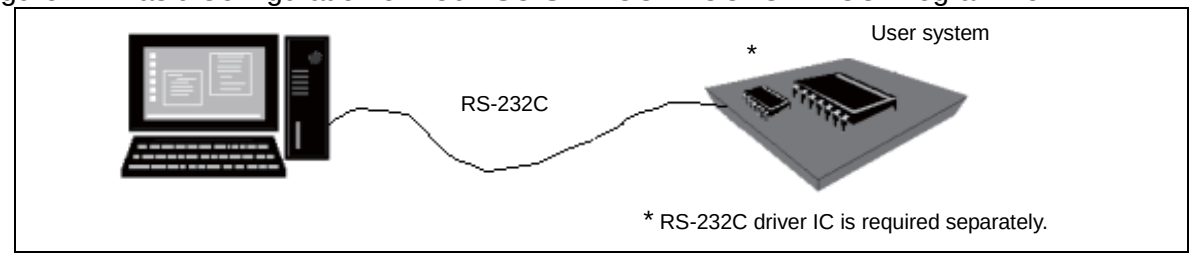
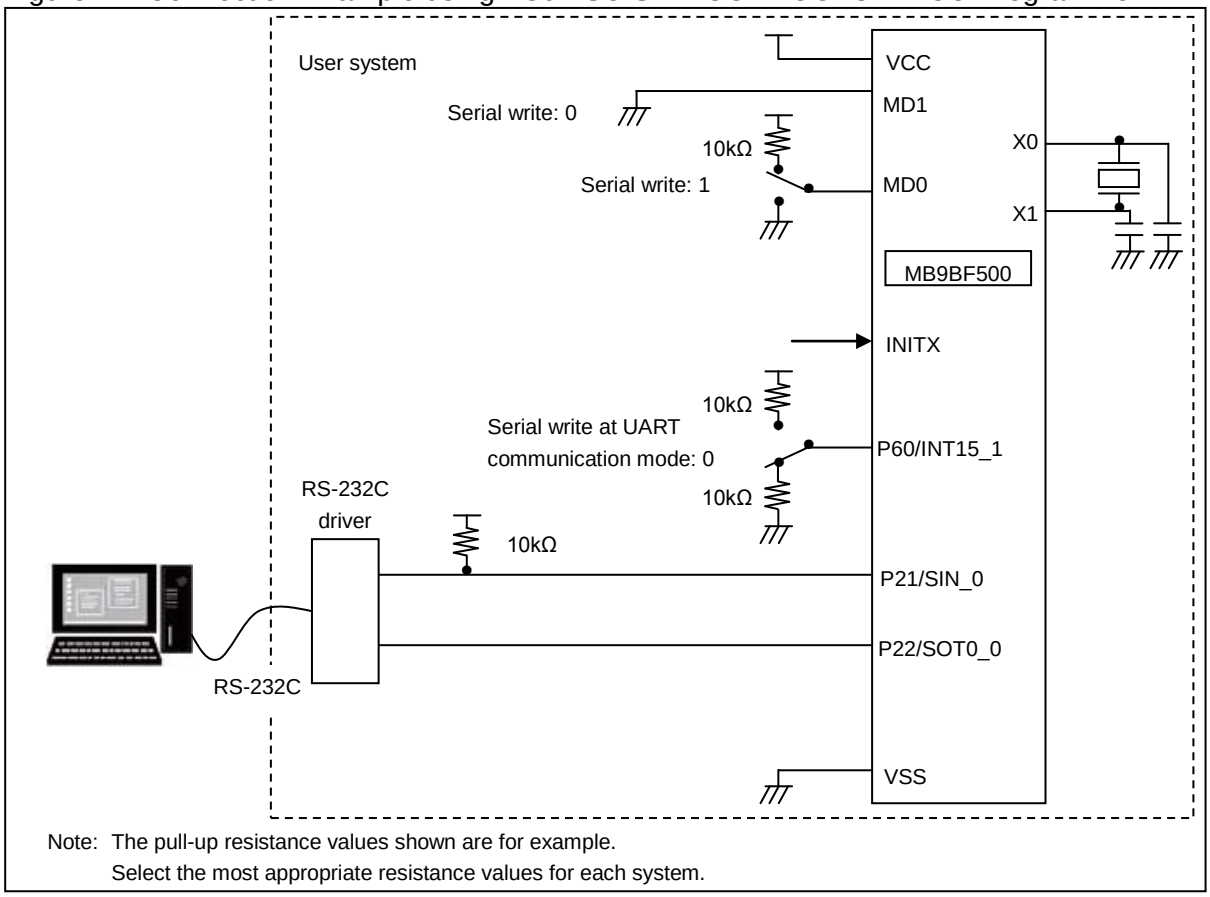


Table 1-1 System Configuration of FUJITSU SEMICONDUCTOR MCU Programmer

Name	Specifications
FUJITSU SEMICONDUCTOR MCU Programmer	Software (In case you request the data, contact to FUJITSU sales representatives.)
RS-232C cable	Sold on the market.

Figure 1-2 Connection Example using FUJITSU SEMICONDUCTOR MCU Programmer



Note: The pull-up resistance values shown are for example.
Select the most appropriate resistance values for each system.

Table 1-2 Oscillating frequency and communication baud rate available for clock asynchronous serial communication

Source Oscillating Frequency	Communication Baud Rate
4MHz	9600bps
8MHz	19200bps
16MHz	38400bps
24MHz	57600bps
48MHz	115200bps

■ Basic Configuration of FUJITSU USB DIRECT Programmer (USB Serial Write)

FUJITSU USB DIRECT Programmer writes data, through USB communication mode, to built-in flash memory of a microcontroller when the PC and the user system are connected through a USB cable.

Figure 1-3 shows the basic configuration of FUJITSU USB DIRECT Programmer, and Table 1-3 lists the system configuration.

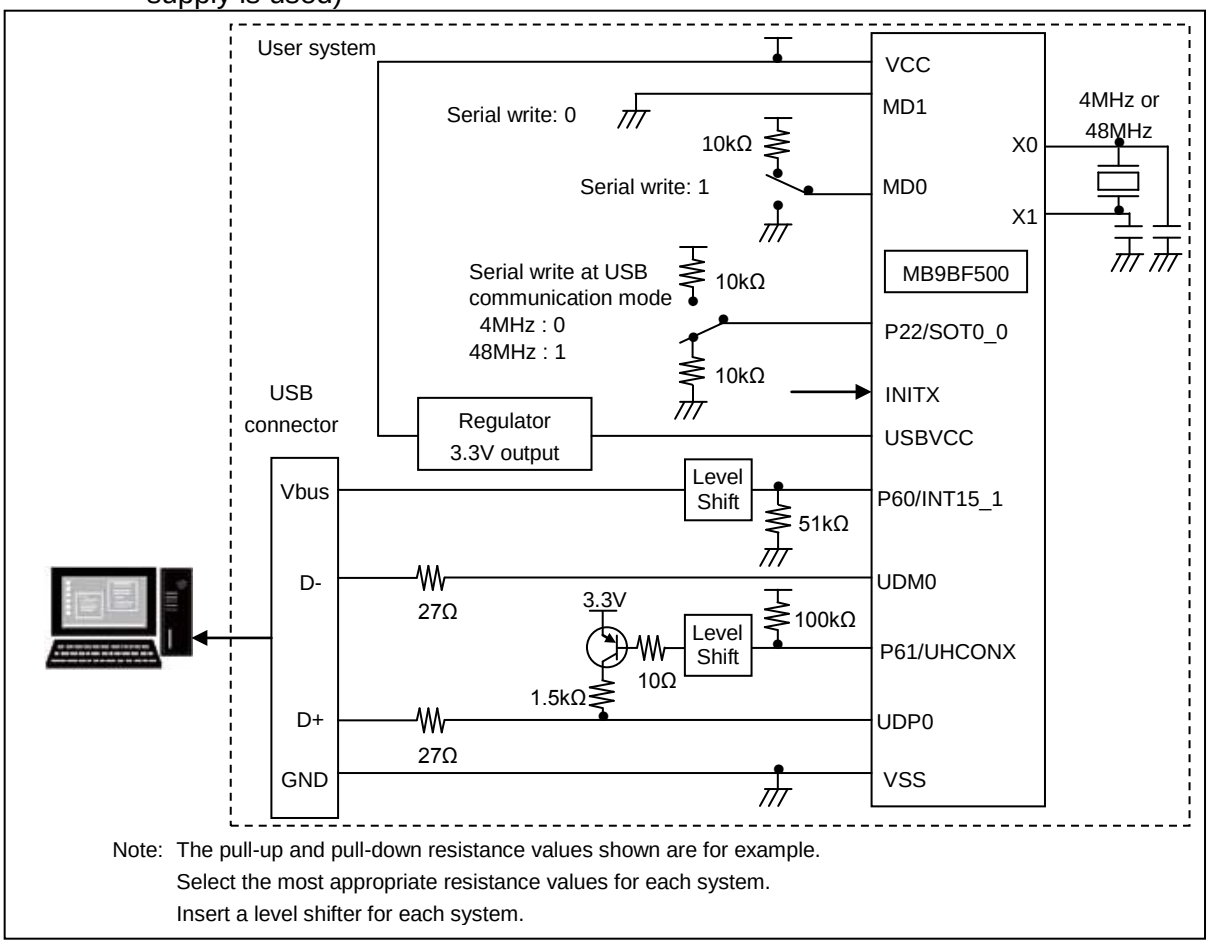
Figure 1-3 Basic Configuration of FUJITSU USB DIRECT Programmer

Table 1-3 System Configuration of FUJITSU USB DIRECT Programmer

Name	Specifications
FUJITSU USB DIRECT Programmer	Software (In case you request the data, contact to FUJITSU sales representatives.)
USB cable	Sold on the market.

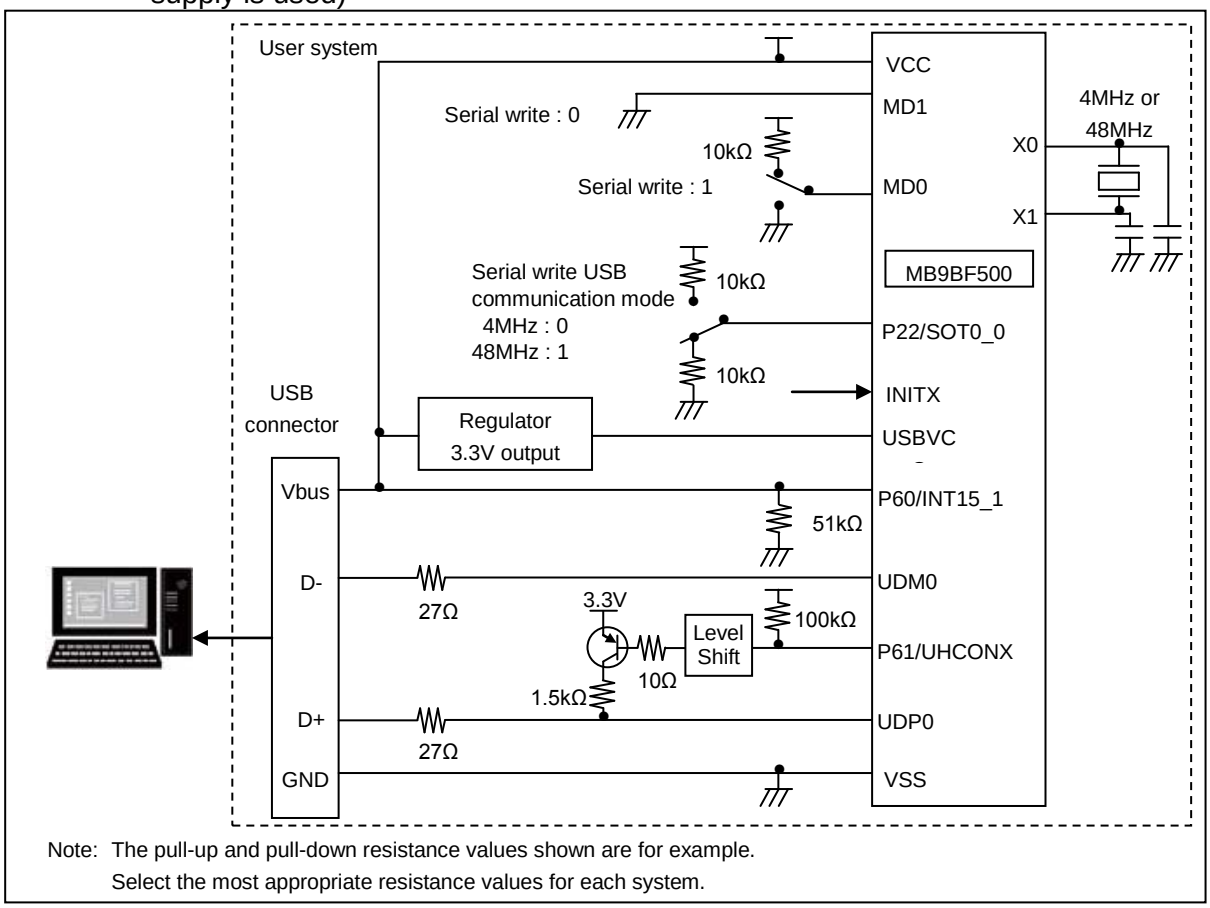
For connection examples, see the manual (help section) of FUJITSU USB DIRECT Programmer.

Figure 1-4 Connection example using FUJITSU USB DIRECT Programmer (own power supply is used)



Note: The pull-up and pull-down resistance values shown are for example.
Select the most appropriate resistance values for each system.
Insert a level shifter for each system.

Figure 1-5 Connection example using FUJITSU USB DIRECT Programmer (bus power supply is used)



Note: The pull-up and pull-down resistance values shown are for example.
Select the most appropriate resistance values for each system.

1.2. Pins Used

This section explains the used pins.

Table 1-4 Pins used for serial write

Pins	Function	Supplement
MD0, MD1	Mode pin	Performing an external reset or turning on the power after setting MD0=H and MD1=L enters the serial write mode. When attaching a pull-up or pull-down resistor, avoid long wiring.
X0, X1	Oscillation pin	See the "Data Sheet" for the source oscillation clock (main clock) frequencies that can be used in serial write mode. (Restrictions apply to clock asynchronous communication. For details, see Table 1-2.)
P22/SOT0_0	UART serial data output pin/ USB source oscillating frequency select pin	When the communication mode is set to UART, this pin becomes a serial data output pin when communication begins after the serial write mode is activated. When the communication mode is set to USB, this pin controls the frequency for source oscillation clock. P22=L : source oscillation frequency : 4MHz P22=H : source oscillation frequency: 48MHz
P21/SIN0_0	Clock synchronous/ asynchronous select pin/UART serial data input pin	Setting the input level of this pin to "H" until the start of communication enables the clock asynchronous communication mode, and setting it to "L" enables the clock synchronous communication mode. When the communication mode is set to UART, this pin can be used as a serial data input pin when communication begins after the serial write mode is activated.
P60/INT15_1	Communication mode select pin	The communication mode is determined by the input level of this pin at reset to shift to the serial write mode. Setting this pin to "H" enables the USB communication mode, and setting it to "L" enables the UART communication mode.
P61/UHCONX	Pull-up control pin for UDP0	This pin controls the pull-up of USB side (+) when the communication mode is USB. UHCONX=L : Connect the pull-up resistor UHCONX=H : Disconnect the pull-up resistor
UDP0	USB I/O pin	This pin becomes an input/output pin of USB side (+) when the communication mode is set to USB.
UDM0	USB I/O pin	This pin becomes an input/output pin of USB side (-) when the communication mode is set to USB.
INITX	Reset pin	-
VCC	Power supply pin	For writing, supply power to the microcontroller from the user system.
USBVCC	Power supply pin for USB I/O	Do not apply supply voltage exceeding the maximum rating of 4.0V.
VSS	GND pin	-

CM91-10102-2E

FUJITSU SEMICONDUCTOR • CONTROLLER MANUAL

FM3 Family

32-BIT MICROCONTROLLER

MB9BF500 Series

FLASH PROGRAMMING MANUAL

April 2010 the second edition

Published **FUJITSU SEMICONDUCTOR LIMITED**

Edited Sales Promotion Dept.