The following document contains information on Cypress products. The document has the series name, product name, and ordering part numbering with the prefix “MB”. However, Cypress will offer these products to new and existing customers with the series name, product name, and ordering part number with the prefix “CY”.

How to Check the Ordering Part Number

1. Go to www.cypress.com/pcn.
2. Enter the keyword (for example, ordering part number) in the **SEARCH PCNS** field and click **Apply**.
3. Click the corresponding title from the search results.
4. Download the Affected Parts List file, which has details of all changes

For More Information

Please contact your local sales office for additional information about Cypress products and solutions.

About Cypress

Cypress is the leader in advanced embedded system solutions for the world's most innovative automotive, industrial, smart home appliances, consumer electronics and medical products. Cypress' microcontrollers, analog ICs, wireless and USB-based connectivity solutions and reliable, high-performance memories help engineers design differentiated products and get them to market first. Cypress is committed to providing customers with the best support and development resources on the planet enabling them to disrupt markets by creating new product categories in record time. To learn more, go to www.cypress.com.

FM3, MB9A310/110 Series

Flash Programming Guide I

Doc. No. 002-07775 Rev. *A

Cypress Semiconductor
198 Champion Court
San Jose, CA 95134-1709
http://www.cypress.com

Preface

Purpose of this manual and intended readers

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

Sample programs and development environment

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

* : 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.
Cypress Semiconductor assumes no responsibility for any damage that may occur as a result of using a sample program.

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

1. Overview

This series is equipped with 64 Kbytes to 512 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.

1.1 Flash Memory Features

- Usable capacity:
  Minimum configuration: 64 Kbytes
  Maximum configuration: 512 Kbytes

Note:

Because this series stores ECC codes, it is equipped with additional flash memory of 7 bits for every 4 bytes of memory described above.

- High-speed flash:
  At 40 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 series consists of 64 Kbytes to 512 Kbytes flash memory region, a security code region, and a built-in CR trimming data region.

The figure below shows the address and sector structure of the flash memory built into this series as well as the address of security/CR trimming data.

Note: See "Chapter Flash Security" for details on the security.

Note: See Section "4.6 CRTRMM (CR Trimming Data Mirror Register)" and "Chapter High-Speed CR Trimming" of the Peripheral Manual for details on the Built-in High-Speed CR trimming data.

Figure 1. Memory map of MB9AF111 flash memory

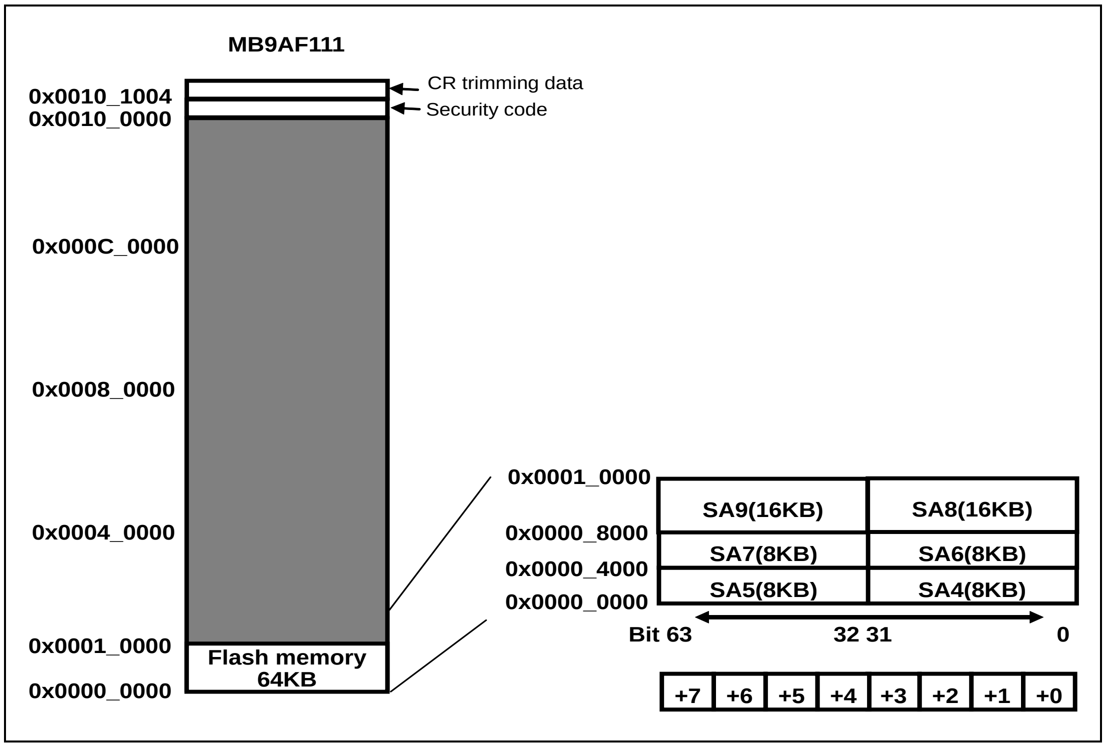


Figure 2. Memory map of MB9AF112/312 flash memory

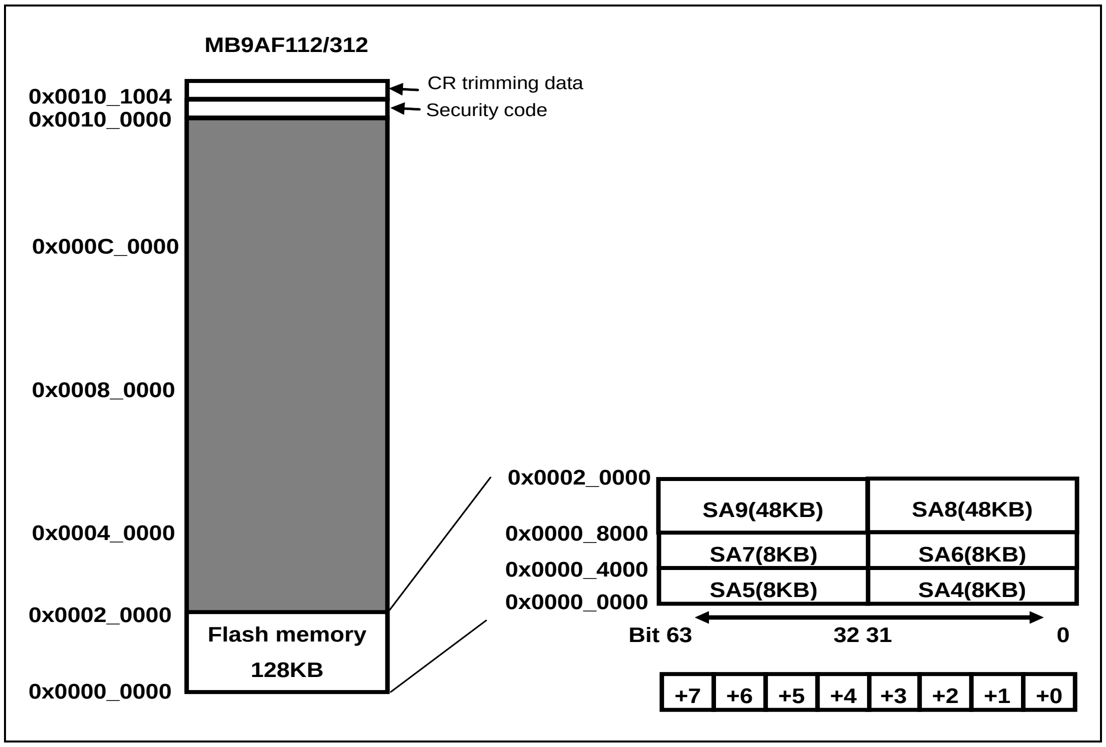


Figure 3. Memory map of MB9AF114/314 flash memory

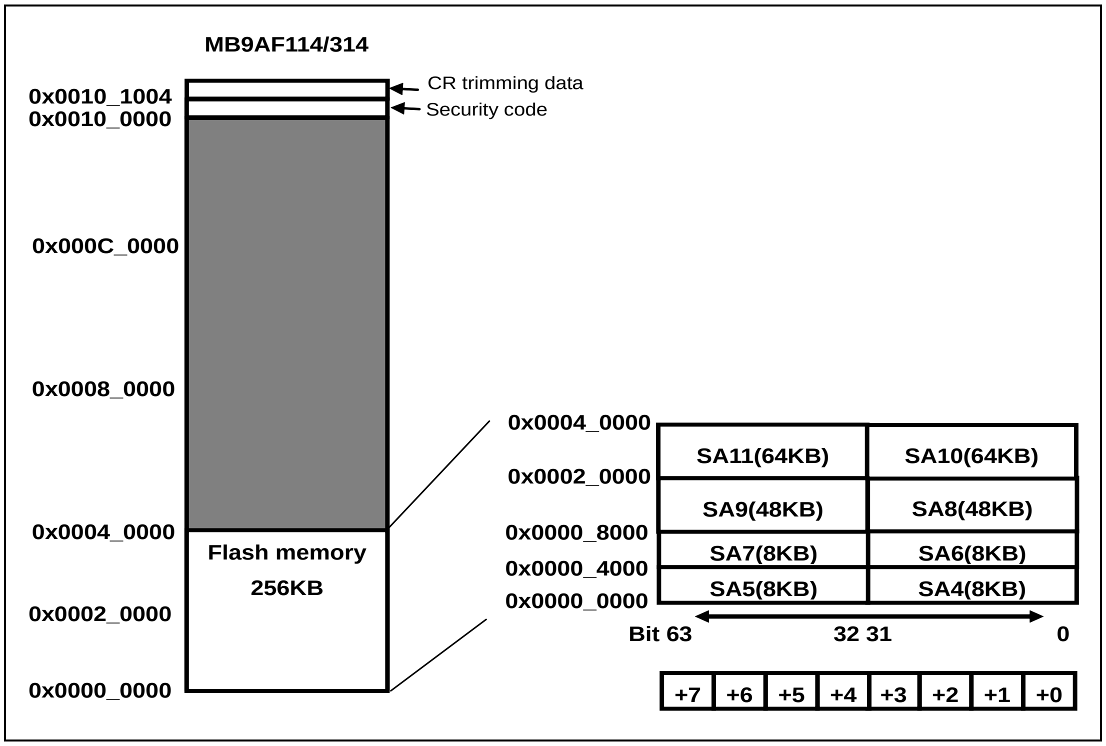


Figure 4. Memory map of MB9AF115/315 flash memory

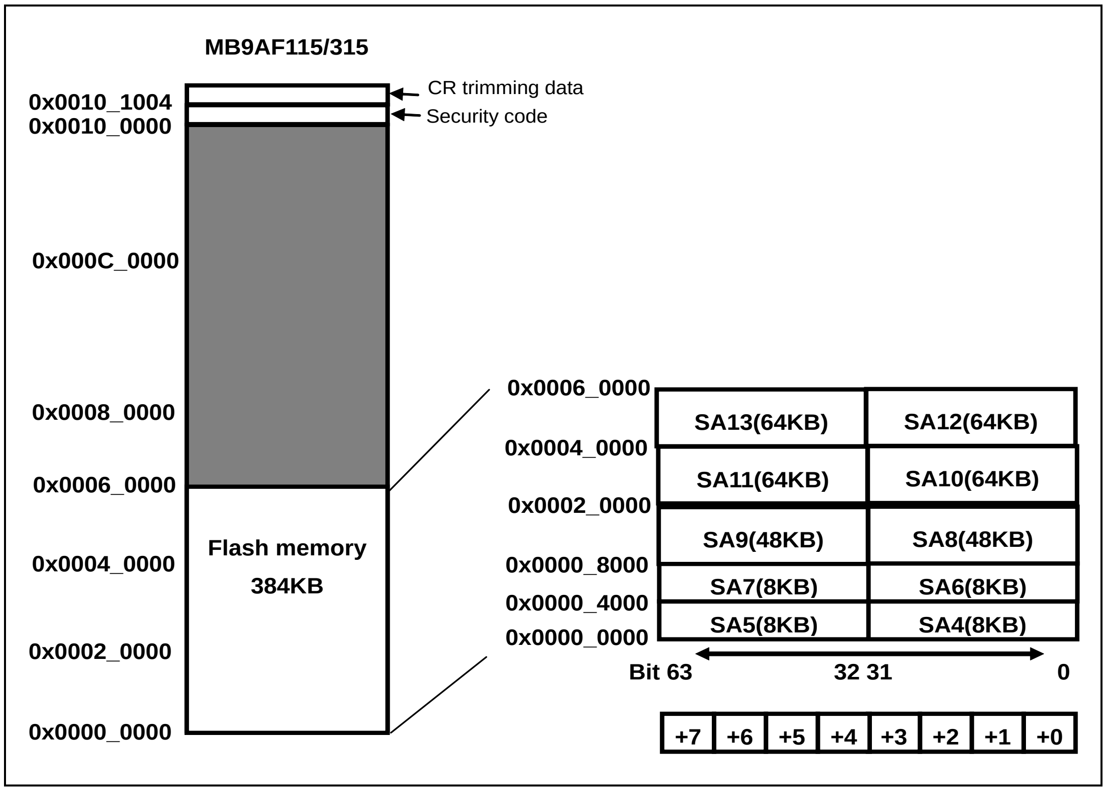


Figure 5. Memory map of MB9AF116/316 flash memory

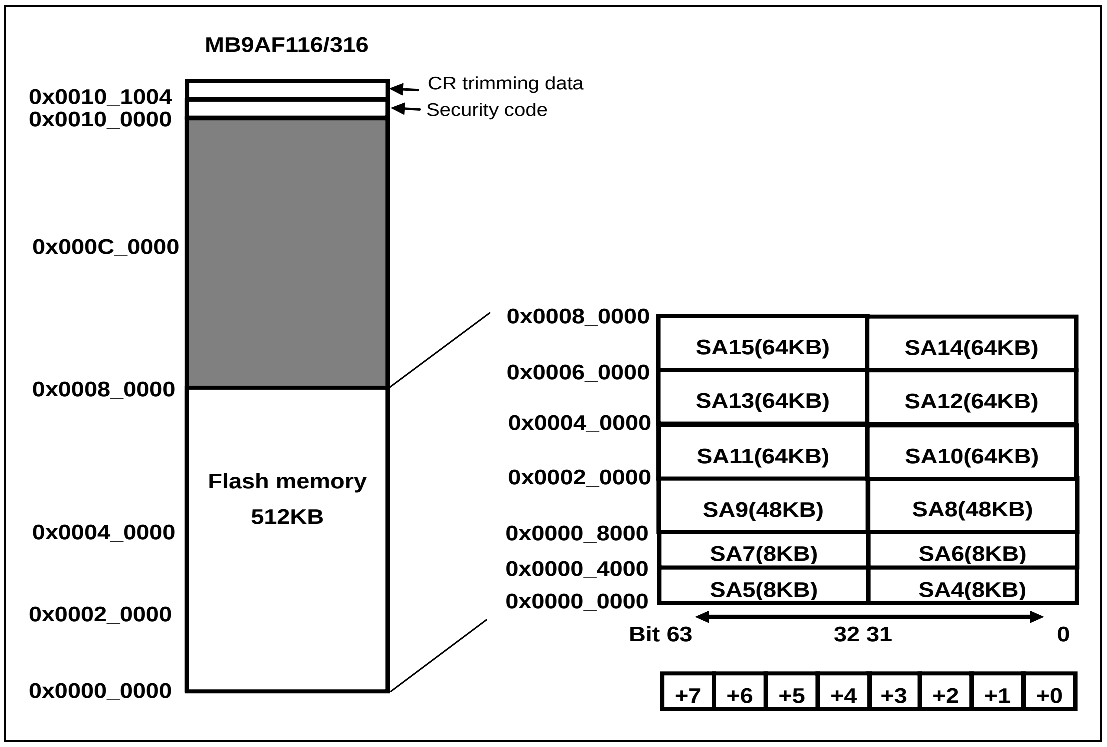
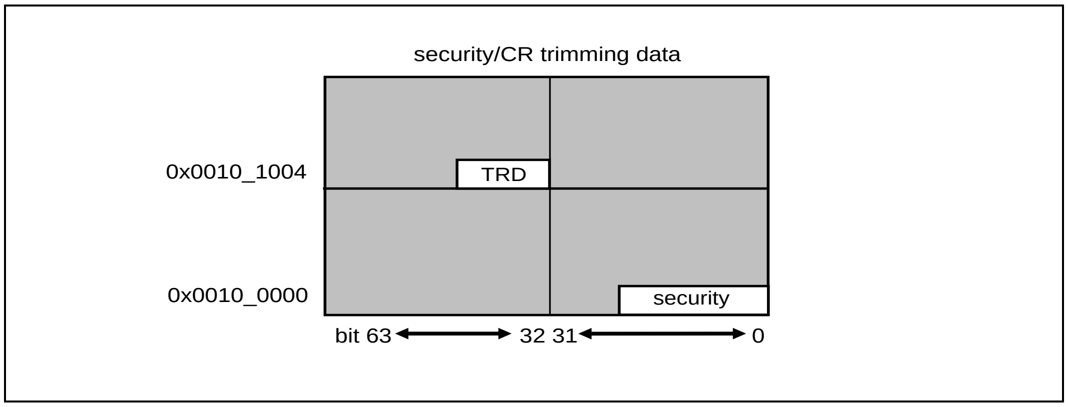


Figure 6. Address of security/CR trimming data



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).

3.1.1 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.

3.1.2 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 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 2 shows the command sequences.

Table 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	--	--	--	--	--	--	--	--	--	--
Write	4	0x1550	0xAA	0xAA8	0x55	0x1550	0xA0	PA	PD	--	--	--	--
Chip erase	6	0x1550	0xAA	0xAA8	0x55	0x1550	0x80	0x1550	0xAA	0xAA8	0x55	0x1550	0x10
Sector erase	6	0x1550	0xAA	0x AA8	0x55	0x1550	0x80	0x1550	0xAA	0xAA8	0x55	SA	0x30
Sector erase suspended	1	0xXXX	0xB0	--	--	--	--	--	--	--	--	--	--
Sector erase restarting	1	0xXXX	0x30	--	--	--	--	--	--	--	--	--	--

X: Any value

PA: Write address

SA: Sector address (Specify any address within the address range of the sector to erase)

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 16 bits. The upper 16 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.
- "Address within the address range" specified when setting or erasing the security code should be 0x0010_0000.
- "Address within the address range" specified when setting or erasing the CR trimming data should be 0x0010_1004.

3.2.2 Command Operating Explanations

This section explains the command operating.

3.2.2.1 Read/Reset Command

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

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.

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

3.2.2.2 Program (Write) Command

The automatic programming algorithm can be activated and the data is written to the flash memory by issuing 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 issuing the fourth command 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 command is not recognized properly if the fourth write command (write data cycle) is issued to an odd address. Always issue it 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.

3.2.2.3 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.

3.2.2.4 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 35 μs has elapsed (timeout interval), the automatic programming algorithm is activated and the sector erase operation begins.

To erase multiple sectors, issue the sector erase code (0x30) which is the sixth write code of the sector erase command to the address of the sector to erase within 35μs (timeout interval). If the sector erase code is not issued within the timeout interval, the sector erase code added after the timeout interval has elapsed may become inactive.

3.2.2.5 Sector Erase Suspended Command

By issuing the sector erase suspended command during sector erase or during command timeout, sector erase can be suspended. In the sector erase suspended state, the read operation of memory cells of the sector not to erase is made possible. See Section "3.3.5 Sector Erase Suspended Operation" for details on the actual operation.

Note:

This command is only valid during sector erase. It is ignored even if it is issued during chip erase or during write.

3.2.2.6 Sector Erase Restart Command

In order to restart the erase operation in the sector erase suspended state, issue the sector erase restart command. Issuing the sector erase restart command returns the flash memory to the sector erase state and restarts the erase operation. See Section "3.3.6 Sector Erase Restart Operation" for details on the actual operation.

Note:

This command is only valid during sector erase suspended. It is ignored even if it is issued during sector erase.

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.

3.2.3.1 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 7 shows the bit structure of the hardware sequence flags.

Figure 7. 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	TOGG2	Undefined	Undefined

In the event of byte access

bit	7	6	5	4	3	2	1	0
	DPOL	TOGG	TLOV	Undefined	SETI	TOGG2	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 immediately 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 shows the correspondence between each bit of the hardware sequence flags and the status of the flash memory.

Table 3. List of hardware sequence flag states

State				DPOL	TOGG	TLOV	SETI	TOGG2
Running	Write operation			Inverted data [1]	Toggle	0	0	0
	Erase operation	Chip erase		0	Toggle	0	1	Toggle
		Sector erase	timeout interval	0	Toggle	0	0	Toggle
			erase	0	Toggle	0	1	Toggle
		Sector erase suspended	Read (Sector to erase)	0	0	0	1	Toggle
			Read(Sector not to erase)	Data [1]	Data [1]	Data [1]	Data [1]	Data [1]
Time limit exceeded	Write operation			Inverted data [1]	Toggle	1	0	0
	Sector/chip erase			0	Toggle	1	1	Toggle

[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 bit 7 written at the last command sequence (PD). 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 all regions of flash memory.
  - 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".

- During sector erase suspended
  - When this bit is read out by specifying an address in the sector specified as sector erase:

    Reads out "0".
  - When this bit is read out by specifying an address in the sector other than specified as sector erase:

    Reads out the value of bit 7 of a specified address.

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 alternatively 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.
- During sector erase suspended
  - When this bit is read out by specifying an address in the sector specified as sector erase:

    Reads out "0".
  - When this bit is read out by specifying an address in the sector other than specified as sector erase:

    Reads out the value of bit 6 of a specified address.

[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 DPOL 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 DPOL bit remains invalid, and "1" and "0" are continuously read out alternatively 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.

- During sector erase suspended
  - When this bit is read out by specifying an address in the sector specified as sector erase:

    Reads out "0".
  - When this bit is read out by specifying an address in the sector other than specified as sector erase:

    Reads out the value of bit 5 of a specified address.

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 35 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 timeout interval

The following sector erase code (0x30) can be accepted.

1: Sector erase timeout interval exceeded

In this case, if the DPOL 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 suspended (0xB0) are ignored until the internal flash memory erase operation has finished.

- During sector erase suspended
  - When this bit is read out by specifying an address in the sector specified as sector erase:

    Reads out "1".
  - When this bit is read out by specifying an address in the sector other than specified as sector erase:

    Reads out the value of bit 3 of a specified address.

[bit2] TOGG2 : Toggle flag bit

In the sector erase suspended state, a read access can be made to a sector not to erase, but not to a sector to erase. Therefore, this toggle bit can be detected depending on whether the read address is a sector to erase or not and whether the read data toggles or not during sector erase suspended.

- During writing

  Reads out "0".
- During sector erase or chip erase

  When this bit is read out continuously, it alternatively returns "1" and "0" (toggles).

- During sector erase suspended
  - ☐ When this bit is read out by specifying an address in the sector specified as sector erase:

    When this bit is read out continuously, it alternatively returns "1" and "0" (toggles).
  - ☐ When this bit is read out by specifying an address in the sector other than specified as sector erase:

    Reads out the value of bit 2 of a specified address.

[bit1:0] 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.

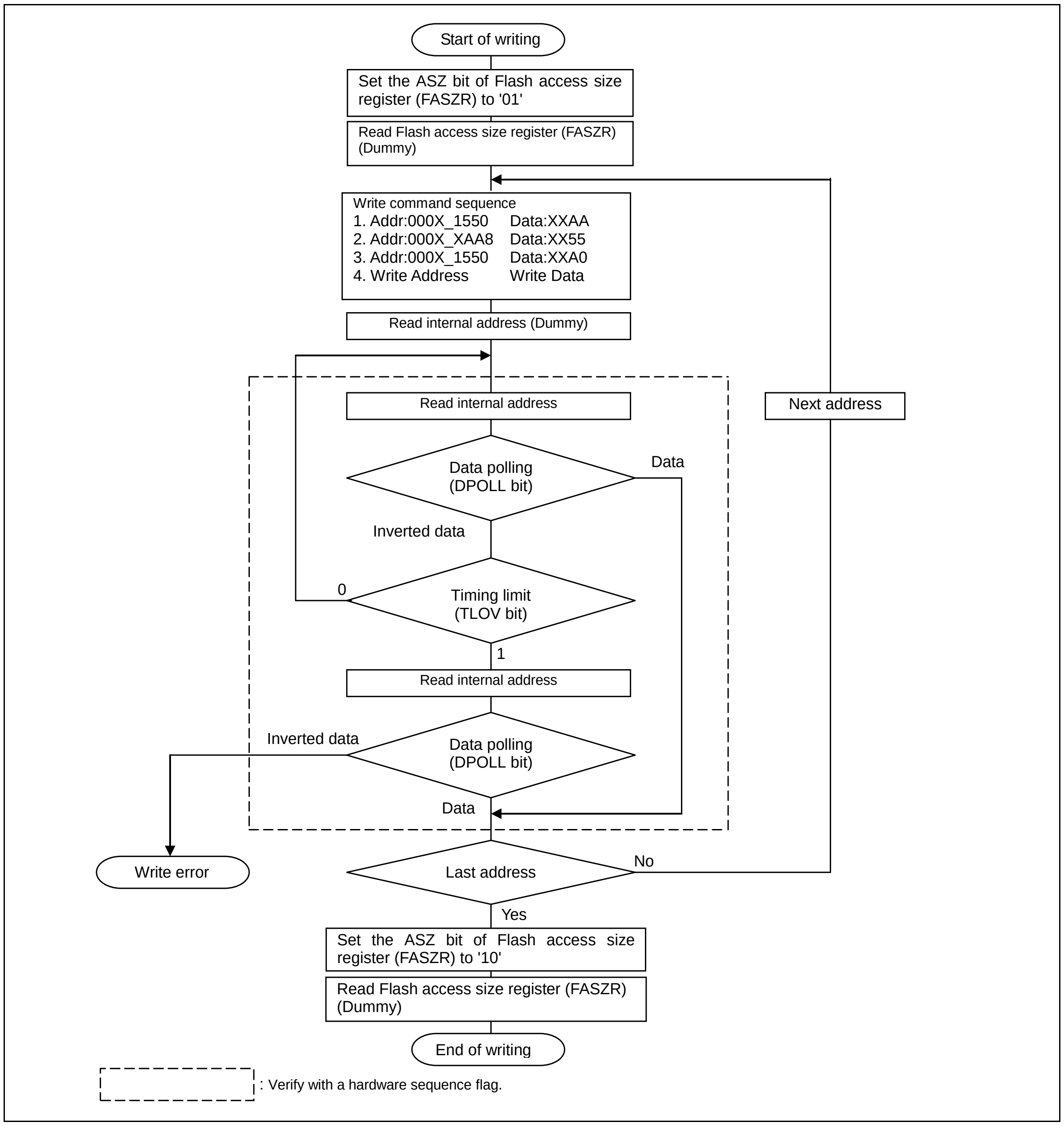
1. The write command is issued to the target sector sequentially

   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 DPOL 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 bit 7 written at the last command sequence (PD) is read out.

Figure 8 shows an example of a write operation to the flash memory.

Figure 8. Example write operation


Start of writing
Set the ASZ bit of Flash access size register (FASZR) to '01'
Read Flash access size register (FASZR) (Dummy)
Write command sequence
1. Addr:000X_1550 Data:XXAA
2. Addr:000X_XAA8 Data:XX55
3. Addr:000X_1550 Data:XXA0
4. Write Address Write Data
Read internal address (Dummy)
Read internal address
Next address
Data polling (DPOLL bit)
Data
Inverted data
0
Timing limit (TLOV bit)
1
Read internal address
Inverted data
Data polling (DPOLL bit)
Data
Write error
Last address
No
Yes
Set the ASZ bit of Flash access size register (FASZR) to '10'
Read Flash access size register (FASZR) (Dummy)
End of writing
: Verify with a hardware sequence flag.

Notes:

- See Section "3.2 Automatic Programming Algorithm" for details on the write command.
- Because the value of the DPOL 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 issued 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 series, 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. Issue the chip erase command sequentially to the target sector

   The automatic programming algorithm is activated and the chip erase operation of the flash memory begins.

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 DPOL 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.

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. Issue the sector erase command sequentially to the target sector

   Once 35 μs has elapsed (the timeout interval), the automatic programming algorithm activates and the sector erase operation begins.

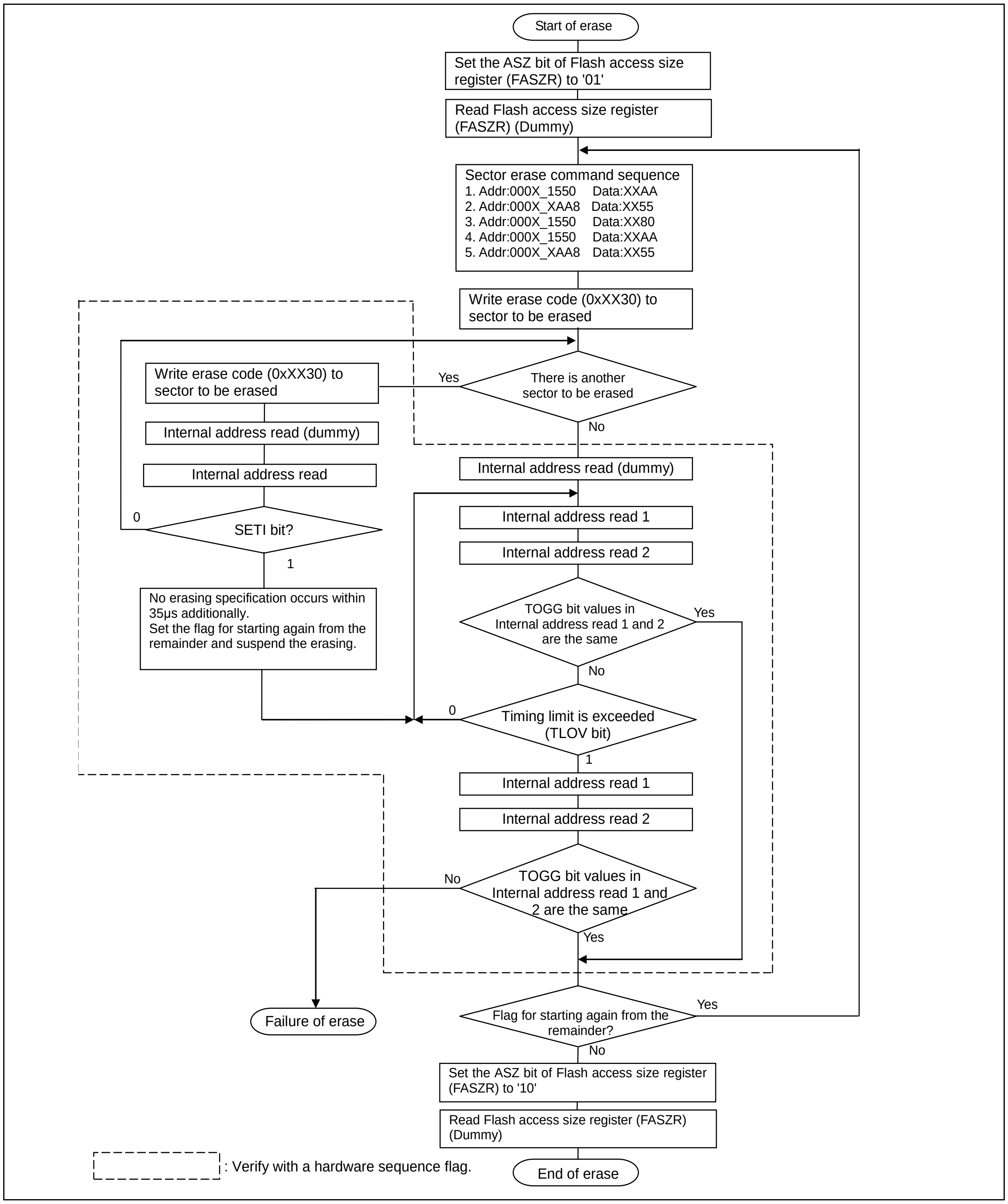
   To erase multiple sectors, issue the erase code (0x30) to an address in the sector to erase within 35μs (the timeout interval). If the code is issued after the timeout interval has elapsed, the added sector erase code may be invalid.

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 DPOL 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 9 shows an example of the sector erase procedure for the case of using the TOGG bit for confirmation.

Figure 9. Example sector erase procedure


Start of erase
Set the ASZ bit of Flash access size register (FASZR) to '01'
Read Flash access size register (FASZR) (Dummy)
Sector erase command sequence
1. Addr:000X_1550 Data:XXAA
2. Addr:000X_XAA8 Data:XX55
3. Addr:000X_1550 Data:XX80
4. Addr:000X_1550 Data:XXAA
5. Addr:000X_XAA8 Data:XX55
Write erase code (0xXX30) to sector to be erased
There is another sector to be erased
Yes
No
Write erase code (0xXX30) to sector to be erased
Internal address read (dummy)
Internal address read
SETI bit?
0
1
No erasing specification occurs within 35μs additionally.
Set the flag for starting again from the remainder and suspend the erasing.
Internal address read (dummy)
Internal address read 1
Internal address read 2
TOGG bit values in Internal address read 1 and 2 are the same
Yes
No
Timing limit is exceeded (TLOV bit)
0
1
Internal address read 1
Internal address read 2
TOGG bit values in Internal address read 1 and 2 are the same
No
Yes
Failure of erase
Flag for starting again from the remainder?
Yes
No
Set the ASZ bit of Flash access size register (FASZR) to '10'
Read Flash access size register (FASZR) (Dummy)
End of erase
: Verify with a hardware sequence flag.

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 DPOL 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 the sector erase command or the erase suspended command is issued during sector erase, including the timeout interval, it is ignored.

3.3.5 Sector Erase Suspended Operation

This section explains the sector erase suspended operation.

When the sector erase suspended command is sent during sector erase or in the command timeout state, the flash memory makes a transition to the sector erase suspended state and temporarily suspends the erase operation .By sending the erase restart command, the flash memory is returned to the sector erase state and can restart the suspended erase operation. However, even if the flash memory has changed from the command timeout state to the sector erase suspended state, when the erase restart command is written properly, the flash memory does not make a transition to the command timeout state but make a transition to the sector erase state and restarts the sector erase operation immediately.

3.3.5.1 Sector Erase Suspended Operation

Sector erase is suspended in the following steps:

1. Write the sector erase suspended command to an arbitrary address within the address range of the flash memory during the time between the command timeout interval and the sector erase interval.
2. If the sector erase suspended command is issued during the command timeout interval, stop timeout immediately and suspend the erase operation. If the sector erase suspended command is issued during sector erase, it takes up to 35μs until erasing is actually stopped.

Notes:

- See Section "3.2 Automatic Programming Algorithm" for details on the sector erase suspended command.
- Sector erase can only be suspended during the time between the command timeout interval and the sector erase interval. Chip erase cannot be suspended. In addition, even if the sector erase suspended command is issued again during sector erase suspended, it is ignored.

3.3.5.2 State after Sector Erase Suspended

If a sector to erase is read out after sector erase suspended, the hardware sequence flag is read out. On the other hand, if a sector not to erase is read out, data of a memory cell is read out.

Note:

New write and erase commands are ignored in the sector erase suspended state.

3.3.6 Sector Erase Restart Operation

This section explains the operation for restarting sector erase during sector erase suspended.

When the sector erase restart command is issued to an arbitrary address while sector erase is suspended, sector erase can be restarted.
When the sector erase restart command is issued, the sector erase operation during sector erase suspended is restarted .See Section "3.2 Automatic Programming Algorithm " for details on the sector erase restart command.

Notes:

- The sector erase restart command is only valid during sector erase suspended. Even if the sector erase restart command is issued during sector erase, it is ignored.
- After the sector erase restart command is issued, it takes more than 2ms until the sector erase operation is restarted. Therefore, when erase restart and erase stop are repeated at intervals less than this time, timing

limit is exceeded while no erase operation is in progress. If the sector erase suspended command is to be issued again after the sector erase restart command is issued, leave an interval more than 2ms after the sector erase restart command is issued.

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 series, 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")Perform a dummy read, after setting the FASZR register.
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: EER) 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 prevent an unexpected reset like Watch Dog Timer from occurring during the writing and deleting.
- 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.
- Immediately after issuing the automatic programming algorithm command to the flash memory, always perform a dummy read before reading the data that is actually wanted. If data is read immediately after issuing the automatic programming algorithm command, the read value cannot be guaranteed.
- If you are to cause this device to make a transition to the low power consumption mode, always ensure that the operation of the flash memory automatic programming algorithm has finished.
  See "CHAPTER Low Power Consumption Mode" of the Peripheral Manual for details on the low power consumption mode.
- Because ECC bits are added in this series, 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.

4.1 List of Registers

Abbreviated Register Name	Register Name	Reference
FASZR	Flash Access Size Register	4.2
FRWTR	Flash Read Wait Register	4.3
FSTR	Flash Status Register	4.4
FSYNDN	Flash Sync Down Register	4.5
CRTRMM	CR Trimming Data Mirror Register	4.6

4.2 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 register, after changing this register.

4.3 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							0	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 (default value) 01: Setting prohibited 10: Setting prohibited 11: Setting prohibited

Note:

- Setting a value other than "00" to this register is prohibited.

4.4 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.

4.5 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=0b00(0cycle wait) and SD=0b011, 0+3=3 wait

Note:

Perform a dummy read to register, after changing this register.

4.6 CRTRMM (CR Trimming Data Mirror Register)

This section explains the CRTRMM.

This is the mirror register of the CR trimming data.
A value of this register can be used in the user mode and the serial writer mode.

bit	31	9 0
Field	Reserved	TRMM
Access		R
Init		*

Field	Bit	Description
TRMM	9:0	*: Reads out lower 10 bits of an address of 0x101004.

[bit9:0] TRMM : CR Trimming Data Mirror Register

After reset is released, store the data in an address of 0x101004 of the flash memory region into this register. See "CHAPTER High-Speed CR Trimming" of the Peripheral Manual for details on the High-Speed CR trimming data.

Note:

After the flash memory is erased, as this register is cleared when reset is issued in a chip, the stored CR trimming data is lost. Therefore, before this register is cleared, save the CR trimming data stored in the register on the RAM, etc.

5. Revision History

Document Revision History

Document Title: FM3, MB9A310/110 Series, Flash Programming Guide I **Document Number:002-07775**			
Revision	Issue Date	Origin of Change	Description of Change
**	04/25/2011	AKIH	Initial Release
*A	03/02/2016	AKIH	Migrated Spansion guide “MN706-00007-1v0-E“ to Cypress format