



# S6E2CC/C5/C4/C3/C2/C1 Series

## 32-bit Microcontroller

### FM4 Family Flash Programming Specifications

Document Number: 002-04913 Rev. \*D

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

## Copyrights

© Cypress Semiconductor Corporation, 2014-2019. This document is the property of Cypress Semiconductor Corporation and its subsidiaries ("Cypress"). This document, including any software or firmware included or referenced in this document ("Software"), is owned by Cypress under the intellectual property laws and treaties of the United States and other countries worldwide. Cypress reserves all rights under such laws and treaties and does not, except as specifically stated in this paragraph, grant any license under its patents, copyrights, trademarks, or other intellectual property rights. If the Software is not accompanied by a license agreement and you do not otherwise have a written agreement with Cypress governing the use of the Software, then Cypress hereby grants you a personal, non-exclusive, nontransferable license (without the right to sublicense) (1) under its copyright rights in the Software (a) for Software provided in source code form, to modify and reproduce the Software solely for use with Cypress hardware products, only internally within your organization, and (b) to distribute the Software in binary code form externally to end users (either directly or indirectly through resellers and distributors), solely for use on Cypress hardware product units, and (2) under those claims of Cypress's patents that are infringed by the Software (as provided by Cypress, unmodified) to make, use, distribute, and import the Software solely for use with Cypress hardware products. Any other use, reproduction, modification, translation, or compilation of the Software is prohibited.

TO THE EXTENT PERMITTED BY APPLICABLE LAW, CYPRESS MAKES NO WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, WITH REGARD TO THIS DOCUMENT OR ANY SOFTWARE OR ACCOMPANYING HARDWARE, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. No computing device can be absolutely secure. Therefore, despite security measures implemented in Cypress hardware or software products, Cypress shall have no liability arising out of any security breach, such as unauthorized access to or use of a Cypress product. CYPRESS DOES NOT REPRESENT, WARRANT, OR GUARANTEE THAT CYPRESS PRODUCTS, OR SYSTEMS CREATED USING CYPRESS PRODUCTS, WILL BE FREE FROM CORRUPTION, ATTACK, VIRUSES, INTERFERENCE, HACKING, DATA LOSS OR THEFT, OR OTHER SECURITY INTRUSION (collectively, "Security Breach"). Cypress disclaims any liability relating to any Security Breach, and you shall and hereby do release Cypress from any claim, damage, or other liability arising from any Security Breach. In addition, the products described in these materials may contain design defects or errors known as errata which may cause the product to deviate from published specifications. To the extent permitted by applicable law, Cypress reserves the right to make changes to this document without further notice. Cypress does not assume any liability arising out of the application or use of any product or circuit described in this document. Any information provided in this document, including any sample design information or programming code, is provided only for reference purposes. It is the responsibility of the user of this document to properly design, program, and test the functionality and safety of any application made of this information and any resulting product. "High-Risk Device" means any device or system whose failure could cause personal injury, death, or property damage. Examples of High-Risk Devices are weapons, nuclear installations, surgical implants, and other medical devices. "Critical Component" means any component of a High-Risk Device whose failure to perform can be reasonably expected to cause, directly or indirectly, the failure of the High-Risk Device, or to affect its safety or effectiveness. Cypress is not liable, in whole or in part, and you shall and hereby do release Cypress from any claim, damage, or other liability arising from any use of a Cypress product as a Critical Component in a High-Risk Device. You shall indemnify and hold Cypress, its directors, officers, employees, agents, affiliates, distributors, and assigns harmless from and against all claims, costs, damages, and expenses, arising out of any claim, including claims for product liability, personal injury or death, or property damage arising from any use of a Cypress product as a Critical Component in a High-Risk Device. Cypress products are not intended or authorized for use as a Critical Component in any High-Risk Device except to the limited extent that (i) Cypress's published data sheet for the product explicitly states Cypress has qualified the product for use in a specific High-Risk Device, or (ii) Cypress has given you advance written authorization to use the product as a Critical Component in the specific High-Risk Device and you have signed a separate indemnification agreement.

Cypress, the Cypress logo, Spansion, the Spansion logo, and combinations thereof, WICED, PSoC, CapSense, EZ-USB, F-RAM, and Traveo are trademarks or registered trademarks of Cypress in the United States and other countries. For a more complete list of Cypress trademarks, visit [cypress.com](http://cypress.com). Other names and brands may be claimed as property of their respective owners.

Arm and Cortex are trademarks of Arm Limited (or its subsidiaries) in the US and/or elsewhere.

# Preface



## **Purpose of This Document and Intended Readers**

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

## **Organization of This Document**

This document consists of the following 4 chapters.

### **CHAPTER 1 MainFlash Memory**

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

### **CHAPTER 2 Flash Security**

The flash security feature provides possibilities to protect the content of the flash memory. This chapter 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 Cypress Serial Programmer.

## **Sample Programs and Development Environment**

Cypress offers sample programs free of charge for operating the peripheral functions of the FM4 family. Cypress 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 microcontroller.

## **Microcontroller Support Information:**

<http://www.cypress.com/cypress-microcontrollers>

## **Note:**

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

# How to Use This Document



## Searching for a Function

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

Search from the table of the contents

The table of the contents lists the document 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 "A. Register Map" of "APPENDIXES" in "FM4 Family Peripheral Manual".

## Terminology

This document 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 document 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
- RW : 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 document.

Example: bit7:0 indicates the bits from bit7 to bit0

The values such as for addresses are written as follows in this document.

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



<b>1. MainFlash Memory .....</b>	<b>6</b>
1.1 Overview .....	7
1.2 Configuration .....	8
1.3 Operating Description .....	13
1.4 Registers .....	42
<b>2. Flash Security .....</b>	<b>59</b>
2.1 Overview .....	60
2.2 Operation Explanation .....	60
<b>3. Serial Programming Connection .....</b>	<b>61</b>
3.1 Serial Programmer .....	62
<b>Revision History .....</b>	<b>71</b>
<b>Document Revision History .....</b>	<b>71</b>

# 1. MainFlash Memory



This series is equipped with 1064 KBytes to 2088 KBytes of MainFlash memory.

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

This series has built-in MainFlash memory with a capacity of 1064 KBytes to 2088 KBytes that supports data erasing by all sectors of each macro, data erasing by unit of sector, and data writing by the CPU.

Contents described with “flash memory” are information for the MainFlash memory in this chapter.

## 1.1 . Overview

## 1.2 . Configuration

## 1.3 . Operating Description

## 1.4 . Registers

## 1.1 Overview

This series is equipped with 1064 KBytes to 2088 KBytes of built-in MainFlash memory.

The built-in MainFlash memory can be erased data of sector-by-sector, all-sector of each macro batch erased data, and programmed data in units of half words (16 bits) by the Cortex-M4 CPU.

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

### Flash Memory Features

Usable capacity:

Minimum configuration: 1024 K + 40 Kbytes

Maximum configuration: 2048 K + 40 Kbytes

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:

Up to 72 MHz 0Wait

Up to 200 MHz Allowing Flash accelerator function (prefetch buffer/trace buffer) will achieve 0 Wait at high speed operational frequency

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 algorithm<sup>\*1</sup> to perform writing or erasing.

2. CPU programming mode

This mode allows reading, writing, and erasing of flash memory (automatic algorithm<sup>\*1</sup>). 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 algorithm<sup>\*1</sup>).

Built-in flash security function

(Prevents reading of the content of flash memory by a third party)

See "CHAPTER 2 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, it is not necessary to consider the error correction penalties during software development.

Built-in Dual flash mode

Dual flash mode allows accessing Flash Macro #0 and Flash Macro #1 independently.

Also this flash memory has Re-Map function. It allows assigning any macro to MainFlash memory area.

### 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 algorithm = Embedded Algorithm

## 1.2 Configuration

This series consists of 1064 KBytes to 2088 KBytes MainFlash memory area, a security code area, a High-Speed CR trimming data area, a HTM code area, and a general purpose data area. The MainFlash memory consists of Flash Macro #0 and Flash Macro #1. Also the security code area, the High-Speed CR trimming data area the HTM code area, and the general purpose data area are belonged to the Flash Macro #0.

Figure 1-1 to Figure 1-4 shows the address and sector structure of the MainFlash memory built into this series as well as the address of security/CR trimming data/HTM/general purpose data.

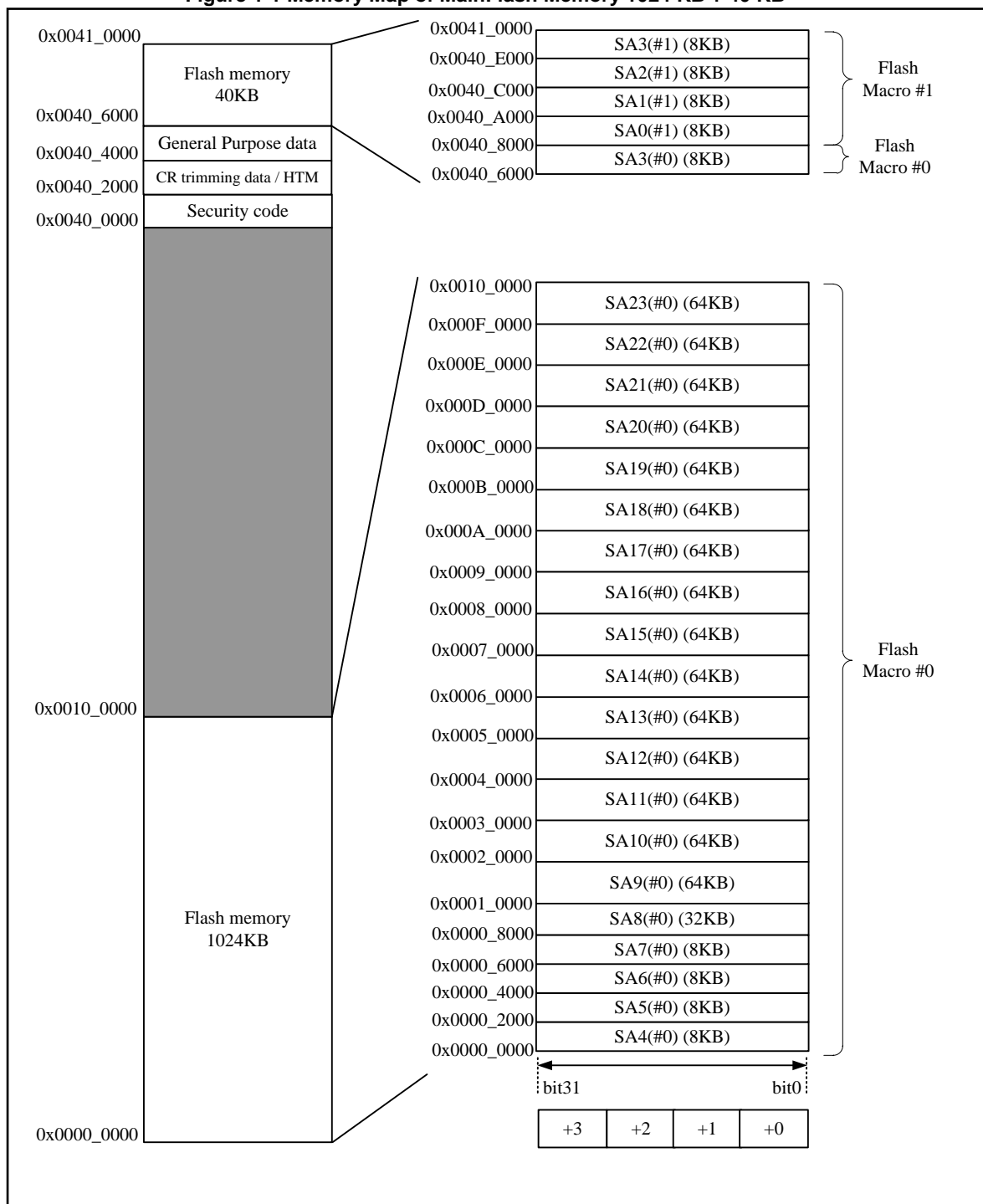
See "CHAPTER Flash Security" for details on the security.

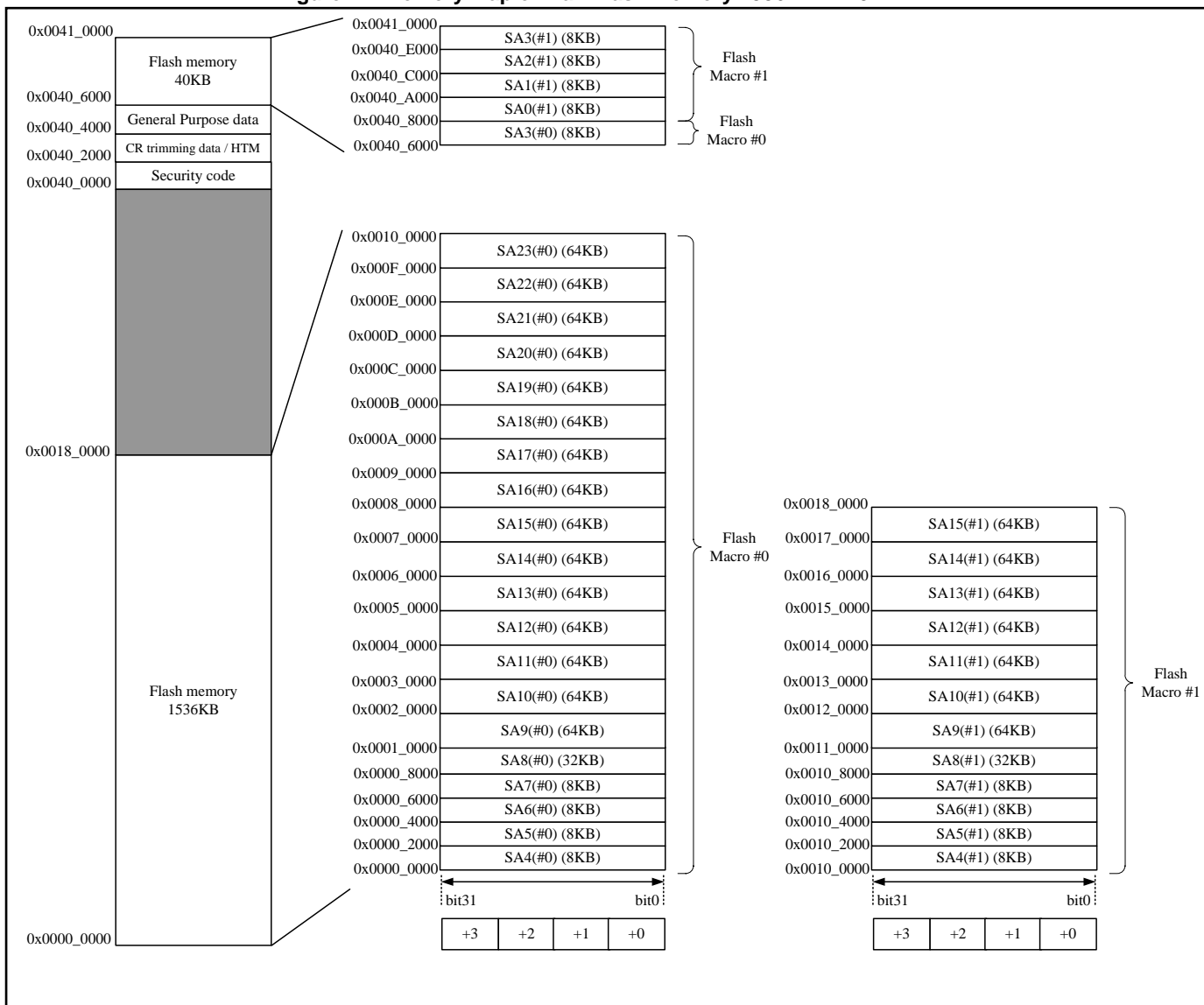
See Section "1.4.10 CRTRMM (CR Trimming Data Mirror Register)" and CHAPTER High-Speed CR Trimming" of the "FM4 Family Peripheral Manual" for details on the High-Speed CR trimming.

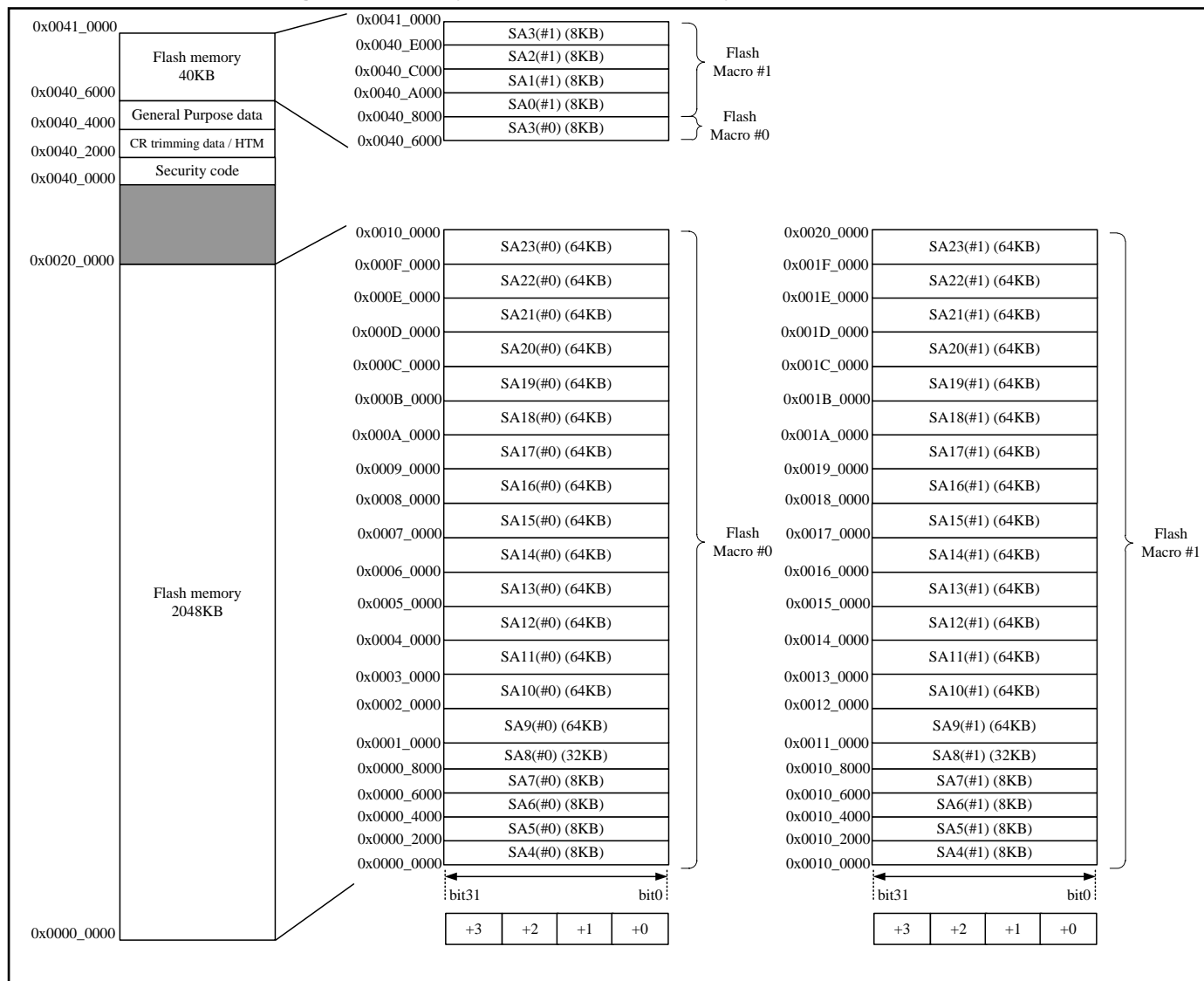
**Table 1-1 MainFlash Memory Capacity of Each Product**

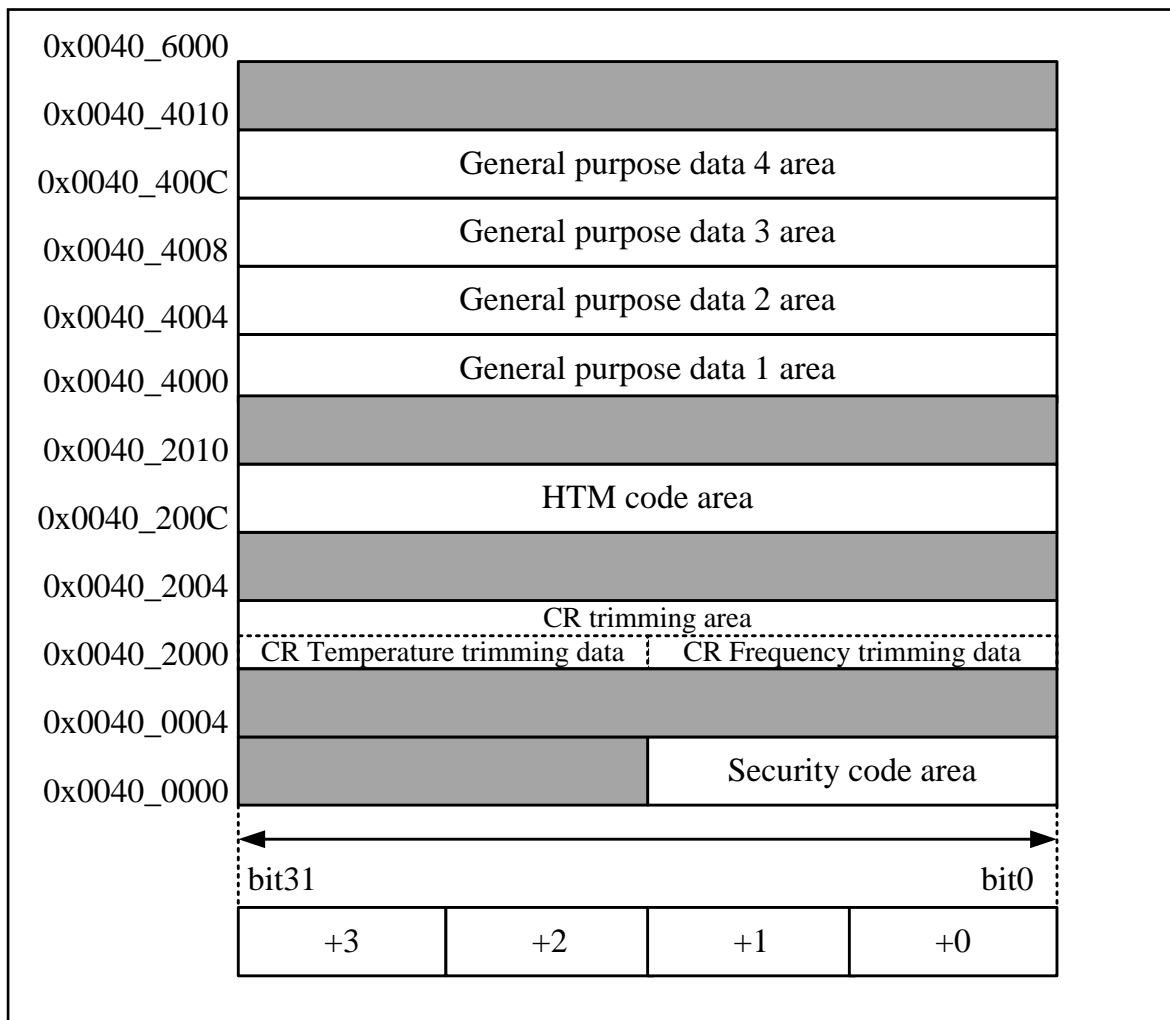
Memory Capacity	1024 KB + 40 KB	1536 KB + 40 KB	2048 KB + 40 KB
product	S6E2C18	S6E2C19	S6E2C1A
	S6E2C28	S6E2C29	S6E2C2A
	S6E2C38	S6E2C39	S6E2C3A
	S6E2C48	S6E2C49	S6E2C4A
	S6E2C58	S6E2C59	S6E2C5A
	S6E2CC8	S6E2CC9	S6E2CCA



**Figure 1-1 Memory Map of MainFlash Memory 1024 KB + 40 KB**


**Figure 1-2 Memory Map of MainFlash Memory 1536 KB + 40 KB**


**Figure 1-3 Memory Map of MainFlash Memory 2048 KB + 40 KB**


**Figure 1-4 Address of Security/CR Trimming Data/HTM/General Purpose Data**

**Figure 1-5 Bit Configuration of the CR Trimming Area**

bit	31	21	20	16	15	10	9	0
Field	Reserved				CR temperature trimming data			
	Reserved				CR Frequency trimming data			

If the HTM code of 0x0000\_0001 is written in the HTM code area of MainFlash memory, the HTM is enabled after turning on the power again.

See Section CHAPTER Debug Interface" of the "FM4 Family Peripheral Manual" for details on the HTM.

Table 1-2 shows the address of the HTM code area and the HTM code.

**Table 1-2 Address of HTM Code and HTM Code**

Address	HTM Code
0x0040_200C	0x0000_0001

## **1.3 Operating Description**

This section explains the MainFlash memory operation.

- 1.3.1 . MainFlash Memory Access Modes
- 1.3.2 . Automatic Algorithm
- 1.3.3 . Explanation of MainFlash Memory Operation
- 1.3.4 . Writing to MainFlash Memory in Products Equipped with ECC
- 1.3.5 . MainFlash Accelerator
- 1.3.6 . Dual flash mode
- 1.3.7 . Date buffer
- 1.3.8 . Cautions When Using MainFlash Memory

### 1.3.1 MainFlash Memory Access Modes

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

CPU ROM mode

CPU programming mode

These modes can be 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 "0b10" (32-bit read), and enables word access.

However, in this mode, it is not possible to execute commands, to activate the automatic 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 "0b01" (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 algorithm can be activated to write or erase data. See Section "1.3.2 Automatic Algorithm" for details on the automatic algorithm.

**Table 1-3 Access Modes of Flash Memory**

Access Mode	Access Size	Automatic Algorithm	Instruction Execution in the Flash Memory
CPU ROM mode	32-bit	disable	enable
CPU programming mode	16-bit	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 "0b10" and the flash memory returns to CPU ROM mode.

### 1.3.2 Automatic Algorithm

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

This section explains the automatic algorithm.

#### 1.3.2.1. Command Sequence

#### 1.3.2.2. Command Operating Explanations

#### 1.3.2.3. Automatic Algorithm Run States

### 1.3.2.1 Command Sequence

The automatic algorithm is activated by sequentially writing half-word (16-bit) data to the MainFlash memory one to six times in a row. This is called a command. Table 1-4 shows the command sequences.

**Table 1-4 Command sequence chart**

Command	No. 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	0xAA8	0xAA	0x554	0x55	0xAA8	0xA0	PA	PD	--	--	--	--
Flash erase	6	0xAA8	0xAA	0x554	0x55	0xAA8	0x80	0xAA8	0xAA	0x554	0x55	0xAA8	0x10
Sector erase	6	0xAA8	0xAA	0x 554	0x55	0xAA8	0x80	0xAA8	0xAA	0x554	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:

- In Table 1-4, the data notation only shows the lower 8 bits. The upper 8 bits can be set to any value.
- Write commands as half-words at any time.
- In Table 1-4, the address notation only shows the lower 16 bits. The upper 16 bits should be set to any address within the address range of the target flash macro. When the address outside the flash macro of flash address range is specified, the command sequence would not operate correctly since the flash memory cannot recognize the command.
- For the address when setting the flash security code, specify the address of "0x0040\_0000".  
While Re-Map function is enabled, specify the address of "200F\_8000"
- For the address when setting or erasing the CR trimming data, specify the address of "0x0040\_2000".  
While Re-Map function is enabled, specify the address of "200F\_A000"
- For the address when setting or erasing the HTM code, specify the address of "0x0040\_200C".  
While Re-Map function is enabled, specify the address of "200F\_A00C"
- When any of the CR trimming data and the HTM code is erased, both of which are erased. It is not possible to erase only one.
- For the address when setting or erasing the general purpose data, specify the address of "0x0040\_4000" to "0x0040\_400C". (general purpose data 1: "0x0040\_4000", general purpose data 2: "0x0040\_4004", general purpose data 3: "0x0040\_4008", general purpose data 4: "0x0040\_400C")  
While Re-Map function is enabled, specify the address of "0x200F\_C000" to "0x200F\_C00C". (general purpose data 1: "0x200F\_C000", general purpose data 2: "0x200F\_C004", general purpose data 3: "0x200F\_C008", general purpose data 4: "0x200F\_C00C")  
When any of the general purpose data is erased, all of the general purpose data are erased.



### 1.3.2.2 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.

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 algorithm exceeds the time limit, the flash memory is returned to the read/reset state by issuing the read/reset command.

See Section "1.3.3.1 Read/Reset Operation" for details on the actual operation.

#### **Program (Write) Command**

The automatic 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 the forth command issuance has finished, the automatic 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 "1.3.3.2 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.*

#### **Flash Erase Command**

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

#### **Note:**

- *All of the sectors in target flash macro are erased by the command. To erase all of the sectors in flash memory, it is necessary to issue the command to each Flash Macro #0 and Flash Macro #1.*

#### **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  $\mu$ s has elapsed (timeout interval), the automatic 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 35  $\mu$ s (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.

### 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 "1.3.3.5 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 flash erase or during write.

### 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 "1.3.3.6 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.

### 1.3.2.3 Automatic Algorithm Run States

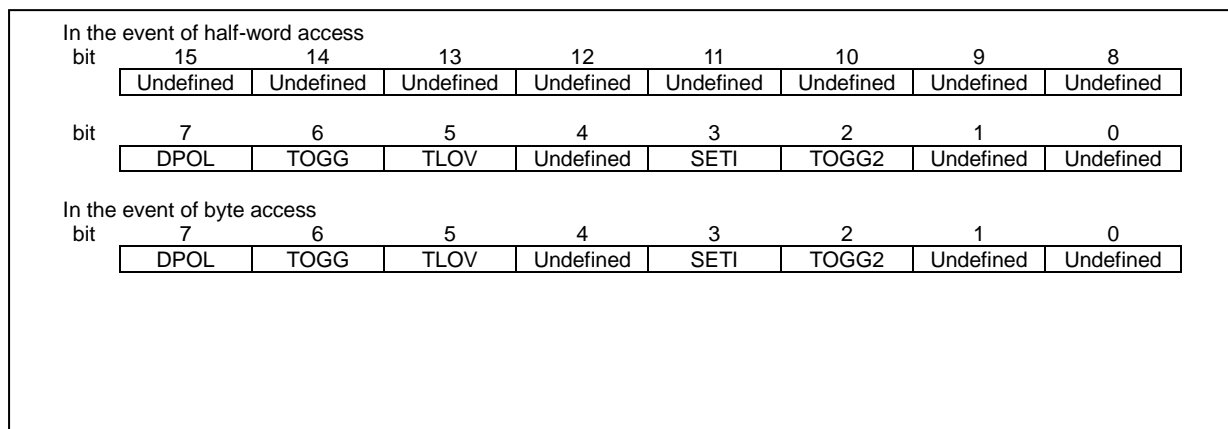
Because writing and erasing of flash memory is performed by the automatic algorithm, whether or not the automatic algorithm is currently executing can be checked using the flash ready bit (FSTR:RDY) and the operating status can be checked using the hardware sequence flags.

### Hardware Sequence Flags

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

Figure 1-6 shows the bit structure of the hardware sequence flags.

**Figure 1-6 Bit Structure of the Hardware Sequence Flags**



**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 MainFlash Memory

Table 1-5 shows the correspondence between each bit of the hardware sequence flags and the status of the flash memory.

**Table 1-5 List of Hardware Sequence Flag States**

State				DPOL	TOGG	TLOV	SETI	TOGG2
Running	Automatic write operation			Inverted data (*1)	Toggle	0	0	0
	Automatic Erase operation	Flash 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 be erased)	0	0	0	1	Toggle
			Read (Sector not to be erased)	Data (*1)	Data (*1)	Data (*1)	Data (*1)	Data (*1)
			Automatic write operation (Sector not to be erased)	Inverted data (*1)	Toggle	0	1	0
Time limit exceeded	Automatic write operation			Inverted data (*1)	Toggle	1	0	0
	Automatic 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 b

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 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 bit7 of data 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 bit7 of the address specified for reading the hardware sequence flags.

During sector erase

While sector erase is executing:

Reads out "0" from all areas of flash memory.

After sector erase finishes:

Always reads out "1".

During flash erase

While flash erase is executing: Always reads out "0".

After flash 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 bit7 of a specified address.

While write is in progress:

Reads out the opposite value (inverse data) of bit7 of data written at the last command sequence (PD).  
This does not access the address that was specified for reading the hardware sequence flags.

**Note:**

- *The data for a specified address cannot be read while the automatic algorithm is running. Confirm that the automatic 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 algorithm is currently running.

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

During write, sector erase, or flash erase

During write, sector erase, or flash erase:

When this bit is read out continuously, it alternately returns "1" and "0" (toggles). The address that was specified for reading the hardware sequence flags is not accessed.

After write, sector erase, or flash 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 bit6 of a specified address.

While write is in progress:

When this bit is read out continuously, it alternately returns "1" and "0" (toggles). The address that was specified for reading the hardware sequence flags is not accessed

**[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 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 flash 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 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 algorithm does not finish. In this case, the value of the DPOL bit remains invalid, and "1" and "0" are continuously read out alternately 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 bit5 of a specified address.

During writing:

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 algorithm is currently executing, that means a failure occurred during the write or erase.

**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  $\mu$ s 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 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 bit3 of a specified address.

During writing:

Reads out "1".

**[bit2] TOGG2: Toggle flag bit**

In the sector erase suspended state, a sector which is not the erase target can be read. However, the erase target sector cannot be read. This toggle bit flag can detect whether the corresponding sector is the erase target sector during the sector erase suspend by checking the toggle operation of the read data.

During writing

Reads out "0".

During sector erase or flash erase

When this bit is read out continuously, "1" and "0" are alternately read (toggle operation).

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, "1" and "0" are alternately read (toggle operation)

When this bit is read out by specifying an address in the sector other than specified as sector erase:

Reads out the value of bit2 of a specified address.

During writing:

Reads out "0".

**[bit1:0] Undefined bits**

### 1.3.3 Explanation of MainFlash Memory Operation

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

#### 1.3.3.1. Read/Reset Operation

#### 1.3.3.2. Write Operation

#### 1.3.3.3. Flash Erase Operation

#### 1.3.3.4. Sector Erase Operation

#### 1.3.3.5. Sector Erase Suspended Operation

#### 1.3.3.6. Sector Erase Restart Operation

#### *1.3.3.1 Read/Reset Operation*

This section explains the read/reset operation.

To place the flash memory in the read/reset state, send read/reset commands to the target sector consecutively. 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.

#### *1.3.3.2 Write Operation*

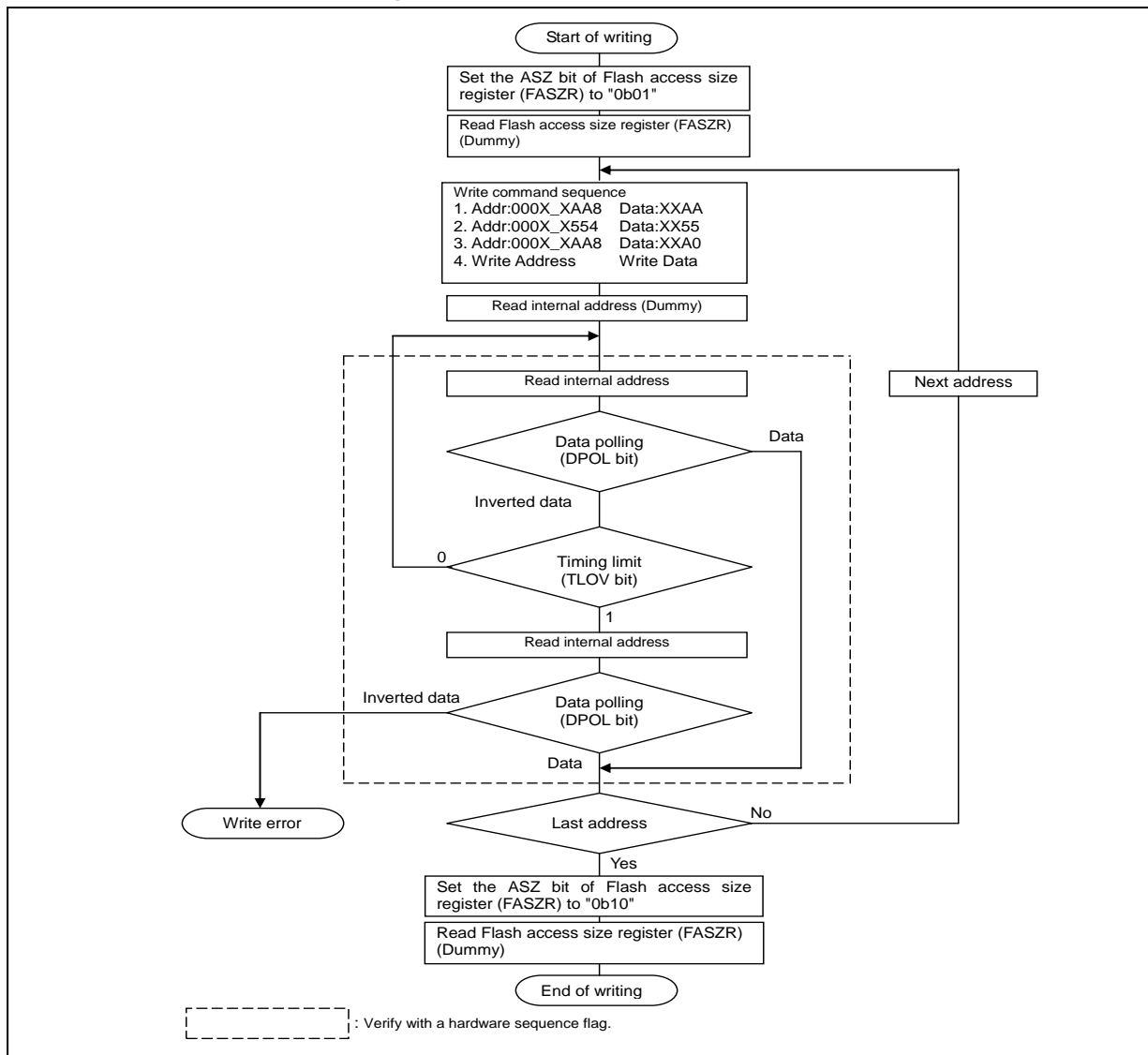
This section explains the write operation.

Writes are performed according to the following procedure.

1. The write command is issued to the target sector sequentially  
The automatic 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 bit7 (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 bit7 written at the last command sequence (PD) is read out.

Figure 1-7 shows an example of a write operation to the flash memory.

Figure 1-7 Example Write Operation


**Notes:**

- See Section "1.3.2 Automatic Algorithm" for details on the write command.
- The address notations in command sequences only show the lower 12 bits. The upper 20bits 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 would not operate correctly since the flash memory cannot recognize the 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 "1.3.4 Writing to MainFlash Memory in Products Equipped with ECC" for details on the procedure.
- You cannot rewrite to the address once you wrote to because the ECC (Error Correction Code) has been changed. To perform rewriting to the same address, erase the address (sector erase or flash erase) in advance.

### 1.3.3.3 Flash Erase Operation

This section explains the flash erase operation.

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

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

See Section "1.3.2 Automatic Algorithm" for details on the flash erase command.

1. Issue the flash erase command sequentially to the target sector

The automatic algorithm is activated and the flash 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 bit7 (the DPOL bit) of the data that was read is "1", that means the flash erase has finished.

The time required to erase the flash is "sector erase time × total number of sectors + flash write time (preprogramming)". Once the flash erase operation has finished, the flash memory returns to read/reset mode.

### 1.3.3.4 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

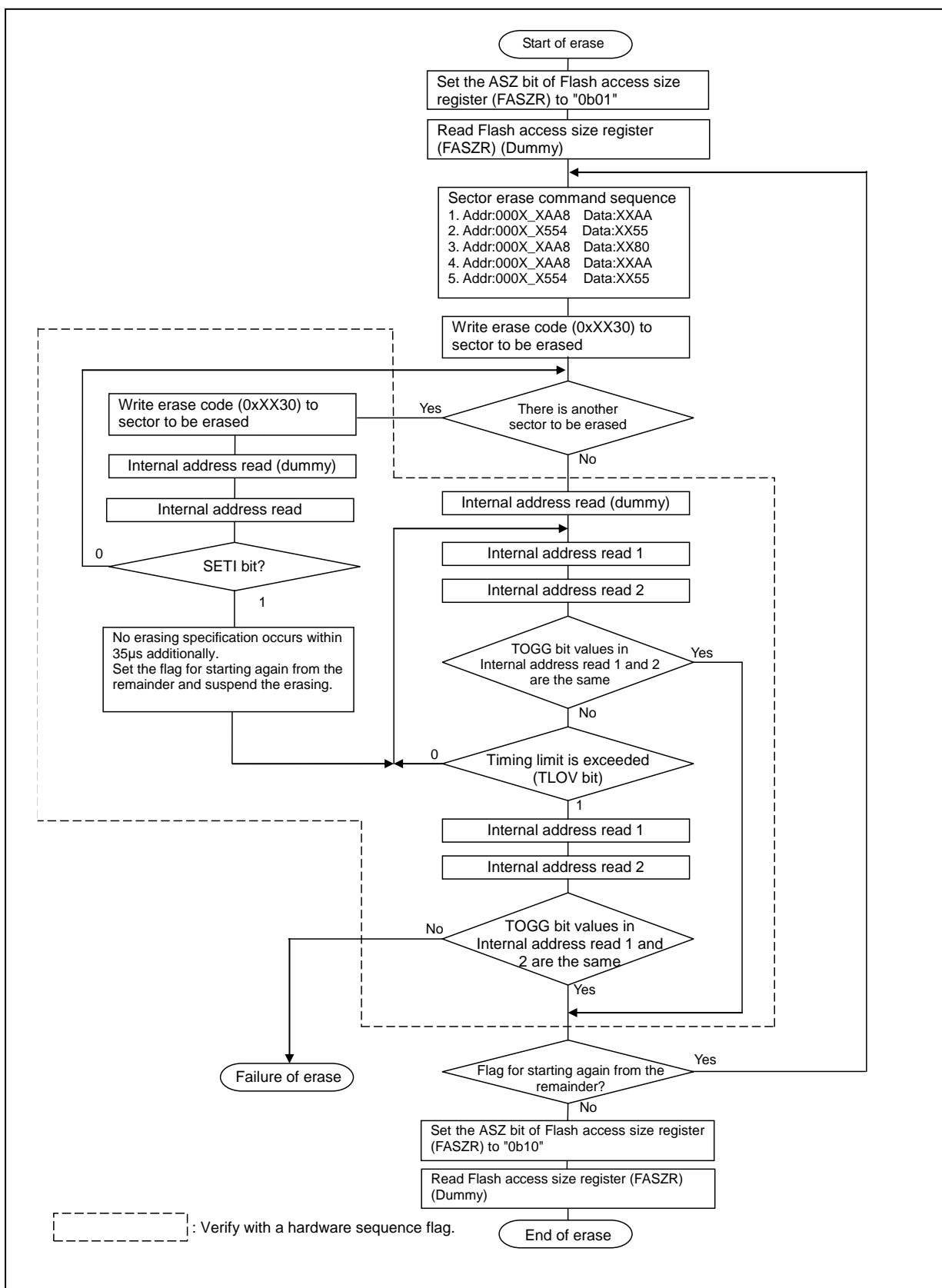
Once 35μs has elapsed (the timeout interval), the automatic algorithm activates and the sector erase operation begins. To erase multiple sectors, issue the erase code (0x30) to an address in the sector to erase within 35μs (the timeout interval). If the code is issued after the timeout interval has elapsed, 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 bit7 (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 1-8 shows an example of the sector erase procedure for the case of using the TOGG bit for confirmation.

Figure 1-8 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 "1.3.2 Automatic Algorithm" for details on the sector erase command.
- The address notations in command sequences only show the lower 12 bits. The upper 20bits 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 would not operate correctly since the flash memory cannot recognize the 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.

### 1.3.3.5 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.

#### 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 35  $\mu$ s until erasing is actually stopped.

**Notes:**

- See Section "1.3.2 Automatic 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. Flash erase cannot be suspended. In addition, even if the sector erase suspended command is issued again during sector erase suspended, it is ignored.

#### 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 erase command is ignored in the sector erase suspended state.

### 1.3.3.6 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 "1.3.2 Automatic 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 2 ms 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 2 ms after the sector erase restart command is issued.*

### 1.3.4 Writing to MainFlash Memory in Products Equipped with ECC

This section explains the writing to MainFlash 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="0b01"/DFASZR:DASZ="0b01")  
Perform a dummy read, after setting the FASZR/DFASZR register.
2. Issue a write command. Write address = PA, Write data = PD[15:0]  
See Section "1.3.3.2 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 "1.3.2.3 Automatic 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="0b10"/DFASZR:DASZ="0b10")  
Perform a dummy read, after setting the FASZR/DFASZR 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: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

#### Notes:

- The flash access size setting in the DualFlash area is set by the dual flash access size bits (DFASZR:DASZ)
- You cannot rewrite to the address once you wrote to because the ECC (Error Correction Code) has been changed. To perform rewriting to the same address, erase the address (sector erase or flash erase) in advance.

### 1.3.5 MainFlash Accelerator

This section explains the MainFlash accelerator.

This series is equipped with Flash accelerator for instruction code to achieve 0 wait at high speed operation (MAX: 200 MHz).

The Flash accelerator has the following functions:

1. Prefetch Buffer  
Addresses will be prefetched to save the instructions in the prefetch buffer. The prefetch buffer consists of 128 bits × 4. If the address hits in this buffer, the value will be output with 0 Wait.
2. Trace Buffer  
16 Kbyte RAM is employed for trace buffer. Values read from the Flash memory will be stored in this buffer at all times. After instruction fetch, if the value has been stored in the trace buffer, it becomes buffer hit and output the value with 0 Wait.

**Notes:**

- *Number of bits and columns of the prefetch buffer varies depending on each series of FM4. Number of CPU cycles is different even if it is the same program depending on each series. For detail, see “Flash Programming Specifications” of each series.*

Flash Accelerator operating flow at RWT="0b10" in FRWTR register and the number of Wait are shown in [Figure 1-9](#).

Prefetch buffer access occurs at initial state. If the address do not hit in the prefetch buffer, it becomes prefetch miss. Then, it waits for one cycle and the access is switched to the trace buffer. However, if the value is hit in the trace buffer, it becomes buffer hit and outputs the value stored in the trace buffer with 0 Wait.

If the address do not hit in the trace buffer and a buffer miss occurs, the access will be switched to one for prefetch buffer again. In that time, the access to the flash memory occurs and the wait cycle of 4 or 5 cycle wait is generated.

If the address do not hit in both prefetch buffer and trace buffer, 3 or 4 cycle wait for flash memory access is generated.

When the trace buffer function is disabled by register setting (See Section "1.4.5 FBFCR (Flash Buffer Control Register)"), switch from prefetch buffer to trace buffer does not occur. At the prefetch miss, it requires 3 or 4 cycle wait cycle for flash memory access.

Flash Accelerator operating flow at RWT="0b11" in FRWTR register and the number of Wait are shown in [Figure 1-10](#). The number of Wait is different from RWT="0b10" in FRWTR register.

Figure 1-9 Flash Accelerator Operating Flow (FRWTR.RWT="0b10")

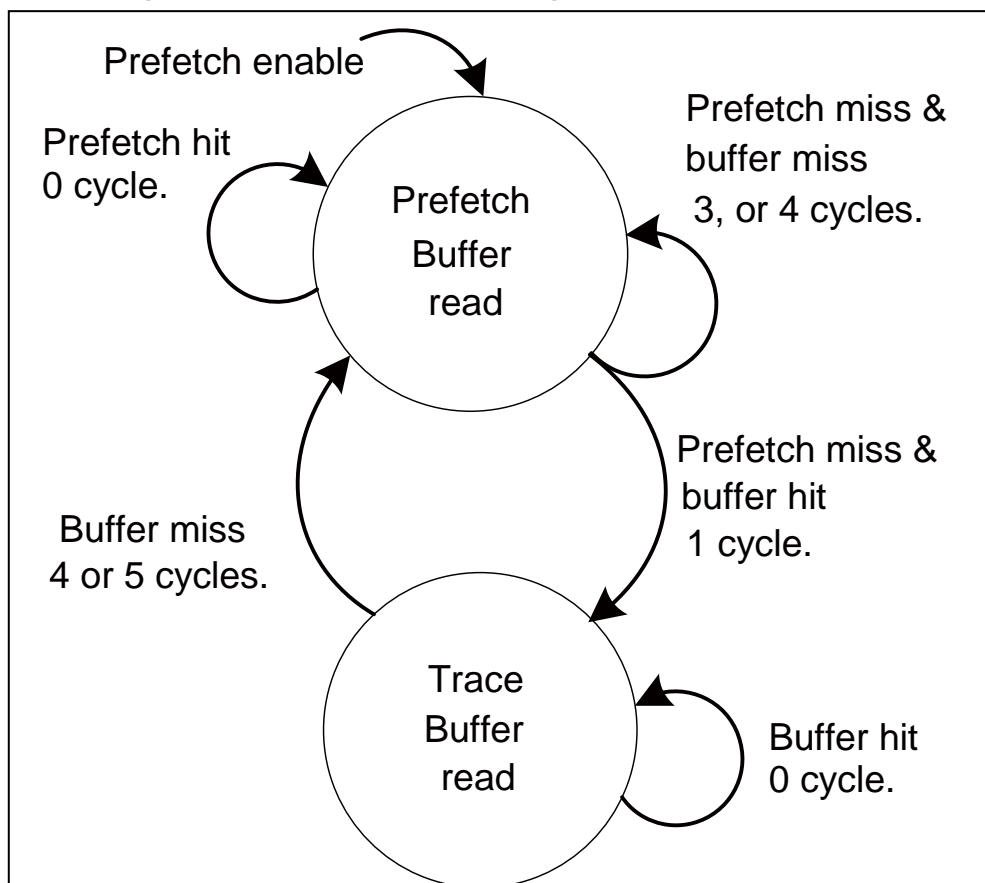
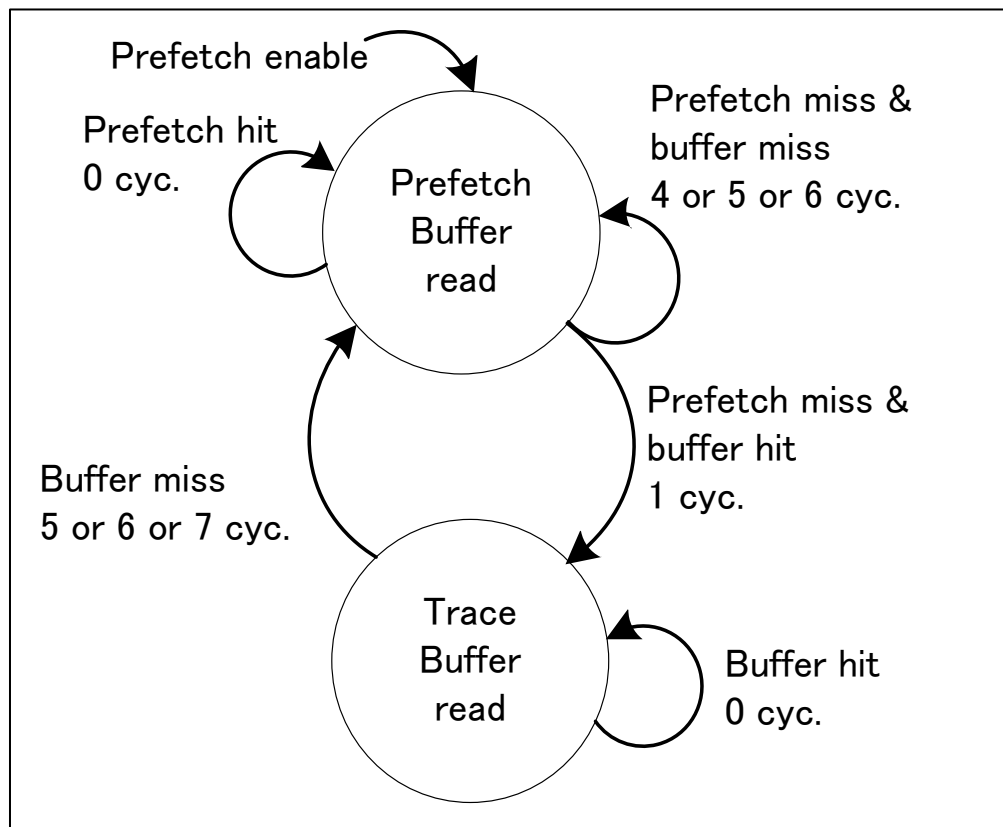


Figure 1-10 Flash Accelerator Operating Flow (FRWTR.RWT="0b11")



After a reset, RWT bits in FRWTR register becomes "0b11" to enter flash accelerator mode and operate the prefetch buffer function but the trace buffer function has still been stopped. In order to activate this function, "1" must be written to BE bit in FBFCR (Flash Buffer Control Register). See "1.4.5 FBFCR (Flash Buffer Control Register)" for details.



### 1.3.6 Dual flash mode

This section explains the Dual flash mode.

#### 1.3.6.1. Configuration (Dual flash mode)

#### 1.3.6.2. Re-Map function

#### 1.3.6.3. Access to the DualFlash area

#### 1.3.6.4. Setting Procedure

Depending on the product, there are some restrictions on this mode.

**Table 1-6 In Dual Flash Mode, the Restrictions of Each Product**

Memory Capacity	1024 KB + 40 KB	1536 KB + 40 KB	2048 KB + 40 KB
product	S6E2C18 S6E2C28 S6E2C38 S6E2C48 S6E2C58 S6E2CC8	S6E2C19 S6E2C29 S6E2C39 S6E2C49 S6E2C59 S6E2CC9	S6E2C1A S6E2C2A S6E2C3A S6E2C4A S6E2C5A S6E2CCA
Dual flash mode (DFCTRLR.DFE="1")	Setting is permitted	Setting is permitted	Setting is permitted
Re-Map function (DFCTRLR.RME="1")	Setting is prohibited	Setting is prohibited	Setting is permitted

#### 1.3.6.1 Configuration (Dual flash mode)

When the dual flash mode is enabled (DFCTRLR : DFE="1"), the address and sector structure of the MainFlash memory is changed.

The dual flash mode allows accessing MainFlash area and DualFlash area independently.

While dual flash mode, [Figure 1-11](#) to [Figure 1-13](#) shows the address and sector structure of the MainFlash memory built into this series as well as the address of security/CR trimming data/HTM/general purpose data.

**Note:**

- While the dual flash mode is disabled, access to the DualFlash area is prohibited.

Figure 1-11 Memory Map of MainFlash Memory 1024 KB + 40 KB in Dual Flash Mode

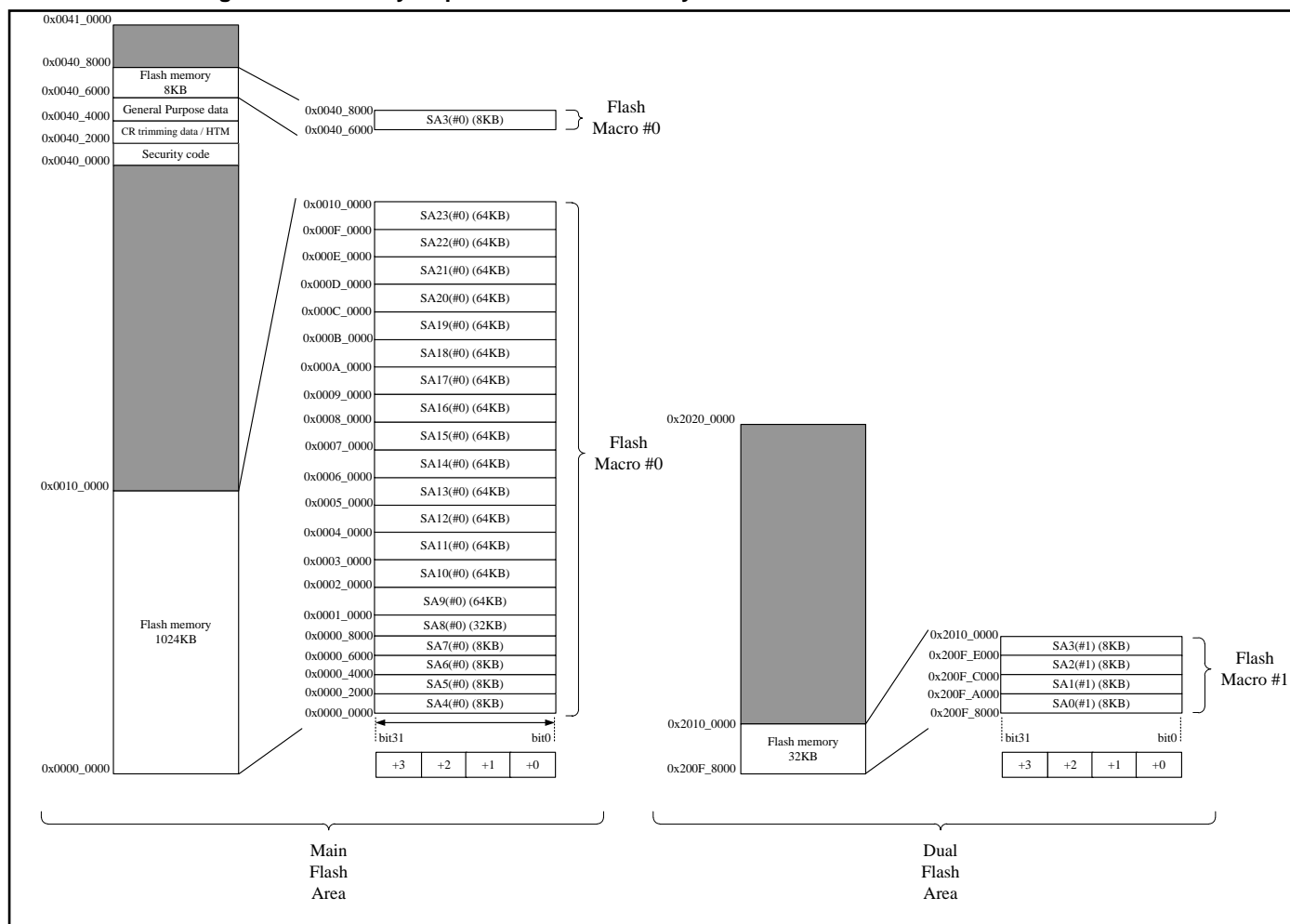


Figure 1-12 Memory Map of MainFlash Memory 1536 KB + 40 KB in Dual Flash Mode

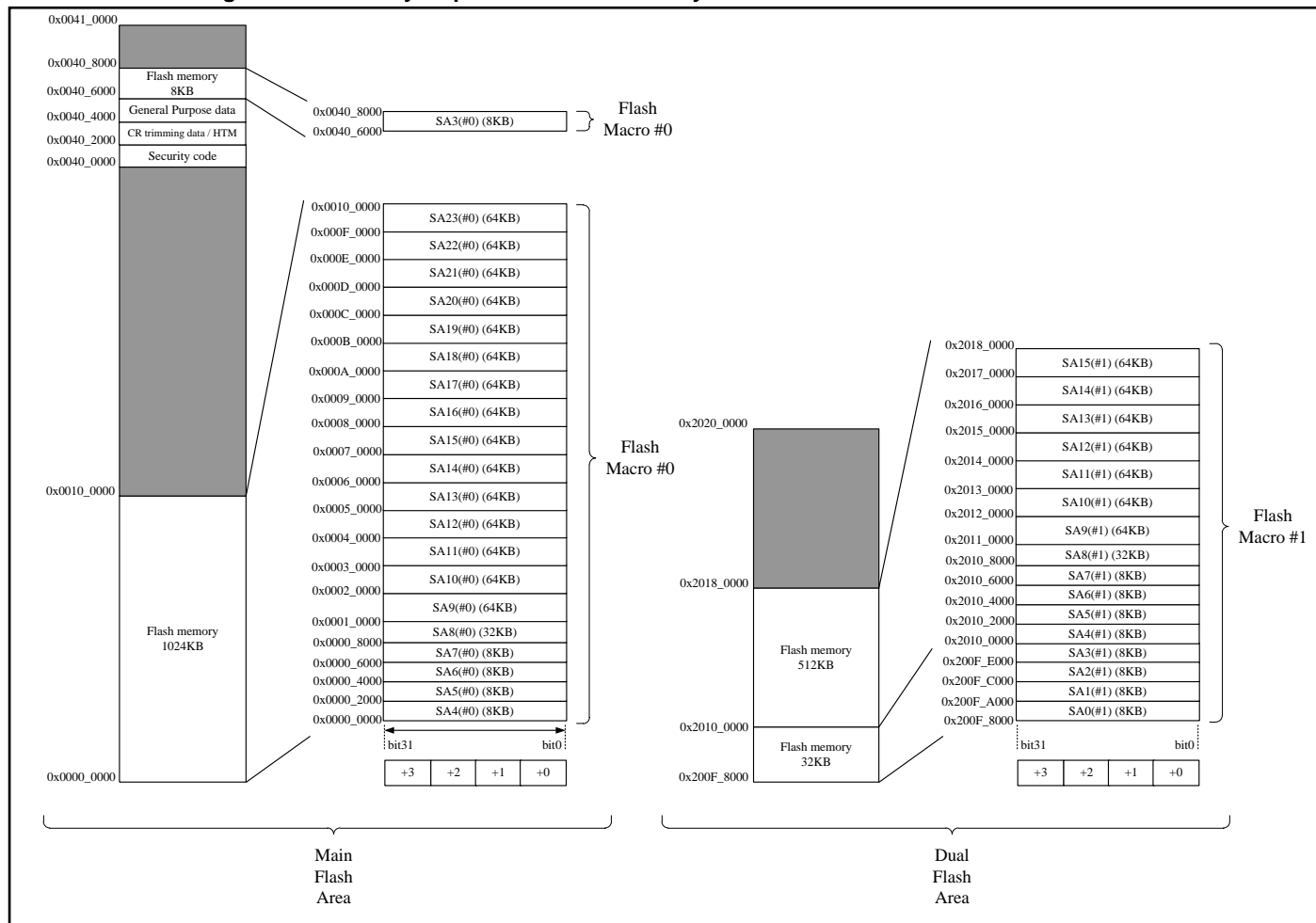
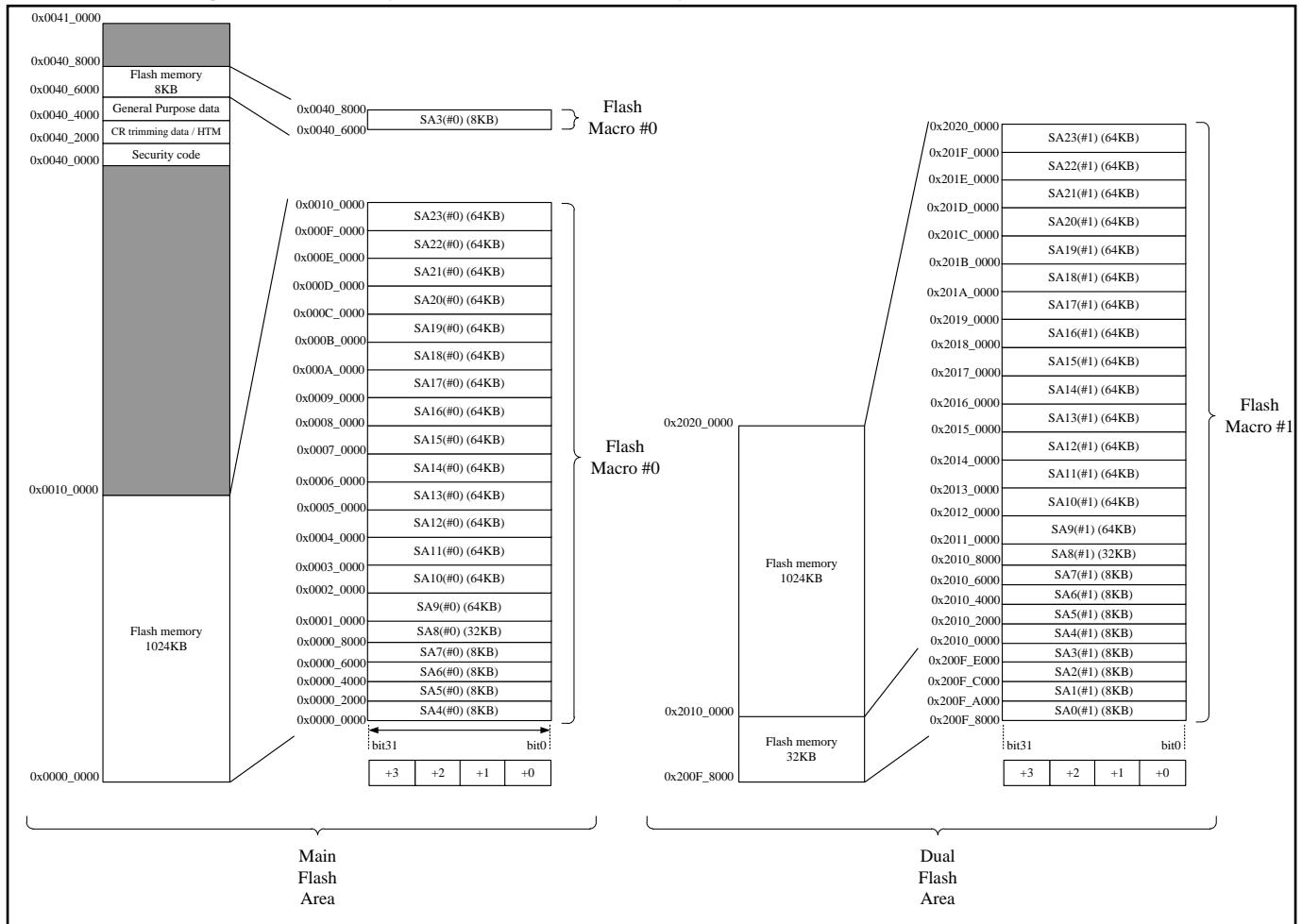


Figure 1-13 Memory Map of MainFlash Memory 2048 KB + 40 KB in Dual Flash Mode



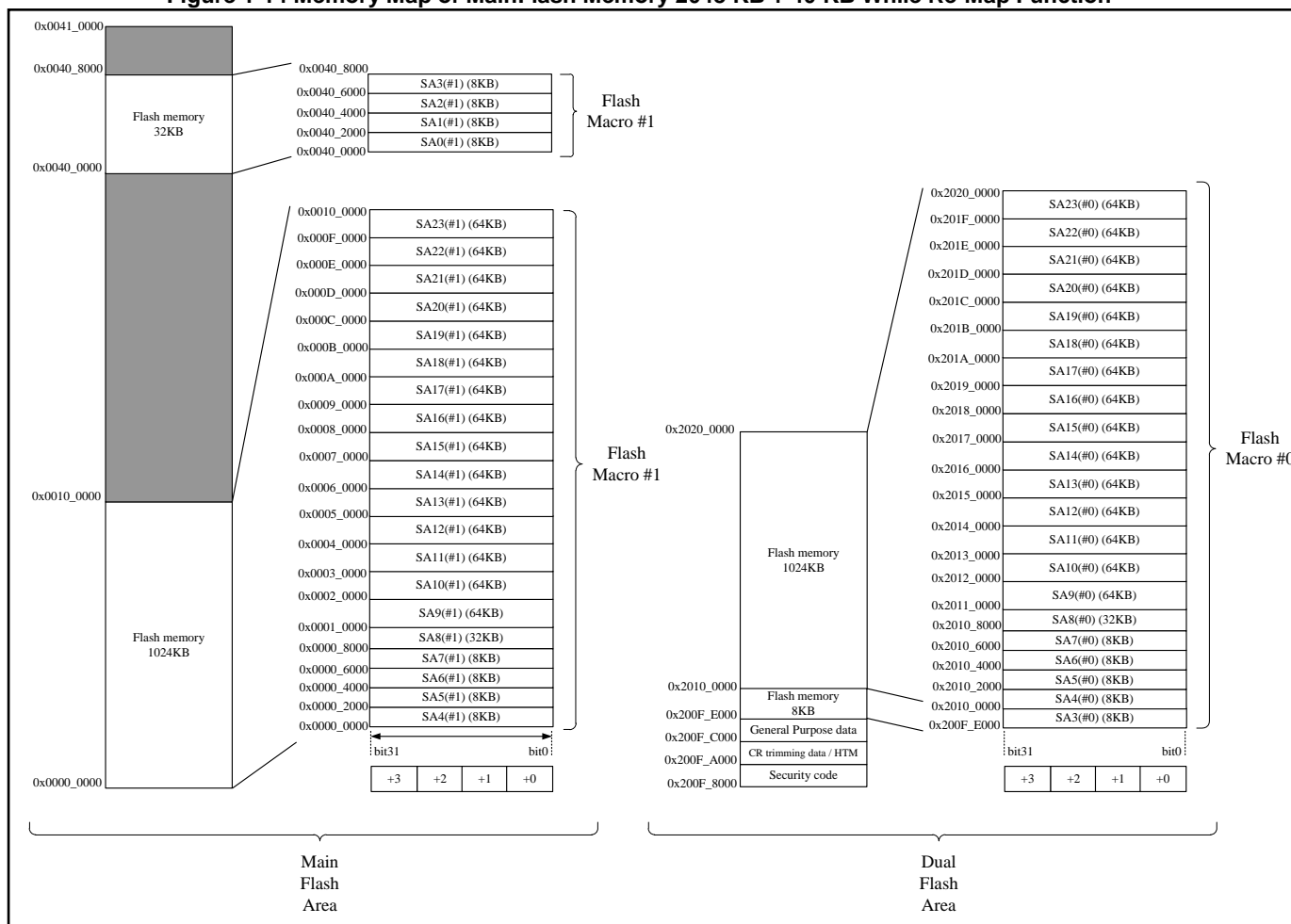
### 1.3.6.2 Re-Map function

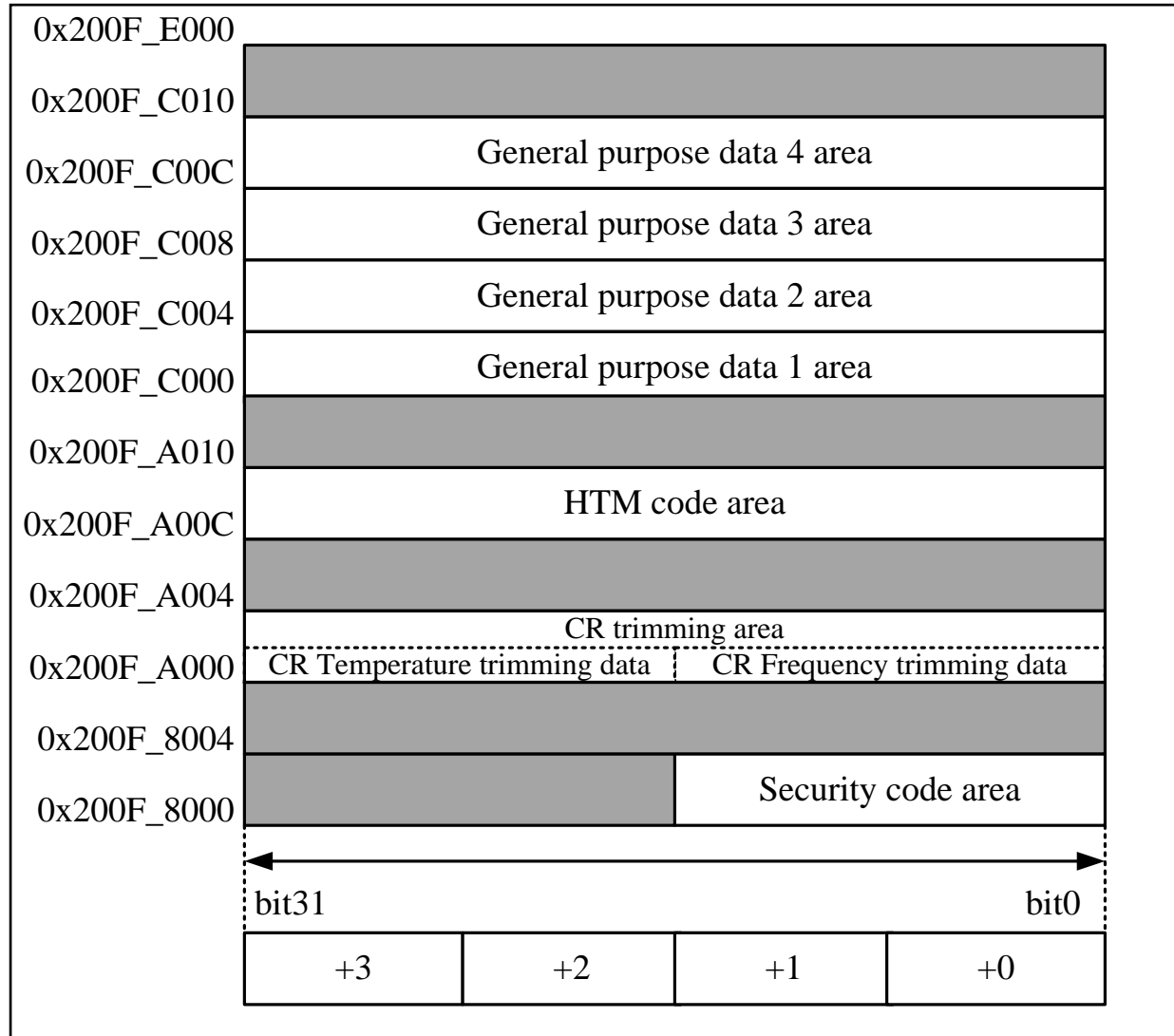
When Re-Map function is enabled (DFCTRLR : RME="1"), Flash Macro #1 is assigned to the MainFlash area.

In addition, Flash Macro #0 is assigned to the DualFlash area.

Figure 1-14 and Figure 1-15 show the address and sector structure of the MainFlash memory as well as the address of security/CR trimming data/HTM/general purpose data while Re-Map function is enabled.

**Figure 1-14 Memory Map of MainFlash Memory 2048 KB + 40 KB While Re-Map Function**



**Figure 1-15 Address of Security/CR Trimming Data/HTM/General Purpose Data While Re-Map Function**

**Note:**

- When Re-Map function is enabled, the address of security/CR trimming data/general purpose data is assigned to the DualFlash area. Be careful not to accidentally perform deleting or writing.

### 1.3.6.3 Access to the DualFlash area

The following two access modes are available for accessing the DualFlash area from the CPU.

CPU ROM mode

CPU programming mode

These modes can be selected by the dual flash access size bits (DFASZR:DASZ).

#### **CPU ROM Mode**

This mode only allows reading of flash memory data.

This mode is entered by setting the dual flash access size bits (DFASZR:DASZ) to "0b10" (32-bit read), and enables word access.

However, in this mode, it is not possible to execute commands, to activate the automatic 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 dual flash access size bits (DFASZR: DASZ) to "0b01" (16-bit read/write), and enables flash programming.

Because word access is not possible in this mode, programs that are contained in the DualFlash area 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 algorithm can be activated to write or erase data. See Section "[1.3.2 Automatic Algorithm](#)" for details on the automatic algorithm.

**Table 1-7 Access Modes of Flash Memory**

Access Mode	Access Size	Automatic Algorithm	Instruction Execution in the DualFlash Area
CPU ROM mode	32-bit	disable	enable
CPU programming mode	16-bit	enable	Prohibited

Flash Accelerator/Sync Down/Interrupt/ECC ERR Address Capture functions are not equipped in the DualFlash area.

Set the Dual Flash Read Wait Cycle bit (DFRWTR : DRWT) for the access method to the DualFlash area.

### 1.3.6.4 Setting Procedure

This section explains the setting procedure for the DualFlash mode.

In addition, writing to the DFCTRLR register is not possible when the DualFlash mode is enabled.

In order to perform writing to the DFCTRLR register again, issue a reset command.

1. Release a reset (disable the DualFlash mode)
2. JMP for programs other than those in the Flash area.
3. Set the DualFlash mode (DFCTRLR : WKEY="0xEACC", RME="0"/"1", DFE="1")
4. JMP for programs in the MainFlash area.
5. Set Flash Accelerator and trace buffer function
6. Set Access Size (DFASZR register) and Read Wait (DFRWTR register) of DualFlash area
7. Issue a reset command if changing the DualFlash mode setting

#### Notes:

- Do not perform DualFlash mode settings after the trace buffer function is enabled (FBFCR : BE="1").
- Do not perform DualFlash mode settings with instructions written in flash memory. Re-write programs other than those in the Flash area.
- Writing to the DFCTRLR register is invalid when the DualFlash mode is enabled (DFCTRLR : DFE="1"). In order to perform writing again, issue a reset command.
- Do not perform an access to the DualFlash area when the DualFlash mode is disabled.
- DualFlash mode settings are invalid in any mode other than user mode.

### 1.3.7 Data buffer

This section explains the data buffer.

This series is equipped with data buffer of 128 bits × 2 in D-Code bus and DualFlash area.

1. D-Code bus data buffer  
 When the mode is CPU ROM mode (FASZR:ASZ="0b10") and FRWTR register RWT="0b10"/"0b11", D-Code bus data buffer is enabled.  
 Up to 2 sets of data read from D-Code bus in the past is stored in 128-bit units. If the address hits in this buffer, it becomes buffer hit and the value is output with 0 Wait.  
 In addition, FASZR register, FRWTR register, and DFCTRLR register is rewritten, the data stored in the data buffer is cleared.
2. DualFlash data buffer  
 When the mode is DualFlash mode (DFCTRLR: DFE="1") with CPU ROM mode (DFASZR:DASZ="0b10") and DFRWTR register DRWT="0b10"/"0b11", the data buffer in DualFlash area is enabled.  
 Up to 2 sets of data read from DualFlash area in the past is stored in 128-bit units. If the address hits in this buffer, it becomes buffer hit and the value is output with 0 Wait.  
 In addition, DFASZR register, and DFRWTR register is rewritten, the data stored in the data buffer is cleared.

#### Notes:

- For data buffer, data is stored in 128-bit units. Any data cannot be stored transcending the address boundary of 128-bits.
- Number of bits and columns of the Data buffer varies depending on each series of FM4. Number of CPU cycles is different even if it is the same program depending on each series.  
 For detail, see "Flash Programming Specifications" of each series.



### 1.3.8 Cautions When Using MainFlash Memory

This section explains the cautions when using MainFlash 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 Watchdog Timer from occurring during the writing and deleting.

If the CPU programming mode is configured (ASZ="0b01") in the ASZ[1:0] bits of the flash access size register (FASZR), do not execute any programs in the flash memory except the DualFlash area. The correct values will not be retrieved and the program will run out of control.

If the CPU programming mode is configured (ASZ="0b01") in the ASZ[1:0] bits of the flash access size register (FASZR) and the interrupt vector table is in the flash memory except the DualFlash area, ensure that no interrupt requests occur. The correct values will not be retrieved and the program will run out of control.

If the CPU programming mode is configured (DASZ="0b01") in the DASZ[1:0] bits of the dual flash access size register (DFASZR), do not execute any programs in the DualFlash area. The correct values will not be retrieved and the program will run out of control.

If the CPU programming mode is configured (DASZ="0b01") in the DASZ[1:0] bits of the dual flash access size register (DFASZR) and the interrupt vector table is in the DualFlash area, ensure that no interrupt requests occur. The correct values will not be retrieved and the program will run out of control.

If the CPU programming mode is configured (ASZ="0b01") in the ASZ[1:0] bits of the flash access size register (FASZR), do not transition to low power consumption mode.

If the CPU ROM mode is configured (ASZ="0b10") 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="0b01") 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.

If the CPU programming mode is configured (DASZ="0b01") in the DASZ[1:0] bits of the dual flash access size register (DFASZR), do not transition to low power consumption mode.

If the CPU ROM mode is configured (DASZ="0b10") in the DASZ[1:0] bits of the dual flash access size register (DFASZR), do not write to the flash memory.

If the CPU programming mode is configured (DASZ="0b01") in the DASZ[1:0] bits of the dual flash access size register (DFASZR), always write to the flash memory in half-words. Do not write in bytes.

Immediately after issuing the automatic 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 algorithm command, the read value cannot be guaranteed.

If the device is forced to transit to the low power consumption mode, ensure the operations of the flash memory automatic algorithm is completed. See "CHAPTER Low Power Consumption Mode" of the "FM4 Family Peripheral Manual" for details on the low power consumption mode.

Since ECC bits are added in this series, it is necessary to perform data programming in unites of 32 bits by using 2 times for 16bit writes. See Section ["1.3.4 Writing to MainFlash Memory in Products Equipped with ECC"](#) for details on the procedure.

You cannot rewrite to the address once you wrote to because the ECC (Error Correction Code) has been changed. To perform rewriting to the same address, erase the address (sector erase or flash erase) in advance.

## 1.4 Registers

This section explains the registers.

### List of Registers

Abbreviated Register Name	Register Name	Reference
FASZR	Flash Access Size Register	1.4.1
FRWTR	Flash Read Wait Register	1.4.2
FSTR	Flash Status Register	1.4.3
FSYNDN	Flash Sync Down Register	1.4.4
FBFCR	Flash Buffer Control Register	1.4.5
FICR	Flash Interrupt Register	1.4.6
FISR	Flash Interrupt Status Register	1.4.7
FICLR	Flash Interrupt Clear Register	1.4.8
DFCTRLR	Dual Flash mode Control Register	1.4.9
CRTRMM	CR Trimming Data Mirror Register	1.4.10
FGPDM1	Flash General Purpose Data Mirror Register1	1.4.11
FGPDM2	Flash General Purpose Data Mirror Register2	1.4.12
FGPDM3	Flash General Purpose Data Mirror Register3	1.4.13
FGPDM4	Flash General Purpose Data Mirror Register4	1.4.14
FERRAD	Flash ECC ERR Address Capture Register	1.4.15
DFASZR	Dual Flash Access Size Register	1.4.16
DFRWTR	Dual Flash Read Wait Register	1.4.17
DFSTR	Flash Status Register	1.4.18

### 1.4.1 FASZR (Flash Access Size Register)

This section explains the FASZR.

This register configures the access size for flash memory except DualFlash area. After reset is released, ASZ is set to "0b10" (32-bit read), and the flash memory enters CPU ROM mode. To put the flash memory into CPU programming mode, set ASZ to "0b01".

bit	7	6	5	4	3	2	1	0
Field	Reserved						ASZ	
Attribute							RW	RW
Initial Value							1	0

#### [bit7:2] Reserved bits

The read values are undefined. Ignored on write.

#### [bit1:0] ASZ: Access Size

Specifies the access size of the flash memory.

Field	bit	Description
ASZ	1:0	Flash Access Size 00: Setting prohibited 01: 16-bit read/write (CPU programming mode) 10: 32-bit read (CPU ROM mode: Initial value) 11: Setting prohibited

#### Notes:

- When ASZ is set to "0b01", 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 area except for flash memory.
- Perform a dummy read to register, after changing this register.
- When ASZ="0b01", BS bit and BE bit in FBFCR register are both cleared to "0", and the trace buffer function is set to OFF.

## 1.4.2 FRWTR (Flash Read Wait Register)

This section explains the FRWTR.

This register is effective when ASZ="0b10" (32-bit read mode).

It configures the access method for flash memory except DualFlash area.

bit	7	6	5	4	3	2	1	0
Field	Reserved						RWT	
Attribute							RW	RW
Initial Value							1	1

### [bit7:2] Reserved bits

The read values are undefined. Ignored on write.

### [bit1:0] RWT: Read Wait Cycle

Specifies the access method for flash memory.

Field	bit	Description
RWT	1:0	Read Wait Cycle 00: 0 cycle wait mode (0 latency) This setting can be used when HCLK is 72 MHz or less. 01: Setting prohibited 10: Flash Accelerator mode 0 This setting can be used when HCLK is 160MHz or less. This setting must be used when HCLK is over 72 MHz. 11: Flash Accelerator mode 1 (Initial value) This setting must be used when HCLK is over 160 MHz

In flash accelerator mode, allowing operating Flash Accelerator prefetch buffer function achieves 0 Wait at high speed operational frequency (up to 200 MHz).

After the Flash Accelerator mode is allowed, allowing operating Flash Accelerator trace buffer function (See Section "[1.4.5 FBFCR \(Flash Buffer Control Register\)](#)") achieves additional progress of performance.

When HCLK is 72 MHz or less, 0 cycle wait mode (RWT = "0b00") is suitable for CPU operation.

In flash accelerator mode, allowing operating the data buffer function. (See Section "[1.3.7 Data buffer](#)")

### Notes:

- Do not set RWT to "0b00"(0 cycle wait mode) if HCLK exceeds 72 MHz.  
While RWT setting is "0b00", HCLK must not exceed 72 MHz.
- Do not set RWT to "0b10"(flash accelerator mode 0) if HCLK exceeds 160 MHz.  
While RWT setting is "0b10", HCLK must not exceed 160 MHz.
- Perform a dummy read to register, after changing this register.

### 1.4.3 FSTR (Flash Status Register)

This section explains the FSTR.

This is a status register of flash memory except DualFlash area.

bit	7	6	5	4	3	2	1	0
Field	Reserved					ERR	HNG	RDY
Attribute						RW	R	R
Initial Value						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
ERR	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 Flag Bit"). If this bit becomes "1", issue a reset command. (See Section "[1.3.2.1 Command Sequence](#)")

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

### 1.4.4 FSYNNDN (Flash Sync Down Register)

This section explains the FSYNNDN.

The wait cycle is inserted in the read access to the flash memory at the CPU ROM mode. Current 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		
Attribute						RW	RW	RW
Initial Value						0	0	0

#### [bit7:3] Reserved bits

The read values are undefined. Ignored on write.

#### [bit2:0] SD: Sync Down

The wait cycle is inserted in the lead access of the flash memory.

Field	bit	Description
SD	2:0	000: 0(Initial value) 001: +1 Wait 010: Setting is prohibited. 011: +3 Wait 100: Setting is prohibited. 101: +5 Wait 110: Setting is prohibited. 111: +7 Wait

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

Example)

RWT="0b00" (0cycle wait and SD="0b011", 0+3=3 wait

#### Notes:

- This register is valid only when RWT bits in FRWTR register is set to "00". In Flash Accelerator mode (RWT="0b10"/RWT="0b11"), the value of this register is ignored.
- Perform a dummy read to register, after changing this register.

### 1.4.5 FBFCR (Flash Buffer Control Register)

This section explains the FBFCR.

In flash accelerator mode (RWT = "0b10"/RWT = "0b11" in FRWTR register), allowing operating FLASH Accelerator trace buffer function by this register will further improve the performance.

bit	7	6	5	4	3	2	1	0
Field	Reserved						BS	BE
Attribute							R	RW
Initial value							0	0

#### [bit7:2] Reserved bits

The read values are undefined. Ignored on write.

#### [bit1] BS: Buffer Status

Field	bit	Description
BS	1	Buffer Status 0: Trace buffer function is in stop or in initializing. 1: Trace buffer function operation is allowed.

#### [bit0] BE: Buffer Enable

Field	bit	Description
BE	0	Buffer Enable 0: Trace buffer function will be stopped. 1: Trace buffer function operation is allowed.

After the trace buffer function operation is allowed (after "1" is written to BE bit), trace buffer initialization will be started. After HCLK × 1025 cycles, the initialization will be completed and the trace buffer enters into operation. BS bit will be set to "1" at this time.

The prefetch buffer will still be functioning while initializing the trace buffer (BE = "1" and BS = "0"), allowing access to the flash memory. When changed to BS = "1" and the trace buffer is in operation, the trace buffer will automatically start tracing.

### 1.4.6 FICR (Flash Interrupt Control Register)

This section explains FICR.

This register is used to enable the interrupt of Flash memory except DualFlash area.

bit	7	6	5	4	3	2	1	0
Field	Reserved					ERRIE	HNGIE	RDYIE
Attribute						RW	RW	RW
Initial value						0	0	0

#### [bit7:3] Reserved bits

The read values are undefined. Ignored on write.

#### [bit2] ERRIE : Flash ECC Error Interrupt Enable

This bit enables ECC error correction interrupt. When ERRIF bit of FISR register is "1" and this bit is "1", an interrupt to CPU is generated.

Field	bit	Description
ERRIE	2	Flash ECC Error Interrupt Enable 0: ECC error correction interrupt is disabled. (Initial value) 1: ECC error correction interrupt is enabled.

#### [bit1] HNGIE : Flash HANG Interrupt Enable

This bit enables flash HANG interrupt. When HANGIF bit of FISR register is "1" and this bit is "1", an interrupt to CPU is generated.

Field	bit	Description
HNGIE	1	Flash HANG Interrupt Enable 0: Flash HANG interrupt is disabled. (Initial value) 1: Flash HANG interrupt is enabled.

#### [bit0] RDYIE : Flash RDY Interrupt Enable

This bit enables Flash RDY interrupt. When RDYIF bit of FISR register is "1" and this bit is "1", an interrupt to CPU is generated.

Field	bit	Description
RDYIE	0	Flash RDY Interrupt Enable 0: Flash RDY interrupt is disabled. (Initial value) 1: Flash RDY interrupt is enabled.

### 1.4.7 FISR (Flash Interrupt Status Register)

This section explains FISR.

This register indicates the interrupt state of Flash memory except DualFlash area.

bit	7	6	5	4	3	2	1	0
Field	Reserved					ERRIF	HNGIF	RDYIF
Attribute						R	R	R
Initial value						0	0	0

#### [bit7:3] Reserved bits

The read values are undefined. Ignored on write.

#### [bit2] ERRIF : Flash ECC Error Interrupt Flag

When the generation of ECC error correction of Flash read data is detected, this bit is set to "1". This bit is set at the rising edge of ERR signal. This bit is cleared by writing "1" to ERRC bit of FICLR register.

Field	bit	Description
ERRIF	2	Flash ECC Error Interrupt Flag 0: The generation of ECC error correction is not detected. 1: The generation of ECC error correction is detected.

#### [bit1] HNGIF : Flash HANG Interrupt Flag

When the Flash HANG state is detected, this bit is set to "1". This bit is set at the rising edge of HNG signal. This bit is cleared by writing "1" to HNGC bit of FICLR register.

Field	bit	Description
HNGIF	1	Flash HANG Interrupt Flag 0: Flash HANG state is not detected. 1: Flash HANG state is detected.

#### [bit0] RDYIF : Flash RDY Interrupt Flag

When Flash RDY state is detected, this bit is set to "1". This bit is set at the rising edge of RDY signal. This bit is cleared by writing "1" to RDYC bit of FICLR register.

Field	bit	Description
RDYIF	0	Flash RDY Interrupt Flag 0: Flash RDY state is not detected. 1: Flash RDY state is detected.



### 1.4.8 FICLR (Flash Interrupt Clear Register)

This section explains FICLR.

This register is used to clear the interrupt state of Flash memory except DualFlash area.

bit	7	6	5	4	3	2	1	0
Field	Reserved					ERRIC	HNGIC	RDYIC
Attribute						RW	RW	RW
Initial value						0	0	0

#### [bit7:3] Reserved bits

The read values are undefined. Ignored on write.

#### [bit2] ERRIC : Flash ECC Error Interrupt Clear

This bit clears the ERR interrupt flag. By writing "1" to this bit, ERRIF bit of FISR register is cleared to "0".

Field	bit	Description
ERRIC	2	Flash ECC Error Interrupt Clear At write 0: ECC error correction interrupt flag (ERRIF) is not changed. 1: ECC error correction interrupt flag (ERRIF) is cleared. At read "0" is read out.

#### [bit1] HNGIC : Flash HANG Interrupt Clear

This bit clears HNG interrupt flag. By writing "1" to this bit, HNGIF bit of FISR register is cleared to "0".

Field	bit	Description
HNGIC	1	Flash HANG Interrupt Clear At write 0: Flash HANG interrupt flag (HNGIF) is not changed. 1: Flash HANG interrupt flag (HNGIF) is cleared. At read "0" is read out.

#### [bit0] RDYIC : Flash RDY Interrupt Clear

This bit clears RDY interrupt flag. By writing "1" to this bit, RDYIF bit of FISR register is cleared to "0".

Field	bit	Description
RDYIC	0	Flash RDY Interrupt Clear At write 0: Flash RDY interrupt flag (RDYIF) is not changed. 1: Flash RDY interrupt flag (RDYIF) is cleared. At read "0" is read out.

### 1.4.9 DFCTRLR (Dual Flash mode Control Register)

This section explains DFCTRLR.

This register is used to control the dual flash mode.

Depending on the product, there are some restrictions on this mode. See Section "1.3.6 Dual flash mode" for details.

bit	31	16	15	2	1	0
Field	WKEY				RME	DFE
Attribute	RW				RW	RW
Initial value	0				0	0

#### [bit31:16] WKEY : Write Key

When "0xEACC" is written to the WKEY bit at the same time, writing to the DFE bit and the RME bit can be performed.

Field	bit	Description
WKEY	31:16	Write Key On write: 0xEACC : Writing to the RME bit and DFE bit is valid. Other than 0xEACC: Writing to the RME bit and DFE bit is invalid. On read: "0" is always read.

#### [bit15:2] Reserved bits

The read values are undefined. Ignored on write.

#### [bit1] RME : Re-Map Enable

This bit set the allocation of the DualFlash area in the dual flash mode

Field	bit	Description
RME	1	Re-Map Enable 0: Flash Macro #1 is assigned to DualFlash area. (Initial value) 1: Flash Macro #0 is assigned to DualFlash area.

#### [bit1] DFE : Dual Flash mode Enable

This bit set the dual flash mode.

Field	bit	Description
DFE	1	Dual Flash mode Enable 0: the dual flash mode is disabled. (Initial value) 1: the dual flash mode is enabled.

#### Notes:

- In order to write to the DFE bit and the RME bit, write WKEY="0xEACC" at the same time. When WKEY="0xEACC" is not written, the writing is invalid.
- Writing is invalid if writing the RME bit only. Write DFE="1" at the same time.
- Always perform writes to DFCTRLR register using word access (32-bit access).
- Writing to this register is invalid when the DualFlash mode is enabled (when DFE="1"). In order to perform writing again, issue a reset command.
- Do not perform writing to this register after the trace buffer function is enabled (after BE="1" is set for the FBFCR register).
- Do not change this register with instructions written in flash memory. Re-write programs other than those in the Flash area.
- Perform a dummy read to register, after changing this register.
- Writing is invalid in any mode other than user mode.

### 1.4.10 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	20	16	9	0
Field	Reserved		TTRMM	Reserved	TRMM
Attribute			R		R
Initial value			*		*

#### [bit31:21] Reserved bits

The read values are undefined. Ignored on write.

#### [bit20:16] TTRMM : Temperature CR Trimming Data Mirror Register

After reset is released, store the bit[4:0] in an address of "0x0040\_2002" (temperature trimming data) of the flash memory area into this register.

See "CHAPTER High-Speed CR Trimming" of the "FM4 Family Peripheral Manual" for details on the CR temperature trimming data.

Field	bit	Description
TTRMM	20:16	*: Reads out bit[4:0] of an address of "0x0040_2002".

#### [bit15:10] Reserved bits

The read values are undefined. Ignored on write.

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

After reset is released, store the bit[9:0] in an address of "0x0040\_2000" (frequency trimming data) of the flash memory area into this register.

See "CHAPTER High-Speed CR Trimming" of the "FM4 Family Peripheral Manual" for details on the CR Frequency trimming data.

Field	bit	Description
TRMM	9:0	*: Reads out bit[9:0] of an address of "0x0040_2000".

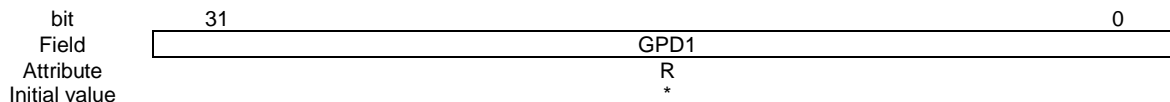
#### Notes:

- After the flash memory is lost, 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.
- When Re-Map function is enabled, this value does not change.

### 1.4.11 FGPDM1 (Flash General Purpose Data Mirror Register1)

This section explains the FGPDM1.

This is the mirror register of the general purpose data1.



#### [bit31:0] GPD1 : General Purpose Data1

After reset is released, store the bit[31:0] in an address of "0x0040\_4000" (general purpose data1) of the flash memory area into this register.

Field	bit	Description
GPD1	31:0	*: Reads out bit[31:0] of an address of "0x0040_4000".

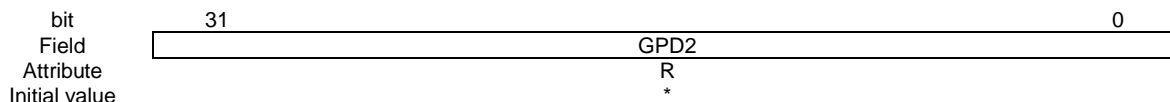
#### Notes:

- After the flash memory is lost, as this register is cleared when reset is issued in a chip, the stored general purpose data1 is lost. Therefore, before this register is cleared, save the general purpose data1 stored in the register on the RAM, etc.
- When Re-Map function is enabled, this value does not change.

### 1.4.12 FGPD2 (Flash General Purpose Data Mirror Register2)

This section explains the FGPD2.

This is the mirror register of the general purpose data2.



#### [bit31:0] GPD2 : General Purpose Data2

After reset is released, store the bit[31:0] in an address of "0x0040\_4004" (general purpose data2) of the flash memory area into this register.

Field	bit	Description
GPD2	31:0	*: Reads out bit[31:0] of an address of "0x0040_4004".

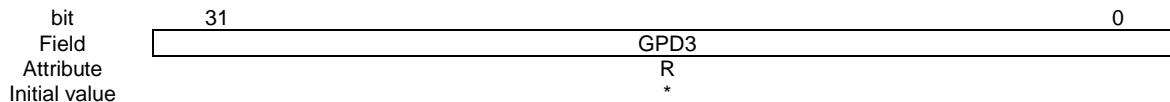
#### Notes:

- After the flash memory is lost, as this register is cleared when reset is issued in a chip, the stored general purpose data2 is lost. Therefore, before this register is cleared, save the general purpose data2 stored in the register on the RAM, etc.
- When Re-Map function is enabled, this value does not change.

### 1.4.13 FGPD3 (Flash General Purpose Data Mirror Register3)

This section explains the FGPD3.

This is the mirror register of the general purpose data3.



#### [bit31:0] GPD3 : General Purpose Data3

After reset is released, store the bit[31:0] in an address of "0x0040\_4008" (general purpose data3) of the flash memory area into this register.

Field	bit	Description
GPD3	31:0	*: Reads out bit[31:0] of an address of "0x0040_4008".

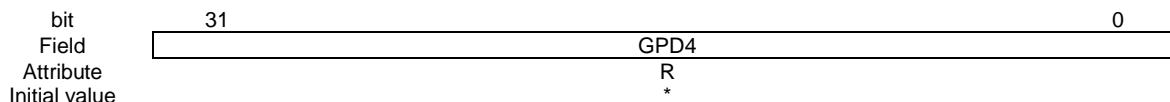
#### Notes:

- After the flash memory is lost, as this register is cleared when reset is issued in a chip, the stored general purpose data3 is lost. Therefore, before this register is cleared, save the general purpose data3 stored in the register on the RAM, etc.
- When Re-Map function is enabled, this value does not change.

#### 1.4.14 FGPDM4 (Flash General Purpose Data Mirror Register4)

This section explains the FGPDM4.

This is the mirror register of the general purpose data4.



##### [bit31:0] GPD4 : General Purpose Data4

After reset is released, store the bit[31:0] in an address of "0x0040\_400C" (general purpose data4) of the flash memory area into this register.

Field	bit	Description
GPD4	31:0	*: Reads out bit[31:0] of an address of "0x0040_400C".

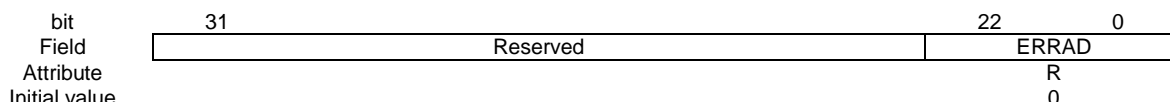
##### Notes:

- After the flash memory is lost, as this register is cleared when reset is issued in a chip, the stored general purpose data4 is lost. Therefore, before this register is cleared, save the general purpose data4 stored in the register on the RAM, etc.
- When Re-Map function is enabled, this value does not change.

#### 1.4.15 FERRAD (Flash ECC ERR Address Capture Register)

This section explains FERRAD.

This register saves the address when ECC error correction of read data of Flash memory except DualFlash area is generated.



##### [bit31:23] Reserved bits

The read values are undefined. Ignored on write.

##### [bit22:0] ERRAD : Flash ECC ERR Address Capture Register

This register saves the address when ECC error correction of read data of Flash memory except DualFlash area is generated.

Field	bit	Description
ERRAD	22:0	Saves the address when ECC error correction is generated.

##### Note:

- An address once stored is retained until ERR bit of FSTR register is set to "1" again. That is to say, without clearing FSTR:ERR bit, the address stored at first is stored irrespective of the continuous generation of ERR.

### 1.4.16 DFASZR (Dual Flash Access Size Register)

This section explains the DFASZR.

In the dual flash mode (DFCTRLR:DFE="1"), specifies the access size of DualFlash area.

bit	7	6	5	4	3	2	1	0
Field	Reserved						DASZ	
Attribute							RW	RW
Initial Value							1	0

#### [bit7:2] Reserved bits

The read values are undefined. Ignored on write.

#### [bit1:0] DASZ: Dual Flash Access Size

Specifies the access size of the flash memory.

Field	bit	Description
DASZ	1:0	Dual Flash Access Size 00: Setting prohibited 01: 16-bit read/write (CPU programming mode) 10: 32-bit read (CPU ROM mode: Initial value) 11: Setting prohibited

#### Notes:

- When DASZ is set to "0b01", 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 DualFlash area. Overwrite this register from a program in any other area except for the DualFlash area.
- Perform a dummy read to register, after changing this register.
- While dual flash mode is disabled (DFCTRLR : DFE="0"), writing to DFASZR is prohibited.



### 1.4.17 DFRWTR (Dual Flash Read Wait Register)

This section explains the DFRWTR.

In the dual flash mode (DFCTRLR:DFE="1"), this register is effective when ASZ="0b10" (32-bit read mode). It configures the access method for the DualFlash area.

bit	7	6	5	4	3	2	1	0
Field	Reserved						DRWT	
Attribute							RW	RW
Initial Value							1	1

#### [bit7:2] Reserved bits

The read values are undefined. Ignored on write.

#### [bit1:0] DRWT: Dual Flash Read Wait Cycle

Specifies the access method for flash memory.

Field	bit	Description
DRWT	1:0	Read Wait Cycle 00: 0 cycle wait mode (0 latency) This setting can be used when HCLK is 72 MHz or less. 01: Setting prohibited 10: 4 cycles wait mode This setting can be used when HCLK is 160MHz or less. This setting must be used when HCLK is over 72 MHz. 11: 6 cycles wait mode (Initial value) This setting must be used when HCLK is over 160 MHz

When HCLK is 72 MHz or less, 0 cycle wait mode (DRWT = "0b00") is suitable for CPU operation.

In DRWT="0b10" or DRWT="0b11", allowing operating the data buffer function. (See Section "1.3.7 Data buffer")

#### Notes:

- Do not set DRWT to "0b00"(0 cycle wait mode) if HCLK exceeds 72 MHz.  
While DRWT setting is "0b00", HCLK must not exceed 72 MHz.
- Do not set DRWT to "0b10" if HCLK exceeds 160 MHz.  
While DRWT setting is "0b10", HCLK must not exceed 160 MHz.
- Perform a dummy read to register, after changing this register.
- While dual flash mode is disabled (DFCTRLR : DFE="0"), writing to DFRWTR is prohibited.

### 1.4.18 DFSTR (Flash Status Register)

This section explains the DFSTR.

In the dual flash mode (DFCTRLR:DFE="1"), this is a status register of the DualFlash area.

bit	7	6	5	4	3	2	1	0
Field	Reserved					DFERR	DFHNG	DFRDY
Attribute						RW	R	R
Initial Value						0	0	X

#### [bit7:3] Reserved bits

The read values are undefined. Ignored on write.

#### [bit2] DFERR: Dual Flash ECC Error

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

Field	bit	Description
DFERR	2	Dual 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] DFHNG: Dual Flash Hang

Indicates whether the DualFlash area of flash memory is in the HANG state. Flash memory enters the HANG state if the timing is exceeded (See "[bit5] TLOV: Timing Limit Exceeded Flag Bit"). If this bit becomes "1", issue a reset command. (See Section "1.3.2.1 Command Sequence")

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

Field	Bit	Description
DFHNG	1	Dual Flash Hang 0: The DualFlash area of flash memory HANG state has not been detected. 1: The DualFlash area of flash memory HANG state has been detected.

#### [bit0] DFRDY: Dual Flash Rdy

Indicates whether a DualFlash area of flash memory write or erase operation using the automatic algorithm is in progress or finished. Also indicates whether the DualFlash mode switching is in progress or finished. While an operation is in progress, data cannot be written and the DualFlash area cannot be erased.

Field	Bit	Description
DFRDY	0	Dual 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 algorithm command is issued, ignore the value of this bit as read the first time after a command is issued.

#### Note:

- While dual flash mode is disabled (DFCTRLR : DFE="0"), the read value is invalid.

## 2. Flash Security



The flash security function protects contents of the MainFlash memory.  
This section describes the overview and operations of the flash security.

2.1 . Overview

2.2 . Operation Explanation

## 2.1 Overview

This section explains the overview of the flash security.

If the protection code of 0x0001 is written in the security code area of MainFlash memory, access to the MainFlash memory is restricted. Once the flash memory is protected, performing the flash erase operation only can unlock the function otherwise read/write access to the MainFlash 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 2-1 shows the address and the protection code of the security code.

**Table 2-1 Address of Security Code and Protection Code**

Address	Protection Code
0x0040_0000	0x0001

**Note:**

The above address of security code is the address while the DualFlash mode is disabled. When the DualFlash mode and the Re-map function are enabled, the address is changed. For the detail, please refer to Chapter “MainFlash Memory”.

## 2.2 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

1. Issue the flash erase command to the Flash Macro #1 of MainFlash memory.
2. Confirm if the flash erase operation for the Flash Macro #1 of MainFlash memory is completed.
3. Issue the flash erase command to the Flash Macro #0 of MainFlash memory where the security code is stored.
4. The security is released by all the reset factors or power-on after the execution of flash erase.

### Operation with Security Enabled

The operations with security enabled vary depending on each mode.

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

**Table 2-2 Flash Operation with Security Enabled**

Mode	Mode Pin MD[1:0]	Access to Flash			Access from JTAG Pins
		Flash 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 flash 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.*

*When the security is released, erase the data of Flash Macro #1 of MainFlash memory at first. The data of Flash Macro #0 of MainFlash memory cannot be erased before erasing the data of Flash Macro #1 of MainFlash memory.*

## 3. Serial Programming Connection



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

### 3.1 . Serial Programmer

## **3.1 Serial Programmer**

Cypress 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.

### **3.1.1 . Basic Configuration**

### **3.1.2 . Pins Used**

### 3.1.1 Basic Configuration

This section explains the basic configuration.

#### Basic Configuration of FLASH MCU Programmer (Clock Asynchronous Serial Write)

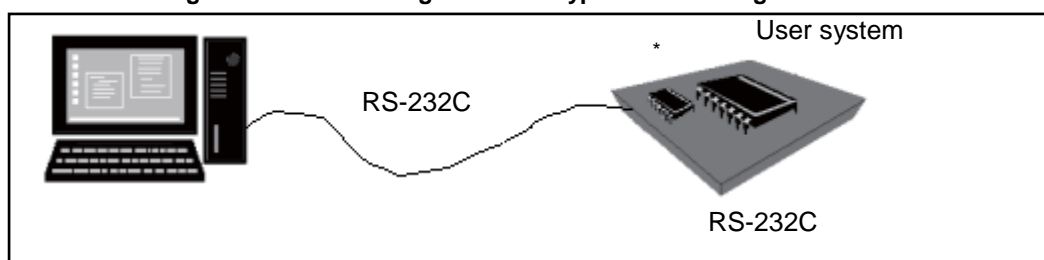
FLASH MCU Programmer writes data, through clock asynchronous serial communication, to built-in flash memory of a microcontroller installed in the user system when the PC and the user system are connected through RS-232C cable.

In these series, serial programming (UART communication mode) is possible by any clock, crystal oscillator or external clock or built-in High-speed CR oscillator.

If flash erase is executed to the flash memory that its flash security is enabled when built-in CR oscillator is used instead of external crystal oscillator, serial communication will be disconnected after the erase operation and then CR trimming data will be lost. When flash security is enabled, use external crystal oscillator.

Figure 3-1 shows the basic configuration of FLASH MCU Programmer, and Table 3-1 lists the system configuration.

**Figure 3-1 Basic Configuration of Cypress MCU Programmer**



**Table 3-1 System Configuration of FLASH MCU Programmer**

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

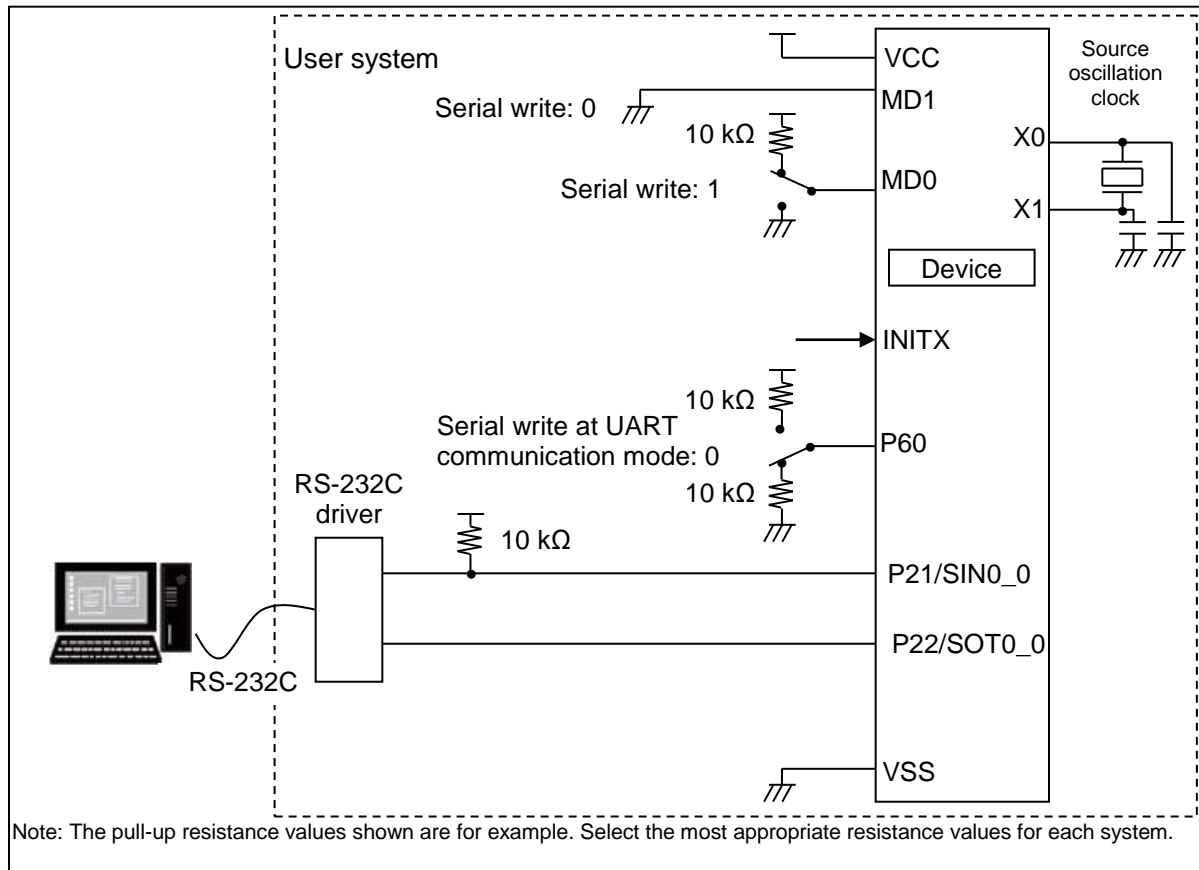
### Connection Example of RS-232C I/F

The following shows a connection example of RS-232C I/F.

When Crystal oscillator is used as the source oscillation clock

Figure 3-2 shows a connection example of FS-232C I/F when a crystal oscillator is used as a source oscillation clock. When crystal oscillator is used, the communication will start with a baud rate of 115200[bps]. Table 3-2 shows available frequencies and communication baud rates at start-up.

**Figure 3-2 Connection Example When Crystal Oscillator is Used**





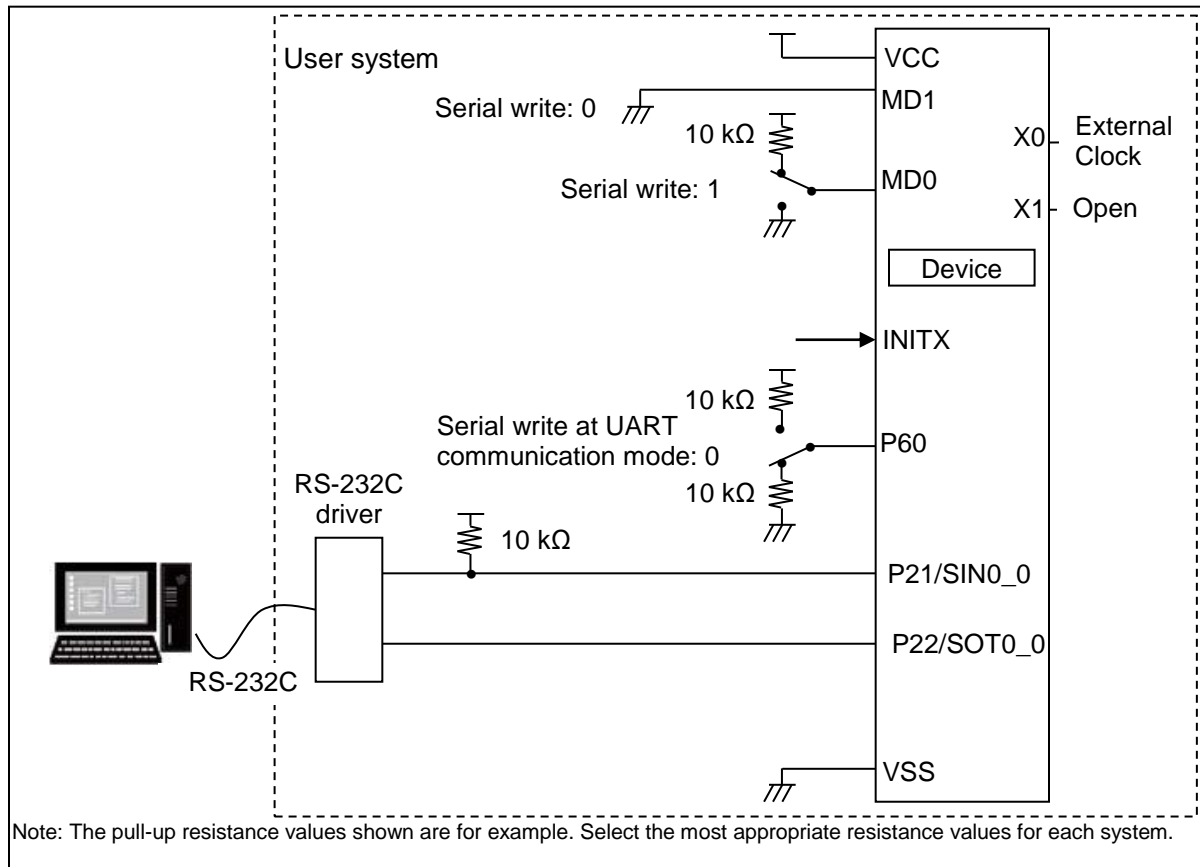
When external clock is used as the source oscillation clock

Figure 3-3 shows a connection example of FS-232C I/F when an external clock is used as a source oscillation clock. When external clock is used, the communication will start with a baud rate of 115200[bps]. Table 3-2 shows available frequencies and communication baud rates at start-up.

When external clock is used, there is a restriction below.

- If user system needs to use X1 pin as GPIO and cannot make the pin open, the clock oscillation may be unstable. When the X1 pin cannot be opened, please use built-in high-speed CR oscillator for reliable communication.

**Figure 3-3 Connection Example When External Clock is Used**



**Table 3-2 Available Oscillating Frequency and Communication Baud Rate at Start-Up for Clock Asynchronous Serial Communication**

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

When built-in high-speed CR oscillator is used as a source oscillation clock

Figure 3-4 shows a connection example of FLASH MCU Programmer when a built-in high-speed CR oscillator is used as a source oscillation clock.

When neither crystal oscillator nor external clock is connected to X0/X1 pins, the built-in high-speed CR oscillator is connected for communication.

When built-in high-speed CR oscillator is used, the communication baud rate will start with a baud rate of 9600[bps], and then it will be changed to 115200[bps].

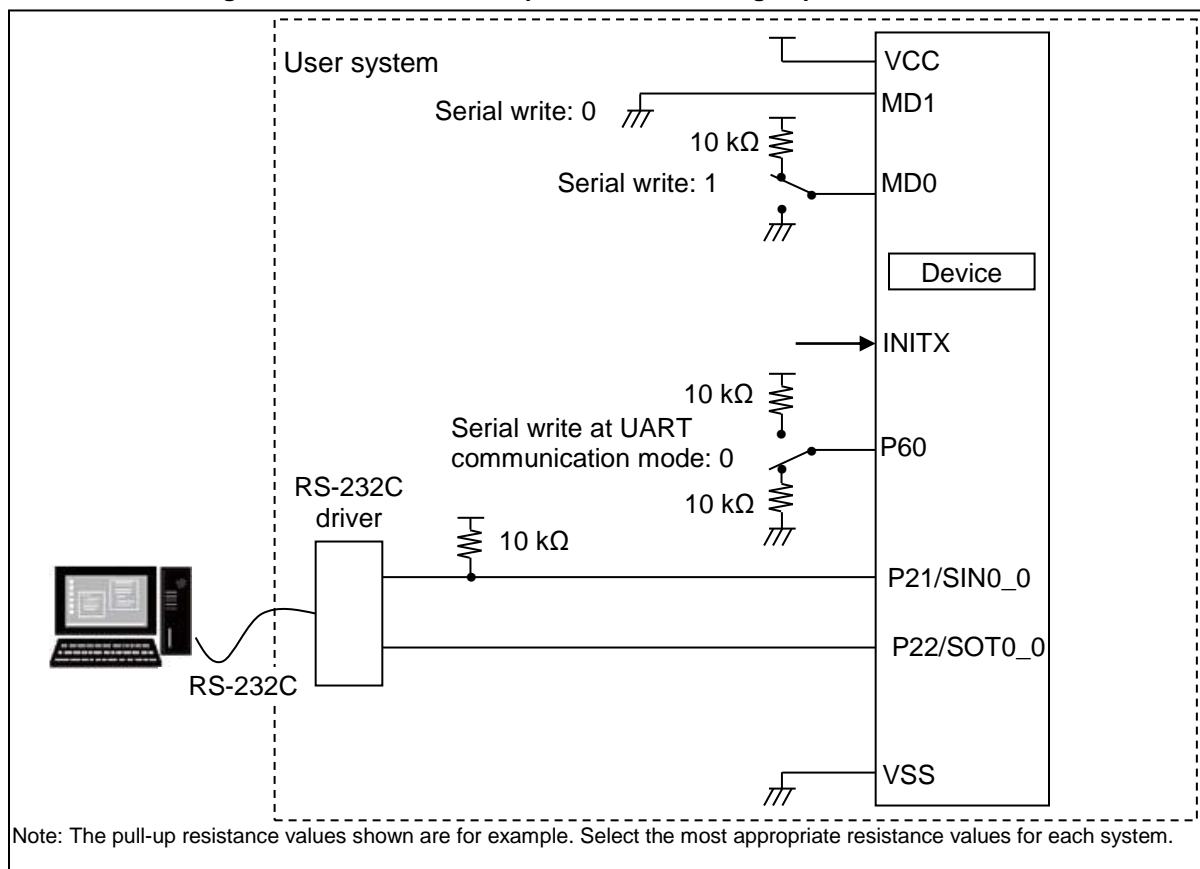
The following are the restrictions when built-in high-speed CR oscillator is used

Because the oscillation frequency of the built-in high-speed CR oscillator would fluctuate due to temperature and voltage change, the allowable baud rate error range might be exceeded.

For using the built-in high-speed CR oscillator, see "Built-in CR Oscillation Specifications" in "Data Sheet" of the product used.

If flash erase is executed to the flash memory that its flash security is enabled when built-in CR oscillator is used instead of external crystal oscillator, serial communication will be disconnected after the erase operation and then CR trimming data will be lost. When flash security is enabled, use external crystal oscillator.

**Figure 3-4 Connection Example When Built-in High-speed CR Oscillator is Used**

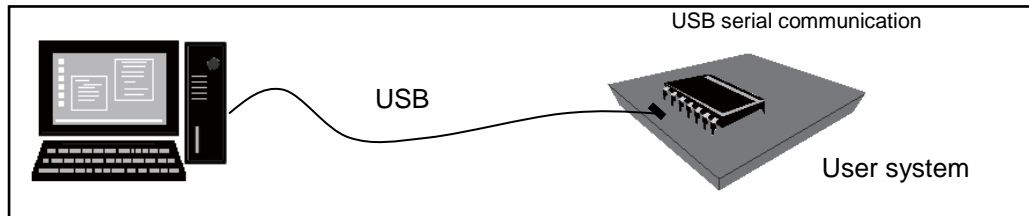


### Basic Configuration of FLASH USB DIRECT Programmer (USB Serial Write)

FLASH 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 3-5 shows the basic configuration of FLASH USB DIRECT Programmer, and Table 3-3 lists the system configuration.

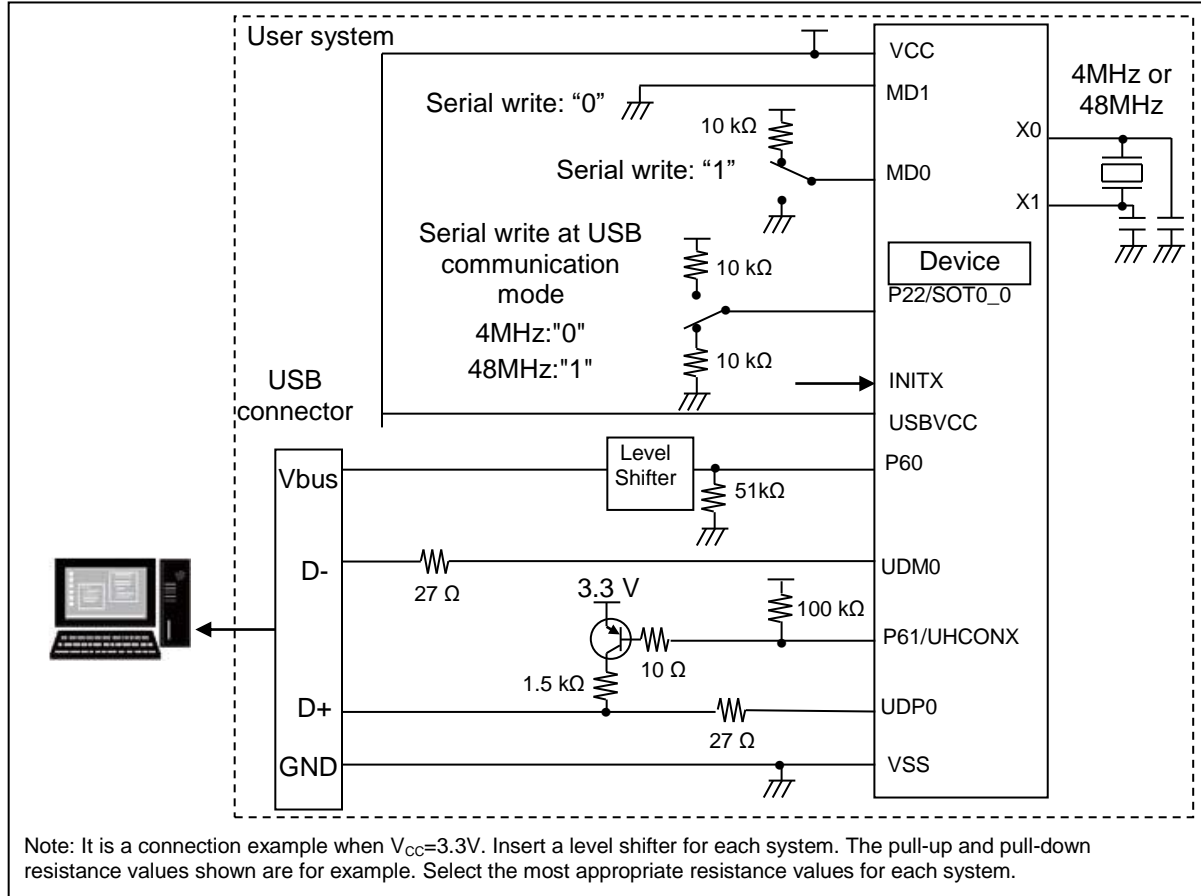
**Figure 3-5 Basic Configuration of FLASH USB DIRECT Programmer**



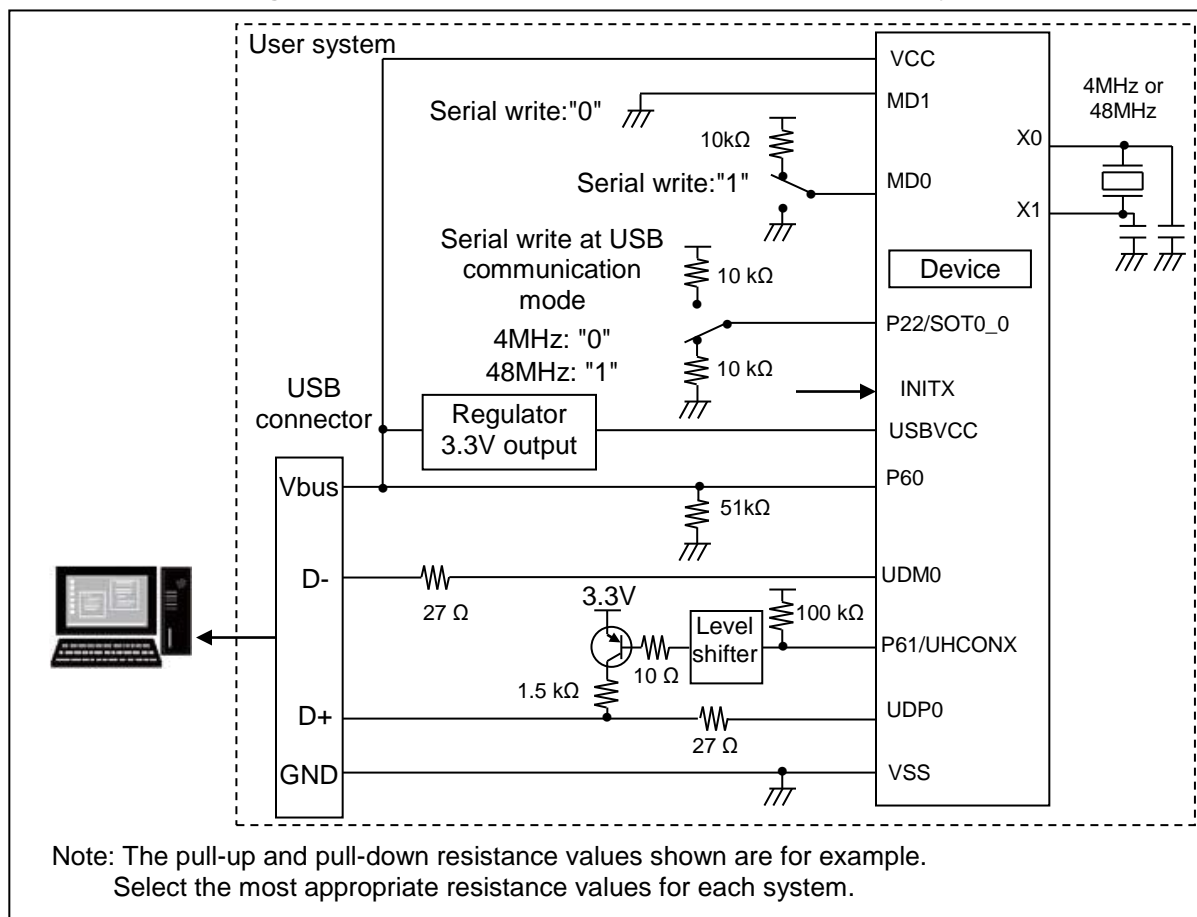
**Table 3-3 System Configuration of FLASH USB DIRECT Programmer**

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

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

**Figure 3-6 Connection Example of USB I/F (Own Power Supply is Used.)**


-----  
User system



### 3.1.2 Pins Used

This section explains the used pins.

**Table 3-4 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 <a href="#">Table 3-2.</a> )
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: 4 MHz P22=H: source oscillation frequency: 48 MHz
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	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 (D+) 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 (D+) when the communication mode is set to USB.
UDM0	USB I/O pin	This pin becomes an input/output pin of USB side (D-) 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	-
VSS	GND pin	-

**Note:**

- Note that initial states of un-used pins in the serial writer mode are the same as initial states of them in the user mode. Refer to "Pin Status in Each CPU State" section in "Data Sheet" of the product used and relevant chapters in the "FM4 Family Peripheral Manual".

# Revision History



## Document Revision History

Document Title: S6E2CC/C5/C4/C3/C2/C1 Series 32-bit Microcontroller FM4 Family Flash Programming Specifications			
Document Number: 002-04913			
Revision	Issue Date	Origin of Change	Description of Change
**	09/30/2014	AKIH	Initial release
*A	04/25/2016	AKIH	Migrated to Cypress format
*B	06/09/2017	YSAT	Adapted Cypress new logo
*C	10/24/2017	NOSU	Add a note about address notation in command sequences. Add an use case of external clock with FLASH MCU Programmer Add a note that CR trimming data will be lost when erase operation is executed to security enabled flash memory. Add a note for initial states of un-used pins in the serial writer mode.
*D	02/27/2019	HTER	Changed title and category