User's Manual

32

# RH850/E2x-FCC1

## Flash Memory

## User's Manual: Hardware Interface

Renesas microcontroller

All information contained in these materials, including products and product specifications, represents information on the product at the time of publication and is subject to change by Renesas Electronics Corp. without notice. Please review the latest information published by Renesas Electronics Corp. through various means, including the Renesas Electronics Corp. website (http://www.renesas.com).

Renesas Electronics
www.renesas.com

Rev.1.30 Dec, 2019

# Notice

1. Descriptions of circuits, software and other related information in this document are provided only to illustrate the operation of semiconductor products and application examples. You are fully responsible for the incorporation or any other use of the circuits, software, and information in the design of your product or system. Renesas Electronics disclaims any and all liability for any losses and damages incurred by you or third parties arising from the use of these circuits, software, or information.
2. Renesas Electronics hereby expressly disclaims any warranties against and liability for infringement or any other claims involving patents, copyrights, or other intellectual property rights of third parties, by or arising from the use of Renesas Electronics products or technical information described in this document, including but not limited to, the product data, drawings, charts, programs, algorithms, and application examples.
3. No license, express, implied or otherwise, is granted hereby under any patents, copyrights or other intellectual property rights of Renesas Electronics or others.
4. You shall not alter, modify, copy, or reverse engineer any Renesas Electronics product, whether in whole or in part. Renesas Electronics disclaims any and all liability for any losses or damages incurred by you or third parties arising from such alteration, modification, copying or reverse engineering.
5. Renesas Electronics products are classified according to the following two quality grades: “Standard” and “High Quality”. The intended applications for each Renesas Electronics product depends on the product’s quality grade, as indicated below.
   “Standard”: Computers; office equipment; communications equipment; test and measurement equipment; audio and visual equipment; home electronic appliances; machine tools; personal electronic equipment; industrial robots; etc.
   “High Quality”: Transportation equipment (automobiles, trains, ships, etc.); traffic control (traffic lights); large-scale communication equipment; key financial terminal systems; safety control equipment; etc.
   Unless expressly designated as a high reliability product or a product for harsh environments in a Renesas Electronics data sheet or other Renesas Electronics document, Renesas Electronics products are not intended or authorized for use in products or systems that may pose a direct threat to human life or bodily injury (artificial life support devices or systems; surgical implantations; etc.), or may cause serious property damage (space system; undersea repeaters; nuclear power control systems; aircraft control systems; key plant systems; military equipment; etc.). Renesas Electronics disclaims any and all liability for any damages or losses incurred by you or any third parties arising from the use of any Renesas Electronics product that is inconsistent with any Renesas Electronics data sheet, user’s manual or other Renesas Electronics document.
6. When using Renesas Electronics products, refer to the latest product information (data sheets, user’s manuals, application notes, “General Notes for Handling and Using Semiconductor Devices” in the reliability handbook, etc.), and ensure that usage conditions are within the ranges specified by Renesas Electronics with respect to maximum ratings, operating power supply voltage range, heat dissipation characteristics, installation, etc. Renesas Electronics disclaims any and all liability for any malfunctions, failure or accident arising out of the use of Renesas Electronics products outside of such specified ranges.
7. Although Renesas Electronics endeavors to improve the quality and reliability of Renesas Electronics products, semiconductor products have specific characteristics, such as the occurrence of failure at a certain rate and malfunctions under certain use conditions. Unless designated as a high reliability product or a product for harsh environments in a Renesas Electronics data sheet or other Renesas Electronics document, Renesas Electronics products are not subject to radiation resistance design. You are responsible for implementing safety measures to guard against the possibility of bodily injury, injury or damage caused by fire, and/or danger to the public in the event of a failure or malfunction of Renesas Electronics products, such as safety design for hardware and software, including but not limited to redundancy, fire control and malfunction prevention, appropriate treatment for aging degradation or any other appropriate measures. Because the evaluation of microcomputer software alone is very difficult and impractical, you are responsible for evaluating the safety of the final products or systems manufactured by you.
8. Please contact a Renesas Electronics sales office for details as to environmental matters such as the environmental compatibility of each Renesas Electronics product. You are responsible for carefully and sufficiently investigating applicable laws and regulations that regulate the inclusion or use of controlled substances, including without limitation, the EU RoHS Directive, and using Renesas Electronics products in compliance with all these applicable laws and regulations. Renesas Electronics disclaims any and all liability for damages or losses occurring as a result of your noncompliance with applicable laws and regulations.
9. Renesas Electronics products and technologies shall not be used for or incorporated into any products or systems whose manufacture, use, or sale is prohibited under any applicable domestic or foreign laws or regulations. You shall comply with any applicable export control laws and regulations promulgated and administered by the governments of any countries asserting jurisdiction over the parties or transactions.
10. It is the responsibility of the buyer or distributor of Renesas Electronics products, or any other party who distributes, disposes of, or otherwise sells or transfers the product to a third party, to notify such third party in advance of the contents and conditions set forth in this document.
11. This document shall not be reprinted, reproduced or duplicated in any form, in whole or in part, without prior written consent of Renesas Electronics.
12. Please contact a Renesas Electronics sales office if you have any questions regarding the information contained in this document or Renesas Electronics products.

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

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

(Rev.4.0-1 November 2017)

# Notes for CMOS devices

**(1) Voltage application waveform at input pin:**

Waveform distortion due to input noise or a reflected wave may cause malfunction. If the input of the CMOS device stays in the area between VIL (MAX) and VIH (MIN) due to noise, etc., the device may malfunction. Take care to prevent chattering noise from entering the device when the input level is fixed, and also in the transition period when the input level passes through the area between VIL (MAX) and VIH (MIN).

**(2) Handling of unused input pins:**

Unconnected CMOS device inputs can be cause of malfunction. If an input pin is unconnected, it is possible that an internal input level may be generated due to noise, etc., causing malfunction. CMOS devices behave differently than Bipolar or NMOS devices. Input levels of CMOS devices must be fixed high or low by using pull-up or pull-down circuitry. Each unused pin should be connected to power supply or GND via a resistor if there is a possibility that it will be an output pin. All handling related to unused pins must be judged separately for each device and according to related specifications governing the device.

**(3) Precaution against ESD:**

A strong electric field, when exposed to a MOS device, can cause destruction of the gate oxide and ultimately degrade the device operation. Steps must be taken to stop generation of static electricity as much as possible, and quickly dissipate it when it has occurred. Environmental control must be adequate. When it is dry, a humidifier should be used. It is recommended to avoid using insulators that easily build up static electricity. Semiconductor devices must be stored and transported in an anti-static container, static shielding bag or conductive material. All test and measurement tools including work benches and floors should be grounded. The operator should be grounded using a wrist strap. Semiconductor devices must not be touched with bare hands. Similar precautions need to be taken for PW boards with mounted semiconductor devices.

**(4) Status before initialization:**

Power-on does not necessarily define the initial status of a MOS device. Immediately after the power source is turned ON, devices with reset functions have not yet been initialized. Hence, power-on does not guarantee output pin levels, I/O settings or contents of registers. A device is not initialized until the reset signal is received. A reset operation must be executed immediately after power-on for devices with reset functions.

**(5) Power ON/OFF sequence:**

In the case of a device that uses different power supplies for the internal operation and external interface, as a rule, switch on the external power supply after switching on the internal power supply. When switching the power supply off, as a rule, switch off the external power supply and then the internal power supply. Use of the reverse power on/off sequences may result in the application of an overvoltage to the internal elements of the device, causing malfunction and degradation of internal elements due to the passage of an abnormal current. The correct power on/off sequence must be judged separately for each device and according to related specifications governing the device.

**(6) Input of signal during power off state:**

Do not input signals or an I/O pull-up power supply while the device is not powered. The current injection that results from input of such a signal or I/O pull-up power supply may cause malfunction and the abnormal current that passes in the device at this time may cause degradation of internal elements. Input of signals during the power off state must be judged separately for each device and according to related specifications governing the device.

# Table of Contents

# Section 1 Features

The features of the flash memory are described below. See *Section 41, Flash Memory*, in *the User's Manual: Hardware* for information on the capacity, block configuration, and addresses of the flash memory in a given product.

### Flash Memory Programming/Erasure

A dedicated sequencer for the flash memory (flash sequencer) executes programming and erasure via the peripheral-bus. The flash sequencer also supports the programming/processing suspension/resumption and BGO (background operation)[Note 1]/Dual Operation[Note 1].

**Note 1.** As for the condition to enable the BGO and Dual Operation, refer to the user's manual for this product.

### Security Functions

The flash memory incorporates hardware functions to prevent illicit tampering.

### Protection Functions

The flash memory incorporates hardware functions to prevent erroneous writing.

### Interrupts

The flash memory supports an interrupt to indicate completion of processing by the flash sequencer and an error interrupt to indicate erroneous operations.

### DMA

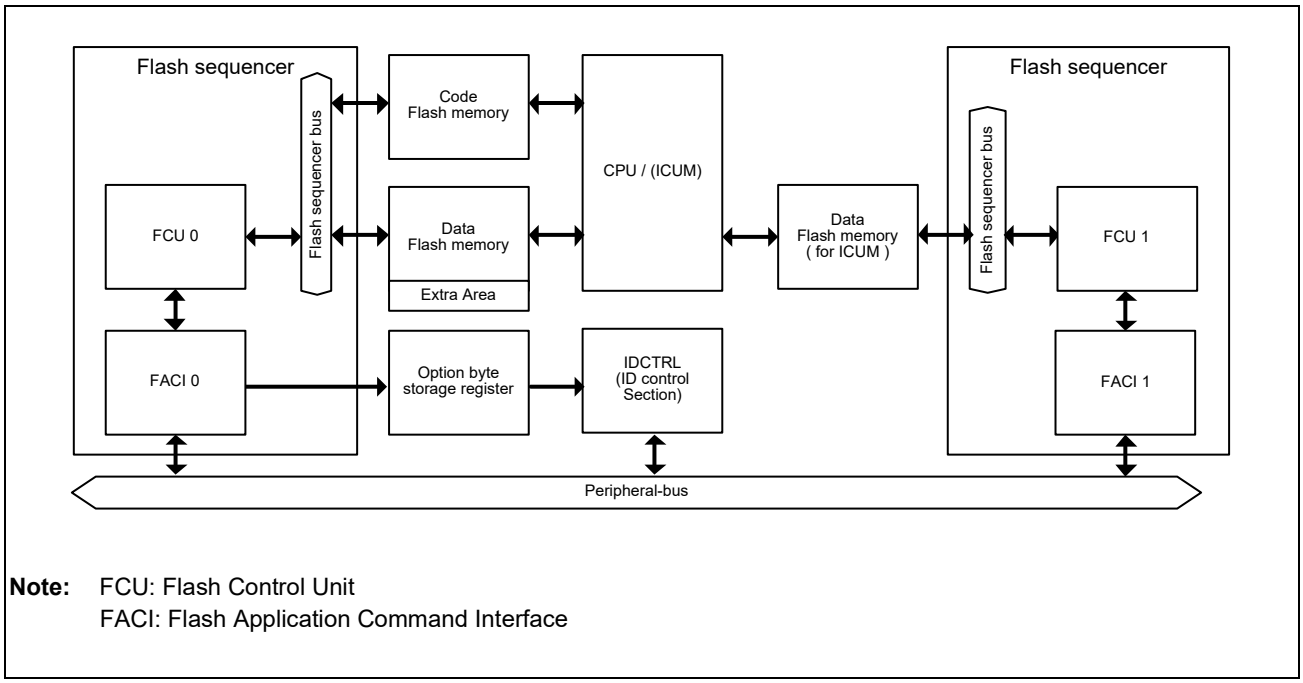
The data flash memory can be programmed using the DMA.

# Section 2 Module Configuration

Modules related to the flash memory are configured as shown in **Figure 2.1**. The flash sequencer is configured of Flash Control Unit (FCU) and Flash Application Command Interface (FACI). The FCU executes basic control of programming/erasure of the flash memory. The FACI receives FACI commands via the peripheral-bus and controls FCU operations accordingly.

The product having Data flash memory exclusively for ICUM has another one set of sequencer. This sequencer controls the Data flash memory exclusively for ICUM individually.

In the transfer operations in response to a reset, the FACI transfers the data (configuration settings, security settings) from flash memory to the option byte storage registers (FACI reset transfer). The ID control section performs each ID authentication. This section compares the ID in Flash memory (Security Setting Area) with the value input into in the registers for authentication in this section.




**Note:** FCU: Flash Control Unit
FACI: Flash Application Command Interface

Figure 2.1 Configuration of Flash Memory Related Modules

# Section 3 Address Map

**Table 3.1** gives information on all of these areas.

Table 3.1 Information on the Hardware Interface Area

| Area | Address | Capacity | Peripheral Group |
|---|---|---|---|
| Area containing the various registers of the hardware | See **Section 4, Registers**. | See **Section 4, Registers**. | See **Table 4.1 Register Base Address**. |
| FACI0 command-issuing area | FFA2 $0000_H$ | 4 bytes | 6 |
| FACI1 command-issuing area | FFA3 $0000_H$ | 4 bytes | 6 |
| Security Setting Area*1 | FF30 $0040_H$ | 448 bytes | 1 |
| Configuration Setting Area*1 | FF32 $0040_H$ | 192 bytes | 1 |
| Block Protection Area*1 | FF32 $2040_H$ | 192 bytes | 1 |

Note 1. These areas are generically called “extra area”.

Refer to *Section 41, Flash Memory*, in *the User’s Manual: Hardware* for information on the addresses of the flash memory, etc.

# Section 4 Registers

This section gives information on the registers. For registers that are not specifically mentioned, only reset them to their initial states.

For information on the option bytes, see *Section 41, Flash Memory*, in *the User's Manual: Hardware* of each product.

For information of the registers to control from ICUM, see *the RH850/E2x ICUMD User's Manual.*

## 4.1 Register Base Address

Register addresses are given as offsets from the base addresses.

Table 4.1 Register Base Address

| Base Address Name | Base Address | Bus Group |
|---|---|---|
| <FLASH_FA0_base> | FFA2 0000$_H$ | Peripheral Group 6 |
| <FLASH_FA1_base> | FFA3 0000$_H$ | Peripheral Group 6 |
| <FLASH_RG0_base> | FFA1 0000$_H$ | Peripheral Group 6 |
| <FLASH_RG1_base> | FFA1 8000$_H$ | Peripheral Group 6 |
| <IDCTRL_base> | FFA0 8000$_H$ | Peripheral Group 6 |
| <SYSCTRL_base> | FF70 0000$_H$ | Peripheral Group 6 |

## 4.2 Registers Related to Programming/Erasure of Flash Memory

**Table 4.2** shows the list of registers related to programming/erasure of flash memory.

Table 4.2 Registers Related to Programming/Erasure of Flash Memory

| Unit Name | Module Name | Register Name | Symbol | Address | Access Size |
|---|---|---|---|---|---|
| FACI0 | FLASH_RG0 | Flash Access Status 0 | FASTAT_0 | <FLASH_RG0_base> + $0010_H$ | 8 |
| FACI0 | FLASH_RG0 | Flash Access Error Interrupt Enable 0 | FAEINT_0 | <FLASH_RG0_base> + $0014_H$ | 8 |
| FACI0 | FLASH_RG0 | Code Flash Memory Area Select Register | FAREASELC | <FLASH_RG0_base> + $0020_H$ | 16 |
| FACI0 | FLASH_RG0 | Flash Command Start Address 0 | FSADDR_0 | <FLASH_RG0_base> + $0030_H$ | 32 |
| FACI0 | FLASH_RG0 | Flash Command End Address 0 | FEADDR_0 | <FLASH_RG0_base> + $0034_H$ | 32 |
| FACI0 | FLASH_RG0 | Flash Status_0 | FSTATR_0 | <FLASH_RG0_base> + $0080_H$ | 32 |
| FACI0 | FLASH_RG0 | Flash Programming/Erasure Mode Entry 0 | FENTRYR_0 | <FLASH_RG0_base> + $0084_H$ | 16 |
| FACI0 | FLASH_RG0 | Flash Sequencer Set-up Initialize 0 | FSUINITR_0 | <FLASH_RG0_base> + $008C_H$ | 16 |
| FACI0 | FLASH_RG0 | FACI Command 0 | FCMDR_0 | <FLASH_RG0_base> + $00A0_H$ | 16 |
| FACI0 | FLASH_RG0 | Flash Programming/Erasure Status 0 | FPESTAT_0 | <FLASH_RG0_base> + $00C0_H$ | 16 |
| FACI0 | FLASH_RG0 | Data Flash Blank Check Control 0 | FBCCNT_0 | <FLASH_RG0_base> + $00D0_H$ | 8 |
| FACI0 | FLASH_RG0 | Data Flash Blank Check Status 0 | FBCSTAT_0 | <FLASH_RG0_base> + $00D4_H$ | 8 |
| FACI0 | FLASH_RG0 | Programmed Area Start Address 0 | FPSADDR_0 | <FLASH_RG0_base> + $00D8_H$ | 32 |
| FACI0 | FLASH_RG0 | Flash Sequencer Process Switch 0 | FCPSR_0 | <FLASH_RG0_base> + $00E0_H$ | 16 |
| FACI0 | FLASH_RG0 | Flash ECC Encoder Monitor 0 | FECCEMON_0 | <FLASH_RG0_base> + $0100_H$ | 16 |
| FACI0 | FLASH_RG0 | Flash ECC Test Mode 0 | FECCTMD_0 | <FLASH_RG0_base> + $0104_H$ | 16 |
| FACI0 | FLASH_RG0 | Flash Dummy ECC 0 | FDMYECC_0 | <FLASH_RG0_base> + $0108_H$ | 16 |
| FACI1 | FLASH_RG1 | Flash Access Status 1 | FASTAT_1 | <FLASH_RG1_base> + $0010_H$ | 8 |
| FACI1 | FLASH_RG1 | Flash Access Error Interrupt Enable 1 | FAEINT_1 | <FLASH_RG1_base> + $0014_H$ | 8 |
| FACI1 | FLASH_RG1 | Flash Command Start Address 1 | FSADDR_1 | <FLASH_RG1_base> + $0030_H$ | 32 |
| FACI1 | FLASH_RG1 | Flash Command End Address 1 | FEADDR_1 | <FLASH_RG1_base> + $0034_H$ | 32 |
| FACI1 | FLASH_RG1 | Flash Status_1 | FSTATR_1 | <FLASH_RG1_base> + $0080_H$ | 32 |
| FACI1 | FLASH_RG1 | Flash Programming/Erasure Mode Entry 1 | FENTRYR_1 | <FLASH_RG1_base> + $0084_H$ | 16 |
| FACI1 | FLASH_RG1 | Flash Sequencer Set-up Initialize 1 | FSUINITR_1 | <FLASH_RG1_base> + $008C_H$ | 16 |
| FACI1 | FLASH_RG1 | FACI Command 1 | FCMDR_1 | <FLASH_RG1_base> + $00A0_H$ | 16 |
| FACI1 | FLASH_RG1 | Flash Programming/Erasure Status 1 | FPESTAT_1 | <FLASH_RG1_base> + $00C0_H$ | 16 |
| FACI1 | FLASH_RG1 | Data Flash Blank Check Control 1 | FBCCNT_1 | <FLASH_RG1_base> + $00D0_H$ | 8 |
| FACI1 | FLASH_RG1 | Data Flash Blank Check Status 1 | FBCSTAT_1 | <FLASH_RG1_base> + $00D4_H$ | 8 |
| FACI1 | FLASH_RG1 | Programmed Area Start Address 1 | FPSADDR_1 | <FLASH_RG1_base> + $00D8_H$ | 32 |
| FACI1 | FLASH_RG1 | Flash Sequencer Process Switch 1 | FCPSR_1 | <FLASH_RG1_base> + $00E0_H$ | 16 |
| FACI1 | FLASH_RG1 | Flash ECC Encoder Monitor 1 | FECCEMON_1 | <FLASH_RG1_base> + $0100_H$ | 16 |
| FACI1 | FLASH_RG1 | Flash ECC Test Mode 1 | FECCTMD_1 | <FLASH_RG1_base> + $0104_H$ | 16 |
| FACI1 | FLASH_RG1 | Flash Dummy ECC 1 | FDMYECC_1 | <FLASH_RG1_base> + $0108_H$ | 16 |
| IDCTRL | IDCTRL*2 | Serial Programmer ID Input n (n=0 to 7) | SPIDINn (n=0 to 7) | <IDCTRL_base> + $0000_H$ to $001C_H$ | 32 |
| IDCTRL | IDCTRL*2 | Data Flash ID Input n (n=0 to 7) | DFIDINn (n=0 to 7) | <IDCTRL_base> + $0020_H$ to $003C_H$ | 32 |
| IDCTRL | IDCTRL*2 | OCD ID Input n (n=0 to 7) | OCDIDINn (n=0 to 7) | <IDCTRL_base> + $0040_H$ to $005C_H$ | 32 |

Table 4.2 Registers Related to Programming/Erasure of Flash Memory

| Unit Name | Module Name | Register Name | Symbol | Address | Access Size |
|---|---|---|---|---|---|
| IDCTRL | IDCTRL*2 | C-TEST ID Input n (n=0 to 7) | CTESTIDINn (n=0 to 7) | <IDCTRL_base> + $0060_H$ to $007C_H$ | 32 |
| IDCTRL | IDCTRL*2 | Customer ID Input n (n=0 to 7) | CUSTIDINn (n=0 to 7)*2 | <IDCTRL_base> + $0080_H$ to $009C_H$ | 32 |
| IDCTRL | IDCTRL*2 | ID Authentication Status | IDST | <IDCTRL_base> + $01FC_H$ | 32 |
| IDCTRL | IDCTRL*2 | RHSIF ID Input n (n=0 to 7) | RHSIFIDINn (n=0 to 7) | <IDCTRL_base> + $0200_H$ to $021C_H$ | 32 |
| IDCTRL | IDCTRL*2 | ID Authentication Status 2 | IDST2 | <IDCTRL_base> + $03FC_H$ | 32 |
| SYSCTRL | SYSCTRL | FHVE15 Control Register | FHVE15 | <SYSCTRL_base> + $3804_H$ | 32 |
| SYSCTRL | SYSCTRL | FHVE3 Control Register | FHVE3 | <SYSCTRL_base> + $3800_H$ | 32 |
| FACI0 | FLASH_FA0 | FACI0 command-issuing area | FA | <FLASH_FA0_base>+ $0000_H$ | 32,16*1, 8*1 |
| FACI1 | FLASH_FA1 | FACI1 command-issuing area | FA | <FLASH_FA1_base>+ $0000_H$ | 32,16*1, 8*1 |

Note 1. By 16-bit access, only an address of lower 2 bytes is accessible.
By 8-bit access, only an address of lower 1 bytes is accessible.

Note 2. IDCTRL (Flash protection module) is protected by PBG61 channel number 8 or 9.
The register protected in PBG61 channel number 9 is RHSIFIDINn and IDST2.
Other registers are protected in PBG61 channel number 8.
For details, see *Section 38, Functional Safety*, in *the User's Manual: Hardware*.

Table 4.3 Register Reset Conditions

| | Reset Source | | | | | |
|---|---|---|---|---|---|---|
| Unit Name | Power Up Reset | System Reset 1 | System Reset 2 | Application Reset | Module Reset | JTAG Reset |
| FACI0 | ✓ | ✓ | ✓ | — | — | — |
| FACI1 | ✓ | ✓ | ✓ | — | — | — |
| IDCTRL | ✓ | ✓ | ✓ | — | — | — |
| FHVE15 | ✓ | ✓ | ✓ | ✓ | — | — |
| FHVE3 | ✓ | ✓ | ✓ | ✓ | — | — |

## 4.3 FASTAT_n — Flash Access Status Register (n = 0,1)

FASTAT_n indicates access error status for code/data flash. If either of CFAE/CMDLK/DFAE bits in FASTAT_n is set to 1, flash sequencer enters the command lock state (see **Section 8.2, Error Protection**). To cancel the command lock state, After setting the CFAE bit and DFAE bit in the FASTAT_n register to 0, and then issue a status clearing or forced stop command to each FACI.

**Access:** This register can be read / written in 8-bit units.

**Address:** <FLASH_RGn_base> + $0010_H$ (n = 0,1)

**Value after reset:** $00_H$

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | CFAE | — | — | CMDLK | DFAE | — | — | ECRCT |
| Value after reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W | R/W*1 | R | R | R | R/W*1 | R | R | R |

Note 1. Only 0 can be written to clear flag after 1 is read.

Table 4.4 FASTAT_n Register Contents (1/2)

| Bit Position | Bit Name | Function |
|---|---|---|
| 7 | CFAE | Code Flash Access Error<br>Indicates whether or not code flash access error has been generated. If this bit becomes 1, ILGLERR bit in FSTATR_n is set to 1 and flash sequencer enters the command lock state.<br>0: No code flash access error has occurred.<br>1: Code flash access error has occurred.<br>[Setting Condition]<br>FACI command*1 has been issued to wrong address*2 in code flash programming/erasure mode.<br>[Clearing Condition]<br>(1) 0 is written after reading 1 from this bit.<br>(2) Status Clearing or Forced Stop command processing is started. |
| 6, 5 | — | Reserved<br>These bits are always read as “0”. Write value should always be “0”. |
| 4 | CMDLK | Command Lock<br>Indicates whether flash sequencer is in the command lock state.<br>0: Flash sequencer is not in the command lock state.<br>1: Flash sequencer is in the command lock state.<br>[Setting Condition]<br>FACI detects error and enters the command lock state.<br>[Clearing Condition]<br>Status Clearing or Forced Stop command processing is started. |

Table 4.4 FASTAT_n Register Contents (2/2)

| Bit Position | Bit Name | Function |
|---|---|---|
| 3 | DFAE | Data Flash Access Error<br>Indicates whether or not data flash access error has been generated. If this bit becomes 1, ILGLERR bit in FSTATR_n is set to 1 and flash sequencer enters the command lock state.<br>0: No data flash access error has occurred.<br>1: Data flash access error has occurred.<br>[Setting Conditions]<br>FACI command*1 has been issued to wrong address*2 in data flash programming/erasure mode.<br>[Clearing Condition]<br>(1) 0 is written after reading 1 from this bit.<br>(2) Status Clearing or Forced Stop command processing is started. |
| 2, 1 | — | Reserved<br>These bits are always read as “0”. Write value should always be “0”. |
| 0 | ECRCT | Error Correction<br>Indicates that a 1-bit error has been corrected when the flash sequencer reads the flash memory (Configuration Setting Area, Block Protection Area, Security Setting Area or P/E parameter table)<br>0: 1-bit error has not been corrected.<br>1: 1-bit error has been corrected.<br>[Clearing conditions]<br>Status Clearing or Forced Stop command processing is started. |

Note 1. FSADDR_n register is not used in Status Clearing, Forced Stop, Programming/erasure suspension and Programming/erasure resumption. FSADDR_n value does not influence on these command operations, and CFAE or DFAE is not set for these commands even if FSADDR_n is wrong address in code flash programming/erasure mode or data flash programming/erasure mode.

Note 2. Capacities of the code flash memory and data flash memory vary from product to product. See the flash memory section in the user's manual of the applicable product.

## 4.4 FAEINT_n — Flash Access Error Interrupt Enable Register (n = 0,1)

FAEINT_n enables or disables output of flash access error (FLERR) interrupt.

In this product, a flash access error interrupt is connected to ECM and becomes the error factor of the ECM.

But it is notified to ICUM as an interrupt. For details, see *the RH850/E2x ICUMD User's Manual*.

**Access:** This register can be read / written in 8-bit units.

**Address:** <FLASH_RGn_base> + $0014_H$ (n = 0,1)

**Value after reset:** $99_H$

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | CFAEIE | — | — | CMDLKIE | DFAEIE | — | — | ECRCTIE |
| Value after reset | 1 | 0 | 0 | 1 | 1 | 0 | 0 | 1 |
| R/W | R/W | R | R | R/W | R/W | R | R | R/W |

Table 4.5 FAEINT_n Register Contents

| Bit Position | Bit Name | Function |
|---|---|---|
| 7 | CFAEIE | Code Flash Access Error Interrupt Enable<br>Enables or disables the FLERR interrupt request when code flash access error occurs and CFAE bit in FASTAT_n becomes 1.<br>0: Does not generate the FLERR interrupt request when FASTAT_n.CFAE = 1.<br>1: Generates the FLERR interrupt request when FASTAT_n.CFAE = 1. |
| 6, 5 | — | Reserved<br>These bits are always read as "0". Write value should always be "0". |
| 4 | CMDLKIE | Command Lock Interrupt Enable<br>Enables or disables the FLERR interrupt request when flash sequencer enters the command lock state and CMDLK bit in FASTAT_n becomes 1.<br>0: Does not generate the FLERR interrupt request when FASTAT_n.CMDLK = 1.<br>1: Generates the FLERR interrupt request when FASTAT_n.CMDLK = 1. |
| 3 | DFAEIE | Data Flash Access Error Interrupt Enable<br>Enables or disables the FLERR interrupt request when data flash access error occurs and DFAE bit in FASTAT_n becomes 1.<br>0: Does not generate the FLERR interrupt request when FASTAT_n.DFAE = 1.<br>1: Generates the FLERR interrupt request when FASTAT_n.DFAE = 1. |
| 2, 1 | — | Reserved<br>These bits are always read as "0". Write value should always be "0". |
| 0 | ECRCTIE | Error Correction Interrupt Enable<br>Enables or disables the FLERR interrupt request when a 1-bit error has been corrected and the ECRCT bit in FASTAT_n has been set to 1 on the flash memory read (Configuration Setting Area, Block Protection Area, Security Setting Area or P/E parameter table).<br>0: Does not generate the FLERR interrupt request when FASTAT_n.ECRCT = 1.<br>1: Generates the FLERR interrupt request when FASTAT_n.ECRCT = 1. |

## 4.5 FAREASELC — Code Flash Memory Area Select Register

This register is used to specify the user boot area as the target area for FACI command processing.

The FAREASELC value is initialized when the SUINIT bit in the FSUINITR_0 register is set to 1. It is also initialized by a reset.

**Access:** This register can be read / written in 16-bit units.

**Address:** <FLASH_RGn_base> + $0020_H$ (n = 0)

**Value after reset:** $0000\ 0000_H$

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | KEY | | | | | | | | — | — | — | — | — | — | CFAS | — |
| Value after reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W | R/W*1 | R/W*1 | R/W*1 | R/W*1 | R/W*1 | R/W*1 | R/W*1 | R/W*1 | R | R | R | R | R | R | R/W *2, *3, *4 | R |

Note 1. Written data is not stored in this bit. The value read is always $00_H$.

Note 2. Writing to this bit is enabled only when $3B_H$ is written to the KEY bits.

Note 3. This bit can be written in serial programming mode, and break status of on chip debugger mode.

Note 4. This bit can be written when FRDY bit in FSTATR_0 register is “1”. Writing to this bit in FRDY = “0” is ignored.

Table 4.6 FAREASELC Register Contents

| Bit Position | Bit Name | Function |
|---|---|---|
| 15 to 8 | KEY | Key Code<br>These bits enable or disable CFAS bit modification |
| 7 to 2 | — | Reserved<br>These bits are always read as “0”. Write value should always be “0”. |
| 1 | CFAS | User boot area Selection<br>0: User boot area is not selected.<br>1: User boot area is selected. |
| 0 | — | Reserved<br>These bits are always read as “0”. Write value should always be “0”. |

**CAUTION**

- In the case of CFAS = 0: When FACI command has been issued to user boot area, it becomes the Code flash access error.
- In the case of CFAS = 1: When FACI command has been issued to user area, it becomes the Code flash access error.

## 4.6 FSADDR_n — Flash Command Start Address Register (n = 0,1)

FSADDR_n specifies the start address of the target area for command processing when an FACI command (programming, Multi programming, DMA programming, Block erasure, Area erasure, blank checking, Setting of Configuration Settings, Setting of Block Protection Settings or Setting of Security Settings) is issued.

FSADDR_n value is initialized when SUINIT bit in FSUINITR_n is set to 1. It is also initialized by a reset.

**Access:** This register can be read / written in 32-bit units.

**Address:** <FLASH_RGn_base> + $0030_H$ (n = 0,1)

**Value after reset:** $0000\ 0000_H$

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | FSADDR[31:16] | | | | | | | | | | | | | | | |
| Value after reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W | R/W*1 | R/W*1 | R/W*1 | R/W*1 | R/W*1 | R/W*1 | R/W*1 | R/W*1 | R/W*1 | R/W*1 | R/W*1 | R/W*1 | R/W*1 | R/W*1 | R/W*1 | R/W*1 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | FSADDR[15:0] | | | | | | | | | | | | | | | |
| Value after reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W | R/W*1 | R/W*1 | R/W*1 | R/W*1 | R/W*1 | R/W*1 | R/W*1 | R/W*1 | R/W*1 | R/W*1 | R/W*1 | R/W*1 | R/W*1 | R/W*1 | R | R |

Note 1. This bit can be written when the FRDY bit in the FSTATR_n register is 1.Writing to this bit while the FRDY bits is 0 is ignored.

Table 4.7 FSADDR_n Register Contents (1/2)

| Bit Position | Bit Name | Function |
|---|---|---|
| 31 to 0 | FSADDR[31:0] | Start Address of FACI Command Processing<br>These bits specify the start address of the FACI command processing.<br>Bits 31 to 28 are ignored in the FACI command processing for the code flash memory.<br>Bits 31 to 21 are ignored in the FACI command processing for the data flash memory.<br>Bits 27 to 0 are used in the inconsistency check between FSADDR and FEADDR in area erasure.*1<br>Lower address bits for smaller address than boundary below are also ignored.<br>Bits 27 to 4 are used to generate the address parity for code flash programming. |

Table 4.7 FSADDR_n Register Contents (2/2)

| Bit Position | Bit Name | Function | |
|---|---|---|---|
| | | Command | Address Boundary |
| | | Programming (user area): | 512 bytes |
| | | Programming (user boot area): | 512 bytes |
| | | Programming (data area): | 4 bytes |
| | | Multi Programming (data area): 8-byte write: | 8 bytes |
| | | Multi Programming (data area): 16-byte write: | 16 bytes |
| | | Multi Programming (data area): 32-byte write: | 32 bytes |
| | | DMA programming: | 4 bytes |
| | | Block erasure (user area): | 16k or 64k bytes |
| | | Block erasure (user boot area): | 64k bytes |
| | | Block erasure (data area): | 64 bytes |
| | | Area erasure (data area): | 64 bytes |
| | | Blank checking: | 4 bytes |
| | | Setting of Configuration Settings: 4-byte setting | 4 bytes |
| | | Setting of Configuration Settings: 32-byte setting | 32 bytes |
| | | Setting of Block Protection Settings: 4-byte setting | 4 bytes |
| | | Setting of Block Protection Settings: 32-byte setting | 32 bytes |
| | | Setting of Security Settings: 4-byte setting | 4 bytes |
| | | Setting of Security Settings: 32-byte setting | 32 bytes |

Note 1. In order to get correct inconsistency check results at all time, bits 27 to 21 of FSADDR and FEADDR should have same value.

## 4.7 FEADDR_n — Flash Command End Address Register (n = 0,1)

FEADDR_n specifies the end address in the target area in Area erasure or blank checking command processing.

When Area erasure command is used, address specified in FSADDR_n should be equal to or smaller than address in FEADDR_n. If setting of FSADDR_n and FEADDR_n are inconsistent, FACI detects error and flash sequencer enters command lock state.

When blank check addressing mode is set to incremental mode (i.e. FBCCNT_n.BCDIR = 0), address specified in FSADDR_n should be equal to or smaller than address in FEADDR_n. Conversely, address in FSADDR_n should be equal to or larger than address in FEADDR_n when blank check addressing mode is set to decremental mode (i.e. FBCCNT_n.BCDIR = 1). If setting of each BCDIR_n, FSADDR_n, and FEADDR_n are inconsistent, FACI detects error and flash sequencer enters the command lock state. (See **Section 8.2, Error Protection**.)

In addition, target area should be within one Data Flash[Note 1]. If Area Erasure or blank checking command is processed crossing over different Data Flash boundary, FACI detects error and flash sequencer enters command lock state.

FEADDR_n value is initialized when SUINIT bit in each FSUINITR_n is set to 1. It is also initialized by a reset.

**Note 1.** The 96 KBytes Data Flash bank consists of two 48 KBytes Data Flash (EEP) called EEP0-0 and EEP0-1, or EEP1-0 and EEP1-1. 32 KBytes Data Flash bank for ICUM consists of one 32 Kbytes Data Flash called EEP2. These EEPs are treated as different Data Flash.

**Access:** This register can be read / written in 32-bit units.

**Address:** <FLASH_RGn_base> + $0034_H$ (n = 0,1)

**Value after reset:** $0000\ 0000_H$

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | FEADDR[31:16] | | | | | | | | | | | | | | | |
| Value after reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W | R/W*1 | R/W*1 | R/W*1 | R/W*1 | R/W*1 | R/W*1 | R/W*1 | R/W*1 | R/W*1 | R/W*1 | R/W*1 | R/W*1 | R/W*1 | R/W*1 | R/W*1 | R/W*1 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | FEADDR[15:0] | | | | | | | | | | | | | | | |
| Value after reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W | R/W*1 | R/W*1 | R/W*1 | R/W*1 | R/W*1 | R/W*1 | R/W*1 | R/W*1 | R/W*1 | R/W*1 | R/W*1 | R/W*1 | R/W*1 | R/W*1 | R | R |

Note 1. This bit can be written when the FRDY bit in the each FSTATR_n register is 1.Writing to this bit while the FRDY bits is 0 is ignored.

Table 4.8 FEADDR_n Register Contents

| Bit Position | Bit Name | Function |
|---|---|---|
| 31 to 0 | FEADDR[31:0] | End Address of FACI Command Target Area<br>Specifies end address of target area in the Area Erasure or blank checking command.<br>In case of Area Erasure command,<br>Bits 27 to 0 are used in the inconsistency check between FSADDR and FEADDR.*1<br>Bits 31 to 21 and 5 to 0 are ignored in command processing.<br>In case of blank checking command,<br>Bits 31 to 21, 1 and 0 are ignored in command processing. |

Note 1. In order to get correct inconsistency check results at all time, bits 27 to 21 of FSADDR and FEADDR should have same value.

## 4.8 FSTATR_n — Flash Status Register (n = 0,1)

FSTATR_n indicates flash sequencer status.

**Access:** This register can be read in 8-, 16-, or 32-bit units.

**Address:** <FLASH_RGn_base> + $0080_H$ (n = 0,1)

**Value after reset:** $0000\ 8000_H$

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | — | — | — | — | — | — | SECDTCT*1 | SECCRCT*1 | ILGCOMERR | FESETERR | SECERR | OTERR | — | EBFULL | BPLDTCT*1 | BPLCRCT*1 |
| Value after reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W | R | R | R | R | R | R | R | R | R | R | R | R | R | R | R | R |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | FRDY | ILGLERR | ERSERR | PRGERR | SUSRDY | DBFULL | ERSSPD | PRGSPD | — | FHVEERR | CFGDTCT*1 | CFGCRCT*1 | TBLDTCT | TBLCRCT | — | — |
| Value after reset | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W | R | R | R | R | R | R | R | R | R | R | R | R | R | R | R | R |

Note 1. Only FSTATR_0 has these bits. It is reserved bit in FSTATR_1 and read data is always “0”.

Table 4.9 FSTATR_n Register Contents (1/5)

| Bit Position | Bit Name | Function |
|---|---|---|
| 31 to 26 | — | Reserved<br>These bits are always read as “0”. |
| 25 | SECDTCT | Security setting area ECC 2-Bit Error Detection Monitoring Bit<br>Indicates that 2-bit error has been detected when flash sequencer is reading Security Setting Area. Flash sequencer reads Security Setting Area in Setting of Security Settings command processing. When this bit is “1”, flash sequencer is in command lock state.<br>0: No error has been detected.<br>1: An error has been detected.<br>[Clearing condition]<br>Status clearing or forced stop command processing is started. |
| 24 | SECCRCT | Security setting area ECC 1-Bit Error Correction Monitoring Bit<br>Indicates that 1-bit error has been corrected when flash sequencer is reading Security Setting Area. Flash sequencer reads Security Setting Area in Setting of Security Settings command processing. When this bit is “1”, flash sequencer continues command processing, and does not enter command lock state.<br>0: No error has been corrected.<br>1: An error has been corrected.<br>[Clearing condition]<br>Status clearing or forced stop command processing is started. |
| 23 | ILGCOMERR | Illegal Command Error<br>Refer to **Table 8.1**. When this bit is “1”, flash sequencer enters command lock state.<br>[Setting conditions]<br>An error has been detected.<br>[Clearing condition]<br>Status clearing or forced stop command processing is started. |

Table 4.9 FSTATR_n Register Contents (2/5)

| Bit Position | Bit Name | Function |
|---|---|---|
| 22 | FESETERR | FENTRY Setting Error<br>Refer to **Table 8.1**. When this bit is “1”, flash sequencer enters command lock state.<br>[Setting conditions]<br>An error has been detected.<br>[Clearing condition]<br>Status clearing or forced stop command processing is started. |
| 21 | SECERR | Security Error<br>Refer to **Table 8.1**. When this bit is “1”, flash sequencer enters command lock state.<br>[Setting conditions]<br>An error has been detected.<br>[Clearing condition]<br>Status clearing or forced stop command processing is started. |
| 20 | OTERR | Other Error<br>Refer to **Table 8.1**. When this bit is “1”, flash sequencer enters command lock state.<br>[Setting conditions]<br>An error has been detected.<br>[Clearing condition]<br>Status clearing or forced stop command processing is started. |
| 19 | — | Reserved<br>These bits are always read as “0”. |
| 18 | EBFULL | FDMYECC Buffer Full<br>Indicates the FDMYECC buffer status when issuing the programming command. Flash sequencer incorporates a buffer for FDMYECC bit (ECC buffer).<br>It is possible to use FDMYECC_n register as the ECC buffer by setting the ECCDISE bit in the FECCTMD_n register to 1. When FDMYECC is written to while EBFULL bit is 1, flash sequencer inserts a wait in the peripheral-bus.<br>0: The ECC buffer is not full.<br>1: The ECC buffer is full.<br>[Setting condition]<br>The ECC buffer becomes full while issuing the programming command.<br>[Clearing condition]<br>The ECC buffer becomes not full. |
| 17 | BPLDTCT | Block Protection Area ECC 2-Bit Error Detection Monitoring Bit<br>Indicates that 2-bit error has been detected when flash sequencer is reading Block Protection Area. Flash sequencer reads Block Protection Area in Programming for the code flash memory, Block Erasure for the code flash memory, or Setting of Block Protect Settings command processing.<br>When this bit is “1”, flash sequencer is in command lock state.<br>0: No error has been detected.<br>1: An error has been detected.<br>[Clearing condition]<br>Status clearing or forced stop command processing is started. |
| 16 | BPLCRCT | Block Protection Area ECC 1-Bit Error Correction Monitoring Bit<br>Indicates that 1-bit error has been corrected when flash sequencer is reading Block Protection Area. Flash sequencer reads Block Protection Area in Programming for the code flash memory, Block Erasure for the code flash memory or Setting of Block Protect Settings command processing.<br>When this bit is “1”, flash sequencer continues command processing, and does not enter command lock state.<br>0: No error has been corrected.<br>1: An error has been corrected.<br>[Clearing condition]<br>Status clearing or forced stop command processing is started. |

Table 4.9 FSTATR_n Register Contents (3/5)

| Bit Position | Bit Name | Function |
|---|---|---|
| 15 | FRDY | Flash Ready<br>Indicates the processing state in flash sequencer.*1<br>0: Processing of the command (programming, DMA programming, Multi programming, Block erasure, Area erasure, programming/erasure suspension, programming/erasure resumption, forced stop, blank checking, Setting of Configuration Settings, Setting of Block Protection Settings, Setting of Security Settings) is in progress.<br>1: None of the above is in progress.<br>[Setting Conditions]<br>Flash sequencer completes processing.<br>Flash sequencer suspends processing by a programming/erasure suspension command.<br>Flash sequencer terminates processing by a forced stop command.<br>[Clearing Conditions]<br>When the flash sequencer accepts the FACI command<br>For a programming, Multi programming, DMA programming, config programming, Setting of Configuration Settings, Setting of Block Protection Settings, or Setting of Security Settings command, after the first write access to the FACI command-issuing area.<br>For other commands, after the last write access to the FACI command-issuing area. |
| 14 | ILGLERR | Illegal Error<br>Indicates that flash sequencer has detected an illegal command or illegal flash memory access. When this bit is 1, flash sequencer is in the command lock state.<br>0: Flash sequencer has not detected any illegal command or illegal flash memory access.<br>1: Flash sequencer has detected an illegal command or illegal flash memory access<br>[Setting conditions] (see **Section 8.2, Error Protection**)<br>Flash sequencer has detected an illegal command.<br>Flash sequencer has detected an illegal flash memory access.<br>FENTRYR_n setting is illegal.<br>[Clearing condition]<br>Status clearing or forced stop command processing is started |
| 13 | ERSERR | Erasure Error<br>Indicates result of code or data flash erasure by flash sequencer. When this bit is 1, flash sequencer is in the command lock state.<br>0: Erasure processing has been completed successfully<br>1: An error has occurred during erasure<br>[Setting conditions]<br>An error has occurred during erasure.<br>[Clearing condition]<br>Status clearing or forced stop command processing is started. |
| 12 | PRGERR | Programming Error<br>Indicates the result of code or data flash programming by flash sequencer.<br>When this bit is 1, flash sequencer is in the command lock state.<br>0: Programming has been completed successfully<br>1: An error has occurred during programming<br>[Setting conditions]<br>An error has occurred during programming.<br>[Clearing condition]<br>Status clearing or forced stop command processing is started. |

Table 4.9 FSTATR_n Register Contents (4/5)

| Bit Position | Bit Name | Function |
|---|---|---|
| 11 | SUSRDY | Suspend Ready<br>Indicates whether flash sequencer is ready to accept the programming/erasure suspension command.<br>0: Flash sequencer cannot accept a programming/erasure suspension command.<br>1: Flash sequencer can accept a programming/erasure suspension command.<br>[Setting condition]<br>After initiating programming or erasure, Flash sequencer entered a state where it is ready to accept a programming/erasure suspension command.<br>[Clearing conditions]<br>Flash sequencer has accepted a programming/erasure suspension or forced stop command. (after the write access to the FACI command-issuing area is completed)<br>Flash sequencer has entered the command lock state during programming or erasure.<br>Programming / Multi Programming / Block erasure / Area erasure command processing is completed. |
| 10 | DBFULL | Data Buffer Full<br>Indicates the data buffer status when issuing the programming command.<br>Flash sequencer incorporates a buffer for write data (data buffer). When issuing the flash memory write data to the FACI command-issuing area while the data buffer is full, Flash sequencer inserts a wait in the peripheral-bus.<br>0: The data buffer is not full.<br>1: The data buffer is full.<br>[Setting condition]<br>The data buffer becomes full while issuing the programming command.<br>[Clearing condition]<br>The data buffer becomes not full. |
| 9 | ERSSPD | Erasure-Suspended Status<br>Indicates that flash sequencer has entered the erasure command suspension process or erasure-suspended status.<br>0: Flash sequencer is in status other than the below mentioned.<br>1: Flash sequencer is in erasure suspension process or erasure-suspended status.<br>[Setting condition]<br>Flash sequencer has initiated a programming/erasure suspension command during block erasure or Area erasure command processing.<br>[Clearing conditions]<br>Flash sequencer has accepted a programming/erasure resumption command. (after the write access to the FACI command-issuing area is completed)<br>Forced stop command processing is started. |
| 8 | PRGSPD | Programming Suspension Status<br>Indicates that flash sequencer has entered the programming command suspension process or programming suspension status.<br>0: Flash sequencer is in status other than the below mentioned.<br>1: Flash sequencer is in programming suspension process or programming suspension status.<br>[Setting condition]<br>Flash sequencer has initiated a programming/erasure suspension command during programming or Multi programming command processing.<br>[Clearing condition]<br>Flash sequencer has accepted a programming/erasure resumption command. (after the write access to the FACI command-issuing area is completed)<br>Forced stop command processing is started. |
| 7 | — | Reserved<br>These bits are always read as “0”. |

Table 4.9 FSTATR_n Register Contents (5/5)

| Bit Position | Bit Name | Function |
|---|---|---|
| 6 | FHVEERR | FHVE Setting Error<br>Indicates the violation of the flash memory programming/erasure protection in the FHVE3 register. When FHVEERR bit is 1, the flash sequencer is in the command lock state.<br>0: No error has occurred.<br>1: An error has occurred.<br>[Clearing condition]<br>Forced stop command processing is started. |
| 5 | CFGDTCT | Configuration Setting Area ECC 2-Bit Error Detection Monitoring Bit<br>Indicates that 2-bit error has been detected when flash sequencer is reading Configuration Setting Area. Flash sequencer reads Configuration Setting Area in Setting of Configuration Settings command processing When this bit is “1”, flash sequencer is in command lock state.<br>0: No error has been detected.<br>1: An error has been detected.<br>[Clearing condition]<br>Status clearing or forced stop command processing is started. |
| 4 | CFGCRCT | Configuration Setting Area ECC 1-Bit Error Correction Monitoring Bit<br>Indicates that 1-bit error has been corrected when flash sequencer is reading Configuration Setting Area. Flash sequencer reads Configuration Setting Area in Setting of Configuration Settings command processing. When this bit is “1”, flash sequencer continues command processing, and does not enter command lock state.<br>0: No error has been corrected.<br>1: An error has been corrected.<br>[Clearing condition]<br>Status clearing or forced stop command processing is started. |
| 3 | TBLDTCT | 2-Bit Error Detection Monitor (P/E Parameter Table*2)<br>Indicates that a 2-bit error has been detected on reading the P/E parameter table.<br>Flash sequencer reads the P/E parameter table in programming, Multi programming, DMA programming, Block erasure, Area erasure, blank checking, Setting of Configuration Settings, Setting of Block Protection Settings, Setting of Security Settings command processing. When this bit is 1, the flash sequencer is in the command lock state.<br>0: No error has been detected.<br>1: An error has been detected.<br>[Clearing condition]<br>Status clearing or forced stop command processing is started. |
| 2 | TBLCRCT | 1-Bit Error Correction Monitor (P/E Parameter Table*2)<br>Indicates that a 1-bit error has been corrected on reading the P/E parameter table.<br>Flash sequencer reads the P/E parameter table in programming, Multi programming, DMA programming, Block erasure, Area erasure, blank checking, Setting of Configuration Settings, Setting of Block Protection Settings, Setting of Security Settings command processing. When this bit is 1, the flash sequencer does not enter command lock state.<br>0: No error has been corrected.<br>1: An error has been corrected.<br>[Clearing condition]<br>Status clearing or forced stop command processing is started. |
| 1 to 0 | — | Reserved<br>These bits are always read as “0”. |

Note 1. The flash memory is not readable while FRDY is “0” without using the BGO function.

Note 2. The P/E parameter table is the data in the flash memory which cannot be written by the user.

## 4.9 FENTRYR_n — Flash Programming/Erasure Mode Entry Register (n = 0,1)

FENTRYR_n specifies programming/erasure mode for code flash or data flash. To specify programming/erasure mode for code flash or data flash so that flash sequencer can accept FACI commands, set either of FENTRYD or FENTRYC bit to 1.

Note that if this register is set to a value other than $0000_H$, $0001_H$ and $0080_H$, ILGLERR bit in the FSTATR_n register will be set and flash sequencer will enter the command lock state.

FENTRY_n value is initialized when SUINIT bit in each FSUINITR_n is set to 1. It is also initialized by a reset.

**Access:** This register can be read / written in 16-bit units.

**Address:** <FLASH_RGn_base> + $0084_H$ (n = 0,1)

**Value after reset:** $0000_H$

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | KEY | | | | | | | | FENTRYD | — | — | — | — | — | — | FENTRYC |
| Value after reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W | R/W*1 | R/W*1 | R/W*1 | R/W*1 | R/W*1 | R/W*1 | R/W*1 | R/W*1 | R/W *2,*3,*4 | R | R | R | R | R | R | R/W *2,*3,*4 |

Note 1. Written data is not stored in this bit. The value read is always $00_H$.

Note 2. Writing to this bit is enabled only when $AA_H$ is written to the KEY bits.

Note 3. This bit can be written when FRDY bit in each FSTATR_n register is “1”. Writing to this bit in FRDY = “0” is ignored.

Note 4. The value written to the FENTRYD/C bit may not be reflected immediately.
If ICUM is disabled: Dummy read is required for reading written value.
If ICUM is enabled: The operation from ICUMD is required for reading written value.
For details, see *the RH850/E2x ICUMD User's Manual*.

Table 4.10 FENTRYR_n Register Contents (1/2)

| Bit Position | Bit Name | Function |
|---|---|---|
| 15 to 8 | KEY | Key Code<br>These bits enable or disable FENTRYD and FENTRYC bits modification. |
| 7 | FENTRYD | Data Flash Programming/Erasure Mode Entry<br>This bit specifies programming/erasure mode for data flash.<br>0: Data flash is in read mode<br>1: Data flash is in programming/erasure mode<br>[Setting condition]<br>1 is written to FENTRYD while write enabling conditions are satisfied and FENTRYR_n is $0000_H$.<br>[Clearing conditions]<br>A value other than $AA_H$ is written to KEY in FENTRYR_n while FRDY bit is 1.<br>0 is written to FENTRYD while the write enabling conditions are satisfied.<br>FENTRYR_n is written to while FENTRYR_n is not $0000_H$, and the write enabling conditions are satisfied. |
| 6 to 1 | — | Reserved<br>These bits are always read as “0”. Write value should always be “0”. |

Table 4.10 FENTRYR_n Register Contents (2/2)

| Bit Position | Bit Name | Function |
|---|---|---|
| 0 | FENTRYC | Code Flash Programming/Erasure Mode Entry<br>This bit specifies programming/erasure mode for code flash.<br>0: Code flash is in read mode<br>1: Code flash is in programming/erasure mode<br>[Setting condition]<br>1 is written to FENTRYC while write enabling conditions are satisfied and FENTRYR_n is $0000_H$.<br>[Clearing conditions]<br>A value other than $AA_H$ is written to KEY in FENTRYR_n while FRDY bit is 1.<br>0 is written to FENTRYC while the write enabling conditions are satisfied.<br>FENTRYR_n is written to while FENTRYR_n is not $0000_H$ and the write enabling conditions are satisfied. |

## 4.10 FSUINITR_n — Flash Sequencer Set-up Initialize Register (n = 0,1)

FSUINITR_n register is used for initialization of flash sequencer set-up.

**Access:** This register can be read / written in 16-bit units.

**Address:** <FLASH_RGn_base> + $008C_H$ (n = 0,1)

**Value after reset:** $0000_H$

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | KEY | | | | | | | | — | — | — | — | — | — | — | SUINIT |
| Value after reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W | R/W*1 | R/W*1 | R/W*1 | R/W*1 | R/W*1 | R/W*1 | R/W*1 | R/W*1 | R | R | R | R | R | R | R | R/W*2,*3 |

Note 1. Written data is not stored in this bit. The value read is always $00_H$.

Note 2. Writing to this bit is enabled only when $2D_H$ is written to the KEY bits.

Note 3. This bit can be written when FRDY bit in each FSTATR_n register is “1”. Writing to this bit in FRDY = “0” is ignored.

Table 4.11 FSUINITR_n Register Contents

| Bit Position | Bit Name | Function |
|---|---|---|
| 15 to 8 | KEY | Key Code<br>These bits enable or disable SUINIT bit modification. |
| 7 to 1 | — | Reserved<br>These bits are always read as “0”. Write value should always be “0”. |
| 0 | SUINIT | Set-up Initialization<br>Initializes following flash sequencer set-up registers.<br>FEADDR_n<br>FCPSR_n<br>FSADDR_n<br>FENTRYR_n<br>FBCCNT_n<br>FAREASELC (only by FSUINITR_0)<br>0: The above flash sequencer set-up registers keep its’ value.<br>1: The above flash sequencer set-up registers are initialized. |

## 4.11 FCMDR_n — FACI Command Register (n = 0,1)

FCMDR_n stores commands that Flash sequencer has accepted.

**Access:** This register can be read in 16-bit units.

**Address:** <FLASH_RGn_base> + $00A0_H$ (n = 0,1)

**Value after reset:** $FFFF_H$

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | CMDR | | | | | | | | PCMDR | | | | | | | |
| Value after reset | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| R/W | R | R | R | R | R | R | R | R | R | R | R | R | R | R | R | R |

Table 4.12 FCMDR_n Register Contents

| Bit Position | Bit Name | Function |
|---|---|---|
| 15 to 8 | CMDR | Command<br>These bits store the latest command accepted by each Flash sequencer. |
| 7 to 0 | PCMDR | Previous Command<br>These bits store previous command accepted by each Flash sequencer. |

Table 4.13 States of FCMDR after Acceptance of the Various Commands

| FACI Command | CMDR | PCMDR |
|---|---|---|
| Programming | $E8_H$ | Previous Command |
| Multi Programming | $ED_H$ | Previous Command |
| DMA programming | $EA_H$ | Previous Command |
| Block erasure | $D0_H$ | $20_H$ |
| Area erasure | $D0_H$ | $23_H$ |
| Programming/erasure suspension | $B0_H$ | Previous Command |
| Programming/erasure resumption | $D0_H$ | Previous Command |
| Status clearing | $50_H$ | Previous Command |
| Forced stop | $B3_H$ | Previous Command |
| Blank checking | $D0_H$ | $71_H$ |
| Setting of Configuration Settings | $40_H$ | Previous Command |
| Setting of Block Protection Settings | $80_H$ | Previous Command |
| Setting of Security Settings | $60_H$ | Previous Command |

## 4.12 FPESTAT_n — Flash Programming/Erasure Status Register (n = 0,1)

FPESTAT_n is used to indicate the result of programming or erasure of the flash memory.

**Access:** This register can be read in 16-bit units.

**Address:** <FLASH_RGn_base> + $00C0_H$ (n = 0,1)

**Value after reset:** $0000_H$

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | — | — | — | — | — | — | — | — | PEERRST | | | | | | | |
| Value after reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W | R | R | R | R | R | R | R | R | R | R | R | R | R | R | R | R |

Table 4.14 FPESTAT_n Register Contents

| Bit Position | Bit Name | Function |
|---|---|---|
| 15 to 8 | — | Reserved<br>These bits are always read as “0”. |
| 7 to 0 | PEERRST | Programming/Erasure Error Status<br>Indicates the source of error that occurs during programming/erasure for code flash or data flash. This bit value is only valid if ERSERR or PRGERR bit value in each FSTATR_n register is 1, while FRDY bit in each FSTATR_n register is 1.<br>When ERSERR and PRGERR are 0, the PEERRST bit retains the value to indicate the source of error that previously occurred.<br>$00_H$: No error<br>$02_H$: A programming error caused<br>$12_H$: An erasure error caused<br>Other than above: Reserved |

## 4.13 FBCCNT_n — Data Flash Blank Check Control Register (n = 0,1)

FBCCNT_n specifies addressing mode in blank checking command processing. FBCCNT_n value is initialized when SUINIT bit in each FSUINITR_n is set to 1. It is also initialized by a reset.

**Access:** This register can be read / written in 8-bit units.

**Address:** <FLASH_RGn_base> + $00D0_H$ (n = 0,1)

**Value after reset:** $00_H$

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | — | — | — | — | — | — | — | BCDIR |
| Value after reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W | R | R | R | R | R | R | R | R/W |

Table 4.15 FBCCNT_n Register Contents

| Bit Position | Bit Name | Function |
|---|---|---|
| 7 to 1 | — | Reserved<br>These bits are always read as “0”. Write value should always be “0”. |
| 0 | BCDIR | Blank Check Direction<br>Specifies addressing mode in blank checking operation.<br>0: Blank checking is executed from smaller address to larger address. (Incremental mode)<br>1: Blank checking is executed from larger address to smaller address. (Decremental mode) |

## 4.14 FBCSTAT_n — Data Flash Blank Check Status Register (n = 0,1)

FBCSTAT_n stores check results by executing the blank checking command.

**Access:** This register can be read in 8-bit units.

**Address:** <FLASH_RGn_base> + $00D4_H$ (n = 0,1)

**Value after reset:** $00_H$

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | — | — | — | — | — | — | — | BCST |
| Value after reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W | R | R | R | R | R | R | R | R |

Table 4.16 FBCSTAT_n Register Contents

| Bit Position | Bit Name | Function |
|---|---|---|
| 7 to 1 | — | Reserved<br>These bits are always read as “0”. |
| 0 | BCST | Blank Check Status<br>Indicates the result of the blank checking command.<br>0: The target area is not written (no writing after erasing. blank)<br>1: The target area is filled with 0s and/or 1s. |

## 4.15 FPSADDR_n — Programmed Area Start Address Register (n = 0,1)

FPSADDR_n indicates address of the first programmed data which is found in blank checking command execution.

**Access:** This register can be read in 32-bit units.

**Address:** <FLASH_RGn_base> + $00D8_H$ (n = 0,1)

**Value after reset:** $0000\ 0000_H$

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | — | — | — | — | — | — | — | — | — | — | — | PSADR[20:16] | | | | |
| Value after reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W | R | R | R | R | R | R | R | R | R | R | R | R | R | R | R | R |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | PSADR[15:0] | | | | | | | | | | | | | | | |
| Value after reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W | R | R | R | R | R | R | R | R | R | R | R | R | R | R | R | R |

Table 4.17 FPSADDR_n Register Contents

| Bit Position | Bit Name | Function |
|---|---|---|
| 31 to 21 | — | Reserved<br>These bits are always read as 0. |
| 20 to 0 | PSADR | Programmed Area Start Address<br>Indicates address of the first programmed data which is found in blank checking command execution. These bits stores address offset from the top address in the data flash memory.<br>This register value is only valid if BCST bit value in FBCSTAT_n register is 1, while FRDY bit in FSTATR_n register is 1.<br>When BCST bit is 0, the PSDRA bit holds the address that previously checked. |

## 4.16 FCPSR_n — Flash Sequencer Process Switch Register (n = 0,1)

FCPSR_n selects a function to make the FCU suspend erasure. FCPSR_n value is initialized when SUINIT bit in each FSUINITR_n is set to 1. It is also initialized by a reset.

**Access:** This register can be read / written in 16-bit units.

**Address:** <FLASH_RGn_base> + $00E0_H$ (n = 0,1)

**Value after reset:** $0000_H$

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | ESUSPMD |
| Value after reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W | R | R | R | R | R | R | R | R | R | R | R | R | R | R | R | R/W |

Table 4.18 FCPSR_n Register Contents

| Bit Position | Bit Name | Function |
|---|---|---|
| 15 to 1 | — | Reserved<br>These bits are always read as “0”. Write value should always be “0”. |
| 0 | ESUSPMD | Erasure-Suspended Mode<br>Selects erasure-suspended mode to be entered when the programming/erasure suspension command is issued while flash sequencer is erasing flash memory.<br>(see **Section 6.3.13, Programming/Erasure Suspension Command**)<br>ESUSPMD bit should be set before issuing the Block erasure or Area erasure command.<br>0: Suspension-priority mode<br>1: Erasure-priority mode |

## 4.17 FECCEMON_n — Flash ECC Encoder Monitor Register (n = 0,1)

FECCEMON_n monitors the outputs from the address parity generator and ECC encoder.

**Access:** This register can be read in 16-bit units.

**Address:** <FLASH_RGn_base> + $0100_H$ (n = 0,1)

**Value after reset:** $FFFF_H$

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | — | — | — | — | — | — | FAPARM | FECCM[8:0] | | | | | | | | |
| Value after reset | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| R/W | R | R | R | R | R | R | R | R | R | R | R | R | R | R | R | R |

Table 4.19 FECCEMON_n Register Contents

| Bit Position | Bit Name | Function |
|---|---|---|
| 15 to 10 | — | Reserved<br>These bits are always read as “0”. |
| 9 | FAPARM | Address Parity Monitor<br>Indicates the output from the address parity generator.<br>In code flash programming/erasure mode<br>This bit indicates the output from the address parity generator.<br>In data flash programming/erasure mode<br>This bit is fixed to 1. |
| 8 to 0 | FECCM[8:0] | ECC Monitor<br>Indicates the ECC encoder output.<br>In code flash programming/erasure mode<br>The FECCM[8] to FECCM[0] bits indicate the ECC encoder output for the code flash memory.<br>In data flash programming/erasure mode<br>The FECCM[8] and FECCM[7] bits are fixed to 1.<br>The FECCM[6] to FECCM[0] bits indicate the ECC encoder output for the data flash memory. |

## 4.18 FECCTMD_n — Flash ECC Test Mode Register (n = 0,1)

FECCTMD_n sets the ECC test function for the flash memory.

**Access:** This register can be read / written in 16-bit units.

**Address:** <FLASH_RGn_base> + $0104_H$ (n = 0,1)

**Value after reset:** $0030_H$

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | KEY | | | | | | | | — | — | CECCVE | DECCVE | — | — | — | ECCDISE |
| Value after reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 |
| R/W | R/W*1 | R/W*1 | R/W*1 | R/W*1 | R/W*1 | R/W*1 | R/W*1 | R/W*1 | R | R | R/W*2 | R/W*2 | R | R | R | R/W*2 |

Note 1. Written data is not stored in this bit. The value read is always $00_H$.

Note 2. Writing to this bit is enabled only when $A6_H$ is written to the KEY bits.

Table 4.20 FECCTMD_n Register Contents

| Bit Position | Bit Name | Function |
|---|---|---|
| 15 to 8 | KEY | Key Code<br>These bits enable or disable CECCVE, DECCVE, and ECCDISE bits modification. |
| 7,6 | — | Reserved<br>These bits are always read as “0”. Write value should always be “0”. |
| 5 | CECCVE | Code Flash Memory ECC Area Verify Enable<br>Specifies the verify operation on programming/erasure the code flash memory.<br>0: Verifies the data area only.<br>1: Verifies the data area and the ECC area. |
| 4 | DECCVE | Data Flash Memory ECC Area Verify Enable<br>Specifies the verify operation on programming/erasure the data flash memory.<br>0: Verifies the data area only.<br>1: Verifies the data area and the ECC area. |
| 3 to 1 | — | Reserved<br>These bits are always read as “0”. Write value should always be “0”. |
| 0 | ECCDISE | ECC Encoder Disable<br>Disables the address parity generator and the ECC encoder. If the address parity generator and the ECC encoder are disabled, the FDMYECC value is written to the flash memory.<br>0: The address parity generator and the ECC encoder are enabled.<br>1: The address parity generator and the ECC encoder are disabled. |

## 4.19 FDMYECC_n — Flash Dummy ECC Register (n = 0,1)

FDMYECC_n specifies the address parity and ECC value to be written into the flash memory when the ECCDISE bit in the FECCTMD_n register is 1. The bit functions in code flash programming/erasure mode are different from those in data flash programming/erasure mode as shown below.

**Access:** This register can be read / written in 16-bit units.

**Address:** <FLASH_RGn_base> + $0108_H$ (n = 0,1)

**Value after reset:** $FFFF_H$

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | — | — | — | — | — | — | DMYAPAR | DMYECC[8:0] | | | | | | | | |
| Value after reset | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| R/W | R | R | R | R | R | R | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Table 4.21 FDMYECC_n Register Contents (in Code Flash Programming/Erasure Mode)

| Bit Position | Bit Name | Function |
|---|---|---|
| 15 to 10 | — | Reserved<br>These bits are always read as 1. Write value should always be 1. |
| 9 | DMYAPAR | Dummy Address Parity<br>Specifies the address parity value when the ECCDISE bit is 1. |
| 8 to 0 | DMYECC[8:0] | Dummy ECC<br>Specify the ECC value when the ECCDISE bit is 1. |

Table 4.22 FDMYECC_n Register Contents (in Data Flash Programming/Erasure Mode)

| Bit Position | Bit Name | Function |
|---|---|---|
| 15 to 10 | — | Reserved<br>These bits are always read as 1. Write value should always be 1. |
| 9 | DMYAPAR | Reserved<br>This bit is always read as 1. Write value should always be 1. |
| 8 to 7 | DMYECC[8:7] | Reserved<br>This bit is always read as 1. Write value should always be 1. |
| 6 to 0 | DMYECC[6:0] | Dummy ECC<br>Specify the ECC value when the ECCDISE bit is 1. |

## 4.20 SPIDINn — Serial Programmer ID Input Register (n = 0 to 7)

SPIDINn is for the input of an ID for use in authentication at the time of serial-programming. The ID is authenticated by comparing the 256-bit ID that has been set in advance in Security Setting Area of flash memory with the value in the SPIDIN0 to SPIDIN7 registers. The ID which is stored in Security Setting Area of the flash memory can be set by the Setting of Security Settings command.

SPIDINn have to be inputted in sequence from lowest register number.

**Access:** This register can be read / written in 32-bit units.

**Address:** <IDCTRL_base> + $00000_H$ + n × $4_H$ (n = 0 to 7)

**Value after reset:** $0000\ 0000_H$

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | SPIDINn[31:16]*1 | | | | | | | | | | | | | | | |
| Value after reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | SPIDINn[15:0]*1 | | | | | | | | | | | | | | | |
| Value after reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Note 1. n = 0 to 7

Table 4.23 SPIDINn Register Contents

| Bit Position | Bit Name | Function |
|---|---|---|
| 31 to 0 | SPIDINn[31:0] | ID for Use in Authentication of Serial-Programming<br>The ID for use in authentication at the time of Serial-programming is input to these bits.<br>Authentication of the ID is executed by comparing the 256-bit ID that has been set in advance in Security Setting Area of flash memory with the value in the SPIDINn[31:0] bits.<br>The correspondence of the 256-bit ID compared with SPIDINn[31:0] is as follows.<br>ID[31:0]: SPIDIN0[31:0]<br>ID[63:32]: SPIDIN1[31:0]<br>ID[95:64]: SPIDIN2[31:0]<br>ID[127:96]: SPIDIN3[31:0]<br>ID[159:128]: SPIDIN4[31:0]<br>ID[191:160]: SPIDIN5[31:0]<br>ID[223:192]: SPIDIN6[31:0]<br>ID[255:224]: SPIDIN7[31:0] |

## 4.21 DFIDINn — Data Flash ID Input Register (n = 0 to 7)

DFIDINn is for the input of an ID for use in authentication for Data Flash Protection. The ID is authenticated by comparing the 256-bit ID that has been set in advance in Security Setting Area of flash memory with the value in the DFIDIN0 to DFIDIN7 registers. The ID which is stored in Security Setting Area of the flash memory can be set by the Setting of Security Settings command.

DFIDINn have to be inputted in sequence from lowest register number.

**Access:** This register can be read / written in 32-bit units.

**Address:** <IDCTRL_base> + $00020_H$ + n × $4_H$ (n = 0 to 7)

**Value after reset:** $0000\ 0000_H$

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | DFIDINn[31:16]*1 | | | | | | | | | | | | | | | |
| Value after reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | DFIDINn[15:0]*1 | | | | | | | | | | | | | | | |
| Value after reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Note 1. n = 0 to 7

Table 4.24 DFIDINn Register Contents

| Bit Position | Bit Name | Function |
|---|---|---|
| 31 to 0 | DFIDINn[31:0] | ID for Use in Authentication for Data Flash Protection<br>The ID for use in authentication at the time of release Data Flash ID Protection.<br>Authentication of the ID is executed by comparing the 256-bit ID that has been set in advance in Security Setting Area of flash memory with the value in the DFIDINn[31:0] bits.<br>The correspondence of the 256-bit ID compared with DFIDINn[31:0] is as follows.<br>ID[31:0]: DFIDIN0[31:0]<br>ID[63:32]: DFIDIN1[31:0]<br>ID[95:64]: DFIDIN2[31:0]<br>ID[127:96]: DFIDIN3[31:0]<br>ID[159:128]: DFIDIN4[31:0]<br>ID[191:160]: DFIDIN5[31:0]<br>ID[223:192]: DFIDIN6[31:0]<br>ID[255:224]: DFIDIN7[31:0] |

## 4.22 OCDIDINn — OCD ID Input Register (n = 0 to 7)

OCDIDINn is for the input of an ID for use in authentication for on-chip debug settings Protection. The ID is authenticated by comparing the 256-bit ID that has been set in advance in Security Setting Area of flash memory with the value in the OCDIDIN0 to OCDIDIN7 registers. The ID which is stored in Security Setting Area of the flash memory can be set by the Setting of Security Settings command.

OCDIDINn have to be inputted in sequence from lowest register number.

**Access:** This register can be read / written in 32-bit units.

**Address:** <IDCTRL_base> + $00040_H$ + n × $4_H$ (n = 0 to 7)

**Value after reset:** $0000\ 0000_H$

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | OCDIDINn[31:16]*1 | | | | | | | | | | | | | | | |
| Value after reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | OCDIDINn[15:0]*1 | | | | | | | | | | | | | | | |
| Value after reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Note 1. n = 0 to 7

Table 4.25 OCDIDINn Register Contents

| Bit Position | Bit Name | Function |
|---|---|---|
| 31 to 0 | OCDIDINn[31:0] | ID for Use in Authentication for on-chip debug settings Protection<br>The ID for use in authentication at the time of release OCDID Protection.<br>Authentication of the ID is executed by comparing the 256-bit ID that has been set in advance in Security Setting Area of flash memory with the value in the OCDIDINn[31:0] bits.<br>The correspondence of the 256-bit ID compared with OCDIDINn[31:0] is as follows.<br>ID[31:0]: OCDIDIN0[31:0]<br>ID[63:32]: OCDIDIN1[31:0]<br>ID[95:64]: OCDIDIN2[31:0]<br>ID[127:96]: OCDIDIN3[31:0]<br>ID[159:128]: OCDIDIN4[31:0]<br>ID[191:160]: OCDIDIN5[31:0]<br>ID[223:192]: OCDIDIN6[31:0]<br>ID[255:224]: OCDIDIN7[31:0] |

## 4.23 CTESTIDINn — C-TEST ID Input Register (n = 0 to 7)

CTESTIDINn is for the input of an ID for use in authentication for product test mode entry ID Protection. The ID is authenticated by comparing the 256-bit ID that has been set in advance in Security Setting Area of flash memory with the value in the CTESTIDIN0 to CTESTIDIN7 registers. The ID which is stored in Security Setting Area of the flash memory can be set by the Setting of Security Settings command.

CTESTIDINn have to be inputted in sequence from lowest register number.

**Access:** This register can be read / written in 32-bit units.

**Address:** <IDCTRL_base> + $00060_H$ + n × $4_H$ (n = 0 to 7)

**Value after reset:** $0000\ 0000_H$

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | CTESTIDINn[31:16]*1 | | | | | | | | | | | | | | | |
| Value after reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | CTESTIDINn[15:0]*1 | | | | | | | | | | | | | | | |
| Value after reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Note 1. n = 0 to 7

Table 4.26 CTESTIDINn Register Contents

| Bit Position | Bit Name | Function |
|---|---|---|
| 31 to 0 | CTESTIDINn[31:0] | ID for Use in Authentication for Test Mode Entry Protection.<br>The ID for use in authentication at the time of release CTESTID Protection.<br>Authentication of the ID is executed by comparing the 256-bit ID that has been set in advance in Security Setting Area of flash memory with the value in the CTESTIDINn[31:0] bits.<br>The correspondence of the 256-bit ID compared with CTESTIDINn[31:0] is as follows.<br>ID[31:0]: CTESTIDIN0[31:0]<br>ID[63:32]: CTESTIDIN1[31:0]<br>ID[95:64]: CTESTIDIN2[31:0]<br>ID[127:96]: CTESTIDIN3[31:0]<br>ID[159:128]: CTESTIDIN4[31:0]<br>ID[191:160]: CTESTIDIN5[31:0]<br>ID[223:192]: CTESTIDIN6[31:0]<br>ID[255:224]: CTESTIDIN7[31:0] |

## 4.24 CUSTIDINn — Customer ID Input Register (n = 0 to 7)

CUSTIDINn is for the input of an ID for use in authentication for Code Flash / Security Setting / Block Protection Setting / Configuration Setting Protection. The ID is authenticated by comparing the 256-bit ID that has been set in advance in Security Setting Area of flash memory with the value in the CUSTIDIN0 to CUSTIDIN7 registers. The ID which is stored in Security Setting Area of the flash memory can be set by the Setting of Security Settings command.

CUSTIDINn have to be inputted in sequence from lowest register number.

**Access:** This register can be read / written in 32-bit units.

**Address:** <IDCTRL_base> + $00080_H$ + n × $4_H$ (n = 0 to 7)

**Value after reset:** $0000\ 0000_H$

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | CUSTIDINn[31:16]*1 | | | | | | | | | | | | | | | |
| Value after reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | CUSTIDINn[15:0]*1 | | | | | | | | | | | | | | | |
| Value after reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Note 1. n = 0 to 7

Table 4.27 CUSTIDINn Register Contents

| Bit Position | Bit Name | Function |
|---|---|---|
| 31 to 0 | CUSTIDINn[31:0] | ID for Use in Authentication for Code Flash / Security Setting / Block Protection Setting / Configuration Setting Protection<br>The ID for use in authentication at the time of release Customer ID Protection.<br>Authentication of the ID is executed by comparing the 256-bit ID that has been set in advance in Security Setting Area of flash memory with the value in the CUSTIDINn[31:0] bits.<br>The correspondence of the 256-bit ID compared with CUSTIDINn[31:0] is as follows.<br>ID[31:0]: CUSTIDIN0[31:0]<br>ID[63:32]: CUSTIDIN1[31:0]<br>ID[95:64]: CUSTIDIN2[31:0]<br>ID[127:96]: CUSTIDIN3[31:0]<br>ID[159:128]: CUSTIDIN4[31:0]<br>ID[191:160]: CUSTIDIN5[31:0]<br>ID[223:192]: CUSTIDIN6[31:0]<br>ID[255:224]: CUSTIDIN7[31:0] |

## 4.25 IDST —ID Authentication Status Register

IDST indicates ID authentication results.

**Access:** This register can be read in 32-bit units.

**Address:** <IDCTRL_base> + 001FC$_H$

**Value after reset:** 0000 00XX$_H$

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — |
| Value after reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W | R | R | R | R | R | R | R | R | R | R | R | R | R | R | R | R |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | — | — | — | — | — | — | — | — | — | — | — | CUSTIDR | CTESTIDR | OCDIDR | DFIDR | SPIDR |
| Value after reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0/1*5 | 0/1*4 | 0/1*3 | 0/1*2 | 0/1*1 |
| R/W | R | R | R | R | R | R | R | R | R | R | R | R | R | R | R | R |

Note 1. When serial Programming ID[255:0] = All-0, initial value is 0. Otherwise initial value is 1.

Note 2. When Data Flash ID[255:0] = All-1 or All-0, initial value is 0. Otherwise initial value is 1.
When Data Flash ID[255:0] = All-1, the value is always 0 because authentication is not required.

Note 3. When OCD ID[255:0] = All-1 or All-0, initial value is 0. Otherwise initial value is 1.
When OCD ID[255:0] = All-1, the value is always 0 because authentication is not required.

Note 4. When C-TEST ID[255:0] = All-1 or All-0, initial value is 0. Otherwise initial value is 1.
When C-TEST ID[255:0] = All-1, the value is always 0 because authentication is not required.

Note 5. When Customer ID[255:0] = All-1 or All-0, initial value is 0. Otherwise initial value is 1.
When Customer ID[255:0] = All-1, the value is always 0 because authentication is not required.

Table 4.28 IDST Register Contents (1/2)

| Bit Position | Bit Name | Function |
|---|---|---|
| 31 to 5 | — | Reserved<br>These bits are always read as 0. |
| 4 | CUSTIDR | Customer ID Authentication Status<br>This bit indicates the result of comparing the Customer ID with the CUSTIDINn registers.<br>0: The IDs match (protection unlocked).<br>1: The IDs do not match (protection locked). |
| 3 | CTESTIDR | C-TEST ID Authentication Status<br>This bit indicates the result of comparing the C-TEST ID with the CTESTIDINn registers.<br>0: The IDs match (protection unlocked).<br>1: The IDs do not match (protection locked). |
| 2 | OCDIDR | OCD ID Authentication Status<br>This bit indicates the result of comparing the OCD ID with the OCDIDINn registers.<br>0: The IDs match (protection unlocked).<br>1: The IDs do not match (protection locked). |
| 1 | DFIDR | Data-Flash ID Authentication Status<br>This bit indicates the result of comparing the Data Flash ID with the DFIDINn registers.<br>0: The IDs match (protection unlocked).<br>1: The IDs do not match (protection locked). |

Table 4.28 IDST Register Contents (2/2)

| Bit Position | Bit Name | Function |
|---|---|---|
| 0 | SPIDR | Serial-Programming ID Authentication Status<br>This bit indicates the result of comparing the Serial Programming ID with the SPIDINn registers.<br>0: The IDs match (protection unlocked).<br>1: The IDs do not match (protection locked). |

## 4.26 RHSIFIDINn — RHSIF ID Input Register (n = 0 to 7)

RHSIFIDINn is for the input of an ID for use in authentication for RHSIF settings Protection. The ID is authenticated by comparing the 256-bit ID that has been set in advance in Security Setting Area of flash memory with the value in the RHSIFIDIN0 to RHSIFIDIN7 registers. The ID which is stored in Security Setting Area of the flash memory can be set by the Setting of Security Settings command.

RHSIFIDINn have to be inputted in sequence from lowest register number.

**Access:** This register can be read / written in 32-bit units.

**Address:** <IDCTRL_base> + $00200_H$ + n × $4_H$ (n = 0 to 7)

**Value after reset:** $0000\ 0000_H$

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | RHSIFIDINn[31:16]*1 | | | | | | | | | | | | | | | |
| Value after reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | RHSIFIDINn[15:0]*1 | | | | | | | | | | | | | | | |
| Value after reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Note 1. n = 0 to 7

Table 4.29 RHSIFIDINn Register Contents

| Bit Position | Bit Name | Function |
|---|---|---|
| 31 to 0 | RHSIFIDINn[31:0] | ID for Use in Authentication for RHSIF settings Protection<br>The ID for use in authentication at the time of release RHSIF ID Protection.<br>Authentication of the ID is executed by comparing the 256-bit ID that has been set in advance in Security Setting Area of flash memory with the value in the RHSIFIDINn[31:0] bits.<br>The correspondence of the 256-bit ID compared with RHSIFIDINn[31:0] is as follows.<br>ID[31:0]: RHSIFIDIN0[31:0]<br>ID[63:32]: RHSIFIDIN1[31:0]<br>ID[95:64]: RHSIFIDIN2[31:0]<br>ID[127:96]: RHSIFIDIN3[31:0]<br>ID[159:128]: RHSIFIDIN4[31:0]<br>ID[191:160]: RHSIFIDIN5[31:0]<br>ID[223:192]: RHSIFIDIN6[31:0]<br>ID[255:224]: RHSIFIDIN7[31:0] |

## 4.27 IDST2 —ID Authentication Status Register 2

IDST2 indicates ID authentication result.

**Access:** This register can be read in 32-bit units.

**Address:** <IDCTRL_base> + 003FC$_H$

**Value after reset:** 0000 000X$_H$

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — |
| Value after reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W | R | R | R | R | R | R | R | R | R | R | R | R | R | R | R | R |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | RHSIFIDR |
| Value after reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0/1*1 |
| R/W | R | R | R | R | R | R | R | R | R | R | R | R | R | R | R | R |

Note 1. When RHSIF ID[255:0] = ALL-1 or All-0, initial value is 0. Otherwise, initial value is 1.
When RHSIF ID[255:0] = All-1, the value is always 0 because authentication is not required.

Table 4.30 IDST2 Register Contents

| Bit Position | Bit Name | Function |
|---|---|---|
| 31 to 1 | — | Reserved<br>These bits are always read as 0. |
| 0 | RHSIFIDR | RHSIF ID Authentication Status<br>This bit indicates the result of comparing the RHSIF ID with the RHSIFIDINn registers.<br>0: The IDs match (protection unlocked).<br>1: The IDs do not match (protection locked). |

## 4.28 FHVE15 — FHVE15 Control Register

FHVE15 is a readable and writable register for protecting the flash memory against programming and erasure.

To proceed with programming and erasure of the flash memory, set both the FHVE3 and FHVE15 registers to 0000 0001$_H$ that allows this.

If these registers are set to 0000 0000$_H$ that does not allow programming and erasure of the flash memory, the following commands cannot be executed.

Programming / Multi Programming / DMA programming

Block erasure / Area erasure / Blank checking

Programming/erasure suspension / Programming/erasure resumption

Setting of Configuration Settings

Setting of Block Protection Settings

Setting of Security Settings

**Access:** This register can be read / written in 32-bit units.

**Address:** <SYSCTRL_base> + 3804$_H$

**Value after reset:** 0000 0000$_H$

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — |
| Value after reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W | R | R | R | R | R | R | R | R | R | R | R | R | R | R | R | R |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | FHVE 15CNT |
| Value after reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W | R | R | R | R | R | R | R | R | R | R | R | R | R | R | R | R/W |

Table 4.31 FHVE15 Register Contents

| Bit Position | Bit Name | Function |
|---|---|---|
| 31 to 1 | — | Reserved<br>These bits are always read as 0. Write value should always be 0. |
| 0 | FHVE15CNT | 0: Programming/erasure/DataFlashBlankChecking disabled<br>1: Programming/erasure/DataFlashBlankChecking enabled |

## 4.29 FHVE3 — FHVE3 Control Register

FHVE3 is a readable and writable register for protecting the flash memory against programming and erasure.

To proceed with programming and erasure of the flash memory, set both the FHVE3 and FHVE15 registers to 0000 0001$_H$ that allows this.

If these registers are set to 0000 0000$_H$ that does not allow programming and erasure of the flash memory, the following commands cannot be executed. Issuing any of the following commands leads to setting of the FHVEERR bit in the FSTATRn register to 1.

Programming / Multi Programming / DMA programming

Block erasure / Area erasure / Blank checking

Programming/erasure suspension / Programming/erasure resumption

Setting of Configuration Settings

Setting of Block Protection Settings

Setting of Security Settings

**Access:** This register can be read / written in 32-bit units.

**Address:** <SYSCTRL_base> + 3800$_H$

**Value after reset:** 0000 0000$_H$

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — |
| Value after reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W | R | R | R | R | R | R | R | R | R | R | R | R | R | R | R | R |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | — | — | — | — | — | — | — | — | — | — | — | — | — | — | — | FHVE3CNT |
| Value after reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W | R | R | R | R | R | R | R | R | R | R | R | R | R | R | R | R/W |

Table 4.32 FHVE3 Register Contents

| Bit Position | Bit Name | Function |
|---|---|---|
| 31 to 1 | — | Reserved<br>These bits are always read as 0. Write value should always be 0. |
| 0 | FHVE3CNT | 0: Programming/erasure/DataFlashBlankChecking disabled<br>1: Programming/erasure/DataFlashBlankChecking enabled |

# Section 5 Flash Sequencer Operating Modes

The flash sequencer has three operating modes as shown in **Figure 5.1**. The mode is shifted by the write to the FENTRYR_0/1 register.

When the FENTRYR_0/1 register is $0000_H$, the flash sequencer is in read mode. In this mode, it does not accept the FACI command. The code flash memory and the data flash memory are both readable.

When the FENTRYR_0 register is $0001_H$, the flash sequencer is in code flash programming/erasure mode where the code flash memory can be programmed/erased by the FACI command. In this mode, the data flash memory is not readable (except data flash for ICUM) while the flash sequencer is processing state (FRDY bit in FSTATR_0 register is "0" or FHVEERR bit in FSTARTR_0 is "1"). In addition, Flash sequencer controls one bank of code flash memory, and this bank of code flash memory is not readable while the flash sequencer is processing state (FRDY bit in FSTATR_0 register is "0" or FHVEERR bit in FSTARTR_0 is "1"). Other bank of code flash memory is readable (i.e. BGO: background operation).

When the FENTRYR_0/1 register is $0080_H$, the flash sequencer is in data flash programming/erasure mode where the data flash memory can be programmed/erased by the FACI command. In this mode, Flash sequencer controls one bank of data flash memory, and this bank of the data flash memory is not readable while the flash sequencer is processing state (FRDY bit in each FSTATR_n register is "0" or FHVEERR bit in FSTARTR_n is "1"). However, the code flash memory and other bank of data flash is readable (i.e. BGO).

In addition, it is independent to set FENTRYR_0 and FENTRYR_1 to $0080_H$. Data flash for ICUM and other Bank of Data flash can be in at a flash programming/erasure mode (i.e. Dual Operation).

As for the condition to enable the BGO and Dual Operation, refer to the user's manual for this product.

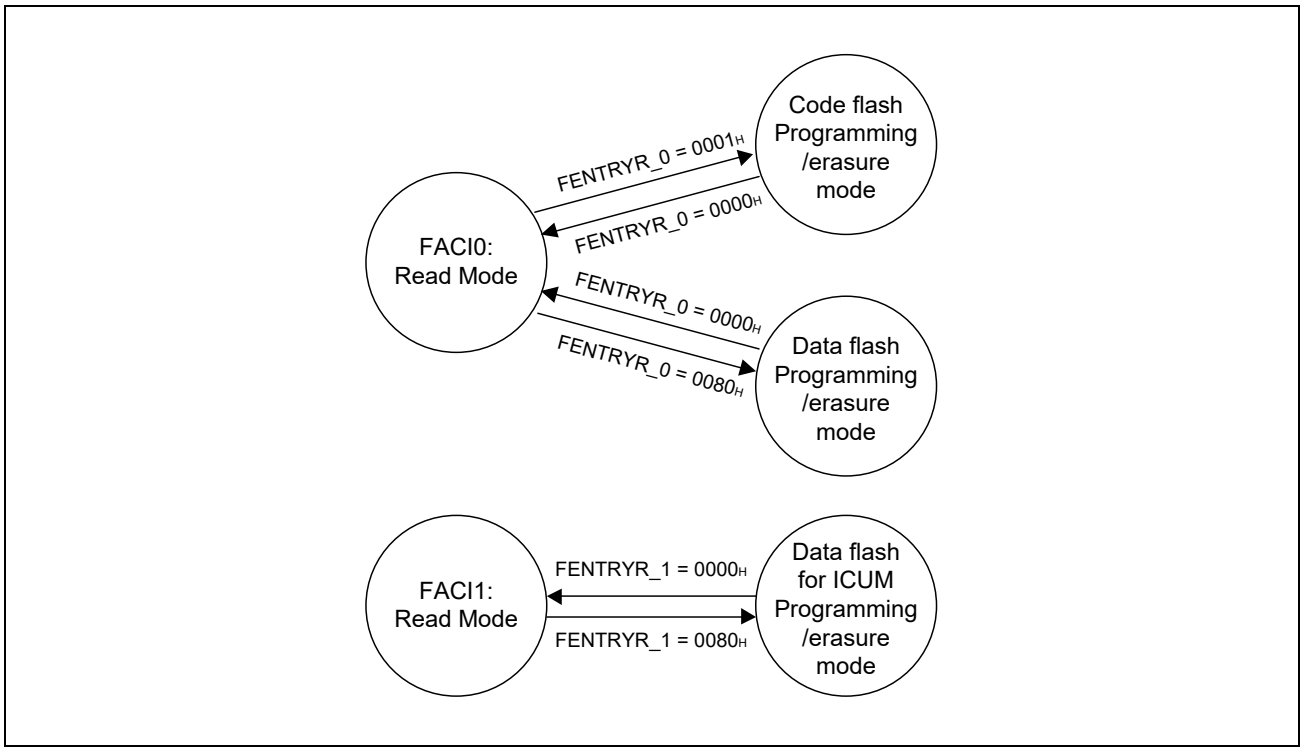



Figure 5.1 Flash Sequencer Modes

# Section 6 FACI Command

## 6.1 List of FACI Commands

Table 6.1 List of FACI Commands

| FACI Command | Function |
|---|---|
| Programming | • User area, user boot area and data area can be programmed.<br>• Programming unit is 512 bytes for user area, 512 bytes for user boot area and 4 bytes for data area. |
| Multi Programming | • Data area can be programmed.<br>• Program unit is 8, 16 or 32 bytes |
| DMA programming | • Data area can be programmed using the DMA controller.<br>• Programming unit is 4 to 48 Kbytes (4 bytes step).<br>• Maximum program unit is limited within one Data Flash.*1<br>• Address boundary between different Data Flash cannot be crossed. |
| Block erasure | • User area, user boot area and data area can be erased.<br>• Erasure unit is one block. |
| Area erasure | • Data area can be erased.<br>• Erasure unit are N blocks. (64 bytes × N = 1, 2, 3 ....)<br>• Maximum erasure unit is limited within one Data Flash.*1<br>• Address boundary between different Data Flash cannot be crossed. |
| Programming/erasure suspension | Programming/MultiProgramming or Block erasure/Area erasure command operation can be suspended. |
| Programming/erasure resumption | Suspended programming/MultiProgramming or Block erasure/Area erasure command operation can be resumed. |
| Status clearing | Error detection and correction flags in the FASTAT_n and FSTATR_n registers are cleared and the flash sequencer is released from “Command Lock” state. |
| Forced stop | Any command operation can be stopped forcibly and the FASTAT_n register and FSTATR_n register are initialized. |
| Blank checking | • Data area can be checked.<br>• Blank checking unit is 4 to 48 Kbytes (4 bytes step).<br>• Maximum blank checking unit is limited within one Data Flash.*1<br>• Address boundary between different Data Flash cannot be crossed. |
| Setting of Configuration Settings | • Settings (Option bytes, Reset Vector, etc.) in Configuration Setting Area can be changed from the initial values for this product.<br>• Setting unit is 4 or 32 bytes. |
| Setting of Block Protection Settings | • Settings (OTP settings, Block Protection Settings) in Block Protection Area can be changed from the initial values for this product.<br>• Setting unit is 4 or 32 bytes. |
| Setting of Security Settings | • Settings (ID Codes, Debugger connection prohibited setting, etc.) in Security Setting Area can be changed from the initial values for this product.<br>• Setting unit is 4 or 32 bytes. |

Note 1. The 96 KBytes Data Flash bank consists of two 48 KBytes Data Flash (EEP) called EEP0-0 and EEP0-1, or EEP1-0 and EEP1-1. 32 KBytes Data Flash bank for ICUM consists of one 32 Kbytes Data Flash called EEP2. These EEPs are treated as different Data Flash.

The FACI commands are issued by the write access to the FACI command-issuing area (see **Table 3.1**). When the write access as shown in **Table 6.2** is issued in the specified state, the flash sequencer executes the processing corresponding to each command (see **Section 6.2, Relationship between Flash Sequencer Status and FACI Commands**).

Table 6.2 Flash Sequencer Command Format

| FACI Command | Number of Write Access | Write Data to FACI Command-issuing Area | | | |
|---|---|---|---|---|---|
| | | 1st Access*2 | 2nd Access*1 | 3rd to (N + 2)th Access | (N + 3)th Access*2 |
| Programming (User area / User boot area)<br>512-byte programming (N = 128) | 131 | $E8_H$ | $80_H$ (=N) | $WD_1$ to $WD_{128}$ | $D0_H$ |
| Programming (Data area)<br>4-byte programming (N = 1) | 4 | $E8_H$ | $01_H$ (=N) | $WD_1$ | $D0_H$ |
| Multi Programming<br>8 bytes (N = 2), 16 bytes (N = 4),<br>32 bytes (N = 8) | N + 3 | $ED_H$ | $02_H$ (=N)<br>$04_H$ (=N)<br>$08_H$ (=N) | $WD_1$ to $WD_N$ | $D0_H$ |
| DMA programming<br>N = 1 to 12288 | N + 2 | $EA_H$ | N | $WD_1$ to $WD_N$ | — |
| Block erasure | 2 | $20_H$ | $D0_H$ | — | — |
| Area erasure | 2 | $23_H$ | $D0_H$ | — | — |
| Programming/erasure suspension | 1 | $B0_H$ | — | — | — |
| Programming/erasure resumption | 1 | $D0_H$ | — | — | — |
| Status clearing | 1 | $50_H$ | — | — | — |
| Forced stop | 1 | $B3_H$ | — | — | — |
| Blank checking | 2 | $71_H$ | $D0_H$ | — | — |
| Setting of Configuration Settings<br>4 bytes (N = 1), 32 bytes (N = 8) | N+3 | $40_H$ | $01_H$ (=N)<br>$08_H$ (=N) | $WD_1$ to $WD_N$ | $D0_H$ |
| Setting of Block Protection Settings<br>4 bytes (N = 1), 32 bytes (N = 8) | N+3 | $80_H$ | $01_H$ (=N)<br>$08_H$ (=N) | $WD_1$ to $WD_N$ | $D0_H$ |
| Setting of Security Settings<br>4 bytes (N = 1), 32 bytes (N = 8) | N+3 | $60_H$ | $01_H$ (=N)<br>$08_H$ (=N) | $WD_1$ to $WD_N$ | $D0_H$ |

**Note:** $WD_N$ (N = 1, 2,…): Nth 32-bit data to be programmed.

Note 1. 8-bit data is written by a command other than DMA programming. By the DMA programming command, 16-bit data is written.

Note 2. 8-bit data is written by a command.

Once the flash sequencer starts processing any command other than status clearing, it sets the FRDY bit in the FSTATR_n register to 0, and when processing of the command is complete, it sets the FRDY bit to 1 (see **Section 4.8, FSTATR_n — Flash Status Register (n = 0,1)**). When the value of the FRDY bit changes from 0 to 1, a flash ready (FRDY) interrupt is generated.

## 6.2 Relationship between Flash Sequencer Status and FACI Commands

The FACI commands are accepted according to the mode/state of the flash sequencer.

The FACI command should be issued after the shift of the flash sequencer to the code flash programming/erasure mode or data flash programming/erasure mode and checking that the flash sequencer has shifted to the mode. To check the state of flash sequencer, use the FSTATR_n and FASTAT_n registers. In addition, error occurrence can be checked by the CMDLK bit in the FASTAT_n register. It is the logical OR of the SECDTCT, ILGCOMERR, FESETERR, SECERR, OTERR, BPLDTCT, ILGLERR, ERSERR, PRGERR, FHVEERR, CFGDTCT, and TBLDTCT bits of the FSTATR_n register.

**Table 6.3** summarizes available flash sequencer commands in each operating mode.

Table 6.3 Flash Sequencer Operation Mode and Available Commands

| Operating Mode | FENTRYR | Available Command |
|---|---|---|
| Read Mode | $0000_H$ | • No command is available. |
| Code flash programming/erasure Mode | $0001_H$ | • Programming<br>• Block erasure<br>• Programming/erasure suspension<br>• Programming/erasure resumption<br>• Status clearing<br>• Forced Stop |
| Data flash programming/erasure Mode | $0080_H$ | • Programming<br>• Multi Programming<br>• DMA programming<br>• Block erasure<br>• Area erasure<br>• Programming/erasure suspension<br>• Programming/erasure resumption<br>• Status clearing<br>• Forced Stop<br>• Blank checking<br>• Setting of Configuration Settings<br>• Setting of Block Protection Settings<br>• Setting of Security Settings |

**Table 6.4** shows the flash sequencer state and the acceptable FACI commands. The table assumes appropriate flash sequencer operation mode is set before issuing the command.

Table 6.4 Flash Sequencer State and Acceptable FACI Commands

| | Programming / Multi Programming / Block Erasure / Area Erasure command processing | Setting of Configuration Settings / Setting of Block Protection Settings / Setting of Security Settings command processing | Programming / Multi Programming / Block Erasure / Area Erasure command suspension processing | Blank checking command processing | DMA Programming command processing | While suspend Programming / Multi Programming command | While suspend Block Erasure / Area Erasure command | While suspend Block Erasure / Area Erasure command, and Programming / Multi Programming command processing | Command Lock state (FRDY = 1) | Command Lock state (FRDY = 0) | Forced Stop command processing | Other |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| FRDY bit | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 0 | 1 | 0 | 0 | 1 |
| SUSRDY bit | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| ERSSPD bit | 0 | 0 | 0/1 | 0/1 | 0 | 0 | 1 | 1 | 0/1 | 0/1 | 0 | 0 |
| PRGSPD bit | 0 | 0 | 0/1 | 0/1 | 0 | 1 | 0 | 0 | 0/1 | 0/1 | 0 | 0 |
| CMDLK bit | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 0 | 0 |
| Programming | X | X | X | X | X | X | O*2 | X | X | X | X | O |
| Multi Programming | X | X | X | X | X | X | O*1,*2 | X | X | X | X | O*1 |
| DMA Programming | X | X | X | X | X | X | O*1,*2 | X | X | X | X | O*1 |
| Block Erasure | X | X | X | X | X | X | X | X | X | X | X | O |
| Area Erasure | X | X | X | X | X | X | X | X | X | X | X | O*1 |
| Programming/erasure suspension | O | X | X | X | X | X | X | X | — | X | X | — |
| Programming/erasure resumption | X | X | X | X | X | O | O | X | X | X | X | X |
| Status clearing | X | X | X | X | X | O | O | X | O | X | X | O |
| Forced Stop | O | O | O | O | O | O | O | O | O | O | O | O |
| Blank checking | X | X | X | X | X | O*1 | O*1 | X | X | X | X | O*1 |
| Setting of Configuration Settings | X | X | X | X | X | X | X | X | X | X | X | O*1 |
| Setting of Block Protection Settings | X | X | X | X | X | X | X | X | X | X | X | O*1 |
| Setting of Security Settings | X | X | X | X | X | X | X | X | X | X | X | O*1 |

**Remarks:**
O: Acceptable
X: Not acceptable (in Command Lock state)
—: Ignored (Not acceptable and not enter Command Lock state)

Note 1. Acceptable only in data flash programming/erasure mode.

Note 2. Acceptable when programming area is other than erasure suspending block. Flash sequencer enters “Command Lock” state and occur ILGLERR (not occur ILGLCOMERR) when programming area is in suspending block.

## 6.3 Use FACI Command

This section describes the overview of FACI command usage.

### 6.3.1 Overview of the Command Usage in Code Flash Programming/Erasure Mode

The overview of the FACI command usage in code flash programming/erasure mode is shown below.

**Table 6.3** lists the available commands in code flash programming/erasure mode.

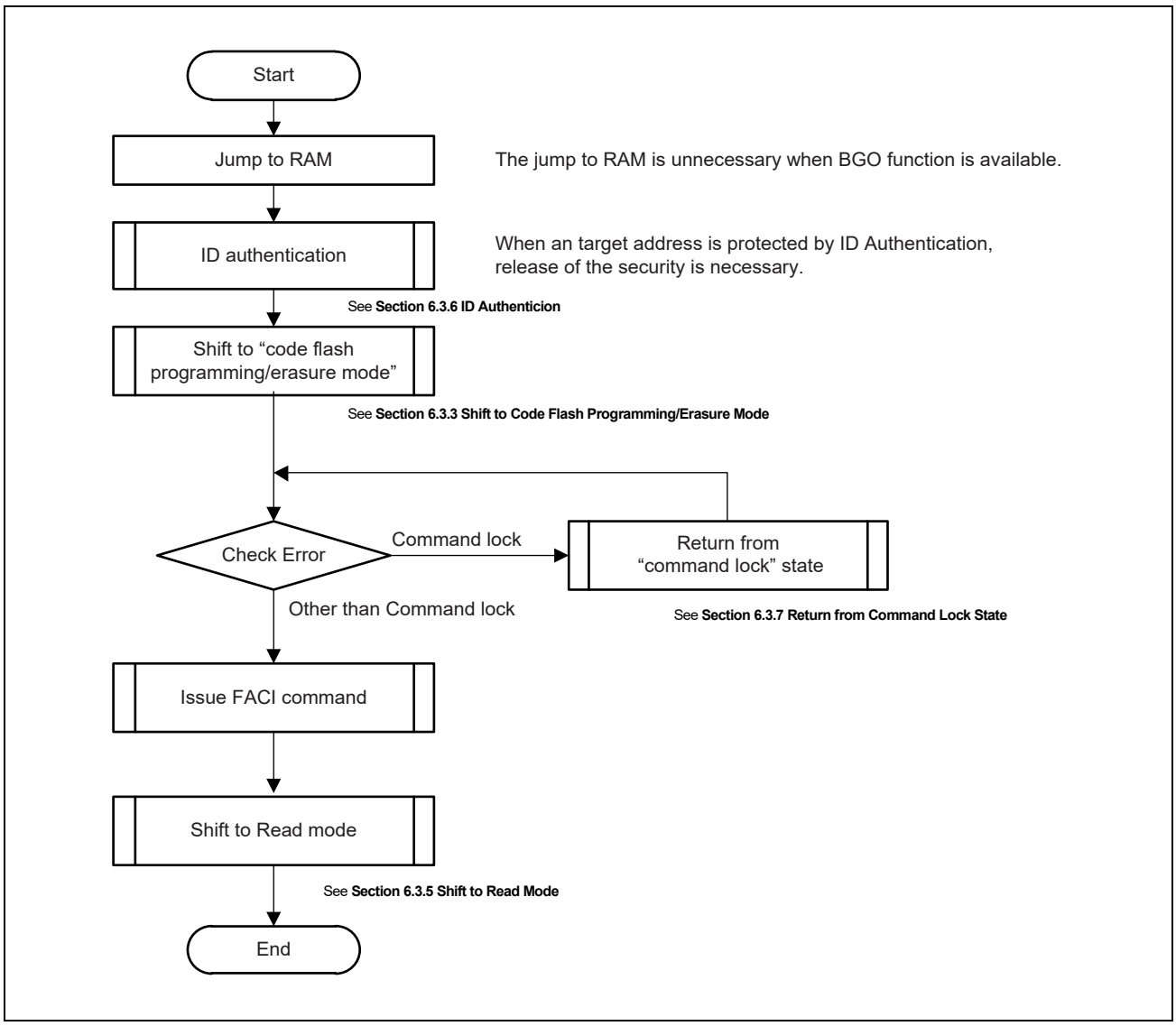



Figure 6.1 Overview of Command Usage in Code Flash Programming/Erasure Mode

### 6.3.2 Overview of the Command Usage in Data Flash Programming/Erasure Mode

The overview of the FACI command usage in data flash programming/erasure mode is shown below.

As for the available commands in data flash programming/erasure mode, refer to **Table 6.3**.

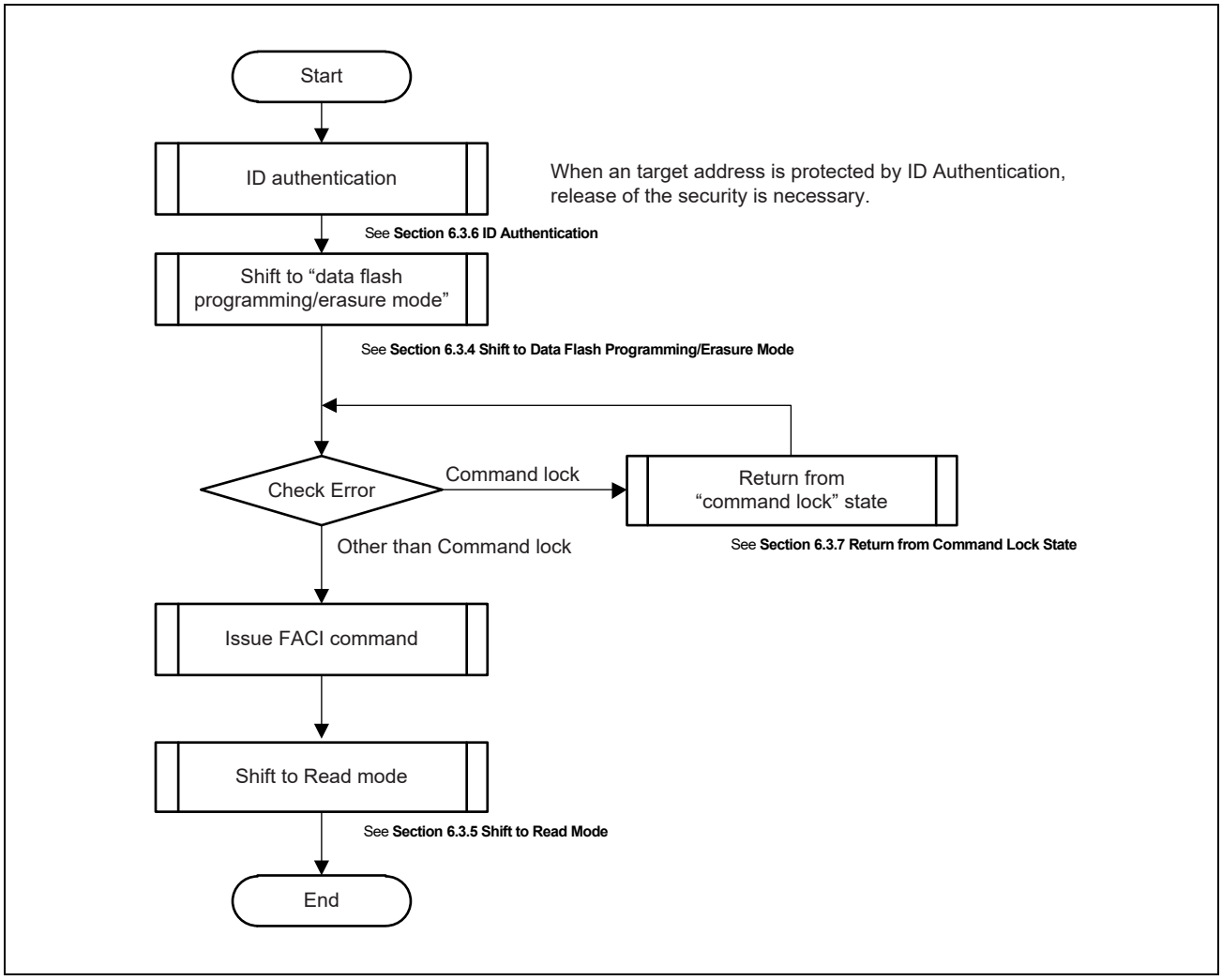
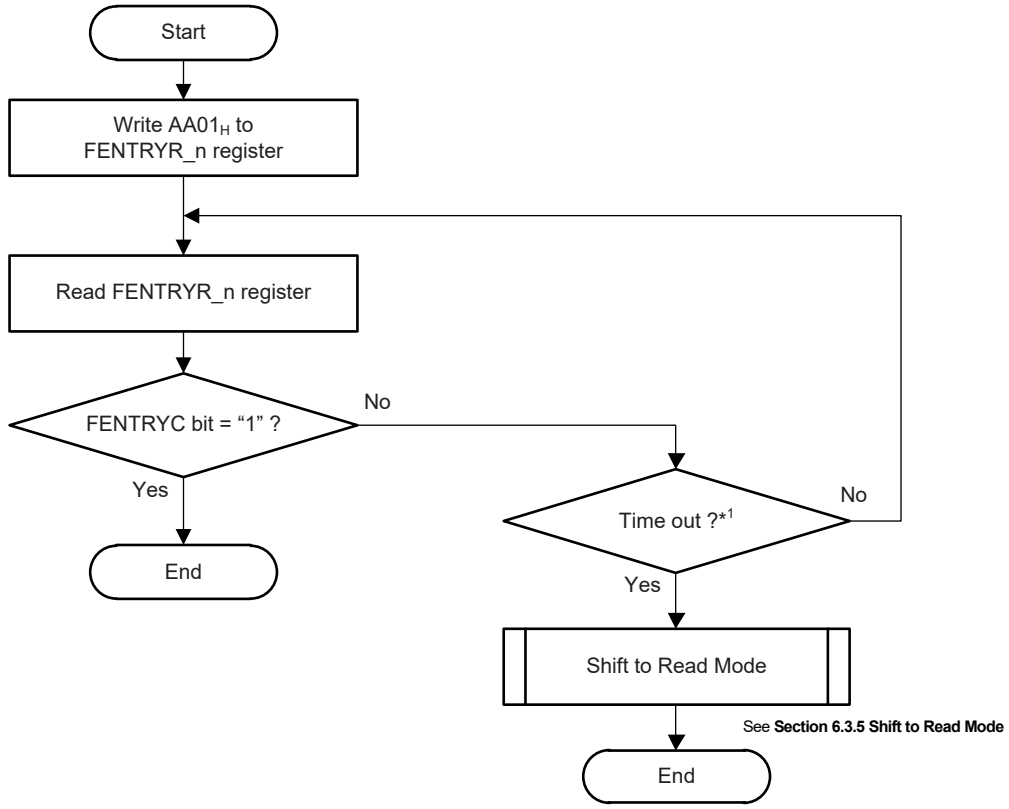



Figure 6.2 Overview of Command Usage in Data Flash Programming/Erasure Mode

### 6.3.3 Shift to Code Flash Programming/Erasure Mode

To use the FACI commands relating the code flash memory, operation should be shifted to the code flash programming/erasure mode. Set the FENTRYRC bit in the each FENTRYR_n to 1 to shift to the code flash programming/erasure mode.

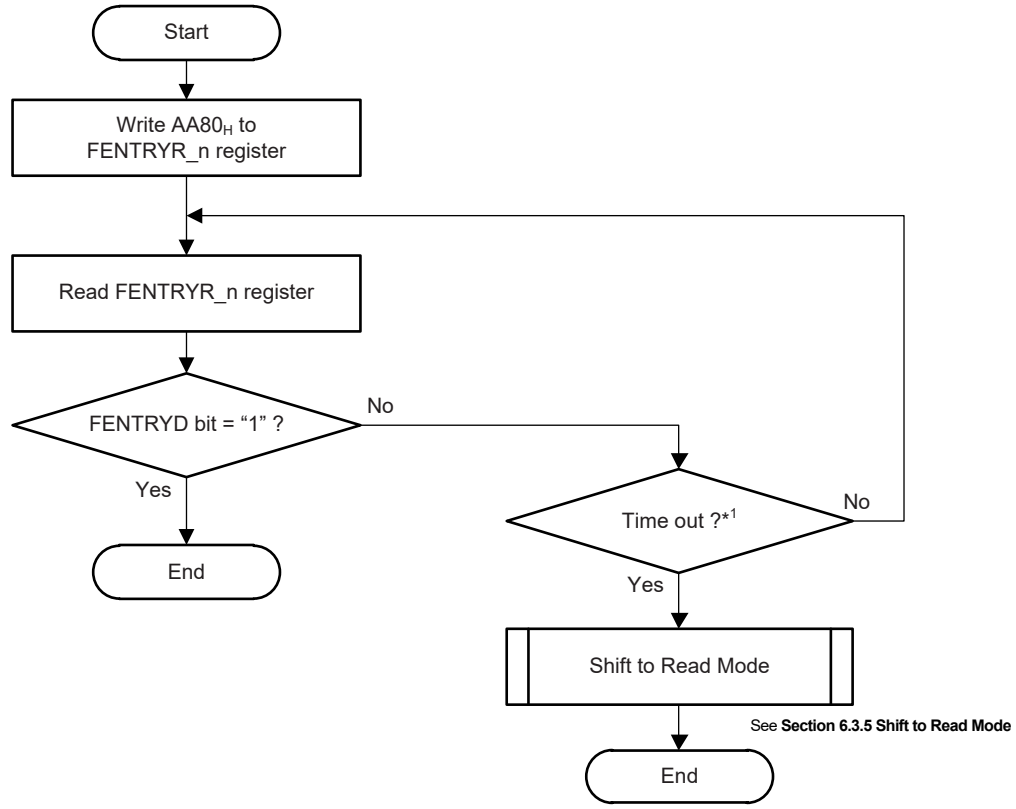



Note 1. If ICUM is disabled: 2 times
If ICUM is enabled: It depends on the time required for software arbitration between ICUMD and Main-PE. See *Section 17, Flash Memory*, in *the RH850/E2x ICUMD User's Manual*.

Figure 6.3 Flow of Shift to Code Flash Programming/Erasure Mode

### 6.3.4 Shift to Data Flash Programming/Erasure Mode

To use the FACI commands relating the data flash memory, operation should be shifted to the data flash programming/erasure mode. Set the FENTRYRD bit in the each FENTRYR_n to 1 to shift to the data flash programming/erasure mode.




Note 1. If ICUM is disabled: 2 times
If ICUM is enabled: It depends on the time required for software arbitration between ICUMD and Main-PE. See *Section 17, Flash Memory*, in *the RH850/E2x ICUMD User's Manual*.

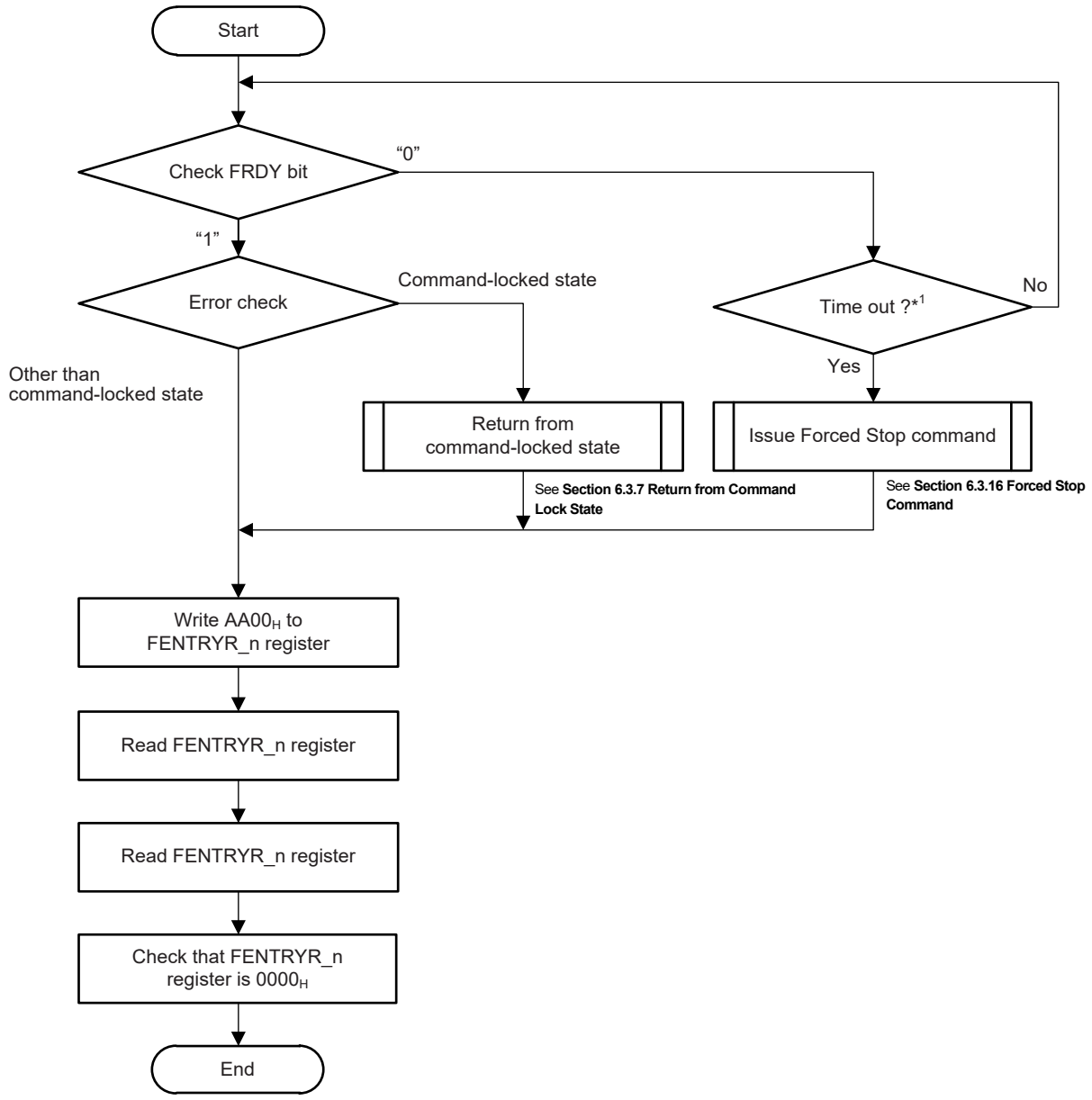
Figure 6.4 Flow of Shift to Data Flash Programming/Erasure Mode

### 6.3.5 Shift to Read Mode

To read the flash memory without using the BGO function, the operation should be shifted to the read mode.

To shift to the read mode, set the FENTRYR_n register to $0000_H$.

When entering the read mode, the flash sequencer processing should be completed and the operation is better to in other than command lock state (except FHVEERR). When entering the read mode, the FHVEERR bit should be “0”.




Note 1. Processing of an FACI command reaching 1.1 times the maximum processing time (see **Section 10, Electrical Characteristics**) can be considered a timeout.
If common processing is used, 1.1 times the maximum erasure time for a 64-Kbyte block of code flash memory can be considered a timeout.

Figure 6.5 Flow of Shift to Read Mode

### 6.3.6 ID Authentication

When a target address is protected by ID Authentication, Release of the security is necessary. For detailed of Address to protect in each ID, see *Section 42, Basic Hardware Protection (BHP)*, in *the User's Manual: Hardware*.

**Figure 6.6** shows the each ID compare method, and how the compare result is checked by IDST or IDST2.

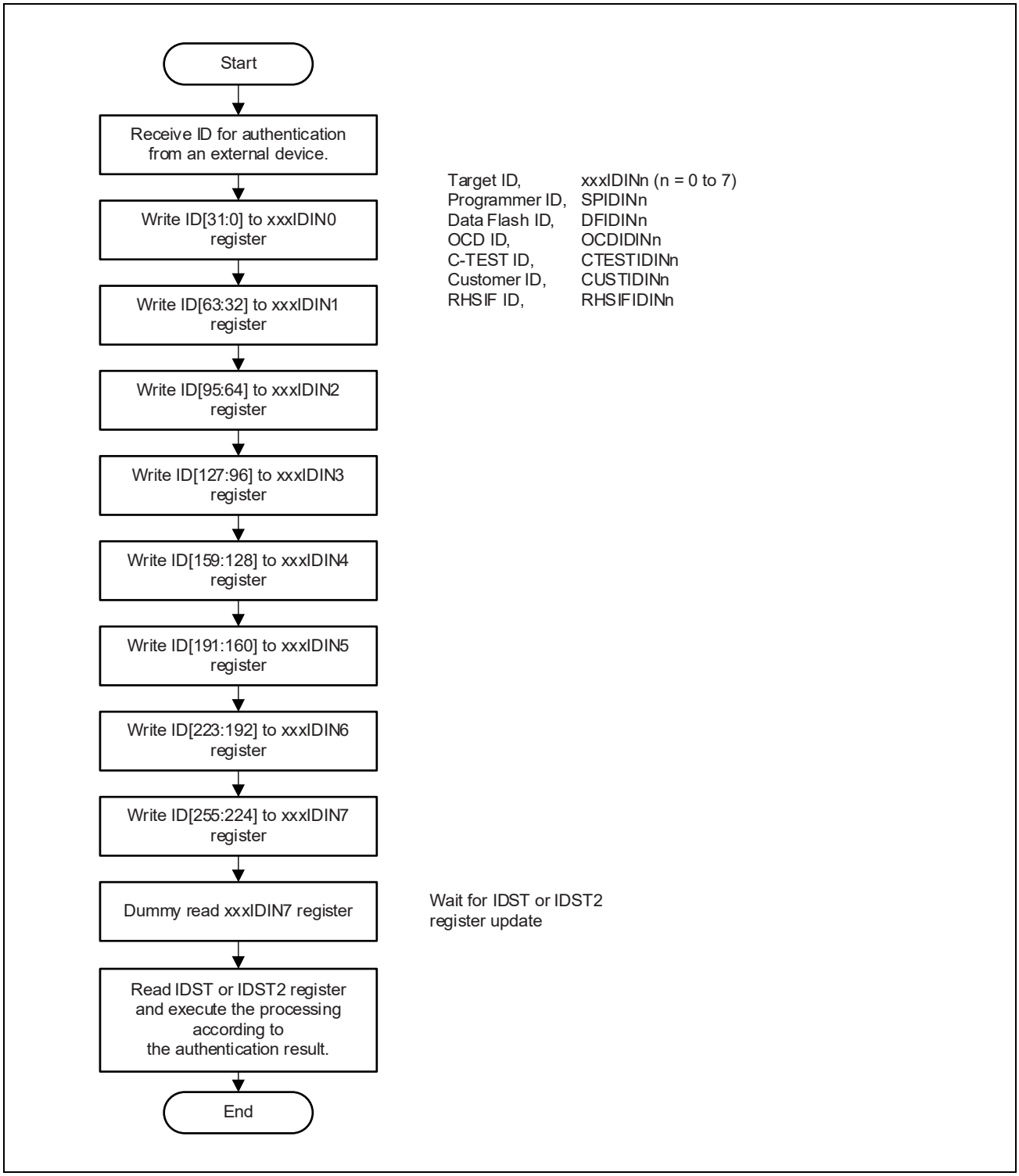


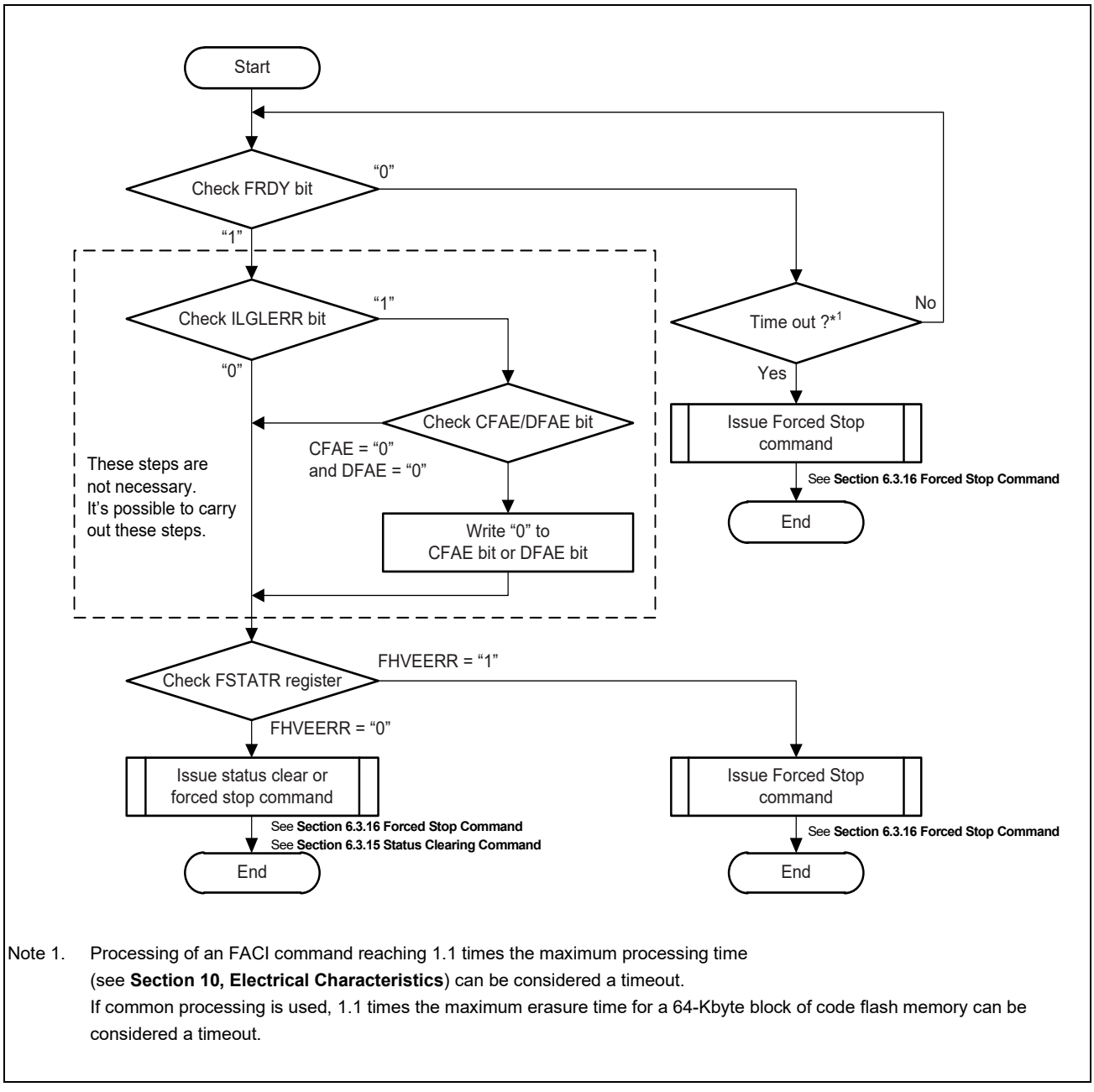

Figure 6.6 Flow of ID Authentication

### 6.3.7 Return from Command Lock State

When the flash sequencer enters the command lock state, FACI commands cannot be accepted. To release the command lock state, use the status clearing command, forced stop command, or FASTAT_n register.

When the command lock state is detected by checking an error before issuing the programming/erasure suspension command, the FRDY bit in the FSTATR_n register may hold 0 without completing the command processing. If the processing is not completed within the maximum programming/erasure time specified by electrical characteristics, it is determined as timeout and the flash sequencer should be stopped by the forced stop command.

The FHVEERR bit in the FSTATR_n register is not changed from 1 to 0 by the status clearing command. When this bit is set to 1, use the forced stop command to release the command lock state. The other bits to be the command lock source can be changed from 1 to 0 by the status clearing or forced stop command.




Note 1. Processing of an FACI command reaching 1.1 times the maximum processing time (see **Section 10, Electrical Characteristics**) can be considered a timeout. If common processing is used, 1.1 times the maximum erasure time for a 64-Kbyte block of code flash memory can be considered a timeout.

Figure 6.7 Return from Command Lock State

### 6.3.8 Issuing of Programming Command

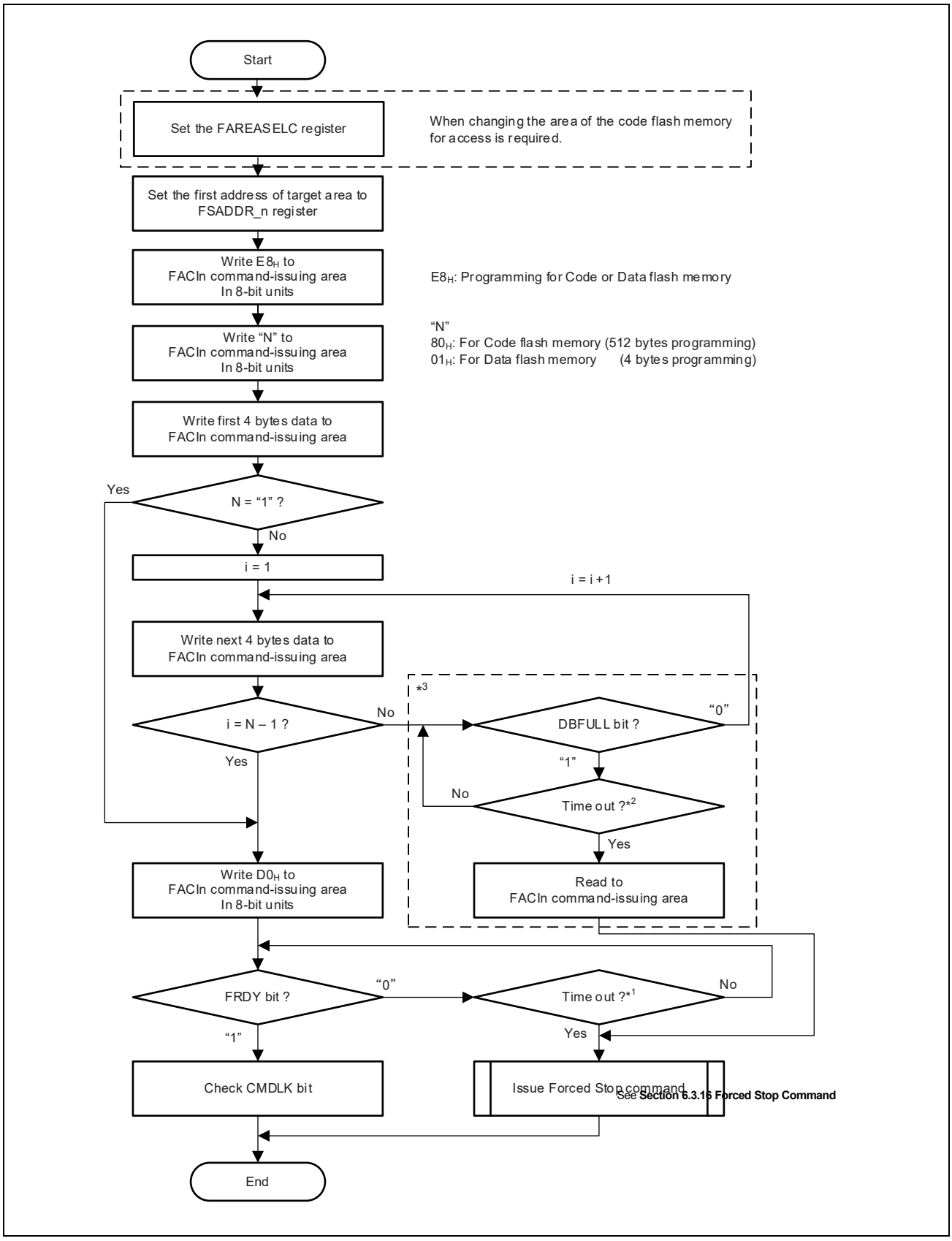
The programming command is used to write to user area, user boot area and data area.

Before issuing the programming command, set the first address of target area to the FSADDR_n register.

Writing $D0_H$ to the FACI command-issuing area at the final access of the FACI command-issuing starts the programming command processing. If the target area of programming command processing contains the area not for writing, write FFFF $FFFF_H$ to the corresponding area.

When using the CAN-FD/CAN bootstrap function in serial programming mode, to switch the target area of programming of the code flash memory, the setting of FAREASELC register must be changed. The FAREASELC register must be set before issuing the programming command.

If issuing the programming command is kept while the FACI internal data buffer is full, wait is generated in the peripheral-bus and it may affect the communication performance of other peripheral IPs. To avoid the wait generation, the DBFULL bit in FSTATR_n should be 0 when FACI commands are issued. In addition, writing to data area does not make the data buffer full.


Start
Set the FAREASELC register
When changing the area of the code flash memory for access is required.
Set the first address of target area to FSADDR_n register
Write E8H to FACIn command-issuing area In 8-bit units
E8H: Programming for Code or Data flash memory
Write "N" to FACIn command-issuing area In 8-bit units
"N"
80H: For Code flash memory (512 bytes programming)
01H: For Data flash memory (4 bytes programming)
Write first 4 bytes data to FACIn command-issuing area
N = "1" ?
Yes
No
i = 1
i = i +1
Write next 4 bytes data to FACIn command-issuing area
*3
i = N – 1 ?
No
Yes
DBFULL bit ?
"0"
"1"
No
Time out ?*2
Yes
Write D0H to FACIn command-issuing area In 8-bit units
Read to FACIn command-issuing area
FRDY bit ?
"0"
Time out ?*1
No
Yes
"1"
Check CMDLK bit
Issue Forced Stop command
See Section 6.3.16 Forced Stop Command
End

Note 1. For the code flash memory, the time of 512-byte programming reaching 1.1 times is judged as a timeout.
For the data flash memory, the time of 4-byte programming reaching 1.1 times is judged as a timeout.
If common processing is used, the time of 512-byte programming of the code flash memory reaching 1.1 times is judged as a timeout. (the maximum time: see **Section 10, Electrical Characteristics**)

Note 2. The time reaching 5 µs is judged as a timeout.

Note 3. DBFULL bit never become “1” when target is data flash, and these steps are not necessary for Data Flash program command. It’s possible to carry out these step.

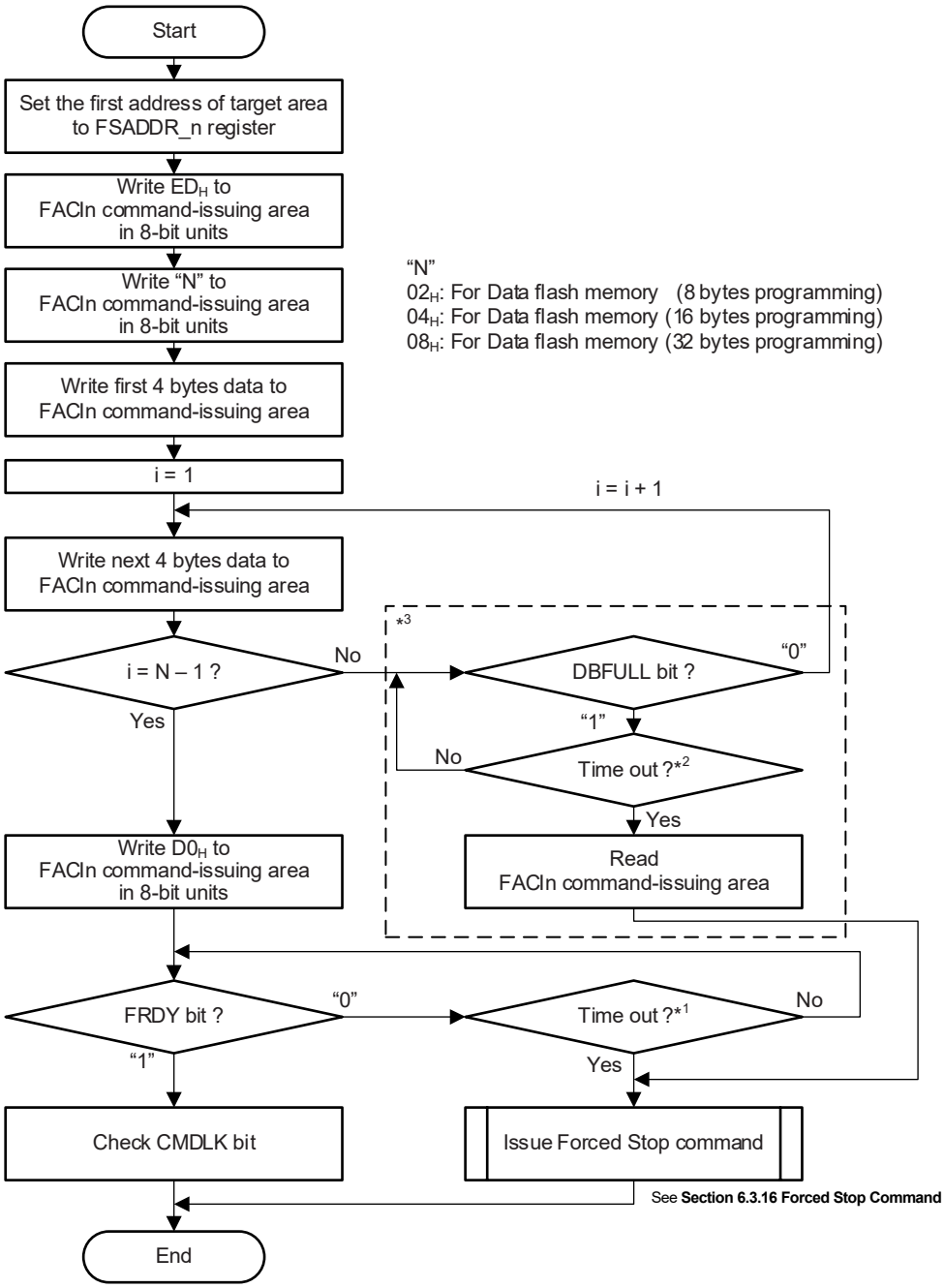
Figure 6.8 Programming Command Usage

### 6.3.9 Issuing of Multi Programming Command

The Multi programming command is used to write to data area.

Before issuing the Multi programming command, set the first address of target area to the FSADDR_n register.

Writing $D0_H$ to the FACI command-issuing area at the final access of the FACI command-issuing starts the programming command processing. If the target area of programming command processing contains the area not for writing, write FFFF $FFFF_H$ to the corresponding area.




Note 1. Processing of a Multi programming command reaching 1.1 times the maximum processing time can be considered a timeout. (the maximum time of Data Flash 8, 16 or 32 Bytes Programming time: See **Section 10, Electrical Characteristics**). If common processing is used, 1.1 times the maximum time for a 32-byte programming of data flash memory can be considered a timeout.

Note 2. The time reaching 5 µs is judged as a timeout.

Note 3. DBFULL bit never become "1" when target is data flash, and these steps are not necessary for Data Flash program command. It's possible to carry out these steps like Code Flash program command.

Figure 6.9 Multi Programming Command Usage

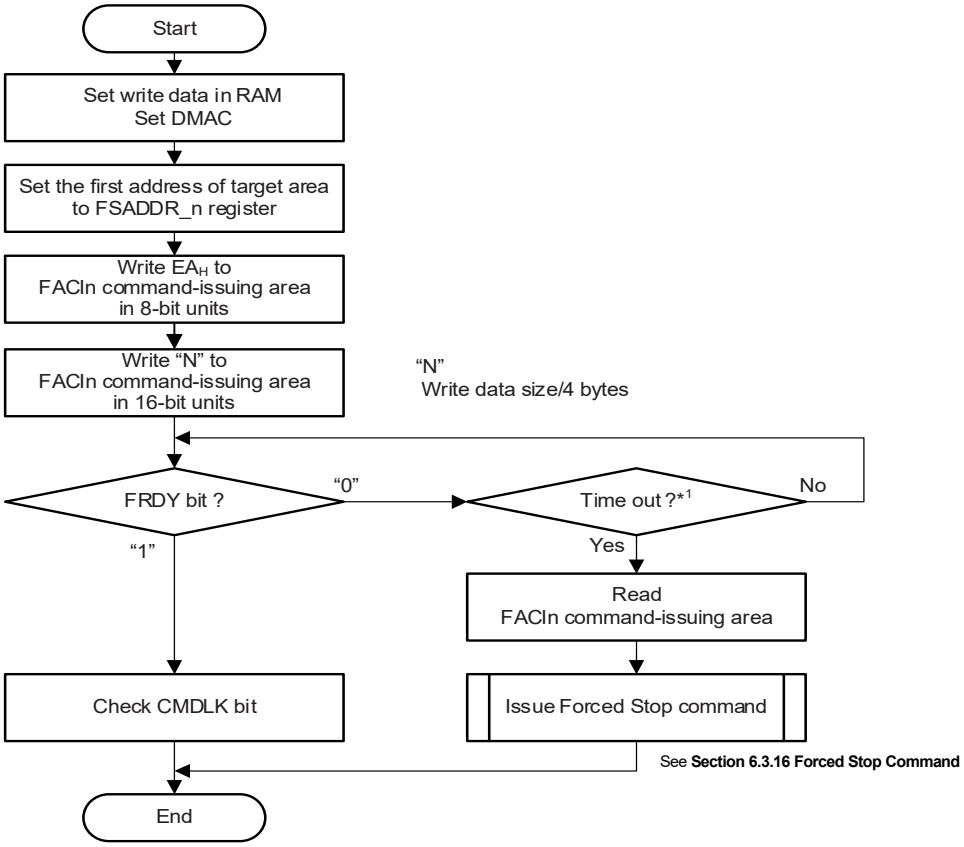
### 6.3.10 DMA Programming Command

The DMA programming command is used to program multiple 4-byte data transferred from the DMAC to the data area. Thus, a large amount of data can be programmed continuously with reduced CPU load.

Maximum program unit is limited within one Data Flash[Note 1]. Address boundary between different Data Flash cannot be crossed.

Before issuing a DMA programming command, set the first address of target area to the FSADDR_n register. Set the write data in the RAM and set the DMAC to perform DMA transfer from the pertinent area to the FACI command-issuing area. FACI issues a data transfer request to DMAC after reception of the DMA programming command and every time the writing of 4 bytes of data is completed. Set up the DMAC so that 4 bytes of data will be transferred for each data transfer request. For the usage of the DMAC, see *Section 7, DMA*, in *the User's Manual: Hardware*.

**Note 1.** The 96 KBytes Data Flash bank consists of two 48 KBytes Data Flash (EEP) called EEP0-0 and EEP0-1, or EEP1-0 and EEP1-1. 32 KBytes Data Flash bank for ICUM consists of one 32 Kbytes Data Flash called EEP2. These EEPs are treated as different Data Flash.




Note 1. Judged based on the sum of the time of 4-byte programming of the data flash memory (the maximum time: see **Section 10, Electrical Characteristics**) × "N" × 1.1 and the time required for DMA transfer of write data.

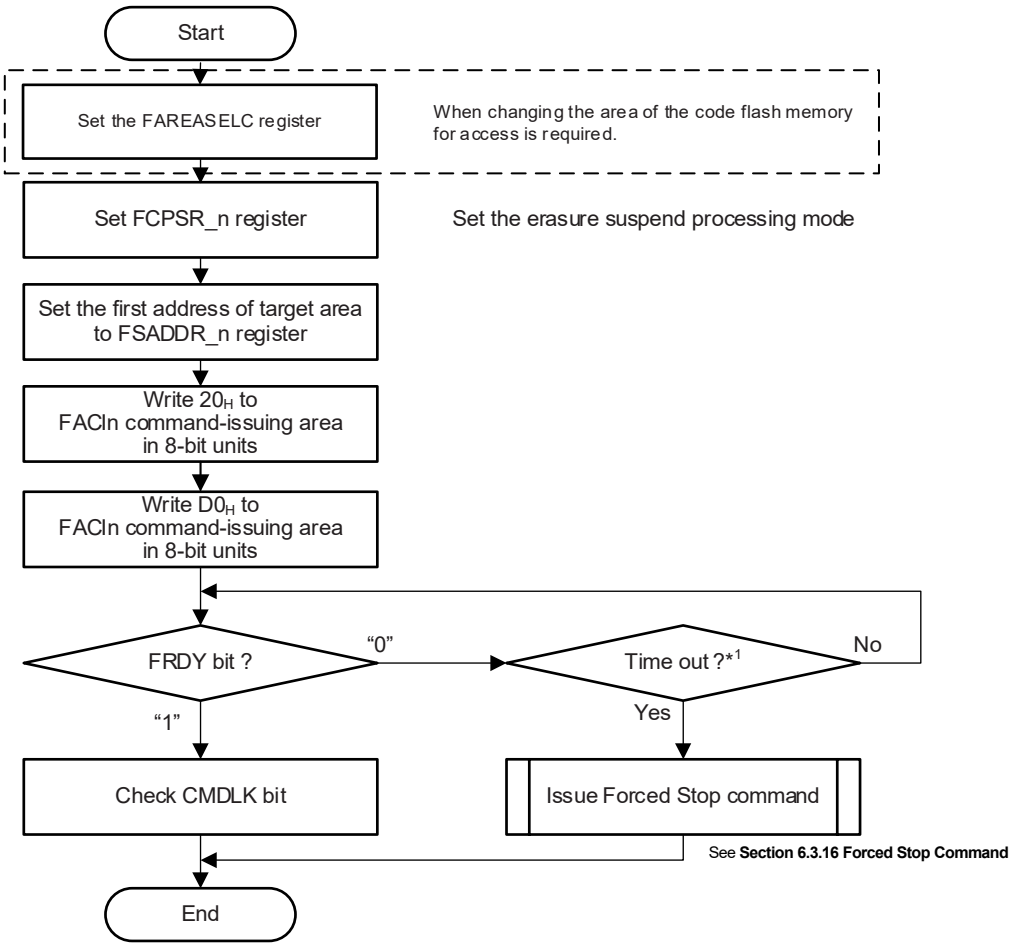
Figure 6.10 DMA Programming Command Usage

### 6.3.11 Block Erasure Command

The block erasure command is used to erase the user area, user boot area and data area. Before issuing the block erasure command, set the first address of target area to the FSADDR_n register. Writing $20_H$ and $D0_H$ to the FACI command-issuing area starts the block erasure command processing. Set the FCPSR_n register before issuing the block erasure command.

When using the CAN-FD/CAN bootstrap function in serial programming mode, to switch the target area of erasure of the code flash memory, the setting of FAREASELC register must be changed. The FAREASELC register must be set before issuing the block erasure command.

To set the FCPSR_n register is required to switch the suspending method by the programming/erasure suspension command (suspension-priority mode/erasure-priority mode).




Note 1. For the code flash memory, the time of erasure of a 64-Kbyte block reaching 1.1 times is judged as a timeout. For the data flash memory, the time of erasure of a 64-byte block reaching 1.1 times is judged as a timeout. If common processing is used, the time of erasure of a 64-Kbyte block of the code flash memory reaching 1.1 times is judged as a timeout. (the maximum time: see **Section 10, Electrical Characteristics**)

Figure 6.11 Block Erasure Command Usage

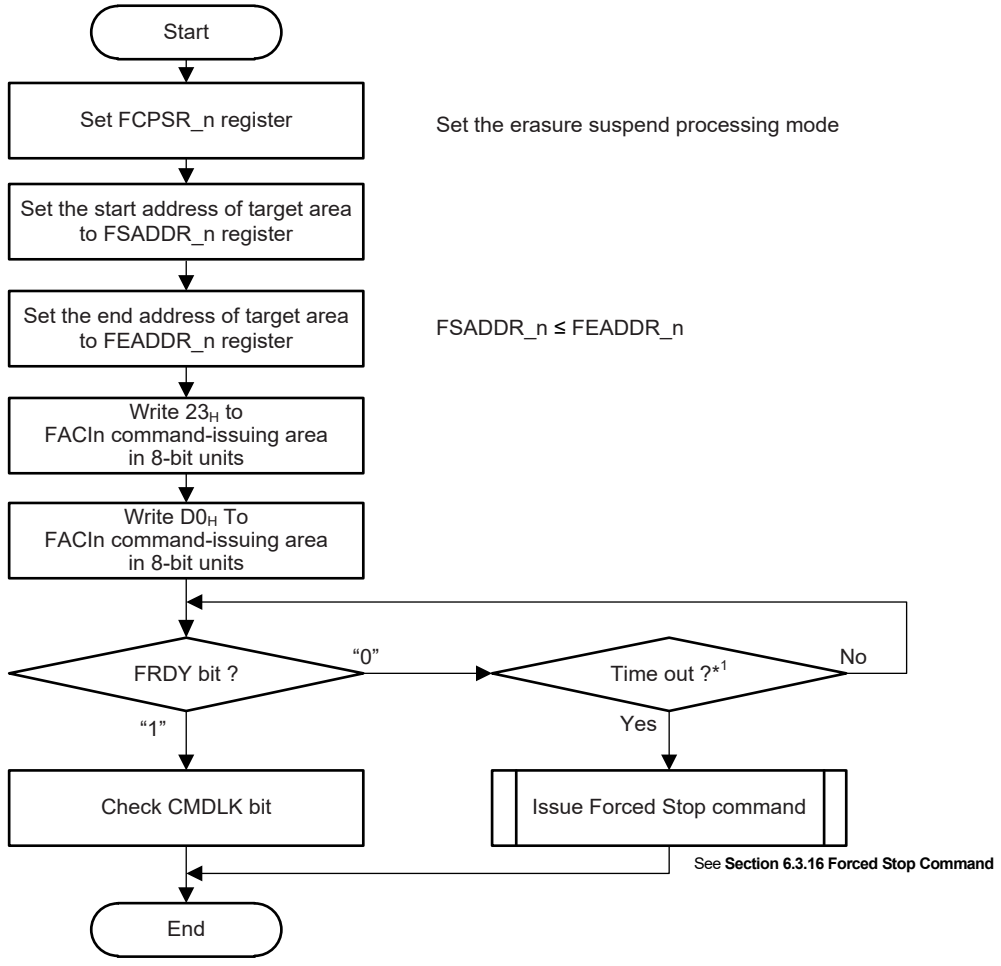
### 6.3.12 Area Erasure Command

The Area erasure command is used to erase the data area. Erase unit are N blocks (64 × N Byte). Before issuing the Area erasure command, set the first address of target area to the FSADDR register. Writing $23_H$ and $D0_H$ to the FACI command-issuing area starts the block erasure command processing. Area erasure command only performs within one Data Flash[Note 1].

If setting of FSADDR and FEADDR are inconsistent, FACI detects error and flash sequencer enters command lock state. Address boundary between different Data Flash cannot be crossed.

To set the FCPSR register is required to switch the suspending method by the programming/erasure suspension command (suspension-priority mode/erasure-priority mode).

**Note 1.** The 96 KBytes Data Flash bank consists of two 48 KBytes Data Flash (EEP) called EEP0-0 and EEP0-1, or EEP1-0 and EEP1-1. 32 KBytes Data Flash bank for ICUM consists of one 32 Kbytes Data Flash called EEP2. These EEPs are treated as different Data Flash.




Note 1. Judged based on the sum of the time of erasure of a 64-byte block erasure of data flash memory × N × 1.1. (the maximum time: see **Section 10, Electrical Characteristics**).

Figure 6.12 Area Erasure Command Usage

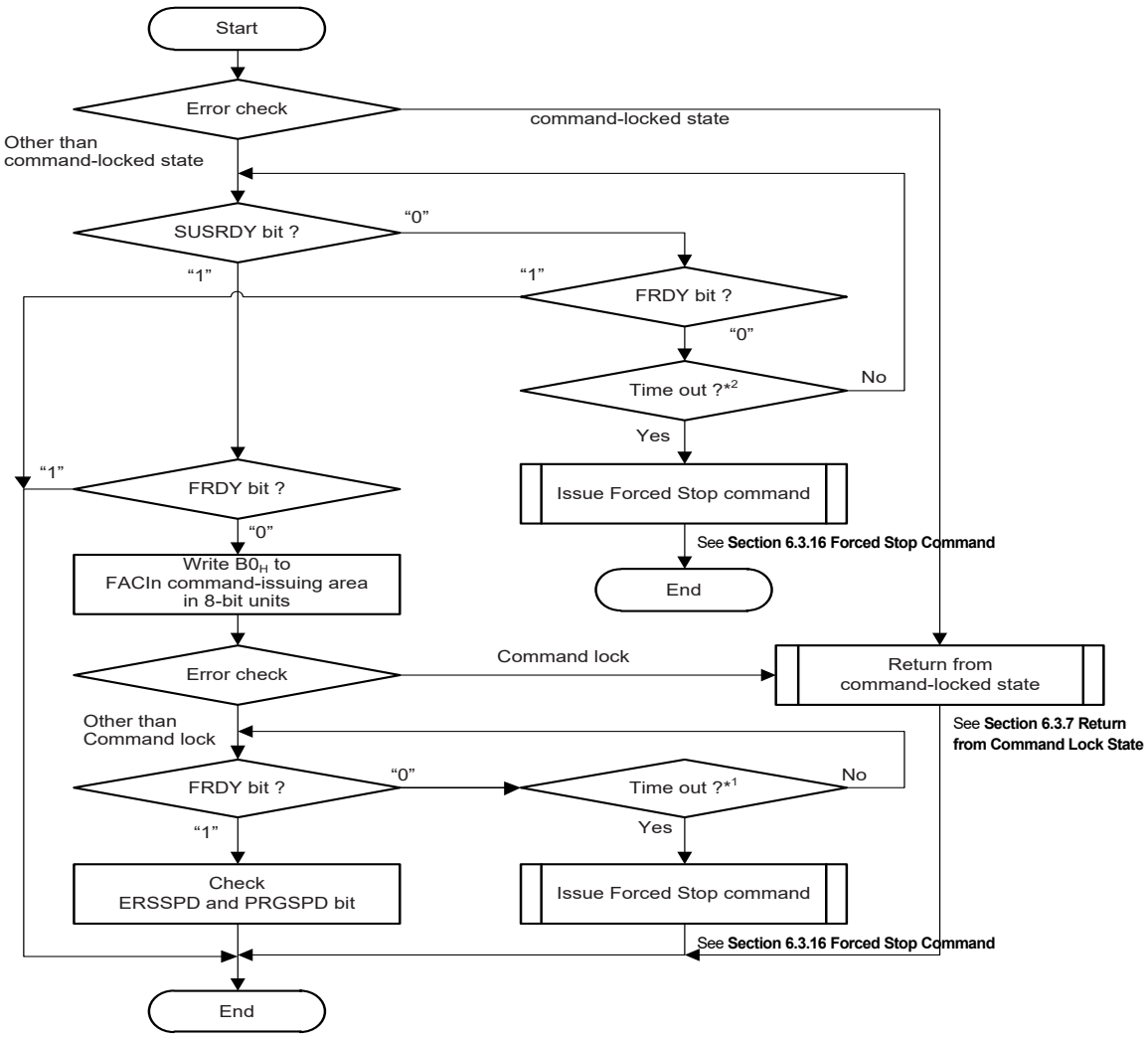
### 6.3.13 Programming/Erasure Suspension Command

The programming/erasure suspension command is used for suspending programming, Multi programming, Block erasure or Area erasure command processing. Before issuing the programming/erasure suspension command, check that CMDLK bit is 0 to ensure that programming, Multi programming, Block erasure or Area erasure command processing is being performed correctly. In addition, check that the SUSRDY bit is 1 to ensure that the programming/erasure suspension command is acceptable. After issuing the programming/erasure suspension command, check CMDLK bit to ensure no error has occurred.

If an error has occurred, the CMDLK bit is set to 1. If programming, Multi programming, Block erasure or Area erasure command processing is complete within the period from when the SUSRDY bit is ensured to be 1 until the programming/erasure suspension command is accepted, no error occurs, hence no transition to a suspended state (the FRDY bit is 1 and both ERSSPD and PRGSPD bits are 0).

Once the programming/erasure suspension command is accepted and programming, Multi programming, Block erasure or Area erasure command processing is normally suspended, flash sequencer enters a suspended state and that FRDY bit is 1 and ERSSPD or PRGSPD bit is 1. After issuing the programming/erasure suspension command and ensuring that flash sequencer has entered suspend state, determine which operation to perform in the succeeding process. If the programming/erasure resumption command is issued in the succeeding process while flash sequencer has not entered a suspended state, an illegal command error occurs and flash sequencer shifts to the command lock state (see **Section 8.2, Error Protection**).

When the operation shifts to the erasure suspend state, writing to the blocks other than those targeted for erasure is enabled. In addition, when the FENTRYR_n register is cleared, the operation shifts to the read mode.




Note 1. For the code flash memory, the suspend latency $t_{SEED}$ during erasure of the code flash memory reaching 1.1 times can be considered as a timeout.
For the data flash memory, the suspend latency $t_{SEED}$ during erasure of the data flash memory reaching 1.1 times can be considered as a timeout.
If common processing is used, 1.1 times $t_{SEED}$ (max.) of the code flash memory can be considered as a timeout. (the maximum time: see **Section 10, Electrical Characteristics**)

Note 2. The time reaching 16 µs is judged as a timeout.

Figure 6.13 Programming/Erasure Suspension Command Usage

### (1) Suspend programming command

If a programming/erasure suspension command is issued while the flash memory is being programmed, the flash sequencer suspends programming. **Figure 6.14** show the suspending operation. Once sequencer enters a state where it is ready to accept the programming/erasure suspension command after the start of programming, SUSRDY bit is set to 1. If the programming/erasure suspension command is issued, the flash sequencer accepts the command and clears SUSRDY bit. If the flash sequencer accepts the command while applying a programming pulse, the flash sequencer continues applying the pulse. After a specified pulse application time has elapsed, the flash sequencer completes applying the pulse, suspends programming, and sets the FSTATR.PRGSPD bit to 1. Once the suspension process is complete, the flash sequencer sets the FRDY bit to 1 and enters programming suspended state. If the flash sequencer accepts the programming/erasure resumption command in this state, the flash sequencer clears the FRDY and PRGSPD bits to 0 and restarts programming.

**Figure 6.14** gives an overview of operation for suspending programming command processing. Upon accepting programming command, sequencer clears FRDY bit to 0 and starts programming.

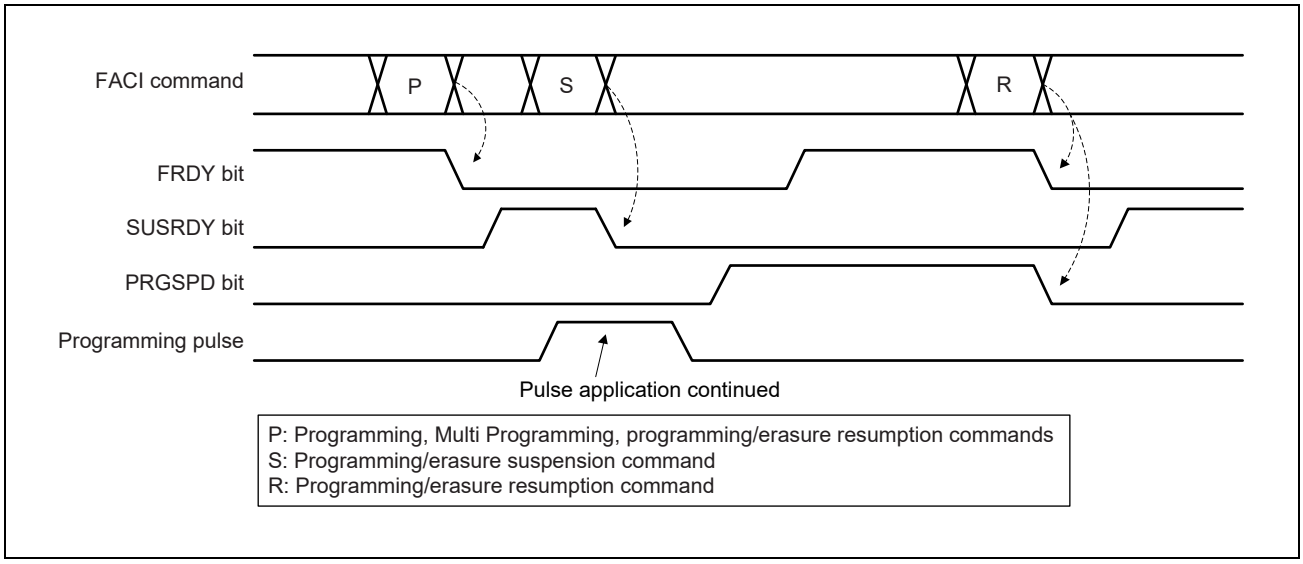



Figure 6.14 Suspend Programming Command

### (2) Suspend erasure command in suspension-priority mode

The flash memory supports the suspension-priority mode as one of the methods for suspending Block erasure or Area erasure command. **Figure 6.15** shows the operation for suspending Block erasure or Area erasure command processing in suspension-priority mode (FCPSR.ESUSPMD = 0). Upon accepting Block erasure or Area erasure command, sequencer clears the FRDY bit to 0 and starts erasing. Once sequencer enters a state where it is ready to accept the programming/erasure suspension command after the start of erasing, the SUSRDY bit is set to 1. If a programming/erasure suspension command is issued, sequencer accepts the command and clears the SUSRDY bit. If sequencer accepts interrupt request during its erasing operation, sequencer starts a suspending process even while applying a pulse and sets ERSSPD bit to 1. Once the suspending process completes, sequencer sets FRDY bit to 1 and enters erasing suspended state. If sequencer accepts programming/erasure resumption command in this state, sequencer clears FRDY and PRGSPD bits to 0 and restarts erasing. The operations of FRDY, SUSRDY, and ERSSPD bits are independent of the erasure-suspended mode.

The setting for erasure-suspended mode affects the control methods for an erasing pulse. In suspension-priority mode, if sequencer accepts interrupt request while applying an erasing pulse A, which has not been suspended previously, sequencer suspends the pulse application, and sequencer enters an erasure-suspended state. After sequencer resumes erasing by accepting a programming/erasure resumption command, sequencer accepts resume request while applying an erasing pulse A, sequencer continues applying the pulse. After a specified pulse application time has elapsed, sequencer completes applying the pulse, and sequencer enters an erasure-suspended state. Next, after sequencer accepts a programming/erasure resumption command, and sequencer starts applying a new pulse B, if sequencer accepts interrupt request, sequencer suspends the pulse application. In suspension-priority mode, delay due to suspension can be reduced because suspension process is given priority by suspending once every pulse application. In suspension-priority mode, if the interval of suspension after resume is longer than $t_{REST1}$ (Resume time: priority on suspension, resume after the 1st suspend for the same pulse), suspend latency will be always $t_{SESD1}$ (Suspend latency: priority on suspension, the 1st suspend for the same pulse).

If the interval of suspension after resume is shorter than $t_{REST1}$, suspend latency becomes either $t_{SESD1}$ or $t_{SESD2}$ (Suspend latency: priority on suspension, the 2nd suspend for the same pulse). In addition ,in code flash of this product, two parts work at the same time internally. The pulse is controlled independently each parts. Therefore, even if one side is the 1st suspend for the same pulse, another becomes the 2nd suspend for the same pulse, and the suspend latency may become $t_{SESD2}$.

(The value of $t_{REST1}$ / $t_{SESD1}$ / $t_{SESD2}$, see **Section 10, Electrical Characteristics**.)

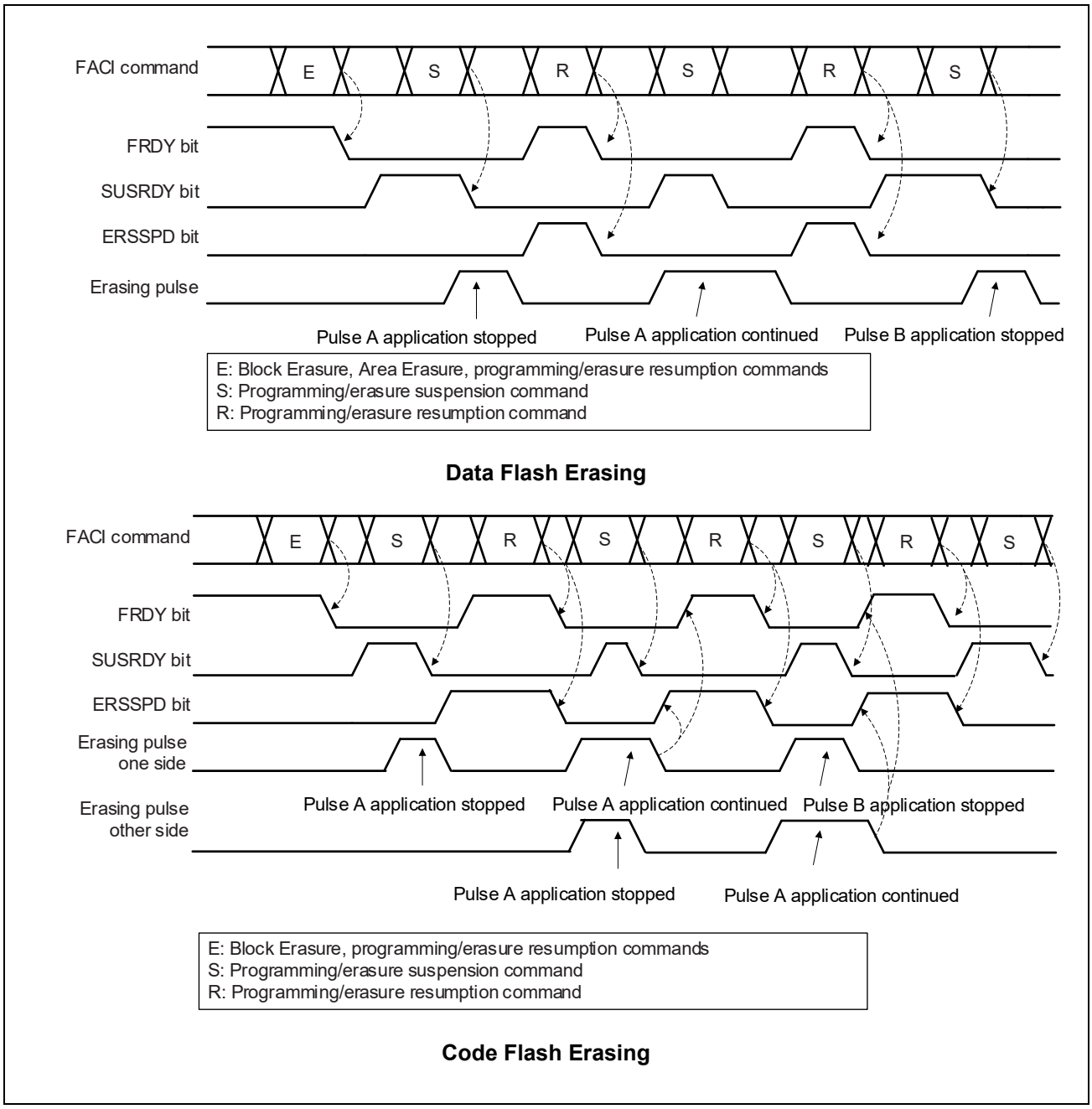



Figure 6.15 Suspend Erasure Command (Suspension-Priority Mode)

### (3) Suspend erasure command in erasure-priority mode

The flash memory supports the erasure-priority mode as one of the methods for suspending Block erasure or Area erasure command. **Figure 6.16** shows the operation for suspending Block erasure or Area erasure command processing in erasure-priority mode (FCPSR.ESUSPMD = 1). The operation for suspending Block erasure or Area erasure command processing in erasure-priority mode is equivalent to that for suspending programming processing.

In erasure-priority mode, if sequencer accepts a programming/erasure suspension command while applying an erasing pulse, sequencer always continues applying the pulse. As processing to reapply an erasing pulse never takes place in this mode, the total time required for Block erasure or Area erasure command processing is shorter than in suspension-priority mode.

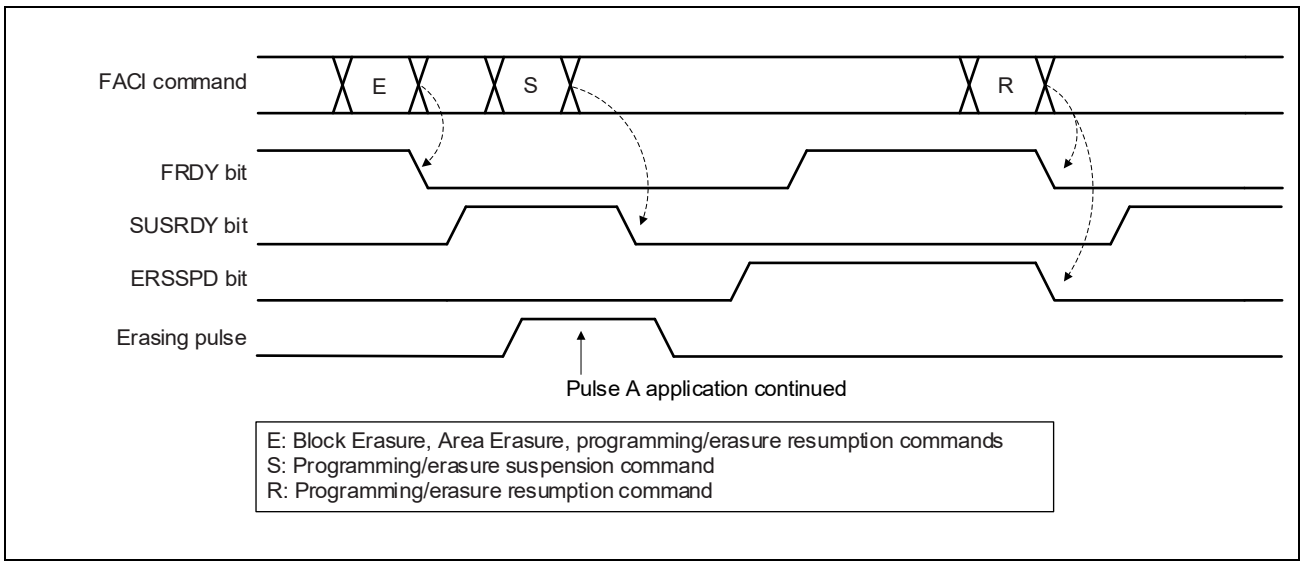
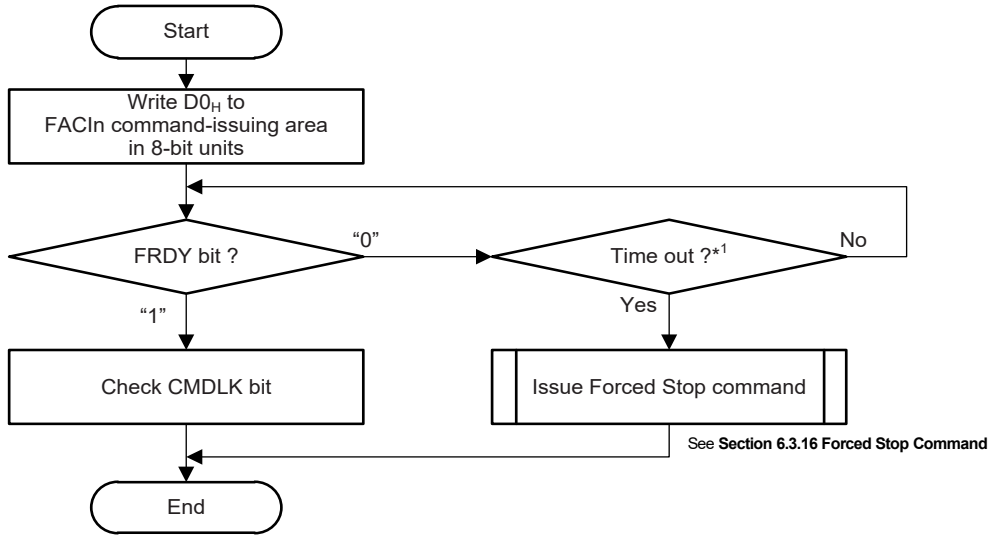



Figure 6.16 Suspend Erasure Command (Erasure-Priority Mode)

### 6.3.14 Programming/Erasure Resumption Command

The programming/erasure resumption command is used for resuming a programming, Multi programming, Block erasure or Area erasure command processing that has been suspended.

If the FENTRYR_n setting has been modified during suspension, issue a programming/erasure resumption command only after resetting FENTRYR_n to the previous value that was held before the programming/erasure suspension command was issued.




Note 1. Processing of an FACI command reaching 1.1 times the maximum processing time can be considered a timeout.
(the maximum time: see **Section 10, Electrical Characteristics**)
If common processing is used, 1.1 times the maximum erasure time for a 64-Kbyte block of code flash memory can be considered a timeout.

Figure 6.17 Programming/Erasure Resumption Command Usage

### 6.3.15 Status Clearing Command

The status clearing command is used to clear the command lock state. (See **Section 6.3.7, Return from Command Lock State**.) The specific bits in FASTAT and FSTATR register can be cleared by the status clearing command.

Refer to **Section 4.3, FASTAT_n — Flash Access Status Register (n = 0,1)** and **Section 4.8, FSTATR_n — Flash Status Register (n = 0,1)** for register bits cleared by the status clearing command.

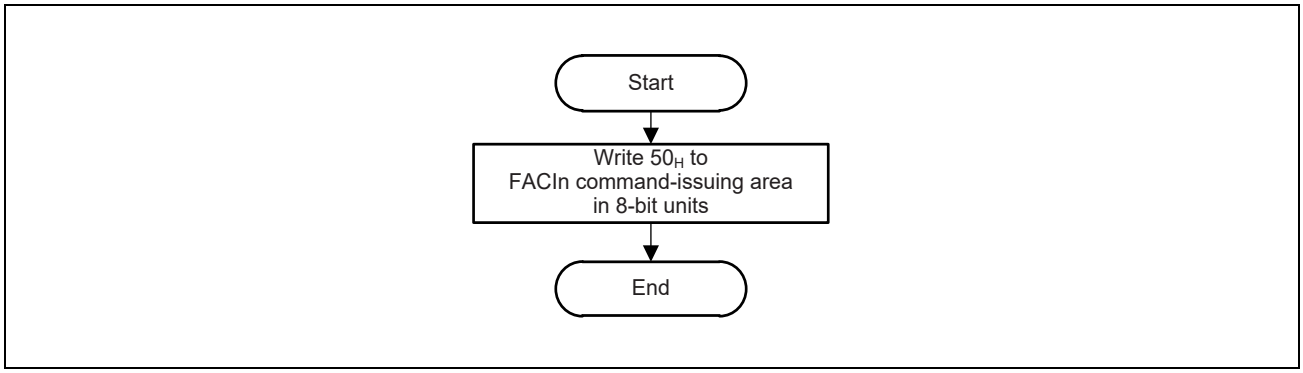
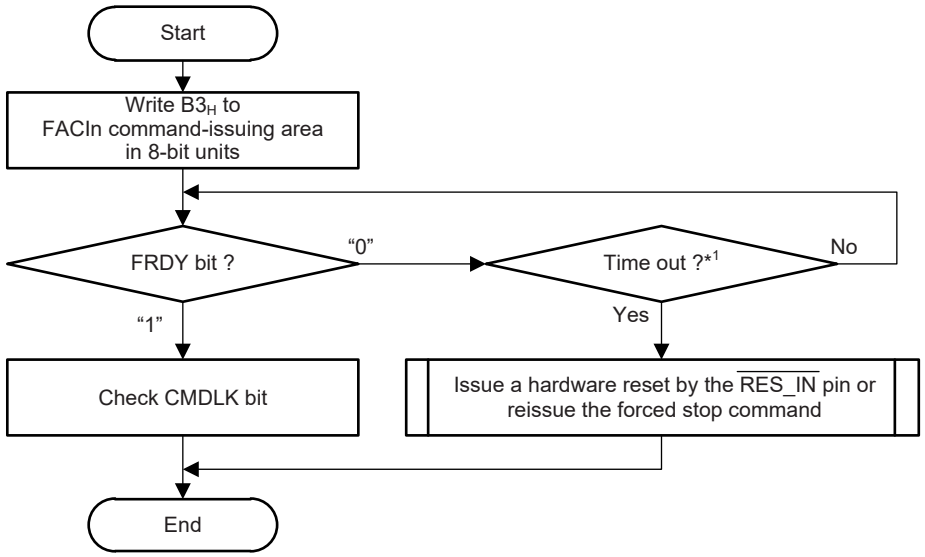



Figure 6.18 Status Clearing Command Usage

### 6.3.16 Forced Stop Command

The forced stop command is used to abort the command being processed by a flash sequencer. While the processing speed of this command is faster than that of programming/erasure suspension commands, it does not guarantee any result of the stopped command operation such as data in programmed or erased area. Furthermore, it is not possible to resume the suspended processing later. The suspended programming or erasure is counted as one from the perspective of programming endurance.

When a forced stop command is issued, the whole FCU and a part of FACIn are initialized as well as the FASTAT_n and FSTATR_n register. This enables forced stop command in recovery from a command lock state or during handling of timeout in flash sequencer operation mode. (See **Section 6.3.7, Return from Command Lock State**.)




Note 1. The forced stop command latency reaching 1.1 times is judged as a timeout.
(the maximum time: see **Section 10, Electrical Characteristics**)

Figure 6.19 Forced Stop Command Usage

Issuing the Forced Stop Command while Another Command is Being Issued

If the forced stop command is used to suspend processing when a timeout of the programming command occurs when checking the DBFULL bit, when a timeout for injection of ECC errors occurs when checking the DBFULL or EBFULL bit, or when a timeout of the DMA programming command occurs, writing to the FACI command issuing area may be handled as writing of data by the programming command. If this is the case, read the FACI command issuing area to intentionally lock commands and issue the forced stop command by following the procedure for returning from the command-locked state.

Locking commands is possible in any case where the unit for reading the FACI command issuing area is 8, 16, or 32 bits.

### 6.3.17 Blank Checking Command

The values of the data flash memory in which no data are programmed after erasure (non-programmed state) are undefined. Thus, the blank checking command is required to confirm the non-programmed state. For how to confirm the non-programmed state of the code flash memory, see **Section 8.4, Blank Checking of Code Flash Memory**.

Before issuing the blank checking command, set addressing mode, start address, and end address to FBCCNT_n, FSADDR_n, and FEADDR_n register, respectively. When blank check addressing mode is set to incremental mode (i.e. FBCCNT_n.BCDIR = 0), address specified in FSADDR_n should be equal to or smaller than address in FEADDR_n. Conversely, address in FSADDR_n should be equal to or larger than address in FEADDR_n when blank check addressing mode is set to decremental mode (i.e. FBCCNT_n.BCDIR = 1).

If setting of BCDIR, FSADDR_n, and FEADDR_n are inconsistent, FACI detects error and flash sequencer enters command lock state.

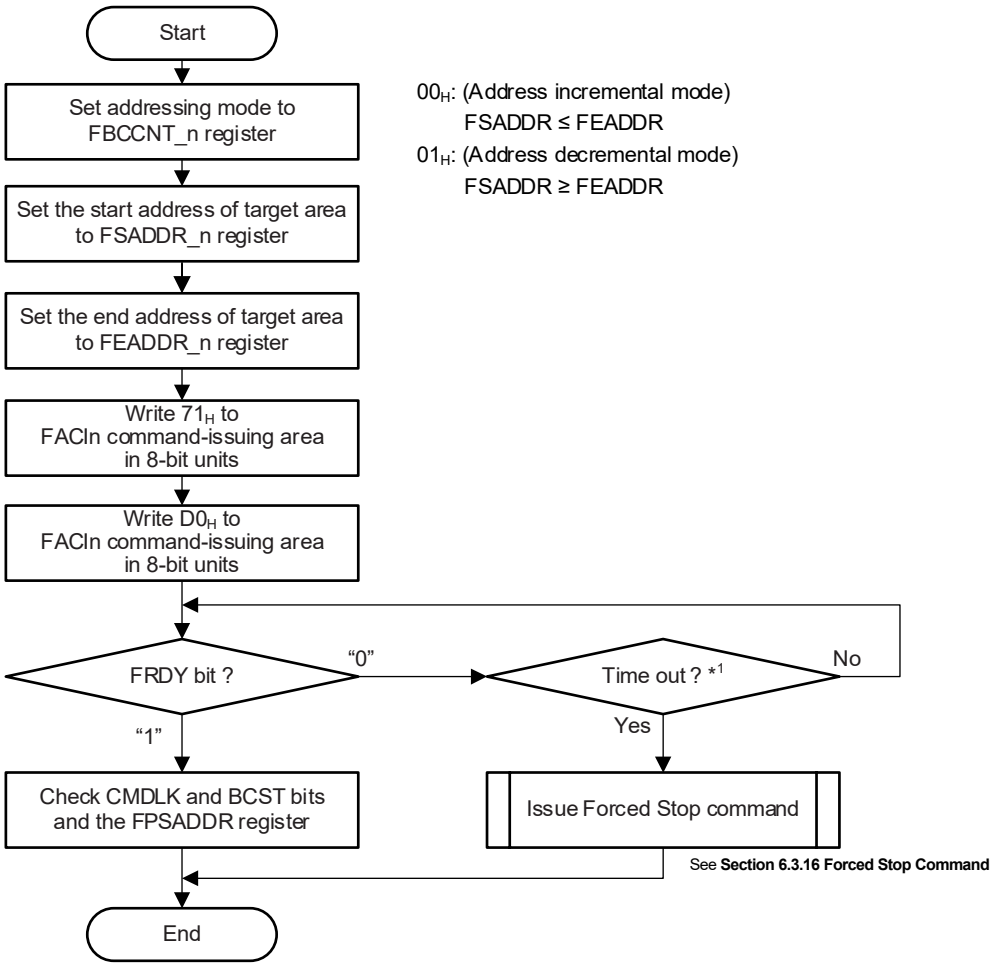
Blank checking command only performs within one Data Flash[Note 1]. Then, make sure to set blank check area not crossing boundary of different Data Flash.

Write $71_H$ and $D0_H$ to the FACI command-issuing area to start blank checking command processing.

Completion of command processing can be confirmed by FRDY bit of FSTATR_n register. At the end of processing, the result of blank checking is stored in the BCST bit in the FBCSTAT_n register. If non-blank data exists within blank checked area, flash sequencer stops blank checking operation. In this case, address of non-blank data is indicated to FPSADDR_n register.

The erasure state can be checked by this command only for an area for which erasure processing has been correctly completed. If erasure is not correctly completed (for example, due to a reset input or power shutdown), the erasure state cannot be checked by this command.

**Note 1.** The 96 KBytes Data Flash bank consists of two 48 KBytes Data Flash (EEP) called EEP0-0 and EEP0-1, or EEP1-0 and EEP1-1. 32 KBytes Data Flash bank for ICUM consists of one 32 Kbytes Data Flash called EEP2. These EEPs are treated as different Data Flash.




Note 1. The blank checking time reaching 1.1 times is judged as a timeout.
(the maximum time: see **Section 10, Electrical Characteristics**)

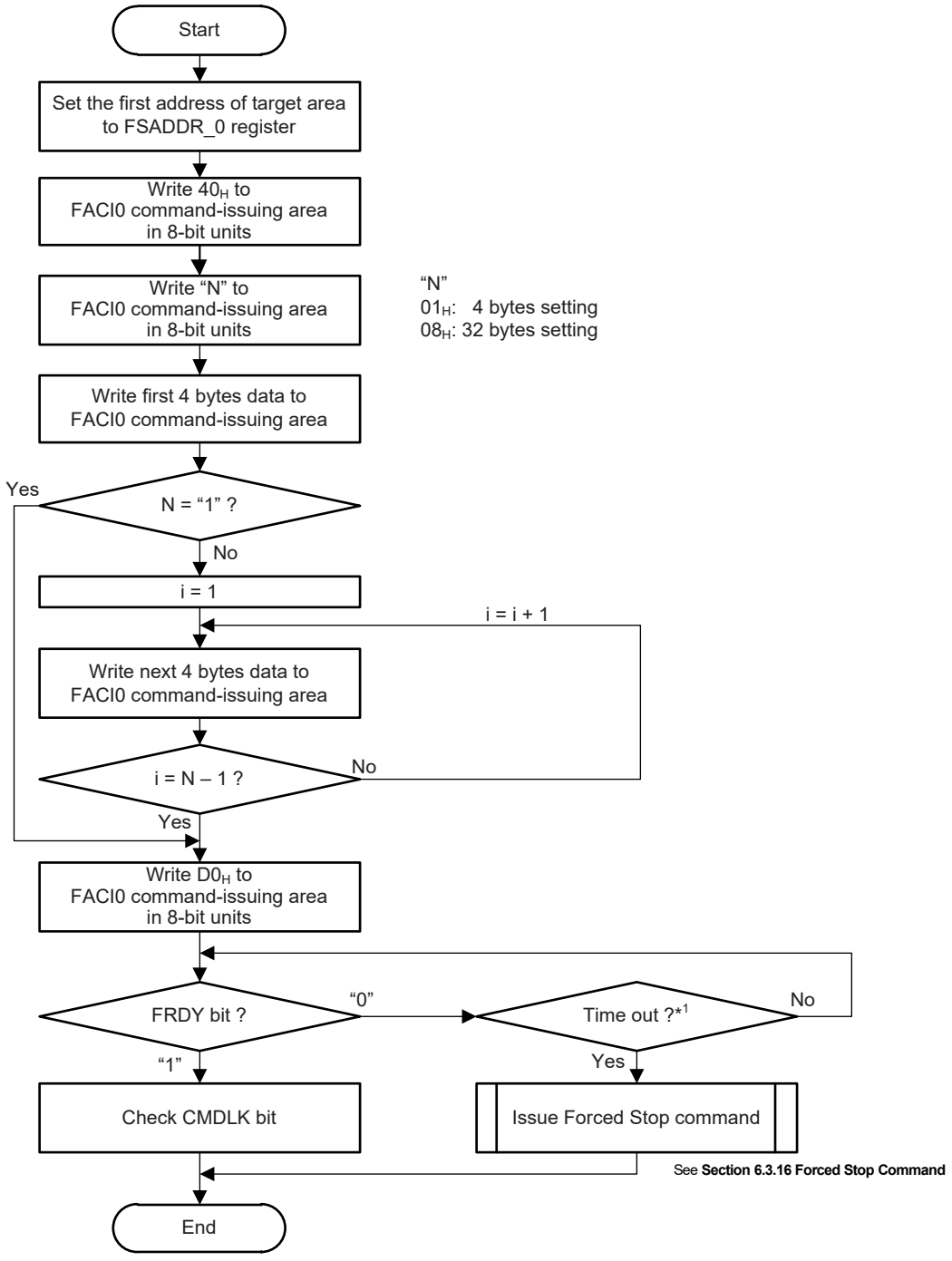
Figure 6.20 Blank Checking Command Usage

### 6.3.18 Setting of Configuration Settings Command

The Setting of configuration Settings command is used for the change of the data in the Configuration Setting Area. (Flash Option Byte, OTP settings for each 4 bytes of Configuration Setting Area.)

The Setting of configuration Settings command can renew data by 4-bytes or 32-bytes unit.

Writing $D0_H$ to the FACI0 command-issuing area at the final access of the FACI command issue starts the setting of configuration settings command processing.




Note 1. The Setting of Configuration settings maximum time of reaching 1.1 times is judged as a timeout.
(the maximum time: see **Section 10, Electrical Characteristics**)

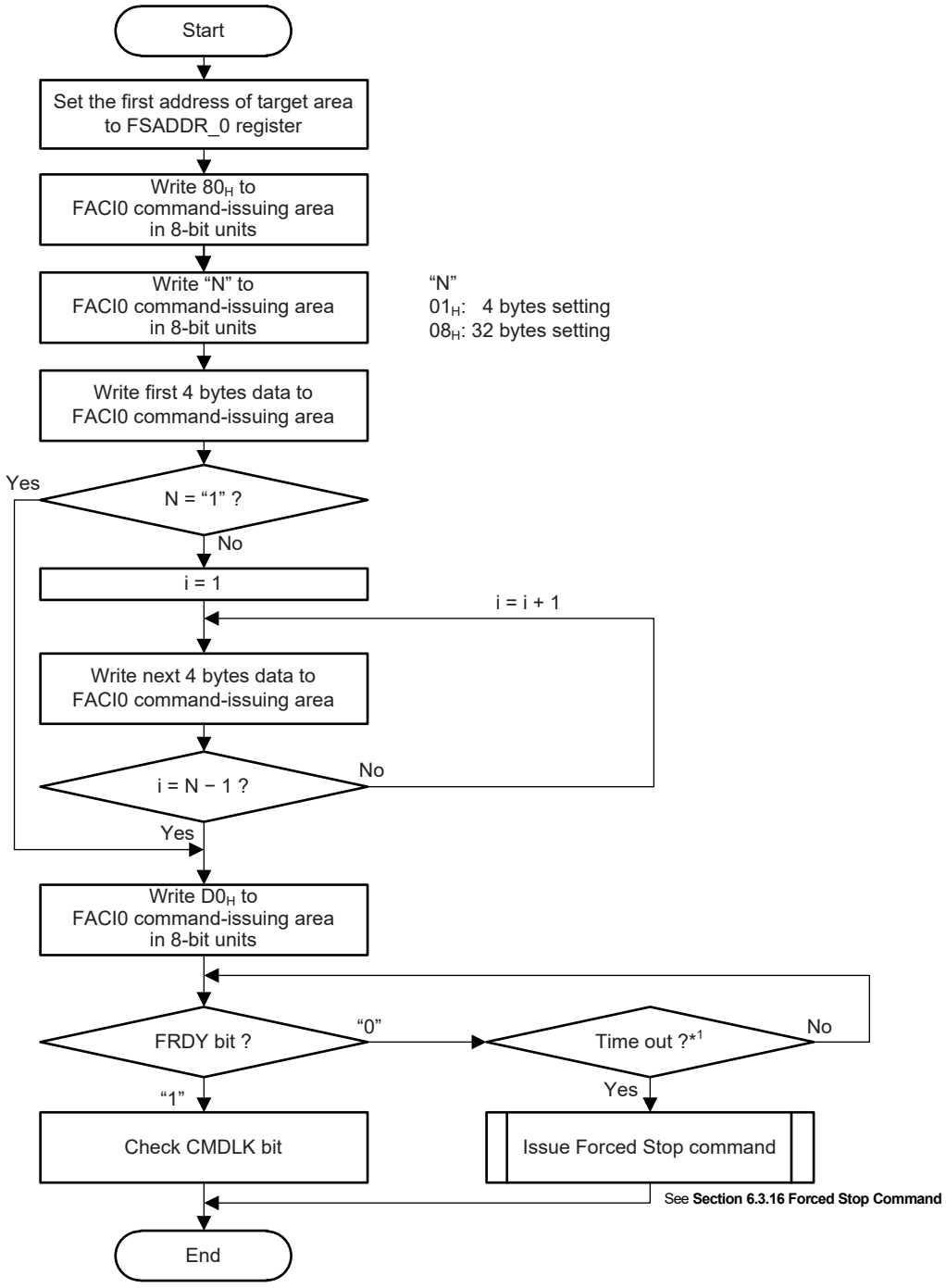
Figure 6.21 Setting of Configuration Settings Command Usage

### 6.3.19 Setting of Block Protection Settings Command

The Setting of Block Protection Settings command is used for the change of the data in the Block Protection Setting Area. (OTP Settings for each block in user area and user boot area, Block Protection settings, etc.)

The Block Protection Settings command can renew data by 4-bytes or 32-bytes unit.

Writing $D0_H$ to the FACI0 command-issuing area at the final access of the FACI command issue starts the setting of Block Protection settings command processing.




Note 1. The Setting of Block Protection settings maximum time of reaching 1.1 times is judged as a timeout.
(the maximum time: see **Section 10, Electrical Characteristics**)

Figure 6.22 Setting of Block Protection Settings Command Usage

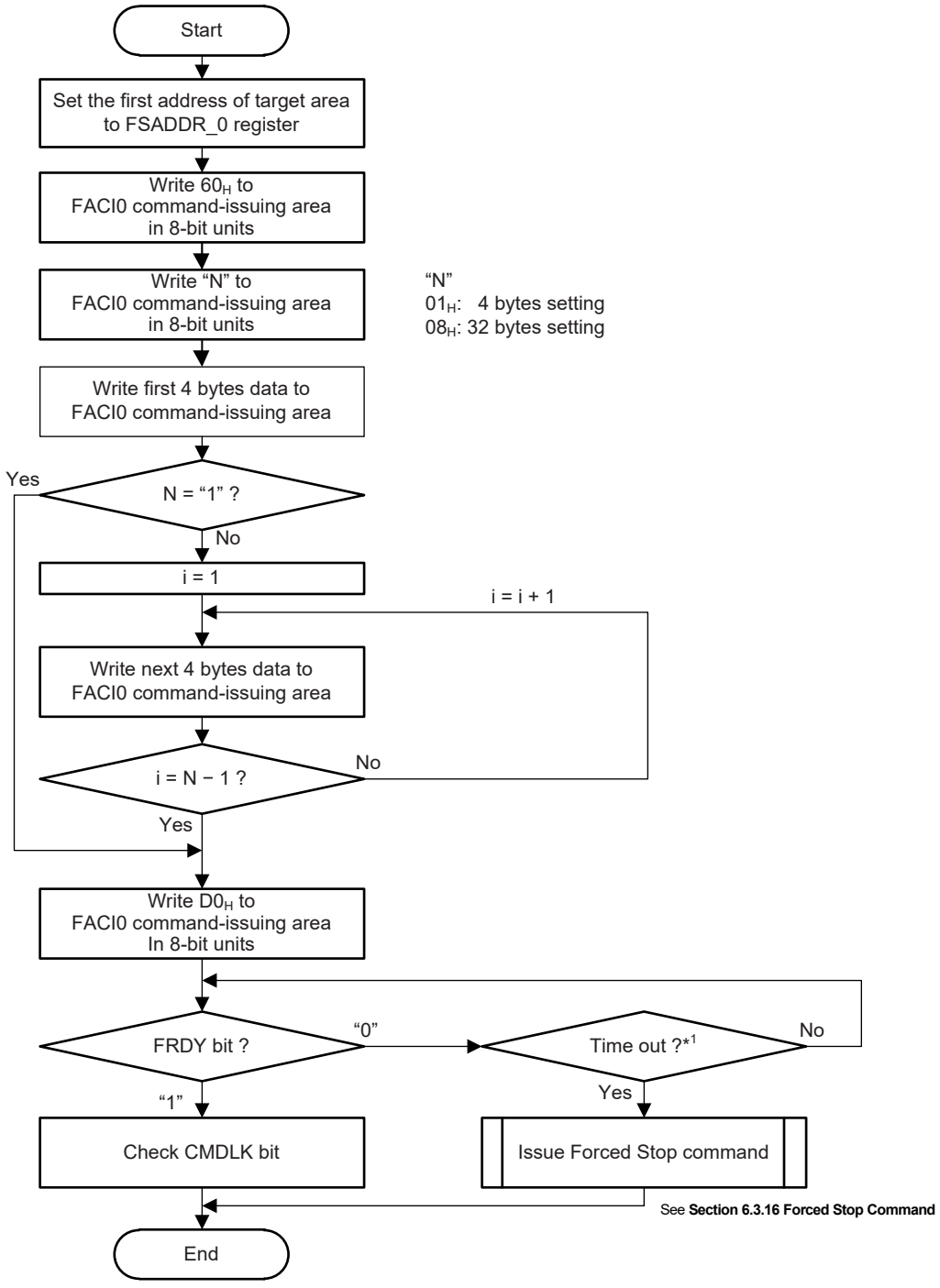
### 6.3.20 Setting of Security Settings Command

The Setting of Security Settings command is used for the change of the data in the Security Setting Area. (Each ID's, Each ID's related Option Byte, OTP settings for each 4bytes of Security Setting Area.)

The Setting of Security Settings command can update data by 4-byte or 32-byte unit.

In a case of ID update, 32-byte unit should be used in order to avoid setting unintentional ID. For updating data except for ID in the Security Setting Area, 4-byte or 32-byte unit can be used.

Writing $D0_H$ to the FACI0 command-issuing area at the final access of the FACI command issue starts the setting of Security settings command processing.




Note 1. The Setting of Security settings maximum time of reaching 1.1 times is judged as a timeout.
(the maximum time: see **Section 10, Electrical Characteristics**)

Figure 6.23 Setting of Security Settings Command Usage

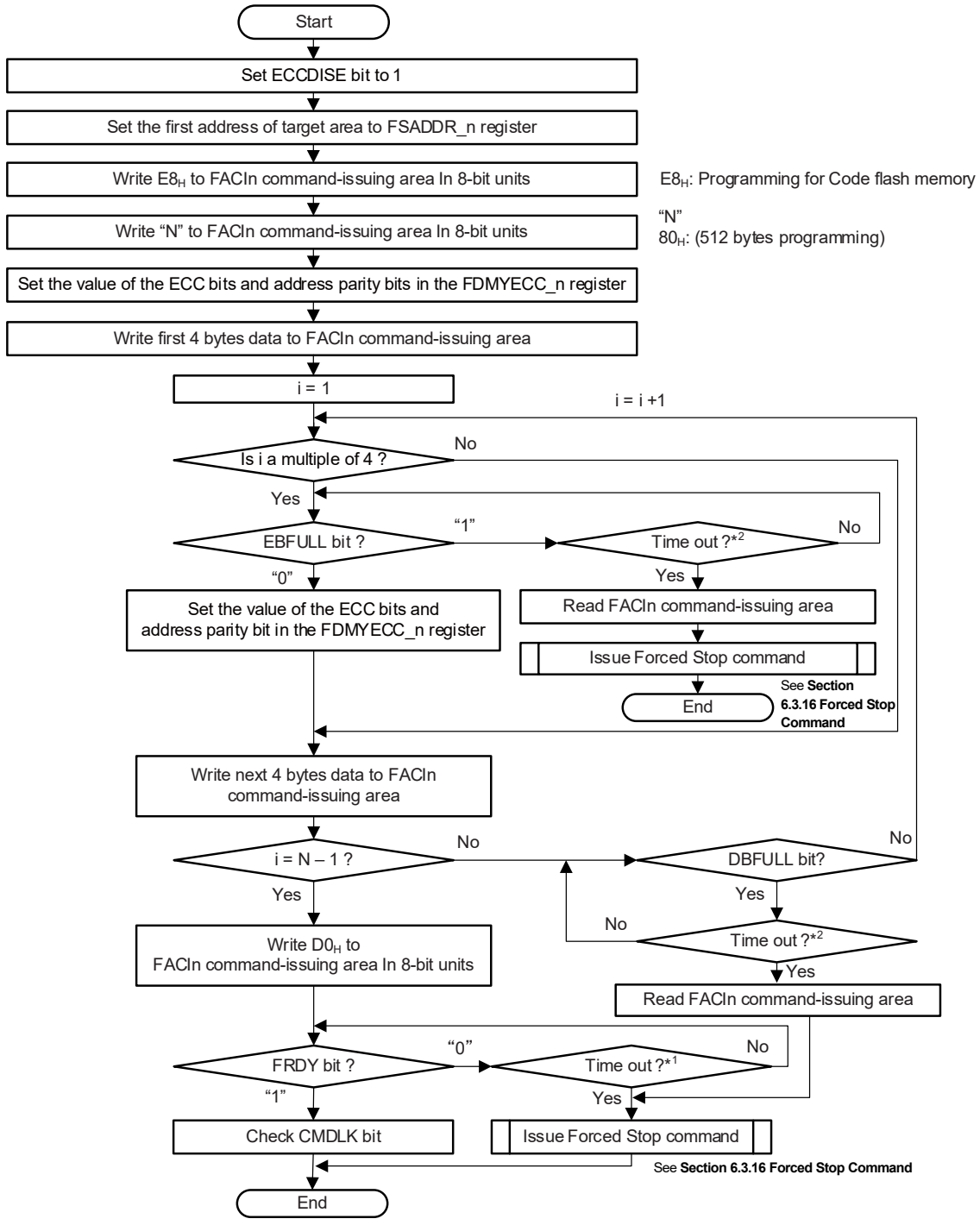
### 6.3.21 Injecting ECC Errors for the Flash Memory

Any value of the ECC bits and address parity bits in the FDMYECC_n register can be written to the flash memory by using a programming command. The function of injecting ECC error to data area can be used only by a programming command in 4-byte units.

Before writing the value set in the FDMYECC_n register to the flash memory, set the ECCDISE bit in the FECCTMD_n register to 1. In addition, set the values for the ECC bits and address parity bit in the FDMYECC_n register before writing the data to the FACI command-issuing area. In the case of the code flash memory, the unit (512 bytes) for writing in response to the programming command differs from the unit (16 bytes) for which the ECC bits and address parity bit are to be added for the data.

Therefore, every time 16 bytes of data are written to the FACI command-issuing area, change the setting in the FDMYECC_n register. In the case of the data flash memory, since the unit (4 bytes) for writing by the programming command is the same as that for the unit (4 bytes) of data for which the ECC bits are to be added, only change the setting in the FDMYECC_n register once before issuing the programming command.

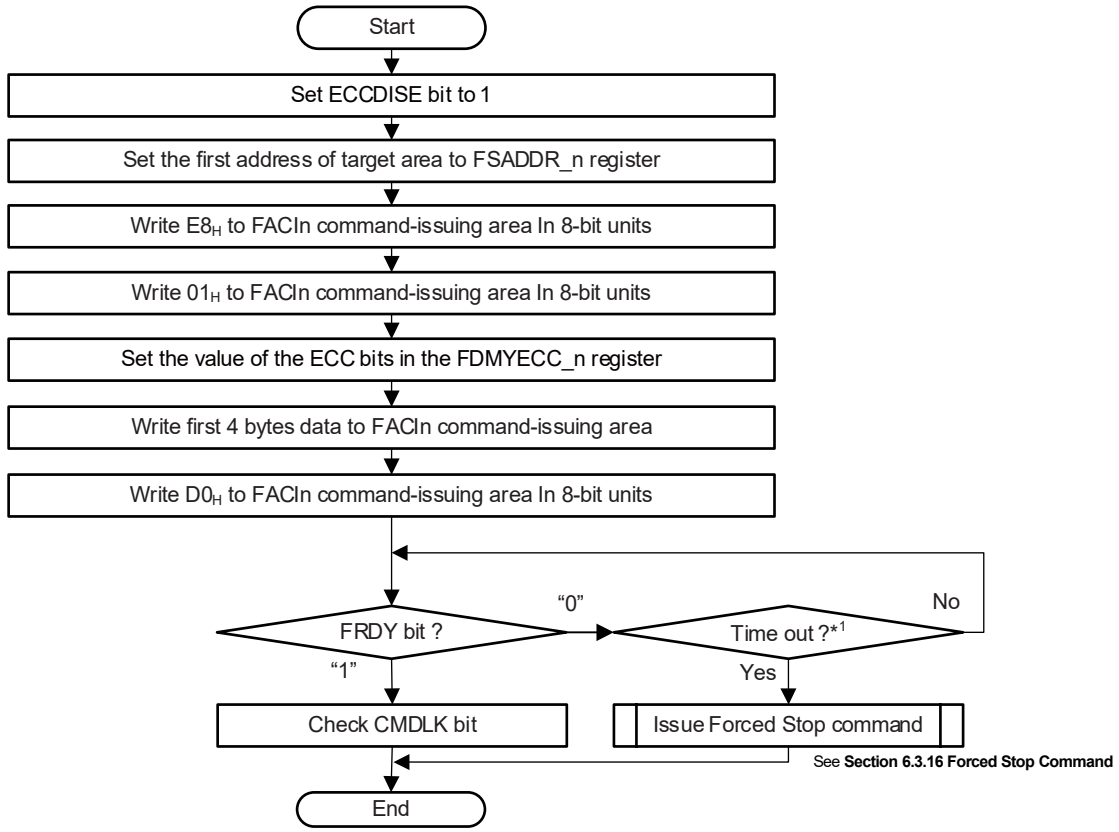
Issuing the command for writing to the FDMYECC_n register while the EBFULL bit in the FSTATR_n register is 1 may lead to a wait being generated on the peripheral bus, which will affect performance in communication with other peripheral IP modules. To avoid the generation of such a wait, write to the FDMYECC_n register while the EBFULL bit in the FSTATR_n register is 0.




Note 1. The time of 512-byte programming (the maximum time: see **Section 10, Electrical Characteristics**) reaching 1.1 times is judged as a timeout.

Note 2. The time reaching 5 µs is judged as a timeout.

Figure 6.24 Injecting an ECC Error for the Code Flash Memory




Note 1. The time of 4-byte programming (the maximum time: see **Section 10, Electrical Characteristics**) reaching 1.1 times is judged as a timeout.

Figure 6.25 Injecting an ECC Error for the Data Flash Memory

# Section 7 Security Function

To protect the code flash memory, data flash memory and Flash extra area, this product has several Flash protection functions.

For detailed of security function, see *Section 42, Basic Hardware Protection (BHP)*, in *the User's Manual: Hardware*.

## 7.1 Protection by ID Authentication

Code flash memory (user area / user boot area) can be protected from programming/erasure by Customer ID authentication (Block Protection). Each code flash block has 4-bit protection setting in Block Protection Setting Area.

Data flash memory (data area) can be protected from programming/erasure by Data Flash ID authentication (Block Protection). This protection is enabled in Flash Option Byte (DPROT).

Configuration Setting Area and Security Setting Area are protected by each associated ID.

If FACI command is issued to target address that was protected, the flash sequencer enters the command lock state.

As for the security releasing method by the ID authentication, see **Section 6.3.6, ID Authentication**.

## 7.2 OTP Function

OTP can be set independently for each block in the code flash memory (user area and user boot area).

OTP can be set independently for each 4bytes of Flash Option Byte in Security Setting Area and Configuration Setting Area.

OTP can be set independently for each 4bits of Customer ID Block Protection settings in Block Protection Area.

Once an OTP is set, it cannot be canceled.

If a Programming or Block Erasure command is issued to a OTP protected block, the flash sequencer enters the command lock state (Occur ILGLERR).

If a Setting of Configuration Settings or Setting of Security Settings command is issued to an address including a OTP protected Flash Option Byte, the flash sequencer does not enter the command lock state (Not Occur ILGLERR).

The OTP protected Flash Option Byte is not updated, but an unprotected Flash Option Bytes located in the same setting unit is updated.

If a Setting of Block Protection Settings command is issued to an address including a OTP protected Customer ID Block Protection setting, the flash sequencer does not enter the command lock state (Not Occur ILGLERR).

The OTP protected setting is not updated, but an unprotected Customer ID Block Protection setting located in the same setting unit is updated.

# Section 8 Protection Function

## 8.1 Software Protection

Software protection function disables flash sequencer command operation according to register settings. If an attempt is made to issue flash sequencer command against software protection, flash sequencer enters command lock state.

### 8.1.1 Protection by FENTRYR

When FENTRYR_n register is set to $0000_H$, flash sequencer is set to read mode. In read mode, FACI commands cannot be accepted.

If an attempt is made to issue FACI command in read mode, flash sequencer enters command lock state.

## 8.2 Error Protection

Error protection function detects an illegal FACI command issued, an illegal access, or a flash sequencer malfunction, and disables FACI command acceptance (command lock state). While flash sequencer is in the command lock state, flash memory cannot be programmed or erased. To cancel command lock state, issue status clearing or forced stop command. Status clearing command can be used only when FRDY bit is 1. Forced stop command can be used regardless of FRDY bit value. While the CMDLKIE bit in FAEINT_n register is 1, a flash access error (FLERR) interrupt is generated if flash sequencer enters command lock state (the CMDLK bit of the FASTAT_n register is 1).

If flash sequencer enters command lock state during programming or erasure processing by the command other than programming/erasure suspension, the flash sequencer continues programming or erasure processing. In this state, programming or erasure processing cannot be suspended by the programming/erasure suspension command. If a command is issued in command lock state, ILGLERR bit becomes 1 and the other bits retain the values set due to the previous error detection.

Table 8.1 Error Protection Type (1/2)

| Error Type | Description | SECDTCT | BPLDTCT | ILGCOMERR | FESETERR | SECERR | OTERR | ILGLERR | ERSERR | PRGERR | FHVEERR | CFGDTCT | TBLDTCT | CFAE | DFAE |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| FENTRYR setting error | The value set in FENTRYR_n is not $0000_H$, $0001_H$, or $0080_H$. | 0/1 | 0/1 | 0/1 | 1 | 0/1 | 0/1 | 1 | 0/1 | 0/1 | 0/1 | 0/1 | 0/1 | 0/1 | 0/1 |
| | The FENTRYR_n setting for resuming operation does not match that for suspending operation. | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Illegal command error | An undefined code has been specified in the first cycle of command. | 0 | 0 | 1 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | The value specified in the last access of the multiple-access FACI command is not $D0_H$ (except for DMA programming). | 0 | 0 | 1 | 0 | 0/1 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0/1 | 0/1 |
| | The value (N) specified in the second write access of FACI command in the programming, Multi programming, DMA programming, setting of Configuration settings, setting of Security Settings, setting Block Protection Settings command is wrong. | 0 | 0 | 1 | 0 | 0/1 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0/1 | 0/1 |
| | Blank checking command has been issued with inconsistent BCDIR, FSADDR, and FEADDR settings.*1 | 0 | 0 | 1 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | Area Erasure command has been issued with inconsistent FSADDR and FEADDR settings.*1 | 0 | 0 | 1 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | Area Erasure command or Blank checking command has been issued with the settings of FSADDR and FEADDR crossing over the different Data Flash boundary.*2 | 0 | 0 | 1 | 0 | 0/1 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0/1 |
| | FACI command has been issued with the access size different from the specification. (See **Table 6.2**.) | 0 | 0 | 1 | 0 | 0/1 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0/1 | 0/1 |
| | FACI command has been issued against FACI command not acceptable mode. (See **Table 6.3**.) | 0 | 0 | 1 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | FACI command has been issued when command acceptance conditions are not satisfied. (See **Table 6.4**.) | 0/1 | 0/1 | 1 | 0/1 | 0/1 | 0/1 | 1 | 0/1 | 0/1 | 0/1 | 0/1 | 0/1 | 0/1 | 0/1 |
| Erase error | An error has occurred during flash memory erasure. | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0/1 | 0 | 0 | 0 | 0 |
| Program error | An error has occurred during flash memory program. | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0/1 | 0 | 0 | 0 | 0 |
| Code flash access error | FACI command has been issued to wrong address in code flash programming/erasure mode. (See **Section 4.3, FASTAT — Flash Access Status Register**.) | 0 | 0 | 0 | 0 | 0/1 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 1 | 0 |
| Data flash access error | Programming,Multi Programming, DMA Programming or block Erasure command has been issued to wrong address in data flash programming/erasure mode. (See **Section 4.3, FASTAT — Flash Access Status Register**.) | 0 | 0 | 0 | 0 | 0/1 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |
| | Address boundary between different Data Flash*2 is crossed by DMA programming in data flash programming/erasure mode. (See **Section 6.3.10, DMA Programming Command**.) | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |
| | Area Erasure or Blank Checking command has been issued to wrong address in data flash programming/erasure mode. (See **Section 4.3, FASTAT — Flash Access Status Register**.) | 0 | 0 | 1 | 0 | 0/1 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |
| | Setting of Security Settings command has been issued to wrong address in data flash programming/erasure mode. (See **Section 4.3, FASTAT — Flash Access Status Register**.) | 0 | 0 | 0 | 0 | 0/1 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |
| | Setting of Configuration Settings command has been issued to wrong address in data flash programming/erasure mode. (See **Section 4.3, FASTAT — Flash Access Status Register**.) | 0 | 0 | 0 | 0 | 0/1 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |
| | Setting of Block Protect Settings command has been issued to wrong address in data flash programming/erasure mode. (See **Section 4.3, FASTAT — Flash Access Status Register**.) | 0 | 0 | 0 | 0 | 0/1 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |

Table 8.1 Error Protection Type (2/2)

| Error Type | Description | SECDTCT | BPLDTCT | ILGCOMERR | FESETERR | SECERR | OTERR | ILGLERR | ERSERR | PRGERR | FHVEERR | CFGDTCT | TBLDTCT | CFAE | DFAE |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Security | A command has been issued against OTP setting area | 0 | 0 | 0 | 0 | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | Code Flash access protection error | 0 | 0 | 0 | 0 | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0/1 | 0 |
| | Data Flash access protection error | 0 | 0 | 0 | 0 | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0/1 |
| | Configuration Setting area access protection error | 0 | 0 | 0 | 0 | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0/1 |
| | Security Setting area access protection error | 0 | 0 | 0 | 0 | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0/1 |
| | Block Protection area access protection error | 0 | 0 | 0 | 0 | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0/1 |
| | The error that security function of ICUMD caused. (See *Table 17.1, List of Security Functions,* in *the RH850/E2x ICUMD User's Manual.*) | 0 | 0 | 0 | 0 | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0/1 | 0/1 |
| Other | FACI command-issuing area has been accessed in read mode. | 0/1 | 0/1 | 0/1 | 0/1 | 0/1 | 1 | 1 | 0/1 | 0/1 | 0/1 | 0/1 | 0/1 | 0/1 | 0/1 |
| | FACI command-issuing area has been read in code flash programming/erasure mode or data flash programming/erasure mode. | 0/1 | 0/1 | 0/1 | 0/1 | 0/1 | 1 | 1 | 0/1 | 0/1 | 0/1 | 0/1 | 0/1 | 0/1 | 0/1 |
| FHVE setting error | FHVE3CNT bit in FHVE3 register has changed to 0 while command processing is provided by the flash sequencer. | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0/1 | 0/1 | 1 | 0 | 0 | 0 | 0 |
| Security Setting Area ECC error | 2-bit error has been detected during Security Setting area access. | 1 | 0 | 0 | 0 | 0/1 | 0 | 0/1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Block Protection Area ECC error | 2-bit error has been detected during Block Protection Setting area access. | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Configuration Area ECC error | 2-bit error has been detected during Configuration Setting area access. | 0 | 0 | 0 | 0 | 0/1 | 0 | 0/1 | 0 | 0 | 0 | 1 | 0 | 0 | 0 |
| P/E Parameter table ECC error | Data with 2-bit error has been detected during P/E Parameter table access. | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 |

Note 1. Inconsistent FSADDR and FEADDR settings are follow.

- FSADDR[27:0] > FEADDR[27:0] (Area Erasure command )
- FSADDR[20:0] > FEADDR[20:0] (Blank checking command (BCDIR=0))
- FSADDR[20:0] < FEADDR[20:0] (Blank checking command (BCDIR=1))

Note 2. The 96 KBytes Data Flash bank consists of two 48 KBytes Data Flash (EEP) called EEP0-0 and EEP0-1, or EEP1-0 and EEP1-1. 32 KBytes Data Flash bank for ICUM consists of one 32 Kbytes Data Flash called EEP2. These EEPs are treated as different Data Flash.

## 8.3 Boot Program Protection

### 8.3.1 User Boot Protection

The user boot area can be programmed/erased by the serial programming. Since this area is usually write-protected for the self-programming, it can be used to store programs such as a boot program safely.

## 8.4 Blank Checking of Code Flash Memory

Note that an ECC error is detected and an exception is generated when code flash memory to which no data have been written after erasure (i.e. in the non-programmed state) is read. When an ECC error occurs, the data values cannot be guaranteed. To confirm the non-programmed state of the memory, confirm that code flash memory data, ECC bits, and address parity bits are all 1. For usage of the ECC function of the code flash memory, see *Section 38*, *Functional Safety*, in *the User's Manual: Hardware*.

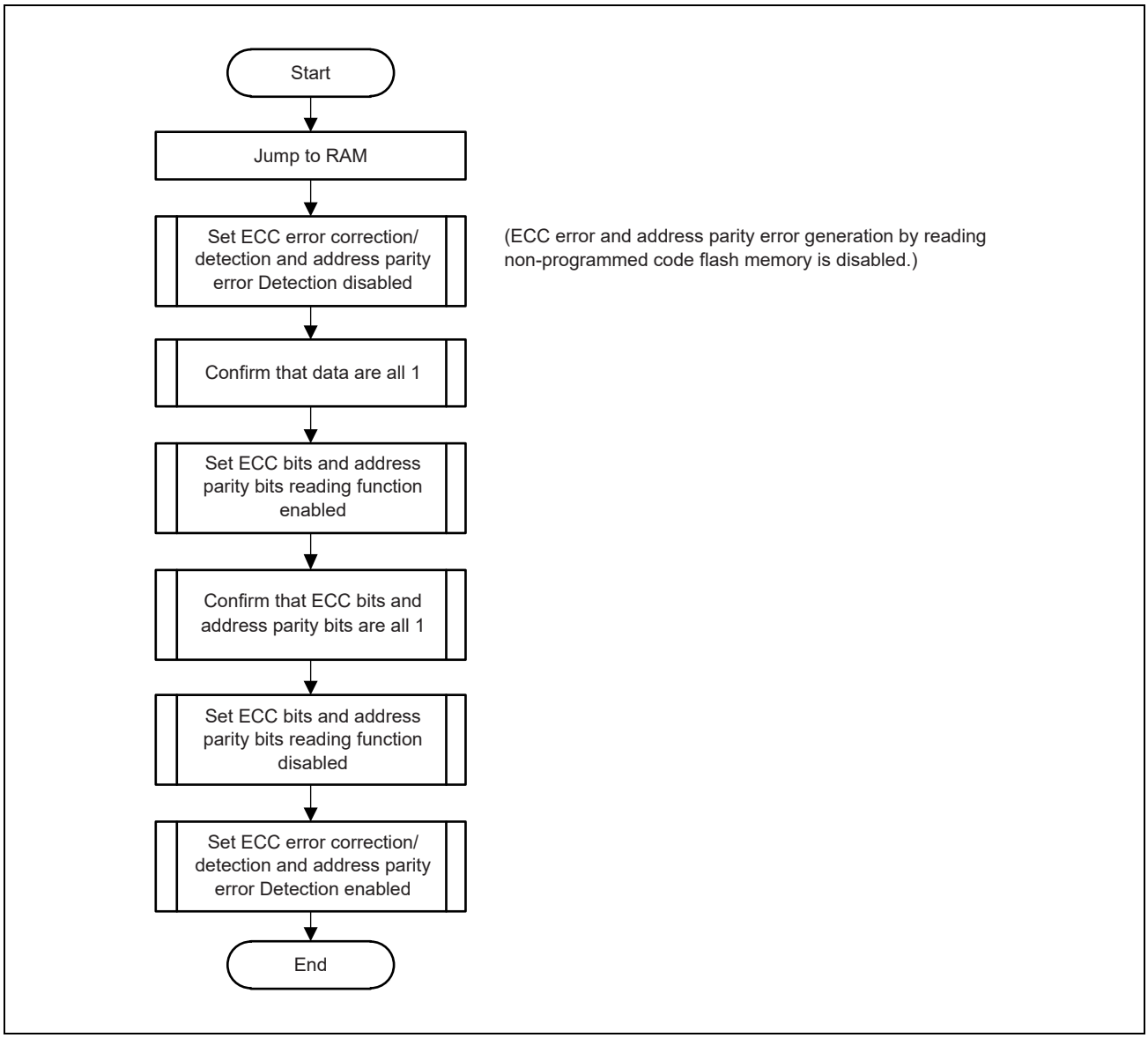



Figure 8.1 Blank Checking of Code Flash Memory

# Section 9 Usage Notes

### (1) Reading Areas where Programming or Erasure was Interrupted

When programming or erasure of an area of flash memory is interrupted, the data stored in the area become undefined. To avoid undefined data that are read out becoming the source of faulty operation, take care not to fetch instructions or read data from areas where programming or erasure was interrupted.

### (2) Prohibition of Additional Writing

Writing to a given area twice is not possible. If you want to update data in an area of flash memory after writing to the area has been completed, erase the area first.

### (3) Resets during Programming and Erasure

In the case of an external reset during programming and erasure, wait for at least width of reset pulse more than the min value once the operating voltage is within the range stipulated in the electrical characteristics after assertion of the reset signal before releasing the device from the reset state.

### (4) Allocation of Vectors for Interrupts and Other Exceptions during Programming and Erasure

Generation of an interrupt or other exception during programming or erasure may lead to fetching of the vector from the code flash memory. Set the address for vector fetching to a different bank of code flash memory which used in programing and erasure, or set the address for vector fetching to an address that is not in the code flash memory. For how to change the address for vector fetching, see *Section 3, CPU System* and *Section 6, Interrupts*, in *the User's Manual: Hardware*.

### (5) Abnormal Termination of Programming and Erasure

Even if programming/erasure ends abnormally due to the generation of an external reset or power shutoff, the programming/erasure state of the flash memory with undefined data cannot be verified or checked. For the area where programming/erasure ends abnormally, the blank check function cannot judge whether the area is erased successfully or not. Erase the area again to prove that the corresponding area is completely erased before using.

### (6) Items Prohibited during Programming / Erasure / Blank check

Do not perform the following operations while the flash memory is programmed, erased or checked for blank.

- Set the operating voltage from the power supply outside the allowed range.
- Update the FHVE15 and FHVE3 values.
- Change the operating frequency of the peripheral clock.

### (7) Securing Coherency after Rewriting Code Flash Memory

After rewriting a code flash area, if a code flash instruction is executed, secure coherency by clearing the instruction cache, and clearing the data buffer (see Usage Notes in *Section 3, CPU System*, in *the User's Manual: Hardware*).

# Section 10 Electrical Characteristics

This section explains the electrical characteristics for the self-programming.

Values are only for processing by hardware. Software overhead is not taken into account.

Note that these electrical characteristics differ from those of when the serial programming is used.

## 10.1 Code Flash Characteristics

Table 10.1 Code Flash Basic Characteristics

| Item | Symbol | Conditions | Min. | Typ. | Max. | Unit |
|---|---|---|---|---|---|---|
| Programming endurance*1 | CWRT | Retained for 20 years*2 | 1000 | — | — | Times |
| Temperature range of programming | TPRG | Tj | –40 | — | +150 | °C |
| Temperature range of reading | TREAD | Tj | –40 | — | +150 | °C |

Note 1. Programming endurance is defined as the number of times each block is erased. Where programming endurance is n times (n = 1000 in this case), each block is erasable n times. For example, given a memory device that has 64-Kbyte erasure blocks, programming in the address range of each 512-byte programming block (128 programming operations, one for each block) in an erasure block and then erasing the block counts as one time in terms of programming endurance. However, programming of a given address range more than once after erasure is not possible (overwriting after programming is prohibited).

Note 2. This is the case when the average Ta is 85°C. This retained period is from when the erasure of the code flash memory has been normally completed.

Table 10.2 Code Flash Programming Characteristics

Conditions: Refer to *Section 47, Electrical Characteristics*, in *the User's Manual: Hardware*

| Item | Condition | Block Size | Min. | Typ. | Max. | Unit |
|---|---|---|---|---|---|---|
| Programming time | Programming endurance < 100 times | 512 B | — | 0.4 | 6 | ms |
| | | 16 KB | — | 13 | 80 | ms |
| | | 64 KB | — | 52 | 320 | ms |
| | | 1 MB | — | 0.9 | 5.2 | s |
| | Programming endurance ≥ 100 times | 512 B | — | 0.5 | 7.2 | ms |
| | | 16 KB | — | 16 | 96 | ms |
| | | 64 KB | — | 64 | 384 | ms |
| Erasing time | Programming endurance < 100 times | 16 KB | — | 39 | 120 | ms |
| | | 64 KB | — | 141 | 480 | ms |
| | | 1 MB | — | 2.3 | 7.7 | s |
| | Programming endurance ≥ 100 times | 16 KB | — | 47 | 144 | ms |
| | | 64 KB | — | 169 | 576 | ms |

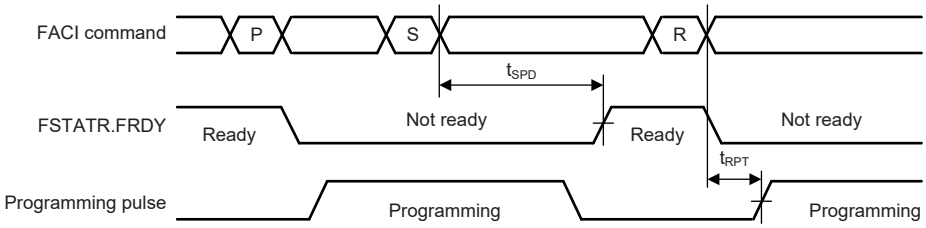
Table 10.3 Suspension/Resumption/Forced Stop

Conditions: Refer to *Section 47, Electrical Characteristics*, in *the User's Manual: Hardware*

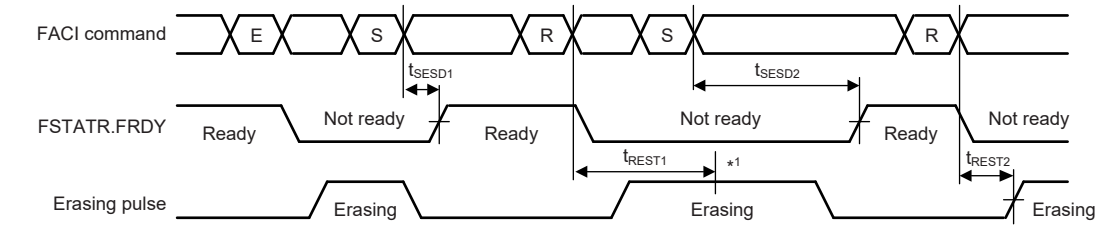
| Item | Symbol | Condition | Min. | Typ. | Max. | Unit |
|---|---|---|---|---|---|---|
| Suspend latency during programming | $t_{SPD}$ | — | — | — | 120 | µs |
| Programming resume time*1 | $t_{RPT}$ | — | — | — | 50 | µs |
| Suspend latency during erasing | $t_{SESD1}$ | Priority on suspension<br>The 1st suspend for the same pulse | — | — | 120 | µs |
| | $t_{SESD2}$ | Priority on suspension<br>The 2nd suspend for the same pulse | — | — | 1.7 | ms |
| | $t_{SEED}$ | Priority on erasure | — | — | 1.7 | ms |
| Erasing resume time*1 | $t_{REST1}$ | Priority on suspension<br>Resume after the 1st suspend for the same pulse | — | — | 1.7 | ms |
| | $t_{REST2}$ | Priority on suspension<br>Resume after the 2nd suspend for the same pulse | — | — | 80 | µs |
| | $t_{REET}$ | Priority on erasure | — | — | 80 | µs |
| Forced stop command latency | $t_{FD}$ | — | — | — | 20 | µs |

Note 1. The time taken for resumption includes an overhead for the resumption of programming or erasure. In suspension-priority mode, a time for reapplication of the erasing pulse that was cut off at the time of suspension is also required. Resume time is defined as time added by programming or erasing due to those sources.

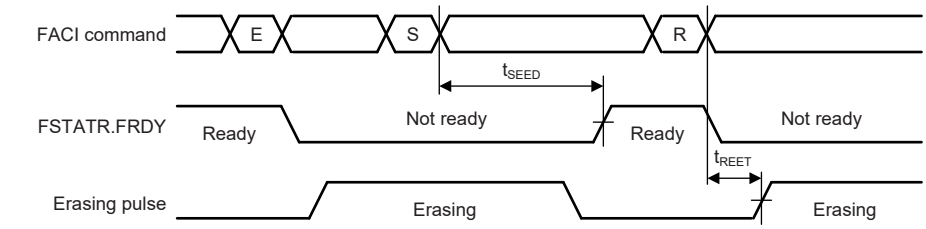
• Programming suspension



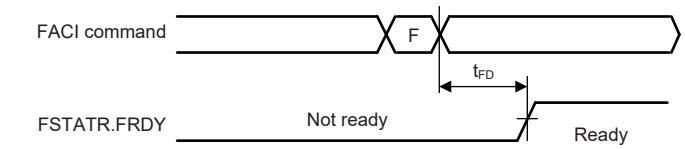

• Suspension of erasure in suspension-priority mode




• Suspension of erasure in erasure-priority mode




• Forced stop




**Remarks:** P: Programming command E: Block erasure command
S: P/E suspension command R: P/E resumption command F: Forced stop command

Note 1. Time for resumption includes time for reapplying the erasing pulse (up to one full pulse) that was cut off at the time of suspension.

Figure 10.1 Timing of Suspension/Resumption/Forced Stop

## 10.2 Data Flash Characteristics

Table 10.4 Data Flash Basic Characteristics

| Item | Symbol | Condition | Min. | Typ. | Max. | Unit |
|---|---|---|---|---|---|---|
| Programming endurance*1 | CWRT | Retained for 20 years*2 | 125000 | — | — | Times |
| (Data Area) | | Retained for 3 years*2 | 250000 | — | — | Times |
| Temperature range of programming | TPRG | Tj | −40 | — | +150 | °C |
| Temperature range of reading | TREAD | Tj | −40 | — | +150 | °C |

Note 1. Programming endurance is defined as the number of times each block is erased. Where programming endurance is n times (n = 125000 in this case), each block is erasable n times. For example, given a memory device that has 64-byte erasure blocks, programming in the address range of each 4-byte programming block (16 programming operations, one for each block) in an erasure block and then erasing the block counts as one time in terms of programming endurance. However, programming of a given address range more than once after erasure is not possible (overwriting after programming is prohibited).

Note 2. This is the case when the average Ta is 85°C. This retained period is from when the erasure of the data flash memory has been normally completed.

Table 10.5 Data Flash Programming Characteristics

Conditions: Refer to *Section 47, Electrical Characteristics*, in *the User's Manual: Hardware*

| Item | Block Size | Min. | Typ. | Max. | Unit |
|---|---|---|---|---|---|
| Programming time | 4 B | — | 0.16 | 1.7 | ms |
| | 8 B | — | 0.32 | 3.4 | ms |
| | 16 B | — | 0.64 | 6.8 | ms |
| | 32 B | — | 1.3 | 12.3 | ms |
| | 64 B | — | 2.6 | 13 | ms |
| Erasing time | 64 B | — | 1.7 | 10 | ms |
| Blank check time*1 | 4 B | — | — | 30 | µs |
| | 64 B | — | — | 100 | µs |
| | 2 KB | — | — | 2.2 | ms |
| Setting of Security Settings | 4 B / 32 B | — | 33 | 273 | ms |
| Setting of Configuration Settings | 4 B / 32 B | — | 16 | 124 | ms |
| Setting of Block Protection | 4 B / 32 B | — | 16 | 124 | ms |

Note 1. When the area size for blank checking is 2 Kbytes or larger, the time will be proportional to that of 2-Kbyte blank checking. When the area size for blank checking is 64 bytes or larger and less than 2 Kbytes, the time will be proportional to that of 64-byte blank checking.

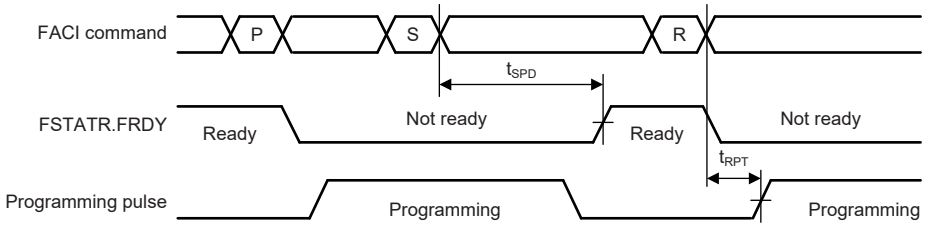
Table 10.6 Suspension/Resumption/Forced Stop

Conditions: Refer to *Section 47, Electrical Characteristics*, in *the User's Manual: Hardware*

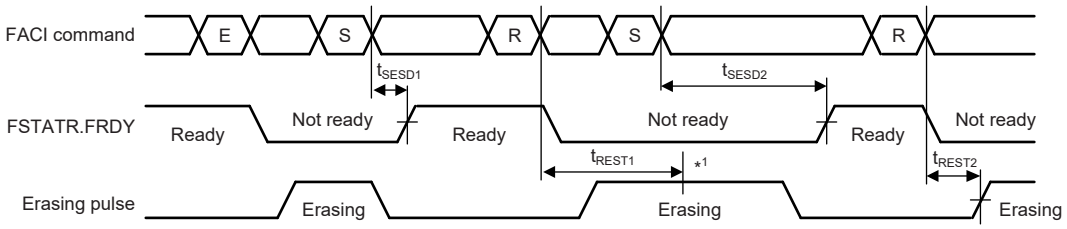
| Item | Symbol | Condition | Min. | Typ. | Max. | Unit |
|---|---|---|---|---|---|---|
| Suspend latency during programming | $t_{SPD}$ | — | — | — | 120 | µs |
| Programming resume time*1 | $t_{RPT}$ | — | — | — | 50 | µs |
| Suspend latency during erasure | $t_{SESD1}$ | Priority on suspension<br>The 1st suspend for the same pulse | — | — | 120 | µs |
| | $t_{SESD2}$ | Priority on suspension<br>The 2nd suspend for the same pulse | — | — | 300 | µs |
| | $t_{SEED}$ | Priority on erasure | — | — | 300 | µs |
| Erasing resume time*1 | $t_{REST1}$ | Priority on suspension<br>Resume after the 1st suspend for the same pulse | — | — | 300 | µs |
| | $t_{REST2}$ | Priority on suspension<br>Resume after the 2nd suspend for the same pulse | — | — | 70 | µs |
| | $t_{REET}$ | Priority on erasure | — | — | 70 | µs |
| Forced stop command latency | $t_{FD}$ | — | — | — | 20 | µs |

Note 1. The time taken for resumption includes an overhead for the resumption of programming or erasure. In suspension-priority mode, a time for reapplication of the erasing pulse that was cut off at the time of suspension is also required. Resume time is defined as time added by programming or erasing due to those sources.

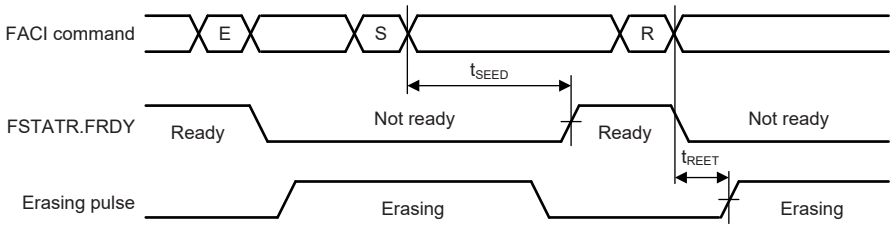
• Programming suspension



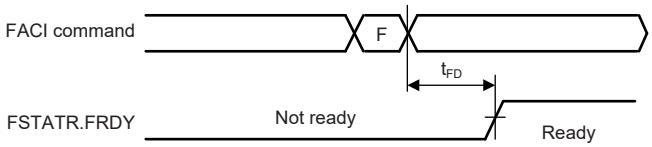

• Erasing suspension in suspension-priority mode




• Erasing suspension in erasure-priority mode




• Forced stop




**Remarks**: P: Programming/Multi Programming command E: Block Erasure/Area Erasure command
S: P/E suspension command R: P/E resumption command F: Forced stop command

Note 1. Time for resumption includes time for reapplying the erasing pulse (up to one full pulse) that was cut off at the time of suspension.

Figure 10.2 Timing of Suspension / Resumption / Forced Stop

RH850/E2x-FCC1 Flash Memory
User's Manual: Hardware Interface

Publication Date: Rev.0.10 Nov 18, 2016
Rev.1.30 Dec 26, 2019

Published by: Renesas Electronics Corporation

**SALES OFFICES** Renesas Electronics Corporation http://www.renesas.com

Refer to "http://www.renesas.com/" for the latest and detailed information.


**Renesas Electronics Corporation**
TOYOSU FORESIA, 3-2-24 Toyosu, Koto-ku, Tokyo 135-0061, Japan

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

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

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

**Renesas Electronics (China) Co., Ltd.**
Room 101-T01, Floor 1, Building 7, Yard No. 7, 8th Street, Shangdi, Haidian District, Beijing 100085, China
Tel: +86-10-8235-1155, Fax: +86-10-8235-7679

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

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

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

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

**Renesas Electronics Malaysia Sdn.Bhd.**
Unit No 3A-1 Level 3A Tower 8 UOA Business Park, No 1 Jalan Pengaturcara U1/51A, Seksyen U1, 40150 Shah Alam, Selangor, Malaysia
Tel: +60-3-5022-1288, Fax: +60-3-5022-1290

**Renesas Electronics India Pvt. Ltd.**
No.777C, 100 Feet Road, HAL 2nd Stage, Indiranagar, Bangalore 560 038, India
Tel: +91-80-67208700

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



© 2019 Renesas Electronics Corporation. All rights reserved.



Colophon 6.0

# RH850/E2x-FCC1

## Flash Memory

Renesas Electronics Corporation

R01UH0707EJ0130