Renesas RA Family

Renesas Boot Firmware for RA8M2 MCU Group

Introduction

This application note describes the communication protocol, command set, and usage of the boot firmware provided with Renesas RA8M2 MCU Group.

Target Device

RA8M2 MCU Group

Contents

1. Terminology

1.1 Boot Firmware

Boot firmware is the program included in the microcontroller to rewrite the MRAM memory.

1.2 MRAM

The following areas are collectively called “MRAM”:

- Code MRAM: The ROM area where the user's program code and data are written.
- Extra MRAM: The RAM area where configuration data is written.

The boot firmware rewrites and reads these areas according to commands given by the user.

1.3 SiP Flash

The ROM area of SIP flash memory mapped inside the MCU is called “SiP Flash”.
The boot firmware rewrites and reads this area according to commands given by the user.

1.4 Memory

In this document, the areas that can be rewritten by boot firmware, such as the following, may be generically referred to as “memory”.
- Flash memory (External flash memory connected to the MCU)
- MRAM
- SiP Flash

1.5 Device Lifecycle Management (DLM)

The Renesas Advanced (RA) Family MCUs adopt the concept of device lifecycle and maintain the lifecycle state inside the device.

The boot firmware controls the executable commands and the range of operations that can be performed with each command in each lifecycle state. In addition, the boot firmware has a user-executable command as the only way to transition lifecycle states.

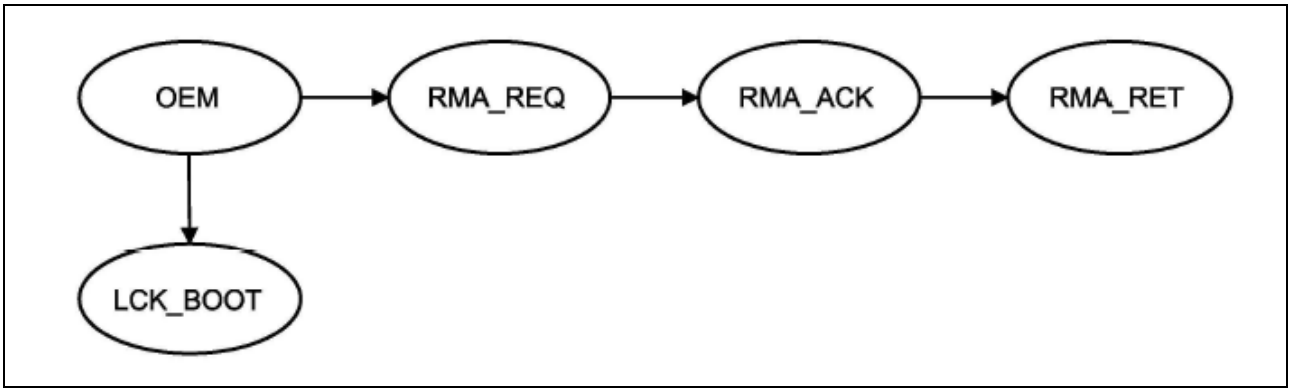


Figure 1. Device Lifecycle States

Table 1. DLM States

DLM State Name	Description
OEM	Original Equipment Manufacturer
LCK_BOOT	LoCKed BOOT interface
RMA_REQ	Return Material Authorization REQuest
RMA_ACK	Return Material Authorization ACKnowledged
RMA_RET	Return Material Authorization RETurn

1.6 Authentication Level (AL)

In the RA8 MCU Series, the executable commands and the range of operations that can be performed with each command are determined not only by the DLM state but also by the authentication level. There are three authentication levels: AL2, AL1, and AL0. The executable operation range is the widest at AL2, and the narrowest at AL0.

Changing the authentication level is possible only when the DLM state is OEM, so that the executable operation range at OEM is more subdivided. On the other hand, changing the authentication level when the DLM state is not OEM is not possible because the DLM state and authentication level are uniquely associated at DLM states other than OEM. To change the authentication level at OEM, change the protection level, then reset the device, or use dedicated boot firmware commands. Level change by the boot firmware command is a temporary change, and the authentication level returns to the level before the change when resetting the device.

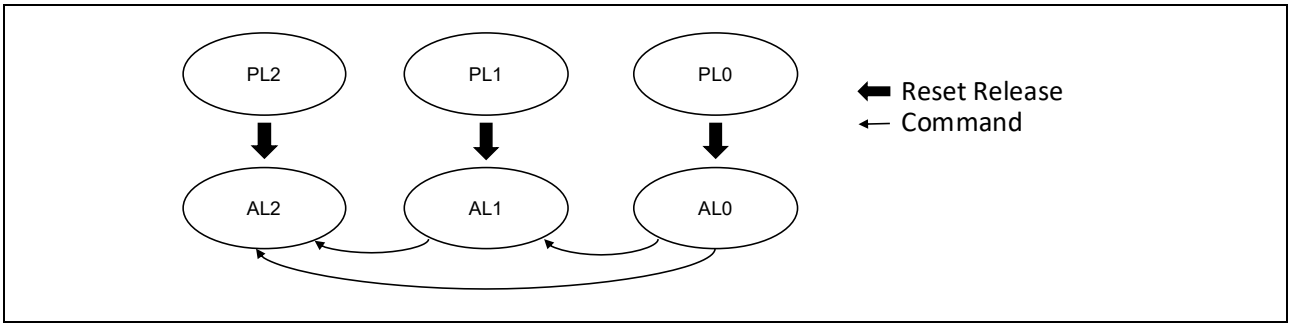


Figure 2. Authentication Level Transitions

Table 2. Authentication Level States

AL State Name	Description
AL2	Authentication Level 2
AL1	Authentication Level 1
AL0	Authentication Level 0

1.7 Protection Level (PL)

Protection level is the initial authentication level: the authentication level when the device boots is determined by the protection level. Like the authentication level, it is not possible to change the protection level when the DLM state is not OEM. To change the protection level at OEM, use dedicated boot firmware commands.

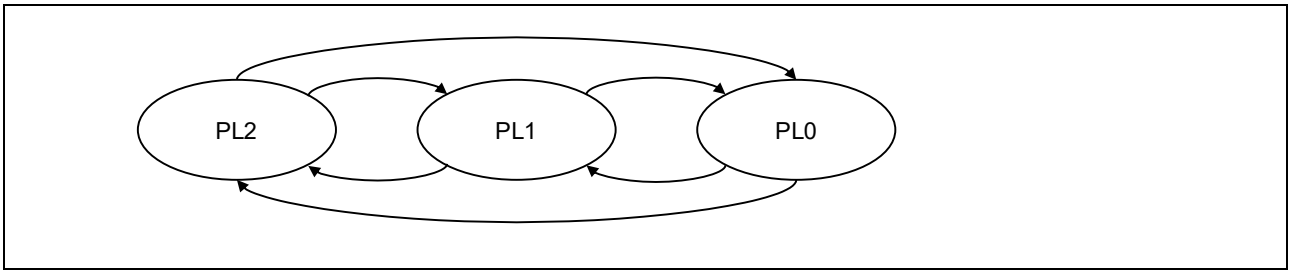


Figure 3. Protection Level Transitions

Table 3. Protection Level States

PL State Name	Description
PL2	Protection Level 2
PL1	Protection Level 1
PL0	Protection Level 0

1.8 Secure/Non-secure

Renesas Advanced (RA) Family MCUs have the attributes of Secure and Non-secure. In particular, the memory area is divided into two exclusive areas, a Secure area and a Non-secure area. The CPU core has two security states, a Secure state and a Non-secure state. The security state of the CPU changes depending on the Secure attribute of the memory where the execution code exists. When the CPU core processes the execution code in the Secure area, it is in the Secure state, and when it processes the execution code in the Non-secure area, it is in the Non-secure state. Then, without going through special procedures, the CPU core transitions from the Non-secure state to the Secure state, or the CPU core is in

the Non-secure state and accesses the memory in the Secure area. By preventing it with the mechanism of the CPU core, it bears a part of the security function of the Renesas Advanced (RA) Family MCUs.

The boot firmware specifies a Secure area and a Non-secure area for the User area and SiP Flash area by a command from the user.

The allocation setting of the Secure area and Non-secure area is referred to as “boundary” in this document. A boundary can be set using the Boundary setting command of the boot firmware.

1.9 Block Protection

Block protection refers to a function that prohibits erasing/writing the specified range of flash memory. The specified range is done in blocks, and there are two types of protection as listed in Table 4.

Table 4. Lock Protection Types

Types of protection	Description
Block protection (BPS)	Protection that can temporarily enable erasing/writing by register setting of the flash sequencer.
Permanent block protection (PBPS)	Protection that permanently disables the release of the Block protection setting.

Although MRAM does not have the concept of “block”, the device has block protection.
Refer to the User's Manual (hardware) for the range and the unit of this protection for MRAM products.
In addition, this protection protects only rewriting for MRAM products because MRAM does not have the concept of “erase”.

1.10 Image

“Image” means data written to flash memory or MRAM using boot firmware.

While “write data” refers to each data to be written, “image” refers to a set of write data to be written to a device or an area.

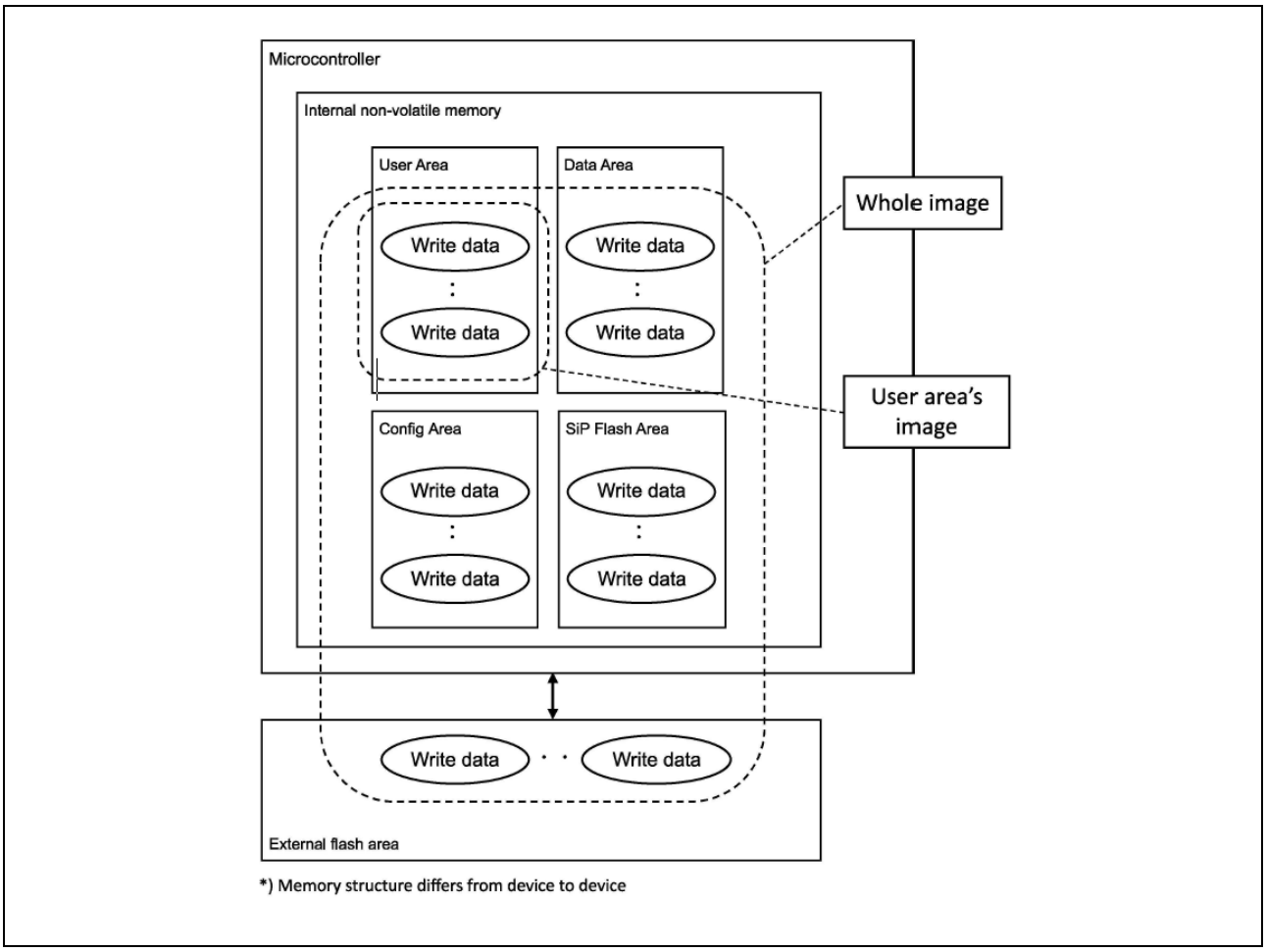


Figure 4. Memory Image Concept

2. System Architecture

Boot firmware has a serial programming interface to send and receive memory control commands between the microcontroller and the host in the serial programming mode. Boot firmware is embedded into the device.

2.1 RA8M2 MCU Group

This chapter describes the system architecture of the RA8M2 MCU Group for the flash memory control.

Table 5. Operating Environment

CPU core	Arm Cortex-M85 (Boot firmware uses only Arm Cortex-M85) Arm Cortex-M33
Max CPU operating frequency	1 GHz (Boot firmware operating frequency: 300 MHz)
Main-OSC	8, 10, 12, 15, 16, 20, 24, 32, 48 MHz • If neither is set, it operates with HOCO. • However, if a Main-OSC whose frequency is around +/- 3% or less of the frequency above is set, there is a possibility that the frequency is misjudged and therefore USB communication fails. To avoid this, it is recommended to choose one of the following options when using USB communication: — Use a Main-OSC whose frequency is the value listed above.

<table>
<tr><td></td><td>— Do not use a Main-OSC and use a Sub-OSC with a frequency that is supported by the specifications of the device.</td></tr>
<tr><td>Operating voltage</td><td><table>
<tr><td>VCC</td><td>1.62 V - 3.63 V</td></tr>
<tr><td>VCC_USB (only USB boot mode)</td><td>3.0 - 3.6 V</td></tr>
<tr><td>VDD</td><td>0.9 V or 0.93 V</td></tr>
</table></td></tr>
<tr><td>Operating mode</td><td>Boot mode</td></tr>
<tr><td>Flash memory</td><td><table>
<tr><td rowspan="4">SiP Flash (ISSI IS25WX}</td><td>Main Flash</td><td>Max 8 MB</td></tr>
<tr><td>OTP area</td><td>Max 65 bytes including permanent lock setting</td></tr>
<tr><td rowspan="2">Protect settings</td><td>SRWD, BP, and TB settings by the Status register</td></tr>
<tr><td>Permanent lock for the Status register by PMR.</td></tr>
</table>
Note1: Way to determine whether SiP Flash is installed in the device
Not all the devices belonging to the RA8M2 group have SiP Flash.
The host can determine whether SiP Flash is installed in the device by the Area information command or the SiP Flash various request command as follows:

- When the Area Information command returns the area information of the SiP Flash area, the device has SiP Flash.

- When an unsupported command error is returned by the SiP Flash various request command, the device does not have SiP Flash.

Note2: ECC of the SiP Flash
Boot firmware enables the ECC function of the SiP Flash when writing to the SiP Flash and disables it when reading from the SiP Flash.
This specification prevents read data from being unintentionally corrected by ECC, and incorrect data is read in both of the following situations:

- Read data written by boot firmware in a user program, which enables the ECC function of SiP Flash.

- Read the data of the boot firmware written by a user program that disables the ECC function of SiP Flash.

The enable/disable is performed by changing the volatile configuration register, as the boot firmware does not change the non-volatile configuration register.

Note3: Protection function of the SiP Flash
The items listed in the “Protect settings” column of the table above are not all the protection functions that the SiP Flash has,
but only the protection functions that can be set by boot firmware commands.</td></tr>
<tr><td>MRAM</td><td><table>
<tr><td>Code MRAM</td><td>User area</td><td>Max.1 MB</td></tr>
<tr><td>Extra MRAM</td><td>User boot area</td><td>None</td></tr>
</table></td></tr>
</table>

		Data area	None
RAM	SRAM: Max 1792 KB (used by the Boot firmware: 256 KB)		
Communication method	• [2-wire UART communication] — (Initial/Min) 9600 bps — (Max) 6 Mbps • [USB communication] — 12 Mbps — When performing USB communication with HOCO, Sub-OSC must be oscillating stably. — USB communication operation is only guaranteed for use with Windows 10 as the host OS. Use with other host OS systems is not guaranteed. • [JTAG/SWD communication] — 25 MHz		

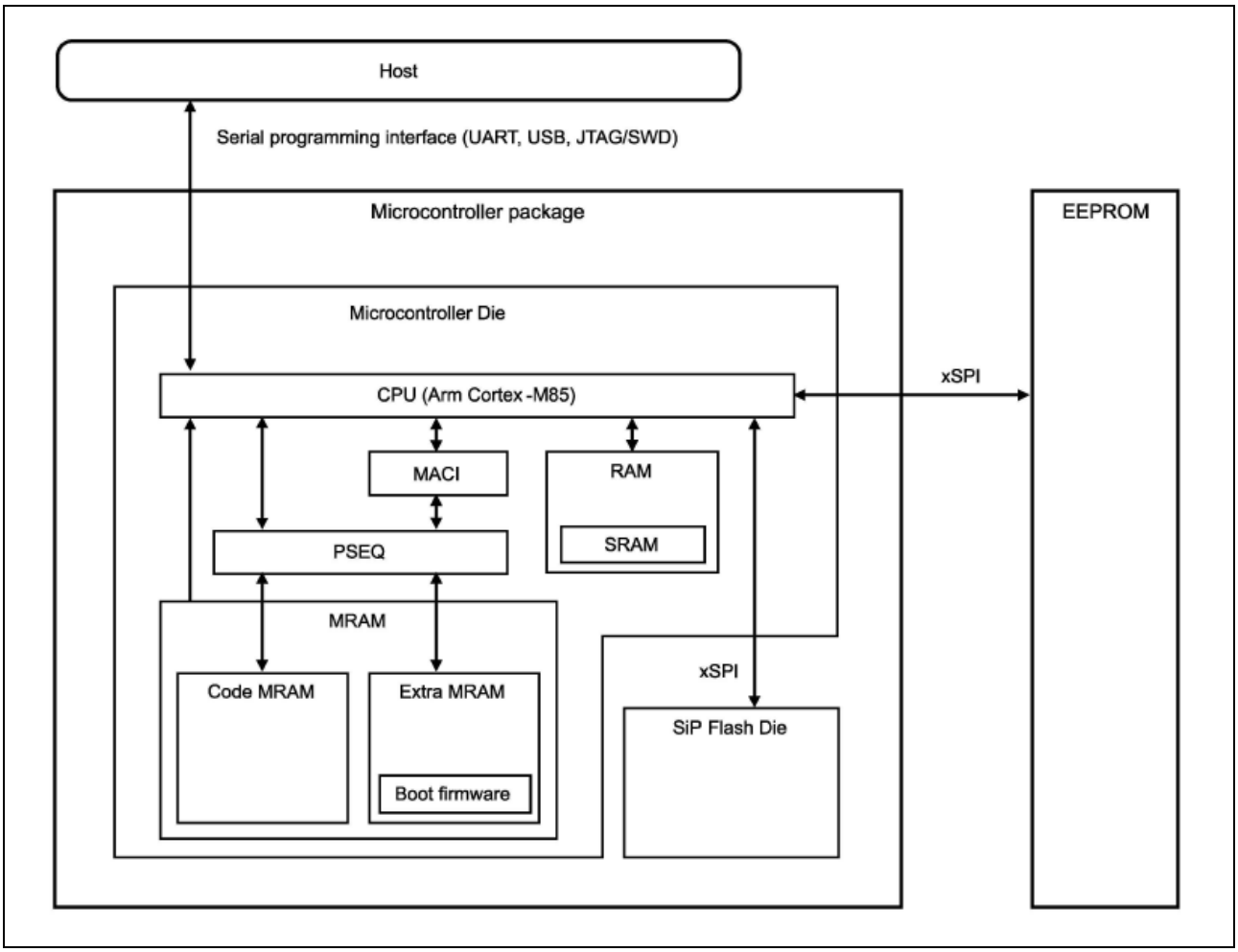


Figure 5. Block Diagram

3. Communication Methods

Boot firmware has interfaces for the following communication methods:

- 2-wire UART communication
- Universal Serial Bus (USB) communication
- JTAG/SWD communication

3.1 2-wire UART communication

Boot firmware supports the 2-wire UART communication as shown in Figure 6.

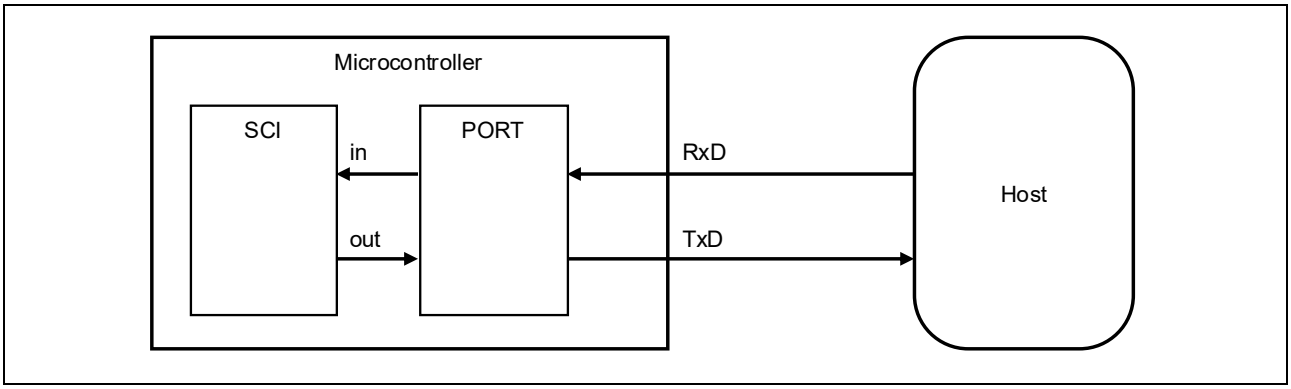


Figure 6. 2-Wire UART Communication

Table 6. UART Settings

Interface	RSCI-3 ch9
RxD	P208, input mode
TxD	P209, output mode
Transfer rate	9600 bps (minimum, until the baudrate setting command) 6 Mbps (maximum)
Data length	8 bits (LSB first)
Parity bit	None
Stop bit	1 bit

Communication is performed at 9600 bps until the baudrate setting command. After the baudrate setting command is completed, communication is performed at the desired transfer rate. The maximum transfer rate that can be communicated with the device is returned by "RMB" of the signature request command.

Note: If the communication cable is disconnected during communication, subsequent operations may not be guaranteed.

3.2 Universal Serial Bus (USB) Communication

Boot firmware supports USB communication as shown in Figure 7.

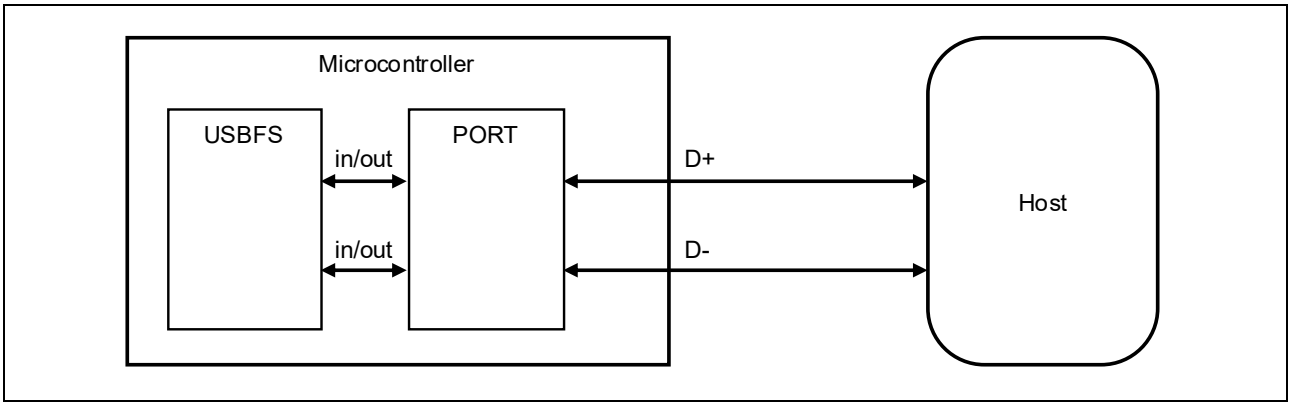


Figure 7. USB Communication

Table 7. USB Settings

Interface	USBFS
VBUS	P407, input mode
D+	Input-output mode
D-	Input-output mode
Transfer rate	12 Mbps (USB 2.0 Full-Speed)
Device class	Communication device class (CDC) • Sub-Class: Abstract Control Model (ACM) • Protocol: Common AT commands
Vendor ID	045Bh (Renesas)
Product ID	0261h
Transfer mode	Control (in/out) Bulk (in/out) Interrupt (in)
Endpoint	EP0: Default control pipe, control transfers (in/out) EP1: TxD pipe, bulk transfers (in) 64 bytes EP2: RxD pipe, bulk transfers (out) 64 bytes EP6: Control pipe, interrupt transfers (in)

Notes:

- If the USB cable is disconnected during communication, subsequent operations are not guaranteed.
- When performing USB communication, the host is notified as self-power mode.
- USB boot does not guarantee operation with bus power.

3.3 JTAG/SWD Communication

Boot firmware supports JTAG/SWD communication. JTAG/SWD communication is enabled by setting a magic code in the JBMDR register during terminal reset.

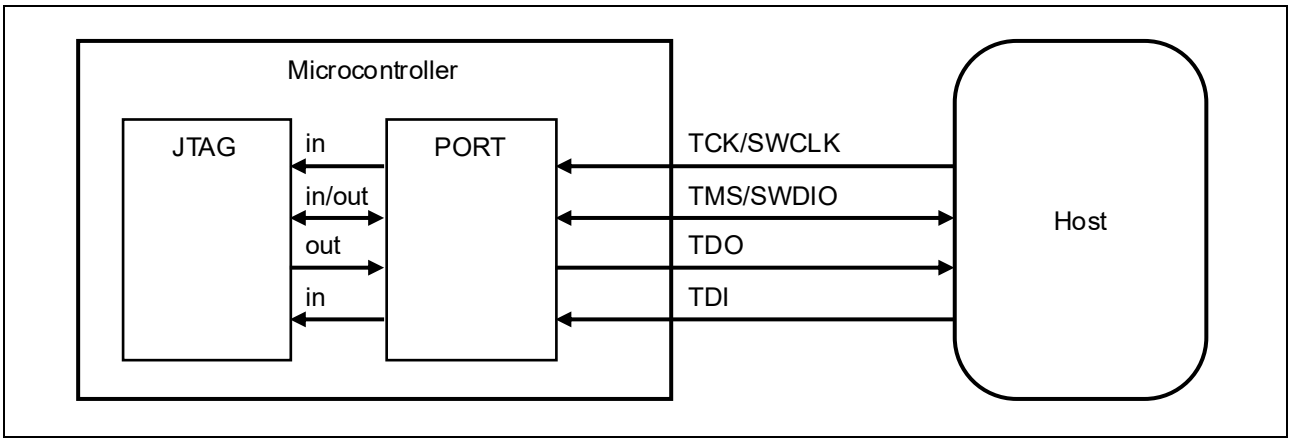


Figure 8. JTAG/SWD Communication

Table 8. JTAG/SWD Settings

[JTAG] TCK	P211, input mode
[JTAG] TMS	P210, input mode
[JTAG] TDO	P209, output mode
[JTAG] TDI	P208, input mode
[SWD] SWCLK	P211, input mode
[SWD] SWDIO	P210, input-output mode
Transfer rate	25 MHz (maximum)
Data length	32 bits
Magic code	A5h

3.3.1 Endianness of Transmission and Reception Data

Store the data transmitted from the host in the JBRDR register in 4-byte words in order from the lower byte.

The data transmitted from the microcontroller is stored in the JBTDR register in 4-byte words in order from the lower byte.

Example: 1-byte data transmission from the host to the microcontroller

Sending data: 55h

JBRDR [31:24]	JBRDR [23:16]	JBRDR [15:8]	JBRDR [7:0]
Don't care	Don't care	Don't care	55h

Example: 7-byte data transmission from the microcontroller to the host

Sending data: 00h, 01h, 02h, 03h

JBTDR [31:24]	JBTDR [23:16]	JBTDR [15:8]	JBTDR [7:0]
03h	02h	01h	00h

Sending data: 04h, 05h, 06h

JBTDR [31:24]	JBTDR [23:16]	JBTDR [15:8]	JBTDR [7:0]
Don't care	06h	05h	04h

3.3.2 Communication Handshake

The host and microcontroller perform a handshake using the JBSTR register in JTAG/SWD communication.

The host must check that JBSTR.RDF = 0 before writing data to JBRDR, and JBSTR.TDE = 0 before reading data from JBTDR.

However, this handshake can be omitted when transmitting and receiving the 5th byte or after, in a packet. Specifically, the host can write JBRDR and read JBTDR without checking JBSTR.

5th byte or after, in a packet, means the following bytes specifically for command and data packets:

Command packet	Command information − ETX
Data packet	Data − ETX

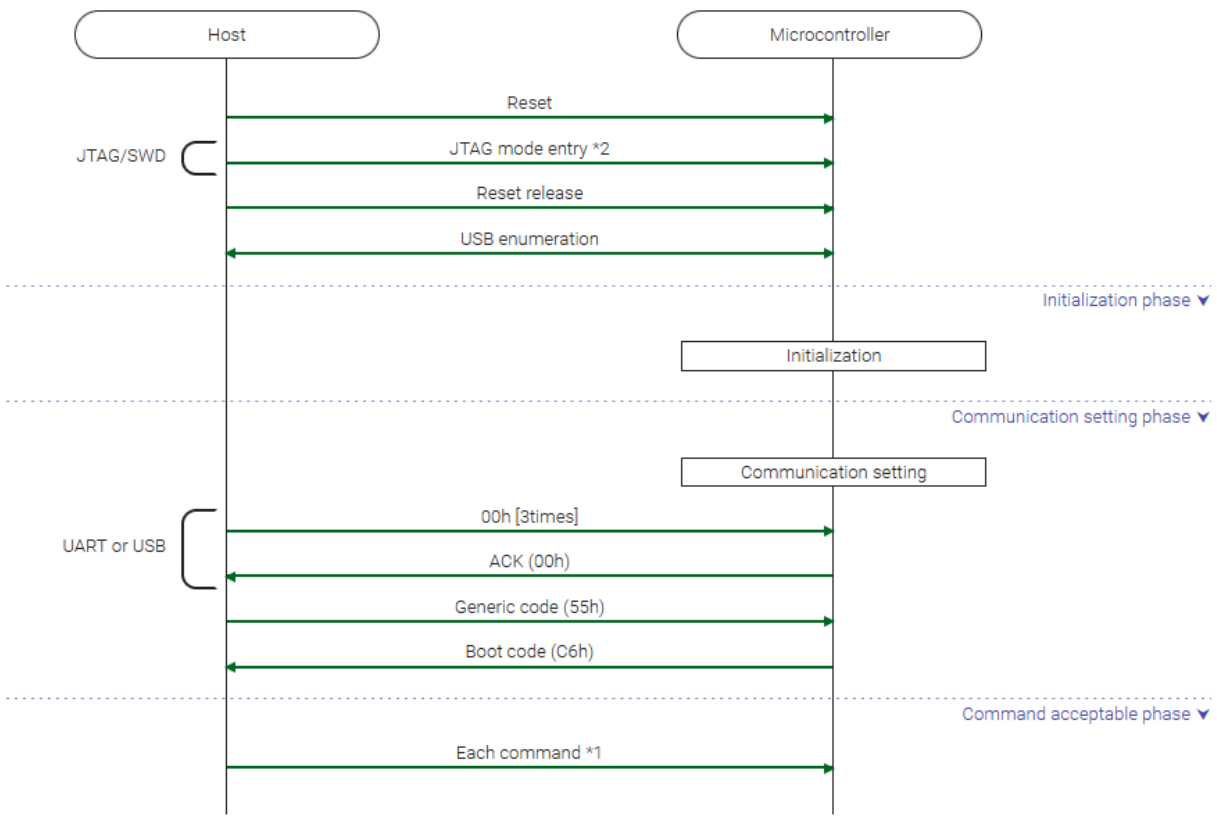
4. General Procedure

Boot firmware transitions phases in the following order after a reset release:

1. Initialization phase
2. Communication setting phase
3. Command the acceptable phase

The above sequence cannot be altered.

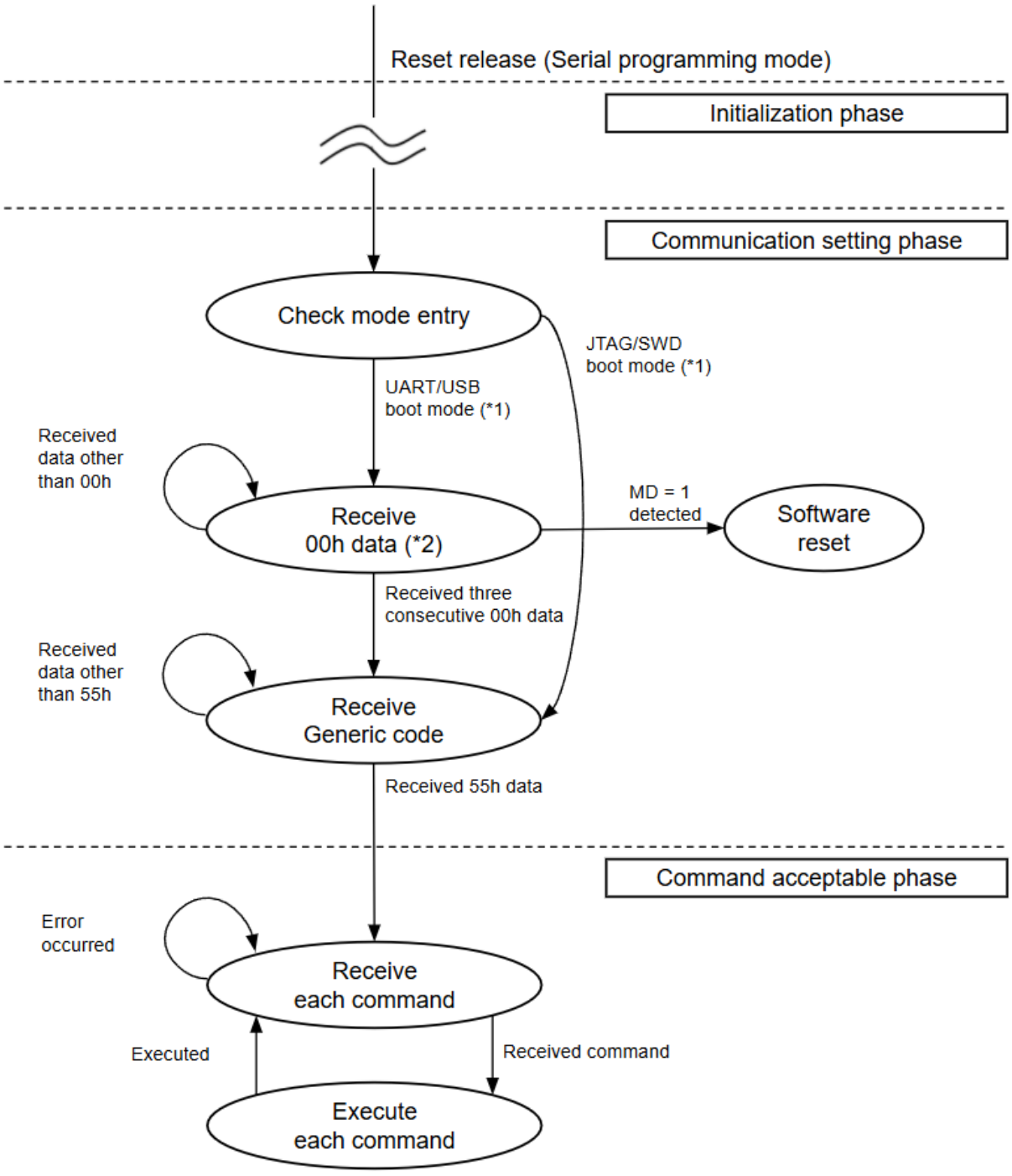
4.1 Sequence Diagram (Generic Sequence)



*1: From the Command acceptable phase, the host and microcontroller send data in turn unless otherwise noted. The host executes the data transmission after the data reception from the microcontroller.
*2: If the magic code “A5H” is set in the JBMDR register during a pin reset, the microcontroller boots into JTAG mode.

Figure 9. Sequence Diagram (Generic Sequence)

4.2 State Transition Diagram (Generic State Transition)



*1: Which communication method is supported depends on the device. This device supports UART, USB, JTAG, and SWD communication.

*2: Boot firmware waits for 00h of data reception by both UART and USB. When 00h is received consecutively three times by either of the communications, the boot firmware adopts the communication by which 00h is obtained as the communication method to be used.

Figure 10. State Transition Diagram (Generic State Transition)

4.3 Initialization Phase

Boot firmware initializes hardware modules in this phase. After that, boot firmware transitions to the “Communication setting phase”.

4.3.1 Processing Procedure

Boot firmware initializes after reset release.

The boot firmware initializes hardware modules and then transitions to the “Communication setting phase”.

4.4 Communication Setting Phase

The boot firmware establishes communication with the host in this phase. Check the connection of each communication method under the conditions shown in Table 9. After receiving the generic code using the established communication method, the boot firmware transitions to the “Command acceptable phase”.

Table 9. Communication Method Determination

Condition	Communication method
Data “00h” was continuously received 3 times by 2-wire UART communication.	2-wire UART communication
Data “00h” was continuously received 3 times by USB communication.	USB communication
DBGSTR.CDBGPWRUPREQ = 1 is set during terminal reset. Magic code “A5h” was set in the JBMDR register during terminal reset. MD pin level is high.	JTAG/SWD communication

4.4.1 Processing Procedure

Boot firmware performs communication settings:

- When all the following conditions are met, the boot firmware performs a software reset:
  - MD = 1
  - JBMDR≠A5h
  - First 8 bytes of User area ≠ all-F
- When all the following conditions are met, JTAG/SWD communication is determined to be selected.
  - When JTAG/SWD communication is selected, boot firmware waits for the generic code without waiting for 00h.
    - MD = 1
    - JBMDR = A5h
- When JTAG/SWD communication is not selected, the boot firmware waits for 00h to be received.
  If 00h is received continuously for 3 bytes in either 2-wire UART communication or USB communication, “ACK” is transmitted. (Data is received until the communication mode is determined).
  The time from when reset is released until 00h can be received is shown in the AC Characteristics.
- The boot firmware waits for the Generic code to be received and sends back the boot code.
  The boot firmware does nothing but wait for Generic code if a code other than Generic code is received.
  - The time from when reset is released until the Generic code can be received is shown in AC Characteristics.
  - Data other than “Boot code” is returned from boot firmware if an unknown error occurred in the Boot firmware.
    This error occurs when the device is damaged or has a similar issue.
- The boot firmware transitions to the “Command acceptable phase” when the transmission of “Boot code” is completed.

4.4.2 Settings of the 2-wire UART Communication

When the device is in serial programming mode, the boot firmware initializes the SCI and waits for reception. By receiving 00h three times consecutively, it is determined that asynchronous 2-wire communication is selected as the communication method. Before receiving 3 bytes, if data other than 00h is received or some data is received from the USB, the count value is reset.

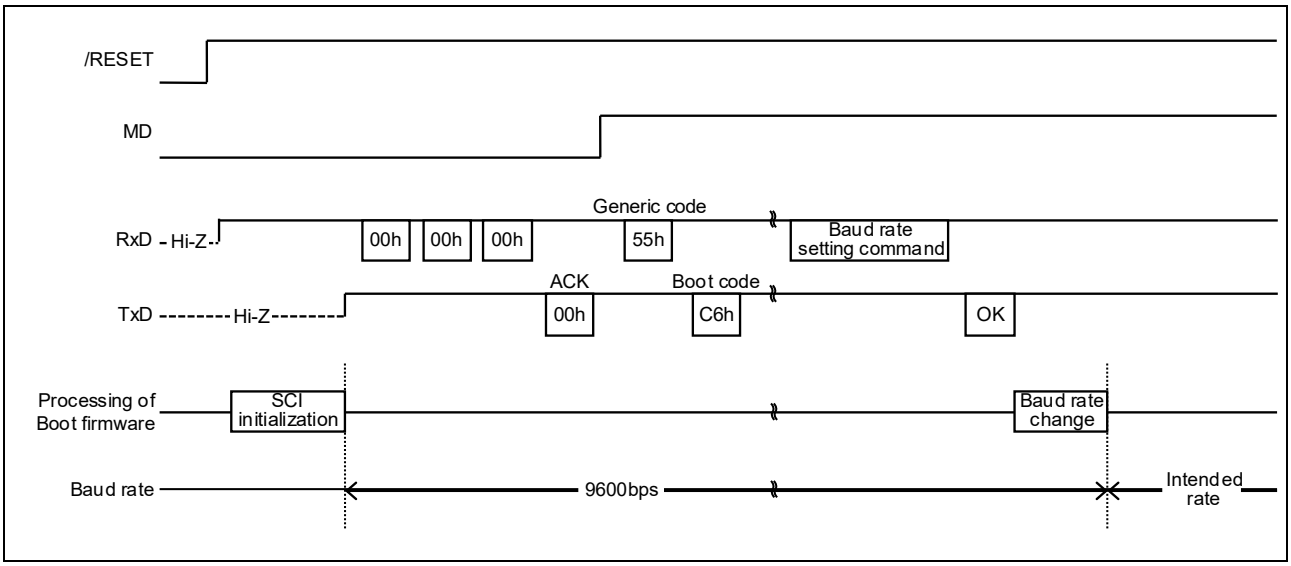


Figure 11. 2-wire UART Communication Setting

* Boot firmware version lower than 3.0 outputs High from TxD after SCI initialization.

Boot firmware version after or equal to 3.0 enables pull-up of TxD after SCI initialization, and outputs High from TxD after 3-byte 00h reception.

By performing the following procedure, communication establishment is completed and the process moves to the “Command acceptable phase”:

1. Receive 3 bytes of 00h data (9600bps) from the host.
   (Perform 00h data transmission until ACK is received in step 2.)
2. Send 00h data (ACK) from boot firmware.
3. Receive 55h data (Generic code) from the host.
4. Send C6h data (Boot code) from boot firmware.

If ACK is not returned even after sending 00h data, check the communication environment and try again from reset release.

4.4.3 Settings of the USB Communication

When the operating mode of the device is serial programming mode, the boot firmware configures the USB into an enumerable state. Set the data communication start by USB Configured status detection. By receiving 00h three times consecutively, it is determined that USB communication is selected as the communication method. Before receiving 3 bytes, if data other than 00h is received or some data is received from UART, the count value is reset.

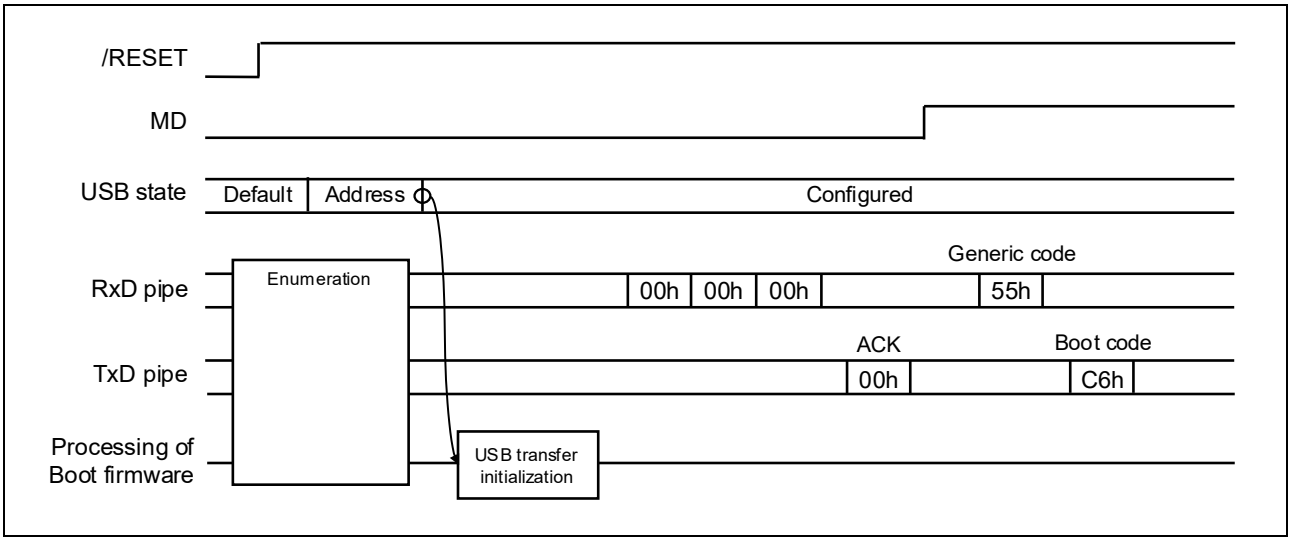


Figure 12. USB Communication Setting

By performing the following procedure, communication establishment is completed and the process moves to the “Command acceptable phase”:

1. When the boot firmware detects the USB Configured state, the USB communication start setting is performed.
2. Receive 3 bytes of 00h data from the host.
   (Perform 00h data transmission until ACK is received in step 3.)
3. Send 00h data (ACK) from boot firmware.
4. Receive 55h data (Generic code) from the host.
5. Send C6h data (Boot code) from boot firmware.

If ACK is not returned even after sending 00h data, check the communication environment and try again from the reset release.

4.4.4 Settings of the JTAG/SWD communication

When the boot firmware detects MD = 1 and JBMDR=A5h, the boot firmware establishes communication with JTAG/SWD communication.

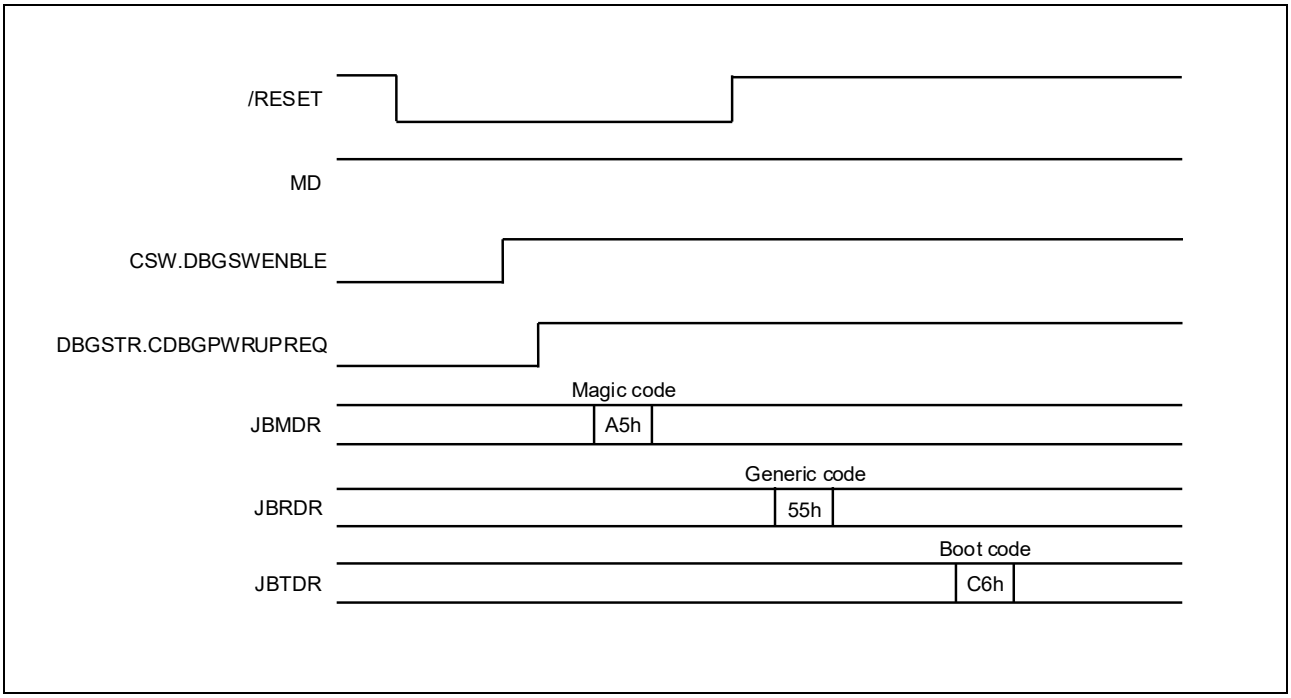


Figure 13. JTAG/SWD Communication Setting

By performing the following procedure, communication establishment is completed and the process moves to the “Command acceptable phase”:

1. Assert the terminal reset.
2. Set CSW.DBGSWENBLE to 1.
3. Set DBGSTR.CDBGPWRUPREQ to 1.
4. Wait until DBGSTR.CDBGPWRUPACK becomes 1.
5. Set JBMDR to A5h.
6. Release the terminal reset.
7. If MD = 1 after following the above procedure, the boot firmware sets the JTAG/SWD communication start setting.
8. Receive 55h data (Generic code) from the host.
9. Send C6h data (Boot code) from the boot firmware.

Follow the steps below to disconnect JTAG/SWD communication with boot firmware:

1. Assert the terminal reset.
2. Set JBMDR to 00h.
3. Set DBGSTR.CDBGPWRUPREQ to 0.
4. Wait until DBGSTR.CDBGPWRUPACK becomes 0.
5. Set CSW.DBGSWENBLE to 0.

4.5 Command Acceptable Phase

Boot firmware accepts the commands in this phase.

4.5.1 Processing Procedure

When the boot firmware receives a command packet, it performs packet analysis:

- The boot firmware recognizes the start of the command packet by receiving SOH. If the boot firmware receives something other than SOH, it waits until SOH is received.
- If ETX is not added to the received command packet, the boot firmware sends a “Packet error”.

- If the SUM of the received command packet is different from the sum value, the boot firmware sends a “Checksum error”.
- If the LNH and LNL of the received command packet are different from the values specified in the packet format, the boot firmware sends a “Packet error”.
- If the CMD in the received command packet is an undefined code, the boot firmware sends an “Unsupported command error”.
- If the LNH and LNL of the received command packet are different from the values specified in each command, the boot firmware sends a “Packet error”.
- When any of the above errors occurs, the boot firmware does not process and returns to the command waiting state.

When the processing above is successfully completed, the boot firmware executes command processing.

When a command is normally finished, boot firmware stays on the “Command acceptable phase”.

5. Packet Format

Use the following packet types:

- Command packet
- Data packet

5.1.1 Elements in the Packet

- CMD: Command code
- RES: Response code
- STS: Status code
- ST2: Status details
- ADR: Failure address
- DLM: Device Lifecycle Management state code

5.1.2 Command Packet

The host sends a command packet to the microcontroller in the following format.

Table 10. Command Packet Format

Symbol	Size	Value	Description
SOH	1 byte	01h	Start of command packet.
LNH	1 byte	-	Packet length (length of “CMD + Command information”) [High].
LNL	1 byte	-	Packet length (length of “CMD + Command information”) [Low].
CMD	1 byte	-	Command code.
Command information	0−255 bytes	-	Command information. Examples: • For Write command: Start/End address • For Baudrate setting command: UART baudrate
SUM	1 byte	-	Sum data of “LNH + LNL + CMD + Command information" (expressed as two's complement). For example: LNH + LNL + CMD + Command information(1) + Command information(2) + ... + Command information(n) + SUM = 00h.
ETX	1 byte	03h	End of packet.

Note: If the host sends data that exceeds 261 bytes, subsequent operations are not guaranteed.

5.1.3 Data Packet

Host and boot firmware send data to each other in the following format.

Table 11. Data Packet Format

Symbol	Size	Value	Description
SOD	1 byte	81h	Start of data packet.
LNH	1 byte	-	Packet length (length of “RES + Data”) [High] (*1).
LNL	1 byte	-	Packet length (length of “RES + Data”) [Low] (*1).
RES	1 byte	-	Response code.
Data	(*3)	-	Transmit data. Examples: • For Write data transmission: Write data. • For Status transmission: Status code (STS), Status details (ST2) and Failure address (ADR).
SUM	1 byte	-	Sum data of “LNH + LNL + RES + Data” (expressed as two's complement) For example: LNH + LNL + RES + Data(1) + Data(2) + ... + Data(n) + SUM = 00h.
ETX	1 byte	03h	End of packet.

Notes:

1. If the host sends a packet whose length is 0 byte or over 1025 bytes, the microcontroller returns a packet with indefinite RES value.
2. If the host sends data that exceeds 1030 bytes, subsequent operations are not guaranteed.
3. The size is 1–1024 bytes. As an exception, the maximum is 1040 bytes only for Encrypted data write command.

5.1.4 CMD: Command Code

Table 12. Command Codes

Value	Name	Description
71h	DLM state transit command	Authentication-free DLM transition.
2Ch	DLM state request command	Get the current DLM state.
72h	Protection level transit command	Protection level transition.
73h	Protection level request command	Get the Protection level.
75h	Authentication level request command	Get the Authentication level.
30h	Authentication command	Authentication-required DLM and AL transition.
28h	Key setting command	Insert the key.
2Ah	User key setting command	Insert the user custom key.
29h	Key verify command	Verify the key.
2Bh	User key verify command	Verify the user custom key.
50h	Initialize command	Initialize all the memory areas.
4Eh	Boundary setting command	Set the boundary.
4Fh	Boundary request command	Get the boundary setting.
51h	Parameter setting command	Set the parameter.
52h	Parameter request command	Get the parameter setting.
4Ah	Lock bit setting command	Set the Lock bit.
4Bh	Lock bit request command	Get the Lock bit setting.
4Ch	ARC configuration setting command	Set the Anti-Rollback Counter configuration.
4Dh	ARC configuration request command	Get the Anti-Rollback Counter configuration.
00h	Inquiry command	Return ACK.
3Ah	Signature request command	Get the signature information.
3Bh	Area information request command	Get the area information.
34h	Baudrate setting command	Set baudrate (only UART).

Value	Name	Description
12h	Erase command	Erase data on target area.
13h	Write command	Write data to target area.
15h	Read command	Read data from target area
18h	CRC command	Cyclic Redundancy Check of target area.
2Eh	OEM root public key setting command	Insert the encrypted hash of root public key.
2Fh	OEM root public key verify command	Verify the encrypted hash of root public key
26h	Code certificate update command	Update the code certificate.
27h	Code certificate check command	Check the code certificate.
38h	SiP Flash various setting command	Execute various setting regarding SiP Flash
39h	SiP Flash various request command	Get various information regarding SiP Flash
36h	External flash memory setting command	Set the external flash memory.
1Ah	Encrypted data write command	Write encrypted data to target area.

5.1.5 RES: Response Code

Table 13. Response Codes

Value	Name	Description
00h | CMD	OK (ongoing normally)	-
80h | CMD	ERR (occurrence of an error)	-

5.1.6 STS: Status Code

Table 14. Status Codes

Value	Name	Description	Notes
00h	Communication is normal [OK]	-	
C0h	Unsupported command error	Received an unsupported command.	(*1)
C1h	Packet error	Abnormality of packet format.	(*1)
C2h	Checksum error	Abnormality of packet's checksum value.	(*1)
D0h	Parameter error	Abnormality of packet parameter.	(*1)
D2h	Invalid address error	Invalid address in the current boundary settings.	(*1)
D3h	Certificate storage error	Certificate storage area is invalid.	(*1)
D5h	Command acceptance error	A command cannot execute in current state.	(*1)
D6h	DLM state unmatched error	Device reset is not asserted after DLM state is changed.	(*1)
D7h	Hardware error	Abnormality of memory value.	(*1)
DAh	Protection error	Accessing protected areas or performing prohibited actions.	(*1)
DBh	Trusted system error	Abnormality from the Trusted system.	(*1)
DCh	Boot loader version error	Abnormality of OEM boot loader version.	(*1)
E4h	Secure error	Access to an area which is inaccessible with current privilege.	(*1)
E5h	Flash access error	Abnormality from the external flash memory access driver.	(*1), (*2), (*3)
E8h	Verify error	Verification of the written data fails.	(*1)
E9h	SiP Flash access error	Abnormality from the SiP Flash memory.	(*1), (*2)
EDh	MRAM sequencer error	MACI command abnormally terminates.	(*1), (*2), (*3),
EEh	MRAM write error	MRAM write without MACI commands abnormally terminates.	(*1), (*2)
E7h	Flash initialization error	Flash memory initialization is abnormal.	(*1)

Notes:
1. When this error occurs, the response code (RES) is ERR.
2. The boot firmware also returns the Status details (ST2) and Failure address (ADR) as additional error information when abnormality from MRAM sequencer.
3. This error occurs when the MRAM sequencer enters the “command lock” state after execution of a MRAM sequencer command.

5.1.7 ST2: Status Details

Table 15. Status Details

Value	Name	Description
MSTATR [31:0]	MRAM status	When a MRAM sequencer error occurs, boot firmware returns the value of the MSTATR register. When not, boot firmware returns FFFFFFFFh. Boot firmware clears the MSTATR register after the status sending, so even when an error occurs, the host can retry the next command without reset release.
FFFFFF00h | Flag Status register [7:0]	SiP Flash status	When a SiP Flash access error occurs, boot firmware returns the value of the Flag Status register. When not, boot firmware returns FFFFFFFFh. Boot firmware clears the error status of SiP Flash after the status sending, so even when an error occurs, the host can retry the next command without reset release.
AAAA0000h – AAAAFFFFh	Trusted system status	When a Trusted system error occurs, boot firmware returns the following detailed information: • AAAA0100h: An invalid magic number is set. • AAAA0101h: Unsupported version is set. • AAAA0102h: Out of range TLV Length is set. • AAAA0103h: Missing required TLV field. • AAAA0104h: The length exceeding the end of the manifest is specified in Length of the TLV field. • AAAA0105h: An invalid image length is set. • AAAA0106h: There is a wrong combination of signature algorithms. • AAAA0200h: Cryptographic processing failure. • AAAA0201h: Verification failed. • AAAA0202h: Unsupported algorithm. • AAAA0204h: Parameter error. • AAAA0300h: CRC mismatch

5.1.8 ADR: Failure Address

Table 16. Failure Address

Value	Name	Description
00000000h–FFFFFFFFh	Failure address	When a MRAM sequencer error, a MRAM write error or a SiP Flash access error occurs, boot firmware returns the error occurrence address. When not, boot firmware returns FFFFFFFFh.

5.1.9 DLM: Device Lifecycle Management State Code

Table 17. DLM State Codes

Value	Name	Description
04h	OEM	Original Equipment Manufacturer
06h	LCK_BOOT	LoCKed BOOT interface
07h	RMA_REQ	Return Material Authorization REQuest
08h	RMA_ACK	Return Material Authorization ACKnowledged
09h	RMA_RET	Return Material Authorization RETurn

6. Command List

Table 18. Command List

Name	Communication Method	DLM State			Prerequisite Command
		OEM	LCK_BOOT	RMA_REQ	
DLM State Transit Command	2-wire UART, USB, JTAG/SWD	◎	(*1)		-
DLM State Request Command	2-wire UART, USB, JTAG/SWD	◎	(*1)	◎	-
Protection Level Transit Command	2-wire UART, USB, JTAG/SWD	◎	(*1)		-
Protection Level Request Command	2-wire UART, USB, JTAG/SWD	◎	(*1)	◎	-
Authentication Level Request Command	2-wire UART, USB, JTAG/SWD	◎	(*1)	◎	-
Authentication Command	2-wire UART, USB, JTAG/SWD	◎	(*1)	◎	-
Key Setting Command	2-wire UART, USB, JTAG/SWD	◎	(*1)		-
User Key Setting Command	2-wire UART, USB, JTAG/SWD	◎	(*1)		-
Key Verify Command	2-wire UART, USB, JTAG/SWD	◎	(*1)	◎	-
User Key Verify Command	2-wire UART, USB, JTAG/SWD	◎	(*1)	◎	-
Initialize Command	2-wire UART, USB, JTAG/SWD	◎	(*1)		-
Boundary Setting Command	2-wire UART, USB, JTAG/SWD	◎	(*1)		-
Boundary Request Command	2-wire UART, USB, JTAG/SWD	◎	(*1)	◎	-
Parameter Setting Command	2-wire UART, USB, JTAG/SWD	◎	(*1)		-
Parameter Request Command	2-wire UART, USB, JTAG/SWD	◎	(*1)	◎	-

ARC configuration setting command	2-wire UART, USB, JTAG/SWD	◎	(*1)		-
ARC configuration request command	2-wire UART, USB, JTAG/SWD	◎	(*1)		-
Inquiry command	2-wire UART, USB, JTAG/SWD	◎	(*1)	◎	-
Signature request command	2-wire UART, USB, JTAG/SWD	◎	(*1)	◎	-
Area information request command	2-wire UART, USB, JTAG/SWD	◎	(*1)	◎	-
Baudrate setting command	2-wire UART, USB, JTAG/SWD	◎	(*1)	◎	-
Erase command	2-wire UART, USB, JTAG/SWD	◎	(*1)		-
Write command	2-wire UART, USB, JTAG/SWD	◎	(*1)		-
Read command	2-wire UART, USB, JTAG/SWD	◎	(*1)		-
CRC command	2-wire UART, USB, JTAG/SWD	◎	(*1)	◎	-
OEM root public key setting command	2-wire UART, USB, JTAG/SWD	◎	(*1)		-
OEM root public key verify command	2-wire UART, USB, JTAG/SWD	◎	(*1)	◎	-
Code certificate update command	2-wire UART, USB, JTAG/SWD	◎	(*1)		-
Code certificate check command	2-wire UART, USB, JTAG/SWD	◎	(*1)	◎	-
SiP Flash various setting command	2-wire UART, USB, JTAG/SWD	◎	(*1)		-

SiP Flash various request command	2-wire UART, USB, JTAG/SWD	◎	(*1)		-
External flash memory setting command	2-wire UART, USB, JTAG/SWD	◎	(*1)		-
Encrypted data write command	2-wire UART, USB, JTAG/SWD	◎	(*1)		-

◎ : Command is available in the state. (If an unavailable command is sent, boot firmware returns “Command acceptance error”.)
*1: LCK_BOOT state never transits to Command acceptable phase because boot firmware executes software reset in the Initialization phase.

6.1 Device Lifecycle Management

The following DLM state transitions can be caused by each command:

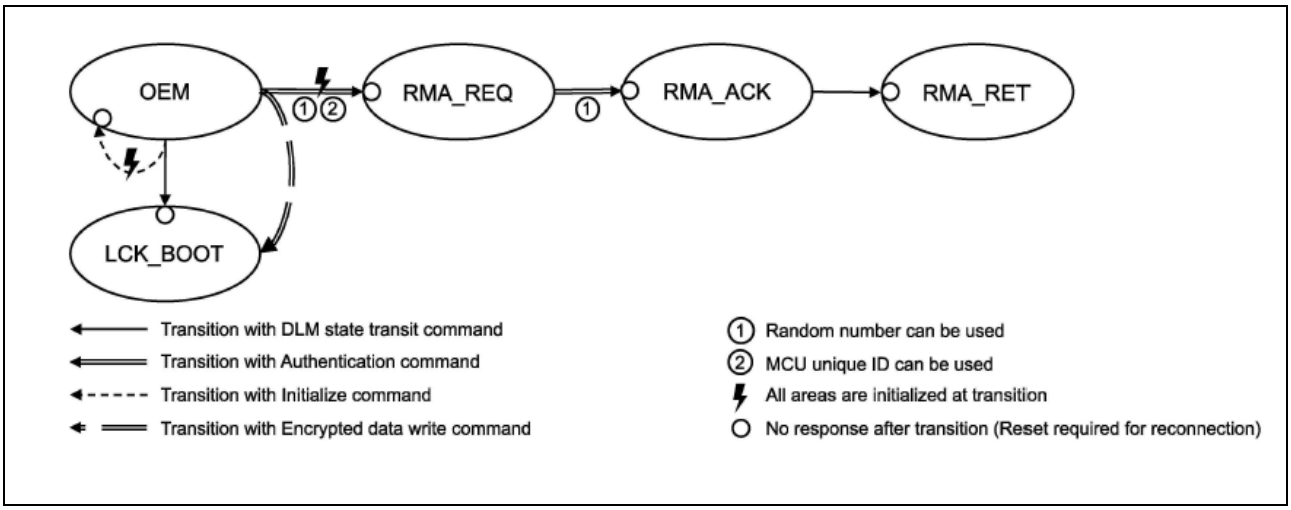


Figure 14. DLM State Transitions

6.2 DLM State Transit Command

This command transitions the DLM state without authentication.

Boot firmware will enter an infinite loop when the DLM state transitions to LCK_BOOT or RMA_RET.

This command requires adherence to conditions described in Command List.

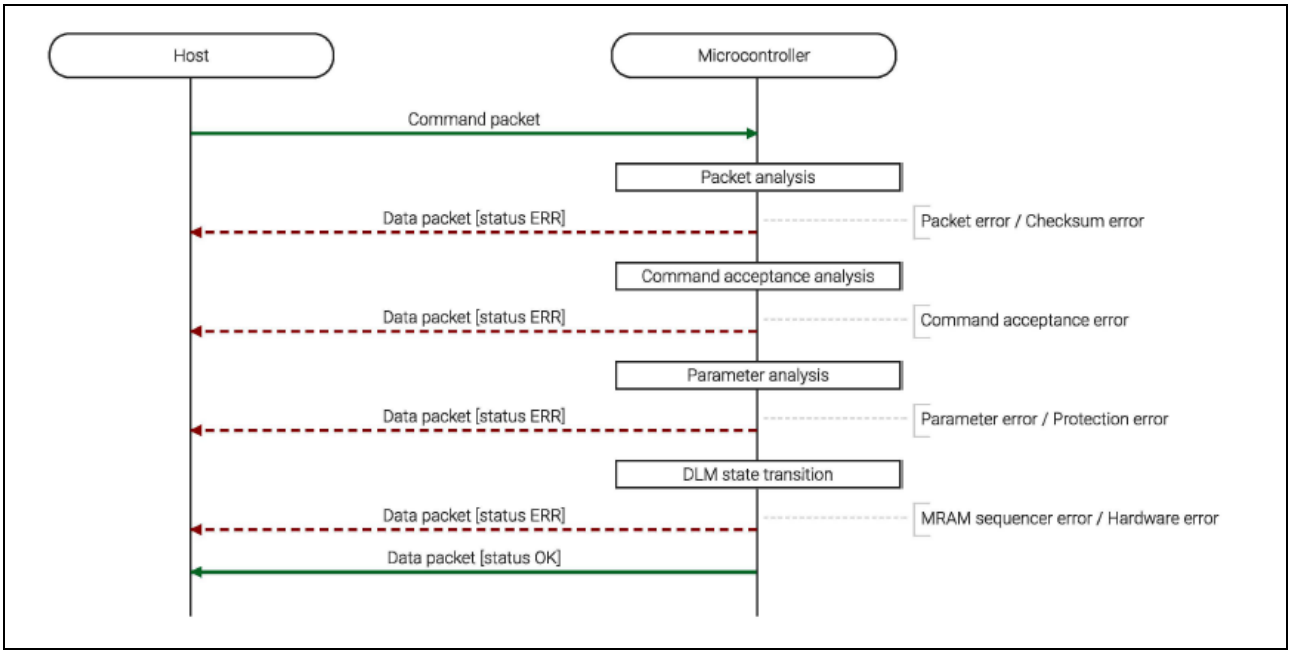


Figure 15. DLM State Transit Command Sequence Diagram

6.2.1 Packets

6.2.1.1 Command Packet

SOH	(1 byte)	01h
LNH	(1 byte)	00h
LNL	(1 byte)	03h
CMD	(1 byte)	71h (DLM state transit command)
SDLM	(1 byte)	Source DLM state code: • 04h: OEM • 08h: RMA_ACK
DDLM	(1 byte)	Destination DLM state code: • 06h: LCK_BOOT • 09h: RMA_RET
SUM	(1 byte)	Sum data
ETX	(1 byte)	03h

6.2.1.2 Data Packet [Status OK]

SOD	(1 byte)	81h
LNH	(1 byte)	00h
LNL	(1 byte)	0Ah
RES	(1 byte)	71h (OK)
STS	(1 byte)	00h (OK)
ST2	(4 bytes)	FFFFFFFFh (unused code)
ADR	(4 bytes)	FFFFFFFFh (unused code)
SUM	(1 byte)	8Dh
ETX	(1 byte)	03h

6.2.1.3 Data Packet [Status ERR]

SOD	(1 byte)	81h
LNH	(1 byte)	00h
LNL	(1 byte)	0Ah
RES	(1 byte)	F1h (ERR)
STS	(1 byte)	Status code
ST2	(4 bytes)	Status details
ADR	(4 bytes)	Failure address
SUM	(1 byte)	Sum data
ETX	(1 byte)	03h

6.2.2 Processing Procedure

Boot firmware receives and analyzes a command packet:

- The boot firmware recognizes the start of the command packet by receiving SOH.
  If the boot firmware receives something other than SOH, it waits until it receives SOH.
- If ETX is not added to the received command packet, the boot firmware sends a “Packet error”.
- If the SUM of the received command packet is different from the sum value, the boot firmware sends a “Checksum error”.
- If the LNH and LNL of the received command packet are different from the values specified in the packet format, the boot firmware sends a “Packet error”.
- If the LNH and LNL of the received command packet are different from the values specified in each command, the boot firmware sends a “Packet error”.
- When any of the above errors occurs, the boot firmware does not process and returns to the command waiting state.
  * Memory contents do not change before command reception.

When the processing above is successfully completed, boot firmware executes the acceptance analysis:

- If this command cannot be executed in the current DLM state, the boot firmware sends a “Command acceptance error”.
- If the device reset is not asserted after the Encrypted data write command execution, the boot firmware sends a “Command acceptance error”.
- When any of the above errors occurs, the boot firmware does not process and returns to the command waiting state.
  * Memory contents do not change before command reception.

When the processing above is successfully completed, boot firmware analyzes the command parameters:

- When SDLM is different from the current DLM state, boot firmware returns “Parameter error”.
- When DDLM is a DLM state that cannot be entered from the current DLM state without authentication, boot firmware returns “Parameter error”.
- If LCK_BOOT is specified for DDLM while the transition to LCK_BOOT is disabled, boot firmware returns “Protection error”.
- When any of the above errors occurs, the boot firmware does not process and returns to the command waiting state.
  * Memory contents do not change before command reception.

When the processing above is successfully completed, boot firmware transitions to the DLM state.

- If an error occurs during DLM state transitioning, boot firmware returns "MRAM sequencer error" and waits for the next command.
  * Check the DLM state after the error has occurred with the DLM state request command.
- If the DLM state after the transition is an invalid value, the boot firmware sends a "Hardware error" and becomes unresponsive.
  Also, if the DLM state after transition is LCK_BOOT or RMA_RET, the boot firmware sends "OK" and does not respond.
- When DLM state transit successful completion, "OK" is returned and the boot firmware waits for the next command.

6.2.3 Status Information from the Microcontroller

(Listed in descending order of priority.)

Condition	STS	ST2	ADR
The received packet does not have ETX.	Packet error	FFFFFFFFh	FFFFFFFFh
Sum data in the packet received is different from the value calculated by the boot firmware.	Checksum error	FFFFFFFFh	FFFFFFFFh
Packet length in the received packet does not comply with the packet format.	Packet error	FFFFFFFFh	FFFFFFFFh
Packet length in the received packet does not comply with the specifications of this command.	Packet error	FFFFFFFFh	FFFFFFFFh
Executing this command is unavailable in the current DLM state.	Command acceptance error	FFFFFFFFh	FFFFFFFFh
Device reset is not asserted after Encrypted data write command execution.	Command acceptance error	FFFFFFFFh	FFFFFFFFh
Source DLM state code is different from the current DLM state.	Parameter error	FFFFFFFFh	FFFFFFFFh
Destination DLM state code is not a transitional DLM state.	Parameter error	FFFFFFFFh	FFFFFFFFh
LCK_BOOT was specified for the Destination DLM state code with the transition to LCK_BOOT disabled.	Protection error	FFFFFFFFh	FFFFFFFFh
MACI detected an error after the command execution in an undisclosed area.	MRAM sequencer error	MRAM status	FFFFFFFFh
DLM state is abnormal.	Hardware error	FFFFFFFFh	FFFFFFFFh
Protection level is abnormal.	Hardware error	FFFFFFFFh	FFFFFFFFh
Successful completion.	OK	FFFFFFFFh	FFFFFFFFh

6.2.4 DLM State Transition

Figure 16 shows the DLM states that can be transitioned by the DLM State Transit command

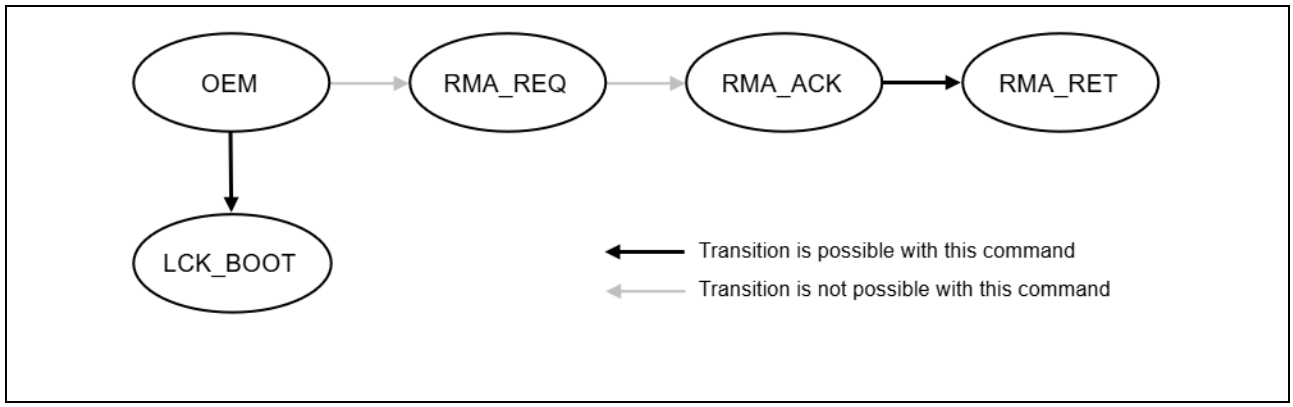


Figure 16. Valid State Transitions for DLM State Transit Command

Source DLM	Destination DLM	Requirements for transition
OEM	LCK_BOOT	Transition to LCK_BOOT (Parameter ID: 02h) is enabled.
RMA_ACK	RMA_RET	-

6.3 DLM State Request Command

This command is used to get the current DLM state.

This command requires adherence to conditions described in Command List.

6.3.1 Sequence Diagram

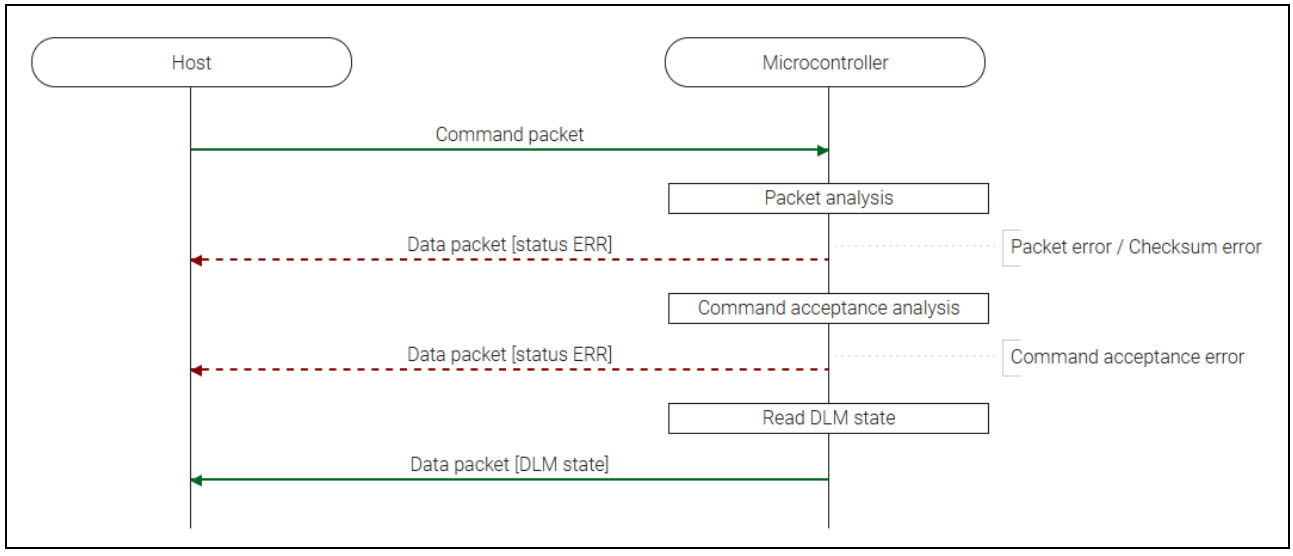


Figure 17. DLM State Request Command Sequence Diagram

6.3.2 Packets

6.3.2.1 Command Packet

SOH	(1 byte)	01h
LNH	(1 byte)	00h
LNL	(1 byte)	01h
CMD	(1 byte)	2Ch (DLM state request command)
SUM	(1 byte)	D3h
ETX	(1 byte)	03h

6.3.2.2 Data Packet [DLM State]

SOD	(1 byte)	81h
LNH	(1 byte)	00h
LNL	(1 byte)	02h
RES	(1 byte)	2Ch (OK)
DLM	(1 byte)	DLM state code
SUM	(1 byte)	Sum data
ETX	(1 byte)	03h

6.3.2.3 Data Packet [Status ERR]

SOD	(1 byte)	81h
LNH	(1 byte)	00h
LNL	(1 byte)	0Ah
RES	(1 byte)	ACh (ERR)
STS	(1 byte)	Status code
ST2	(4 bytes)	Status details
ADR	(4 bytes)	Failure address
SUM	(1 byte)	Sum data
ETX	(1 byte)	03h

6.3.3 Processing Procedure

Boot firmware receives and analyzes a command packet:

- The boot firmware recognizes the start of the command packet by receiving SOH. If the boot firmware receives something other than SOH, it will wait until it receives SOH.
- If ETX is not added to the received command packet, the boot firmware sends a “Packet error”.
- If the SUM of the received command packet is different from the sum value, the boot firmware sends a “Checksum error”.
- If the LNH and LNL of the received command packet are different from the values specified in the packet format, the boot firmware sends a “Packet error”.
- If the LNH and LNL of the received command packet are different from the values specified in each command, the boot firmware sends a “Packet error”.
- When any of the above errors occurs, the boot firmware does not process and returns to the command waiting state.
  * Memory contents do not change before command reception.

When the processing above is successfully completed, boot firmware executes the acceptance analysis:

- If the device reset is not asserted after the Encrypted data write command execution, the boot firmware sends a “Command acceptance error”.
- When the above error occurs, the boot firmware does not process and returns to the command waiting state.
  * Memory contents do not change before command reception.

When the processing above is successfully completed, boot firmware returns the current DLM state:

- Send DLM state and return to command wait state.
  * Memory contents do not change before command reception.

6.3.4 Status Information from the Microcontroller

(Listed in descending order of priority.)

Condition	STS	ST2	ADR
The received packet does not have ETX.	Packet error	FFFFFFFFh	FFFFFFFFh
Sum data in the packet received is different from the value calculated by the boot firmware.	Checksum error	FFFFFFFFh	FFFFFFFFh
Packet length in the received packet does not comply with the packet format.	Packet error	FFFFFFFFh	FFFFFFFFh
Packet length in the received packet does not comply with the specifications of this command.	Packet error	FFFFFFFFh	FFFFFFFFh
Device reset is not asserted after Encrypted data write command execution.	Command acceptance error	FFFFFFFFh	FFFFFFFFh

6.4 Protection Level Transit Command

This command transitions to the Protection level.

This command requires adherence to conditions described in Command List.

6.4.1 Sequence Diagram

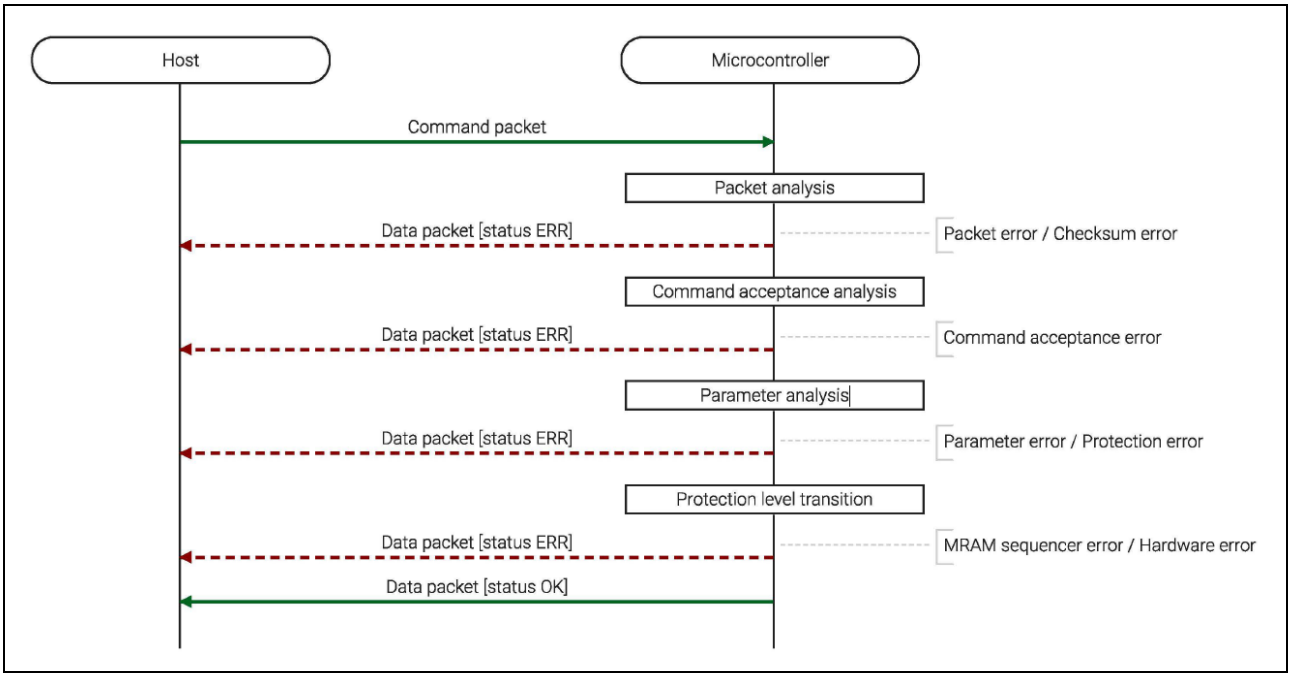


Figure 18. Protection Level Transit Command Sequence Diagram

6.4.2 Packets

6.4.2.1 Command Packet

SOH	(1 byte)	01h
LNH	(1 byte)	00h
LNL	(1 byte)	03h
CMD	(1 byte)	72h (Protection level transit command)
SPL	(1 byte)	Source PL code: • 02h: Protection level 2 • 03h: Protection level 1 • 04h: Protection level 0
DPL	(1 byte)	Destination PL code: • 02h: Protection level 2 • 03h: Protection level 1 • 04h: Protection level 0
SUM	(1 byte)	Sum data
ETX	(1 byte)	03h

6.4.2.2 Data Packet [Status OK]

SOD	(1 byte)	81h
LNH	(1 byte)	00h
LNL	(1 byte)	0Ah
RES	(1 byte)	72h (OK)
STS	(1 byte)	00h (OK)
ST2	(4 bytes)	FFFFFFFFh (unused code)
ADR	(4 bytes)	FFFFFFFFh (unused code)
SUM	(1 byte)	8Ch
ETX	(1 byte)	03h

6.4.2.3 Data Packet [Status ERR]

SOD	(1 byte)	81h
LNH	(1 byte)	00h
LNL	(1 byte)	0Ah
RES	(1 byte)	F2h (ERR)
STS	(1 byte)	Status code
ST2	(4 bytes)	Status details
ADR	(4 bytes)	Failure address
SUM	(1 byte)	Sum data
ETX	(1 byte)	03h

6.4.2.4 Processing Procedure

Boot firmware receives and analyzes a command packet:

- The boot firmware recognizes the start of the command packet by receiving SOH. If the boot firmware receives something other than SOH, it waits until it receives SOH.
- If ETX is not added to the received command packet, the boot firmware sends a “Packet error”.
- If the SUM of the received command packet is different from the sum value, the boot firmware sends a “Checksum error”.
- If the LNH and LNL of the received command packet are different from the values specified in the packet format, the boot firmware sends a “Packet error”.
- If the LNH and LNL of the received command packet are different from the values specified in each command, the boot firmware sends a “Packet error”.
- When any of the above errors occurs, the boot firmware does not process and returns to the command waiting state.
  * Memory contents do not change before command reception.

When the processing above is successfully completed, boot firmware executes the acceptance analysis:

- If this command cannot be executed in the current DLM state, the boot firmware sends a “Command acceptance error”.
- If the device reset is not asserted after the Encrypted data write command execution, the boot firmware sends a “Command acceptance error”.
- When any of the above errors occurs, the boot firmware does not process and returns to the command waiting state.
  * Memory contents do not change before command reception.

When the processing above is successfully completed, boot firmware analyzes the command parameters:

- When SPL is different from the current Protection level, boot firmware returns “Parameter error”.
- When DPL is a Protection level that cannot be transited to from the current Protection level, boot firmware returns “Parameter error”.
- If it is not allowed to transit to the specified DPL in the current Authentication level, boot firmware returns “Protection error”.
- When any of the above errors occurs, the boot firmware does not process and returns to the command waiting state.
  * Memory contents do not change before command reception.

When the processing above is successfully completed, boot firmware transitions the Protection level:

- If an error occurs during transition of the Protection level, boot firmware returns “MRAM sequencer error” and waits for the next command.
  * Check the Protection level after the error has occurred with the Protection level request command.
- If the Protection level after the transition is an invalid value, the boot firmware sends a “Hardware error” and becomes unresponsive.
- When the Protection level transit successfully completes, “OK” is returned and the boot firmware waits for the next command.

6.4.3 Status Information from the Microcontroller

(Listed in descending order of priority.)

Condition	STS	ST2	ADR
The received packet does not have ETX.	Packet error	FFFFFFFFh	FFFFFFFFh
Sum data in the packet received is different from the value calculated by the boot firmware.	Checksum error	FFFFFFFFh	FFFFFFFFh
Packet length in the received packet does not comply with the packet format.	Packet error	FFFFFFFFh	FFFFFFFFh
Packet length in the received packet does not comply with the specifications of this command.	Packet error	FFFFFFFFh	FFFFFFFFh
Executing this command is unavailable in the current DLM state.	Command acceptance error	FFFFFFFFh	FFFFFFFFh
Device reset is not asserted after Encrypted data write command execution.	Command acceptance error	FFFFFFFFh	FFFFFFFFh
Source PL code is different from current Protection level.	Parameter error	FFFFFFFFh	FFFFFFFFh
Destination PL code is not transitional Protection level.	Parameter error	FFFFFFFFh	FFFFFFFFh
Does not meet the Authentication level required for transition to the specified destination PL code.	Protection error	FFFFFFFFh	FFFFFFFFh
MACI detected an error after the command execution in an undisclosed area.	MRAM sequencer error	MRAM status	FFFFFFFFh
Protection level is abnormal.	Hardware error	FFFFFFFFh	FFFFFFFFh
Successful completion.	OK	FFFFFFFFh	FFFFFFFFh

6.4.4 Protection Level Transition

Figure 19 shows the Protection level that can be transited by this command.

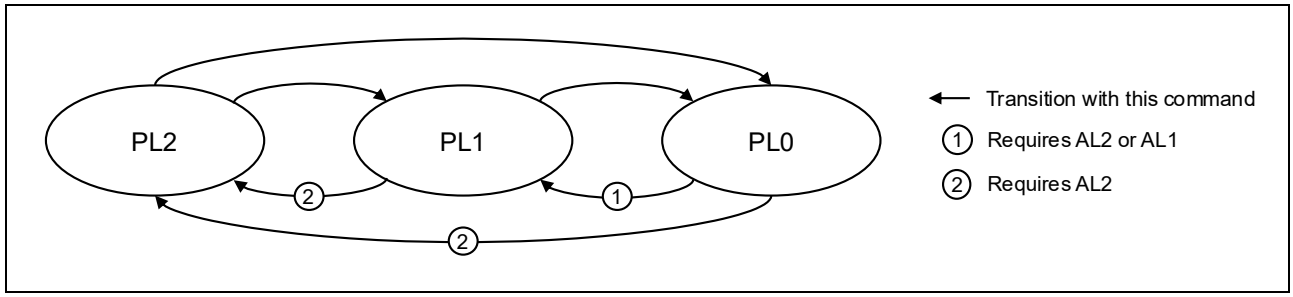


Figure 19. Valid Protection Level Transitions

Source PL	Destination PL	Current Authentication level		
		AL2	AL1	AL0
PL0	PL1	OK	OK	Protection error
	PL2		Protection error	
PL1	PL0		OK	N/A (Impossible combination)
	PL2		Protection error	
PL2	PL0		N/A (Impossible combination)	
	PL1			

6.5 Protection Level Request Command

This command is used to get the current Protection level.

This command requires adherence to conditions described in Command List.

6.5.1 Sequence Diagram

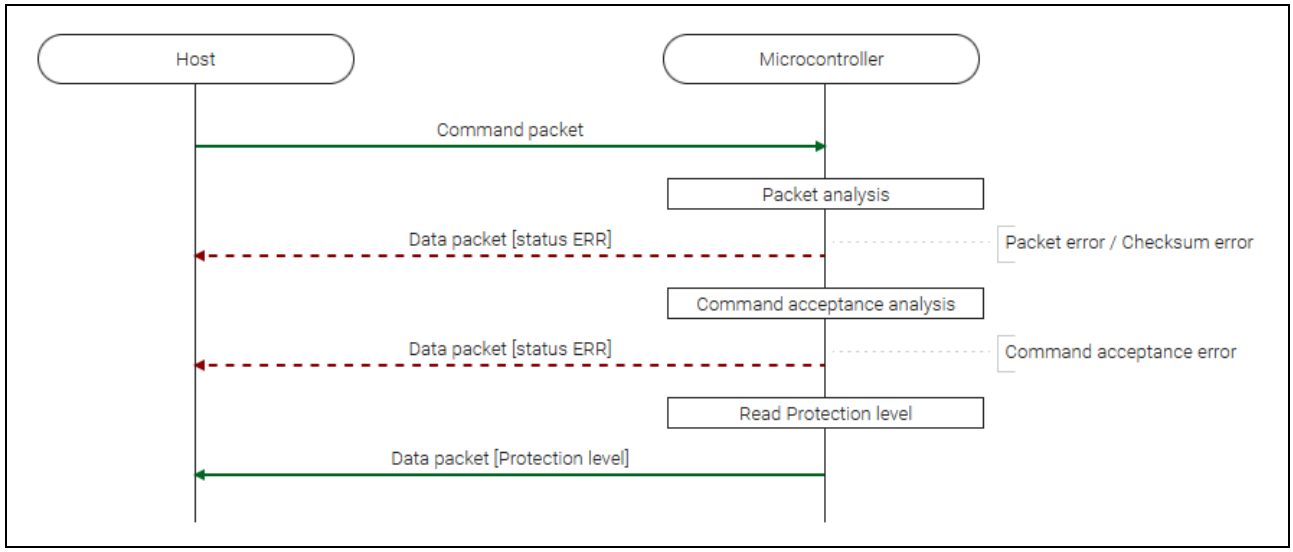


Figure 20. Protection Level Request Sequence Diagram

6.5.2 Packets

6.5.2.1 Command Packet

SOH	(1 byte)	01h
LNH	(1 byte)	00h
LNL	(1 byte)	01h
CMD	(1 byte)	73h (Protection level request command)
SUM	(1 byte)	8Ch
ETX	(1 byte)	03h

6.5.2.2 Data Packet [Protection Level]

SOD	(1 byte)	81h
LNH	(1 byte)	00h
LNL	(1 byte)	02h
RES	(1 byte)	73h (OK)
CPL	(1 byte)	Current PL code • 02h: Protection level 2 • 03h: Protection level 1 • 04h: Protection level 0
SUM	(1 byte)	Sum data
ETX	(1 byte)	03h

6.5.2.3 Data Packet [Status ERR]

SOD	(1 byte)	81h
LNH	(1 byte)	00h
LNL	(1 byte)	0Ah
RES	(1 byte)	F3h (ERR)
STS	(1 byte)	Status code
ST2	(4 bytes)	Status details
ADR	(4 bytes)	Failure address
SUM	(1 byte)	Sum data
ETX	(1 byte)	03h

6.5.3 Processing Procedure

Boot firmware receives and analyzes a command packet:

- The boot firmware recognizes the start of the command packet by receiving SOH.
  If the boot firmware receives something other than SOH, it waits until it receives SOH.
- If ETX is not added to the received command packet, the boot firmware sends a “Packet error”.
- If the SUM of the received command packet is different from the sum value, the boot firmware sends a “Checksum error”.
- If the LNH and LNL of the received command packet are different from the values specified in the packet format, the boot firmware sends a “Packet error”.
- If the LNH and LNL of the received command packet are different from the values specified in each command, the boot firmware sends a “Packet error”.
- When any of the above errors occurs, the boot firmware does not process and returns to the command waiting state.
  * Memory contents do not change before command reception.

When the processing above is successfully completed, boot firmware executes the acceptance analysis:

- If device reset is not asserted after the Encrypted data write command execution, the boot firmware sends a “Command acceptance error”.
- When the above error occurs, the boot firmware does not process and returns to the command waiting state.
  * Memory contents do not change before command reception.

When the processing above is successfully completed, boot firmware returns Protection level:

- Send Protection level and return to command wait state.
  * Memory contents do not change before command reception.

6.5.4 Status Information from the Microcontroller

(Listed in descending order of priority.)

Condition	STS	ST2	ADR
The received packet does not have ETX.	Packet error	FFFFFFFFh	FFFFFFFFh
Sum data in the packet received is different from the value calculated by the boot firmware.	Checksum error	FFFFFFFFh	FFFFFFFFh
Packet length in the received packet does not comply with the packet format.	Packet error	FFFFFFFFh	FFFFFFFFh
Packet length in the received packet does not comply with the specifications of this command.	Packet error	FFFFFFFFh	FFFFFFFFh
Device reset is not asserted after Encrypted data write command execution.	Command acceptance error	FFFFFFFFh	FFFFFFFFh

6.6 Authentication Level Request Command

This command is used to get the current Authentication level.

This command requires adherence to conditions described in Command List.

6.6.1 Sequence Diagram

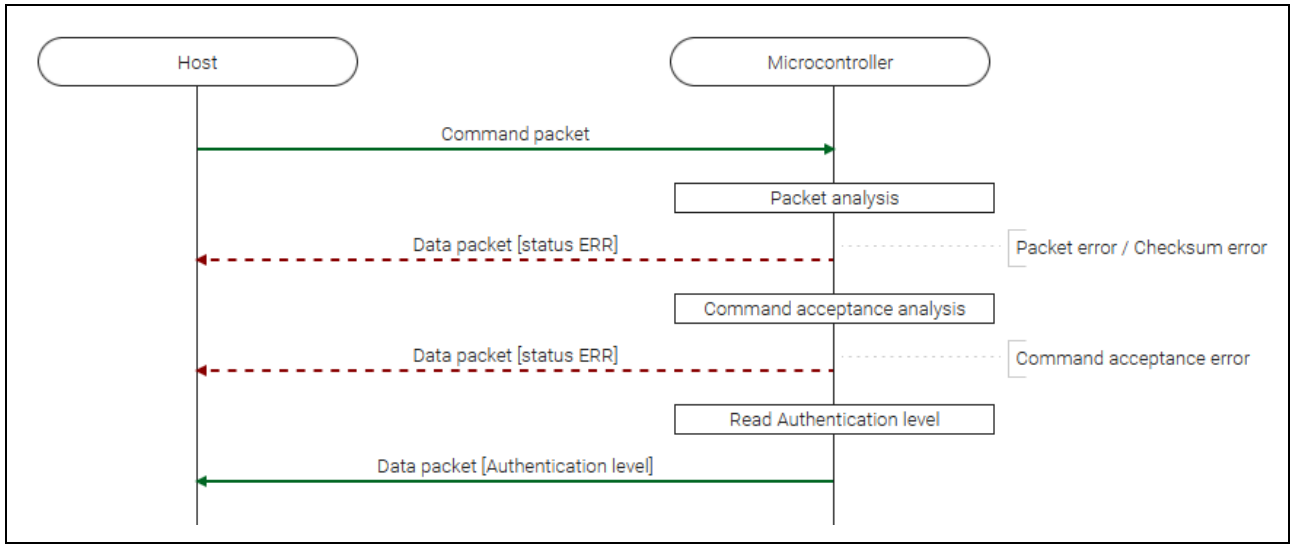


Figure 21. Authentication Level Request Command Sequence Diagram

6.6.2 Packets

6.6.2.1 Command Packet

SOH	(1 byte)	01h
LNH	(1 byte)	00h
LNL	(1 byte)	01h
CMD	(1 byte)	75h (Authentication level request command)
SUM	(1 byte)	8Ah
ETX	(1 byte)	03h

6.6.2.2 Data Packet [Authentication Level]

SOD	(1 byte)	81h
LNH	(1 byte)	00h
LNL	(1 byte)	02h
RES	(1 byte)	75h (OK)
CAL	(1 byte)	Current AL code • 02h: Authentication level 2 • 03h: Authentication level 1 • 04h: Authentication level 0
SUM	(1 byte)	Sum data
ETX	(1 byte)	03h

6.6.2.3 Data Packet [Status ERR]

SOD	(1 byte)	81h
LNH	(1 byte)	00h
LNL	(1 byte)	0Ah
RES	(1 byte)	F5h (ERR)
STS	(1 byte)	Status code
ST2	(4 bytes)	Status details
ADR	(4 bytes)	Failure address
SUM	(1 byte)	Sum data
ETX	(1 byte)	03h

6.6.3 Processing procedure

Boot firmware receives and analyzes a command packet:

- The boot firmware recognizes the start of the command packet by receiving SOH.
  If the boot firmware receives something other than SOH, it waits until it receives SOH.
- If ETX is not added to the received command packet, the boot firmware sends a “Packet error”.
- If the SUM of the received command packet is different from the sum value, the boot firmware sends a “Checksum error”.
- If the LNH and LNL of the received command packet are different from the values specified in the packet format, the boot firmware sends a “Packet error”.
- If the LNH and LNL of the received command packet are different from the values specified in each command, the boot firmware sends a “Packet error”.
- When any of the above errors occurs, the boot firmware does not process and returns to the command waiting state.
  * Memory contents do not change before command reception.

When the processing above is successfully completed, boot firmware executes the acceptance analysis:

- If the device reset is not asserted after the Encrypted data write command execution, the boot firmware sends a “Command acceptance error”.
- When the above error occurs, the boot firmware does not process and returns to the command waiting state.
  * Memory contents do not change before command reception.

When the processing above is successfully completed, boot firmware returns Authentication level:

- Send Authentication level and return to the command wait state.
  * Memory contents do not change before command reception.

6.6.4 Status Information from the Microcontroller

(Listed in descending order of priority.)

Condition	STS	ST2	ADR
The received packet does not have ETX.	Packet error	FFFFFFFFh	FFFFFFFFh
Sum data in the packet received is different from the value calculated by the boot firmware.	Checksum error	FFFFFFFFh	FFFFFFFFh
Packet length in the received packet does not comply with the packet format.	Packet error	FFFFFFFFh	FFFFFFFFh
Packet length in the received packet does not comply with the specifications of this command.	Packet error	FFFFFFFFh	FFFFFFFFh
Device reset is not asserted after Encrypted data write command execution.	Command acceptance error	FFFFFFFFh	FFFFFFFFh

6.7 Authentication Command

This command authenticates using a key and transitions the DLM state or the Authentication level.

Authentication is executed by the challenge and response method or Unique ID.

Boot firmware erases the memory when the DLM state transits to RMA_REQ. Erase processing at this time is not affected by the block protection settings (BPS, BPS_SEC). As for areas where erasure is not possible, erase processing here refers to writing the initial values.

This command requires adherence to conditions described in Command List.

6.7.1 Sequence Diagram

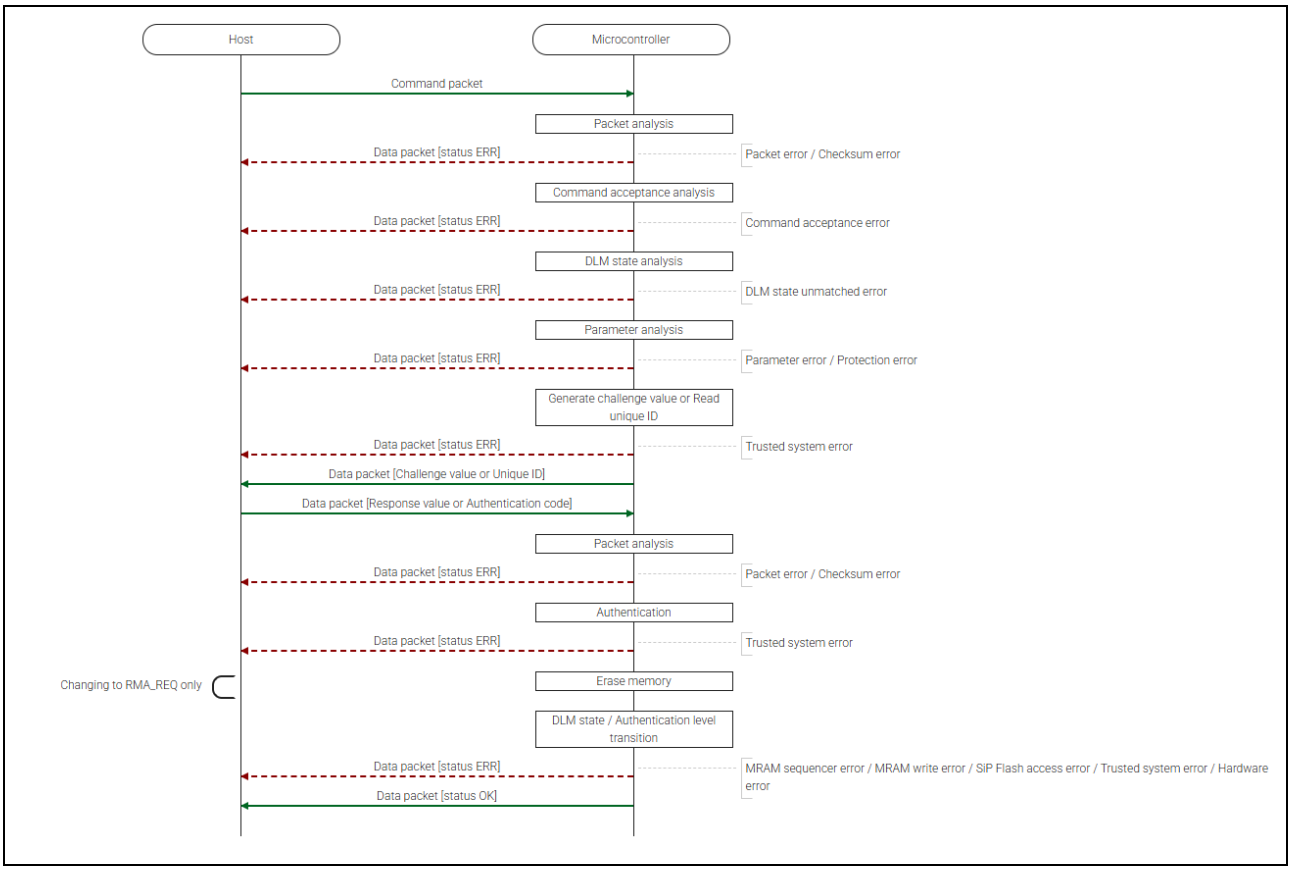


Figure 22. Authentication Command Sequence Diagram

6.7.2 Packets

6.7.2.1 Command Packet

SOH	(1 byte)	01h
LNH	(1 byte)	00h
LNL	(1 byte)	04h
CMD	(1 byte)	30h (Authentication command)
SDLM	(1 byte)	Source DLM/AL code. For DLM transitions: • 04h: OEM • 07h: RMA_REQ For AL transitions: • 03h: AL1 • 04h: AL0
DDLM	(1 byte)	Destination DLM/AL code. For DLM transitions: • 07h: RMA_REQ • 08h: RMA_ACK For AL transitions: • 02h: AL2 • 03h: AL1
CHCT	(1 byte)	Authentication type: • 00h: Random number (Can be used all transit cases.) • 01h: MCU unique ID (Can be used only transit to RMA_REQ.)
SUM	(1 byte)	Sum data
ETX	(1 byte)	03h

6.7.2.2 Data Packet [Challenge Value or Unique ID]

SOD	(1 byte)	81h
LNH	(1 byte)	00h
LNL	(1 byte)	11h
RES	(1 byte)	30h (OK)
CHCD	(16 bytes)	Challenge value or Unique ID For example: 01234567_89AB ... 2233_44556677h -> 01h, 23h, 45h, ... , 55h, 66h, 77h
SUM	(1 byte)	Sum data
ETX	(1 byte)	03h

6.7.2.3 Data Packet [Response Value or Authentication Code]

SOD	(1 byte)	81h
LNH	(1 byte)	00h
LNL	(1 byte)	21h
RES	(1 byte)	30h (OK)
MAC	(32 bytes)	Response value or Authentication code For example: 01234567_89AB ... 2233_44556677h -> 01h, 23h, 45h, ..., 55h, 66h, 77h For more detail on Response value, please refer to 6.7.7 Response Value Calculation.
SUM	(1 byte)	Sum data
ETX	(1 byte)	03h

6.7.2.4 Data Packet [Status OK]

SOD	(1 byte)	81h
LNH	(1 byte)	00h
LNL	(1 byte)	0Ah
RES	(1 byte)	30h (OK)
STS	(1 byte)	00h (OK)
ST2	(4 bytes)	FFFFFFFFh (unused code)
ADR	(4 bytes)	FFFFFFFFh (unused code)
SUM	(1 byte)	CEh
ETX	(1 byte)	03h

6.7.2.5 Data Packet [Status ERR]

SOD	(1 byte)	81h
LNH	(1 byte)	00h
LNL	(1 byte)	0Ah
RES	(1 byte)	B0h (ERR)
STS	(1 byte)	Status code
ST2	(4 bytes)	Status details
ADR	(4 bytes)	Failure address
SUM	(1 byte)	Sum data
ETX	(1 byte)	03h

6.7.3 Processing Procedure

Boot firmware receives and analyzes a command packet:

- The boot firmware recognizes the start of the command packet by receiving SOH.
  If the boot firmware receives something other than SOH, it waits until it receives SOH.
- If ETX is not added to the received command packet, the boot firmware sends a “Packet error”.
- If the SUM of the received command packet is different from the sum value, the boot firmware sends a “Checksum error”.
- If the LNH and LNL of the received command packet are different from the values specified in the packet format, the boot firmware sends a “Packet error”.
- If the LNH and LNL of the received command packet are different from the values specified in each command, the boot firmware sends a “Packet error”.
- When any of the above errors occurs, the boot firmware does not process and returns to the command waiting state.
  * Memory contents do not change before command reception.

When the processing above is successfully completed, boot firmware executes the acceptance analysis:

- If this command cannot be executed in the current DLM state, the boot firmware sends a “Command acceptance error”.
- If HUK has been zeroized, the boot firmware sends a “Command acceptance error”
- If the device reset is not asserted after the Encrypted data write command execution, the boot firmware sends a “Command acceptance error”.
- When any of the above errors occurs, the boot firmware does not process and returns to the command waiting state.
  * Memory contents do not change before command reception.

When the processing above is successfully completed, the boot firmware performs DLM state analysis:

- If the currently active DLM state does not match the stored DLM state, the boot firmware sends a “DLM state unmatched error”.
- When the above error occurs, the boot firmware does not process and returns to the command waiting state.
  * Memory contents do not change before command reception.

When the processing above is successfully completed, boot firmware analyzes the command parameters:

- When SDLM is different from the current DLM state or Authentication level, boot firmware returns “Parameter error”.
- When SDLM and DDLM are not a transitional combination, boot firmware returns “Parameter error”.
- When any of the following conditions is met, boot firmware returns “Protection error”:
  - Authentication with AL2_KEY is disabled and DDLM is RMA_REQ.
  - Authentication with AL2_KEY is disabled and DDLM is AL2.
  - Authentication with AL1_KEY is disabled and DDLM is AL1.
- When CHCT is not a challenge type that can be used to transition the DLM state, boot firmware returns “Parameter error”.
- When any of the above errors occurs, the boot firmware does not process and returns to the command waiting state.
  * Memory contents do not change before command reception.

When the processing above is successfully completed, boot firmware sends data packet [Challenge value or Unique ID]:

- If the Challenge value/Unique ID is successfully generated, the boot firmware sends the value.
- If the Trusted system becomes abnormal after the Challenge value/Unique ID generation, the boot firmware returns nothing and does not respond.
  * Memory contents do not change before command reception.
- If the Challenge value/Unique ID generation fails, the boot firmware sends a “Trusted system error” and returns to the command wait state.
  * Memory contents do not change before command reception.

Boot firmware receives and analyzes a data packet [Response value or Authentication code] after the processing above:

- Boot firmware detects the beginning of a data packet by receiving SOD.
  When boot firmware receives other data than SOD, it discards the data and waits for the next data until SOD is sent.
- When the received data packet does not have ETX, “Packet error” is returned.
- When SUM in the received data packet is different from the value calculated by boot firmware, “Checksum error” is returned.
- When LNH and LNL in the received data packet do not comply with the packet format, “Packet error” is returned.
- When RES in the received data packet is different from defined values, “Packet error” is returned.
- When LNH and LNL in the received data packet do not comply with the specifications of this command, “Packet error” is returned.
- When any of the above errors occurs, the boot firmware does not process and returns to the command waiting state.
  * Memory contents do not change before command reception.

When the processing above is successfully completed, boot firmware authenticates with received Response value or Authentication code:

- If the Trusted system becomes abnormal after authentication, the boot firmware returns nothing and does not respond.
  * Memory contents do not change before command reception.
- When authentication fails, “Trusted system error” is returned and the boot firmware waits for the next command.
  * Memory contents do not change before command reception.

When authentication is successfully completed and the DLM state transits to RMA_REQ, boot firmware erases the memory.
* This command initializes the memory even if initialization is disabled. (refer to the Parameter request command).

- If an error occurs during initialization, the boot firmware sends a “MRAM sequencer error” or “MRAM write error”; “SiP Flash access error” and returns to the command wait state. Also, if the Trusted system becomes abnormal after initialization of the Trusted system, the boot firmware returns nothing and no response.

When the Authentication is successfully completed (for transition to RMA_REQ, erase of memory is also successful), boot firmware executes transition:

- If the Trusted system becomes abnormal during transition, the boot firmware returns nothing and does not respond.
  * Check the DLM state after the error has occurred with the DLM state request command.
- If an error occurs during transition, boot firmware returns “MRAM sequencer error” or “Trusted system error” and waits for the next command.
  * Check the DLM state after the error has occurred with the DLM state request command.
- If the DLM state after the transition is an invalid value, the boot firmware sends a “Hardware error” and becomes unresponsive.
- If the above error does not occur, the boot firmware sends “OK” and becomes unresponsive (DLM transition) or waits for the next command (Authentication level transition).
  * When the DLM state transitions to RMA_REQ, each area of the memory is in the following state:
  - User area is all-FFh except for the following:
    - Blocks for which 0 is set for permanent block protection setting (PBPS, PBPS_SEC).
    * Not affected by block protection settings (BPS, BPS_SEC).
  - The Config area is written the value at shipment, except for the following:
    - Area protected by OFSPS.
    - Block protection setting (BPS, BPS_SEC) for blocks in which 0 is set for permanent block protection setting (PBPS, PBPS_SEC).

  The SiP Flash area is written the value at shipment, except for the following:
    - Status register when the permanent lock is set.
    - Area protected by BP, and also when the permanent lock of the Status register is set.
    - Area protected by something other than BP.

6.7.4 Status Information from the Microcontroller

(Listed in descending order of priority.)

Condition	STS	ST2	ADR
The received packet does not have ETX.	Packet error	FFFFFFFFh	FFFFFFFFh
Sum data in the packet received is different from the value calculated by the boot firmware.	Checksum error	FFFFFFFFh	FFFFFFFFh
Packet length in the received packet does not comply with the packet format.	Packet error	FFFFFFFFh	FFFFFFFFh
Packet length in the received packet does not comply with the specifications of this command.	Packet error	FFFFFFFFh	FFFFFFFFh
Executing this command is unavailable in the current DLM state.	Command acceptance error	FFFFFFFFh	FFFFFFFFh
HUK has been zeroized.	Command acceptance error	FFFFFFFFh	FFFFFFFFh
Device reset is not asserted after Encrypted data write command execution.	Command acceptance error	FFFFFFFFh	FFFFFFFFh
The currently active DLM state does not match the stored DLM state.	DLM state unmatched error	FFFFFFFFh	FFFFFFFFh
SDLM is different from current DLM state or AL.	Parameter error	FFFFFFFFh	FFFFFFFFh
SDLM and DDLM are not a transitional combination.	Parameter error	FFFFFFFFh	FFFFFFFFh
AL2 or RMA_REQ specified for DDLM with AL2_KEY disabled.	Protection error	FFFFFFFFh	FFFFFFFFh
AL1 specified for DDLM with AL1_KEY disabled.	Protection error	FFFFFFFFh	FFFFFFFFh
Authentication type is different from the value specified by this command.	Parameter error	FFFFFFFFh	FFFFFFFFh
Challenge value/Unique ID generation failed.	Trusted system error	FFFFFFFFh	FFFFFFFFh
The response code of the received data packet is different from the value specified by this command.	Packet error	FFFFFFFFh	FFFFFFFFh
Authentication failed.	Trusted system error	FFFFFFFFh	FFFFFFFFh
An error detected when MRAM programming which does not use MACI command.	MRAM write error	FFFFFFFFh	Failure address
MACI detected an error after the command execution in the disclosed area.	MRAM sequencer error	MRAM status	Failure address
MACI detected an error after the command execution in an undisclosed area.	MRAM sequencer error	MRAM status	FFFFFFFFh
An error occurred while erasing SiP Flash.	SiP Flash access error	SiP Flash status	Failure address
DLM state is abnormal.	Hardware error	FFFFFFFFh	FFFFFFFFh
Protection level is abnormal.	Hardware error	FFFFFFFFh	FFFFFFFFh
Successful completion.	OK	FFFFFFFFh	FFFFFFFFh

6.7.5 DLM State Transition

The following section shows the DLM state that can be transitioned by this command.

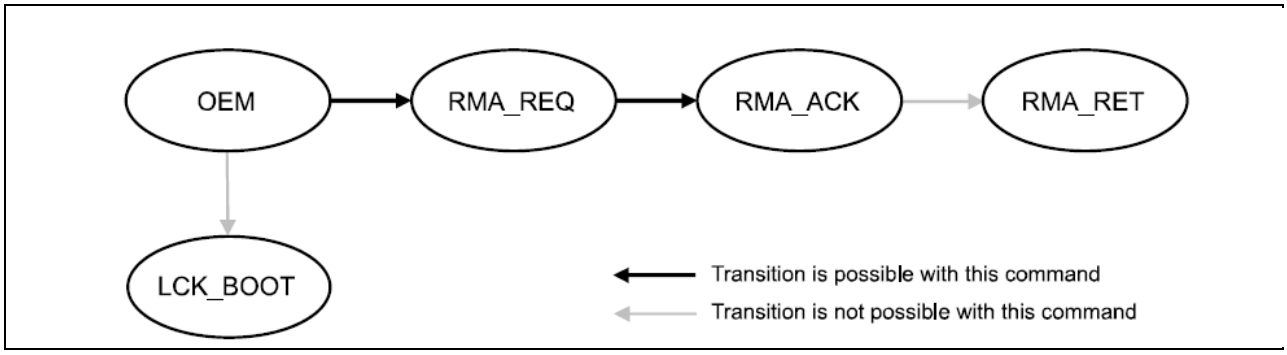


Figure 23. DLM State Transition

Source DLM state	Destination DLM state	Required key	Requirements for transition
OEM	RMA_REQ	RMA_KEY	Authentication using AL2_KEY (Parameter ID: 03h) is enabled.
RMA_REQ	RMA_ACK	RMA_ACK_KEY	

6.7.6 Authentication Level Transition

Figure 24 shows the Authentication level that can be transition by this command.

(Authentication level transition is possible only when DLM state is “OEM”.)

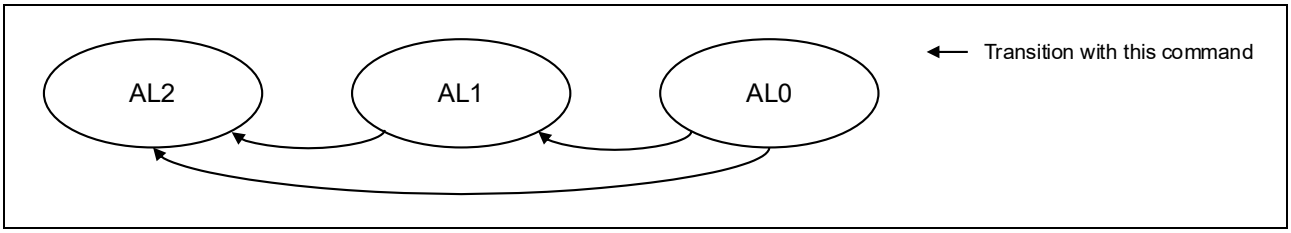


Figure 24. Valid Authentication Level Transitions

Source AL	Destination AL	Required key	Requirements for transition
AL0	AL1	AL1_KEY	Authentication using AL1_KEY (Parameter ID: 04h) is enabled.
	AL2	AL2_KEY	Authentication using AL2_KEY (Parameter ID: 03h) is enabled.
AL1	AL2		

6.7.7 Response Value Calculation

Response = AES -128 CMAC (Key, 128-bit challenge)

*Fill “1” to the lower 16 bytes of the MAC on the Data Packet because the calculated Response is 16 bytes.

6.8 Key Setting Command

This command sets the authentication key for the device. The authentication key must be specified in the DLM state that can set the key.

This command requires adherence to the conditions described in Command List.

6.8.1 Sequence Diagram

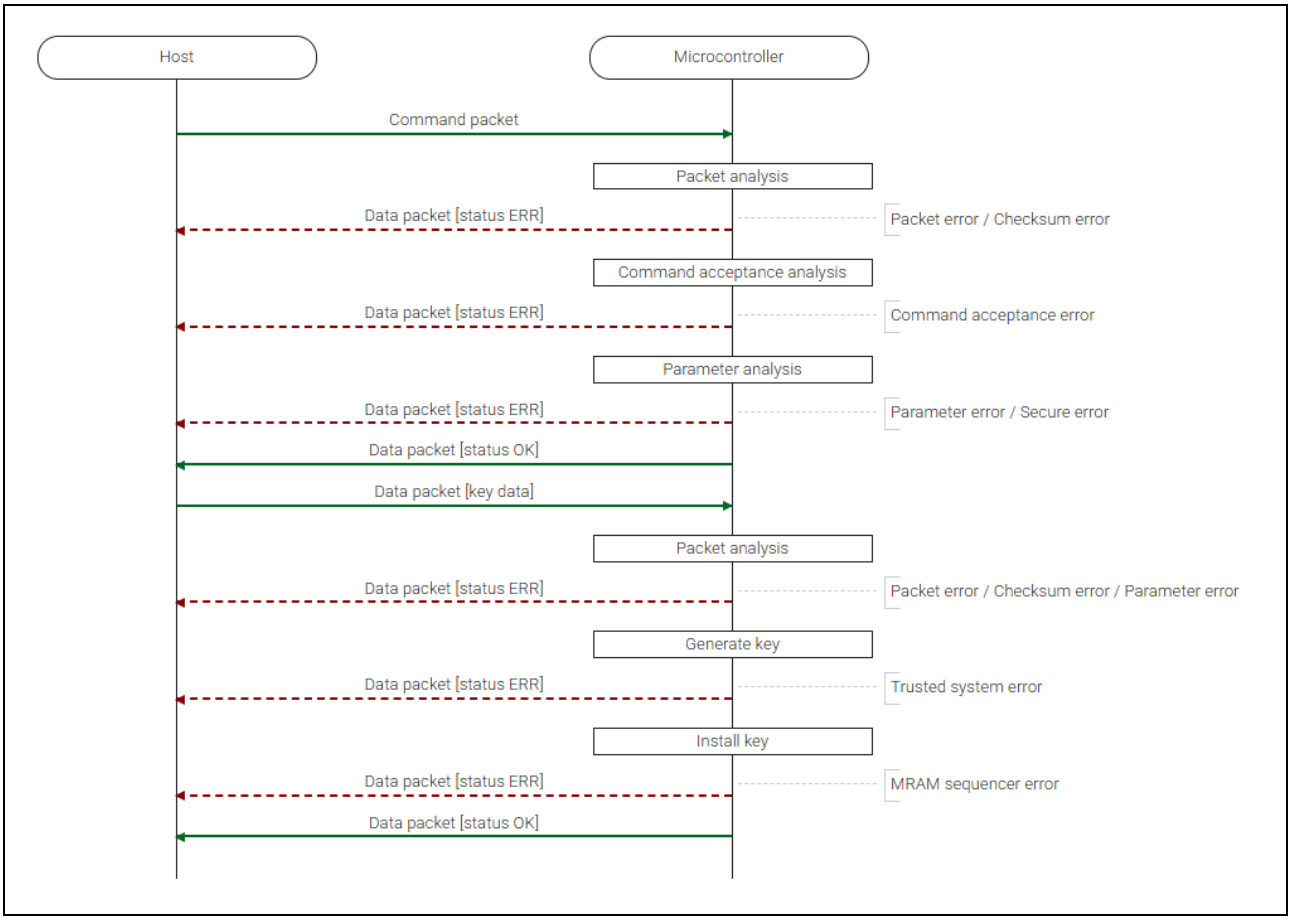


Figure 25. Key Setting Command Sequence Diagram

6.8.2 Packets

6.8.2.1 Command Packet

SOH	(1 byte)	01h
LNH	(1 byte)	00h
LNL	(1 byte)	02h
CMD	(1 byte)	28h (Key setting command)
KYTY	(1 byte)	Key type: • 01h: AL2_KEY • 02h: AL1_KEY • 03h: RMA_KEY
SUM	(1 byte)	Sum data
ETX	(1 byte)	03h

6.8.2.2 Data Packet [Key Data]

<table>
<tr><td>SOD</td><td>(1 byte)</td><td>81h</td></tr>
<tr><td>LNH</td><td>(1 byte)</td><td>00h</td></tr>
<tr><td>LNL</td><td>(1 byte)</td><td>55h</td></tr>
<tr><td>RES</td><td>(1 byte)</td><td>28h (OK)</td></tr>
<tr><td>SKR</td><td>(4 bytes)</td><td>Shared key ring number.
For example: 01234567h -> 01h, 23h, 45h, 67h</td></tr>
<tr><td>ESKY</td><td>(32 bytes)</td><td>Wrapped install key (W-UFPK).
For example: 01234567_89AB ... 2233_44556677h -> 01h, 23h, 45h, ... , 55h, 66h, 77h</td></tr>
<tr><td>IVEC</td><td>(16 bytes)</td><td>Initialization Vector.
For example: 01234567_89AB ... 2233_44556677h -> 01h, 23h, 45h, ... , 55h, 66h, 77h</td></tr>
<tr><td>EOKY</td><td>(32 bytes)</td><td>Install data (Encrypted key | MAC).
Encrypted key (0–15 bytes) + MAC (16-31 bytes)
For example: If the install data is as follows, the host should send EOKY in the order shown in the lower table.
Install data:
<table>
<tr><td colspan="8">Encrypted key</td></tr>
<tr><td>00</td><td>01</td><td>02</td><td>03</td><td>04</td><td>05</td><td>06</td><td>07</td></tr>
<tr><td>08</td><td>09</td><td>0A</td><td>0B</td><td>0C</td><td>0D</td><td>0E</td><td>0F</td></tr>
<tr><td colspan="8">MAC</td></tr>
<tr><td>10</td><td>11</td><td>12</td><td>13</td><td>14</td><td>15</td><td>16</td><td>17</td></tr>
<tr><td>18</td><td>19</td><td>1A</td><td>1B</td><td>1C</td><td>1D</td><td>1E</td><td>1F</td></tr>
</table>
Order of sending EOKY:
<table>
<tr><td>1st</td><td>2nd</td><td>3rd</td><td>4th</td><td>5th</td><td>6th</td><td>7th</td><td>8th</td></tr>
<tr><td>00</td><td>01</td><td>02</td><td>03</td><td>04</td><td>05</td><td>06</td><td>07</td></tr>
<tr><td>9th</td><td>10th</td><td>11th</td><td>12th</td><td>13th</td><td>14th</td><td>15th</td><td>16th</td></tr>
<tr><td>08</td><td>09</td><td>0A</td><td>0B</td><td>0C</td><td>0D</td><td>0E</td><td>0F</td></tr>
<tr><td>17th</td><td>18th</td><td>19th</td><td>20th</td><td>21st</td><td>22nd</td><td>23rd</td><td>24th</td></tr>
<tr><td>10</td><td>11</td><td>12</td><td>13</td><td>14</td><td>15</td><td>16</td><td>17</td></tr>
<tr><td>25th</td><td>26th</td><td>27th</td><td>28th</td><td>29th</td><td>30th</td><td>31st</td><td>32nd</td></tr>
<tr><td>18</td><td>19</td><td>1A</td><td>1B</td><td>1C</td><td>1D</td><td>1E</td><td>1F</td></tr>
</table></td></tr>
<tr><td>SUM</td><td>(1 byte)</td><td>Sum data</td></tr>
<tr><td>ETX</td><td>(1 byte)</td><td>03h</td></tr>
</table>

6.8.2.3 Data Packet [Status OK]

SOD	(1 byte)	81h
LNH	(1 byte)	00h
LNL	(1 byte)	0Ah
RES	(1 byte)	28h (OK)
STS	(1 byte)	00h (OK)
ST2	(4 bytes)	FFFFFFFFh (unused code)
ADR	(4 bytes)	FFFFFFFFh (unused code)
SUM	(1 byte)	D6h
ETX	(1 byte)	03h

6.8.2.4 Data Packet [Status ERR]

SOD	(1 byte)	81h
LNH	(1 byte)	00h
LNL	(1 byte)	0Ah
RES	(1 byte)	A8h (ERR)
STS	(1 byte)	Status code
ST2	(4 bytes)	Status details
ADR	(4 bytes)	Failure address
SUM	(1 byte)	Sum data
ETX	(1 byte)	03h

6.8.3 Processing Procedure

Boot firmware receives and analyzes a command packet:

- The boot firmware recognizes the start of the command packet by receiving SOH.
  If the boot firmware receives something other than SOH, it waits until it receives SOH.
- If ETX is not added to the received command packet, the boot firmware sends a “Packet error”.
- If the SUM of the received command packet is different from the sum value, the boot firmware sends a “Checksum error”.
- If the LNH and LNL of the received command packet are different from the values specified in the packet format, the boot firmware sends a “Packet error”.
- If the LNH and LNL of the received command packet are different from the values specified in each command, the boot firmware sends a “Packet error”.
- When any of the above errors occur, the boot firmware does not process and returns to the command waiting state.
  * Memory contents do not change before command reception.

When the processing above is successfully completed, boot firmware executes the acceptance analysis:

- If this command cannot be executed in the current DLM state, the boot firmware sends a “Command acceptance error”.
- If HUK has been zeroized, the boot firmware sends a “Command acceptance error”.
- If the device reset is not asserted after the Encrypted data write command execution, the boot firmware sends a “Command acceptance error”.
- When any of the above errors occur, the boot firmware does not process and returns to the command waiting state.
  * Memory contents do not change before command reception.

When the processing above is successfully completed, boot firmware executes the parameter analysis:

- When KYTY is an unspecified value, boot firmware returns “Parameter error” and waits for the next command.
  * Memory contents do not change before command reception.
- When KYTY cannot be set in the current Authentication level, boot firmware returns “Secure error” and waits for the next command.
  * Memory contents do not change before command reception.
- If the above error does not occur, the boot firmware sends “OK”.

When the processing above is successfully completed, the boot firmware receives and analyzes the data packet:

- Boot firmware detects the beginning of a data packet by receiving SOD.
  When boot firmware receives other data than SOD, it discards the data and waits for the following data until SOD is sent.
- When the received data packet does not have ETX, “Packet error” is returned.
- When the SUM in the received data packet is different from the value calculated by the boot firmware, “Checksum error” is returned.
- When LNH and LNL in the received data packet do not comply with the packet format, “Packet error” is returned.
- When RES in the received data packet is different from the defined values, “Packet error” is returned.
- When the number of received data exceeds the value specified in the command in the received data packet, “Parameter error” is returned.
- When any of the above errors occur, the boot firmware does not process and returns to the command waiting state.
  * Memory contents do not change before command reception.

When the processing above is successfully completed, boot firmware generates Key index (Wrapped key):

- If the Trusted system becomes abnormal after creating a key index (Wrapped key), the boot firmware returns nothing and does not respond.
  * Memory contents do not change before command reception.
- If the generation of the Key index (wrapped key) fails, the boot firmware sends a “Trusted system error” and returns to the command waiting state.
  * Memory contents do not change before command reception.

When the processing above is successfully completed, boot firmware writes Key index to memory:

- If an error occurs while writing, the boot firmware sends a “MRAM sequencer error” and returns to the command wait state.
- When the authentication key setting is successfully completed, the boot firmware returns “OK” and waits for the next command.

.

6.8.4 Status Information from the Microcontroller

(Listed in descending order of priority.)

Condition	STS	ST2	ADR
The received packet does not have ETX.	Packet error	FFFFFFFFh	FFFFFFFFh
Sum data in the packet received is different from the value calculated by the boot firmware.	Checksum error	FFFFFFFFh	FFFFFFFFh
Packet length in the received packet does not comply with the packet format.	Packet error	FFFFFFFFh	FFFFFFFFh
Packet length in the received packet does not comply with the specifications of this command.	Packet error	FFFFFFFFh	FFFFFFFFh
Executing this command is unavailable in the current DLM state.	Command acceptance error	FFFFFFFFh	FFFFFFFFh
HUK has been zeroized.	Command acceptance error	FFFFFFFFh	FFFFFFFFh
Device reset is not asserted after Encrypted data write command execution.	Command acceptance error	FFFFFFFFh	FFFFFFFFh
The specified Key type is an unspecified value.	Parameter error	FFFFFFFFh	FFFFFFFFh
The specified Key type cannot be inserted at the current Authentication level.	Secure error	FFFFFFFFh	FFFFFFFFh
The response code of the received data packet is different from the value specified by this command.	Packet error	FFFFFFFFh	FFFFFFFFh
The total length of received data packets exceed the value specified in the command.	Parameter error	FFFFFFFFh	FFFFFFFFh
Authentication key generation failed.	Trusted system error	FFFFFFFFh	FFFFFFFFh
MACI detected an error after the command execution in an undisclosed area.	MRAM sequencer error	MRAM status	FFFFFFFFh
Successful completion.	OK	FFFFFFFFh	FFFFFFFFh

6.8.5 Key type that can be set in each Authentication Level

Table 19 shows the Key types that can be set in each Authentication level.

Table 19. Key Types for each Authentication Level

Authentication level	Key type
AL2	AL2_KEY AL1_KEY RMA_KEY
AL1	AL1_KEY

6.9 User Key Setting Command

This command generates Key index (Wrapped key) using Wrapped install key (W-UFPK) and Install data (Encrypted key | MAC) received from the host and saves it in the specified area. Write processing at this time is not affected by the block protection settings (BPS, BPS_SEC).

This command requires adherence to conditions described in Command List.

6.9.1 Sequence Diagram

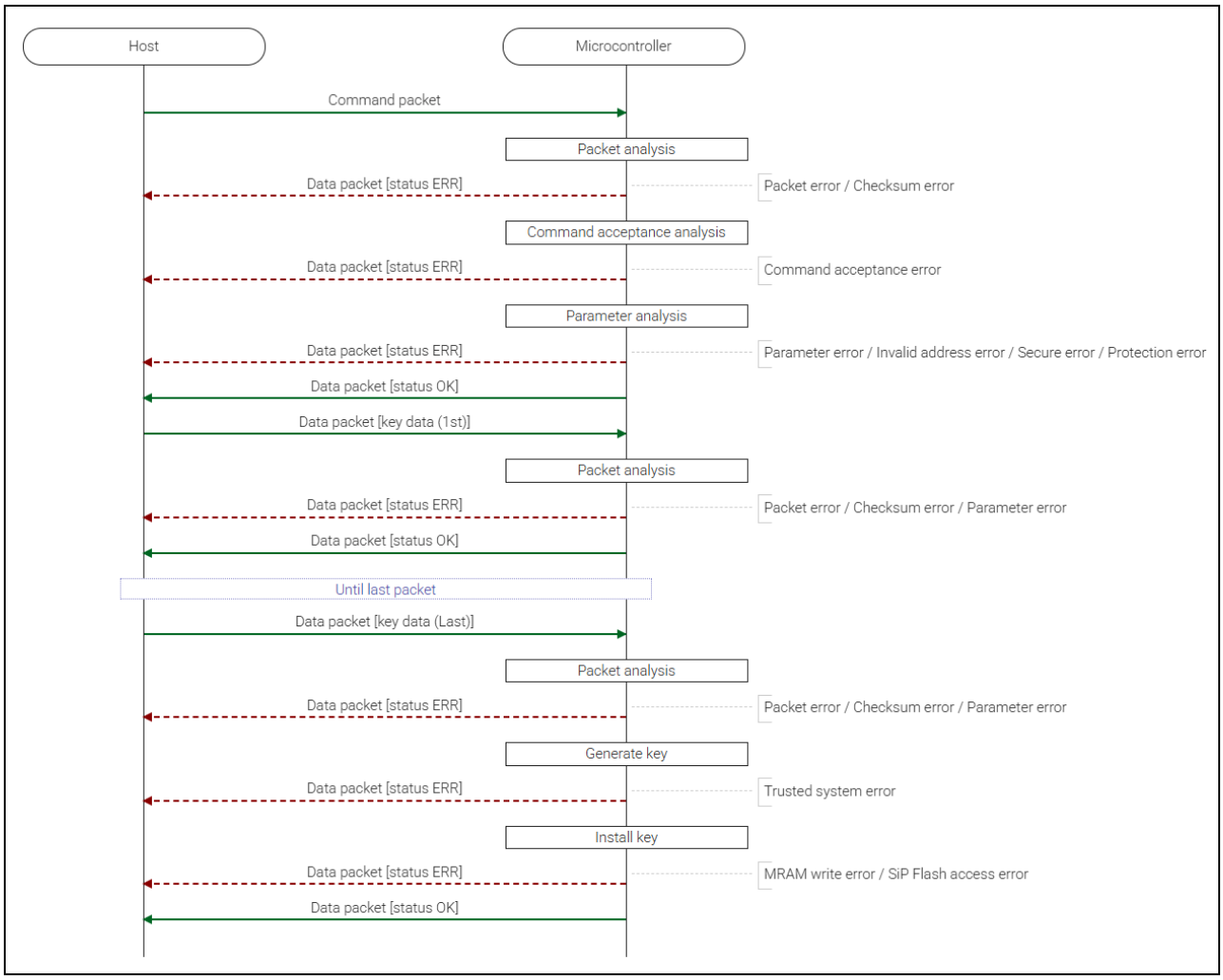


Figure 26. User Key Setting Command Sequence Diagram

6.9.2 Packets

6.9.2.1 Command Packet

SOH	(1 byte)	01h
LNH	(1 byte)	00h
LNL	(1 byte)	06h
CMD	(1 byte)	2Ah (User key setting command)
KADR	(4 bytes)	Key setting address. For example: 00004000h -> 00h, 00h, 40h, 00h
ENTY	(1 byte)	User key type. Refer to User key list.
SUM	(1 byte)	Sum data
ETX	(1 byte)	03h

6.9.2.2 Data Packet [Key Data (1st)]

<table>
<tr><td>SOD</td><td>(1 byte)</td><td>81h</td></tr>
<tr><td>LNH</td><td>(1 byte)</td><td>N + 53 (Higher 1 byte)</td></tr>
<tr><td>LNL</td><td>(1 byte)</td><td>N + 53 (Lower 1 byte)</td></tr>
<tr><td>RES</td><td>(1 byte)</td><td>2Ah (OK)</td></tr>
<tr><td>SKR</td><td>(4 bytes)</td><td>Shared key ring number.
For example: 01234567h -> 01h, 23h, 45h, 67h</td></tr>
<tr><td>ESKY</td><td>(32 bytes)</td><td>Wrapped install key (W-UFPK).
For example: 01234567_89AB ... 2233_44556677h -> 01h, 23h, 45h, ... , 55h, 66h, 77h</td></tr>
<tr><td>IVEC</td><td>(16 bytes)</td><td>Initialization vector.
For example: 01234567_89AB ... 2233_44556677h -> 01h, 23h, 45h, ... , 55h, 66h, 77h</td></tr>
<tr><td>ENKY</td><td>(N bytes)</td><td>Install data (Encrypted key | MAC) .
For example: If the key type is ECC P-192 Private Key, the host should send ENKY in the order shown in the lower table.
Install data:
<table>
<tr><td colspan="8">Encrypted Key</td></tr>
<tr><td>00</td><td>01</td><td>02</td><td>03</td><td>04</td><td>05</td><td>06</td><td>07</td></tr>
<tr><td>08</td><td>09</td><td>0A</td><td>0B</td><td>0C</td><td>0D</td><td>0E</td><td>0F</td></tr>
<tr><td>10</td><td>11</td><td>12</td><td>13</td><td>14</td><td>15</td><td>16</td><td>17</td></tr>
<tr><td>18</td><td>19</td><td>1A</td><td>1B</td><td>1C</td><td>1D</td><td>1E</td><td>1F</td></tr>
<tr><td colspan="8">MAC</td></tr>
<tr><td>20</td><td>21</td><td>22</td><td>23</td><td>24</td><td>25</td><td>26</td><td>27</td></tr>
<tr><td>28</td><td>29</td><td>2A</td><td>2B</td><td>2C</td><td>2D</td><td>2E</td><td>2F</td></tr>
</table>
Order of sending ENKY:
<table>
<tr><td>1st</td><td>2nd</td><td>3rd</td><td>4th</td><td>5th</td><td>6th</td><td>7th</td><td>8th</td></tr>
<tr><td>00</td><td>01</td><td>02</td><td>03</td><td>04</td><td>05</td><td>06</td><td>07</td></tr>
<tr><td>9th</td><td>10th</td><td>11th</td><td>12th</td><td>13th</td><td>14th</td><td>15th</td><td>16th</td></tr>
<tr><td>08</td><td>09</td><td>0A</td><td>0B</td><td>0C</td><td>0D</td><td>0E</td><td>0F</td></tr>
<tr><td>17th</td><td>18th</td><td>19th</td><td>20th</td><td>21st</td><td>22nd</td><td>23rd</td><td>24th</td></tr>
<tr><td>10</td><td>11</td><td>12</td><td>13</td><td>14</td><td>15</td><td>16</td><td>17</td></tr>
<tr><td>25th</td><td>26th</td><td>27th</td><td>28th</td><td>29th</td><td>30th</td><td>31st</td><td>32nd</td></tr>
<tr><td>18</td><td>19</td><td>1A</td><td>1B</td><td>1C</td><td>1D</td><td>1E</td><td>1F</td></tr>
<tr><td>33rd</td><td>34th</td><td>35th</td><td>36th</td><td>37th</td><td>38th</td><td>39th</td><td>40th</td></tr>
<tr><td>20</td><td>21</td><td>22</td><td>23</td><td>24</td><td>25</td><td>26</td><td>27</td></tr>
<tr><td>41st</td><td>42nd</td><td>43rd</td><td>44th</td><td>45th</td><td>46th</td><td>47th</td><td>48th</td></tr>
<tr><td>28</td><td>29</td><td>2A</td><td>2B</td><td>2C</td><td>2D</td><td>2E</td><td>2F</td></tr>
</table></td></tr>
<tr><td>SUM</td><td>(1 byte)</td><td>Sum data</td></tr>
<tr><td>ETX</td><td>(1 byte)</td><td>03h</td></tr>
</table>

N = 1 ~ 972

6.9.2.3 Data Packet [Key Data (2nd to Last)]

SOD	(1 byte)	81h
LNH	(1 byte)	N + 1 (Higher 1 byte)
LNL	(1 byte)	N + 1 (Lower 1 byte)
RES	(1 byte)	2Ah (OK)
ENKY	(N bytes)	Install data (Encrypted key | MAC). *Order of sending: Low -> ... -> High
SUM	(1 byte)	Sum data
ETX	(1 byte)	03h

N = 1 ~ 1024

6.9.2.4 Data Packet [Status OK]

SOD	(1 byte)	81h
LNH	(1 byte)	00h
LNL	(1 byte)	0Ah
RES	(1 byte)	2Ah (OK)
STS	(1 byte)	00h (OK)
ST2	(4 bytes)	FFFFFFFFh (unused code)
ADR	(4 bytes)	FFFFFFFFh (unused code)
SUM	(1 byte)	D4h
ETX	(1 byte)	03h

6.9.2.5 Data Packet [Status ERR]

SOD	(1 byte)	81h
LNH	(1 byte)	00h
LNL	(1 byte)	0Ah
RES	(1 byte)	AAh (ERR)
STS	(1 byte)	Status code
ST2	(4 bytes)	Status details
ADR	(4 bytes)	Failure address
SUM	(1 byte)	Sum data
ETX	(1 byte)	03h

6.9.3 Processing Procedure

Boot firmware receives and analyzes a command packet:

- The boot firmware recognizes the start of the command packet by receiving SOH.
  If the boot firmware receives something other than SOH, it waits until it receives SOH.
- If ETX is not added to the received command packet, the boot firmware sends a “Packet error”.
- If the SUM of the received command packet is different from the sum value, the boot firmware sends a “Checksum error”.
- If the LNH and LNL of the received command packet are different from the values specified in the packet format, the boot firmware sends a “Packet error”.
- If the LNH and LNL of the received command packet are different from the values specified in each command, the boot firmware sends a “Packet error”.
- When any of the above errors occurs, the boot firmware does not process and returns to the command waiting state.
  * Memory contents do not change before command reception.

When the processing above is successfully completed, the boot firmware executes the acceptance analysis:

- If this command cannot be executed in the current DLM state, the boot firmware sends a “Command acceptance error”.
- If HUK has been zeroized, the boot firmware sends a “Command acceptance error”.
- When the above error occurs, the boot firmware does not process and returns to the command waiting state.
  * Memory contents do not change before command reception.

When the processing above is successfully completed, boot firmware analyzes the command parameters:

- If ENTY is not specified as Key type, the boot firmware sends a “Parameter error”.
- If the area for Key index size from KADR is not included in the User area or Data area specified in the area information, the boot firmware sends a “Parameter error”.
- If the area from KADR to Key index size is across different KOAs, the boot firmware sends a “Parameter error”.
- If the WAU for the specified area is 0, the boot firmware sends a “Parameter error”.
- If KADR is not specified in the WAU of the area, the boot firmware sends a “Parameter error”.
- If the specified range contains addresses that are inaccessible with the current boundary settings, the boot firmware sends an “Invalid address error”.
- If the current Authentication level is AL1 and the specified range includes a secure area, the boot firmware sends a “Secure error”.
- If the current Authentication level is AL0, the boot firmware sends a “Secure error”.
- If the area for the key index size from KADR contains a permanent protected block, the boot firmware sends a “Protection error”.
- When any of the above errors occurs, the boot firmware does not process and returns to the command waiting state.
  * Memory contents do not change before command reception.
- If the above errors do not occur, the boot firmware sends “OK”.

When the processing above is successfully completed, boot firmware receives and analyzes data packet:

- Boot firmware detects the beginning of a data packet by receiving SOD. When boot firmware receives other data than SOD, it discards the data and waits for the next data until SOD is sent.
- When the received data packet does not have ETX, “Packet error” is returned.
- When SUM in the received data packet is different from the value calculated by boot firmware, “Checksum error” is returned.
- When LNH and LNL in the received data packet do not comply with the packet format, “Packet error” is returned.
- When RES in the received data packet is different from defined values, “Packet error” is returned.
- When the number of accumulated ENKY data exceeds the Install data size indicated by ENTY in the received data packet, the boot firmware sends a “Parameter error”.
- When any of the above errors occurs, the boot firmware does not process and returns to the command waiting state.
  * Memory contents do not change before command reception.
- If the key data has not been received, the boot firmware receives the next data packet.

When all key data has been received, the boot firmware generates a key index (Wrapped key):

- If the Trusted system becomes abnormal after creating a key index (Wrapped key), the boot firmware returns nothing and does not respond.
  * Memory contents do not change before command reception.
- If the generation of Key index (Wrapped key) fails, the boot firmware sends a “Trusted system error” and returns to the command waiting state.
  * Memory contents do not change before command reception.

When the processing above is successfully completed, boot firmware writes Key index to the dedicated area:

- If an error occurs while writing, the boot firmware sends a “MRAM write error” or “SiP Flash access error” and returns to the command wait state.
- If the key index (wrapped key) is successfully saved to the device, the boot firmware sends “OK” and returns to the command wait state.

6.9.4 Status Information from the Microcontroller

(Listed in descending order of priority.)

Condition	STS	ST2	ADR
The received packet does not have ETX.	Packet error	FFFFFFFFh	FFFFFFFFh
Sum data in the packet received is different from the value calculated by the boot firmware.	Checksum error	FFFFFFFFh	FFFFFFFFh
Packet length in the received packet does not comply with the packet format.	Packet error	FFFFFFFFh	FFFFFFFFh
Packet length in the received packet does not comply with the specifications of this command.	Packet error	FFFFFFFFh	FFFFFFFFh
Executing this command is unavailable in the current DLM state.	Command acceptance error	FFFFFFFFh	FFFFFFFFh
HUK has been zeroized.	Command acceptance error	FFFFFFFFh	FFFFFFFFh
User key type is not specified as Key type.	Parameter error	FFFFFFFFh	FFFFFFFFh
The area from Key setting address to key index size does not fit in the range of User area and Data area specified by area information.	Parameter error	FFFFFFFFh	FFFFFFFFh
The area from Key setting address to Key index size spans different types of area.	Parameter error	FFFFFFFFh	FFFFFFFFh
The key storage area WAU is 0.	Parameter error	FFFFFFFFh	FFFFFFFFh
Key setting address is not specified in the WAU for the area.	Parameter error	FFFFFFFFh	FFFFFFFFh
The area from the Key setting address to the Key index size contains addresses that are inaccessible with the current boundary settings.	Invalid address error	FFFFFFFFh	FFFFFFFFh
The current Authentication level is AL1, and the Key setting address contains a Secure region.	Secure error	FFFFFFFFh	FFFFFFFFh
The current Authentication level is AL0.	Secure error	FFFFFFFFh	FFFFFFFFh
There is a block with permanent block protection in the area from the Key setting address to the Key index size.	Protection error	FFFFFFFFh	FFFFFFFFh
The response code of the received data packet is different from the value specified by this command.	Packet error	FFFFFFFFh	FFFFFFFFh
In the received data packet, the cumulative number of Install data exceeds the Install data size of the key specified by User key type.	Parameter error	FFFFFFFFh	FFFFFFFFh
Key index (wrapped key) generation failed.	Trusted system error	FFFFFFFFh	FFFFFFFFh
An error detected when MRAM programming which does not use MACI command.	MRAM write error	FFFFFFFFh	Failure address

An error occurred while programming SiP Flash.	SiP Flash access error	SiP Flash status	Failure address
Successful completion.	OK	FFFFFFFFh	FFFFFFFFh

6.9.4.1 User Key List

The list of user keys specified by this command is shown in Table 20.

Table 20. User Key List

Key type	Installation key	Install data size (bytes)	Key index size (bytes)
05h	AES-128	32	36
06h	AES-192	48	52
07h	AES-256	48	52
08h	AES-128 XTS	48	52
09h	AES-256 XTS	80	84
0Ah	RSA-1024 Public key	160	164
0Bh	RSA-1024 Private key	272	276
0Ch	RSA-2048 Public key	288	292
0Dh	RSA-2048 Private key	528	532
0Eh	RSA-3072 Public key	416	420
0Fh	RSA-3072 Private key	784	788
10h	RSA-4096 Public key	544	548
11h	RSA-4096 Private key	1040	1044
12h	ECC P192 Public key	80	84
13h	ECC P192 Private key	48	52
14h	ECC P224 Public key	80	84
15h	ECC P224 Private key	48	52
16h	ECC P256 Public key	80	84
17h	ECC P256 Private key	48	52
18h	ECC P384 Public key	112	116
19h	ECC P384 Private key	64	68
1Ah	HMAC-SHA224	48	52
1Bh	HMAC-SHA256	48	52
1Ch	ECC P256r1 Public Key	80	84
1Dh	ECC P256r1 Private Key	48	52
1Eh	ECC P384r1 Public Key	112	116
1Fh	ECC P384r1 Private Key	64	68
20h	ECC P512r1 Public Key	144	148
21h	ECC P512r1 Private Key	80	84
22h	ECC secp256k1 Public Key	80	84
23h	ECC secp256k1 Private Key	48	52
24h	ECC P521 Public Key	176	180
25h	ECC P521 Private Key	96	100
26h	Ed25519 Public Key	48	52
27h	Ed25519 Private Key	48	52
28h	HMAC-SHA384	64	68
29h	HMAC-SHA512	80	84
2Ah	HMAC-SHA512-224	80	84
2Bh	HMAC-SHA512-256	80	84
2Ch	HMAC-SHA3-224	48	52
2Dh	HMAC-SHA3-256	48	52
2Eh	HMAC-SHA3-384	64	68
2Fh	HMAC-SHA3-512	80	84
30h	ChaCha20-Poly-1305	48	52
FEh	RSA-2048 Public Key for TLS	288	292
FFh	Key update key	48	52

6.10 Key Verify Command

This command verifies the authentication key that setting to device.

This command requires adherence to conditions described in Command List.

6.10.1 Sequence Diagram

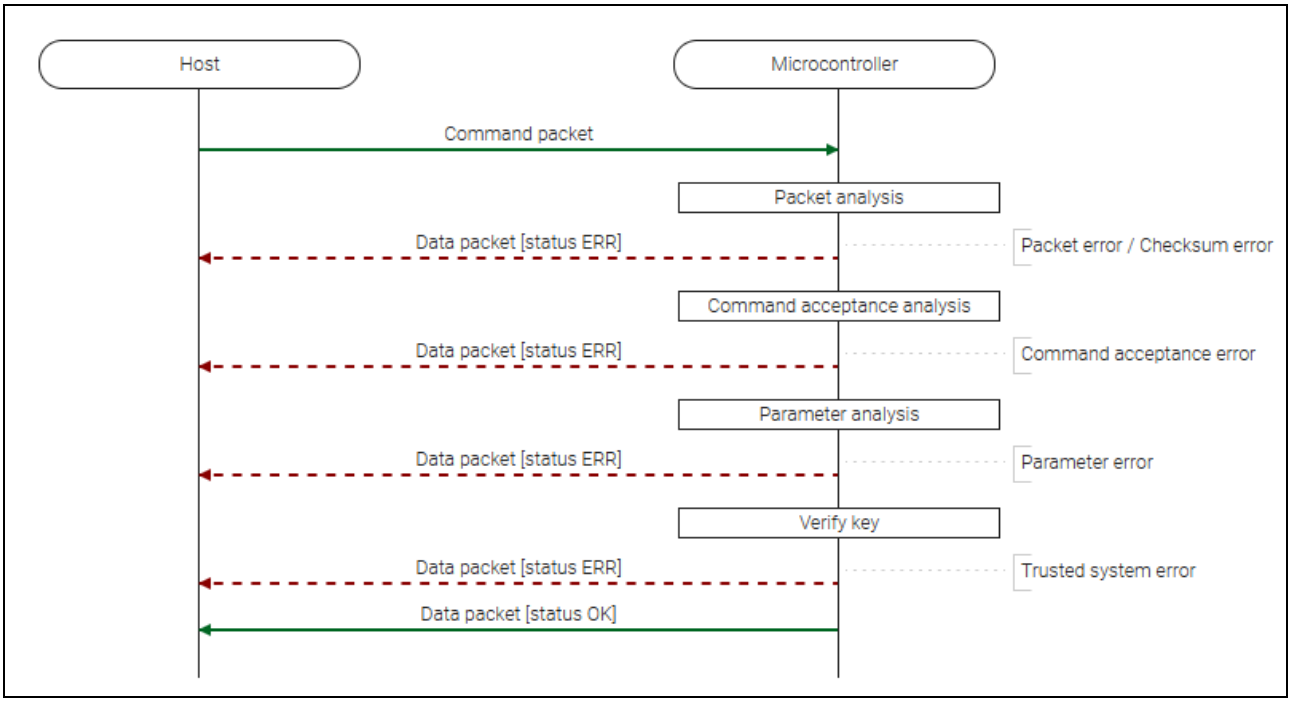


Figure 27. Key Verify Command Sequence Diagram

6.10.2 Packets

6.10.2.1 Command Packet

SOH	(1 byte)	01h
LNH	(1 byte)	00h
LNL	(1 byte)	02h
CMD	(1 byte)	29h (Key verify command)
KYTY	(1 byte)	Key type: • 01h: AL2_KEY • 02h: AL1_KEY • 03h: RMA_KEY
SUM	(1 byte)	Sum data
ETX	(1 byte)	03h

6.10.2.2 Data Packet [Status OK]

SOD	(1 byte)	81h
LNH	(1 byte)	00h
LNL	(1 byte)	0Ah
RES	(1 byte)	29h (OK)
STS	(1 byte)	00h (OK)
ST2	(4 bytes)	FFFFFFFFh (unused code)
ADR	(4 bytes)	FFFFFFFFh (unused code)
SUM	(1 byte)	D5h
ETX	(1 byte)	03h

6.10.2.3 Data Packet [Status ERR]

SOD	(1 byte)	81h
LNH	(1 byte)	00h
LNL	(1 byte)	0Ah
RES	(1 byte)	A9h (ERR)
STS	(1 byte)	Status code
ST2	(4 bytes)	Status details
ADR	(4 bytes)	Failure address
SUM	(1 byte)	Sum data
ETX	(1 byte)	03h

6.10.2.4 Processing Procedure

Boot firmware receives and analyzes a command packet:

- The boot firmware recognizes the start of the command packet by receiving SOH.
  If the boot firmware receives something other than SOH, it waits until it receives SOH.
- If ETX is not added to the received command packet, the boot firmware sends a “Packet error”.
- If the SUM of the received command packet is different from the sum value, the boot firmware sends a “Checksum error”.
- If the LNH and LNL of the received command packet are different from the values specified in the packet format, the boot firmware sends a “Packet error”.
- If the LNH and LNL of the received command packet are different from the values specified in each command, the boot firmware sends a “Packet error”.
- When any of the above errors occurs, the boot firmware does not process and returns to the command waiting state.
  * Memory contents do not change before command reception.

When the processing above is successfully completed, boot firmware executes the acceptance analysis:

- If HUK has been zeroized, the boot firmware sends a “Command acceptance error”.
- If the device reset is not asserted after the Encrypted data write command execution, the boot firmware sends a “Command acceptance error”.
- When the above error occurs, the boot firmware does not process and returns to the command waiting state.
  * Memory contents do not change before command reception.

When the processing above is successfully completed, boot firmware executes the parameter analysis:

- If KYTY is an unsupported key type, the boot firmware sends a “Parameter error” and returns to the command wait state.
  * Memory contents do not change before command reception.

When the processing above is successfully completed, boot firmware verifies Key index (Wrapped key).

- If verification of key index (Wrapped key) fails, the boot firmware sends a “Trusted system error” and returns to the command wait state.
  If the Trusted system becomes abnormal during verification of key index (Wrapped key), the boot firmware returns nothing and does not respond.
  * Memory contents do not change before command reception.
- If the verification of the key index (Wrapped key) is completed successfully, the boot firmware sends “OK” and returns to the command wait state.
  * Memory contents do not change before command reception.

6.10.3 Status Information from the Microcontroller

(Listed in descending order of priority.)

Condition	STS	ST2	ADR
The received packet does not have ETX.	Packet error	FFFFFFFFh	FFFFFFFFh
Sum data in the packet received is different from the value calculated by the boot firmware.	Checksum error	FFFFFFFFh	FFFFFFFFh
Packet length in the received packet does not comply with the packet format.	Packet error	FFFFFFFFh	FFFFFFFFh
Packet length in the received packet does not comply with the specifications of this command.	Packet error	FFFFFFFFh	FFFFFFFFh
HUK has been zeroized.	Command acceptance error	FFFFFFFFh	FFFFFFFFh
Device reset is not asserted after Encrypted data write command execution.	Command acceptance error	FFFFFFFFh	FFFFFFFFh
Key type is not supported key type.	Parameter error	FFFFFFFFh	FFFFFFFFh
Verify the authentication key failed.	Trusted system error	FFFFFFFFh	FFFFFFFFh
Successful completion.	OK	FFFFFFFFh	FFFFFFFFh

6.11 User Key Verify Command

This command verifies the authentication key that is set in the device.

This command requires adherence to conditions described in Command List.

6.11.1 Sequence Diagram

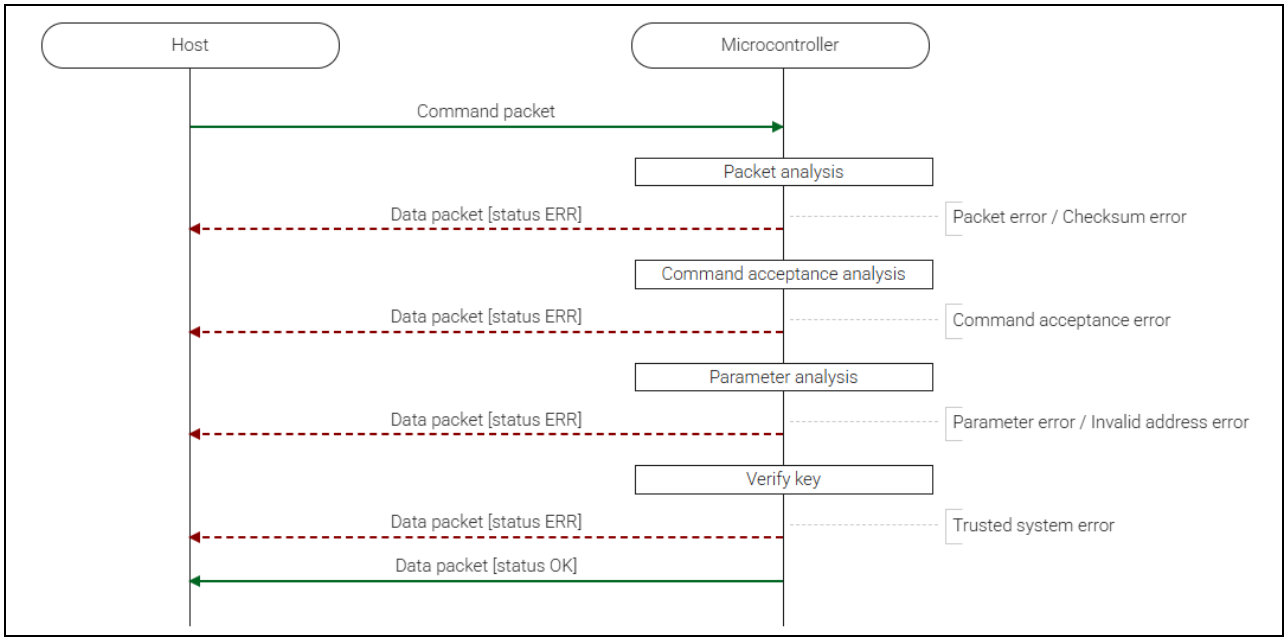


Figure 28. User Key Verify Command Sequence Diagram

6.11.2 Packets

6.11.2.1 Command Packet

SOH	(1 byte)	01h
LNH	(1 byte)	00h
LNL	(1 byte)	06h
CMD	(1 byte)	2Bh (User key verify command)
KADR	(4 bytes)	Key address. For example: 00004000h -> 00h, 00h, 40h, 00h
ENTY	(1 byte)	User key type. Supports the same key type as User key setting command.
SUM	(1 byte)	Sum data
ETX	(1 byte)	03h

6.11.2.2 Data Packet [Status OK]

SOD	(1 byte)	81h
LNH	(1 byte)	00h
LNL	(1 byte)	0Ah
RES	(1 byte)	2Bh (OK)
STS	(1 byte)	00h (OK)
ST2	(4 bytes)	FFFFFFFFh (unused code)
ADR	(4 bytes)	FFFFFFFFh (unused code)
SUM	(1 byte)	D3h
ETX	(1 byte)	03h

6.11.2.3 Data Packet [Status ERR]

SOD	(1 byte)	81h
LNH	(1 byte)	00h
LNL	(1 byte)	0Ah
RES	(1 byte)	ABh (ERR)
STS	(1 byte)	Status code
ST2	(4 bytes)	Status details
ADR	(4 bytes)	Failure address
SUM	(1 byte)	Sum data
ETX	(1 byte)	03h

6.11.3 Processing Procedure

Boot firmware receives and analyzes a command packet:

- The boot firmware recognizes the start of the command packet by receiving SOH.
  If the boot firmware receives something other than SOH, it waits until it receives SOH.
- If ETX is not added to the received command packet, the boot firmware sends a “Packet error”.
- If the SUM of the received command packet is different from the sum value, the boot firmware sends a “Checksum error”.
- If the LNH and LNL of the received command packet are different from the values specified in the packet format, the boot firmware sends a “Packet error”.
- If the LNH and LNL of the received command packet are different from the values specified in each command, the boot firmware sends a “Packet error”.
- When any of the above errors occurs, the boot firmware does not process and returns to the command waiting state.
  * Memory contents do not change before command reception

When the processing above is successfully completed, boot firmware executes the acceptance analysis.

- If HUK has been zeroized, the boot firmware sends a “Command acceptance error”.
- When the above error occurs, the boot firmware does not process and returns to the command waiting state.
  *Memory contents do not change before command reception.

When the processing above is successfully completed, boot firmware executes the parameter analysis:

- If ENTY is not specified as Key type, the boot firmware sends a “Parameter error”.
- If the area for Key index size from KADR is not included in the User area or SiP Flash area specified in the area information, the boot firmware sends a “Parameter error”.
- If the area from KADR to Key index size is across different KOAs, the boot firmware sends a “Parameter error”.
- If the WAU for the specified area is 0, the boot firmware sends a “Parameter error”.
- If KADR is not specified in the WAU of the area, the boot firmware sends a “Parameter error”.
- If the specified range contains addresses that are inaccessible with the current boundary settings, the boot firmware sends an “Invalid address error”.
- When any of the above errors occur, the boot firmware does not process and returns to the command waiting state.
  * Memory contents do not change before command reception.

When the processing above is successfully completed, boot firmware verifies the authentication key:

- When there is a mismatch in the authentication key stored in the device, boot firmware returns “Trusted system error”.
  If the Trusted system becomes abnormal during key verification, the boot firmware returns nothing and does not respond.
  * Memory contents do not change before command reception.
- If the above error does not occur, the boot firmware sends “OK”.
  * Memory contents do not change before command reception.

6.11.4 Status Information from the Microcontroller

(Listed in descending order of priority.)

Condition	STS	ST2	ADR
The received packet does not have ETX.	Packet error	FFFFFFFFh	FFFFFFFFh
Sum data in the packet received is different from the value calculated by the boot firmware.	Checksum error	FFFFFFFFh	FFFFFFFFh
Packet length in the received packet does not comply with the packet format.	Packet error	FFFFFFFFh	FFFFFFFFh
Packet length in the received packet does not comply with the specifications of this command.	Packet error	FFFFFFFFh	FFFFFFFFh
HUK has been zeroized.	Command acceptance error	FFFFFFFFh	FFFFFFFFh
User key type is not specified as the Key type.	Parameter error	FFFFFFFFh	FFFFFFFFh
The area from the Key address to the Key index size does not fit in the range of User area and SiP Flash area specified by area information.	Parameter error	FFFFFFFFh	FFFFFFFFh
The area from the Key address to the Key index size spans different types of area.	Parameter error	FFFFFFFFh	FFFFFFFFh
The key storage area WAU is 0.	Parameter error	FFFFFFFFh	FFFFFFFFh
Key address is not specified in the WAU for the area.	Parameter error	FFFFFFFFh	FFFFFFFFh
The area from the Key address to the Key index size contains addresses that are inaccessible with the current boundary settings.	Invalid address error	FFFFFFFFh	FFFFFFFFh
Key index verify failed.	Trusted system error	FFFFFFFFh	FFFFFFFFh
Successful completion.	OK	FFFFFFFFh	FFFFFFFFh

6.12 Initialize Command

This command initializes the following areas and transits the Protection level state to PL2:

- User area
- Config area
- SiP Flash area (Status register and Main Flash)
- Boundary setting
- Key index (Wrapped key)

Initialization used here means that erasure for erasable areas and writing initial values for non-erasable areas. Initialization processing at this time is not affected by the block protection settings (BPS, BPS_SEC).

Only Main Flash area and Status register are erased for SiP Flash. Data other than these are not erased. Even for Main Flash, ranges that meet any of the followings are not erased:

- Range where protection by BP is set, and also permanent lock of status register is set
- Range where protection by other than BP is set

This command requires adherence to conditions described in Command List.

6.12.1 Sequence Diagram

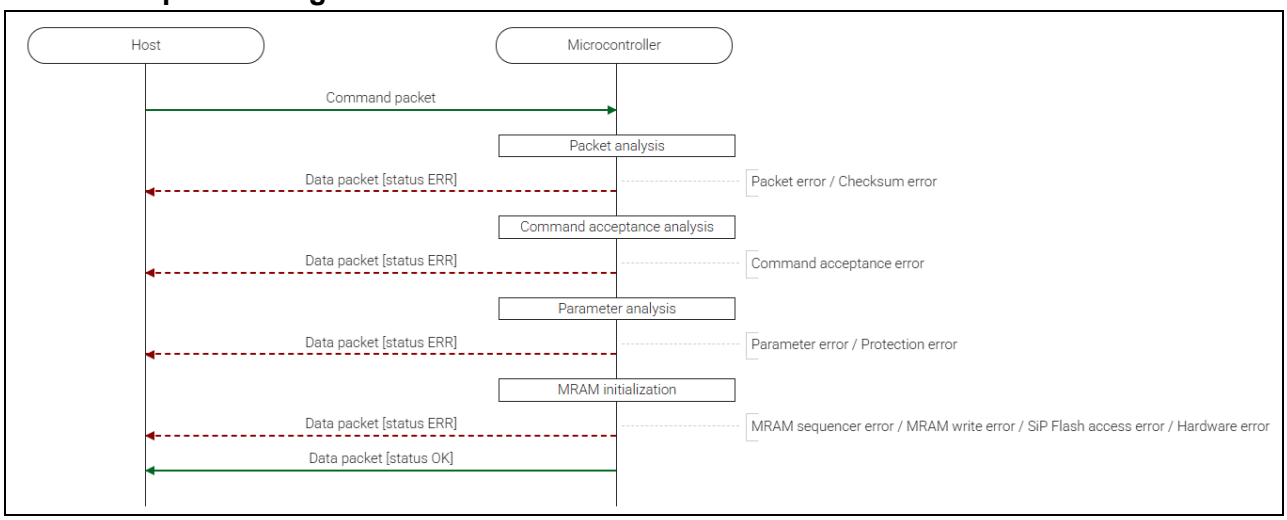


Figure 29. Initialize Command Sequence Diagram

6.12.2 Packets

6.12.2.1 Command Packet

SOH	(1 byte)	01h
LNH	(1 byte)	00h
LNL	(1 byte)	03h
CMD	(1 byte)	50h (Initialize command)
SDLM	(1 byte)	Source DLM state code: • 04h: OEM
DDLM	(1 byte)	Destination DLM state code: • 04h: OEM
SUM	(1 byte)	Sum data
ETX	(1 byte)	03h

6.12.2.2 Data Packet [Status OK]

SOD	(1 byte)	81h
LNH	(1 byte)	00h
LNL	(1 byte)	0Ah
RES	(1 byte)	50h (OK)
STS	(1 byte)	00h (OK)
ST2	(4 bytes)	FFFFFFFFh (unused code)
ADR	(4 bytes)	FFFFFFFFh (unused code)
SUM	(1 byte)	AEh
ETX	(1 byte)	03h

6.12.2.3 Data Packet [Status ERR]

SOD	(1 byte)	81h
LNH	(1 byte)	00h
LNL	(1 byte)	0Ah
RES	(1 byte)	D0h (ERR)
STS	(1 byte)	Status code
ST2	(4 bytes)	Status details
ADR	(4 bytes)	Failure address
SUM	(1 byte)	Sum data
ETX	(1 byte)	03h

6.12.3 Processing Procedure

Boot firmware receives and analyzes a command packet:

- The boot firmware recognizes the start of the command packet by receiving SOH.
  If the boot firmware receives something other than SOH, it waits until it receives SOH.
- If ETX is not added to the received command packet, the boot firmware sends a “Packet error”.
- If the SUM of the received command packet is different from the sum value, the boot firmware sends a “Checksum error”.
- If the LNH and LNL of the received command packet are different from the values specified in the packet format, the boot firmware sends a “Packet error”.
- If the LNH and LNL of the received command packet are different from the values specified in each command, the boot firmware sends a “Packet error”.
- When any of the above errors occurs, the boot firmware does not process and returns to the command waiting state.
  * Memory contents do not change before command reception.

When the processing above is successfully completed, boot firmware executes the acceptance analysis:

- If this command cannot be executed in the current DLM state, the boot firmware sends a “Command acceptance error”.
- If device reset is not asserted after the Encrypted data write command execution, the boot firmware sends a “Command acceptance error”.
- When any of the above errors occurs, the boot firmware does not process and returns to the command waiting state.
  * Memory contents do not change before command reception.

When the processing above is successfully completed, the boot firmware analyzes the command parameters:

- When SDLM does not match with the current DLM state, “Parameter error” is returned.
- When DDLM is not OEM, “Parameter error” is returned.
- When initialization is disabled, “Protection error” is returned.
- When authentication with AL2_KEY is disabled, “Protection error” is returned.
- When UMOTP area is not All“1”, “Protection error” is returned.
- When the above error occurs, the boot firmware does not process and returns to the command waiting state.
  * Memory contents do not change before command reception

When the processing above is successfully completed, boot firmware executes memory initialization:

- If an error occurs during initialization, the boot firmware sends a “MRAM sequencer error” or “MRAM write error”, “SiP Flash access error” and returns to the command wait state.
- If the Protection level is an invalid value, the boot firmware sends a “Hardware error” and becomes unresponsive.
- If initialization is completed normally, the boot firmware sends “OK” and does not respond.
  * The memory is in the following state and the Protection level is PL2:
    - User area : Intialized
    - SiP Flash area : Intialized, only Status register and Main Flash area
    - Config area : Value at shipment
    - Key index : Intialized
    - Boudary setting : Intialized

6.12.4 Status Information from the Microcontroller

(Listed in descending order of priority.)

Condition	STS	ST2	ADR
The received packet does not have ETX.	Packet error	FFFFFFFFh	FFFFFFFFh
Sum data in the packet received is different from the value calculated by the boot firmware.	Checksum error	FFFFFFFFh	FFFFFFFFh
Packet length in the received packet does not comply with the packet format.	Packet error	FFFFFFFFh	FFFFFFFFh
Packet length in the received packet does not comply with the specifications of this command.	Packet error	FFFFFFFFh	FFFFFFFFh
Executing this command is unavailable in the current DLM state.	Command acceptance error	FFFFFFFFh	FFFFFFFFh
Device reset is not asserted after Encrypted data write command execution.	Command acceptance error	FFFFFFFFh	FFFFFFFFh
Source DLM state code is different from the current DLM state.	Parameter error	FFFFFFFFh	FFFFFFFFh
Destination DLM state code is not OEM.	Parameter error	FFFFFFFFh	FFFFFFFFh
Initialization is disabled.	Protection error	FFFFFFFFh	FFFFFFFFh
AL2_KEY is disable.	Protection error	FFFFFFFFh	FFFFFFFFh
UMOTP is not All“1”.	Protection error		
An error detected when MRAM programming which does not use MACI command.	MRAM write error	FFFFFFFFh	Failure address
MACI detected an error after the command execution in disclosed area.	MRAM sequencer error	MRAM status	Failure address
MACI detected an error after the command execution in an undisclosed area.	MRAM sequencer error	MRAM status	FFFFFFFFh
An error occurred while erasing SiP Flash.	SIP Flash access error	SIP Flash status	Failure address
Protection level is abnormal.	Hardware error	FFFFFFFFh	FFFFFFFFh
Successful completion.	OK	FFFFFFFFh	FFFFFFFFh

6.12.5 Precautions

(1) The following parameters are not initialized by this command. For details on each parameter, refer to Parameter setting command.
- Disable of authentication using AL1_KEY
- Disable transition to LCK_BOOT
- Master CPU selection
- Disable security extension of CPU1

(2) The following areas are not initialized by this command.
- Anti-rollback counter area
- External flash area

(3) Only Status register and Main Flash area are erased for SiP Flash. However, areas where protection cannot be released are not erased.
- As a result, after executing this command, the following data remains not erased and the protection level of the device is changed back to PL2.
- Value of Status register, when permanent lock of Status register is set
- Data in Main Flash where protection by BP is set, when permanent lock of Status register is set and also protection by BP is set
- Data in Main Flash where protection by other than BP is set
- Data in other than Status register and Main Flash
- Data leakage etc by this can be prevented by disabling initialization using Parameter setting command.

6.12.6 Protection Level Transition

The transition of Protection level by this command is shown below.

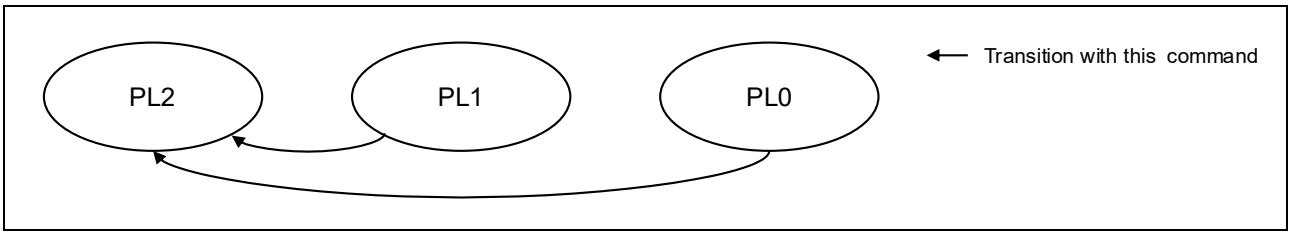


Figure 30. Protection Level Transitions

6.13 Boundary Setting Command

This command receives the boundary setting and stores it in the device.

The accessible addresses of the following areas change depending on the boundary settings:

- User area
- SIP Flash area

This command requires adherence to conditions described in Command List.

6.13.1 Sequence Diagram

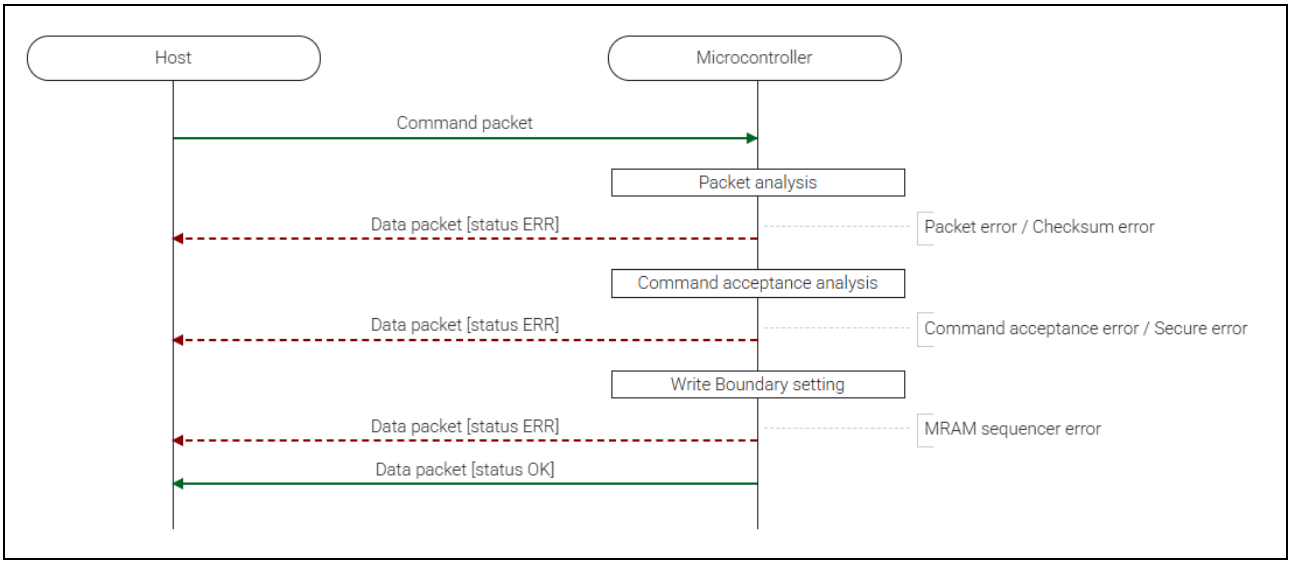


Figure 31. Boundary Setting Command Sequence Diagram

6.13.2 Packets

6.13.2.1 Command Packet

SOH	(1 byte)	01h
LNH	(1 byte)	00h
LNL	(1 byte)	0Bh
CMD	(1 byte)	4Eh (Boundary setting command)
RSV	(2 bytes)	0000h (unused code)
CMS	(2 bytes)	Size of Code Flash Secure region [KB]. For example: 0100h -> 01h, 00h (256 KB) * 32 KB align
RSV	(2 bytes)	0000h (unused code)
RSV	(2 bytes)	0000h (unused code)
SFS	(2 bytes)	Size of SiP Flash Secure region [KB] For example: 0100h -> 01h, 00h (256 KB) * 32 KB align Note that this field is reserved and ignored for products without SiP Flash.
SUM	(1 byte)	Sum data
ETX	(1 byte)	03h

* If CMS and SFS does not comply with alignment, boot firmware rounds down them.

6.13.2.2 Data Packet [Status OK]

SOD	(1 byte)	81h
LNH	(1 byte)	00h
LNL	(1 byte)	0Ah
RES	(1 byte)	4Eh (OK)
STS	(1 byte)	00h (OK)
ST2	(4 bytes)	FFFFFFFFh (unused code)
ADR	(4 bytes)	FFFFFFFFh (unused code)
SUM	(1 byte)	Sum data
ETX	(1 byte)	03h

6.13.2.3 Data Packet [Status ERR]

SOD	(1 byte)	81h
LNH	(1 byte)	00h
LNL	(1 byte)	0Ah
RES	(1 byte)	CEh (ERR)
STS	(1 byte)	Status code
ST2	(4 bytes)	Status details
ADR	(4 bytes)	Failure address
SUM	(1 byte)	Sum data
ETX	(1 byte)	03h

6.13.3 Processing Procedure

Boot firmware receives and analyzes a command packet:

- The boot firmware recognizes the start of the command packet by receiving SOH.
  If the boot firmware receives something other than SOH, it waits until it receives SOH.
- If ETX is not added to the received command packet, the boot firmware sends a “Packet error”.
- If the SUM of the received command packet is different from the sum value, the boot firmware sends a “Checksum error”.
- If the LNH and LNL of the received command packet are different from the values specified in the packet format, the boot firmware sends a “Packet error”.
- If the LNH and LNL of the received command packet are different from the values specified in each command, the boot firmware sends a “Packet error”.
- When any of the above errors occurs, the boot firmware does not process and returns to the command waiting state.
  * Memory contents do not change before command reception.

When the processing above is successfully completed, boot firmware executes the acceptance analysis:

- If this command cannot be executed in the current DLM state, the boot firmware sends a “Command acceptance error”.
- If the device reset is not asserted after the Encrypted data write command execution, the boot firmware sends a “Command acceptance error”.
- If the current Authentication level is AL1 or AL0, the boot firmware sends a “Secure error”.
- When any of the above errors occurs, the boot firmware does not process and returns to the command waiting state.
  * Memory contents do not change before command reception.

When the processing above is successfully completed, boot firmware writes the boundary setting:

- If an error occurs while writing, the boot firmware sends a “MRAM sequencer error” and returns to the command wait state.
- When the write processing is normally finished, boot firmware returns “OK” and waits for the next command.

6.13.4 Status Information from the Microcontroller

(Listed in descending order of priority.)

Condition	STS	ST2	ADR
The received packet does not have ETX.	Packet error	FFFFFFFFh	FFFFFFFFh
Sum data in the packet received is different from the value calculated by the boot firmware.	Checksum error	FFFFFFFFh	FFFFFFFFh
Packet length in the received packet does not comply with the packet format.	Packet error	FFFFFFFFh	FFFFFFFFh
Packet length in the received packet does not comply with the specifications of this command.	Packet error	FFFFFFFFh	FFFFFFFFh
Executing this command is unavailable in the current DLM state.	Command acceptance error	FFFFFFFFh	FFFFFFFFh
Device reset is not asserted after Encrypted data write command execution.	Command acceptance error	FFFFFFFFh	FFFFFFFFh
Authentication level is AL1 or AL0.	Secure error	FFFFFFFFh	FFFFFFFFh
MACI detected an error after the command execution in an undisclosed area.	MRAM sequence error	MRAM status	FFFFFFFFh
Successful completion.	OK	FFFFFFFFh	FFFFFFFFh

6.13.5 Example of Use

The relationship between boundary settings and secure regions are shown below.

Example: CMS=0200h, SFS=0800h

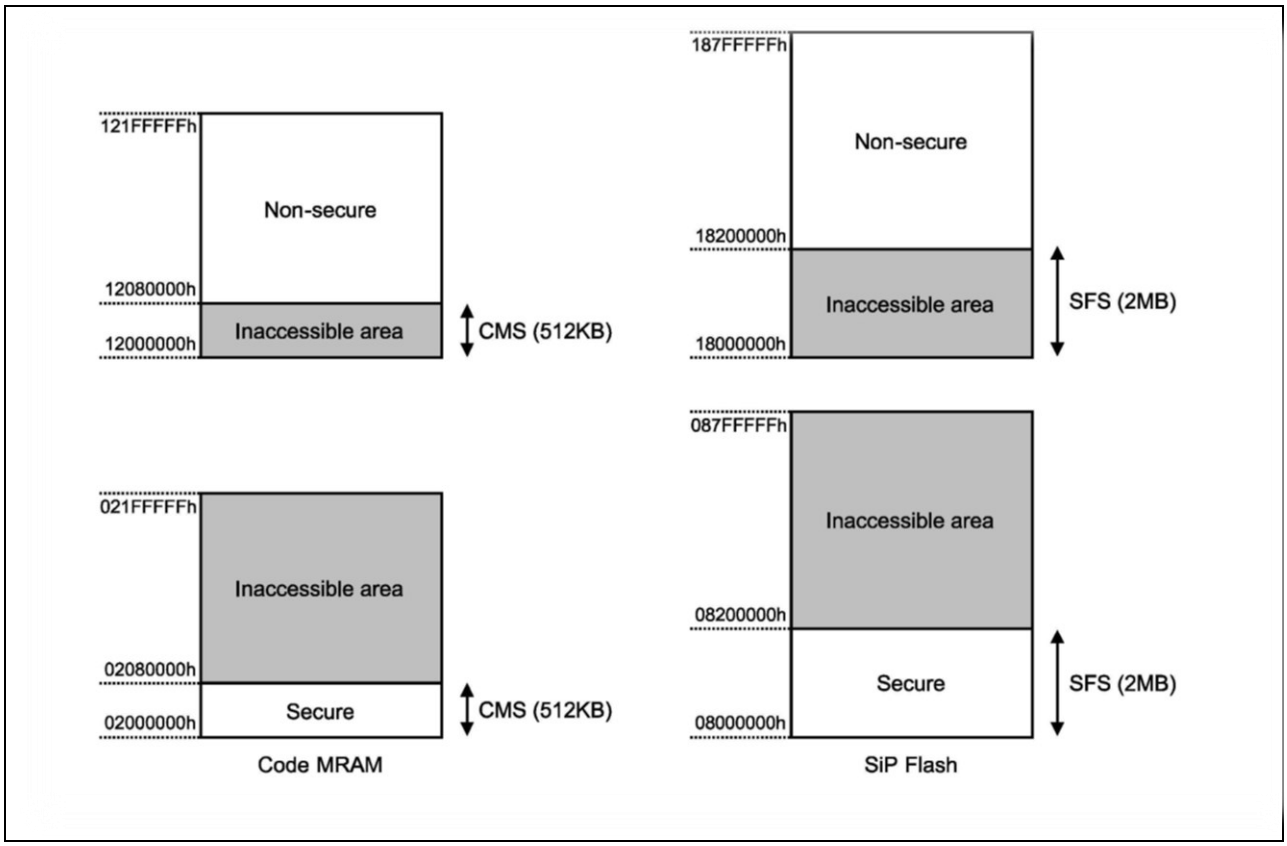


Figure 32. Boundary Setting Example

6.14 Boundary Request Command

This command sends the boundary setting value to the host. (Returns the value currently stored in the device.)

This command requires adherence to conditions described in Command List.

6.14.1 Sequence Diagram

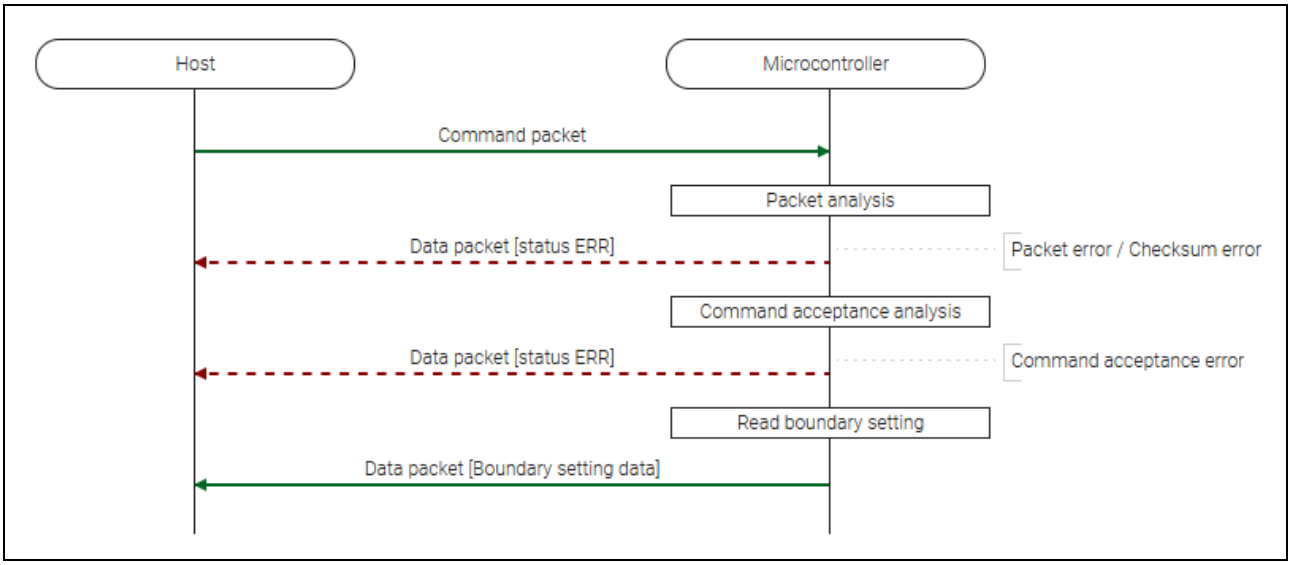


Figure 33. Boundary Request Command Sequence Diagram

6.14.2 Packets

6.14.2.1 Command Packet

SOH	(1 byte)	01h
LNH	(1 byte)	00h
LNL	(1 byte)	01h
CMD	(1 byte)	4Fh (Boundary request command)
SUM	(1 byte)	B0h
ETX	(1 byte)	03h

6.14.2.2 Data packet [Boundary Setting Data]

SOD	(1 byte)	81h
LNH	(1 byte)	00h
LNL	(1 byte)	0Bh
RES	(1 byte)	4Fh (OK)
RSV	(2 bytes)	0000h (unused code)
CMS	(2 bytes)	Size of Code Flash Secure region [KB]. For example: 0100h->01h, 00h (256 KB)
RSV	(2 bytes)	0000h (unused code)
RSV	(2 bytes)	0000h (unused code)
SFS	(2 bytes)	Size of SiP Flash Secure region [KB] For example: 0100h -> 01h, 00h (256 KB) Note that this field is reserved and always 0000h for products without SiP Flash.
SUM	(1 byte)	Sum data
ETX	(1 byte)	03h

6.14.2.3 Data Packet [Status ERR]

SOD	(1 byte)	81h
LNH	(1 byte)	00h
LNL	(1 byte)	0Ah
RES	(1 byte)	CFh (ERR)
STS	(1 byte)	Status code
ST2	(4 bytes)	Status details
ADR	(4 bytes)	Failure address
SUM	(1 byte)	Sum data
ETX	(1 byte)	03h

6.14.3 Processing Procedure

Boot firmware receives and analyzes a command packet:

- The boot firmware recognizes the start of the command packet by receiving SOH.
  If the boot firmware receives something other than SOH, it waits until it receives SOH.
- If ETX is not added to the received command packet, the boot firmware sends a “Packet error”.
- If the SUM of the received command packet is different from the sum value, the boot firmware sends a “Checksum error”.
- If the LNH and LNL of the received command packet are different from the values specified in the packet format, the boot firmware sends a “Packet error”.
- If the LNH and LNL of the received command packet are different from the values specified in each command, the boot firmware sends a “Packet error”.
- When any of the above errors occurs, the boot firmware does not process and returns to the command waiting state.
  * Memory contents do not change before command reception.

When the processing above is successfully completed, boot firmware executes the acceptance analysis:

- If the device reset is not asserted after the Encrypted data write command execution, the boot firmware sends a “Command acceptance error”.
- When the above error occurs, the boot firmware does not process and returns to the command waiting state.
  * Memory contents do not change before command reception.

When the processing above is successfully completed, boot firmware returns boundary setting.

- Boot firmware sends “Boundary information” and waits for next command.
  * Memory contents do not change before command reception.

6.14.4 Status Information from the Microcontroller

(Listed in descending order of priority.)

Condition	STS	ST2	ADR
The received packet does not have ETX.	Packet error	FFFFFFFFh	FFFFFFFFh
Sum data in the received packet is different from the value calculated by the boot firmware.	Checksum error	FFFFFFFFh	FFFFFFFFh
Packet length in the received packet does not comply with the packet format.	Packet error	FFFFFFFFh	FFFFFFFFh
Packet length in the received packet does not comply with the specifications of this command.	Packet error	FFFFFFFFh	FFFFFFFFh
Device reset is not asserted after Encrypted data write command execution.	Command acceptance error	FFFFFFFFh	FFFFFFFFh

6.15 Parameter Setting Command

This command stores the received parameter in the device.

This command requires adherence to conditions described in Command List.

6.15.1 Sequence Diagram

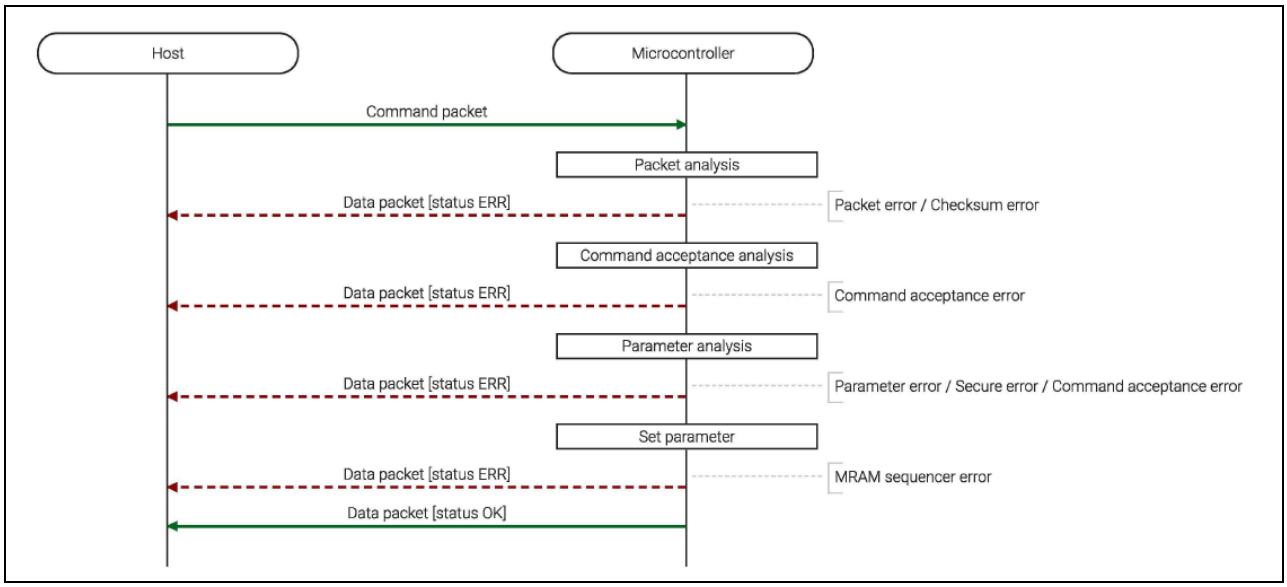


Figure 34. Parameter Setting Command Sequence Diagram

6.15.2 Packets

6.15.2.1 Command Packet

<table>
<tr><td>SOH</td><td>(1 byte)</td><td>01h</td></tr>
<tr><td>LNH</td><td>(1 byte)</td><td>00h</td></tr>
<tr><td>LNL</td><td>(1 byte)</td><td>03h</td></tr>
<tr><td>CMD</td><td>(1 byte)</td><td>51h (Parameter setting command)</td></tr>
<tr><td>PMID</td><td>(1 byte)</td><td>Parameter ID Specifiable parameter:
<table>
<tr><th>PMID</th><th>Parameter description</th><th>Specifiable at</th><th>Specifiable after Encrypted data write command</th></tr>
<tr><td>01h</td><td>Disable initialization</td><td>AL2/AL1/AL0</td><td>Specifiable</td></tr>
<tr><td>02h</td><td>Disable LCK_BOOT</td><td>AL2/AL1</td><td>Specifiable</td></tr>
<tr><td>03h</td><td>Disable AL2_key</td><td>AL2</td><td>Specifiable</td></tr>
<tr><td>04h</td><td>Disable AL1_key</td><td>AL2/AL1</td><td>Non-specifiable</td></tr>
<tr><td>05h</td><td>Master CPU selection</td><td>AL2</td><td>Non-specifiable</td></tr>
<tr><td>06h</td><td>Disable security extension of CPU1</td><td>AL2</td><td>Non-specifiable</td></tr>
</table></td></tr>
<tr><td>PRMT</td><td>(1 byte)</td><td>Parameter data:
• [PMID=01h]
- 00h: Disable initialization
• [PMID=02h]
- 00h: Disable transition to LCK_BOOT
• [PMID=03h]
- 00h: Disable of authentication using AL2_KEY (*1)
• [PMID=04h]
- 00h: Disable of authentication using AL1_KEY
• [PMID=05h)
- Change master CPU from CPU0 to CPU1
• [PMID=06h)
- 00h: CPU1 security extension is disabled</td></tr>
<tr><td>SUM</td><td>(1 byte)</td><td>Sum data</td></tr>
<tr><td>ETX</td><td>(1 byte)</td><td>03h</td></tr>
</table>

*1: When disabled, initialization and transition to RMA_REQ are also impossible.

6.15.2.2 Data Packet [Status OK]

SOD	(1 byte)	81h
LNH	(1 byte)	00h
LNL	(1 byte)	0Ah
RES	(1 byte)	51h (OK)
STS	(1 byte)	00h (OK)
ST2	(4 bytes)	FFFFFFFFh (unused code)
ADR	(4 bytes)	FFFFFFFFh (unused code)
SUM	(1 byte)	Sum data
ETX	(1 byte)	03h

6.15.2.3 Data Packet [Status ERR]

SOD	(1 byte)	81h
LNH	(1 byte)	00h
LNL	(1 byte)	0Ah
RES	(1 byte)	D1h (ERR)
STS	(1 byte)	Status code
ST2	(4 bytes)	Status details
ADR	(4 bytes)	Failure address
SUM	(1 byte)	Sum data
ETX	(1 byte)	03h

6.15.3 Processing Procedure

Boot firmware receives and analyzes a command packet:

- The boot firmware recognizes the start of the command packet by receiving SOH.
  If the boot firmware receives something other than SOH, it waits until it receives SOH.
- If ETX is not added to the received command packet, the boot firmware sends a “Packet error”.
- If the SUM of the received command packet is different from the sum value, the boot firmware sends a “Checksum error”.
- If the LNH and LNL of the received command packet are different from the values specified in the packet format, the boot firmware sends a “Packet error”.
- If the LNH and LNL of the received command packet are different from the values specified in each command, the boot firmware sends a “Packet error”.
- When any of the above errors occurs, the boot firmware does not process and returns to the command waiting state.
  * Memory contents do not change before command reception.

When the processing above is successfully completed, boot firmware executes the acceptance analysis:

- If this command cannot be executed in the current DLM state, the boot firmware sends a “Command acceptance error”.
- When the above error occurs, the boot firmware does not process and returns to the command waiting state.
  * Memory contents do not change before command reception.

When the processing above is successfully completed, boot firmware analyzes the command parameters:

- When designated PMID is unsupported, “Parameter error” is returned.
- When designated PMID is cannot be set in the current Authentication level, “Secure error” is returned.
- If both the following conditions are met, the boot firmware sends a “Command acceptance error”:
  - Device reset is not asserted after Encrypted data write command execution.
  - Parameter ID that is non-specifiable after Encrypted data write command is specified.
- If PRMT is not the specified value, the boot firmware sends a “Parameter error” and returns to the command wait state.
- When any of the above errors occurs, the boot firmware does not process and returns to the command waiting state.
  * Memory contents do not change before command reception.

When the processing above is successfully completed, boot firmware writes parameter setting.

- If an error occurs while writing, the boot firmware sends a “MRAM sequencer error” and returns to the command wait state.
- When the write processing is normally finished, boot firmware returns “OK” and waits for the next command.

6.15.4 Status Information from the Microcontroller

(Listed in descending order of priority.)

Condition	STS	ST2	ADR
The received packet does not have ETX.	Packet error	FFFFFFFFh	FFFFFFFFh
Sum data in the packet received is different from the value calculated by the boot firmware.	Checksum error	FFFFFFFFh	FFFFFFFFh
Packet length in the received packet does not comply with the packet format.	Packet error	FFFFFFFFh	FFFFFFFFh
Packet length in the received packet does not comply with the specifications of this command.	Packet error	FFFFFFFFh	FFFFFFFFh
Executing this command is unavailable in the current DLM state.	Command acceptance error	FFFFFFFFh	FFFFFFFFh
The specified Parameter ID is an unsupported value.	Parameter error	FFFFFFFFh	FFFFFFFFh
The specified Parameter ID cannot be set at the current Authentication level.	Secure error	FFFFFFFFh	FFFFFFFFh
Both the following conditions are met: • Device reset is not asserted after Encrypted data write command execution. • Parameter ID that is non-specifiable after Encrypted data write command is specified.	Command acceptance error	FFFFFFFFh	FFFFFFFFh
Parameter data is not the specified value.	Parameter error	FFFFFFFFh	FFFFFFFFh
MACI detected an error after the command execution in an undisclosed area.	MRAM sequence error	MRAM status	FFFFFFFFh
Successful completion.	OK	FFFFFFFFh	FFFFFFFFh

6.15.5 Parameter Details

The following shows the parameter data (PRMT) details.

[Disable setting for the function]

- PRMT[2:0]: 000b
- PRMT[7:3]: Any value can be specified (ignored when writing).

* PRMT[2:0] accepts only 000b. If the specified parameter has been already set, the boot firmware does not write but returns OK.

* Once disabled, the function cannot be enabled again.

6.16 Parameter Request Command

This command reads the specified parameter from the device and sends it to the host. (Returns the value currently stored in the device.)

This command requires adherence to conditions described in Command List.

6.16.1 Sequence Diagram

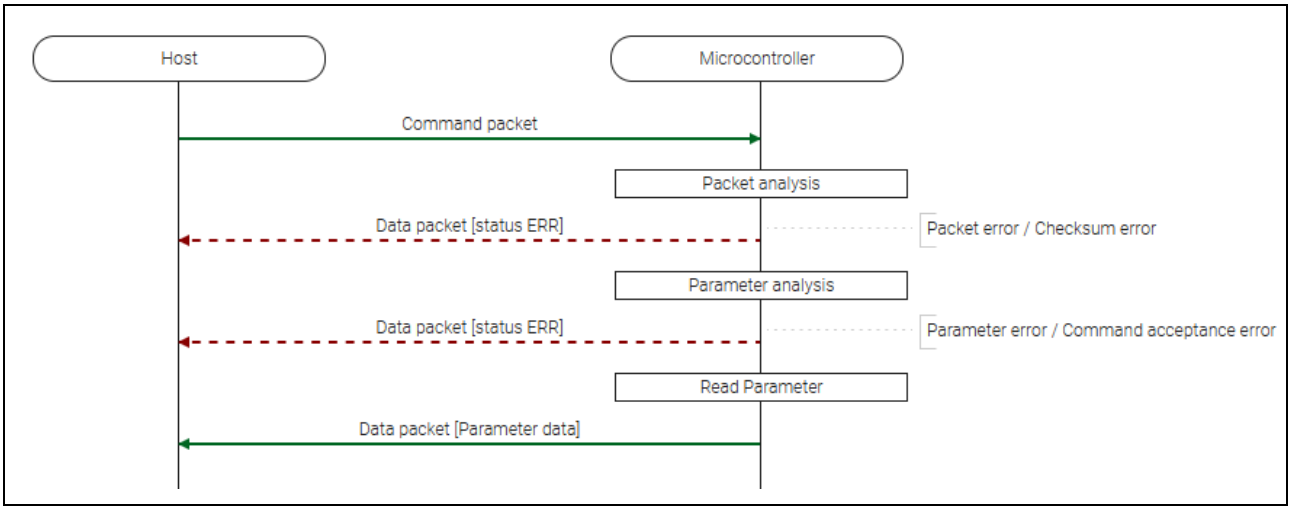


Figure 35. Parameter Request Command Sequence Diagram

6.16.2 Packets

6.16.2.1 Command Packet

<table>
<tr><td>SOH</td><td>(1 byte)</td><td>01h</td></tr>
<tr><td>LNH</td><td>(1 byte)</td><td>00h</td></tr>
<tr><td>LNL</td><td>(1 byte)</td><td>02h</td></tr>
<tr><td>CMD</td><td>(1 byte)</td><td>52h (Parameter request command)</td></tr>
<tr><td>PMID</td><td>(1 byte)</td><td>Parameter ID
Specifiable parameter:
<table>
<tr><th>PMID</th><th>Parameter description</th><th>Specifiable after Encrypted data write command</th></tr>
<tr><td>01h</td><td>Disable initialization</td><td>Specifiable</td></tr>
<tr><td>02h</td><td>Disable LCK_BOOT</td><td>Specifiable</td></tr>
<tr><td>03h</td><td>Disable AL2_key</td><td>Specifiable</td></tr>
<tr><td>04h</td><td>Disable AL1_key</td><td>Non-specifiable</td></tr>
<tr><td>05h</td><td>Master CPU selection</td><td>Non-specificable</td></tr>
<tr><td>06h</td><td>Disable security extension of CPU1</td><td>Non-specifiable</td></tr>
</table></td></tr>
<tr><td>SUM</td><td>(1 byte)</td><td>Sum data</td></tr>
<tr><td>ETX</td><td>(1 byte)</td><td>03h</td></tr>
</table>

6.16.2.2 Data Packet [Parameter Data]

SOD	(1 byte)	81h
LNH	(1 byte)	00h
LNL	(1 byte)	02h
RES	(1 byte)	52h (OK)
PRMT	(1 byte)	Parameter data: • [PMID=01h] - 00h: Initialization is disabled. - 07h: Initialization is enabled. • [PMID=02h] - 00h: Transition to LCK_BOOT is disabled. - 07h: Transition to LCK_BOOT is enabled. • [PMID=03h] - 00h: Authentication using AL2_KEY is disabled (*1). - 07h: Authentication using AL2_KEY is enabled. • [PMID=04h] - 00h: Authentication using AL1_KEY is disabled. - 07h: Authentication using AL1_KEY is enabled. • [PMID=05h) - 00h • CPU1 is the master CPU - 07h: CPUO is the master CPU • [PMID=06h) - 00h: CPU1 security extension is disabled - 07h: CPU1 security extension is enabled
SUM	(1 byte)	Sum data
ETX	(1 byte)	03h

*1: When disabled, initialization and transition to RMA_REQ are also impossible.

6.16.2.3 Data Packet [Status ERR]

SOD	(1 byte)	81h
LNH	(1 byte)	00h
LNL	(1 byte)	0Ah
RES	(1 byte)	D2h (ERR)
STS	(1 byte)	Status code
ST2	(4 bytes)	Status details
ADR	(4 bytes)	Failure address
SUM	(1 byte)	Sum data
ETX	(1 byte)	03h

6.16.3 Processing Procedure

Boot firmware receives and analyzes a command packet:

- The boot firmware recognizes the start of the command packet by receiving SOH.
  If the boot firmware receives something other than SOH, it waits until it receives SOH.
- If ETX is not added to the received command packet, the boot firmware sends a “Packet error”.
- If the SUM of the received command packet is different from the sum value, the boot firmware sends a “Checksum error”.
- If the LNH and LNL of the received command packet are different from the values specified in the packet format, the boot firmware sends a “Packet error”.
- If the LNH and LNL of the received command packet are different from the values specified in each command, the boot firmware sends a “Packet error”.
- When any of the above errors occurs, the boot firmware does not process and returns to the command waiting state.
  * Memory contents do not change before command reception.

When the processing above is successfully completed, boot firmware analyzes the command parameters:

- When designated PMID is unsupported, “Parameter error” is returned.
- If both the following conditions are met, the boot firmware sends a “Command acceptance error”:
- Device reset is not asserted after Encrypted data write command execution.
- Parameter ID that is non-specifiable after Encrypted data write command is specified.
- When any of the above errors occurs, the boot firmware does not process and returns to the command waiting state.
  * Memory contents do not change before command reception.

When the processing above is successfully completed, boot firmware returns the parameter setting:

- Boot firmware sends a parameter and waits for the next command.
  * Memory contents do not change before command reception.

6.16.4 Status Information from the Microcontroller

(Listed in descending order of priority.)

Condition	STS	ST2	ADR
The received packet does not have ETX.	Packet error	FFFFFFFFh	FFFFFFFFh
Sum data in the packet received is different from the value calculated by the boot firmware.	Checksum error	FFFFFFFFh	FFFFFFFFh
Packet length in the received packet does not comply with the packet format.	Packet error	FFFFFFFFh	FFFFFFFFh
Packet length in the received packet does not comply with the specifications of this command.	Packet error	FFFFFFFFh	FFFFFFFFh
The specified Parameter ID is an unsupported value.	Parameter error	FFFFFFFFh	FFFFFFFFh
Both the following conditions are met: • Device reset is not asserted after Encrypted data write command execution. • Parameter ID that is non-specifiable after Encrypted data write command is specified.	Command acceptance error	FFFFFFFFh	FFFFFFFFh

6.16.5 Parameter Details

The following details shows the parameter data (PRMT).

[The function is disabled]

- PRMT[2:0]: 000b
- PRMT[7:3]: Always returns 0

[The function is enabled]

- PRMT[2:0]: 111b
- PRMT[7:3]: Always returns 0

6.17 ARC Configuration Setting Command

This command sets the received Anti-Rollback Counter configuration data to the device.

This command requires adherence to conditions described in Command List.

6.17.1 Sequence Diagram

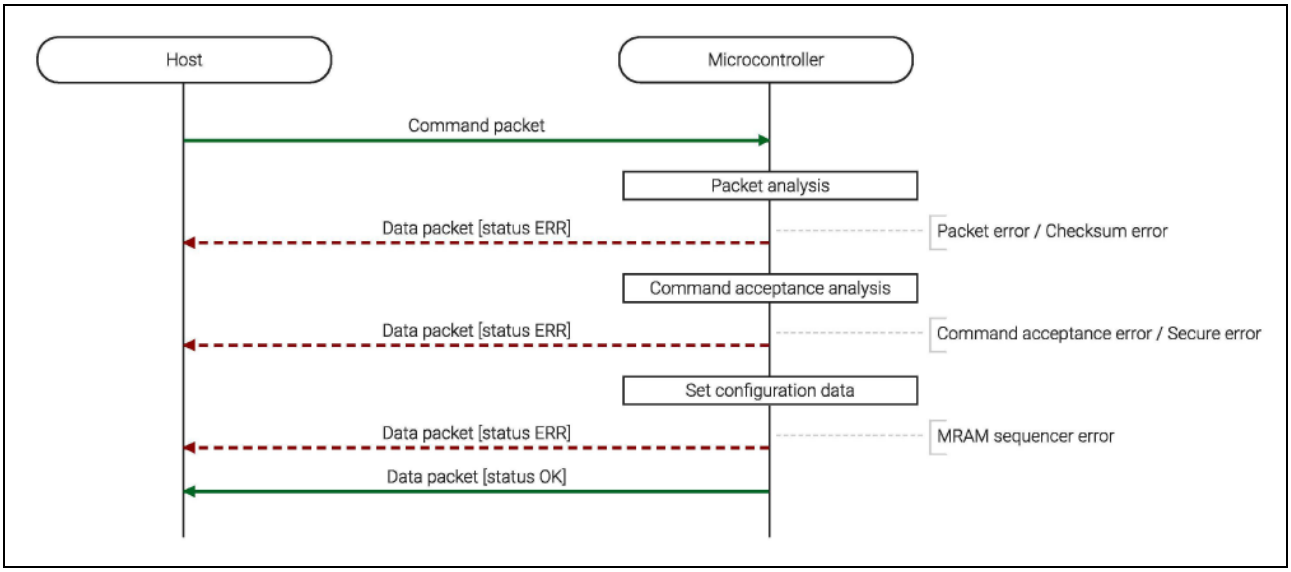


Figure 36. ARC Configuration Setting Command Sequence Diagram

6.17.2 Packets

6.17.2.1 Command Packet

SOH	(1 byte)	01h
LNH	(1 byte)	00h
LNL	(1 byte)	05h
CMD	(1 byte)	4Ch (ARC configuration setting command)
ARC	(4 bytes)	Anti-Rollback Configuration data. First received data is written to lower address of ARC configuration area.
SUM	(1 byte)	Sum data
ETX	(1 byte)	03h

6.17.2.2 Data Packet [Status OK]

SOD	(1 byte)	81h
LNH	(1 byte)	00h
LNL	(1 byte)	0Ah
RES	(1 byte)	4Ch (OK)
STS	(1 byte)	00h (OK)
ST2	(4 bytes)	FFFFFFFFh (unused code)
ADR	(4 bytes)	FFFFFFFFh (unused code)
SUM	(1 byte)	Sum data
ETX	(1 byte)	03h

6.17.2.3 Data Packet [Status ERR]

SOD	(1 byte)	81h
LNH	(1 byte)	00h
LNL	(1 byte)	0Ah
RES	(1 byte)	CCh (ERR)
STS	(1 byte)	Status code
ST2	(4 bytes)	Status details
ADR	(4 bytes)	Failure address
SUM	(1 byte)	Sum data
ETX	(1 byte)	03h

6.17.3 Processing Procedure

Boot firmware receives and analyzes a command packet:

- The boot firmware recognizes the start of the command packet by receiving SOH.
  If the boot firmware receives something other than SOH, it will wait until it receives SOH.
- If ETX is not added to the received command packet, the boot firmware sends a “Packet error”.
- If the SUM of the received command packet is different from the sum value, the boot firmware sends a “Checksum error”.
- If the LNH and LNL of the received command packet are different from the values specified in the packet format, the boot firmware sends a “Packet error”.
- If the LNH and LNL of the received command packet are different from the values specified in each command, the boot firmware sends a “Packet error”.
- When any of the above errors occurs, the boot firmware does not process and returns to the command waiting state.
  * Memory contents do not change before command reception.

When the processing above is successfully completed, boot firmware executes the acceptance analysis:

- If this command cannot be executed in the current DLM state, the boot firmware sends a “Command acceptance error”.
- If the current Authentication level is AL1 or AL0, the boot firmware sends a “Secure error” and returns to the command waiting state.
- When any of the above errors occurs, the boot firmware does not process and returns to the command waiting state.
  * Memory contents do not change before command reception.

When the processing above is successfully completed, boot firmware writes Anti-Rollback Counter setting:

- If an error occurs while writing, the boot firmware sends a “MRAM sequencer error” and returns to the command wait state.
- When the write processing is normally finished, boot firmware returns “OK” and waits for the next command.

6.17.4 Status Information from the Microcontroller

(Listed in descending order of priority.)

Condition	STS	ST2	ADR
The received packet does not have ETX.	Packet error	FFFFFFFFh	FFFFFFFFh
Sum data in the received packet is different from the value calculated by the boot firmware.	Checksum error	FFFFFFFFh	FFFFFFFFh
Packet length in the received packet does not comply with the packet format.	Packet error	FFFFFFFFh	FFFFFFFFh
Packet length in the received packet does not comply with the specifications of this command.	Packet error	FFFFFFFFh	FFFFFFFFh
Executing this command is unavailable in the current DLM state.	Command acceptance error	FFFFFFFFh	FFFFFFFFh
Current Authentication level is AL1 or AL0.	Secure error	FFFFFFFFh	FFFFFFFFh
MACI detected an error after the command execution.	MRAM sequence error	MRAM status	Failure address
Successful completion.	OK	FFFFFFFFh	FFFFFFFFh

6.17.5 Mapping of Anti-Rollback Counter Configuration Data

Refer to User's Manual of the device for the mapping of Anti-Rollback counter configuration data.

Table 21. Mapping of RA8M2 MCU Group

Address	Bit	Data
02E1_7930h	15:6	(Reserved)
	5	ARCBL_LK
	4:1	ARCNS_LK[3:0]
	0	ARCS_LK
02E1_7931h	15:0	(Reserved)
02E1_7932h	15:2	(Reserved)
	1:0	CNF_ARCNS[1:0]
02E1_7933h	15:0	(Reserved)

6.18 ARC Configuration Request Command

This command reads Anti-Rollback Counter configuration data and sends them to the host.

This command requires adherence to conditions described in Command List.

6.18.1 Sequence Diagram

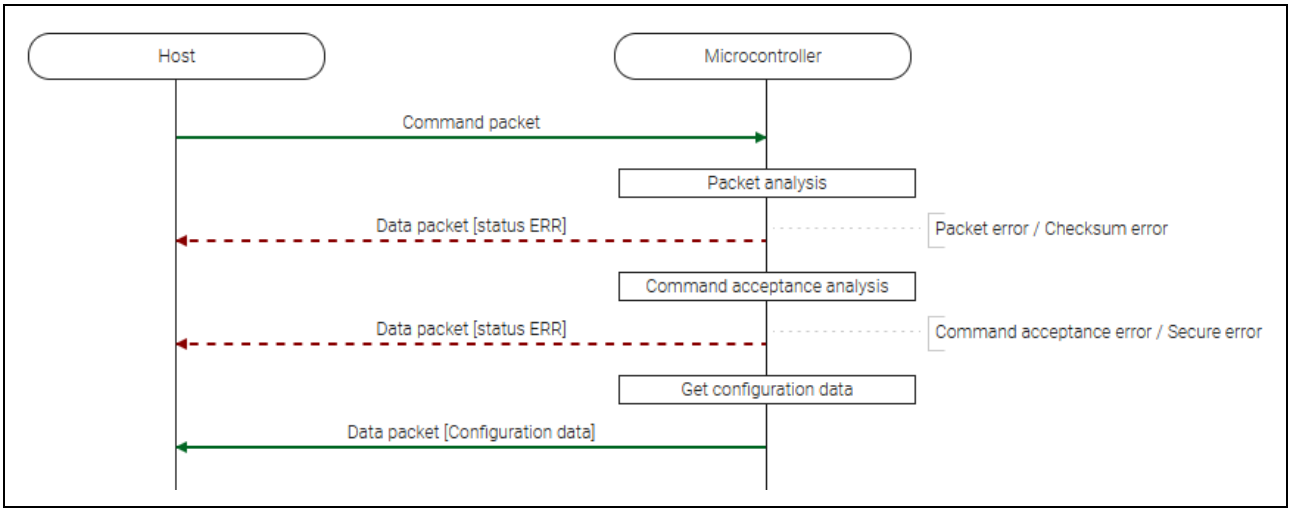


Figure 37. ARC Configuration Request Command Sequence Diagram

6.18.2 Packets

6.18.2.1 Command Packet

SOH	(1 byte)	01h
LNH	(1 byte)	00h
LNL	(1 byte)	01h
CMD	(1 byte)	4Dh (ARC configuration request command)
SUM	(1 byte)	Sum data
ETX	(1 byte)	03h

6.18.2.2 Data Packet [Configuration Data]

SOD	(1 byte)	81h
LNH	(1 byte)	00h
LNL	(1 byte)	05h
RES	(1 byte)	4Dh (OK)
ARC	(4 bytes)	Anti-Rollback Configuration data. Data written in lower address of ARC configuration area is sent first.
SUM	(1 byte)	Sum data
ETX	(1 byte)	03h

6.18.2.3 Data Packet [Status ERR]

SOD	(1 byte)	81h
LNH	(1 byte)	00h
LNL	(1 byte)	0Ah
RES	(1 byte)	CDh (ERR)
STS	(1 byte)	Status code
ST2	(4 bytes)	Status details
ADR	(4 bytes)	Failure address
SUM	(1 byte)	Sum data
ETX	(1 byte)	03h

6.18.3 Processing Procedure

Boot firmware receives and analyzes a command packet:

- The boot firmware recognizes the start of the command packet by receiving SOH.
  If the boot firmware receives something other than SOH, it waits until it receives SOH.
- If ETX is not added to the received command packet, the boot firmware sends a “Packet error”.
- If the SUM of the received command packet is different from the sum value, the boot firmware sends a “Checksum error”.
- If the LNH and LNL of the received command packet are different from the values specified in the packet format, the boot firmware sends a “Packet error”.
- If the LNH and LNL of the received command packet are different from the values specified in each command, the boot firmware sends a “Packet error”.
- When any of the above errors occurs, the boot firmware does not process and returns to the command waiting state.
  * Memory contents do not change before command reception.

When the processing above is successfully completed, boot firmware executes the acceptance analysis:

- If this command cannot be executed in the current DLM state, the boot firmware sends a “Command acceptance error”.
- If the current Authentication level is AL1 or AL0, the boot firmware sends a “Secure error” and returns to the command waiting state.
- When any of the above errors occurs, the boot firmware does not process and returns to the command waiting state.
  * Memory contents do not change before command reception.

When the processing above is successfully completed, boot firmware returns Anti-Rollback Counter setting:

- Boot firmware send “Anti-Rollback Counter information” and waits for next command.
  * Memory contents do not change before command reception.

6.18.4 Status Information from the Microcontroller

(Listed in descending order of priority.)

Condition	STS	ST2	ADR
The received packet does not have ETX.	Packet error	FFFFFFFFh	FFFFFFFFh
Sum data in the received packet is different from the value calculated by the boot firmware.	Checksum error	FFFFFFFFh	FFFFFFFFh
Packet length in the received packet does not comply with the packet format.	Packet error	FFFFFFFFh	FFFFFFFFh
Packet length in the received packet does not comply with the specifications of this command.	Packet error	FFFFFFFFh	FFFFFFFFh
Executing this command is unavailable in the current DLM state.	Command acceptance error	FFFFFFFFh	FFFFFFFFh
Current Authentication level is AL1 or AL0.	Secure error	FFFFFFFFh	FFFFFFFFh

6.19 Inquiry Command

This command is used to check if boot firmware is “Command acceptable phase” or not.

This command requires adherence to conditions described in Command List.

6.19.1 Sequence Diagram

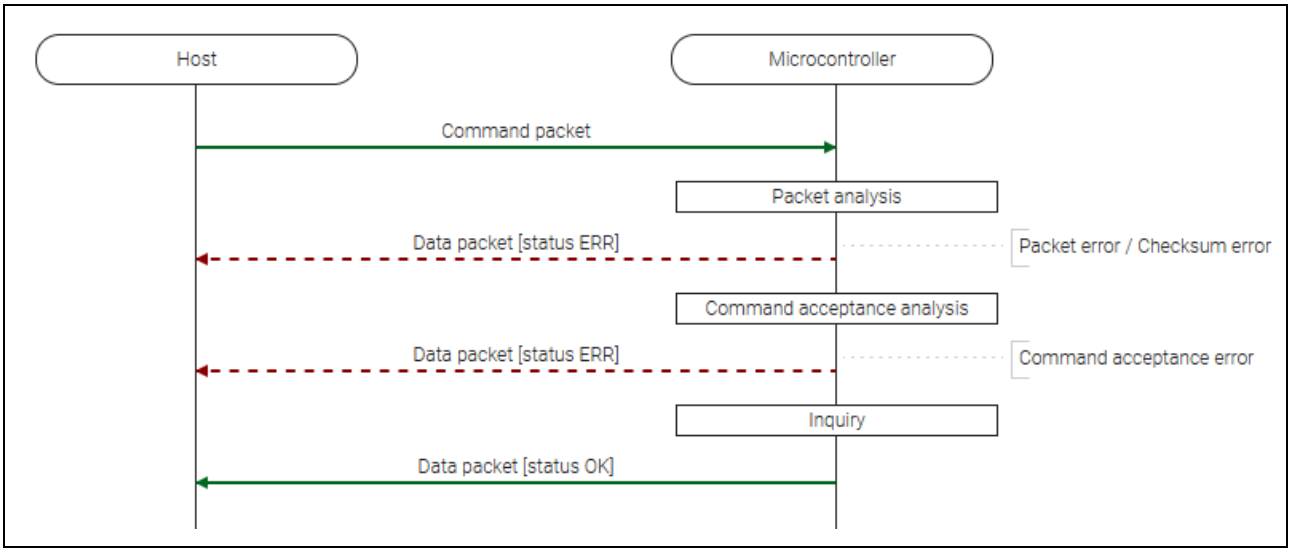


Figure 38. Inquiry Command Sequence Diagram

6.19.2 Packets

6.19.2.1 Command Packet

SOH	(1 byte)	01h
LNH	(1 byte)	00h
LNL	(1 byte)	01h
CMD	(1 byte)	00h (Inquiry command)
SUM	(1 byte)	FFh
ETX	(1 byte)	03h

6.19.2.2 Data Packet [Status OK]

SOD	(1 byte)	81h
LNH	(1 byte)	00h
LNL	(1 byte)	0Ah
RES	(1 byte)	00h (OK)
STS	(1 byte)	00h (OK)
ST2	(4 bytes)	FFFFFFFFh (unused code)
ADR	(4 bytes)	FFFFFFFFh (unused code)
SUM	(1 byte)	FEh
ETX	(1 byte)	03h

6.19.2.3 Data Packet [Status ERR]

SOD	(1 byte)	81h
LNH	(1 byte)	00h
LNL	(1 byte)	0Ah
RES	(1 byte)	80h (ERR)
STS	(1 byte)	Status code
ST2	(4 bytes)	Status details
ADR	(4 bytes)	Failure address
SUM	(1 byte)	Sum data
ETX	(1 byte)	03h

6.19.3 Processing Procedure

Boot firmware receives and analyzes a command packet:

- The boot firmware recognizes the start of the command packet by receiving SOH.
  If the boot firmware receives something other than SOH, it waits until it receives SOH.
- If ETX is not added to the received command packet, the boot firmware sends a “Packet error”.
- If the SUM of the received command packet is different from the sum value, the boot firmware sends a “Checksum error”.
- If the LNH and LNL of the received command packet are different from the values specified in the packet format, the boot firmware sends a “Packet error”.
- If the LNH and LNL of the received command packet are different from the values specified in each command, the boot firmware sends a “Packet error”.
- When any of the above errors occurs, the boot firmware does not process and returns to the command waiting state.
  * Memory status does not change before command reception.

When the processing above is successfully completed, boot firmware executes the acceptance analysis:

- If the device reset is not asserted after the Encrypted data write command execution, the boot firmware sends a “Command acceptance error”.
- When the above error occurs, the boot firmware does not process and returns to the command waiting state.
  * Memory status does not change before command reception.

When the processing above is successfully completed, boot firmware executes the inquiry processing:

- The boot firmware sends “OK”.
  * Memory status does not change before command reception.

6.19.4 Status Information from the Microcontroller

(Listed in descending order of priority.)

Condition	STS	ST2	ADR
The received packet does not have ETX.	Packet error	FFFFFFFFh	FFFFFFFFh
Sum data in the received packet is different from the value calculated by the boot firmware.	Checksum error	FFFFFFFFh	FFFFFFFFh
Packet length in the received packet does not comply with the packet format.	Packet error	FFFFFFFFh	FFFFFFFFh
Packet length in the received packet does not comply with the specifications of this command.	Packet error	FFFFFFFFh	FFFFFFFFh
Device reset is not asserted after Encrypted data write command execution.	Command acceptance error	FFFFFFFFh	FFFFFFFFh
The process has ended normally.	OK	FFFFFFFFh	FFFFFFFFh

6.20 Signature Request Command

This command sends the information of the device signature to the host.

This command requires adherence to conditions described in Command List.

6.20.1 Sequence Diagram

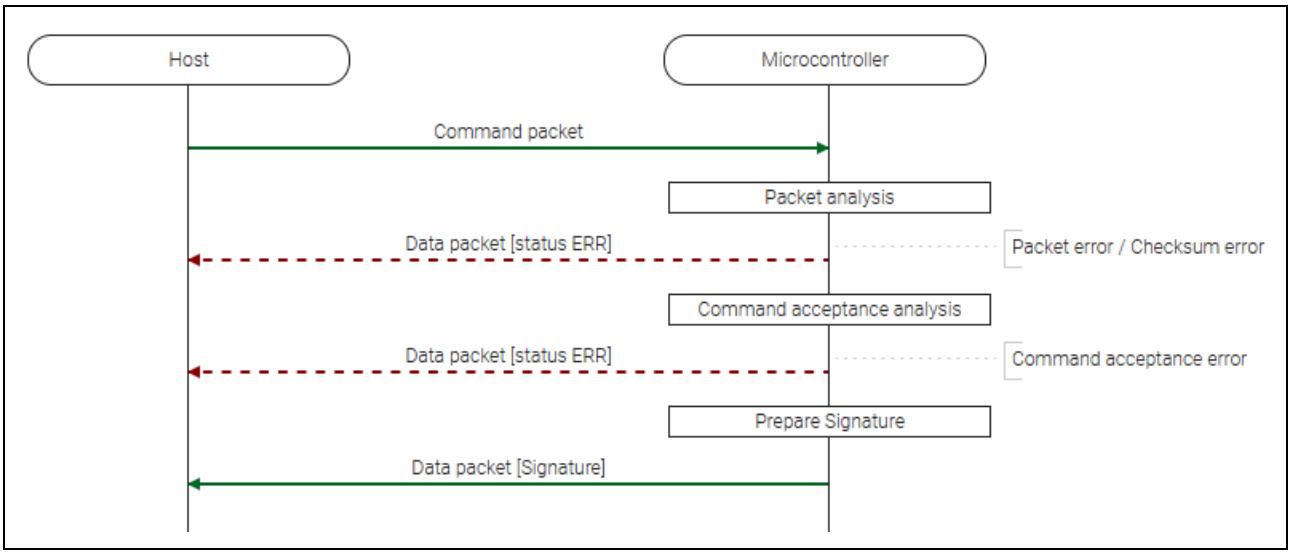


Figure 39. Signature Request Command Sequence Diagram

6.20.2 Packets

6.20.2.1 Command Packet

SOH	(1 byte)	01h
LNH	(1 byte)	00h
LNL	(1 byte)	01h
CMD	(1 byte)	3Ah (Signature request command)
SUM	(1 byte)	C5h
ETX	(1 byte)	03h

6.20.2.2 Data Packet [Signature]

SOD	(1 byte)	81h
LNH	(1 byte)	00h
LNL	(1 byte)	2Ah
RES	(1 byte)	3Ah (OK)
RMB	(4 bytes)	Recommended maximum UART baudrate of the device [bps]. *Order of sending: High -> ... -> Low For example: 6 Mbps (6000000bps) -> 00h, 5Bh, 8Dh, 80h
NOA	(1 byte)	Number of accessible areas For example, if the device has 4 areas -> 04h
TYP	(1 byte)	Type code (features and functions of the device): 07h : RA8M2 MCU Group
BFV	(3 byte)	Boot firmware version Order of sending: Major version -> minor version -> build For example, v2.4.1.6 -> 02h, 04h, 10h
DID	(16 bytes)	Device ID 16-byte ID code (unique ID) for identifying the particular MCU
PTN	(16 bytes)	Product type name. Character strings (20h for space) Order of sending example: R7FA6M3AH ->52h, 37h, 46h, 41h, 36h, 4dh, 33h, 41h, 48h, 20h, 20h, 20h, 20h, 20h, 20h, 20h
SUM	(1 byte)	Sum data
ETX	(1 byte)	03h

6.20.2.3 Data Packet [Status ERR]

SOD	(1 byte)	81h
LNH	(1 byte)	00h
LNL	(1 byte)	0Ah
RES	(1 byte)	BAh (ERR)
STS	(1 byte)	Status code
ST2	(4 bytes)	Status details
ADR	(4 bytes)	Failure address
SUM	(1 byte)	Sum data
ETX	(1 byte)	03h

6.20.3 Processing Procedure

Boot firmware receives and analyzes a command packet:

- The boot firmware recognizes the start of the command packet by receiving SOH.
  If the boot firmware receives something other than SOH, it waits until it receives SOH.
- If ETX is not added to the received command packet, the boot firmware sends a “Packet error”.
- If the SUM of the received command packet is different from the sum value, the boot firmware sends a “Checksum error”.
- If the LNH and LNL of the received command packet are different from the values specified in the packet format, the boot firmware sends a “Packet error”.
- If the LNH and LNL of the received command packet are different from the values specified in each command, the boot firmware sends a “Packet error”.
- When any of the above errors occurs, the boot firmware does not process and returns to the command waiting state.
  * Memory status does not change before command reception.

When the processing above is successfully completed, boot firmware executes the acceptance analysis:

- If the device reset is not asserted after the encrypted data write command execution, the boot firmware sends a “Command acceptance error”.
- When the above error occurs, the boot firmware does not process and returns to the command waiting state.
  * Memory status does not change before command reception.

When the processing above is successfully completed, boot firmware returns the signature.

- Send a signature and return to command while waiting.
  * Memory status does not change before command reception.

6.20.4 Status Information from the Microcontroller

(Listed in descending order of priority.)

Condition	STS	ST2	ADR
The received packet does not have ETX.	Packet error	FFFFFFFFh	FFFFFFFFh
Sum data in the received packet is different from the value calculated by the boot firmware.	Checksum error	FFFFFFFFh	FFFFFFFFh
Packet length in the received packet does not comply with the packet format.	Packet error	FFFFFFFFh	FFFFFFFFh
Packet length in the received packet does not comply with the specifications of this command.	Packet error	FFFFFFFFh	FFFFFFFFh
Device reset is not asserted after Encrypted data write command execution.	Command acceptance error	FFFFFFFFh	FFFFFFFFh

6.21 Area Information Request Command

This command sends information of the designated area to the host. The alignment of the target address of command follows this area information.

This command requires adherence to conditions described in Command List.

6.21.1 Sequence Diagram

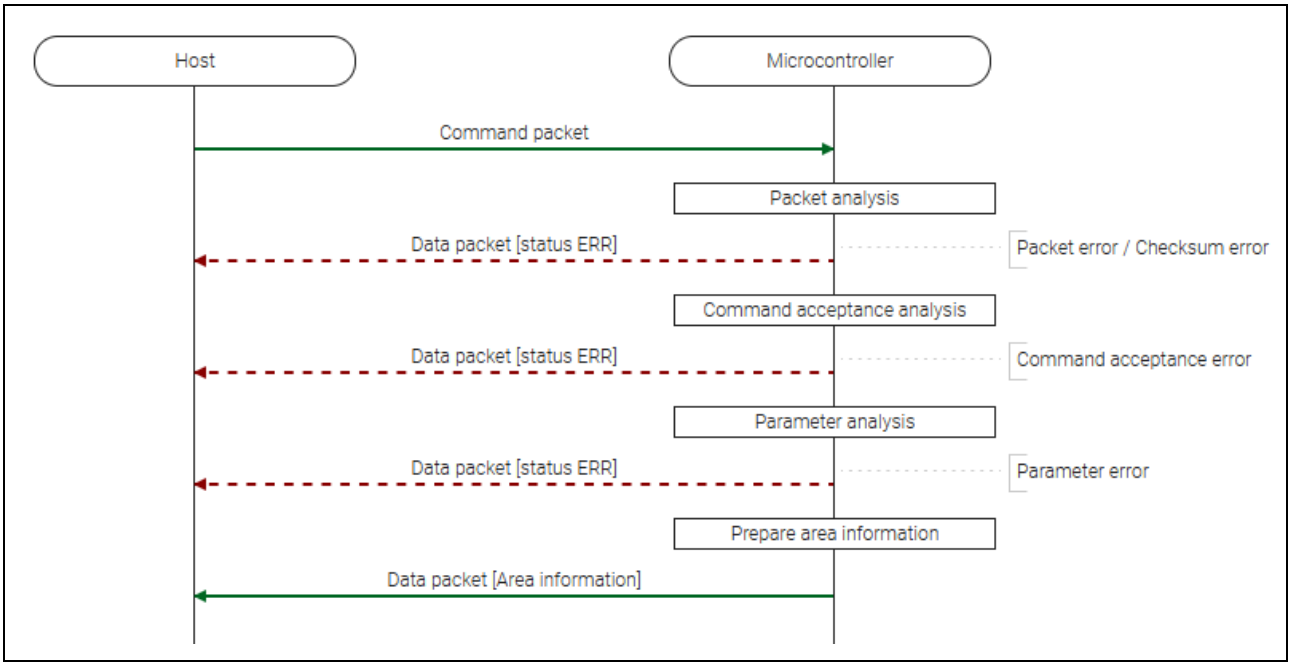


Figure 40. Area Information Request Command Sequence Diagram

6.21.2 Packets

6.21.2.1 Command Packet

SOH	(1 byte)	01h
LNH	(1 byte)	00h
LNL	(1 byte)	02h
CMD	(1 byte)	3Bh (Area information request command)
NUM	(1 byte)	Area number [0–NOA-1]
SUM	(1 byte)	Sum data
ETX	(1 byte)	03h

6.21.2.2 Data Packet [Area Information]

SOD	(1 byte)	81h
LNH	(1 byte)	00h
LNL	(1 byte)	1Ah
RES	(1 byte)	3Bh (OK)
KOA	(1 byte)	Kind of the area: • 0Nh: User area N (*2) • 2Nh: Config area N (*2) • 3Nh: Config OTP area N (*2) • 4Nh: External flash area N (*2) • 5Nh: SiP flash area N (*2)
SAD	(4 bytes)	Start address. Order of sending: High -> ... -> Low For example: 00010000h -> 00h, 01h, 00h, 00h
EAD	(4 bytes)	End address *Order of sending: High -> ... -> Low For example: 001FFFFFh -> 00h, 1Fh, FFh, FFh
EAU	(4 bytes)	Erase access unit (alignment) [byte] (*1) Order of sending: High -> ... -> Low For example: 32 KB (32768 byte) -> 00h, 00h, 80h, 00h Target command: Erase command.
WAU	(4 bytes)	Write access unit (alignment) [byte] (*1) Order of sending: High -> ... -> Low For example: 128 byte -> 00h, 00h, 00h, 80h Target command: Write command, User key setting command, User key verify command, Code certificate update command, Encrypted data write command.
RAU	(4 bytes)	Read access unit (alignment) [byte] (*1) Order of sending: High -> ... -> Low For example: 1 byte -> 00h, 00h, 00h, 01h Target command: Read command.
CAU	(4 bytes)	CRC access unit (alignment) [byte] (*1) Order of sending: High -> ... -> Low For example: 4 byte -> 00h, 00h, 00h, 04h Target command: CRC command
SUM	(1 byte)	Sum data
ETX	(1 byte)	03h

*1: If each access unit is 00000000h, target command is not available for the area.
*2: N = 0–F

6.21.2.3 Data Packet [Status ERR]

SOD	(1 byte)	81h
LNH	(1 byte)	00h
LNL	(1 byte)	0Ah
RES	(1 byte)	BBh (ERR)
STS	(1 byte)	Status code
ST2	(4 bytes)	Status details
ADR	(4 bytes)	Failure address
SUM	(1 byte)	Sum data
ETX	(1 byte)	03h

6.21.3 Processing Procedure

Boot firmware receives and analyzes a command packet:

- The boot firmware recognizes the start of the command packet by receiving SOH.
  If the boot firmware receives something other than SOH, it waits until it receives SOH.
- If ETX is not added to the received command packet, the boot firmware sends a “Packet error”.
- If the SUM of the received command packet is different from the sum value, the boot firmware sends a “Checksum error”.
- If the LNH and LNL of the received command packet are different from the values specified in the packet format, the boot firmware sends a “Packet error”.
- If the LNH and LNL of the received command packet are different from the values specified in each command, the boot firmware sends a “Packet error”.
- When any of the above errors occurs, the boot firmware does not process and returns to the command waiting state.
  * Memory status does not change before command reception.

When the processing above is successfully completed, boot firmware executes the acceptance analysis:

- If device reset is not asserted after the Encrypted data write command execution, the boot firmware sends a “Command acceptance error”.
- When the above error occurs, the boot firmware does not process and returns to the command waiting state.
  * Memory status does not change before command reception.

When the processing above is successfully completed, the boot firmware analyzes the command parameters:

- If the specified NUM is “NOA” returned by “Signature request command” or more, send “Parameter error” and return to command waiting status.
  * Memory status does not change before command reception.
- When the above error occurs, the boot firmware does not process and returns to the command waiting state.
  * Memory status does not change before command reception.

When the processing above is successfully completed, the area information is returned:

- Send area information of specified NUM and return to command waiting status.
  * Memory status does not change before command reception.

6.21.4 Status Information from the Microcontroller

(Listed in descending order of priority.)

Condition	STS	ST2	ADR
The received packet does not have ETX.	Packet error	FFFFFFFFh	FFFFFFFFh
Sum data in the received packet is different from the value calculated by the boot firmware.	Checksum error	FFFFFFFFh	FFFFFFFFh
Packet length in the received packet does not comply with the packet format.	Packet error	FFFFFFFFh	FFFFFFFFh
Packet length in the received packet does not comply with the specifications of this command.	Packet error	FFFFFFFFh	FFFFFFFFh
If Area number in the received packet is a non-existent area number.	Parameter error	FFFFFFFFh	FFFFFFFFh
Device reset is not asserted after Encrypted data write command execution.	Command acceptance error	FFFFFFFFh	FFFFFFFFh

6.21.5 Specific value of Area Information

Example: RA8T2 w/ 8MB SiP Flash

NUM	Area	KOA	SAD	EAD	EAU	WAU	RAU	CAU
0	User area 0 (*3)	00h	02000000h	020FFFFFh (*4)	0 (*1)	32B	1B	32KB
1	User area 1 (*3)	01h	12000000h	120FFFFFh (*4)	0 (*1)	32B	1B	32KB
2	Config area 0	20h	02C9F020h	02C9F0AFh	0 (*1)	16B	1B	16B
3	Config area 1	21h	02C9F0C0h	02C9F3FFh	0 (*1)	16B	1B	16B
4	Config area 2	22h	12C9F400h	12C9F7FFh	0 (*1)	16B	1B	16B
5	Config OTP area 0	30h	02E07600h	02E076FFh	0 (*1)	16B	1B	16B
6	Config OTP area 1	31h	02E17700h	02E1777Fh	0 (*1)	16B	1B	16B
7	Config OTP area 2	32h	12E17780h	12E177FFh	0 (*1)	16B	1B	16B
8	Config OTP area 3	33h	02E17900h	02E1792Fh	0 (*1)	16B	1B	16B
9	External flash area 0 (*2)	40h	80000000h	9FFFFFFFh	1B	1B	1B	1 KB
10	SiP flash area 0 (*3)	50h	08000000h	087FFFFFh (*4)	32KB	64B	1B	32KB
11	SiP flash area 1 (*3)	51h	18000000h	187FFFFFh (*4)	32KB	64B	1B	32KB

For RA8M2 w/o SiP Flash

NUM	Area	KOA	SAD	EAD	EAU	WAU	RAU	CAU
0~8	(Area 0~8 are the same as products w/ SiP Flash)	-	-	-	-	-	-	-
9	External flash area 0 (*2)	40h	70000000h	9FFFFFFFh	1B	1B	1B	1 KB

1. When Access unit is 0, it indicates that the corresponding operation is not supported.
2. Execute “External flash memory setting command” before accessing this area. Access to addresses to which no external flash memory is allocated is not guaranteed.
3. The accessible address changes depending on the boundary settings.
4. These addresses can change depending on the capacity of the MRAM and the SiP Flash of each product.

6.22 Baudrate Setting Command

This command receives baudrate data and change the UART baudrate of the device. If an error occurs, the baudrate is not changed. This command does not change the communication speed except for UART communication.

This command requires adherence to conditions described in Command List.

6.22.1 Sequence Diagram

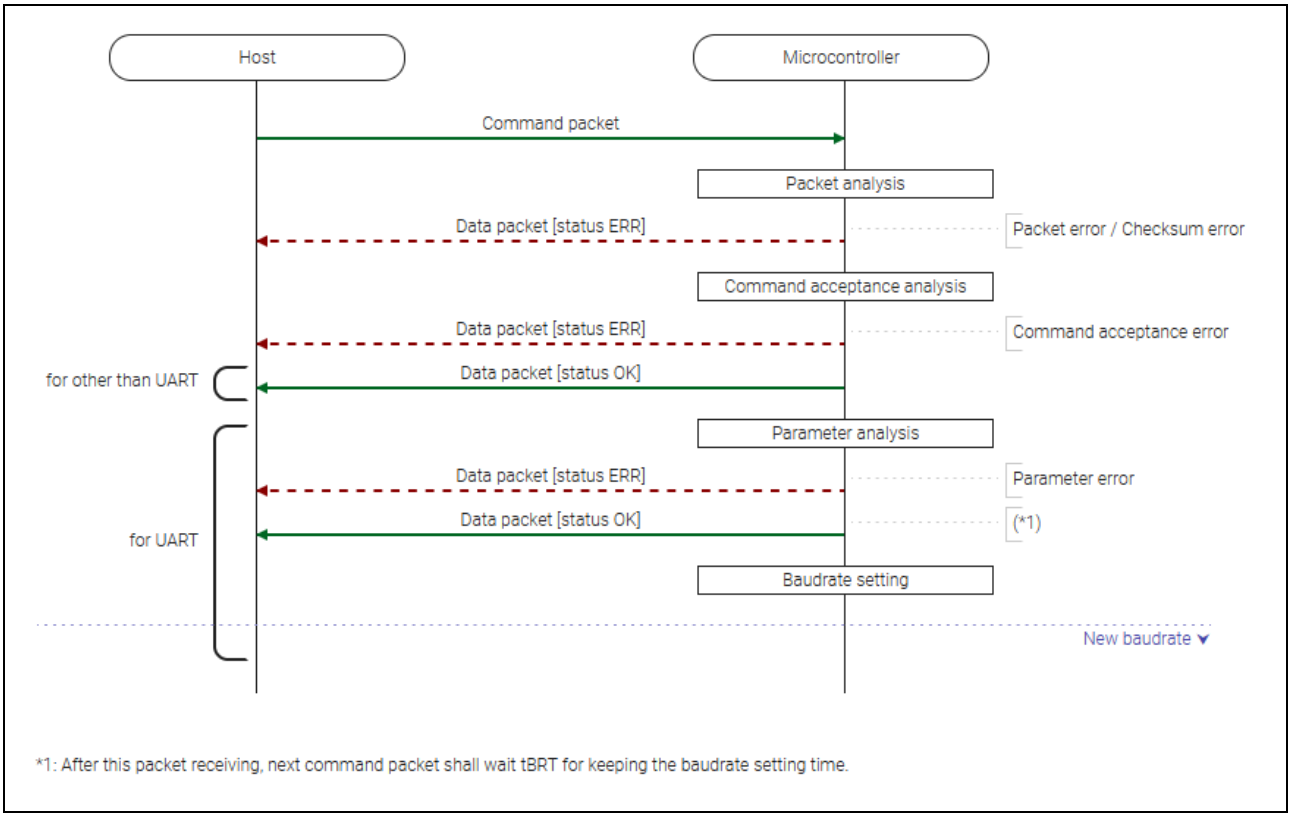


Figure 41. Baudrate Setting Command Sequence Diagram

6.22.2 Packets

6.22.2.1 Command Packet

SOH	(1 byte)	01h
LNH	(1 byte)	00h
LNL	(1 byte)	05h
CMD	(1 byte)	34h (Baudrate setting command)
BRT	(4 bytes)	UART baudrate [bps] You can set one of the following values. Order of sending BRT:
SUM	(1 byte)	Sum data
ETX	(1 byte)	03h

Baudrate	1st	2nd	3rd	4th
9600 bps	00	00	25	80
115200 bps	00	01	C2	00
500 Kbps	00	07	A1	20
1.0 Mbps	00	0F	42	40
1.5 Mbps	00	16	E3	60
2.0 Mbps	00	1E	84	80
4.0 Mbps	00	3D	09	00
6.0 Mbps	00	5B	8D	80

6.22.2.2 Data Packet [Status OK]

SOD	(1 byte)	81h
LNH	(1 byte)	00h
LNL	(1 byte)	0Ah
RES	(1 byte)	34h (OK)
STS	(1 byte)	00h (OK)
ST2	(4 bytes)	FFFFFFFFh (unused code)
ADR	(4 bytes)	FFFFFFFFh (unused code)
SUM	(1 byte)	CAh
ETX	(1 byte)	03h

6.22.2.3 Data Packet [Status ERR]

SOD	(1 byte)	81h
LNH	(1 byte)	00h
LNL	(1 byte)	0Ah
RES	(1 byte)	B4h (ERR)
STS	(1 byte)	Status code
ST2	(4 bytes)	Status details
ADR	(4 bytes)	Failure address
SUM	(1 byte)	Sum data
ETX	(1 byte)	03h

6.22.3 Processing Procedure

Boot firmware receives and analyzes a command packet:

- The boot firmware recognizes the start of the command packet by receiving SOH.
  If the boot firmware receives something other than SOH, it waits until it receives SOH.
- If ETX is not added to the received command packet, the boot firmware sends a “Packet error”.
- If the SUM of the received command packet is different from the sum value, the boot firmware sends a “Checksum error”.
- If the LNH and LNL of the received command packet are different from the values specified in the packet format, the boot firmware sends a “Packet error”.
- If the LNH and LNL of the received command packet are different from the values specified in each command, the boot firmware sends a “Packet error”.
- When any of the above errors occurs, the boot firmware does not process and returns to the command waiting state.
  * Memory status does not change before command reception.

When the processing above is successfully completed, boot firmware executes the acceptance analysis:

- If the device reset is not asserted after the Encrypted data write command execution, the boot firmware sends a “Command acceptance error”.
- When the above error occurs, the boot firmware does not process and returns to the command waiting state.
  * Memory status does not change before command reception.

When the communication mode is not asynchronous 2-wire communication, a response is returned when the processing above ends normally:

- If the communication mode is not asynchronous 2-wire communication, send “OK” and return to the command waiting state.
  * Memory status does not change before command reception.

In asynchronous 2-wire communication, parameter analysis is performed when the processing above is completed successfully:

- Sends “Parameter error” if the specified BRT (Baudrate) is greater than the RMB in the Signature request command.
- Sends “Parameter error” if the specified BRT (Baudrate) is not a supported baudrate value.
- When any of the above errors occurs, the boot firmware does not process and returns to the command waiting state.
  * Memory status does not change before command reception.

In asynchronous 2-wire communication, when the processing above is completed normally, the baud rate is set:

- After sending “OK”, set the baudrate and return to the command waiting state.
  * Memory status does not change before command reception.
  * After the boot firmware returned OK (started the baudrate setting), wait 1 ms before sending next command.

6.22.4 Status Information from the Microcontroller

(Listed in descending order of priority.)

Condition	STS	ST2	ADR
The received packet does not have ETX.	Packet error	FFFFFFFFh	FFFFFFFFh
Sum data in the packet received is different from the value calculated by the boot firmware.	Checksum error	FFFFFFFFh	FFFFFFFFh
Packet length in the received packet does not comply with the packet format.	Packet error	FFFFFFFFh	FFFFFFFFh
Packet length in the received packet does not comply with the specifications of this command.	Packet error	FFFFFFFFh	FFFFFFFFh
Device reset is not asserted after Encrypted data write command execution.	Command acceptance error	FFFFFFFFh	FFFFFFFFh
Received UART baudrate is greater than RMB.	Parameter error	FFFFFFFFh	FFFFFFFFh
Different from the baudrate value supported by the received UART baudrate.	Parameter error	FFFFFFFFh	FFFFFFFFh
Communication mode is different from UART.	OK	FFFFFFFFh	FFFFFFFFh
Started the baudrate setting.	OK	FFFFFFFFh	FFFFFFFFh

Table 22. Baudrate Setting Values

Intended Baudrate	ABCS	BGDM	CKS [1:0]	BRR [7:0]	MDDR [7:0]	Accuracy
9600bps	0	0	00b	A1h	FEh	-0.3%
115200 bps	0	0	00b	0Bh	E2h	-0.2%
500 Kbps	0	0	00b	01h	A3h	-0.5%
1.0 Mbps	0	0	00b	00h	A3h	-0.5%
1.5 Mbps	0	0	00b	00h	F5h	-0.3%
2.0 Mbps	1	0	00b	00h	A3h	-0.5%
4.0 Mbps	1	1	00b	00h	A3h	-0.5%
6.0 Mbps	1	1	00b	00h	F5h	-0.3%
Other	Unavailable	Unavailable	Unavailable	Unavailable	Unavailable	-

6.23 Erase Command

This command erases data in the specified area of the flash memory. The alignment of the target addresses follows the area information returned by the Area information request command. Erasures are executed in order from the start address to the end address by the erase access unit.

Erase processing at this time is not affected by the block protection settings (BPS, BPS_SEC).

This command requires adherence to conditions described in Command List.

6.23.1 Sequence Diagram

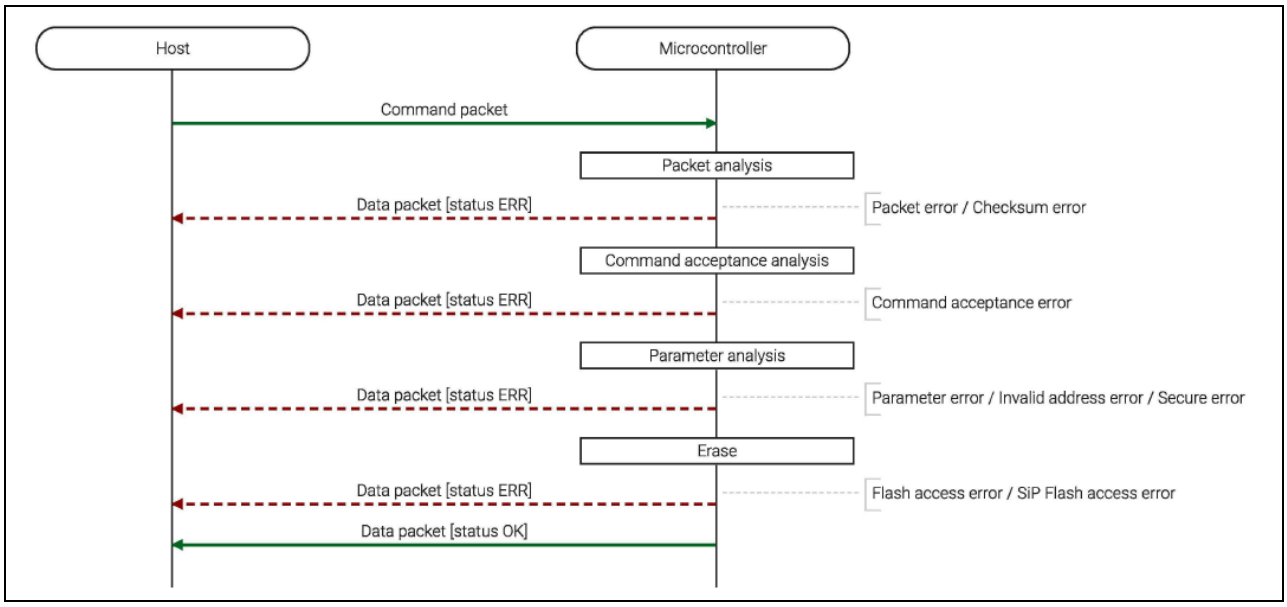


Figure 42. Erase Command Sequence Diagram

6.23.2 Packets

6.23.2.1 Command Packet

SOH	(1 byte)	01h
LNH	(1 byte)	00h
LNL	(1 byte)	09h
CMD	(1 byte)	12h (Erase command)
SAD	(4 bytes)	Start address. For example: 00004000h -> 00h, 00h, 40h, 00h
EAD	(4 bytes)	End address. For example: 003FFFFFh -> 00h, 3Fh, FFh, FFh
SUM	(1 byte)	Sum data
ETX	(1 byte)	03h

6.23.2.2 Data Packet [Status OK]

SOD	(1 byte)	81h
LNH	(1 byte)	00h
LNL	(1 byte)	0Ah
RES	(1 byte)	12h (OK)
STS	(1 byte)	00h (OK)
ST2	(4 bytes)	FFFFFFFFh (unused code)
ADR	(4 bytes)	FFFFFFFFh (unused code)
SUM	(1 byte)	Sum data
ETX	(1 byte)	03h

6.23.2.3 Data Packet [Status ERR]

SOD	(1 byte)	81h
LNH	(1 byte)	00h
LNL	(1 byte)	0Ah
RES	(1 byte)	92h (ERR)
STS	(1 byte)	Status code
ST2	(4 bytes)	Status details
ADR	(4 bytes)	Failure address
SUM	(1 byte)	Sum data
ETX	(1 byte)	03h

6.23.3 Processing Procedure

Boot firmware receives and analyzes a command packet:

- The boot firmware recognizes the start of the command packet by receiving SOH.
  If the boot firmware receives something other than SOH, it waits until it receives SOH.
- If ETX is not added to the received command packet, the boot firmware sends a “Packet error”.
- If the SUM of the received command packet is different from the sum value, the boot firmware sends a “Checksum error”.
- If the LNH and LNL of the received command packet are different from the values specified in the packet format, the boot firmware sends a “Packet error”.
- If the LNH and LNL of the received command packet are different from the values specified in each command, the boot firmware sends a “Packet error”.
- When any of the above errors occurs, the boot firmware does not process and returns to the command waiting state.
  * Memory status does not change before command reception.

When the processing above is successfully completed, boot firmware executes the acceptance analysis:

- If this command cannot be executed in the current DLM state, the boot firmware sends a “Command acceptance error”.
- If the device reset is not asserted after the Encrypted data write command execution, the boot firmware sends a “Command acceptance error”.
- When any of the above errors occurs, the boot firmware does not process and returns to the command waiting state.
  * Memory status does not change before command reception.

When the processing above is successfully completed, the boot firmware analyzes the command parameters:

- If the SAD is greater than EAD, the boot firmware sends a “Parameter error”.
- If the SAD or EAD is outside the range specified in the area information, the boot firmware sends a “Parameter error”.
- If SAD and EAD belong to different KOA, boot firmware sends a “Parameter error”.
- If the EAU for the specified area is 0, the boot firmware sends a “Parameter error”.
- If SAD and EAD are not specified in the EAU of the area, the boot firmware sends a “Parameter error”.
- If the area specified with SAD and EAD includes address that is inaccessible with current boundary setting, the boot firmware sends a “Invalid address error”.
- If the current Authentication level is AL1 and the specified range includes a secure area, the boot firmware sends a “Secure error”.
- If the current Authentication level is AL0, the boot firmware sends a “Secure error”.
- When any of the above errors occurs, the boot firmware does not process and returns to the command waiting state.
  * Memory status does not change before command reception.

When no error occurs, boot firmware executes the erase processing:

- If an error is returned from external flash memory access driver, the boot firmware sends a “Flash access error” and returns to the command wait state.
- If an error occurs during SiP Flash erasure, the boot firmware sends a “SiP Flash access error” and returns to the command wait state.
  * The value of the area after ADR (Failure address) of the memory is undefined.
- When the erase processing is normally finished, boot firmware returns “OK” and waits for the next command.
  * Specified area on memory are erased state.

6.23.4 Status Information from the Microcontroller

(Listed in descending order of priority.)

Condition	STS	ST2	ADR
The received packet does not have ETX.	Packet error	FFFFFFFFh	FFFFFFFFh
Sum data in the received packet is different from the value calculated by the boot firmware.	Checksum error	FFFFFFFFh	FFFFFFFFh
Packet length in the received packet does not comply with the packet format.	Packet error	FFFFFFFFh	FFFFFFFFh
Packet length in the received packet does not comply with the specifications of this command.	Packet error	FFFFFFFFh	FFFFFFFFh
Executing this command is unavailable in the current DLM state.	Command acceptance error	FFFFFFFFh	FFFFFFFFh
Device reset is not asserted after Encrypted data write command execution.	Command acceptance error	FFFFFFFFh	FFFFFFFFh
Start address is bigger than End address.	Parameter error	FFFFFFFFh	FFFFFFFFh
Start address or End address is outside the scope of user area specified in area information.	Parameter error	FFFFFFFFh	FFFFFFFFh
Start address and End address belongs to different types of area.	Parameter error	FFFFFFFFh	FFFFFFFFh
The access unit “EAU” of the specified area is 0.	Parameter error	FFFFFFFFh	FFFFFFFFh
Start address or End address doesn't comply with EAU of the area.	Parameter error	FFFFFFFFh	FFFFFFFFh
The area from Start address to End address contains addresses that are inaccessible with the current boundary settings.	Invalid address error	FFFFFFFFh	FFFFFFFFh
Current Authentication level is AL1, and designated erasure range includes Secure region.	Secure error	FFFFFFFFh	FFFFFFFFh
Current Authentication level is AL0.	Secure error	FFFFFFFFh	FFFFFFFFh
An error occurred while erasing SiP Flash.	SIP Flash access error	SIP Flash status	Failure address
An error occurred in the external flash memory access driver.	Flash access error	FFFFFFFFh	FFFFFFFFh
Successful completion.	OK	FFFFFFFFh	FFFFFFFFh

6.23.5 Precautions

(1) When accessing the external flash area, the driver function for access is called, so send the driver code with the “External flash memory setting command” in advance. In this command, “EraseChip driver” is called when the entire area of External flash area 0 is specified. Otherwise, the “EraseSector driver” is called every time a sector is erased.

Also, access to addresses to which external flash memory is not allocated is not guaranteed.

6.24 Write Command

This command receives data from the host and writes those data to the specified area. The alignment of the target address follows the area information returned by the Area information request command. Writings are executed in order from the start address to the end address by the write access unit.

Write processing at this time is not affected by the block protection settings (BPS, BPS_SEC).

This command requires adherence to conditions described in Command List.

6.24.1 Sequence Diagram

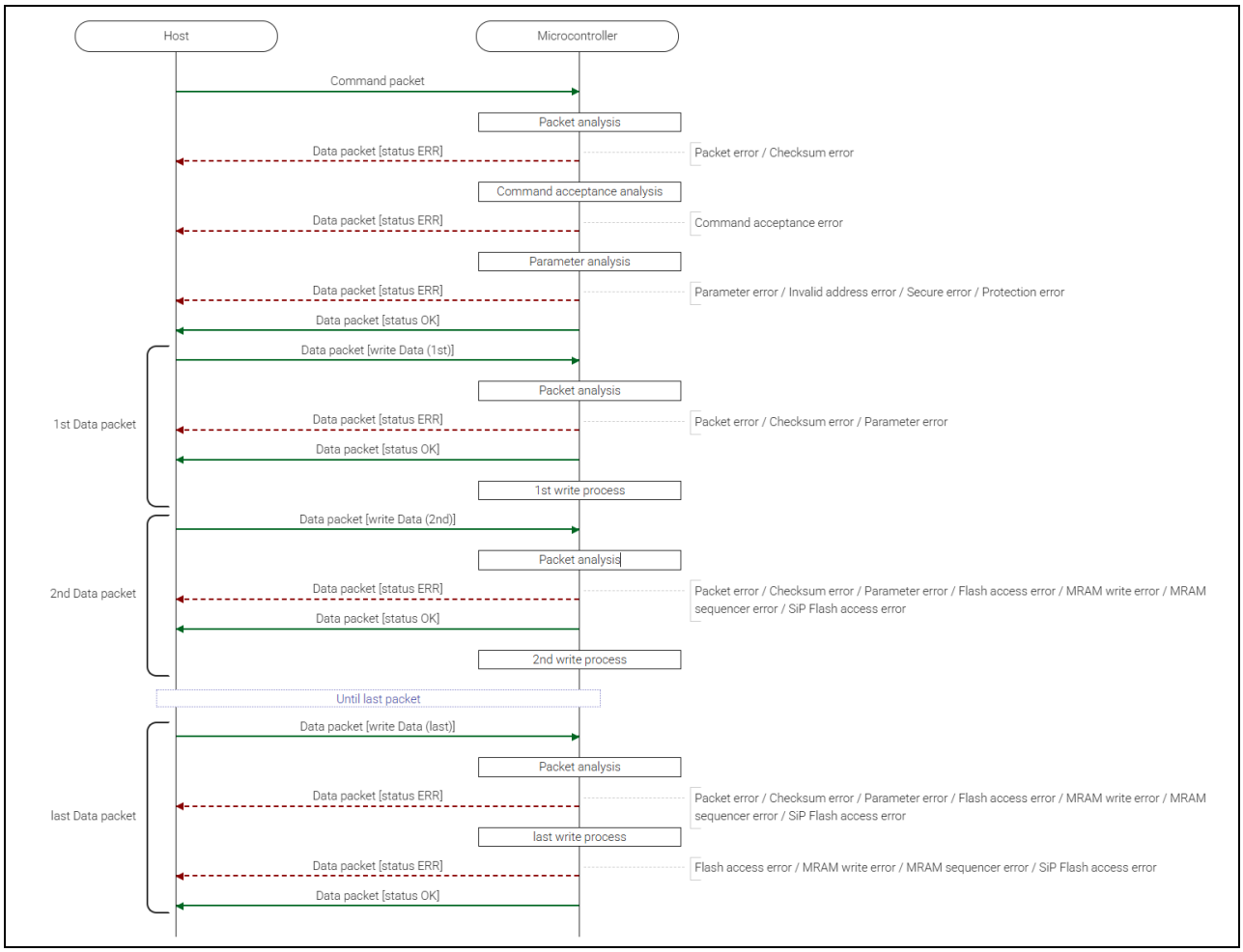


Figure 43. Write Command Sequence Diagram

6.24.2 Packets

6.24.2.1 Command Packet

SOH	(1 byte)	01h
LNH	(1 byte)	00h
LNL	(1 byte)	09h
CMD	(1 byte)	13h (Write command)
SAD	(4 bytes)	Start address. For example: 00004000h -> 00h, 00h, 40h, 00h
EAD	(4 bytes)	End address. For example: 003FFFFFh -> 00h, 3Fh, FFh, FFh
SUM	(1 byte)	Sum data
ETX	(1 byte)	03h

6.24.2.2 Data Packet [Write Data]

SOD	(1 byte)	81h
LNH	(1 byte)	N + 1 (Higher 1 byte)
LNL	(1 byte)	N + 1 (Lower 1 byte)
RES	(1 byte)	13h (OK)
DAT	(N bytes)	Write data
SUM	(1 byte)	Sum data
ETX	(1 byte)	03h

N = 1–1024

*) N must be multiple of 4 when writing to external flash area.

6.24.2.3 Data Packet [Status OK]

SOD	(1 byte)	81h
LNH	(1 byte)	00h
LNL	(1 byte)	0Ah
RES	(1 byte)	13h (OK)
STS	(1 byte)	00h (OK)
ST2	(4 bytes)	FFFFFFFFh (unused code)
ADR	(4 bytes)	FFFFFFFFh (unused code)
SUM	(1 byte)	Sum data
ETX	(1 byte)	03h

6.24.2.4 Data Packet [Status ERR]

SOD	(1 byte)	81h
LNH	(1 byte)	00h
LNL	(1 byte)	0Ah
RES	(1 byte)	93h (ERR)
STS	(1 byte)	Status code
ST2	(4 bytes)	Status details
ADR	(4 bytes)	Failure address
SUM	(1 byte)	Sum data
ETX	(1 byte)	03h

6.24.3 Processing Procedure

Boot firmware receives and analyzes a command packet:

- The boot firmware recognizes the start of the command packet by receiving SOH.
  If the boot firmware receives something other than SOH, it waits until it receives SOH.
- If ETX is not added to the received command packet, the boot firmware sends a “Packet error”.
- If the SUM of the received command packet is different from the sum value, the boot firmware sends a “Checksum error”.
- If the LNH and LNL of the received command packet are different from the values specified in the packet format, the boot firmware sends a “Packet error”.
- If the LNH and LNL of the received command packet are different from the values specified in each command, the boot firmware sends a “Packet error”.
- When any of the above errors occurs, the boot firmware does not process and returns to the command waiting state.
  * Memory status does not change before command reception.

When the processing above is successfully completed, boot firmware executes the acceptance analysis:

- If this command cannot be executed in the current DLM state, the boot firmware sends a “Command acceptance error”.
- If the device reset is not asserted after the Encrypted data write command execution, the boot firmware sends a “Command acceptance error”.
- When any of the above errors occurs, the boot firmware does not process and returns to the command waiting state.
  * Memory status does not change before command reception.

When the processing above is successfully completed, the boot firmware analyzes the command parameters:

- If the SAD is greater than EAD, the boot firmware sends a “Parameter error”.
- If the SAD or EAD is outside the range specified in the area information, the boot firmware sends a “Parameter error”.
- If SAD and EAD belong to different KOA, boot firmware sends a “Parameter error”.
- If the WAU for the specified area is 0, the boot firmware sends a “Parameter error”.
- If SAD and EAD are not specified in the WAU of the area, the boot firmware sends a “Parameter error”.
- If the area specified with SAD and EAD includes address that is inaccessible with the current boundary setting, the boot firmware sends a “Invalid address error”.
- If the current Authentication level is AL1 and the specified range includes a secure area, the boot firmware sends a “Secure error”.
- If the current Authentication level is AL0, the boot firmware sends a “Secure error”.
- When designated writing range includes PBPS block, “Protection error” is returned.
- When designated writing range includes area that the lock bit is set, “Protection error” is returned.
- When any of the above errors occurs, the boot firmware does not process and returns to the command waiting state.
  * Memory status does not change before command reception.
- If the above error does not occur, the boot firmware sends “OK”.

When the processing above is successfully completed, boot firmware receives and analyzes a data packet:

- The boot firmware recognizes the start of the data packet by receiving SOD.
  If the boot firmware receives something other than SOD, it waits until it receives SOD.
- If ETX is not added to the received data packet, the boot firmware sends a “Packet error”.
- If the SUM of the received command packet is different from the sum value, the boot firmware sends a “Checksum error”.
- If the received data packet's LNH and LNL are different from the values specified in the packet format, the boot firmware sends a “Packet error”.
- When RES in the received data packet is different from defined values by each command, “Packet error” is returned.
- When total length of the received data of data packets exceeds the size of specified area, “Parameter error” is returned.
- If size of the write data is not specified in the WAU of the area, the boot firmware sends a “Parameter error”.
- When any of the above errors occurs, the boot firmware does not process and returns to the command waiting state.
  * Memory status does not change before command reception.

When the received data packet is not the last write data, boot firmware returns “OK” and executes the write processing:

- Boot firmware returns “OK” and executes the write processing.
- When the write processing is abnormally finished, boot firmware receives the next data packet, returns “MRAM write error” or “MRAM sequencer error” and waits for the next command.
  * WAU size from failure address (ADR) of memory area are undefined.
- If an error is returned from external flash memory access driver, the boot firmware sends a “Flash access error” and returns to the command wait state.
- When the write processing of SiP Flash is abnormally finished, boot firmware receives the next data packet, returns “SiP Flash access error” and waits for the next command.
  * WAU size from failure address (ADR) of memory area is undefined
- When the write processing is normally finished, boot firmware receives the next data packet.

When the received data packet is the last write data, boot firmware executes the write processing and returns status:

- Boot firmware executes the write processing.
- When the write processing is abnormally finished, boot firmware returns “MRAM write error” or “MRAM sequencer error” and waits for the next command.
  * WAU size from failure address (ADR) of memory area are undefined.
- If an error is returned from external flash memory access driver, the boot firmware sends a “Flash access error” and returns to the command wait state.
- If an error occurs while SiP Flash writing, the boot firmware sends a “SiP Flash access error” and returns to the command wait state.
  * WAU size from failure address (ADR) of memory area is undefined.
- When the write processing is normally finished, boot firmware returns “OK” and waits for the next command.
  * Sent data are written to the specified area on memory.

6.24.4 Status Information from the Microcontroller

(Listed in descending order of priority.)

Condition	STS	ST2	ADR
The received packet does not have ETX.	Packet error	FFFFFFFFh	FFFFFFFFh
Sum data in the received packet is different from the value calculated by the boot firmware.	Checksum error	FFFFFFFFh	FFFFFFFFh
Packet length in the received packet does not comply with the packet format.	Packet error	FFFFFFFFh	FFFFFFFFh
Packet length in the received packet does not comply with the specifications of this command.	Packet error	FFFFFFFFh	FFFFFFFFh
Executing this command is unavailable in the current DLM state.	Command acceptance error	FFFFFFFFh	FFFFFFFFh
Device reset is not asserted after Encrypted data write command execution.	Command acceptance error	FFFFFFFFh	FFFFFFFFh
Start address is bigger than End address.	Parameter error	FFFFFFFFh	FFFFFFFFh
Start address or End address is outside the scope of accessible area specified in area information.	Parameter error	FFFFFFFFh	FFFFFFFFh
Start address and End address belong to different types of area.	Parameter error	FFFFFFFFh	FFFFFFFFh
The access unit “WAU” of the specified area is 0.	Parameter error	FFFFFFFFh	FFFFFFFFh
Start address or End address does not comply with WAU of the area.	Parameter error	FFFFFFFFh	FFFFFFFFh
The area from Start address to End address contains addresses that are inaccessible with the current boundary settings.	Invalid address error	FFFFFFFFh	FFFFFFFFh
Current Authentication level is AL1, and designated writing range includes Secure region.	Secure error	FFFFFFFFh	FFFFFFFFh
Current Authentication level is AL0.	Secure error	FFFFFFFFh	FFFFFFFFh
Designated writing range includes permanent protected blocks.	Protection error	FFFFFFFFh	FFFFFFFFh
Designated writing range includes OFSPS protected area.	Protection error	FFFFFFFFh	FFFFFFFFh
The response code of the received data packet is different from the value specified by this command.	Packet error	FFFFFFFFh	FFFFFFFFh
The total length of received data of data packets exceeds the specified end address.	Parameter error	FFFFFFFFh	FFFFFFFFh
The data size of the data packet does not comply with writing unit of the area.	Parameter error	FFFFFFFFh	FFFFFFFFh
An error detected when MRAM programming which does not use MACI command	MRAM write error	FFFFFFFFh	Failure address
MACI detected an error after the command execution.	MRAM sequencer error	MRAM status	Failure address
An error occurred while programming SiP Flash.	SiP Flash access error	SiP Flash status	Failure address
An error occurred in the External flash memory access driver.	Flash access error	FFFFFFFFh	FFFFFFFFh
Successful completion.	OK	FFFFFFFFh	FFFFFFFFh

6.24.5 Precautions

(1) If permanent block protection in the Config area is set, the protected area cannot be rewritten. Therefore, rewrite the protected area before setting the permanent block protection.

(2) When accessing the external flash area, the driver function for access is called, so send the driver code with the “External flash memory setting command” in advance. This command calls the “Program Data driver”.

Also, access to addresses to which external flash memory is not allocated is not guaranteed.

(3) Config OTP area cannot be written back to 1 after 0 is written.

For Conflg OTP area which has ECG, ECG bits cannot be written back from 0 to 1. Therefore, rewriting is not possible even when the data bits are to be changed from 1 to 0. However, for the following cases rewriting is possible as an exception:

- Rewriting the same value

- The data already written is all“F”

6.25 Read Command

This command reads data from a specified area and sends that data to the host. The alignment of the target addresses follow the area information returned by the area information request command. Readings are executed in order from the start address to the end address by the read access unit.

This command requires adherence to conditions described in Command List.

6.25.1 Sequence Diagram

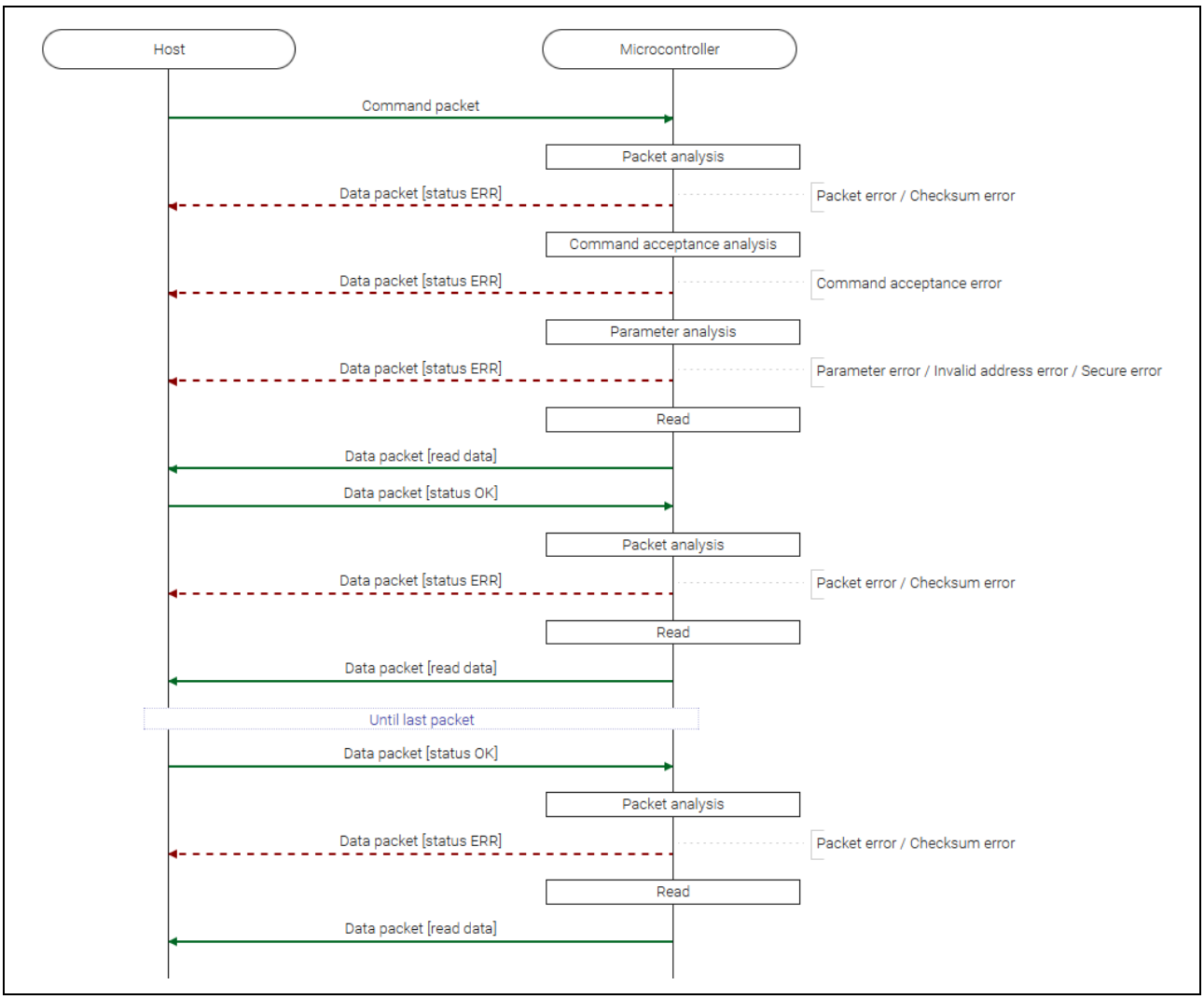


Figure 44. Read Command Sequence Diagram

6.25.2

6.25.2 Packets

6.25.2.1 Command Packet

SOH	(1 byte)	01h
LNH	(1 byte)	00h
LNL	(1 byte)	09h
CMD	(1 byte)	15h (Read command)
SAD	(4 bytes)	Start address For example: 00004000h -> 00h, 00h, 40h, 00h
EAD	(4 bytes)	End address For example: 003FFFFFh -> 00h, 3Fh, FFh, FFh
SUM	(1 byte)	Sum data
ETX	(1 byte)	03h

6.25.2.2 Data Packet [Status OK]

SOD	(1 byte)	81h
LNH	(1 byte)	00h
LNL	(1 byte)	0Ah
RES	(1 byte)	15h (OK)
STS	(1 byte)	00h (OK)
ST2	(4 bytes)	FFFFFFFFh (unused code)
ADR	(4 bytes)	FFFFFFFFh (unused code)
SUM	(1 byte)	Sum data
ETX	(1 byte)	03h

6.25.2.3 Data Packet [Status ERR]

SOD	(1 byte)	81h
LNH	(1 byte)	00h
LNL	(1 byte)	0Ah
RES	(1 byte)	95h (ERR)
STS	(1 byte)	Status code
ST2	(4 bytes)	Status details
ADR	(4 bytes)	Failure address
SUM	(1 byte)	Sum data
ETX	(1 byte)	03h

6.25.2.4 Data Packet [Read Data]

SOD	(1 byte)	81h
LNH	(1 byte)	N + 1 (Higher 1 byte)
LNL	(1 byte)	N + 1 (Lower 1 byte)
RES	(1 byte)	15h (OK)
DAT	(N byte)	Read data
SUM	(1 byte)	Sum data
ETX	(1 byte)	03h

6.25.3 Processing Procedure

Boot firmware receives and analyzes a command packet:

- The boot firmware recognizes the start of the command packet by receiving SOH.
  If the boot firmware receives something other than SOH, it waits until it receives SOH.
- If ETX is not added to the received command packet, the boot firmware sends a “Packet error”.
- If the SUM of the received command packet is different from the sum value, the boot firmware sends a “Checksum error”.
- If the LNH and LNL of the received command packet are different from the values specified in the packet format, the boot firmware sends a “Packet error”.
- If the LNH and LNL of the received command packet are different from the values specified in each command, the boot firmware sends a “Packet error”.
- When any of the above errors occurs, the boot firmware does not process and returns to the command waiting state.
  * Memory status does not change before command reception.

When the processing above is successfully completed, boot firmware executes the acceptance analysis.

- If this command cannot be executed in the current OLM state, the boot firmware sends a “Command acceptance error”.
- If device reset is not asserted after the Encrypted data write command execution, the boot firmware sends a “Command acceptance error”.
- When the above error occurs, the boot firmware does not process and returns to the command waiting state.
  * Memory status does not change before command reception.

When the processing above is successfully completed, boot firmware analyzes the command parameters.

- If the SAD is greater than EAD, the boot firmware sends a “Parameter error”.
- If the SAD or EAD is outside the range specified in the area information, the boot firmware sends a “Parameter error”.
- If SAD and EAD belong to different KOA, the boot firmware sends a “Parameter error”.
- If the RAU for the specified area is 0, the boot firmware sends a “Parameter error”.
- If the area specified with SAD and EAD includes an address that is inaccessible with the current boundary setting, the boot firmware sends a “Invalid address error”.
- When the above error occurs, the boot firmware does not process and returns to the command waiting state.
  * Memory status does not change before command reception.

When the processing above is successfully completed, boot firmware performs a secure analysis.

- If the current Authentication level is AL1 and the specified range includes a secure area, the boot firmware sends a “Secure error”.
- If the current Authentication level is AL0, the boot firmware sends a “Secure error”.

When no error occurs, boot firmware executes the read processing.

- Boot firmware returns the read data (packet-length: Max.1024 bytes).
- When all the data are sent, the boot firmware waits for the next command.
  * Memory status does not change before command reception

If data transmission for the specified size is not completed, the boot firmware receives the data packet and performs packet analysis.

- Boot firmware detects the beginning of a data packet by receiving SOD.
  When boot firmware receives other data than SOD, it discards the data and waits for the next data until SOD is sent.
- When the received data packet does not have ETX, “Packet error” is returned.
- When SUM in the received data packet is different from the value calculated by boot firmware, “Checksum error” is returned.
- When LNH and LNL in the received data packet do not comply with the packet format, “Packet error” is returned.
- When RES in the received data packet is different from defined values, “Packet error” is returned.
- When LNH and LNL in the received data packet do not comply format with this command, “Packet error” is returned.
- When the above error occurs, the boot firmware does not process and returns to the command waiting state.
  * Memory status does not change before command reception
- If the above error does not occur, the boot firmware continues to read and send data.

6.25.4 Status Information from the Microcontroller

(Listed in descending order of priority.)

Condition	STS	ST2	ADR
The received packet does not have ETX.	Packet error	FFFFFFFFh	FFFFFFFFh
Sum data in the packet received is different from the value calculated by the boot firmware.	Checksum error	FFFFFFFFh	FFFFFFFFh
Packet length in the received packet does not comply with the packet format.	Packet error	FFFFFFFFh	FFFFFFFFh
Packet length in the received packet does not comply with the specifications of this command.	Packet error	FFFFFFFFh	FFFFFFFFh
Executing this command is unavailable in current DLM state.	Command acceptance error	FFFFFFFFh	FFFFFFFFh
Device reset is not asserted after Encrypted data write command execution.	Command acceptance error	FFFFFFFFh	FFFFFFFFh
Start address is bigger than End address.	Parameter error	FFFFFFFFh	FFFFFFFFh
Start address or End address is outside the scope of accessible area specified in area information.	Parameter error	FFFFFFFFh	FFFFFFFFh
Start address and End address belong to different type of area.	Parameter error	FFFFFFFFh	FFFFFFFFh
The access unit “RAU” of the specified area is 0.	Parameter error	FFFFFFFFh	FFFFFFFFh
Start address or End address does not comply with RAU of the area.	Parameter error		
The area from Start address to End address contains addresses that are inaccessible with the current boundary settings.	Invalid address error	FFFFFFFFh	FFFFFFFFh
Current Authentication level is AL1, designated reading range is User area, Config OTP area or SiP Flash area, and includes Secure region.	Secure error	FFFFFFFFh	FFFFFFFFh
Current Authentication level is AL0.	Secure error	FFFFFFFFh	FFFFFFFFh
The response code of the received data packet is different from the value specified by this Packet error command.	Packet error	FFFFFFFFh	FFFFFFFFh

6.25.5 Precautions

(1) External flash memory setting command must be executed in advance to access the External flash area. Boot firmware does not call any external flash memory access drivers in this command. When External flash area is specified as SAD/EAD, boot firmware issues read access to the specified address on assumption that Octal SPI has been initialized to Memory mapping mode. Also, access to addresses to which external flash memory is not allocated is not guaranteed.

6.26 CRC Command

This command calculates CRC data from a specified area and sends it to host. The alignment of the target addresses follows the area information returned by the Area information request command. Calculations are executed in order from the start address to the end address by the CRC access unit. This command requires adherence to conditions described in Command List.

Boot firmware uses the following CRC method.

Name	CRC-32-IEEE-802.3
Default value	FFFFFFFFh
Shift direction	Left shift
Polynomial representations	(MSB first) 04C11 DB7h

6.26.1 Sequence Diagram

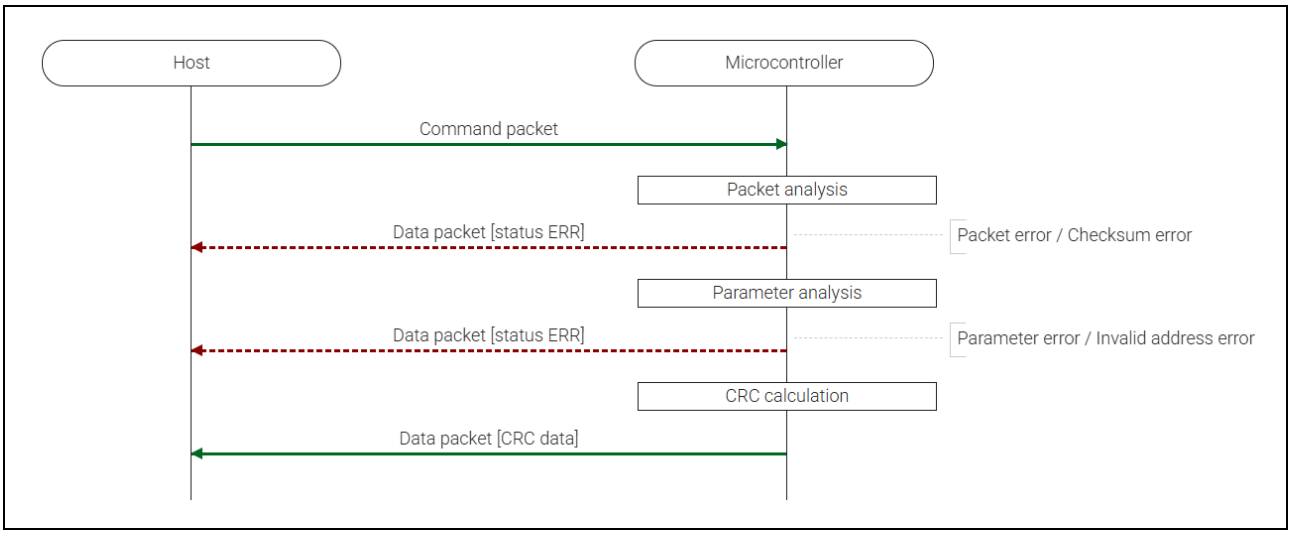


Figure 45. CRC Command Diagram

6.26.2 Packets

6.26.2.1 Command Packet

SOH	(1 byte)	01h
LNH	(1 byte)	00h
LNL	(1 byte)	09h
CMD	(1 byte)	18h (CRC command)
SAD	(4 bytes)	Start address
EAD	(4 bytes)	End address
SUM	(1 byte)	Sum data
ETX	(1 byte)	03h

6.26.2.2 Data Packet [CRC Data]

SOD	(1 byte)	81h
LNH	(1 byte)	00h
LNL	(1 byte)	05h
RES	(1 byte)	18h (OK)
CRC	(4 bytes)	CRC data (result of calculation) e.g.) 01234567h -> 01 h, 23h, 45h, 67h
SUM	(1 byte)	Sum data
ETX	(1 byte)	03h

6.26.2.3 Data Packet [Status ERR]

SOD	(1 byte)	81h
LNH	(1 byte)	00h
LNL	(1 byte)	0Ah
RES	(1 byte)	98h (ERR)
STS	(1 byte)	Status code
ST2	(4 bytes)	Status details
ADR	(4 bytes)	Failure address
SUM	(1 byte)	Sum data
ETX	(1 byte)	03h

6.26.3 Processing Procedure

Boot firmware receives and analyzes a command packet:

- The boot firmware recognizes the start of the command packet by receiving SOH.
  If the boot firmware receives something other than SOH, it waits until it receives SOH.
- If ETX is not added to the received command packet, the boot firmware sends a “Packet error”.
- If the SUM of the received command packet is different from the sum value, the boot firmware sends a “Checksum error”.
- If the LNH and LNL of the received command packet are different from the values specified in the packet format, the boot firmware sends a “Packet error”.
- If the LNH and LNL of the received command packet are different from the values specified in each command, the boot firmware sends a “Packet error”.
- When any of the above errors occurs, the boot firmware does not process and returns to the command waiting state.
  * Memory status do not change before command reception.

When the processing above is successfully completed, the boot firmware analyzes the command parameters.

- If SAD is greater than EAD, the boot firmware sends a “Parameter error”.
- If SAD or EAD is outside the range specified in the area information, the boot firmware sends a “Parameter error”.
- If SAD and EAD belong to different KOA, the boot firmware sends a “Parameter error”.
- If the CAU for the specified area is 0, the boot firmware sends a “Parameter error”.
- If SAD and EAD are not specified in the CAU of the area, the boot firmware sends a “Parameter error”.
- If the area specified with SAD and EAD includes address that is inaccessible with the current boundary setting, the boot firmware sends a “Invalid address error”.
- When the above error occurs, the boot firmware does not process and returns to the command waiting state.
  * Memory status does not change before command reception

When the processing above is successfully completed, boot firmware executes CRC calculation.

- After the CRC calculation, boot firmware returns “CRC data” and waits next command.
  * Memory status does not change before command reception

6.26.4 Status Information from the Microcontroller

(Listed in descending order of priority.)

Condition	STS	ST2	ADR
The received packet does not have ETX.	Packet error	FFFFFFFFh	FFFFFFFFh
Sum data in the packet received is different from the value calculated by the boot firmware.	Checksum error	FFFFFFFFh	FFFFFFFFh
Packet length in the received packet does not comply with the packet format.	Packet error	FFFFFFFFh	FFFFFFFFh
Packet length in the received packet does not comply with the specifications of this command.	Packet error	FFFFFFFFh	FFFFFFFFh
Start address is bigger than End address.	Parameter error	FFFFFFFFh	FFFFFFFFh
Start address or End address is outside the scope of accessible area specified in area information.	Parameter error	FFFFFFFFh	FFFFFFFFh
Start address and End address belongs to different type of the area.	Parameter error	FFFFFFFFh	FFFFFFFFh
The access unit “CAU” of the specified area is 0.	Parameter error	FFFFFFFFh	FFFFFFFFh
Start address or End address doesn't comply with CAU of the area.	Parameter error		
The area from Start address to End address contains addresses that are inaccessible with the current boundary settings.	Invalid address error	FFFFFFFFh	FFFFFFFFh

6.26.5 Precautions

(1) External flash memory setting command must be executed in advance to access the External flash area. Boot firmware does not call any external flash memory access drivers in this command. When External flash area is specified as SAD/EAD, boot firmware issues read access to the specified address on assumption that Octal SPI has been initialized to Memory mapping mode. Also, access to addresses to which external flash memory is not allocated is not guaranteed.

6.27 OEM Root Public Key Setting Command

This command sets the OEM root public key encrypted hash value (OEM_ROOLPK) to device.

This command requires adherence to conditions described in Command List.

6.27.1 Sequence Diagram

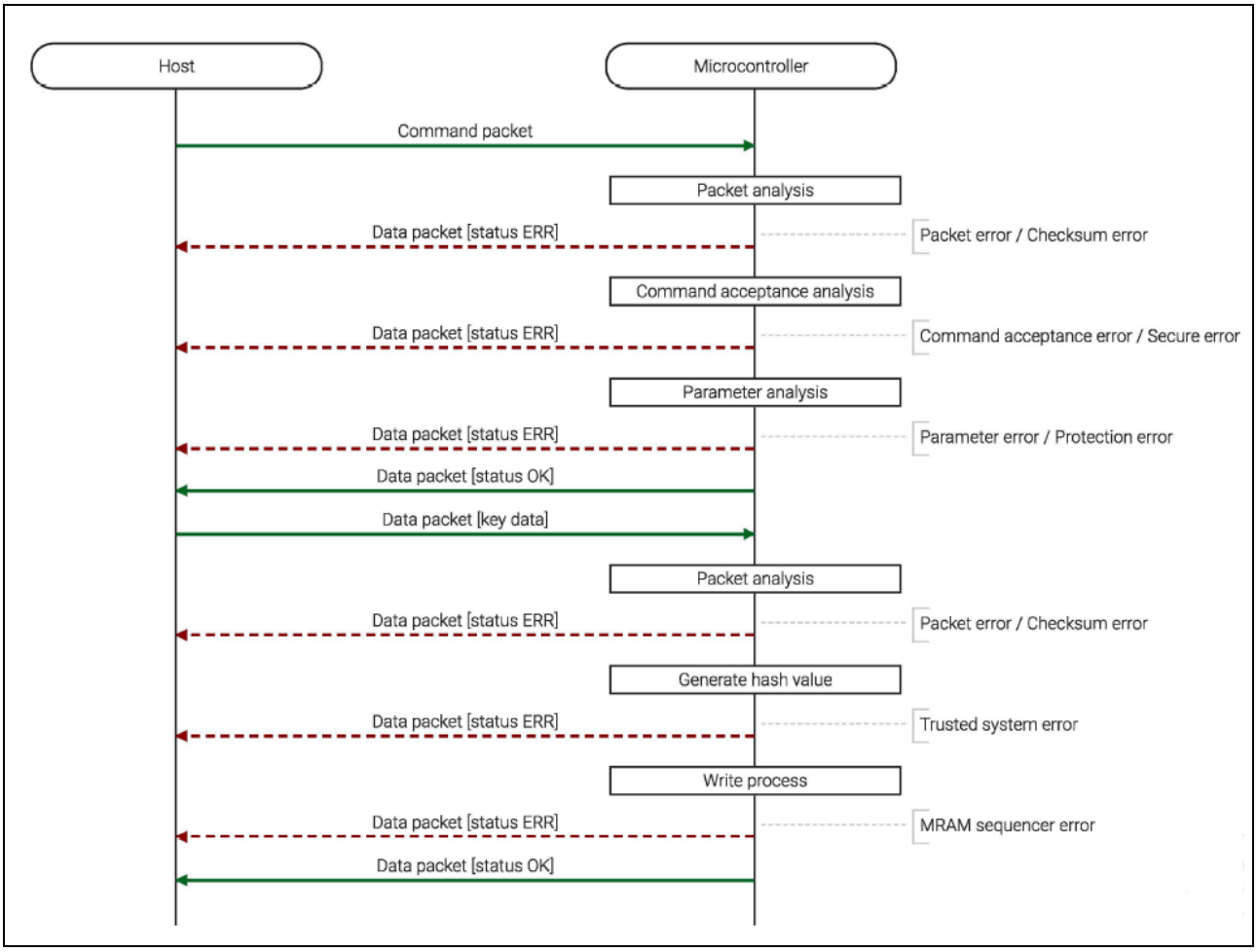


Figure 46. OEM Root Public Key Setting Command Diagram

6.27.2 Packets

6.27.2.1 Command Packet

SOH	(1 byte)	01h
LNH	(1 byte)	00h
LNL	(1 byte)	03h
CMD	(1 byte)	2Eh (OEM root public key setting command)
KID	(1 byte)	Public-key ID 00h: Public-key 0 01h: Public-key 1 02h . Public-key 2 03h : Public-key 3
RSV	(1 byte)	00h (unused code)
SUM	(1 byte)	Sum data
ETX	(1 byte)	03h

6.27.2.2 Data Packet [Key Data]

<table>
<tr><td>SOD</td><td>(1 byte)</td><td>81h</td></tr>
<tr><td>LNH</td><td>(1 byte)</td><td>00h</td></tr>
<tr><td>LNL</td><td>(1 byte)</td><td>85h</td></tr>
<tr><td>RES</td><td>(1 byte)</td><td>2Eh (OK)</td></tr>
<tr><td>SKR</td><td>(4 bytes)</td><td>Shared key ring number
For example: 01234567h -> 01h, 23h, 45h, 67h</td></tr>
<tr><td>ESKY</td><td>(32 bytes)</td><td>Wrapped install key (W-UFPK)
For example: 07 234567 89AB ... 2233_ 44556677h -> 07h, 23h, 45h, ..., 55h, 66h, 77h</td></tr>
<tr><td>IVEC</td><td>(16 bytes)</td><td>Initialization Vector
For example: 01234567 89AB ... 2233_44556677h -> 01h, 23h, 45h, ..., 55h, 66h, 77h</td></tr>
<tr><td>RPK</td><td>(80 bytes)</td><td>OEM root public key (OEM_ROOT_PK I MAC)

Encrypted key (0 - 63 bytes) + MAC (64 - 79 bytes)
For example: If install data is as follows, the Host sends RPK in order shown in the lower table.

Install data:
<table>
<tr><td colspan="8">Encrypted key</td></tr>
<tr><td>00</td><td>01</td><td>02</td><td>03</td><td>04</td><td>05</td><td>06</td><td>07</td></tr>
<tr><td>08</td><td>09</td><td>0A</td><td>0B</td><td>0C</td><td>0D</td><td>0E</td><td>0F</td></tr>
<tr><td>10</td><td>11</td><td>12</td><td>13</td><td>14</td><td>15</td><td>16</td><td>17</td></tr>
<tr><td>18</td><td>19</td><td>1A</td><td>1B</td><td>1C</td><td>1D</td><td>1E</td><td>1F</td></tr>
<tr><td>20</td><td>21</td><td>22</td><td>23</td><td>24</td><td>25</td><td>26</td><td>27</td></tr>
<tr><td>28</td><td>29</td><td>2A</td><td>2B</td><td>2C</td><td>2D</td><td>2E</td><td>2F</td></tr>
<tr><td>30</td><td>31</td><td>32</td><td>33</td><td>34</td><td>35</td><td>36</td><td>37</td></tr>
<tr><td>38</td><td>39</td><td>3A</td><td>3B</td><td>3C</td><td>3D</td><td>3E</td><td>3F</td></tr>
<tr><td colspan="8">MAC</td></tr>
<tr><td>40</td><td>41</td><td>42</td><td>43</td><td>44</td><td>44</td><td>46</td><td>47</td></tr>
<tr><td>48</td><td>49</td><td>4A</td><td>4B</td><td>4C</td><td>4D</td><td>4E</td><td>4F</td></tr>
</table>
Order of sending RPK:
<table>
<tr><td>1st</td><td>2nd</td><td>3rd</td><td>4th</td><td>5th</td><td>6th</td><td>7th</td><td>8th</td></tr>
<tr><td>00</td><td>01</td><td>02</td><td>03</td><td>04</td><td>05</td><td>06</td><td>07</td></tr>
<tr><td>9th</td><td>10th</td><td>11th</td><td>12th</td><td>13th</td><td>14th</td><td>15th</td><td>16th</td></tr>
<tr><td>08</td><td>09</td><td>0A</td><td>0B</td><td>0C</td><td>0D</td><td>0E</td><td>0F</td></tr>
<tr><td>17th</td><td>18th</td><td>19th</td><td>20th</td><td>21st</td><td>22nd</td><td>23rd</td><td>24th</td></tr>
<tr><td>10</td><td>11</td><td>12</td><td>13</td><td>14</td><td>15</td><td>16</td><td>17</td></tr>
<tr><td>25th</td><td>26th</td><td>27th</td><td>28th</td><td>29th</td><td>30th</td><td>31st</td><td>32nd</td></tr>
<tr><td>18</td><td>19</td><td>1A</td><td>1B</td><td>1C</td><td>1D</td><td>1E</td><td>1F</td></tr>
<tr><td>33rd</td><td>34th</td><td>35th</td><td>36th</td><td>37th</td><td>38th</td><td>39th</td><td>40th</td></tr>
<tr><td>20</td><td>21</td><td>22</td><td>23</td><td>24</td><td>25</td><td>26</td><td>27</td></tr>
<tr><td>41st</td><td>42nd</td><td>43rd</td><td>44th</td><td>45th</td><td>46th</td><td>47th</td><td>48th</td></tr>
<tr><td>28</td><td>29</td><td>2A</td><td>2B</td><td>2C</td><td>2D</td><td>2E</td><td>2F</td></tr>
<tr><td>49th</td><td>50th</td><td>51st</td><td>52nd</td><td>53rd</td><td>54th</td><td>55th</td><td>56th</td></tr>
<tr><td>30</td><td>31</td><td>32</td><td>33</td><td>34</td><td>35</td><td>36</td><td>37</td></tr>
<tr><td>57th</td><td>58th</td><td>59th</td><td>60th</td><td>61st</td><td>62nd</td><td>63rd</td><td>64th</td></tr>
<tr><td>38</td><td>39</td><td>3A</td><td>3B</td><td>3C</td><td>3D</td><td>3E</td><td>3F</td></tr>
<tr><td>65th</td><td>66th</td><td>67th</td><td>68th</td><td>69th</td><td>70th</td><td>71st</td><td>72nd</td></tr>
<tr><td>40</td><td>41</td><td>42</td><td>43</td><td>44</td><td>44</td><td>46</td><td>47</td></tr>
<tr><td>73rd</td><td>74th</td><td>75th</td><td>76th</td><td>77th</td><td>78th</td><td>79th</td><td>80th</td></tr>
<tr><td>48</td><td>49</td><td>4A</td><td>4B</td><td>4C</td><td>4D</td><td>4E</td><td>4F</td></tr>
</table>
</td></tr>
</table>

SUM	(1 byte)	Sum data
ETX	(1 byte)	03h

6.27.2.3 Data Packet [Status OK]

SOD	(1 byte)	81h
LNH	(1 byte)	00h
LNL	(1 byte)	0Ah
RES	(1 byte)	2Eh (OK)
STS	(1 bytes)	00h (OK)
ST2	(4 bytes)	FFFFFFFFh (unused code)
ADR	(4 bytes)	FFFFFFFFh (unused code)
SUM	(1 byte)	Sum data
ETX	(1 byte)	03h

6.27.2.4 Data Packet [Status ERR]

SOD	(1 byte)	81h
LNH	(1 byte)	00h
LNL	(1 byte)	0Ah
RES	(1 byte)	AEh (ERR)
STS	(1 byte)	Status code
ST2	(4 bytes)	Status details
ADR	(4 bytes)	Failure address
SUM	(1 byte)	Sum data
ETX	(1 byte)	03h

6.27.3 Processing Procedure

Boot firmware receives and analyzes a command packet:

- The boot firmware recognizes the start of the command packet by receiving SOH.
  If the boot firmware receives something other than SOH, it waits until it receives SOH.
- If ETX is not added to the received command packet, the boot firmware sends a “Packet error”.
- If the SUM of the received command packet is different from the sum value, the boot firmware sends a “Checksum error”.
- If the LNH and LNL of the received command packet are different from the values specified in the packet format, the boot firmware sends a “Packet error”.
- If the LNH and LNL of the received command packet are different from the values specified in each command, the boot firmware sends a “Packet error”.
- When any of the above errors occurs, the boot firmware does not process and returns to the command waiting state.
  * Memory status do not change before command reception.

When the processing above is successfully completed, boot firmware executes the acceptance analysis:

- If this command cannot be executed in the current DLM state, the boot firmware sends a “Command acceptance error”.
- If HUK has been zeroized, the boot firmware sends a “Command acceptance error”.
- If device reset is not asserted after the Encrypted data write command execution, the boot firmware sends a “Command acceptance error”.
- If current Authentication level is AL1 or AL0, the boot firmware sends a “Secure error”.
- When the above error occurs, the boot firmware does not process and returns to the command waiting state.
  * Memory status do not change before command reception.

When the processing above is successfully completed, the boot firmware analyzes the command parameters.

- If KID is not specified as Key type, the boot firmware will send a “Parameter error”.
- If the hash value of public-key ID is already stored in the device, the boot firmware sends a “Protection error”.
- When the above error occurs, the boot firmware does not process and returns to the command waiting state.
  * Memory status does not change before command reception.
- If the above error does not occur, the boot firmware sends “OK”.

When the processing above is successfully completed, boot firmware receives and analyzes data packet.

- Boot firmware detects the beginning of a data packet by receiving SOD. When boot firmware receives other data than SOD, it discards the data and waits for the next data until SOD is sent.
- When the received data packet does not have ETX, “Packet error” is returned.
- When SUM in the received data packet is different from the value calculated by boot firmware, “Checksum error” is returned.
- When LNH and LNL in the received data packet do not comply with the packet format, “Packet error” is returned.
- When RES in the received data packet is different from defined values, “Packet error” is returned.
- When the number of received data exceeds the value specified by the command in the received data packet, the boot firmware sends a “Parameter error”.
- When the above error occurs, the boot firmware does not process and returns to the command waiting state.
  * Memory status does not change before command reception.

When all key data has been received, the boot firmware generates a hash value.

- If the Trusted system becomes abnormal after generating the hash value of OEM_ROOT_PK, the boot firmware returns nothing and no response.
  * Memory status does not change before command reception
- If the hash value of OEM_ROOT_PK fails to be generated, the boot firmware sends a “Trusted system error” and returns to the command waiting state.

* Memory status does not change before command reception

Boot firmware writes hash value of OEM_ROOT PK and MAC after hash value generation.

- If an error occurs while writing hash value of OEM_ROOT PK and MAC value, the boot firmware sends a “MRAM sequencer error” and returns to the command wait state.
  * Memory status is hash value of OEM_ROOT_PK and MAC is indefinite.
- If the hash value of OEM_ROOT_PK and MAC is successfully saved to the device, the boot firmware sends “OK” and returns to the command wait state.
  * The hash value of OEM_ROOT_PK and MAC is written to the memory.

6.27.4 Status Information from the Microcontroller

(Listed in descending order of priority.)

Condition	STS	ST2	ADR
The received packet does not have ETX.	Packet error	FFFFFFFFh	FFFFFFFFh
Sum data in the packet received is different from the value calculated by the boot firmware.	Checksum error	FFFFFFFFh	FFFFFFFFh
Packet length in the received packet do not comply with the packet format.	Packet error	FFFFFFFFh	FFFFFFFFh
Packet length in the received packet do not comply with the specifications of this command.	Packet error	FFFFFFFFh	FFFFFFFFh
Executing this command is unavailable in current DLM state.	Command acceptance error	FFFFFFFFh	FFFFFFFFh
HUK has been zeroized.	Command acceptance error	FFFFFFFFh	FFFFFFFFh
Device reset is not asserted after Encrypted data write command execution.	Command acceptance error	FFFFFFFFh	FFFFFFFFh
Authentication level is AL1 or AL0.	Secure Error	FFFFFFFFh	FFFFFFFFh
The specified Public-key ID is an unsupported value.	Parameter error	FFFFFFFFh	FFFFFFFFh
The Hash value of the specified Public-key ID is already stored in the device.	Protection error	FFFFFFFFh	FFFFFFFFh
The response code of the received data packet is different from the value specified by this command.	Packet error	FFFFFFFFh	FFFFFFFFh
The total length of received data of data packets exceeds the value specified in the command.	Parameter error	FFFFFFFFh	FFFFFFFFh
Hash value generation failed.	Trusted system error	FFFFFFFFh	FFFFFFFFh
MACI detected an error after the command execution in an undisclosed area.	MRAM sequencer error	MRAM status	FFFFFFFFh
Successful completion.	OK	FFFFFFFFh	FFFFFFFFh

6.28 OEM Root Public Key Verify Command

This command verifies the Hash value of OEM root public key which is stored in the device.
This command requires adherence to conditions described in Command List

6.28.1 Sequence Diagram

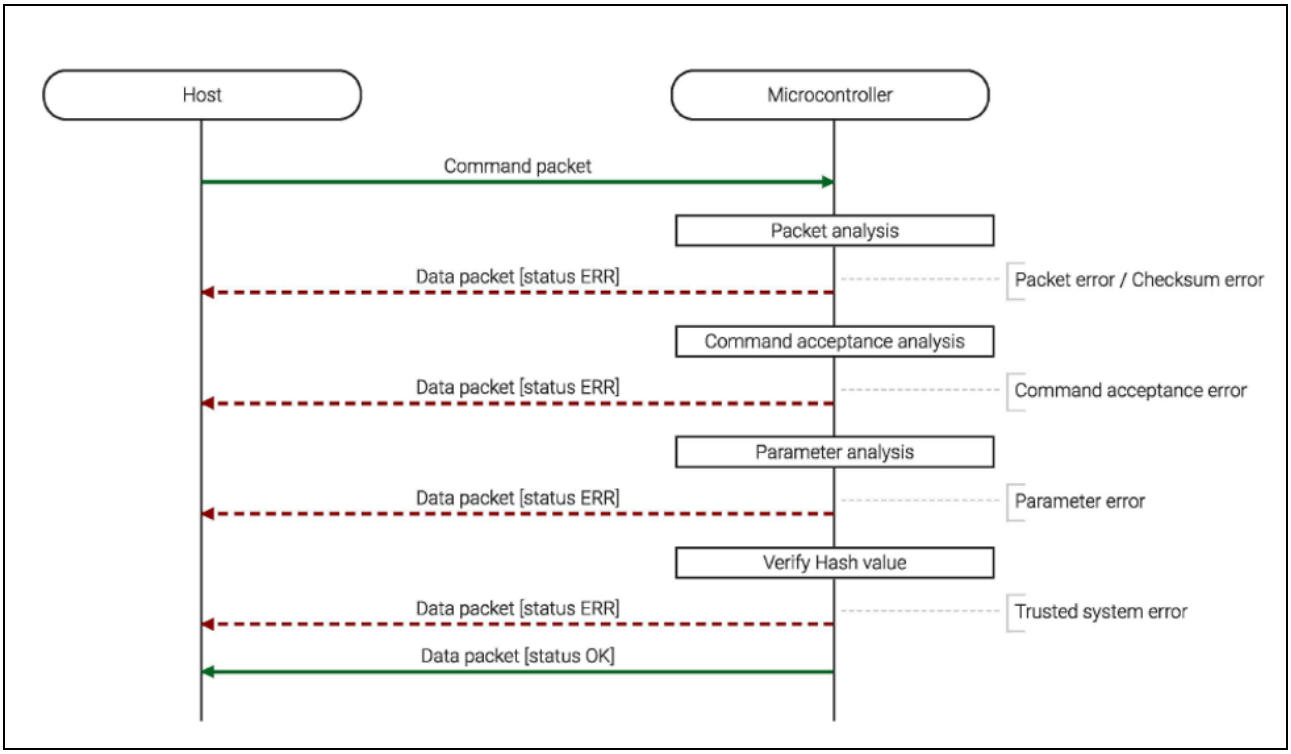


Figure 47. OEM Root Public Key Verify Command

6.28.2 Packets

6.28.2.1 Command Packet

SOH	(1 byte)	01h
LNH	(1 byte)	00h
LNL	(1 byte)	02h
CMD	(1 byte)	2Fh (OEM root public key setting command)
KID	(1 byte)	Public-key ID 00h: Public-key 0 01h: Public-key 1 02h: Public-key 2 03h: Public-key 3
SUM	(1 byte)	Sum data
ETX	(1 byte)	03h

6.28.2.2 Data Packet [Status OK]

SOD	(1 byte)	81h
LNH	(1 byte)	00h
LNL	(1 byte)	0Ah
RES	(1 byte)	2Fh (OK)
STS	(1 byte)	00h (OK)
ST2	(4 bytes)	FFFFFFFFh (unused code)
ADR	(4 bytes)	FFFFFFFFh (unused code)
SUM	(1 byte)	Sum data
ETX	(1 byte)	03h

6.28.2.3 Data Packet [Status ERR]

SOD	(1 byte)	81h
LNH	(1 byte)	00h
LNL	(1 byte)	0Ah
RES	(1 byte)	AFh (ERR)
STS	(1 byte)	Status code
ST2	(4 bytes)	Status details
ADR	(4 bytes)	Failure address
SUM	(1 byte)	Sum data
ETX	(1 byte)	03h

6.28.3 Processing Procedure

Boot firmware receives and analyzes a command packet:

- The boot firmware recognizes the start of the command packet by receiving SOH.
  If the boot firmware receives something other than SOH, it waits until it receives SOH.
- If ETX is not added to the received command packet, the boot firmware sends a “Packet error”.
- If the SUM of the received command packet is different from the sum value, the boot firmware sends a “Checksum error”.
- If the LNH and LNL of the received command packet are different from the values specified in the packet format, the boot firmware sends a “Packet error”.
- If the LNH and LNL of the received command packet are different from the values specified in each command, the boot firmware sends a “Packet error”.
- When any of the above errors occurs, the boot firmware does not process and returns to the command waiting state.
  * Memory status does not change before command reception.

When the processing above is successfully completed, boot firmware executes the acceptance analysis:

- If HUK has been zeroized, the boot firmware sends a 'Command acceptance error'.
- If the device reset is not asserted after the Encrypted data write command execution, the boot firmware sends a “Command acceptance error”.
- When the above error occurs, the boot firmware does not process and returns to the command waiting state.
  * Memory status does not change before command reception

When the processing above is successfully completed, boot firmware analyzes the command parameters:

- If KID is an unsupported value, the boot firmware sends a “Parameter error” and returns to the command wait state.
  * Memory status does not change before command reception

When the processing above is successfully completed, boot firmware verifies the Hash value:

- If verification of Hash value fails, the boot firmware sends a “Trusted system error” and returns to the command wait state.
  * Memory status does not change before command reception
- If the verification of the Hash value is completed successfully, the boot firmware sends “OK” and returns to the command wait state.
  * Memory status does not change before command reception

6.28.4 Status Information from the Microcontroller

(Listed in descending order of priority.)

Condition	STS	ST2	ADR
The received packet does not have ETX.	Packet error	FFFFFFFFh	FFFFFFFFh
Sum data in the received packet is different from the value calculated by the boot firmware.	Checksum error	FFFFFFFFh	FFFFFFFFh
Packet length in the received packet do not comply with the packet format.	Packet error	FFFFFFFFh	FFFFFFFFh
Packet length in the received packet do not comply with the specifications of this command.	Packet error	FFFFFFFFh	FFFFFFFFh
HUK has been zeroized.	Command acceptance error	FFFFFFFFh	FFFFFFFFh
Device reset is not asserted after Encrypted data write command execution.	Command acceptance error	FFFFFFFFh	FFFFFFFFh
The specified Public-key ID is an unsupported value.	Parameter error	FFFFFFFFh	FFFFFFFFh
Hash value verification failed.	Trusted system error	FFFFFFFFh	FFFFFFFFh
Successful completion.	OK	FFFFFFFFh	FFFFFFFFh

6.29 Code Certificate Update Command

This command executes the following depending on the specified MAC type by the Command packet

[MAC type: HMAC-SHA256]

- Check the integrity of the following:
  “Hash of OEM root public key” and “Key certificate”
  “Key certificate” and “Code certificate”
  “Code certificate” and “OEM boot loader”
- Write the “Code certificate” and “MAC value of Code certificate and OEM boot loader” to the area indicated by Code certificate start address
- Update the version of OEM boot loader to the value indicated by Code certificate

[MAC type: None]

- Calculate the CRC value of the OEM boot loader and compare it with the CRC value included in Code certificate.
- Write the “Code certificate” to the area indicated by Code certificate start address

This command requires adherence to the conditions described in Command List.

6.29.1 Sequence Diagram

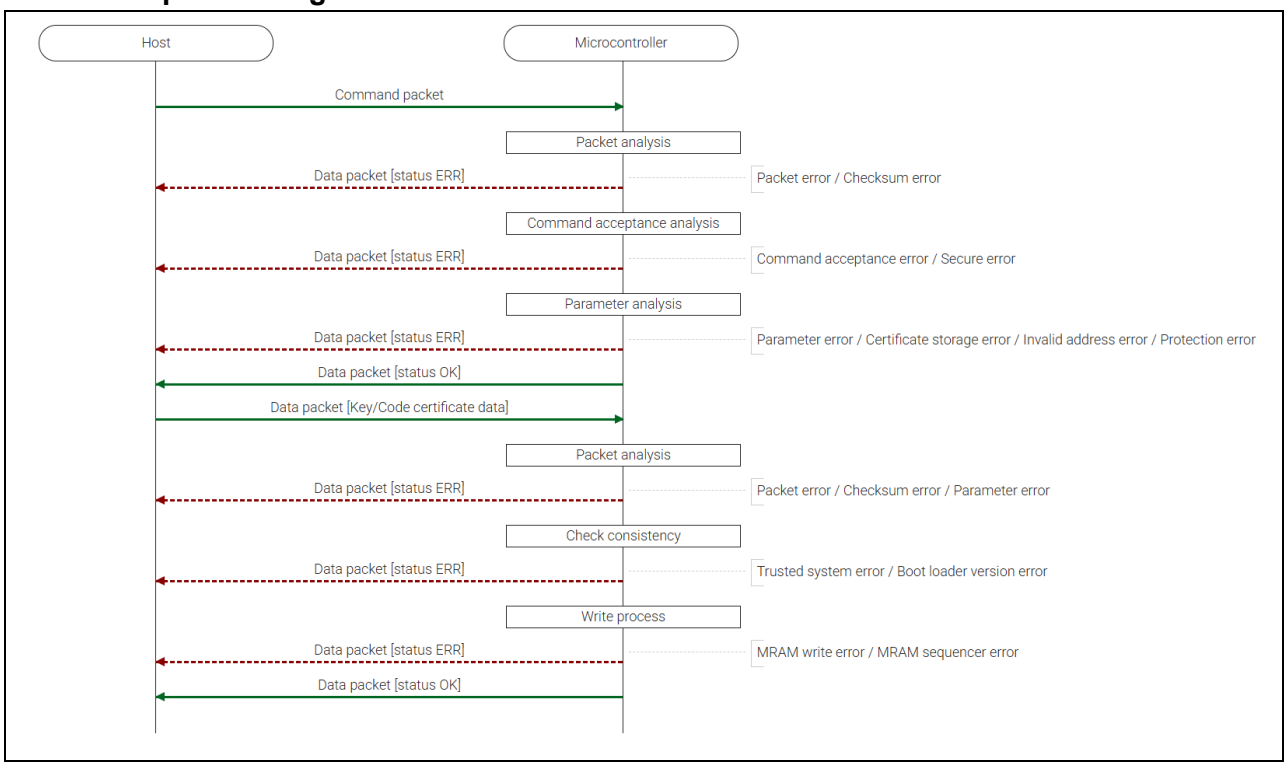


Figure 48. Code Certificate Update Command Diagram

6.29.2 Packets

6.29.2.1 Command Packet

SOH	(1 byte)	01h
LNH	(1 byte)	00h
LNL	(1 byte)	06h
CMD	(1 byte)	26h (Code certificate update command)
MAC	(1 byte)	• MAC type 02h: HMAC-SHA256 • FFh: None (CRC check)
KCS	(2 bytes)	Key certificate size Max. 208 bytes e.g.) 208 bytes -> 00h, D0h
CCS	(2 bytes)	Code certificate size Max. 216 bytes e.g.) 216 bytes -> 00h, D8h
SUM	(1 byte)	Sum data
ETX	(1 byte)	03h

6.29.2.2 Data Packet [Key/Code certificate data]

SOD	(1 byte)	81h
LNH	(1 byte)	N + M + 1 (Higher 1 byte)
LNL	(1 byte)	N + M + 1 (Lower 1 byte)
RES	(1 byte)	26h (OK)
KCD	(N byte)	Key certificate data
CCD	(M byte)	Code certificate data
SUM	(1 byte)	Sum data
ETX	(1 byte)	03h

N = KCS, M = CCS

6.29.2.3 Data Packet [status OK]

SOD	(1 byte)	81h
LNH	(1 byte)	00h
LNL	(1 byte)	0Ah
RES	(1 byte)	26h (OK)
STS	(1 byte)	00h (OK)
ST2	(4 bytes)	FFFFFFFFh (unused code)
ADR	(4 bytes)	FFFFFFFFh (unused code)
SUM	(1 byte)	Sum data
ETX	(1 byte)	03h

6.29.2.4 Data Packet [status ERR]

SOD	(1 byte)	81h
LNH	(1 byte)	00h
LNL	(1 byte)	0Ah
RES	(1 byte)	A6h (ERR)
STS	(1 byte)	Status code
ST2	(4 bytes)	Status details
ADR	(4 bytes)	Failure address
SUM	(1 byte)	Sum data
ETX	(1 byte)	03h

6.29.3 Processing Procedure

Boot firmware receives and analyzes a command packet.

- The boot firmware recognizes the start of the command packet by receiving SOH.
  If the boot firmware receives something other than SOH, it waits until it receives SOH.
- If ETX is not added to the received command packet, the boot firmware sends a “Packet error”.
- If the SUM of the received command packet is different from the sum value, the boot firmware sends a “Checksum error”.
- If the LNH and LNL of the received command packet are different from the values specified in the packet format, the boot firmware sends a “Packet error”.
- If the LNH and LNL of the received command packet are different from the values specified in each command, the boot firmware sends a “Packet error”.
- When the above error occurs, the boot firmware does not process and returns to the command waiting state.
  * Memory status does not change before command reception

When the processing above is successfully completed, boot firmware executes the acceptance analysis.

- If this command cannot be executed in the current DLM state, the boot firmware sends a “Command acceptance error”.
- If HUK has been zeroized, the boot firmware sends a “Command acceptance error”.
- If the current Authentication level is AL 1 or ALO, the boot firmware sends a “Secure error”.
- When the above error occurs, the boot firmware does not process and returns to the command waiting state.
  * Memory status does not change before command reception

When the processing above is successfully completed, boot firmware analyzes the command parameters.

- When any of the following conditions are met, boot firmware returns “Parameter error”.
  - MAC is not specified as MAC type
  - KCS is larger than specified max size (208 bytes)
  - CCS is larger than specified max size (216 bytes)
- If the area for writing “Code certificate” when MAC type is “None” or “Code certificate” and “MAC value of Code certificate and OEM boot loader” when the type is other than “None” extends outside the range of User area, the boot firmware sends a “Certificate storage error”.
- If the area for writing “Code certificate” when MAC type is “None” or “Code certificate” and “MAC value of Code certificate and OEM boot loader” when the type is other than “None” is across different KOAs, the boot firmware sends a “Certificate storage error”.
- If the WAU of the area for writing “Code certificate” when MAC type is 'None' or “Code certificate” and “MAC value of Code certificate and OEM boot loader” when the type is other than “None” is 0, the boot firmware sends a “Certificate storage error”.
- If the area for writing “Code certificate” when MAC type is “None” or “Code certificate” and “MAC value of Code certificate and OEM boot loader” when the type is other than “None” is not specified in the WAU for the addresses, the boot firmware sends a “Certificate storage error”.
- If the area for writing “Code certificate” when MAC type is “None” or “Code certificate” and “MAC value of Code certificate and OEM boot loader” when the type is other than “None” includes address that is inaccessible with current boundary setting, the boot firmware sends a “Invalid address error”
- If the area for writing “Code certificate” when MAC type is “None” or “Code certificate” and “MAC value of Code certificate and OEM boot loader” when the type is other than “None” contains a permanent protected block, the boot firmware sends a “Protection error”.
- If lock bit of the Anti-Rollback Counter for OEM_BL is set when MAC type is other than “None”, the boot firmware sends a “Protection error”.
- When the above error occurs, the boot firmware does not process and returns to the command waiting state.
  * Memory status does not change before command reception
- If the above error does not occur, the boot firmware sends “OK”.

When the processing above is successfully completed, boot firmware receives and analyzes data packet:

- Boot firmware detects the beginning of a data packet by receiving SOD.
  When boot firmware receives other data than SOD, it discards the data and waits for the next data until SOD is sent.
- When the received data packet does not have ETX, “Packet error” is returned.
- When SUM in the received data packet is different from the value calculated by boot firmware, “Checksum error” is returned.
- When LNH and LNL in the received data packet do not comply with the packet format, “Packet error” is returned.
- When RES in the received data packet is different from defined values, “Packet error” is returned.
- When the number of received data exceeds the value specified by the command in the received data packet, the boot firmware sends a “Parameter error”.
- When any of the above errors occurs, the boot firmware does not process and returns to the command waiting state.
  * Memory status does not change before command reception.

When the processing above is successfully completed, boot firmware verifies the consistency.

- If OEM boot loader version indicated by Code certificate is in any cases below when MAC type is other than “None”, the boot firmware sends a “Boot loader version error” and returns to the command
  - If OEM boot loader version indicated by Code certificate is greater than the maximum value of OEM boot loader version defined by device specifications.
  - If OEM boot loader version indicated by Code certificate is less than or equal to the OEM boot loader version that is already written.

  * Memory status does not change before command reception.

- If the Trusted system becomes abnormal after verification of consistency, the boot firmware returns nothing and no response.
  * Memory status does not change before command reception
- If the verification of consistency fails, the boot firmware sends a “Trusted system error” and returns to the command waiting state.
  * Memory status does not change before command reception

When the verification of consistency success. the boot firmware writes “Code certificate” when MAC type is “None” or “Code certificate” and “MAC value of Code certificate and OEM boot loader” when the type to the Code certificate start address.

- If an error occurs while writing “Code certificate” or “MAC value of Code certificate and OEM boot loader”, the boot firmware sends a “MRAM write error” and returns to the command wait state.
  * WAU size from failure address (ADR) of memory area is undefined.
- When the writing of Code certificate is normally finished and MAC type is None (FFh), the boot firmware returns “OK” and waits for the next command.
  * Code certificates are written to memory.

When successful writing and MAC type is other than “None“, the boot firmware updates OEM boot loader version.

- If an error occurs while updating OEM boot loader version, the boot firmware sends a “MRAM sequencer error” and returns to the command wait state.
  * Anti RollBack Counter for OEM_BL of memory area is undefined.
- When the OEM boot loader version is successfully updated, the boot firmware sends “OK” and returns to the command wait state.
  * Code certificate and MAC value are written and OEM boot loader version is updated.

6.29.4 Status Information from the Microcontroller

(Listed in descending order of priority.)

Condition	STS	ST2	ADR
The received packet does not have ETX.	Packet error	FFFFFFFFh	FFFFFFFFh
Sum data in the packet received is different from the value calculated by the boot firmware.	Checksum error	FFFFFFFFh	FFFFFFFFh
Packet length in the received packet does not comply with the packet format.	Packet error	FFFFFFFFh	FFFFFFFFh
Packet length in the received packet does not comply with the specifications of this command.	Packet error	FFFFFFFFh	FFFFFFFFh
Executing this command is unavailable in the current DLM state.	Command acceptance error	FFFFFFFFh	FFFFFFFFh
HUK has been zeroized.	Command acceptance error	FFFFFFFFh	FFFFFFFFh
Authentication level is AL1 or AL0.	Secure error	FFFFFFFFh	FFFFFFFFh
The specified MAC type is an unsupported value.	Parameter error	FFFFFFFFh	FFFFFFFFh
Key certificate size exceeds the specified value.	Parameter error	FFFFFFFFh	FFFFFFFFh
Code certificate size exceeds the specified value.	Parameter error	FFFFFFFFh	FFFFFFFFh
The area for writing “Code certificate” and “MAC value of Code certificate and OEM boot loader (*1)” extends outside the range of User area.	Certificate storage error	FFFFFFFFh	FFFFFFFFh
The area for writing “Code certificate” and “MAC value of Code certificate and OEM boot loader (*1)” spans different Kind of the area.	Certificate storage error	FFFFFFFFh	FFFFFFFFh
The area for writing WAU of “Code certificate” and “MAC value of Code certificate and OEM boot loader(*1)” is 0.	Certificate storage error	FFFFFFFFh	FFFFFFFFh
The area for writing “Code certificate” and “MAC value of Code certificate and OEM boot loader(*1)” is not specified in the WAU for the addresses.	Certificate storage error	Trusted system error	FFFFFFFFh
The area for writing “Code certificate” and “MAC value of Code certificate and OEM boot loader (*1)” contains addresses that are inaccessible with the current boundary settings.	Invalid address error	FFFFFFFFh	FFFFFFFFh
The area for writing “Code certificate” and “MAC value of Code certificate and OEM boot loader (*7)” includes permanent protected block.	Protection error	FFFFFFFFh	FFFFFFFFh
Anti Rollback Counter for OEM_BL Lock bit is set. (*1)	Protection error	FFFFFFFFh	FFFFFFFFh
The response code of the received data packet is different from the value specified by this command.	Packet error	FFFFFFFFh	FFFFFFFFh
The number of received KCD and CCD data in the received data packet is different from the KCS and CCS specified in the command packet.	Parameter error	FFFFFFFFh	FFFFFFFFh
OEM boot loader version indicated by Code certificate is greater than the maximum value of OEM boot loader version defined by device specifications. (*1)	Boot loader version error	FFFFFFFFh	FFFFFFFFh
OEM boot loader version indicated by Code certificate is less than or equal to the OEM boot loader version that is already written. (*1)	Boot loader version error	FFFFFFFFh	FFFFFFFFh

Verification of consistency failed.	Trusted system error	Trusted system status	FFFFFFFFh
An error detected when MRAM programming which does not use MAGI command.	MRAM write error	FFFFFFFFh	Failure address
MACI detected an error after the command execution when updating OEM boot loader version (*1).	MRAM sequencer error	MRAM status	FFFFFFFFh
Successful completion.	OK	FFFFFFFFh	FFFFFFFFh

(*1) Only when MAC type = HMAC-SHA256

6.29.5 Precautions

(1) Use this command after writing “OEM boot loader” to the User area and “Code certificate start address” to the Config OTP area with the Write command or Encrypted data write command in advance.

(2) Use this command after saving the “OEM root public key encrypted Hash value” in the device in advance with the OEM root public key setting command.

(3) Verification fails if data of received Key certificate or Code certificate does not comform to device specifications.

Refer to the user's manual of the device for certificates' specifications.

(4) Key certificate is not necessary when MAC type = None. Specify KCS = 0 and do not send any data as Key certificate data in this case.

6.30 Code Certificate Check Command

This command executes the followings depending on the specified MAC type by the Command packet

[MAC type: HMAC-SHA256]

- Check the consistency of “Code certificate” and “the MAC value of Code certificate and OEM boot loader” which are stored in the device.
- Read and return the version of OEM boot loader which is stored in the device.

[MAC type: None]

- Calculate the CRC value of the OEM boot loader and compare it with the CRC value included in Code certificate which are stored in the device.

This command requires adherence to conditions described in Command List.

6.30.1 Sequence Diagram

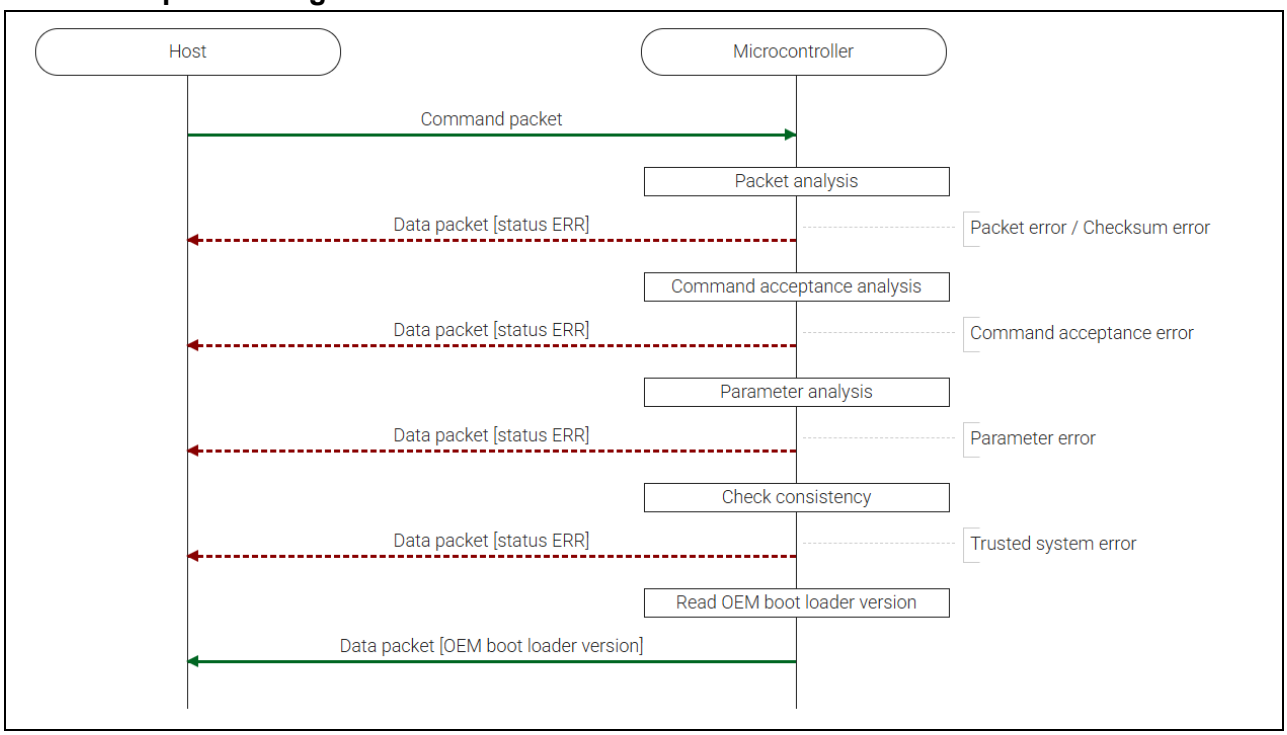


Figure 49. Code Certificate Check Command Diagram

6.30.2 Packets

6.30.2.1 Command Packet

SOH	(1 byte)	01h
LNH	(1 byte)	00h
LNL	(1 byte)	06h
CMD	(1 byte)	27h (Code certificate update command)
MAC	(1 byte)	• MAC type 02h: HMAC-SHA256 • FFh : None (CRC check)
KCS	(2 bytes)	Key certificate size Max. 208 bytes For example: 208 bytes -> 00h, D0h Specify fixed size 208 bytes since KCS is unused in this product.
CCS	(2 bytes)	Code certificate size Max. 216 bytes For example: 216 bytes -> 00h, DBh
SUM	(1 byte)	Sum data
ETX	(1 byte)	03h

6.30.2.2 Data Packet [OEM Boot Loader Version]

SOH	(1 byte)	81h
LNH	(1 byte)	00h
LNL	(1 byte)	05h
RES	(1 byte)	27h (OK)
BLV	(4 bytes)	OEM boot loader version (unused when MAC type = None) For example: Version = 10 -> 00h, 00h, 00h, 0Ah When MAC type = None, this field is not used and always FFFFFFFFh.
SUM	(1 byte)	Sum data
ETX	(1 byte)	03h

6.30.2.3 Data Packet [Status ERR]

SOD	(1 byte)	81h
LNH	(1 byte)	00h
LNL	(1 byte)	0Ah
RES	(1 byte)	A7h (ERR)
STS	(1 byte)	Status code
ST2	(4 bytes)	Status details
ADR	(4 bytes)	Failure address
SUM	(1 byte)	Sum data
ETX	(1 byte)	03h

6.30.3 Processing Procedure

Boot firmware receives and analyzes a command packet:

- The boot firmware recognizes the start of the command packet by receiving SOH.
  If the boot firmware receives something other than SOH, it waits until it receives SOH.
- If ETX is not added to the received command packet, the boot firmware sends a “Packet error”.
- If the SUM of the received command packet is different from the sum value, the boot firmware sends a “Checksum error”.
- If the LNH and LNL of the received command packet are different from the values specified in the packet format, the boot firmware sends a “Packet error”.
- If the LNH and LNL of the received command packet are different from the values specified in each command, the boot firmware sends a “Packet error”.
- When any of the above errors occurs, the boot firmware does not process and returns to the command waiting state.
  * Memory status does not change before command reception.

When the processing above is successfully completed, boot firmware executes the acceptance analysis:

- If HUK has been zeroized, the boot firmware sends a 'Command acceptance error'.
- When any of the above errors occurs, the boot firmware does not process and returns to the command waiting state.
  * Memory status does not change before command reception.

When the processing above is successfully completed, the boot firmware analyzes the command parameters:

- When any of the following conditions are met, boot firmware returns 'Parameter error”.
  - MAC is not specified as MAC type
  - KCS is larger than specified max size (208 bytes)
  - CCS is larger than specified max size (216 bytes)
- When the above error occurs, the boot firmware does not process and returns to the command waiting state.

* Memory status does not change before command reception

When the processing above is successfully completed, the boot firmware checks the consistency.

- If consistency check fails, the boot firmware sends a “Trusted system error” and returns to the command wait state.
  However, boot firmware sends nothing with no response if the Trusted system becomes abnormal during the consistency check.
  * Memory status does not change before command reception

When the verification of consistency is successful, the boot firmware returns version of OEM boot loader.
*When MAC type is “None”, OEM boot loader version is returned all-F.

- When any of the above errors occurs, the boot firmware does not process and returns to the command waiting state.
  * Memory status does not change before command reception.

6.30.4 Status Information from the Microcontroller

(Listed in descending order of priority.)

Condition	STS	ST2	ADR
The received packet does not have ETX.	Packet error	FFFFFFFFh	FFFFFFFFh
Sum data in the packet received is different from the value calculated by the boot firmware.	Checksum error	FFFFFFFFh	FFFFFFFFh
Packet length in the received packet does not comply with the packet format.	Packet error	FFFFFFFFh	FFFFFFFFh
Packet length in the received packet does not comply with the specifications of this command.	Packet error	FFFFFFFFh	FFFFFFFFh
HUK has been zeroized.	Command acceptance error	FFFFFFFFh	FFFFFFFFh
The specified MAC type is an unsupported value.	Parameter error	FFFFFFFFh	FFFFFFFFh
Key certificate size exceeds the specified value.	Parameter error	FFFFFFFFh	FFFFFFFFh
Code certificate size exceeds the specified value.	Parameter error	FFFFFFFFh	FFFFFFFFh
Consistency check failed.	Trusted system error	Trusted system error	FFFFFFFFh

6.31 SiP Flash Various Setting Command

This command executes various settings for SiP Flash.
This command requires adherence to conditions described in Command List.

6.31.1 Sequence Diagram

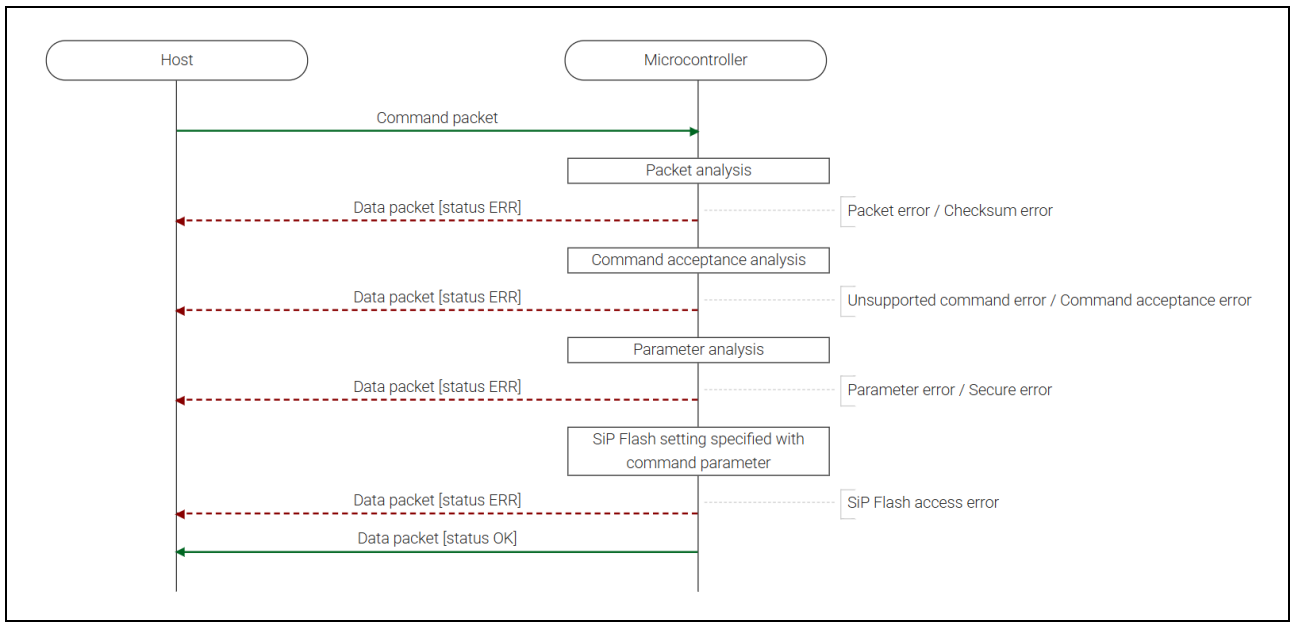


Figure 50. SiP Flash Various Setting Command Diagram

6.31.2 Packets

6.31.2.1 Command Packet

SOH	(1 byte)	01h
LNH	(1 byte)	N + 2 (Higher 1 byte)
LNL	(1 byte)	N + 2 (Lower 1 byte)
CMD	(1 byte)	38h (SiP Flash various setting command)
PRM	(1 byte)	Specify what operation to execute (See “PRM and DAT definition” below)
DAT	(Nbyte)	Data required for the specified operation (See “PRM and DAT definition” below)
SUM	(1 byte)	Sum data
ETX	(1 byte)	03h

N depends on the specified PRM. See “Size” column of “PRM and DAT definition” below.

6.31.2.2 Data Packet [Status OK]

SOD	(1 byte)	81h
LNH	(1 byte)	00h
LNL	(1 byte)	0Ah
RES	(1 byte)	38h (OK)
STS	(1 byte)	00h (OK)
ST2	(4 bytes)	FFFFFFFFh (unused code)
ADR	(4 bytes)	FFFFFFFFh (unused code)
SUM	(1 byte)	Sum data
ETX	(1 byte)	03h

6.31.2.3 Data Packet [Status ERR]

SOD	(1 byte)	81h
LNH	(1 byte)	00h
LNL	(1 byte)	0Ah
RES	(1 byte)	B8h (ERR)
STS	(1 byte)	Status code
ST2	(4 bytes)	Status details
ADR	(4 bytes)	Failure address
SUM	(1 byte)	Sum data
ETX	(1 byte)	03h

6.31.3 Processing Procedure

Boot firmware receives and analyzes a command packet:

- The boot firmware recognizes the start of the command packet by receiving SOH.
  If the boot firmware receives something other than SOH, it waits until it receives SOH.
- If ETX is not added to the received command packet, the boot firmware sends a “Packet error”.
- If the SUM of the received command packet is different from the sum value, the boot firmware sends a “Checksum error”.
- If the LNH and LNL of the received command packet are different from the values specified in the packet format, the boot firmware sends a “Packet error”.
- When any of the above errors occurs, the boot firmware does not process and returns to the command waiting state.
  * Memory status does not change before command reception.

When the processing above is successfully completed, boot firmware executes the acceptance analysis:

- If the product does not have a SiP Flash area, the boot firmware sends a “Unsupported command error”.
- If this command cannot be executed in the current DLM state, the boot firmware sends a “Command acceptance error”.
- When the above error occurs, the boot firmware does not process and returns to the command waiting state.
  * Memory status does not change before command reception.

When the processing above is successfully completed, boot firmware analyzes the command parameters:

- When specified PRM is unspecified value, boot firmware returns “Parameter error”.
- When the size of the received DAT is not the correct size associated with the specified PRM, the boot firmware returns “Parameter error”.
- When any of the following conditions are met, the boot firmware returns “Secure error”:
    - Current Authentication level is AL0
    - When all the following conditions are met:
        - Current Authentication level is AL1
        - The specified PRM is “PMR programming”
        - The area allocated to the secure area by boundary setting is located in the SiP Flash area.
    - When all the following conditions are met:
        - Current Authentication level is AL1
        - The specified PRM is “Status register programming”
        - This is an operation to change the protect setting of the area allocated to the secure area by boundary setting.

- When any of the above errors occurs, the boot firmware does not process and returns to the command waiting state.
  * Memory status does not change before command reception.

When the processing above is successfully completed, the boot firmware rewrites the setting of SiP Flash specified by PRM.

- If an error occurs while writing, the boot firmware sends a “SiP Flash access error” and returns to the command wait state.
  * The value of register or OTP area in memory is undefined.
- When the setting of SiP Flash successful completion, 'OK" is returned and waits for the next command.

6.31.4 Status Information from the Microcontroller

(Listed in descending order of priority.)

Condition	STS	ST2	ADR
The received packet does not have ETX.	Packet error	FFFFFFFFh	FFFFFFFFh
Sum data in the received packet is different from the value calculated by the boot firmware.	Checksum error	FFFFFFFFh	FFFFFFFFh
Packet length in the received packet does not comply with the packet format.	Packet error	FFFFFFFFh	FFFFFFFFh
The product does not have a SiP Flash area.	Unsupported command error	FFFFFFFFh	FFFFFFFFh
Executing this command is unavailable in current DLM state.	Command acceptance error	FFFFFFFFh	FFFFFFFFh
The specified PRM is unspecified value.	Parameter error	FFFFFFFFh	FFFFFFFFh
The size of the received DAT is not the appropriate size according to the specified PRM.	Parameter error	FFFFFFFFh	FFFFFFFFh
The current Authentication level is AL0.	Secure error	FFFFFFFFh	FFFFFFFFh

Condition	STS	ST2	ADR
All of the following conditions are satisfied: -The current Authentication level is AL1 -The specified PRM is “PMR programming” -There is an area assigned as Secure by the boundary setting in SiP Flash area.	Secure error	FFFFFFFFh	FFFFFFFFh
All of the following conditions are satisfied: -The current Authentication level is AL1 -The specified PRM is 'Status register programming” -The operation is going to change protection setting for area assigned as Secure by the boundary setting.	Secure error	FFFFFFFFh	FFFFFFFFh
An error occurred while programming SiP Flash.	SiP Flash access error	SiP Flash status	FFFFFFFFh
Successful completion.	OK	FFFFFFFFh	FFFFFFFFh

6.31.5 Precautions

(1) It is not possible to execute either setting or reading the functions below which SiP Flash has via boot firmware.

- Configuration register
- Non-volatile lock bit
- Password
- Sector protection register

In addition, when these settings are changed from the shipped state by a software other than boot firmware, operations of SiP Flash via boot firmware are not guaranteed. In this case, it is possible again to execute operations of SiP Flash via boot firmware. The user program sets these settings back to the shipped state.

(2) Programming of OTP array is permanently disabled after OTP permanent lock mapped in OTP array itself is set. Programming of PMR is permanently disabled after PMR permanent lock mapped in PMR itself is set.

Programming of Status register is permanently disabled after Status register permanent lock mapped in PMR is set.

(At this time, programming of the SiP Flash area protected by BP is also permanently disabled when BP is set by the Status register).

(3) When SiP Flash programming fails by this command or other commands, that is, when a SiP Flash access error is returned or the programmed data is not reflected, it may be that an operation against a protection function of SiP Flash was executed. The table below shows protection functions of SiP Flash that may cause a failure of SiP Flash programming via boot firmware. Check the settings of these protection functions when programming SiP Flash fails.

Protected area	Protection function	
	Name	Can be enabled by
Main Flash (referred to as SiP Flash area in the area information)	Block protect	Status register
	Lock bit	Non-volatile lock bit
OTP array	OTP permanent lock	OTP control byte placed in OTP array itself
Status register	Status register lock	PMR register
PMR register	PMR lock	PMR register itself

6.31.6 PRM and DAT Definition

This command returns various information about SiP Flash depending on the specified PRM. The meaning of DAT also relies on the specified PRM.

The table below shows the definition of PRM and DAT.

PRM			**DAT**	
Value	**Name**	**Explanation**	**Size**	**Explanation**
10h	OTP programming	Boot firmware programs the OTP array that the SiP Flash has.	65 bytes	DAT specifies data to be programmed to OTP array(*). The 1st byte of DAT is written to 1st byte of OTP array.
11h	Status register programming	Boot firmware programs Status register that the SiP Flash has.	1 byte	DAT specifies data to be programmed to Status register(*). When Status register has volatile bits, the DATs bit where volatile bit is assigned is ignored.
12h	PMR programming	Boot firmware programs PMR register that the SiP Flash has.	1 byte	DAT specifies data to be programmed to PMR register(*).

(*) Refer to the data sheet of the installed SiP Flash for detailed information of these data. What SiP Flash is installed in the device can be determined by various request commands of SiP Flash.

6.32 SiP Flash Various Request Command

This command returns various information about SiP Flash.

This command requires adherence to conditions described in Command List

6.32.1 Sequence Diagram

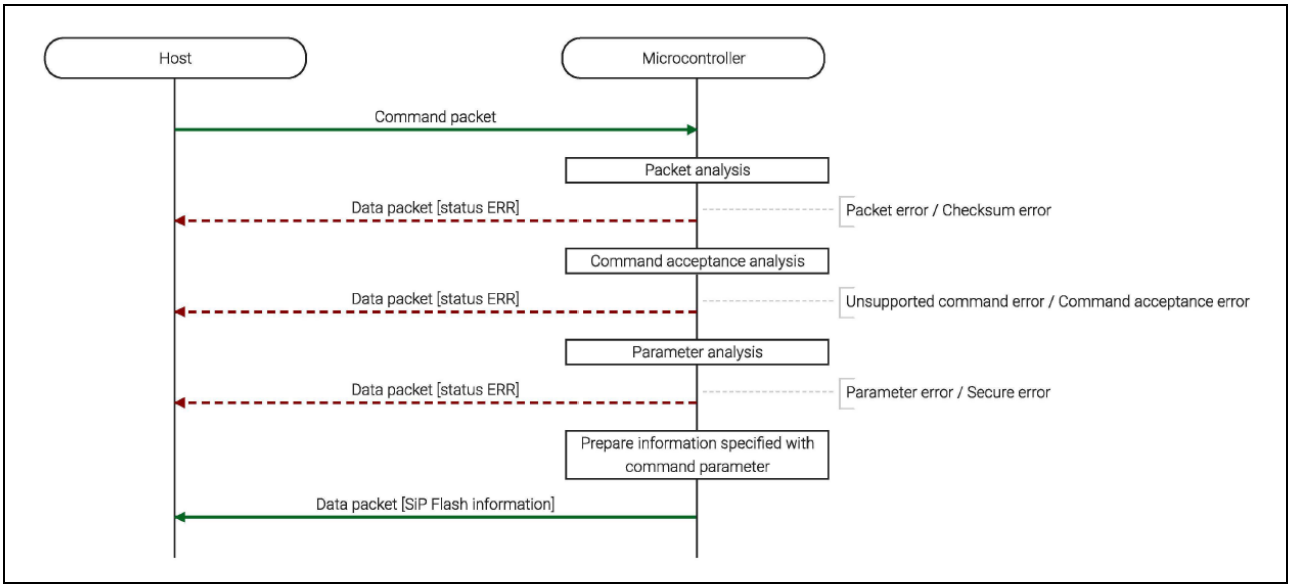


Figure 51. SiP Flash Various Request Command Diagram

6.32.2 Packets

6.32.2.1 Command Packet

SOH	(1 byte)	01h
LNH	(1 byte)	00h
LNL	(1 byte)	02h
CMD	(1 byte)	39h (various request commands of SiP Flash)
PRM	(1 byte)	Specify what information to require (see “PRM and DAT definition” below)
SUM	(1 byte)	Sum data
ETX	(1 byte)	03h

6.32.2.2 Data Packet [SiP Flash information]

SOD	(1 byte)	81h
LNH	(1 byte)	N + 1 (Higher 1 byte)
LNL	(1 byte)	N + 1 (Lower 1 byte)
RES	(1 byte)	39h (OK)
DAT	(N byte)	SiP Flash information (see “PRM and DAT definition” below)
SUM	(1 byte)	Sum data
ETX	(1 byte)	03h

6.32.2.3 Data Packet [Status ERR]

SOD	(1 byte)	81h
LNH	(1 byte)	00h
LNL	(1 byte)	0Ah
RES	(1 byte)	A7h (ERR)
STS	(1 byte)	Status code
ST2	(4 bytes)	Status details
ADR	(4 bytes)	Failure address
SUM	(1 byte)	Sum data
ETX	(1 byte)	03h

6.32.3 Processing Procedure

Boot firmware receives and analyzes a command packet:

- The boot firmware recognizes the start of the command packet by receiving SOH.
  If the boot firmware receives something other than SOH, it waits until it receives SOH.
- If ETX is not added to the received command packet, the boot firmware sends a “Packet error”.
- If the SUM of the received command packet is different from the sum value, the boot firmware sends a “Checksum error”.
- If the LNH and LNL of the received command packet are different from the values specified in the packet format, the boot firmware sends a “Packet error”.
- If the LNH and LNL of the received command packet are different from the values specified in each command, the boot firmware sends a “Packet error”.
- When any of the above errors occurs, the boot firmware does not process and returns to the command waiting state.
  * Memory status does not change before command reception.

When the processing above is successfully completed, boot firmware executes the acceptance analysis.

- If the product does not have a SiP Flash area, the boot firmware sends an “Unsupported command error”.
- If this command cannot be executed in the current DLM state, the boot firmware sends a “Command acceptance error”.
- When the above error occurs, the boot firmware does not process and returns to the command waiting state.
  * Memory status does not change before command reception.

When the processing above is successfully completed, the boot firmware analyzes the command parameters.

- When specified PRM is unspecified value, boot firmware returns “Parameter error”.

When any of the following conditions are met, boot firmware sends “Secure error”:

- When all the following conditions are met:
  - Current Authentication level is AL0
  - The specified PRM is other than “SiP Flash type”.
- When all the following conditions are met:
  - Current Authentication level is AL1
  - The specified PRM is “PMR reading”
  - The area allocated to the secure area by the boundary setting is located in the SiP Flash area.
- When all the following conditions are met:
  - Current Authentication level is AL1
  - The specified PRM is “Status register reading”
  - This entire SiP Flash area is assigned to the secure area by the boundary setting.

- When the above error occurs, the boot firmware does not process and returns to the command waiting state.

  * Memory status does not change before command reception

When the processing above is successfully completed, boot firmware returns information on SiP Flash specified by PRM.

- Boot firmware send “SiP Flash information” and waits for next command.

  * Memory status does not change before command reception.

6.32.4 Status Information from the Microcontroller

(Listed in descending order of priority.)

Condition	STS	ST2	ADR
The received packet does not have ETX.	Packet error	FFFFFFFFh	FFFFFFFFh
Sum data in the received packet is different from the value calculated by the boot firmware.	Checksum error		
Packet length in the received packet does not comply with the packet format.	Packet error	FFFFFFFFh	FFFFFFFFh
Packet length in the received packet does not comply with the specifications of this command.	Packet error	FFFFFFFFh	FFFFFFFFh
The product does not have a SiP Flash area.	Unsupported command error	FFFFFFFFh	FFFFFFFFh
Executing this command is unavailable in current DLM state.	Command acceptance error	FFFFFFFFh	FFFFFFFFh
The specified PRM is unspecified value.	Parameter error	FFFFFFFFh	FFFFFFFFh
All of the following conditions are satisfied: -The current Authentication level is AL0 -The specified PRM is not “SiP Flash type”	Secure error	FFFFFFFFh	FFFFFFFFh
All of the following conditions are satisfied: -The current Authentication level is AL1 -The specified PRM is “PMR reading” -There is an area assigned as Secure by the boundary setting in SiP Flash area.	Secure error	FFFFFFFFh	FFFFFFFFh
All of the following conditions are satisfied: -The current Authentication level is AL1 -The specified PRM is “Status register reading” - The entire SiP Flash area is assigned as Secure by the boundary setting	Secure error	FFFFFFFFh	FFFFFFFFh

6.32.5 PRM and DAT Definition

This command returns various information about SiP Flash depending on the specified PRM. The meaning of DAT also relies on the specified PRM.

The table below shows the definition of PRM and DAT.

PRM			**DAT**	
Value	**Name**	**Explanation**	**Size**	**Explanation**
00h	SiP Flash type request	Boot firmware returns a 4-byte value that shows what SiP Flash is installed in the device.	4 bytes	00000000h: ISSI IS25WX (*) Transmitted in order from the most significant byte to the least significant byte.
10h	OTP reading	Boot firmware reads the OTP array that the SiP Flash has.	65 bytes	Data read from OTP array(*). The 1st byte of DAT is the 7th byte of OTP array.
11h	Status register reading	Boot firmware reads Status register that the SiP Flash has.	1 byte	Data read from Status register (*). When Status register has volatile bits, the DAT bit is fixed to 0h.
12h	PMR reading	Boot firmware reads PMR register that the SiP Flash has.	1 byte	Data read from PMR register(*).

(*) Refer to the data sheet of the installed SiP Flash for detailed information of these data.

What SiP Flash is installed in the device can be determined by this command (PRM = 00h).

6.33 External Flash Memory Setting Command

This command configures the initial settings for external flash area access, receives external flash memory access driver codes from the host and stores them to RAM.

This command must be executed before executing other commands specifying external flash area.

This command requires adherence to conditions described in Command List.

6.33.1 Sequence Diagram

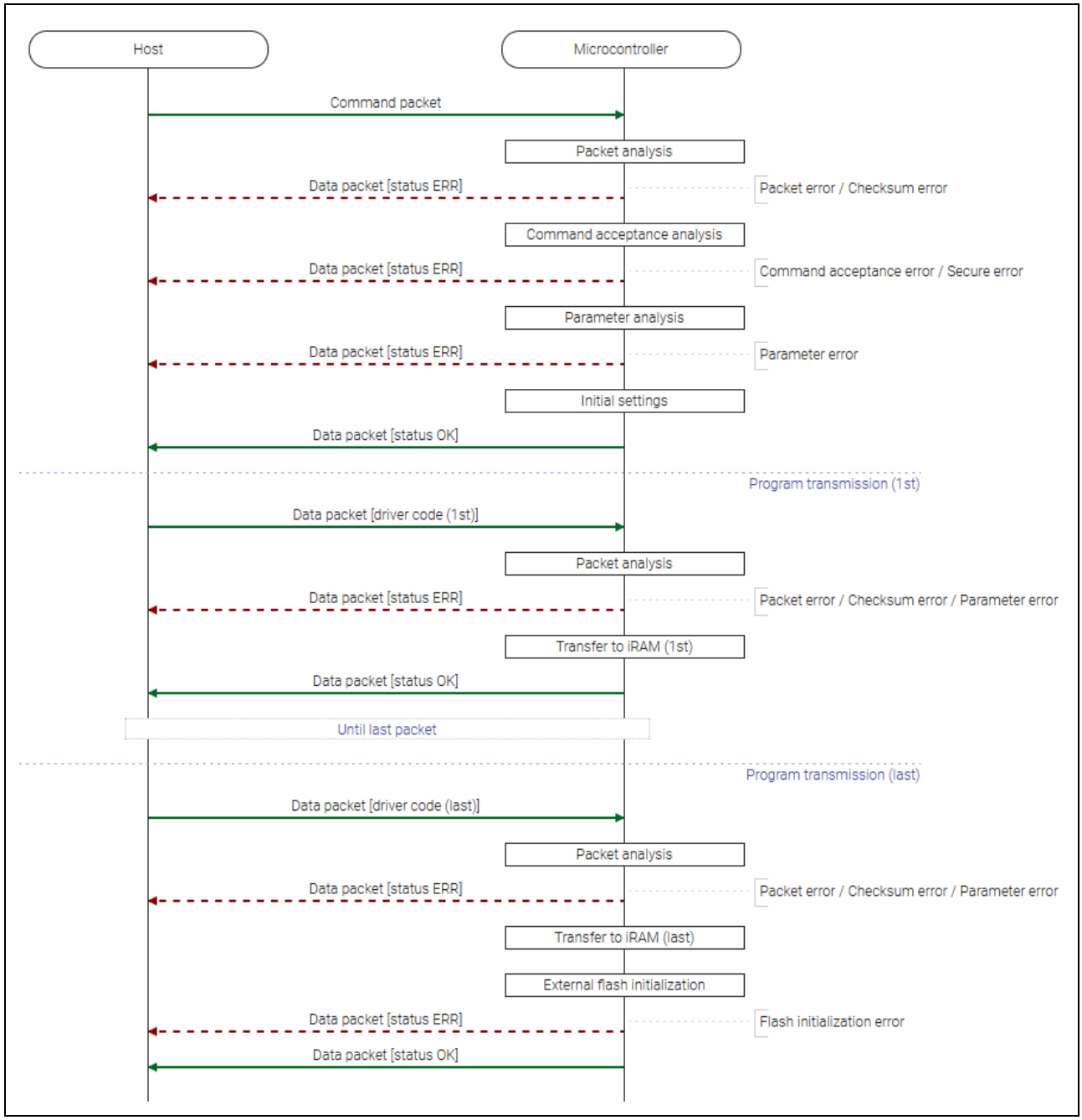


Figure 52. External Flash Memory Setting Command Sequence Diagram

6.33.2 Packets

6.33.2.1 Command Packet

SOH	(1 byte)	01h
LNH	(1 byte)	00h
LNL	(1 byte)	07h
CMD	(1 byte)	36h (External flash memory setting command)
OCK	(1 byte)	OCTACLK frequency: • 00h: 66.66 MHz • 01h: 100 MHz • 02h: 133.33 MHz • 03h: 200 MHz
VCC	(1 byte)	VCC2 voltage: • 00h: Lower than 2.7 V • 01h: Higher than or equal to 2.7 V
LOP	(4 bytes)	Data length of external flash memory access driver [bytes]. For example: 2048 bytes -> 0000_0800h -> 00h, 00h, 08h, 00h
SUM	(1 byte)	Sum data
ETX	(1 byte)	03h

6.33.2.2 Data Packet [Driver Code]

SOD	(1 byte)	81h
LNH	(1 byte)	N + 1 (Higher 1 byte)
LNL	(1 byte)	N + 1 (Lower 1 byte)
RES	(1 byte)	36h (OK)
DAT	(N bytes)	External flash memory access driver
SUM	(1 byte)	Sum data
ETX	(1 byte)	03h

N = 4 – 1024 (must be multiple of 4)

6.33.2.3 Data Packet [Status OK]

SOD	(1 byte)	81h
LNH	(1 byte)	00h
LNL	(1 byte)	0Ah
RES	(1 byte)	36h (OK)
STS	(1 byte)	00h (OK)
ST2	(4 bytes)	FFFFFFFFh (unused code)
ADR	(4 bytes)	FFFFFFFFh (unused code)
SUM	(1 byte)	C8h
ETX	(1 byte)	03h

6.33.2.4 Data Packet [Status ERR]

SOD	(1 byte)	81h
LNH	(1 byte)	00h
LNL	(1 byte)	0Ah
RES	(1 byte)	B6h (ERR)
STS	(1 byte)	Status code
ST2	(4 bytes)	Status details
ADR	(4 bytes)	Failure address
SUM	(1 byte)	Sum data
ETX	(1 byte)	03h

6.33.3 Processing Procedure

Boot firmware receives and analyzes a command packet:

- The boot firmware recognizes the start of the command packet by receiving SOH.
  If the boot firmware receives something other than SOH, it waits until it receives SOH.
- If ETX is not added to the received command packet, the boot firmware sends a “Packet error”.
- If the SUM of the received command packet is different from the sum value, the boot firmware sends a “Checksum error”.
- If the LNH and LNL of the received command packet are different from the values specified in the packet format, the boot firmware sends a “Packet error”.
- If the LNH and LNL of the received command packet are different from the values specified in each command, the boot firmware sends a “Packet error”.
- When any of the above errors occurs, the boot firmware does not process and returns to the command waiting state.
  * Memory status does not change before command reception.

When the processing above is successfully completed, boot firmware executes the acceptance analysis:

- If this command cannot be executed in the current DLM state, the boot firmware sends a “Command acceptance error”.
- If device reset is not asserted after the Encrypted data write command execution, the boot firmware sends a “Command acceptance error”.
- If the current Authentication level is AL0, the boot firmware sends a “Secure error”.
- When any of the above errors occurs, the boot firmware does not process and returns to the command waiting state.
  * Memory status does not change before command reception.

When the processing above is successfully completed, boot firmware analyzes the command parameters:

- If OCK is unspecified value, the boot firmware sends a “Parameter error”.
- If VCC is unspecified value, the boot firmware sends a “Parameter error”.
- If LOP exceeds 37000h bytes, the boot firmware sends a “Parameter error”.
- If LOP is 0 byte, the boot firmware sends a “Parameter error”.
- When any of the above errors occurs, the boot firmware does not process and returns to the command waiting state.
  * Memory status does not change before command reception.

When the processing above is successfully completed, the boot firmware initializes hardware modules for accessing external flash memory:

- The boot firmware initializes hardware modules and sends “OK”.

When the processing above is successfully completed, the boot firmware receives and analyzes data packet:

- Boot firmware detects the beginning of a data packet by receiving SOD.
  When boot firmware receives other data than SOD, it discards the data and waits for the next data until SOD is sent.
- When the received data packet does not have ETX, “Packet error” is returned.
- When SUM in the received data packet is different from the value calculated by boot firmware, “Checksum error” is returned.
- When LNH and LNL in the received data packet do not comply with the packet format, “Packet error” is returned.
- When RES in the received data packet is different from defined values, “Packet error” is returned.
- When the total size of received driver code exceeds the specified LOP, the boot firmware sends a “Parameter error”.
- When any of the above errors occurs, the boot firmware does not process and returns to the command waiting state.
  * Memory status does not change before command reception.

When the received data packet is not the last driver code, the boot firmware sends “OK” after writing the driver code to RAM:

- The boot firmware sends “OK” after writing the driver code to RAM.
- The boot firmware receives the next data packet after sending “OK”.

When the received data packet is the last driver code, the boot firmware writes driver code to RAM:

- The boot firmware writes driver code to RAM.

After driver code reception, the boot firmware initializes hardware resources required to access external flash memory:

- The boot firmware calls the Initialize driver to initialize hardware resources required to access external flash memory.
- If Initialize driver returns FFFFFFFFh, the boot firmware sends “Flash initialization error” and waits for the next command.
- When Initialize driver returns 00000000h, the boot firmware sends “OK” and waits for the next command.

6.33.4 Status Information from the Microcontroller

(Listed in descending order of priority.)

Condition	STS	ST2	ADR
The received packet does not have ETX.	Packet error	FFFFFFFFh	FFFFFFFFh
Sum data in the received packet is different from the value calculated by the boot firmware.	Checksum error	FFFFFFFFh	FFFFFFFFh
Packet length in the received packet does not comply with the packet format.	Packet error	FFFFFFFFh	FFFFFFFFh
Packet length in the received packet does not comply with the specifications of this command.	Packet error	FFFFFFFFh	FFFFFFFFh
Executing this command is unavailable in the current DLM state.	Command acceptance error	FFFFFFFFh	FFFFFFFFh
Device reset is not asserted after Encrypted data write command execution.	Command acceptance error	FFFFFFFFh	FFFFFFFFh
The current Authentication level is AL0.	Secure error	FFFFFFFFh	FFFFFFFFh
The specified OCTACLK frequency is an unspecified value.	Parameter error	FFFFFFFFh	FFFFFFFFh
The specified VCC2 voltage is an unspecified value.	Parameter error	FFFFFFFFh	FFFFFFFFh
LOP exceeds 37000h bytes.	Parameter error	FFFFFFFFh	FFFFFFFFh
LOP is 0 byte.	Parameter error	FFFFFFFFh	FFFFFFFFh
The response code of the received data packet is different from the value specified by this command.	Packet error	FFFFFFFFh	FFFFFFFFh
The total length of received data of data packets exceeds the specified LOP.	Parameter error	FFFFFFFFh	FFFFFFFFh
An error occurred while initializing the external flash memory.	Flash initialization error	FFFFFFFFh	FFFFFFFFh
Successful completion.	OK	FFFFFFFFh	FFFFFFFFh

6.33.5 Precautions

(1) This command does not affect SiP Flash area programming. Programming to External flash area and programming to SiP Flash area can be executed independently of each other.

In addition, even after OCTACLK is changed by this command, boot firmware changes OCTACLK to the appropriate frequency before SiP Flash programming and then changes it back after the programming finishes. Therefore, SiP Flash programming and External flash memory programming does not affect the programming speed of each other.

6.33.6 External Flash Memory Access Driver

The specifications of the external flash memory access driver are described below.

Mapping of the driver area:

Offset address from SRAM0 base	Allocated data	Explanation
+08000h - 0803Fh	Initialize driver entry point	Wrap functions to the driver body.
+08040h - 0807Fh	EraseSector driver entry point	
+08080h - 080BFh	EraseChip driver entry point	
+080C0h - 080FFh	ProgramData driver entry point	
+08100h - 3EFFFh	Driver code body + Stack area	Body of the drivers and the stack. Stack pointer is initialized to the area's end address + 1 (= SRAM0 base + 3F000h).
+3F000h - 3FFFFh	Data buffer area	Data buffer is used for write data of ProgramData driver. Boot firmware stores the write data to this area and passes the pointer to this area by the argument of the ProgramData driver. This area is not intended to be written to by drivers.

Arguments and return value are passed in accordance with “ABI for the Arm 32-bit Architecture”

Value	General register
Return value	r0
Argument 1	r0
Argument 2	r1
Argument 3	r2

[Initialize driver]

API specification
Syntax: int32_t R_Flash_Initialize (uint32_t rfu) Arguments: [in] rfu: Unused (reserved for future use) Return value: 00000000h: Operation succeeded FFFFFFFFh: Error occurred
Function explanation
Initialize the external flash memory interface. This driver is called when external flash memory setting command is executed. It is recommended that this driver executes the following functions: - Initialize Octal SPI Peripheral registers. - Initialize Ports setting used for Octal SPI. - Initialize variables that the drivers use. - Return initialization result by the return value. Boot firmware supports reading external flash memory by only Memory mapping mode of Octal SPI. Reading external flash memory by Manual command mode is not supported. Therefore, this driver has to initialize Octal SPI to Memory mapping mode, so that boot firmware can read external flash memory.

[EraseSector driver]

API specification
Syntax: int32_t R_Flash_EraseSector (uint32_t addr) Arguments: [in] addr: Sector address Return value: Other than FFFFFFFFh: Erased size [byte] FFFFFFFFh: Error occurred
Function explanation
Erase flash memory sector. This driver is called when Erase command is executed to external flash area. It is recommended that this driver executes the following functions: - Erase a sector specified by “addr”. - Return the erased size by the return value, or return FFFFFFFFh when erasure fails. Boot firmware repeats calling this driver until “addr” reaches the specified EAD as in the example below: 1. Erase command is executed (SAD = 0x80000000, EAD = 0x8000FFFF). 2. Boot firmware calls EraseSector driver (addr = 0x80000000). 3. EraseSector driver returns 0x00008000. 4. Boot firmware calls EraseSector driver (addr = 0x80008000). 5. EraseSector driver returns 0x00008000. 6. Boot firmware finishes Erase command since addr exceeds the EAD.

[EraseChip driver]

API specification
Syntax: int32_t R_Flash_EraseChip (void) Return value: 00000000h: Operation succeeded FFFFFFFFh: Error occurred
Function explanation
Erase complete flash. This driver is called when Erase command is executed to the entire external flash area (for example: 0x60000000–0x9FFFFFFF for RA8M2 MCU Group). It is recommended that this driver executes the following functions: - Erase the entire flash memory - Return the erasure result This driver is optional for faster full chip erase. Full chip erase is also achievable by using EraseSector driver. However, if the size of the connected external flash memory is the same as the size of the entire external flash area, to execute full chip erase, this driver must be implemented or the Erase command must be executed twice (separately) to avoid calling this driver.

[ProgramData driver]

API specification
Syntax: int32_t R_Flash_ProgramData (uint32_t addr, const void *data, uint32_t cnt) Arguments: [in] addr: Data address. [in] data: Pointer to a buffer containing the data to be programmed to Flash. [in] cnt: Number of data items to program. Return value: Other than FFFFFFFFh: Programmed size [byte] FFFFFFFFh: Error occurred
Function explanation
Program data to flash memory. This driver is called when Write or Encrypted data write command is executed to external flash area. It is recommended that this driver executes the following functions: - Program the data passed by “data”. Program destination address is “addr” and program length is “cnt”. - Return the programmed size by the return value or return FFFFFFFFh when program fails. Boot firmware repeats calling this driver until “addr” reaches the specified EAD as in the example below: 1. Write command is executed (SAD=0x80000000, EAD=0x800007FF). 2. 1st data is sent (data length is 1024 bytes). 3. Boot firmware calls ProgramData driver (addr=0x80000000, cnt=1024). 4. ProgramData driver returns 1024. 5. 2nd data is sent (data length is 1024 bytes). 6. Boot firmware calls ProgramData driver (addr=0x80000400, cnt=1024). 7. ProgramData driver returns 512. 8. Boot firmware calls ProgramData driver (addr=0x80000600, cnt=512). 9. ProgramData driver returns 512. 10. Boot firmware finishes Write command since addr reaches the EAD.

6.33.7 Device State when the Drivers are Called

Table 23 shows the state of the device when external flash memory access drivers are called.

It is necessary for the drivers to initialize only I/O ports and Octal SPI registers to access external flash memories, since boot firmware initializes other HW resources beforehand.

Table 23. Device State when the Drivers are Called

Item	State	Notes for drivers
CPU Security state	Non-Secure	Drivers can access only to Non-Secure resources.
SAU allocation	Following areas are allocated as Non-Secure: • 32008000h – 3203FFFFh • 50000000h – 50FFFFFFh • 80000000h – 9FFFFFFFh Other addresses are allocated as Secure.	Drivers can access only the following: • RAM area for the drivers • Peripherals marked as Non-Secure • External address space allocated for external flash memory
Stack pointer	Main stack pointer for Non-Secure is initialized to 3203F000h.	Drivers do not need to initialize SP.
SRAM	SRAM0 base + 00000h ~ 07FFFh: Marked as Secure. SRAM0 base + 08000h~ : Marked as Non-Secure.	Drivers can use SRAM0 base + 08000h ~ 3FFFFh. (Although 40000h~ is marked as Non-Secure, SAU allocates this address as Secure as described above.) Boot firmware clears this area with 0 before calling Initialize driver. Therefore, drivers can use the stack area soon after being called without initializing.
Clock	Initialized	Drivers do not need to initialize clock registers. OCTACLK depends on the specified OCTACLK frequency by this command.
Octal SPI	Not initialized. Marked as Non-Secure. Module stop has been released.	Initialize driver needs to initialize Octal SPI registers. Initialize driver does not need to release the module stop.
I/O Port	Not initialized except in LVOCR. Marked as Non-Secure (only ports assignable to Octal SPI)	Initialize driver needs to initialize I/O port registers. However, only LVOCR is initialized by boot firmware depending on specified VCC2 voltage by this command.
Other HW resources	Marked as Secure	–

Note: Drivers cannot use interrupts since interrupt registers are not marked as Non-Secure.
Note: Interruptions to boot firmware may occur during driver execution. Therefore, it is recommended to avoid timing-depending processing such as wait-by-nop operation.

For RA8T2 series that do not have SiP Flash, the device state shown below is also applied.

[Only for RA8T2 series w/o SiP Flash)

Item	State
SAU allocation	Following area is also allocated as Non-Secure - 70000000h - 7FFFFFFFh
Octal SPI	OSPI ch1 is also marked as Non-Secure and the module stop has been released.
I/O Port	Ports assignable to Octal SPI ch1 are also marked as Non-Secure.

6.34 Encrypted Data Write Command

This command receives an encrypted image from the host, decrypts the image, and saves the plain-text image on the device.

In addition, this command changes the device to the specified state, PL0 or LCK_BOOT, when saving the data.

Erase processing and write processing of this command are not affected by the block protection settings (BPS, BPS_SEC).

This command require adherence to conditions described in Command List.

Only the following commands are executable after boot firmware sends status OK to the Command packet of this command until device reset is asserted, regardless of the DLM state at the timing:

Executable command	**Note**
User key setting command	–
User key verify command	–
Parameter setting command	Depending on parameter ID (PMID), see the command's specifications for details.
Parameter request command	
ARC configuration setting command	–
ARC configuration request command	–
CRC command	–
Code certificate update command	–
Code certificate check command	–
SiP Flash various setting command	–
SiP Flash various request command	–

6.34.1 Sequence Diagram

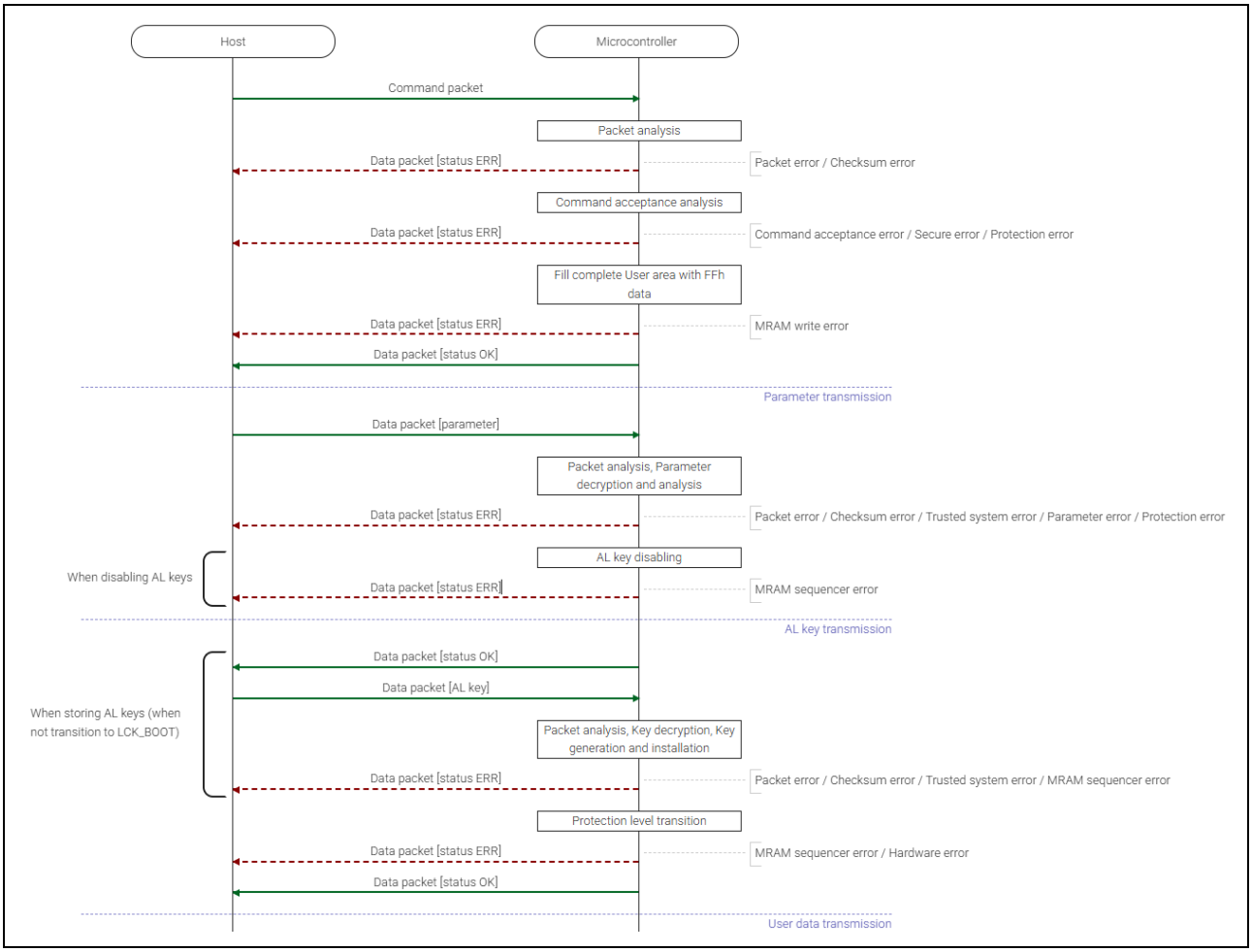


Figure 53. Encrypted Data Write Command Sequence Diagram (Part 1)

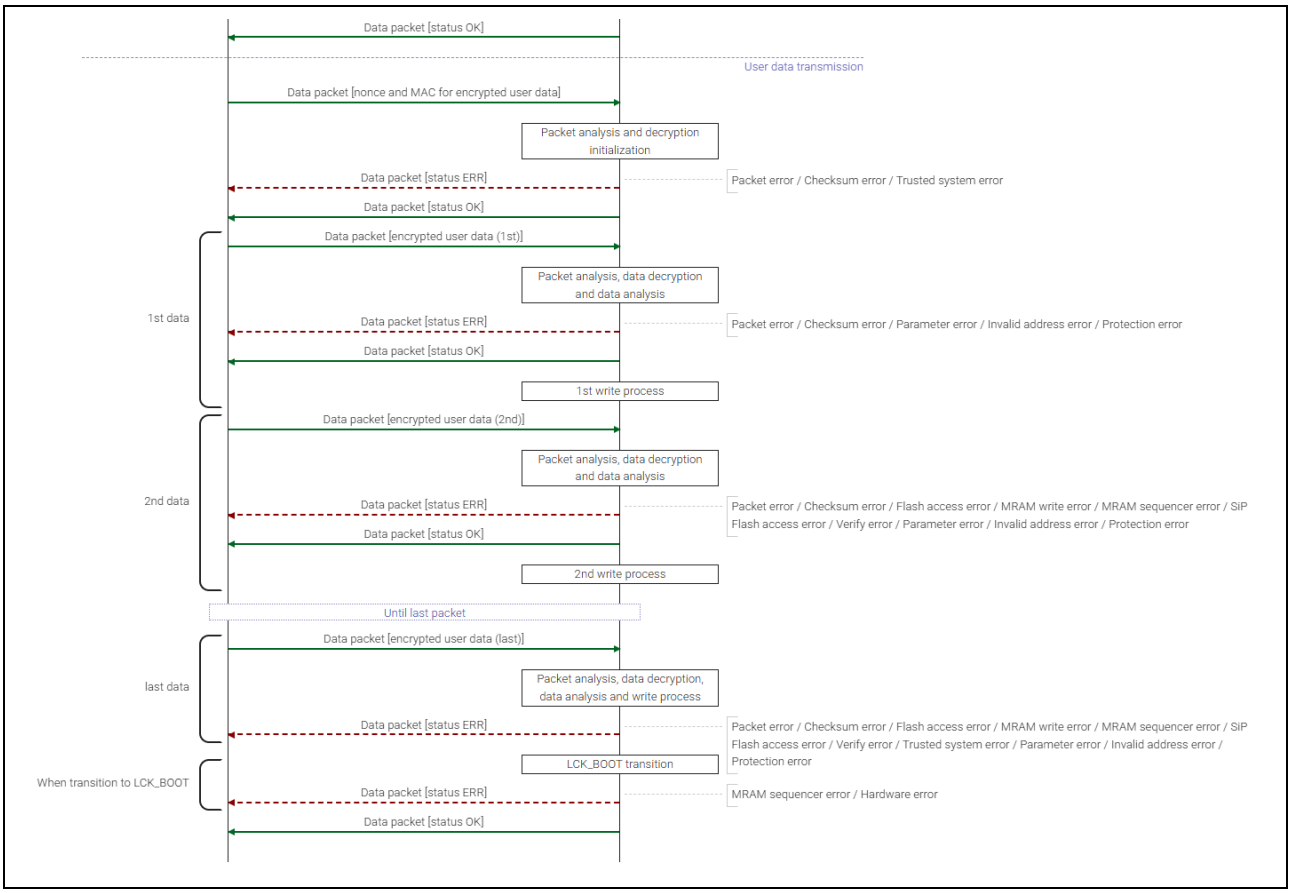


Figure 54. Encrypted Data Write Command Sequence Diagram (Part 2)

6.34.2 Packets

6.34.2.1 Command Packet

SOH	(1 byte)	01h
LNH	(1 byte)	00h
LNL	(1 byte)	55h
CMD	(1 byte)	1Ah (Encrypted data write command)
SKR	(4 bytes)	Shared key ring number
ESKY	(32 bytes)	Wrapped install key (W-UFPK)
IVEC	(16 bytes)	Initialization Vector used for encrypting ENKY
ENKY	(32 bytes)	Encrypted encryption key | MAC. Encryption method is AES128-CBC with CMAC.
SUM	(1 byte)	Sum data
ETX	(1 byte)	03h

6.34.2.2 Data Packet [Parameter]

SOD	(1 byte)	81h
LNH	(1 byte)	00h
LNL	(1 byte)	3Dh
RES	(1 byte)	1Ah (OK)
NCE	(12 bytes)	Nonce used for encrypting parameters. Nonce length is 12 bytes and counter length is 4 bytes.

<table>
<tr>
<td>PRM</td>
<td>(32 bytes)</td>
<td>Encrypted parameters

Encryption method is AES128-CCM mode (NIST SP800-38C)
Data format before encryption:
<table>
<tr><th>1st ~ 4th bytes</th><th>5th byte</th><th>6th byte</th><th>7th byte</th><th>8th byte</th></tr>
<tr><td>LOD</td><td>TRN</td><td>KNUM2</td><td>KNUM1</td><td>(reserved)</td></tr>
<tr><td colspan="5">9th ~ 32th bytes</td></tr>
<tr><td colspan="5">(reserved)</td></tr>
</table>
Parameter details:
<table>
<tr>
<td>LOD (4 bytes)</td>
<td>Total length of “encrypted user data and write address/size”
- This field must be from 00000020h to 00FFFFF0h
- This field must be multiple of 16, encryption block size of AES128

Note that the value of this field is not the same as the data size to be written, but calculated by the sum of the two values below:
- total bytes of data to be written
- 16 byte * N, where N is the total number of ”Data packet [encrypted user data]” to be sent

An example of how to calculate the value of this field is shown below.

[Example of LOD calculation]
Example when writing 524,928-byte write data to the following areas:
Area1: 02000000h ~ 0207FFFFh（524,288 byte）
Area2: 12080000h ~ 120801FFh（512 byte）
Area3: 12080300h ~ 1208037Fh（128 byte）

In this case the number of “Data packet [encrypted user data]” below are sent for sending write data of each area:
Area1: 512 packets (*1)
Area2: 1 packet (*2)
Area3: 1 packet (*2)

The number of Data packet [encrypted user data] is 514 in total, thus LOD is the following value.
LOD = 524,928 + (16 x 514) = 533,152 (=000822A0h)

*1) 512 packets are necessary to send 524,288-byte write data since up to 1024-byte data can be sent by one “Data packet [encrypted user data]”.

*2) One “Data packet [encrypted user data]” is necessary each for write data of these two</td>
</tr>
</table>
</td>
</tr>
</table>

<table>
<tr><td></td><td></td><td>
<table>
<tr><td></td><td>areas; cannot send write data for these two areas by single “Data packet [encrypted user data]”. This is because even when the total size of the write data is 1024 byte or smaller, the write data cannot be sent by one “Data packet [encrypted user data]” when the write destination address is not consecutive.</td></tr>
<tr><td>TRN (1 byte)</td><td>Transition pattern
00h: OEM (PL0)
02h: LCK_BOOT</td></tr>
<tr><td>KNUM2 (1 byte)</td><td>Number of AL2_key to store, specifiable 1 only. This parameter is ignored when transiting to LCK_BOOT (TRN = 02h)</td></tr>
<tr><td>KNUM1 (1 byte)</td><td>Number of AL1_key to store, specifiable from 0 or 1. This parameter is ignored when transiting to LCK_BOOT (TRN = 02h)</td></tr>
<tr><td>reserved</td><td>Reserved data is ignored. Although any value is specifiable, but using random numbers or similarly complex values is recommended for security reasons, rather than using simple value such as 00h or FFh.</td></tr>
</table>
</td></tr>
<tr><td>MAC</td><td>(16 bytes)</td><td>MAC for Encrypted parameters</td></tr>
<tr><td>SUM</td><td>(1 byte)</td><td>Sum data</td></tr>
<tr><td>ETX</td><td>(1 byte)</td><td>03h</td></tr>
</table>

6.34.2.3 Data Packet [AL Key]

<table>
<tr><td>SOD</td><td>(1 byte)</td><td>81h</td></tr>
<tr><td>LNH</td><td>(1 byte)</td><td>N + 29 (Higher 1 byte)</td></tr>
<tr><td>LNL</td><td>(1 byte)</td><td>N + 29 (Lower 1 byte)</td></tr>
<tr><td>RES</td><td>(1 byte)</td><td>1Ah (OK)</td></tr>
<tr><td>NCE</td><td>(12 bytes)</td><td>Nonce used for encrypting AL key data
Nonce length is 12 bytes and counter length is 4 bytes</td></tr>
<tr><td>KEY</td><td>(N byte)</td><td>Encrypted AL key data
Encryption method is AES128-CCM mode (NIST SP800-38C)

AL key data before encryption is made in the steps:
1. Prepare the number of “one AL key data” specified by KNUM2 and KNUM1
2. Combine the prepared “one AL key data’s” in order of AL2 key -> AL1 key

Data format of “one AL key data” is the same as IVEC and ENKY of Key setting command, as shown in two tables below.

“One AL key data” format before encryption:
<table>
<tr><th>1st ~ 16th bytes</th><th>17th ~ 48th bytes</th></tr>
<tr><td>IVEC</td><td>ENKY</td></tr>
</table>
Where IVEC and ENKY means:
<table>
<tr><td>IVEC</td><td>Initialization Vector</td></tr>
<tr><td>ENKY</td><td>Encrypted AL key (0 − 15 bytes) + MAC (16 − 31 bytes)</td></tr>
</table>
As explained above, AL key data is made by combining multiple “one AL key data” in order of AL2 key -> AL1 key. Therefore, for example when KNUM2 = 1 and KNUM1 = 1, AL key data before encryption is as below.

For example: AL key data before encryption when KNUM2 = 1 and KNUM1 = 1:
<table>
<tr><td rowspan="2">AL2 key data</td><td>1st ~ 16th bytes</td><td>17th ~ 48th bytes</td></tr>
<tr><td>IVEC</td><td>ENKY</td></tr>
<tr><td rowspan="2">AL1 key data</td><td>49th ~ 64th bytes</td><td>65th ~ 96th bytes</td></tr>
<tr><td>IVEC</td><td>ENKY</td></tr>
</table>
</td></tr>
<tr><td>MAC</td><td>(16 bytes)</td><td>MAC for Encrypted AL key data</td></tr>
<tr><td>SUM</td><td>(1 byte)</td><td>Sum data</td></tr>
<tr><td>ETX</td><td>(1 byte)</td><td>03h</td></tr>
</table>

N = (KNUM2 + KNUM1) × 48

6.34.2.4 Data Packet [Nonce and MAC for Encrypted User Data]

SOD	(1 byte)	81h
LNH	(1 byte)	00h
LNL	(1 byte)	1Dh
RES	(1 byte)	1Ah (OK)
NCE	(12byte)	Nonce used for encrypting user data. Nonce length is 12 bytes, and counter length is 4 bytes.
MAC	(16 bytes)	MAC for Encrypted user data
SUM	(1 byte)	Sum data
ETX	(1 byte)	03h

6.34.2.5 Data Packet [Encrypted User Data]

<table>
<tr><td>SOD</td><td>(1 byte)</td><td>81h</td></tr>
<tr><td>LNH</td><td>(1 byte)</td><td>N + 1 (Higher 1 byte)</td></tr>
<tr><td>LNL</td><td>(1 byte)</td><td>N + 1 (Lower 1 byte)</td></tr>
<tr><td>RES</td><td>(1 byte)</td><td>1Ah (OK)</td></tr>
<tr><td>DAT</td><td>(N bytes)</td><td>Encrypted user data and write address/size.
Encryption method is AES128-CCM mode (NIST SP800-38C).
Data format before encryption:
<table>
<tr><th>1st ~ 4th bytes</th><th>5th ~ 6th bytes</th><th>7th ~ 16th bytes</th></tr>
<tr><td>SAD</td><td>SIZE</td><td>(Reserved: FFh)</td></tr>
<tr><td colspan="3">17th ~ (n + 16)th bytes (n = 16, 32, 48, ... 1024)</td></tr>
<tr><td colspan="3">User data</td></tr>
</table>
User data and write address/size details:
<table>
<tr><td>SAD - Write address</td><td>Specify write address of the User data/
For example: 02000000h -> 02h, 00h, 00h, 00h</td></tr>
<tr><td>SIZE - Write size</td><td>Specify size of the User data/
For example: 0400h -> 04h, 00h</td></tr>
<tr><td>User data</td><td>Length must be both:
• 1024 bytes of less
• Least common multiple of the following:
- WAU of the write address
- Encryption block size (16 bytes for AES128)</td></tr>
</table>
</td></tr>
<tr><td>SUM</td><td>(1 byte)</td><td>Sum data</td></tr>
<tr><td>ETX</td><td>(1 byte)</td><td>03h</td></tr>
</table>

N = 32, 48, 64, ... 1040

6.34.2.6 Data Packet [Status OK]

SOD	(1 byte)	81h
LNH	(1 byte)	00h
LNL	(1 byte)	0Ah
RES	(1 byte)	1Ah (OK)
STS	(1 byte)	00h (OK)
ST2	(4 bytes)	FFFFFFFFh (unused code)
ADR	(4 bytes)	FFFFFFFFh (unused code)
SUM	(1 byte)	E4h
ETX	(1 byte)	03h

6.34.2.7 Data Packet [Status ERR]

SOD	(1 byte)	81h
LNH	(1 byte)	00h
LNL	(1 byte)	0Ah
RES	(1 byte)	9Ah (ERR)
STS	(1 byte)	Status code
ST2	(4 bytes)	Status details
ADR	(4 bytes)	Failure address
SUM	(1 byte)	Sum data
ETX	(1 byte)	03h

6.34.3 Processing Procedure

Boot firmware receives and analyzes a command packet:

- The boot firmware recognizes the start of the command packet by receiving SOH.
  If the boot firmware receives something other than SOH, it waits until it receives SOH.
- If ETX is not added to the received command packet, the boot firmware sends a “Packet error”.
- If the SUM of the received command packet is different from the sum value, the boot firmware sends a “Checksum error”.
- If the LNH and LNL of the received command packet are different from the values specified in the packet format, the boot firmware sends a “Packet error”.
- If the LNH and LNL of the received command packet are different from the values specified in each command, the boot firmware sends a “Packet error”.
- When any of the above errors occurs, the boot firmware does not process and returns to the command waiting state.

When the processing above is successfully completed, boot firmware executes the acceptance analysis:

- If this command cannot be executed in the current DLM state, the boot firmware sends a “Command acceptance error”.
- If the device reset is not asserted after the Encrypted data write command execution, the boot firmware sends a “Command acceptance error”.
- If the current Authentication level is AL1 or AL0, the boot firmware sends a “Secure error”.
- If any of the following conditions is met, boot firmware sends “Protection error”:
  - SAS.BTFLG is not 1b
  - BANKSEL.BANKSWP[2:0] is not 111b (only for dual mode supported devices)
  - BANKSEL_SEC.BANKSWP[2:0] is not 111b (only for dual mode supported devices)
- If Permanent protected block exists (there is a bit that is 0 in PBPS and PBPS_SEC), the boot firmware sends a “Protection error”.
- When any of the above errors occurs, the boot firmware does not process and returns to the command waiting state.

When the processing above is successfully completed, boot firmware erases the complete User area and Data area:

- If the erasure fails, the boot firmware sends “MRAM write error” and returns to command waiting state.
- If the above error does not occur, the boot firmware sends “OK”.

When the processing above is successfully completed, the boot firmware receives a data packet [parameter] and decrypts and analyzes the parameters:

- Boot firmware receives data packet [parameter].
  * Refer to “Data Packet Reception” below for data packet reception processing.
- Boot firmware decrypts the received parameter.
  If decryption fails, the boot firmware sends a “Trusted system error”.
  However, boot firmware sends nothing and becomes unresponsive if the Trusted system becomes abnormal.
- When decryption is complete, boot firmware checks PRM:
  - Boot firmware sends “Parameter error” if PRM is an unspecified value.
  - Boot firmware sends “Protection error” if LCK_BOOT is specified for TRN when the transition to LCK_BOOT is disabled.
- When any of the above errors occurs, the boot firmware does not process and returns to the command waiting state.

When the processing above is successfully completed, the boot firmware disables AL keys depending on the parameters:

- Boot firmware disables AL keys depending on TRN and KNUM as shown in the following table:

TRN	KNUM2	KNUM1	AL2_key	AL1_key
PL0	1	1	-	-
		0	-	X
LCK_BOOT	ignore	ignore	X	X

X: Disabled
-: Not disabled

- If an error occurs while disabling AL keys, boot firmware sends a “MRAM sequencer error” and returns to the command wait state.

When the processing above is successfully completed, the boot firmware receives and writes AL keys depending on the parameters:

- Boot firmware receives and writes AL keys depending on TRN and KNUM as shown in the following table:

TRN	KNUM2	KNUM1	AL2_key	AL1_key
PL0	1	1	X	X
		0	X	-
LCK_BOOT	ignore	ignore	-	-

X: Receive and write
-: Not receive or write

- Boot firmware sends “OK”.
- Boot firmware receives data packet [AL key].
  * Refer to Data Packet Reception below for data packet reception processing.
- Boot firmware decrypts the received AL key.
  If decryption fails, the boot firmware sends a “Trusted system error”.
  However boot firmware sends nothing and becomes no response if the Trusted system becomes abnormal.
- Boot firmware generates Key index (Wrapped AL key).
  If the generation of Key index (Wrapped AL key) fails, the boot firmware sends a “Trusted system error” and returns to the command waiting state.
  However, boot firmware sends nothing and becomes unresponsive if the Trusted system becomes abnormal.
- Boot firmware writes Key index (Wrapped AL key) to the device.
  If an error occurs while writing Key index (Wrapped AL key), the boot firmware sends a “MRAM sequencer error” and returns to the command wait state.

When the processing above is successfully completed, boot firmware transits the Protection level:

- If an error occurs during Protection level transition, boot firmware returns “MRAM sequencer error” and waits for the next command.
- If the Protection level after the transition is an invalid value, the boot firmware sends a “Hardware error” and becomes unresponsive.
- When Protection level transition is successfully completed, boot firmware sends “OK”.

When the processing above is successfully completed, boot firmware receives data packet [nonce and MAC for encrypted user data] and initializes decryption processing:

- Boot firmware receives data packet [nonce and MAC for encrypted user data].
  * Refer to “Data Packet Reception“ below for data packet reception processing.
- Boot firmware initializes decryption processing.
  If initialization fails, the boot firmware sends a “Trusted system error” and returns to the command waiting state.
  However, boot firmware sends nothing and becomes unresponsive if the Trusted system becomes abnormal.
- When initialization is successfully completed, boot firmware sends “OK”.

When the processing above is successfully completed, boot firmware receives data packet [encrypted user data], decrypts received data, and analyzes decrypted data:

- Boot firmware receives data packet [encrypted user data].
  * Refer to “Data Packet Reception“ below for data packet reception processing.
- Boot firmware decrypts received encrypted user data.
  If decryption fails, the boot firmware sends a “Trusted system error” and returns to the command waiting state.
  However, boot firmware sends nothing and becomes unresponsive if the Trusted system becomes abnormal.
- Boot firmware checks SAD/EAD as described below after decryption of encrypted user data:
  - Boot firmware sends “Parameter error” if:
    - SIZE does not match the length of User data.
    - “SAD ~ (SAD + SIZE - 1)” specifies outside the areas defined in area information.
    - “SAD ~ (SAD + SIZE - 1)” spans different types of area.
    - SAD specifies area whose WAU = 0.
    - SAD is not a multiple of WAU.
    - SAD is not a multiple of encryption block size.
    - SIZE is not a multiple of WAU.
  - Boot firmware sends “Invalid address error” if:
    - The area specified by SAD and SIZE includes address that is inaccessible with the current boundary setting.
  - Boot firmware sends a “Protection error” if:
    - Area specified with SAD and SIZE includes area that the OFSPS is set

When the error above occurs, the boot firmware does not process and returns to the command waiting state.

When the encrypted user data is not the last write data, boot firmware returns “OK” and executes the write processing:

- Boot firmware returns “OK” and executes the write processing.
- If an error occurs while writing the user data, boot firmware receives Data packet, sends “MRAM write error”, “MRAM sequencer error” “Flash access error” or “SiP Flash access error” and returns to the command wait state.
- If the write value and write result do not match at writing to Config area or Config OTP area, boot firmware received Data packet, sends “Verify error” and returns to command waiting state.
- When the write processing is normally finished, the boot firmware receives the next data packet [encrypted user data].

When the encrypted user data is the last write data, the boot firmware executes write processing and then returns a data packet:

- When TRN is “LCK_BOOT”, boot firmware also executes LCK_BOOT transition before data packet transmission.
- Boot firmware does not return “OK” but executes write processing.
- If an error occurs while writing the user data, the boot firmware sends a ”MRAM write error”, “MRAM sequencer error” or “Flash access error” and returns to the command wait state.
- If the write value and write result do not match at writing to Config area or Config OTP area, Boot firmware sends “Verify error” and returns to command waiting state.
- When the write processing is successfully completed and TRN is “LCK_BOOT”, boot firmware executes LCK_BOOT transition.
- If an error occurs during LCK_BOOT transition, boot firmware returns “MRAM sequencer error” and waits for the next command.
- If the DLM state after the transition is an invalid value, the boot firmware sends a “Hardware error” and becomes unresponsive.
- When the processing above is successfully completed, the boot firmware returns “OK” and waits for the next command.

6.34.3.1 Data Packet Reception

Data packet reception processing is described below:

- The boot firmware recognizes the start of the data packet by receiving SOD.
  If the boot firmware receives something other than SOH, it waits until it receives SOD.
- If ETX is not added to the received data packet, the boot firmware sends a “Packet error”.
- If the SUM of the received data packet is different from the sum value, the boot firmware sends a “Checksum error”.
- If the LNH and LNL of the received command packet are different from the values specified in the packet format, the boot firmware sends a “Packet error”.
- When RES in the received data packet is different from defined values by each command, “Packet error” is returned.
- If the LNH and LNL of the received command packet are different from the values specified in each command, the boot firmware sends a “Packet error”.
- If the total received size of “Encrypted user data and write address/size” exceeds LOD, the boot firmware sends a “Parameter error”.
  *) Only when receiving data packet [encrypted user data].
- When any of the above errors occurs, the boot firmware does not process and returns to the command waiting state.

6.34.4 Status Information from the Microcontroller

(Listed in descending order of priority.)

Condition	STS	ST2	ADR
The received packet does not have ETX.	Packet error	FFFFFFFFh	FFFFFFFFh
Sum data in the packet received is different from the value calculated by the boot firmware.	Checksum error	FFFFFFFFh	FFFFFFFFh
Packet length in the received packet does not comply with the packet format.	Packet error	FFFFFFFFh	FFFFFFFFh
Packet length in the received packet does not comply with the specifications of this command.	Packet error	FFFFFFFFh	FFFFFFFFh
Executing this command is unavailable in the current DLM state.	Command acceptance error	FFFFFFFFh	FFFFFFFFh
HUK has been zeroized.	Command acceptance error	FFFFFFFFh	FFFFFFFFh
Device reset is not asserted after Encrypted data write command execution.	Command acceptance error	FFFFFFFFh	FFFFFFFFh
Authentication level is AL1 or AL0.	Secure error	FFFFFFFFh	FFFFFFFFh

Condition	STS	ST2	ADR
SAS.BTFLG is not 1b	Protection error	FFFFFFFFh	FFFFFFFFh
There are blocks protected by permanent block protection (PBPS).	Protection error	FFFFFFFFh	FFFFFFFFh
An error detected when MRAM programming which does not use MACI command	MRAM write error	FFFFFFFFh	FFFFFFFFh
MACI detected an error after the command execution in the disclosed area.	MRAM sequencer error	MRAM status	FFFFFFFFh
The response code of the received data packet is different from the value specified by this command.	Packet error	FFFFFFFFh	FFFFFFFFh
The processing below fails: • Decryption processing • Wrapped AL key generation	Trusted system error	FFFFFFFFh	FFFFFFFFh
Parameter in Encrypted parameters is an unspecified value.	Parameter error	FFFFFFFFh	FFFFFFFFh
“LCK_BOOT” is specified for the Transition pattern when the transition to LCK_BOOT disabled.	Protection error	FFFFFFFFh	FFFFFFFFh
Protection level or DLM state is abnormal.	Hardware error	FFFFFFFFh	FFFFFFFFh
Any condition of the followings is met: - Total received size of “Encrypted user data and write address/size” exceeds LOD - SIZE does not match the length of User data - “SAD ~ (SAD + SIZE - 1)” specifies outside the areas defined in area information - “SAD ~ (SAD + SIZE - 1)” spans different Kind of the area - SAD specifies area whose WAU = 0 - SAD is not multiple of WAU - SAD is not multiple of encryption block size (*) - SIZE is not multiple of WAU *) 16 byte for AES128	Parameter error	FFFFFFFFh	FFFFFFFFh
“SAD ~ (SAD + SIZE - 1)” contains addresses that are inaccessible with the current boundary settings.	Invalid address error	FFFFFFFFh	FFFFFFFFh
“SAD ~ (SAD + SIZE - 1)” contains an area where the OFSPS is set.	Protection error	FFFFFFFFh	FFFFFFFFh
The written value and the write result do not match at writing at Config area or Config OTP area.	Verify error	FFFFFFFFh	FFFFFFFFh
An error occurred while programming SiP Flash.	SiP Flash access error	SiP Flash status	FFFFFFFFh
An error occurred in the External flash memory access driver.	Flash access error	FFFFFFFFh	FFFFFFFFh
Successful completion.	OK	FFFFFFFFh	FFFFFFFFh

6.34.5 Precautions

(1) This command becomes inexecutable after permanent block protection is set.

(2) This command becomes inexecutable if SAS.BTFLG=0b and SAS.FSPR=0b are set:

- SAS.BTFLG = 0b
- BANKSEL.BANKSWP[2:0] ≠ 111b (only for dual mode supported devices)
- BANKSEL_SEC.BANKSWP[2:0] ≠ 111b (only for dual mode supported devices)

(3) If permanent block protection or OFSPS in the Config OTP area is written before the protected area, this command abnormally finishes at writing of the protected area.

To avoid this, Data packet [encrypted user data] for the protected areas must be sent earlier than ones for permanent block protection area or OFSPS area.

(4) Do not set permanent block protection of the area where user keys are to be written when the User key setting command will be used.

Do not set permanent block protection of the area where the Code certificate is to be written when the Code certificate update command will be used.

If they are set, both commands become inexecutable due to Protection errors.

(5) When accessing the external flash area, the driver function for access is called, so send the driver code with the “External flash memory setting command” in advance. This command is called the “Program Data driver”.

Also, access to addresses to which external flash memory is not allocated is not guaranteed.

(6) Config OTP area cannot be written back to 1 after 0 is written.

For Config OTP area which has ECC, ECC bits cannot be written back from 0 to 1. Therefore, note that rewriting is not possible even when the data bits are only to be changed from 1 to 0. However, rewriting is possible as an exception in the following cases:

- Rewriting the same value
- The data already written is all“F”

(7) The theoretical maximum data size that can be written with this command is 16,519,088 bytes (*1), because the maximum value of LOD is 00FFFFF0h (16,777,200) as described in the explanation of LOD.

However, because the write data cannot be sent by one “Data packet [encrypted user data]” when the write destination address is not consecutive as explained, therefore the actual maximum data size that can be written with this command depends on the area information of the product and the write destination address of the data to be written.

*1) 16,519,088-byte data can be sent by 16132 packets of “Data packet [encrypted user data]”; of which the 16131 packets send 1024-byte write data each, and the remaining one packet sends 944-byte write data. Accordingly, when the write data size is 16,519,088-byte,
LOD = (1024×16131) + 944 + (16×16132) = 16,777,200 = 00FFFFF0h = Maximum.

6.34.6 Device State after Command Execution

Table 24 shows the state of the device after this command is executed.

Table 24. Device States after Encrypted Data Write Command Execution

<table>
<tr><th colspan="2" rowspan="2">Command finish timing</th><th colspan="5">Device state</th></tr>
<tr><th>User area</th><th>AL key</th><th>Protection level</th><th>Area specified by SAD</th><th>DLM</th></tr>
<tr><td rowspan="11">Fails at</td><td>Command acceptance analysis</td><td colspan="5">No change</td></tr>
<tr><td>Erasing complete User area</td><td>Undefined</td><td colspan="4" rowspan="2">No change</td></tr>
<tr><td>Decrypting or analyzing the parameter</td><td rowspan="6">Filled with FFh data</td></tr>
<tr><td>Disabling AL key</td><td>Undefined</td><td colspan="3" rowspan="2">No change</td></tr>
<tr><td>Decrypting, generating or writing AL key</td><td>Undefined or disabled depending on KNUM2, KNUM1 and TRN</td></tr>
<tr><td>Transiting Protection level</td><td rowspan="4">Written or disabled depending on KNUM2, KNUM1 and TRN</td><td>Undefined</td><td colspan="2">No change</td></tr>
<tr><td>Initializing decryption</td><td rowspan="3">PL0</td></tr>
<tr><td>Decrypting, checking, or writing user data</td><td>Filled with FFh data (areas specified by SAD are undefined)</td><td>Undefined</td><td>No change</td></tr>
<tr><td>Transiting to LCK_BOOT</td><td>User data written</td><td>User data written</td><td>Undefined</td></tr>
<tr><td colspan="2">Successful completion</td><td colspan="3"></td><td>LCK_BOOT depending on TRN</td></tr>
</table>

6.34.7 DLM State Transitions

The following diagram shows the DLM states that can be transitioned by this command.

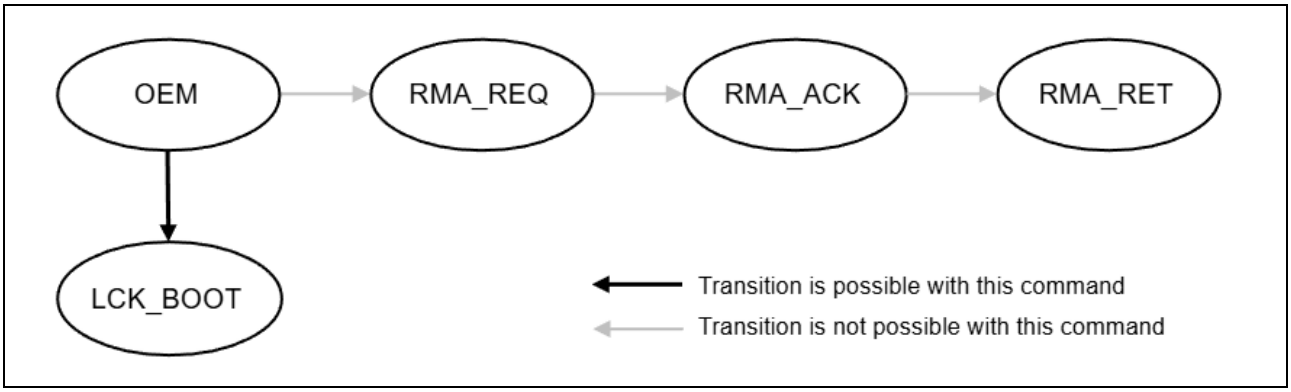


Figure 55. Valid DLM State Transitions for Encrypted Data Write Command

7. Flow Examples

7.1 Beginning Communication

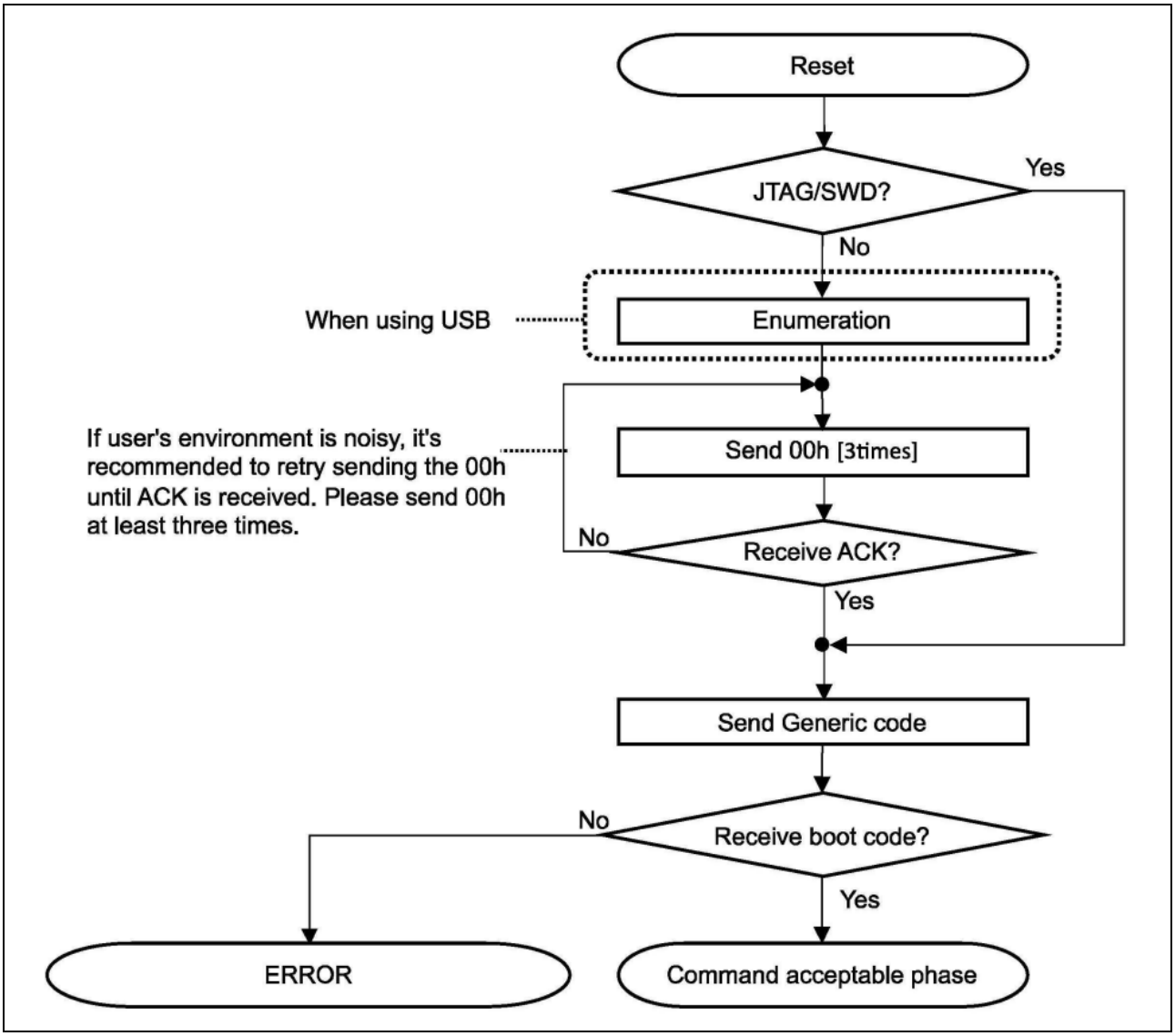


Figure 56. Beginning Communication

7.2 Acquisition of Device Information/Baudrate Settings

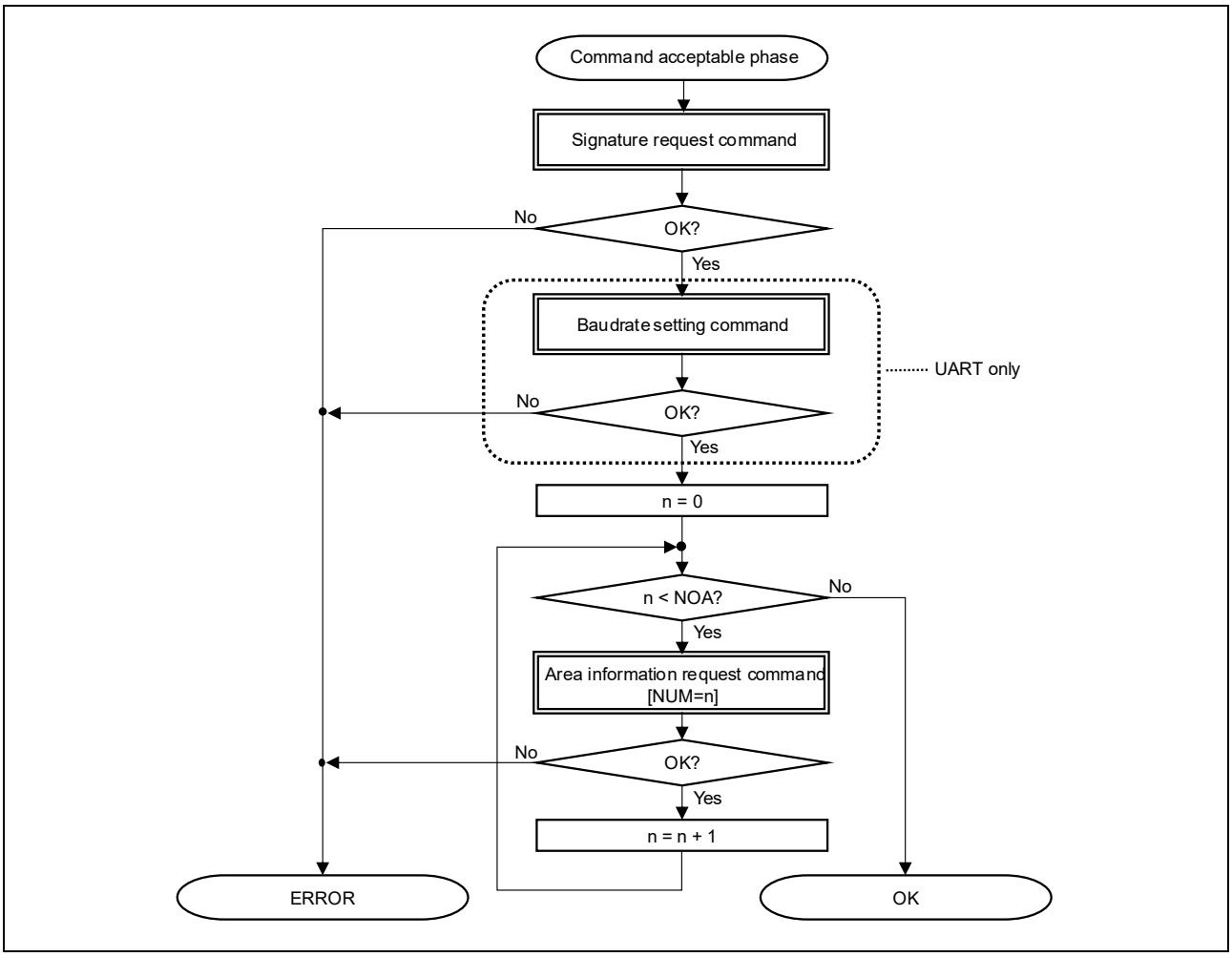


Figure 57. Acquisition of Device Information / Baudrate Settings

7.3 Transiting DLM State

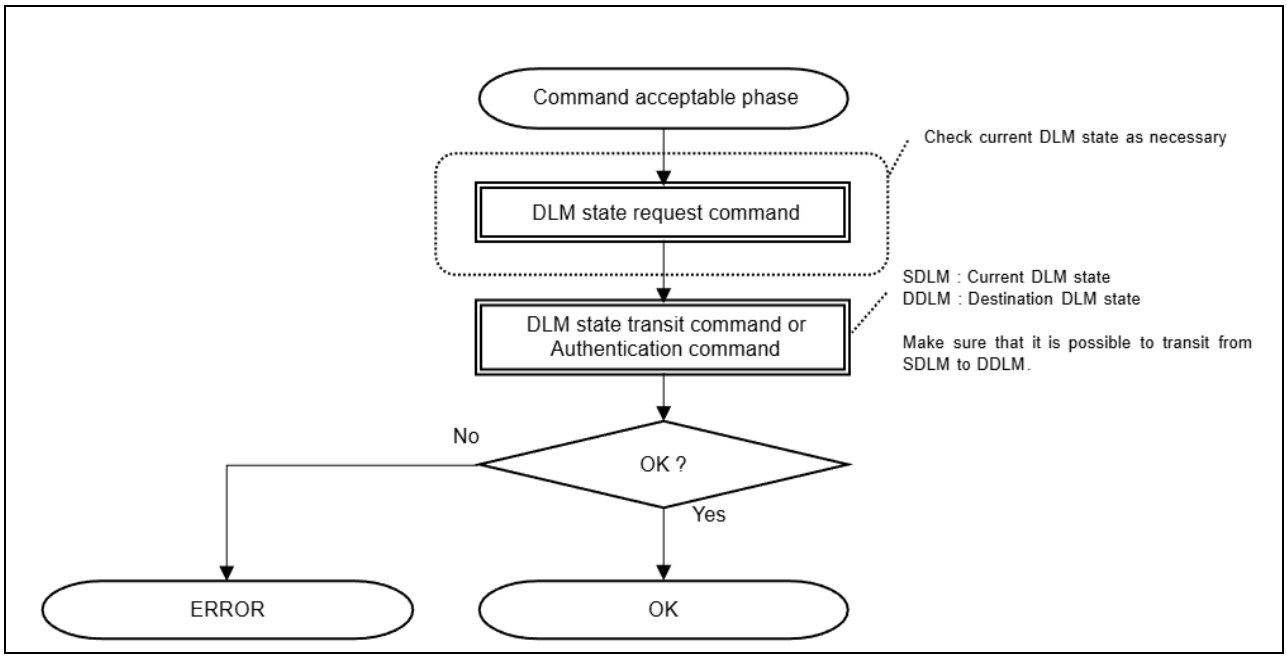


Figure 58. Transiting DLM State

7.4 Transiting Protection Level

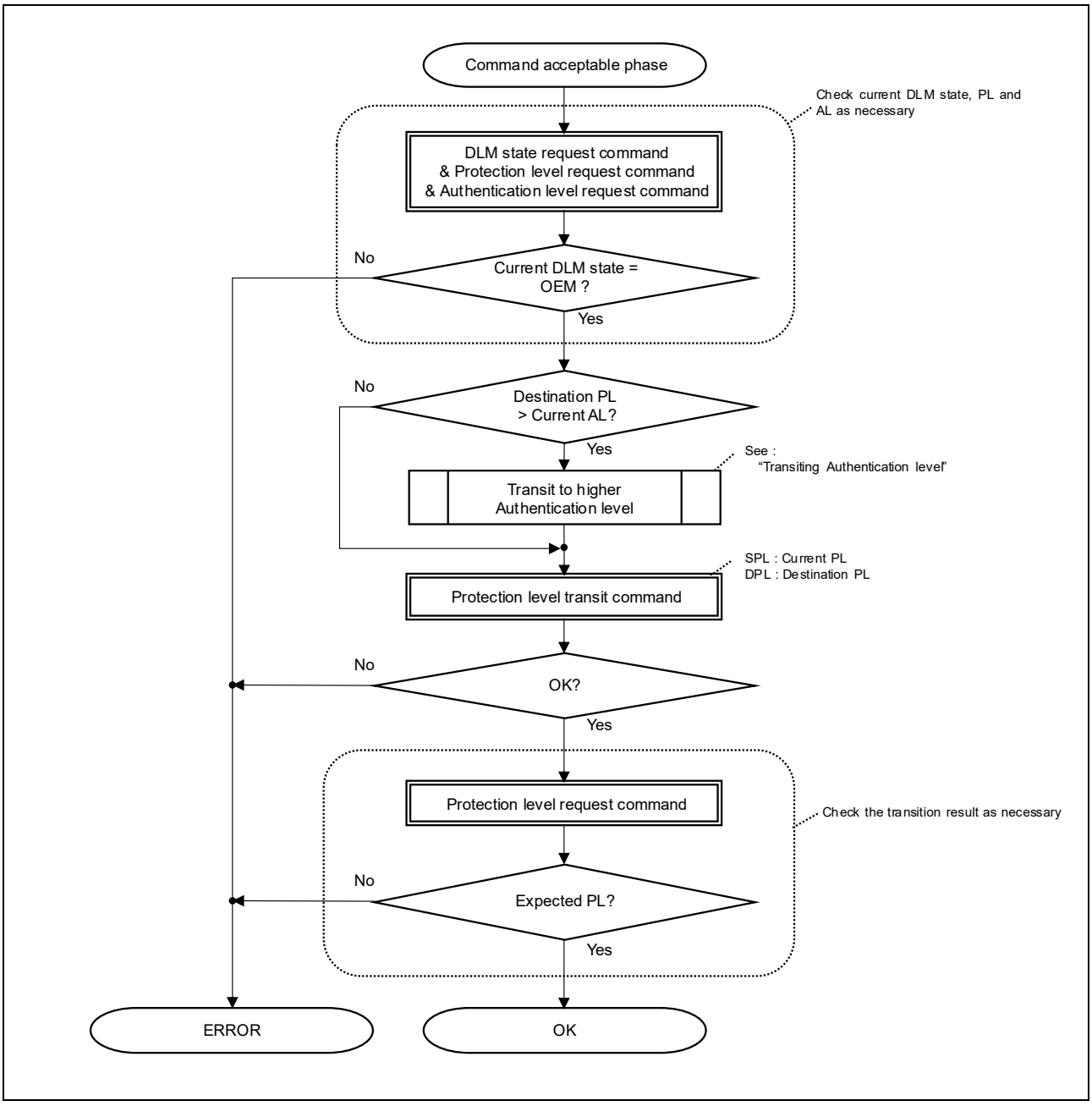


Figure 59. Transiting Protection Level

7.5 Transiting Authentication Level

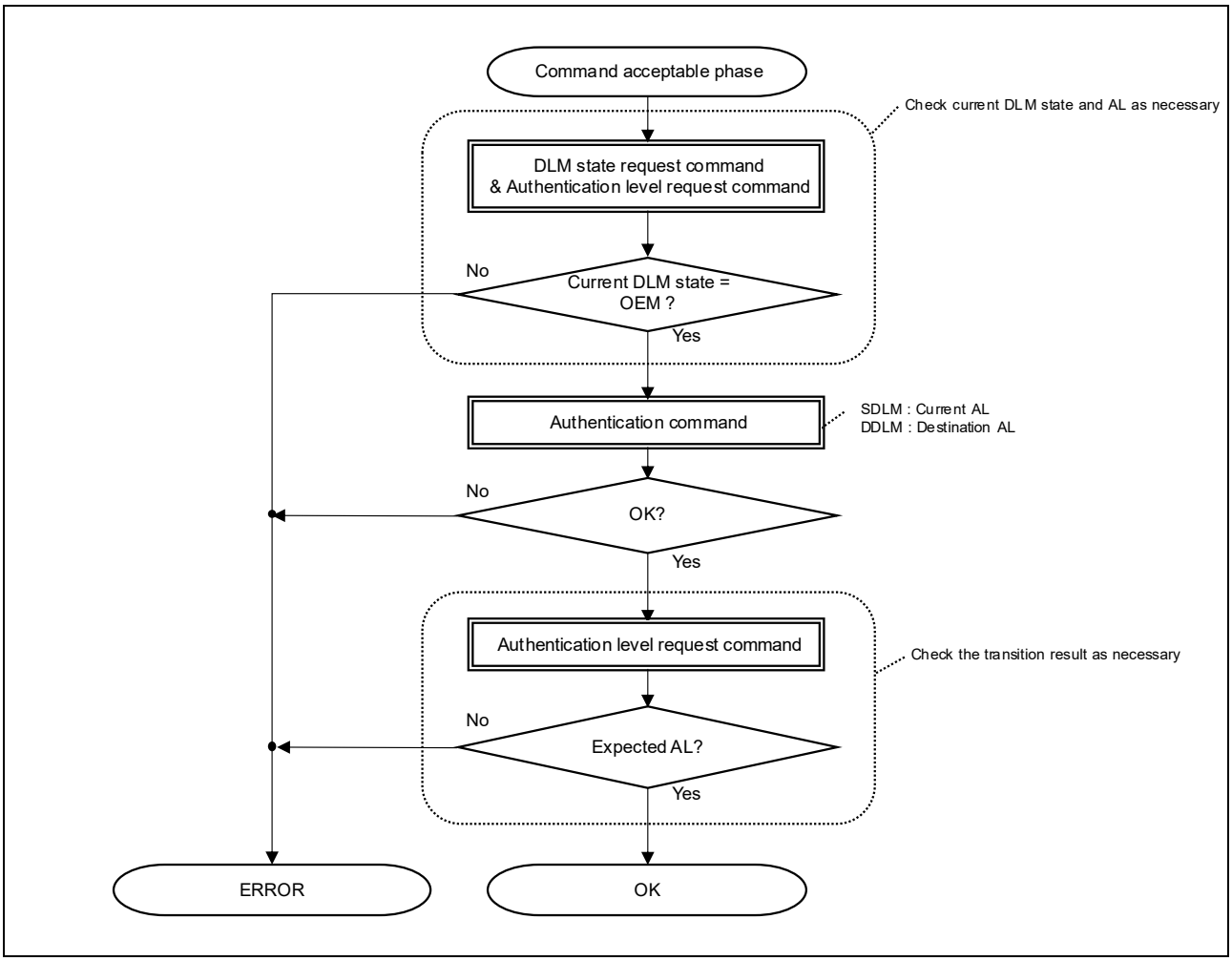


Figure 60. Transiting Authentication Level

7.6 Data Programming

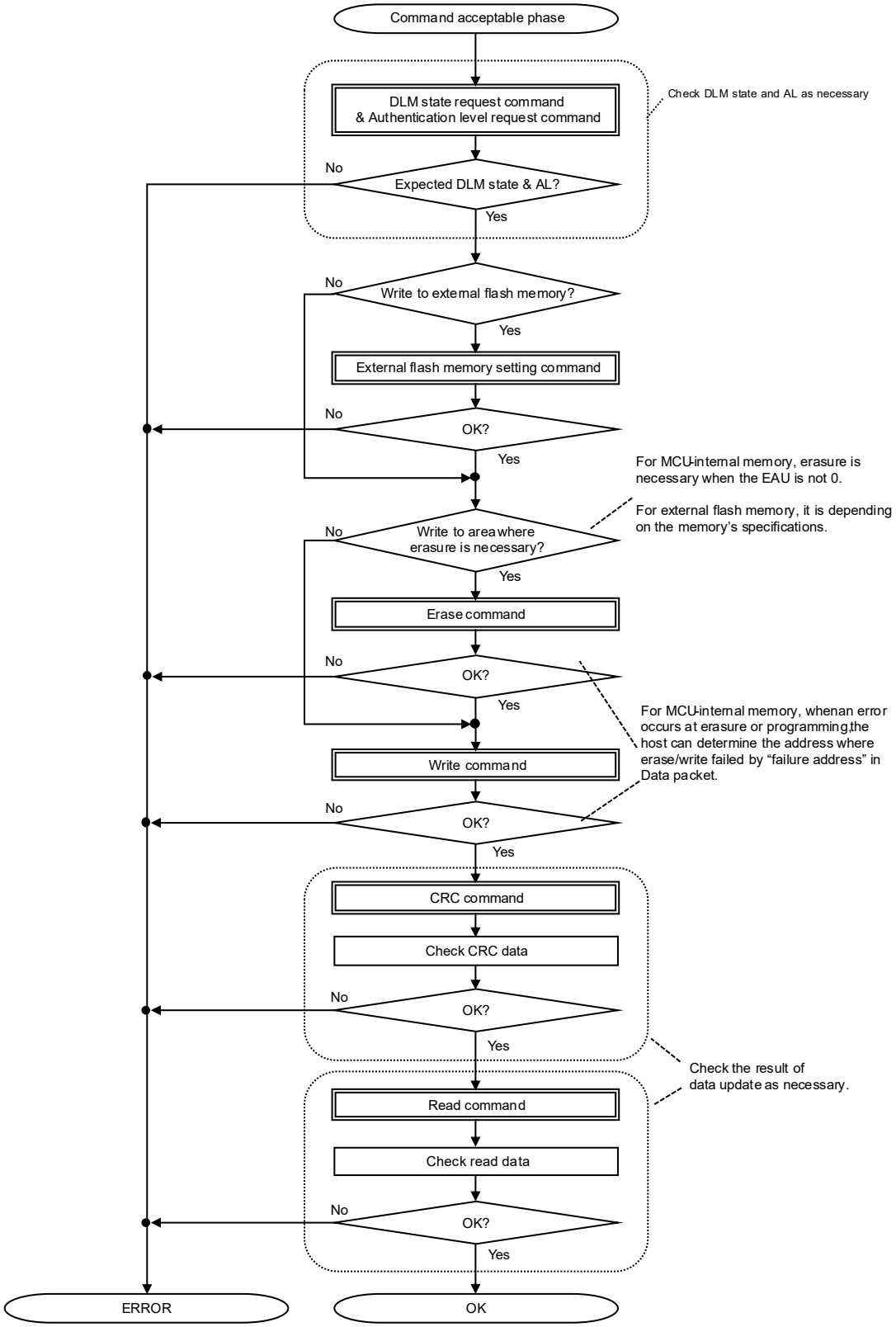


Figure 61. Data Programming

7.7 Encrypted Data Programming

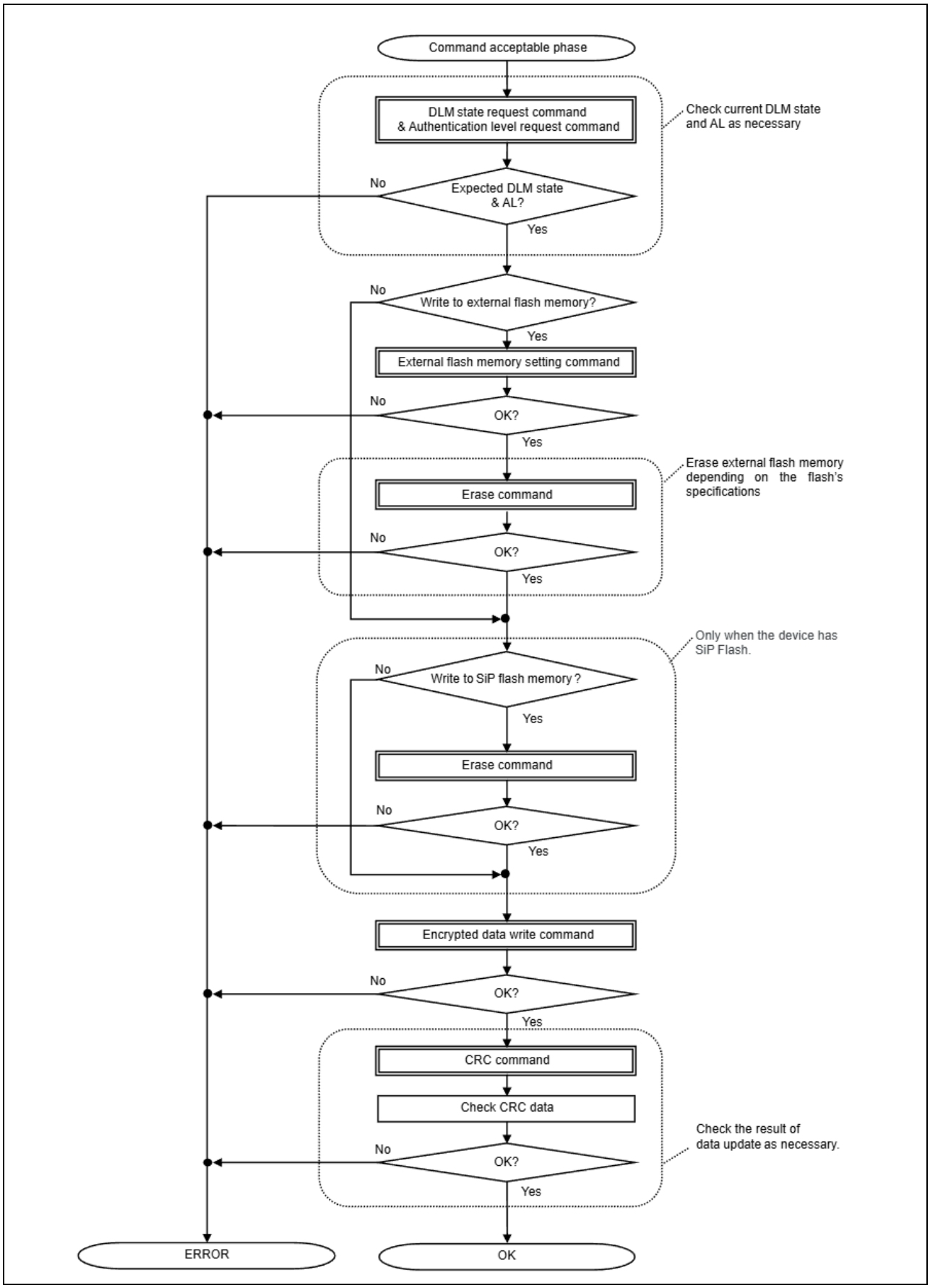


Figure 62. Encrypted Data Programming

7.8 Initialize Memory

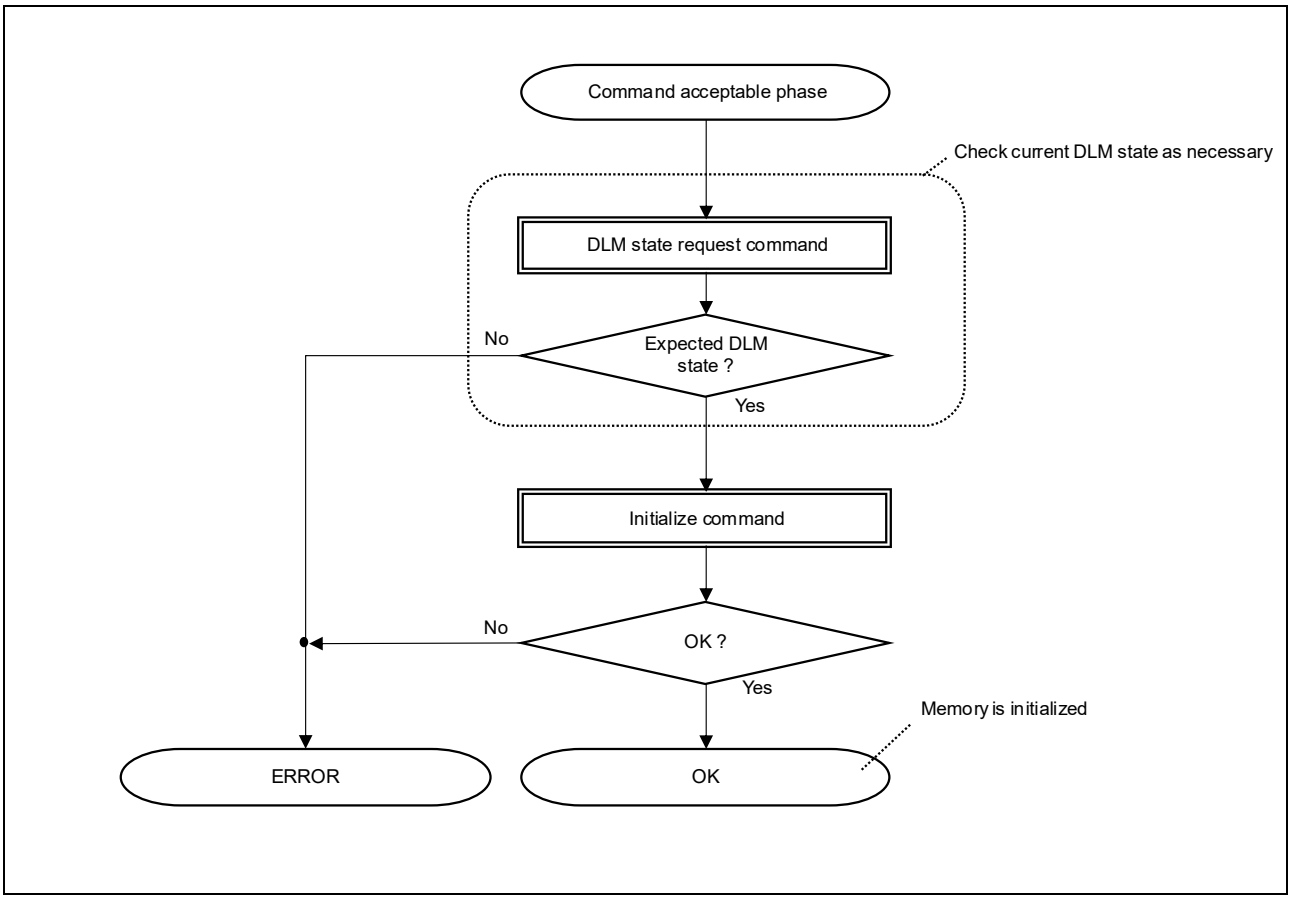


Figure 63. Initializing Memory

7.9 Storing Keys

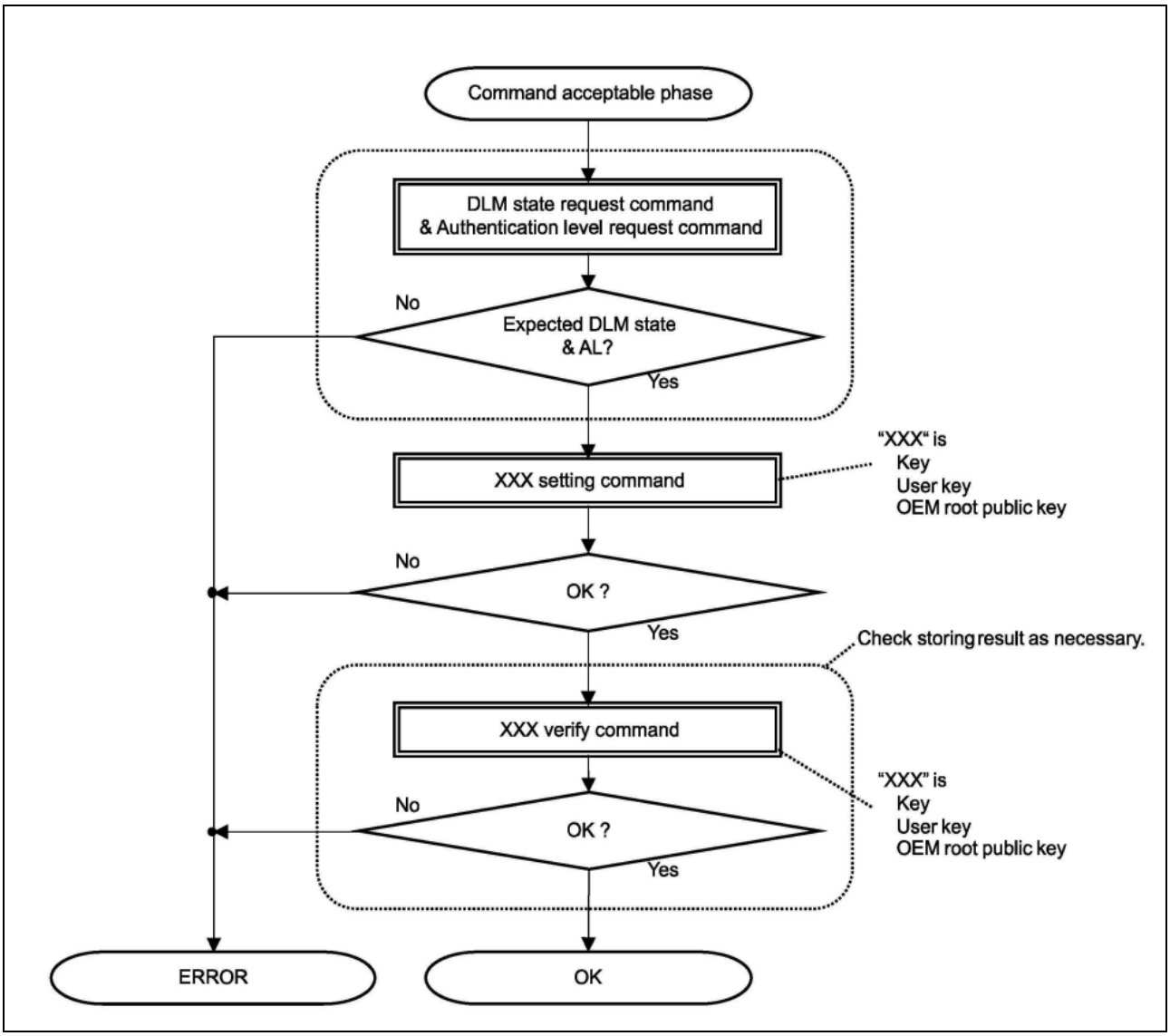


Figure 64. Storing Keys

7.10 Updating Boundary, Parameter, ARC configuration or SiP Flash various setting

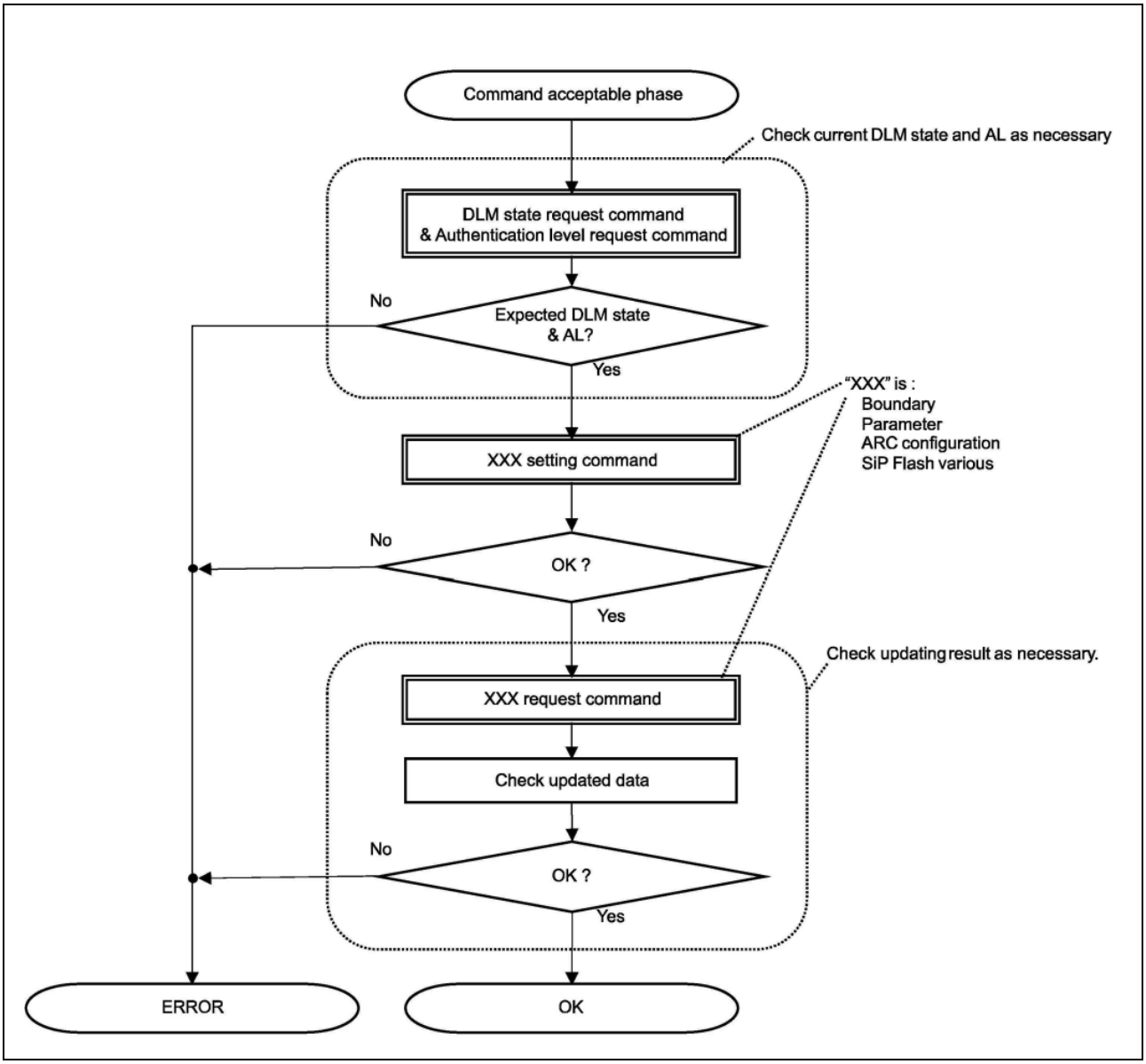


Figure 65. Updating Boundary, Parameter, ARC Configuration or SiP Flash Various Setting

7.11 Storing Code Certificate

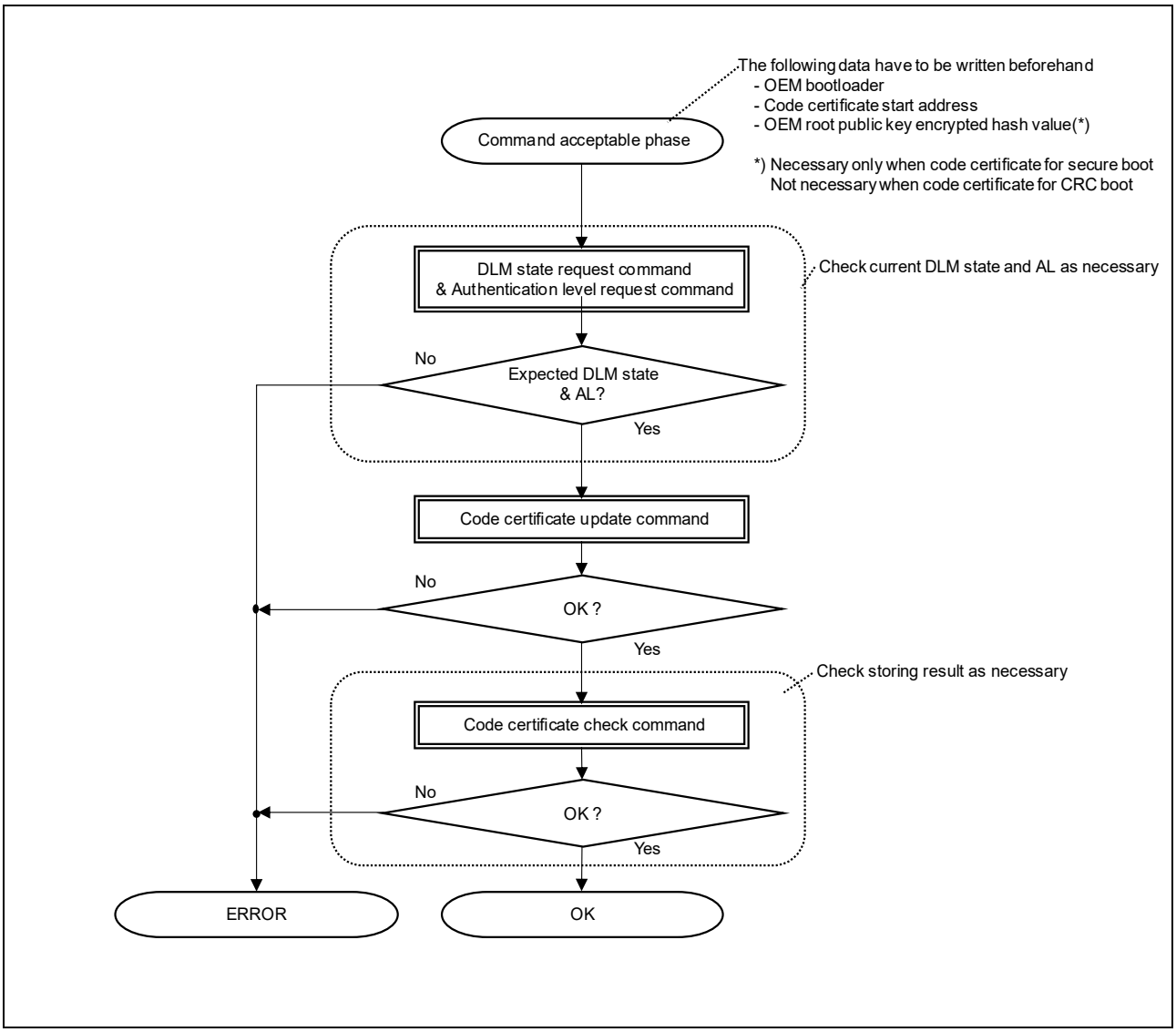


Figure 66. Storing Code Certificate

7.12 Downloading Whole Image

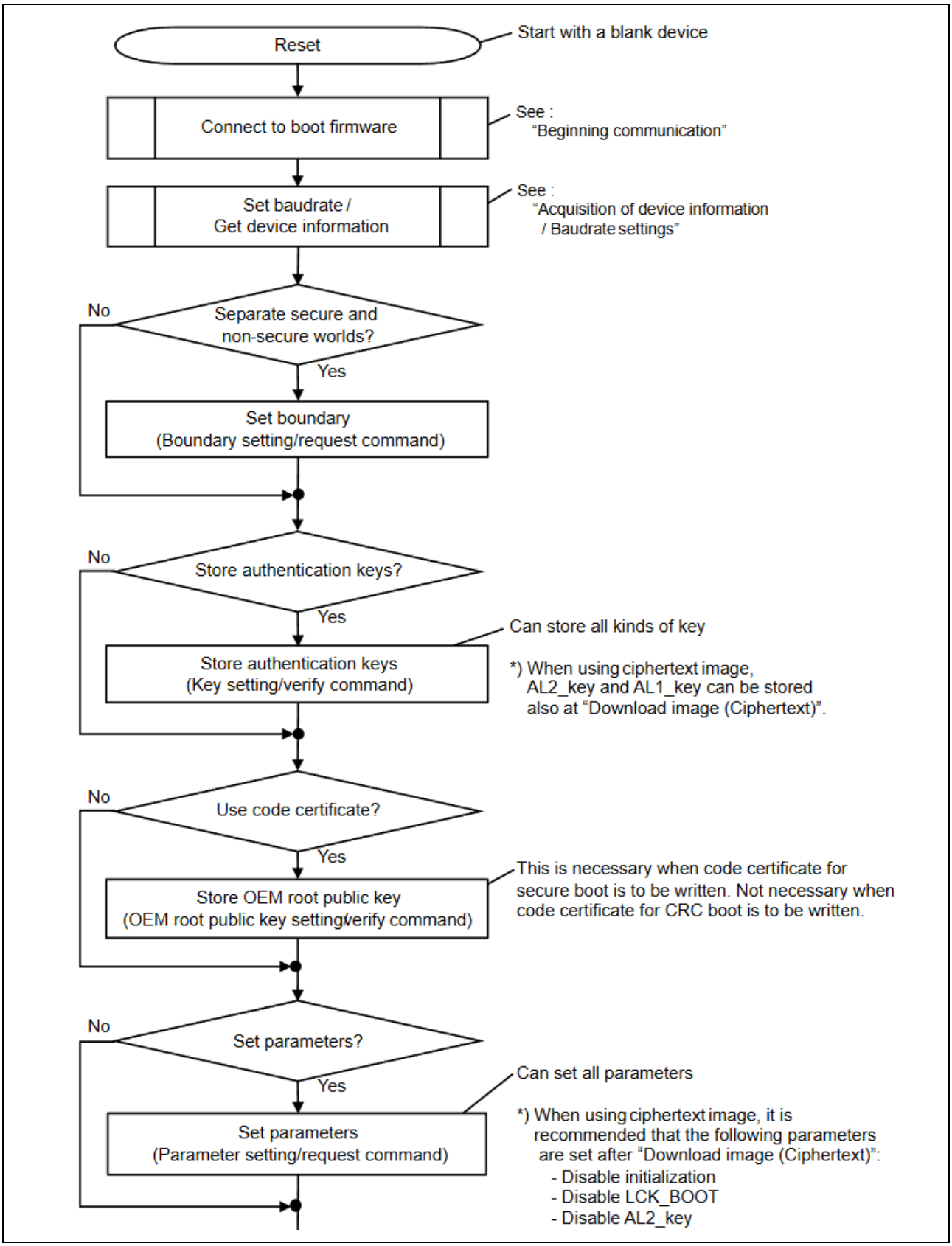


Figure 67. Downloading Whole Image (Part 1)

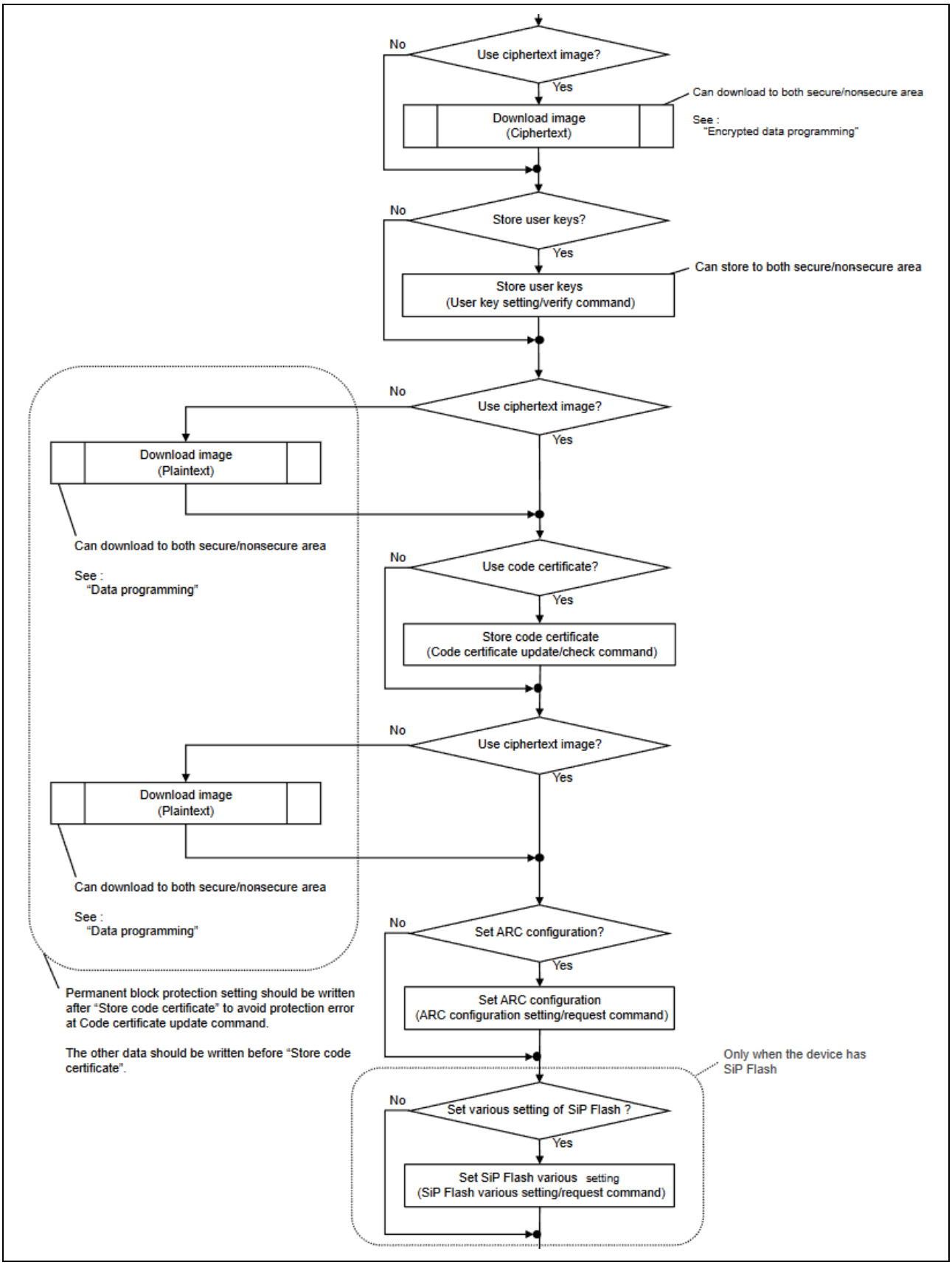


Figure 68. Downloading Whole Image (Part 2)

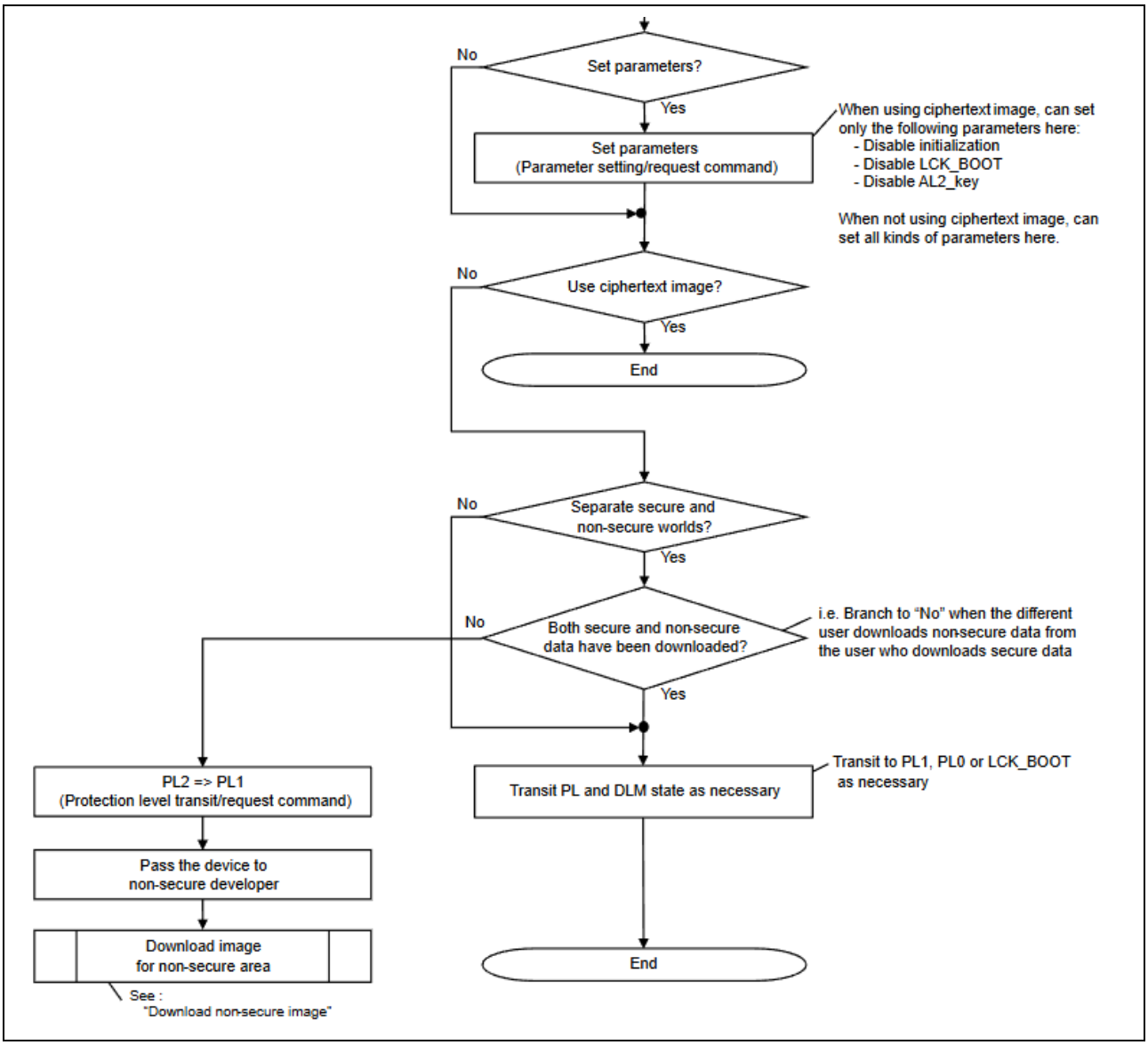


Figure 69. Downloading Whole Image (Part 3)

7.13 Downloading Non-secure Image

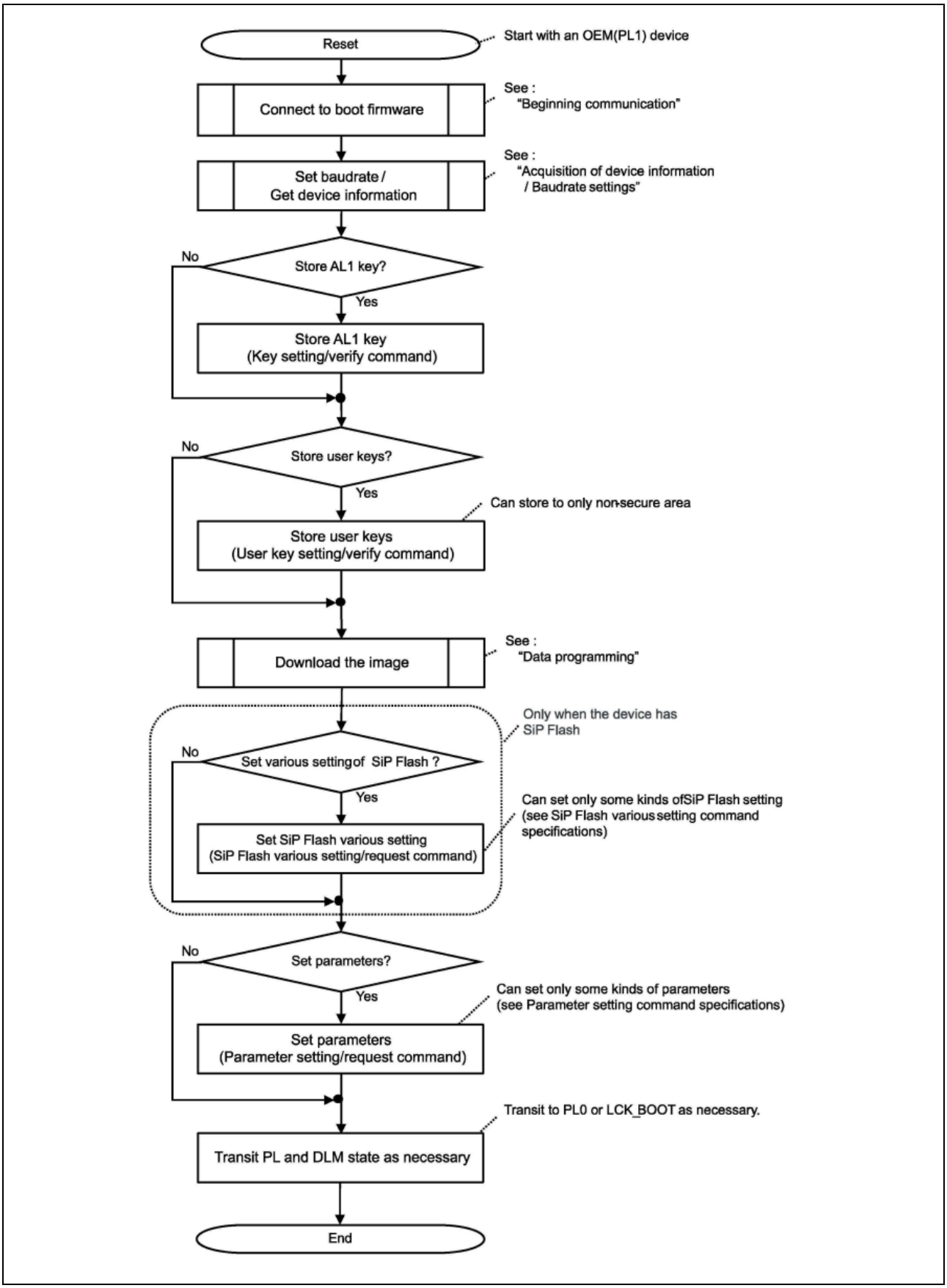


Figure 70. Downloading Non-secure Image

7.14 Command Cancel

For commands that continuously send and receive packets, you can end the command by intentionally sending an error packet and return to the Command acceptable phase.

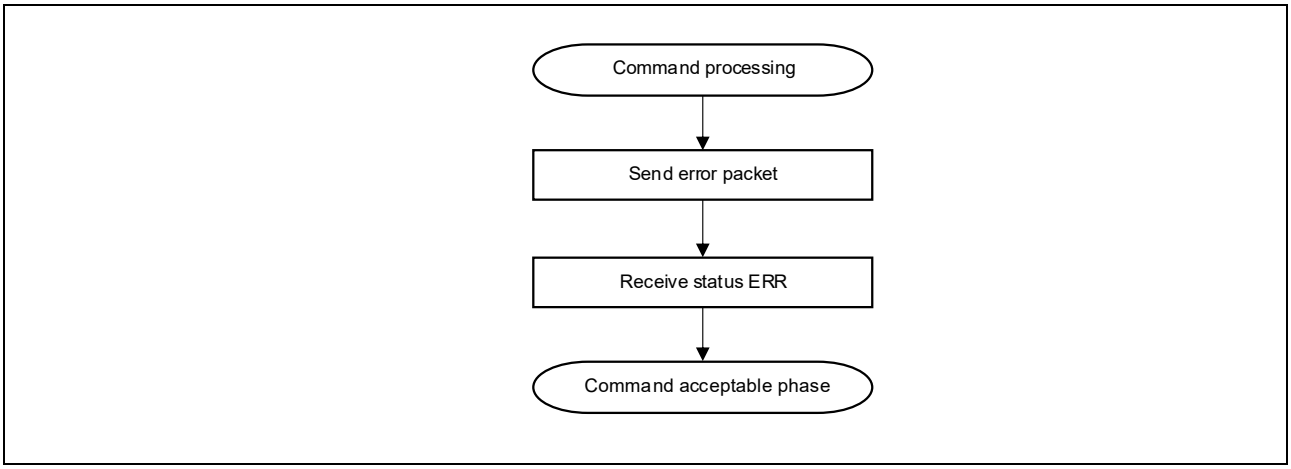


Figure 71. Command Cancel

e.g.) Error packets to end the command.

Command	When to send error packets	Example of the error packet
Authentication command	Data packet [Response value or Authentication code]	
Key setting command	Data packet [key data]	
User key setting command	Data packet [key data]	
Write command	Data packet [write data]	
Read command	Data packet [status OK]	
OEM root public key setting command	Data packet [key data]	
Code certificate update command	Data packet [Key/Code certificate data]	
External flash memory setting command	Data packet [driver code]	
Encrypted data write command	Data packet [Key/encrypted user data]	

SOD	(1 byte)	81h
LNH	(1 byte)	00h
LNL	(1 byte)	01h
RES	(1 byte)	FFh (ERR)
SUM	(1 byte)	00h
ETX	(1 byte)	03h

8. AC Characteristics

8.1.1 Communication Setting Phase

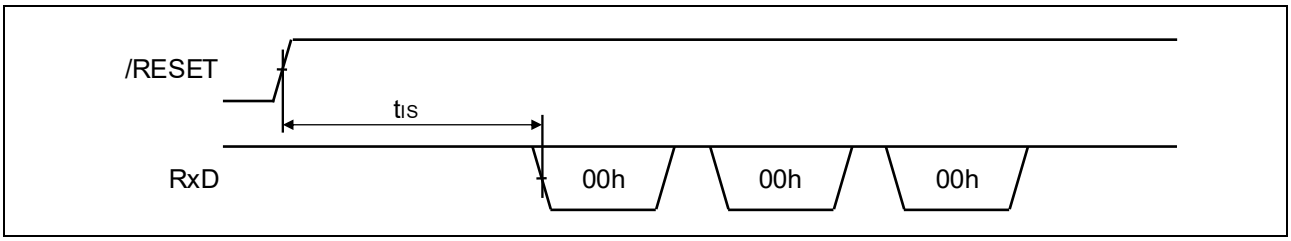


Figure 72. 2-wire UART Communication

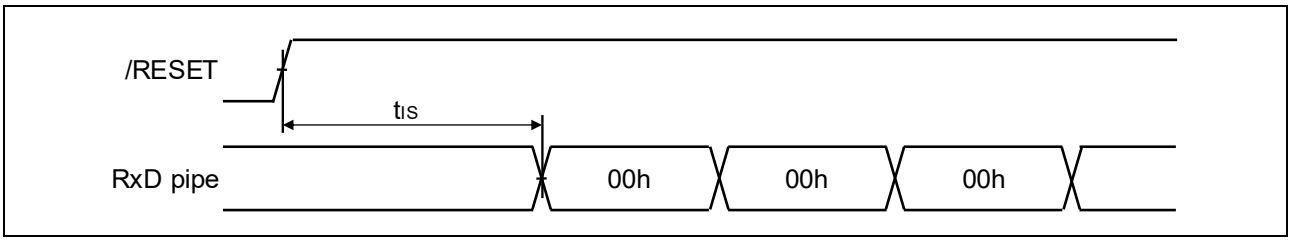


Figure 73. USB Communication

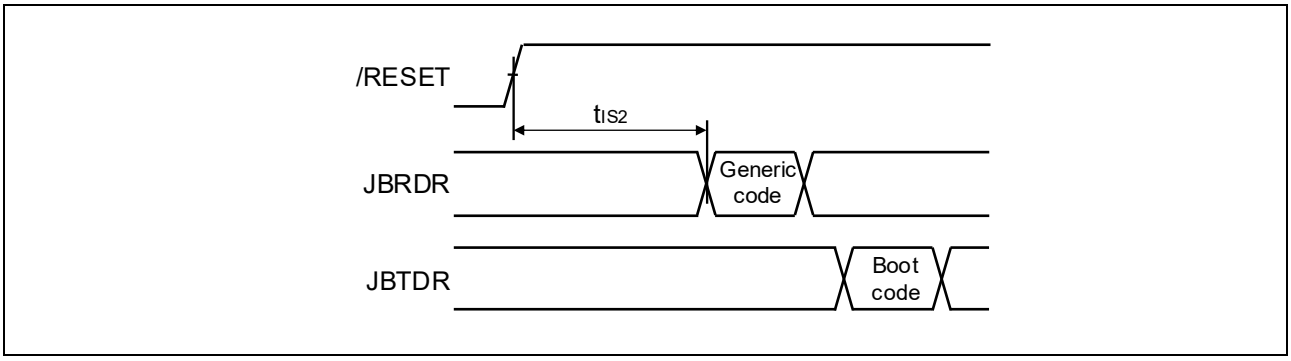


Figure 74. JTAG/SWD Communication Initial Setting Time

Parameter	Symbol	Min	Typ	Max	Unit
Initial setting time (when using Main-OSC)	tIS	-	-	194 (*1)	ms
Initial setting time (when using HOCO)	tIS	-	-	2828 (*1)	ms
Initial setting time 2	tIS2	-	-	136 (*1)	ms
Initial setting time (when using Main-OSC)	tIS	-	-	TBD (*2)	ms
Initial setting time (when using HOCO)	tIS	-	-	TBD (*2)	ms
Initial setting time 2	tIS2	-	-	TBD (*2)	ms

*1: Standard Product
*2: SiP Product

8.1.2 DLM State Transit Command

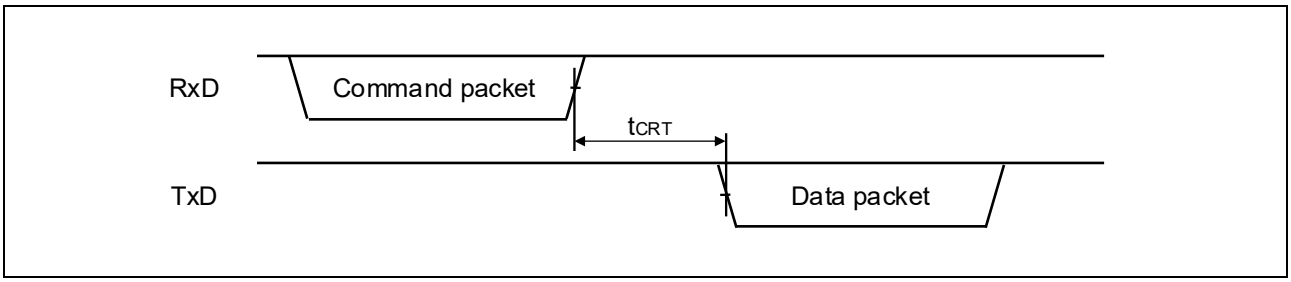


Figure 75. DLM State Transit Command

Parameter	Symbol	Min	Typ	Max	Unit
Command response time	tCRT	-	-	3 (*1)	s
Command response time	tCRT	-	-	TBD (*2)	s

*1: Standard Product
*2: SiP Product

8.1.3 DLM State Request Command

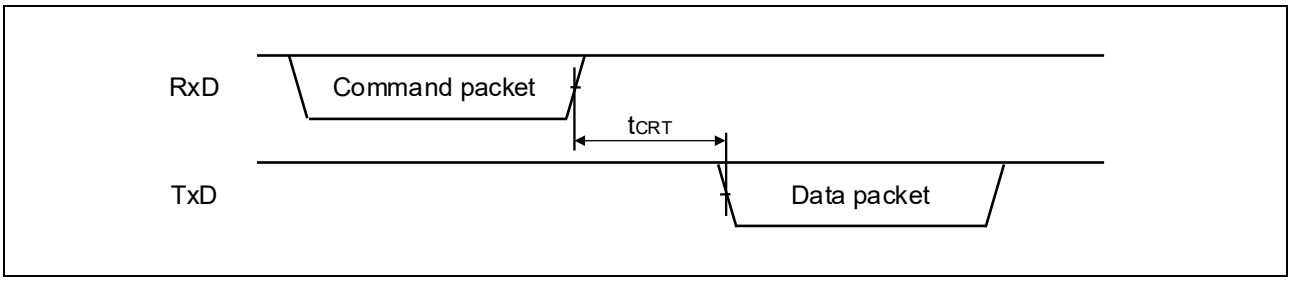


Figure 76. DLM State Request Command

Parameter	Symbol	Min	Typ	Max	Unit
Command response time	tCRT	-	-	3	s

8.1.4 Protection Level Transit Command

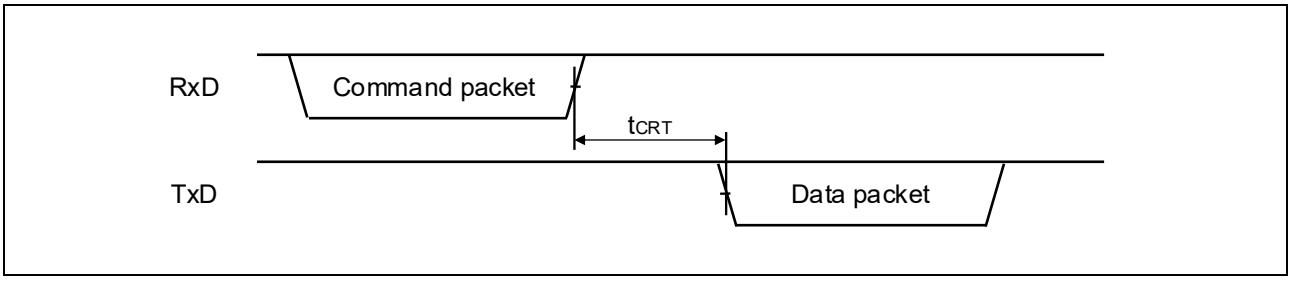


Figure 77. Protection Level Transit Command

Parameter	Symbol	Min	Typ	Max	Unit
Command response time	tCRT	-	-	3 (*1)	s
Command response time	tCRT	-	-	TBD (*2)	s

8.1.5 Protection Level Request Command

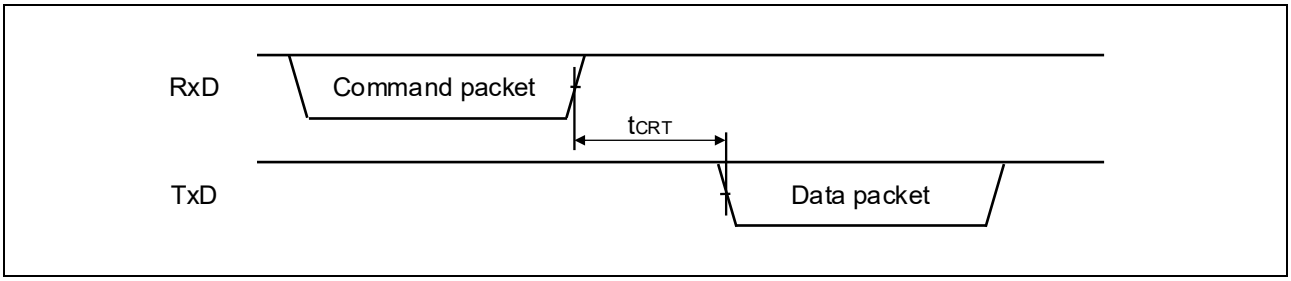


Figure 78. Protection Level Request Command

Parameter	Symbol	Min	Typ	Max	Unit
Command response time	tCRT	-	-	3	s

8.1.6 Authentication Level Request Command

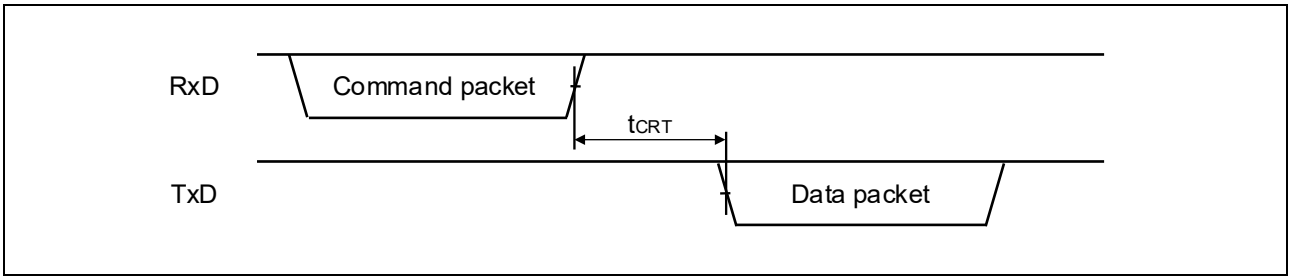


Figure 79. Authentication Level Request Command

Parameter	Symbol	Min	Typ	Max	Unit
Command response time	tCRT	-	-	3	s

8.1.7 Authentication Command

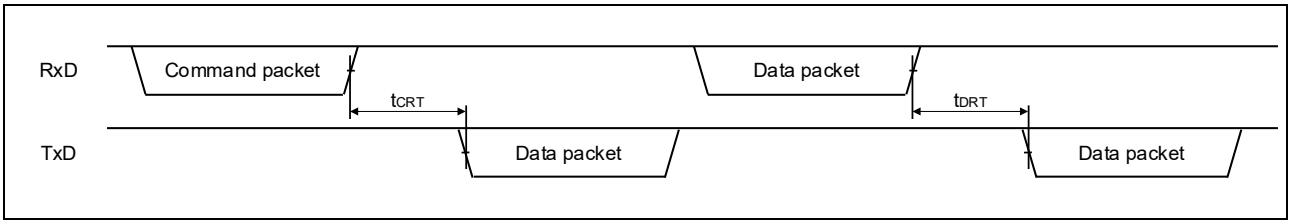


Figure 80. Authentication Command

Parameter	Symbol	Min	Typ	Max	Unit
Command response time	tCRT	-	-	3 (*1)	s
Command response time	tCRT	-	-	TBD (*2)	s
Data response time	tDRT	-	-	60 (*1)	s
Data response time	tDRT	-	-	TBD (*2)	s

*1: Standard Product
*2: SiP Product

8.1.8 Key Setting Command

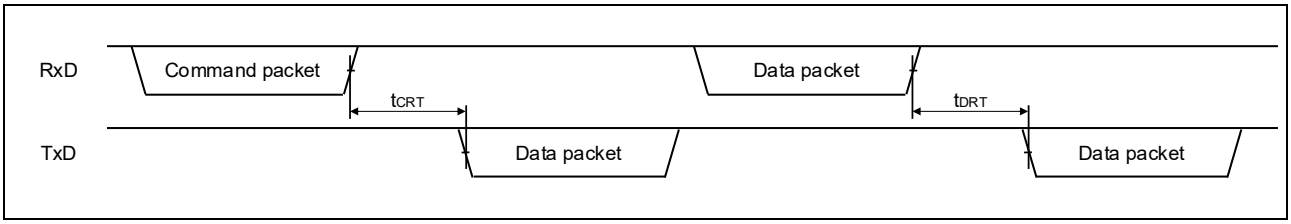


Figure 81. Key Setting Command

Parameter	Symbol	Min	Typ	Max	Unit
Command response time	tCRT	-	-	3 (*1)	s
Command response time	tCRT	-	-	[TBD: V6 (SiP FlashM!DO)VTI~)] (*2)	s
Data response time	tDRT	-	-	3 (*1)	s
Data response time	tDRT	-	-	[TBD: V6 (SiP FlashM!DO)VTI~)] (*2)	s

*1: Standard Product
*2: SiP Product

8.1.9 User Key Setting Command

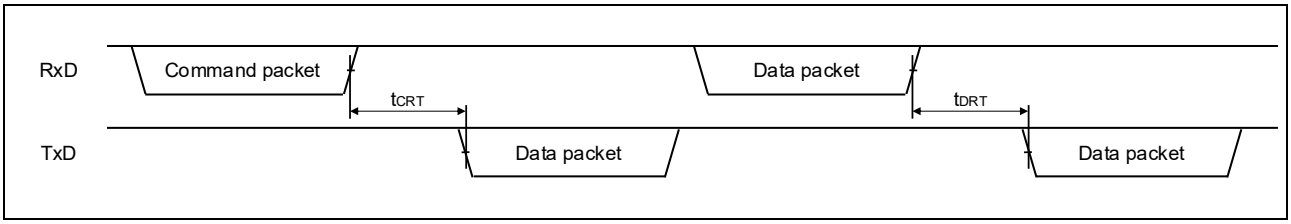


Figure 82. User Key Setting Command

Parameter	Symbol	Min	Typ	Max	Unit
Command response time	tCRT	-	-	3 (*1)	s
Command response time	tCRT	-	-	TBD (*2)	s
Data response time	tDRT	-	-	3 (*1)	s
Data response time	tDRT	-	-	TBD (*2)	s

*1: Standard Product
*2: SiP Product

8.1.10 Key Verify Command

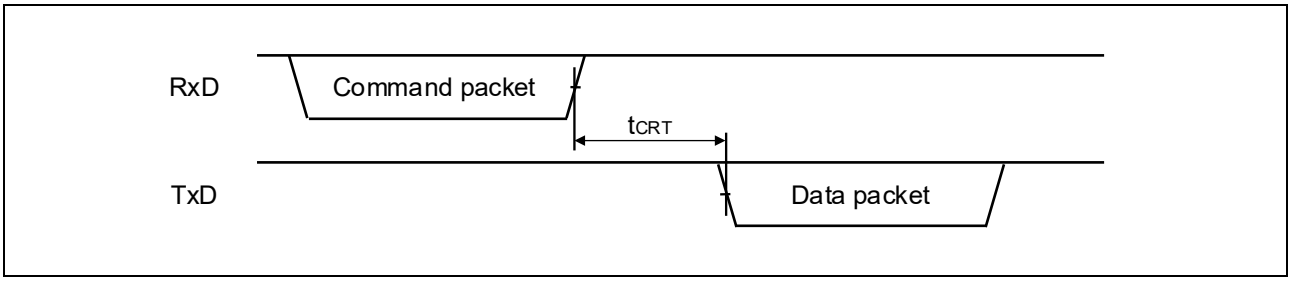


Figure 83. Key Verify Command

Parameter	Symbol	Min	Typ	Max	Unit
Command response time	tCRT	-	-	3 (*1)	s
Command response time	tCRT	-	-	TBD (*2)	s

*1: Standard Product
*2: SiP Product

8.1.11 User Key Verify Command

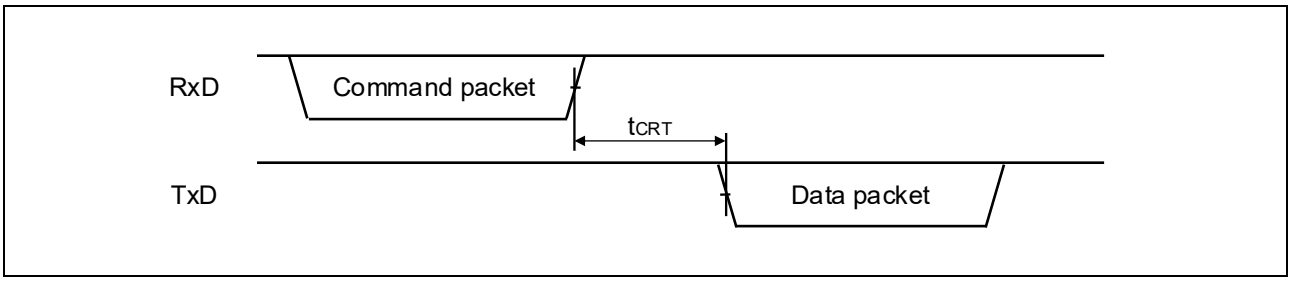


Figure 84. User Key Verify Command

Parameter	Symbol	Min	Typ	Max	Unit
Command response time	tCRT	-	-	3 (*1)	s
Command response time	tCRT	-	-	TBD (*2)	s

*1: Standard Product
*2: SiP Product

8.1.12 Initialize Command

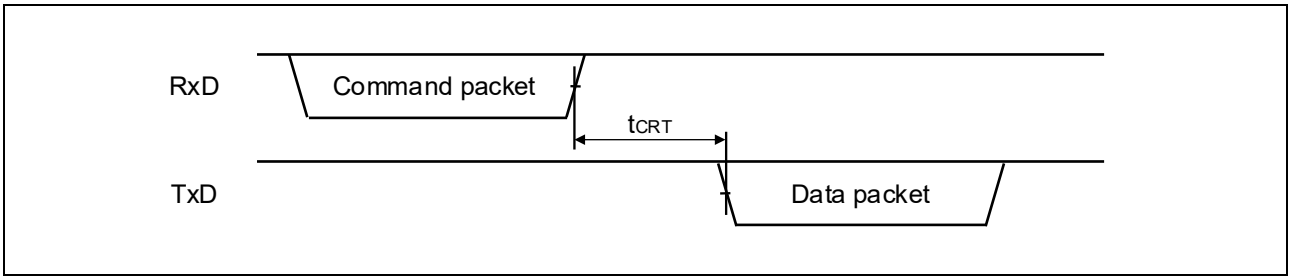


Figure 85. Initialize Command

Parameter	Symbol	Min	Typ	Max	Unit
Command response time	tCRT	-	-	60 (*1)	s
Command response time	tCRT	-	-	TBD (*2)	s

*1: Standard Product
*2: SiP Product

8.1.13 Boundary Setting Command

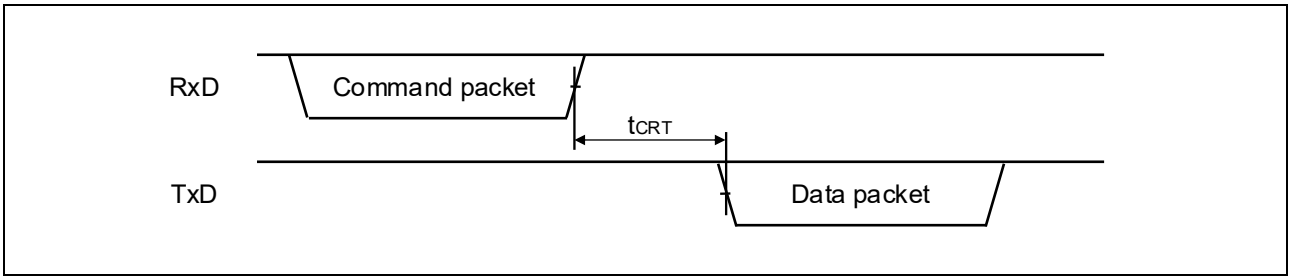


Figure 86. Boundary Setting Command

Parameter	Symbol	Min	Typ	Max	Unit
Command response time	tCRT	-	-	3 (*1)	s
Command response time	tCRT	-	-	TBD (*2)	s

*1: Standard Product
*2: SiP Product

8.1.14 Boundary Request Command

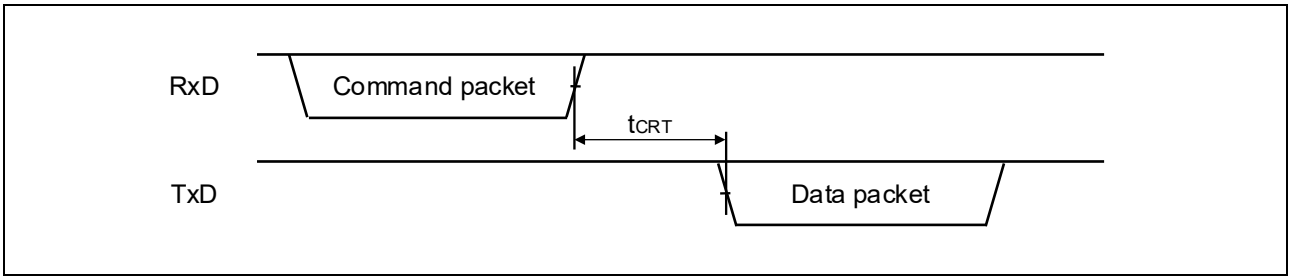


Figure 87. Boundary Request Command

Parameter	Symbol	Min	Typ	Max	Unit
Command response time	tCRT	-	-	3	s

8.1.15 Parameter Setting Command

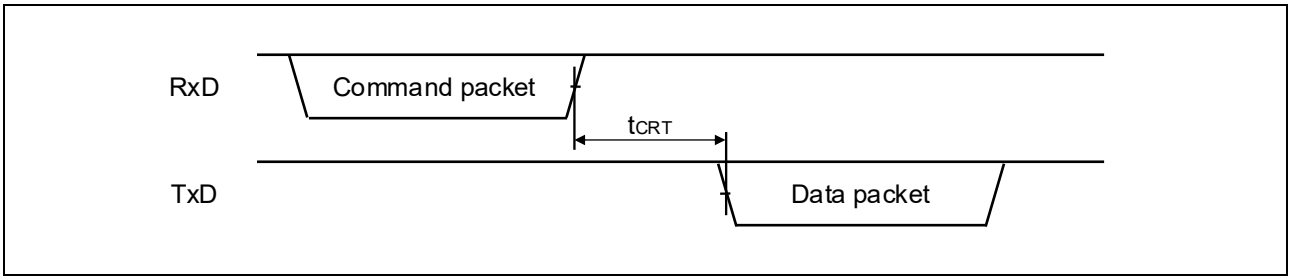


Figure 88. Parameter Setting Command

Parameter	Symbol	Min	Typ	Max	Unit
Command response time	tCRT	-	-	3 (*1)	s
Command response time	tCRT	-	-	TBD (*2)	s

*1: Standard Product
*2: SiP Product

8.1.16 Parameter Request Command

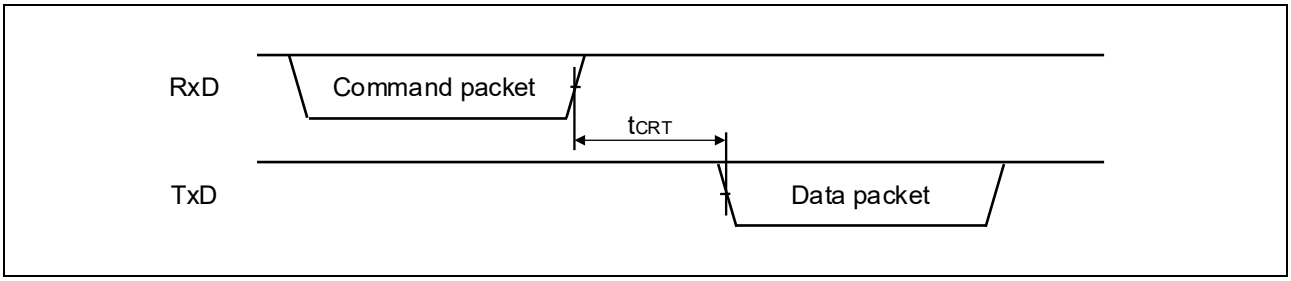


Figure 89. Parameter Request Command

Parameter	Symbol	Min	Typ	Max	Unit
Command response time	tCRT	-	-	3	s

8.1.17 ARC Configuration Setting Command

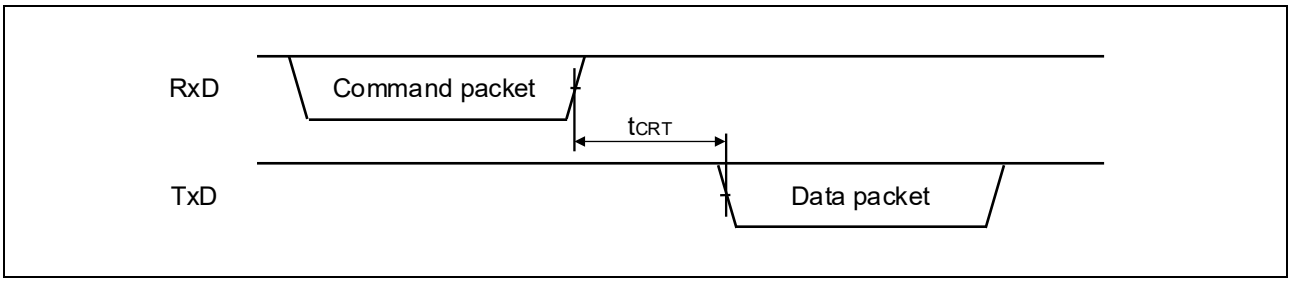


Figure 90. ARC Configuration Setting Command

Parameter	Symbol	Min	Typ	Max	Unit
Command response time	tCRT	-	-	3 (*1)	s
Command response time	tCRT			TBD (*2)	s

*1: Standard Product
*2: SiP Product

8.1.18 ARC Configuration Request Command

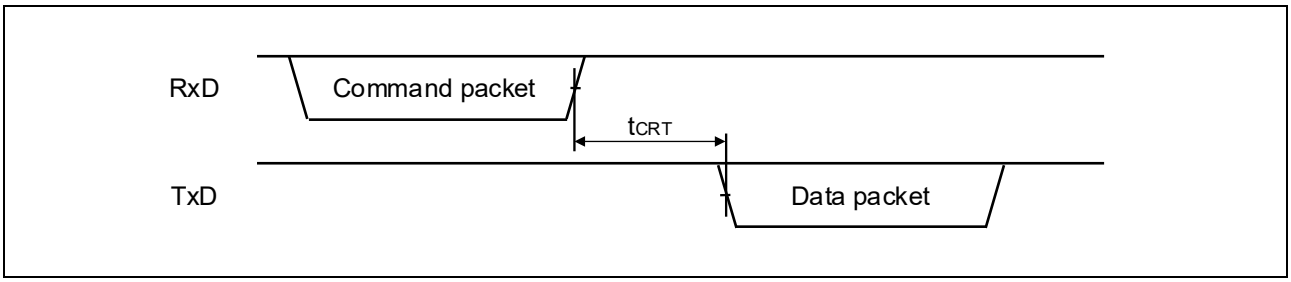


Figure 91. ARC Configuration Request Command

Parameter	Symbol	Min	Typ	Max	Unit
Command response time	tCRT	-	-	3	s

8.1.19 Inquiry Command

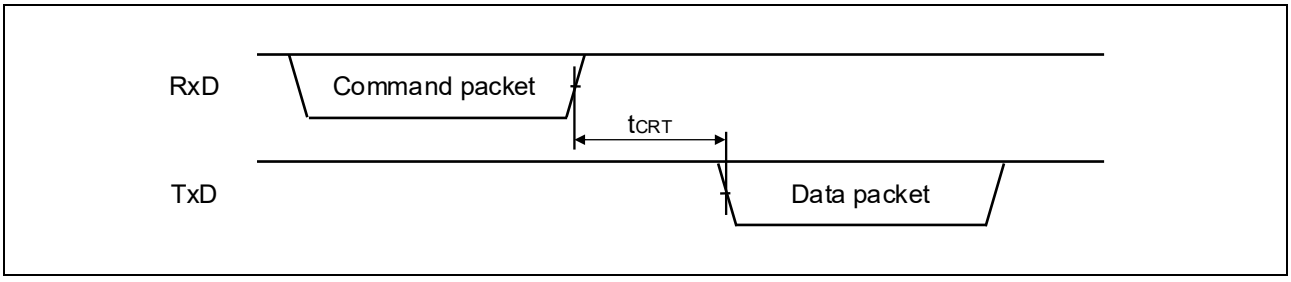


Figure 92. Inquiry Command

Parameter	Symbol	Min	Typ	Max	Unit
Command response time	tCRT	-	-	3	s

8.1.20 Signature Request Command

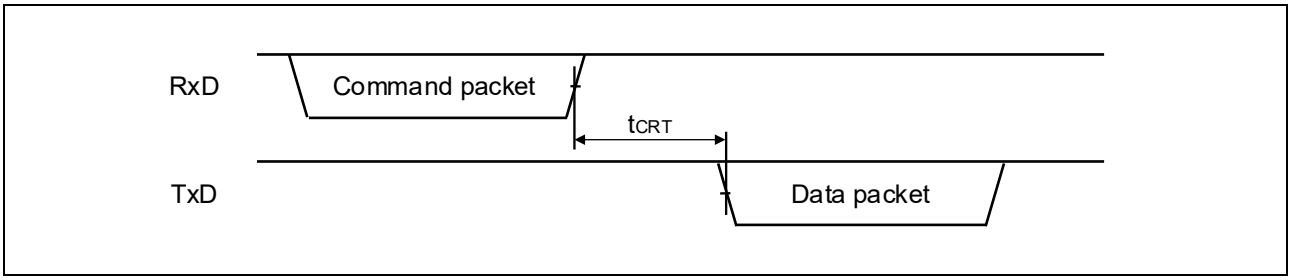


Figure 93. Signature Request Command

Parameter	Symbol	Min	Typ	Max	Unit
Command response time	tCRT	-	-	3	s

8.1.21 Area Information Request Command

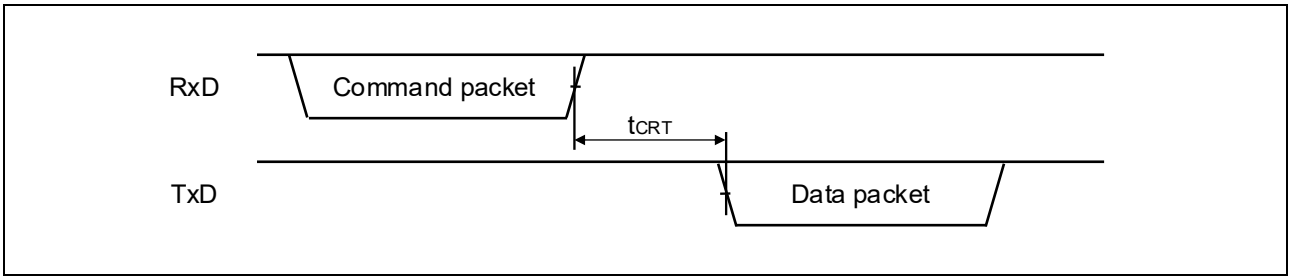


Figure 94. Area Information Request Command

Parameter	Symbol	Min	Typ	Max	Unit
Command response time	tCRT	-	-	3	s

8.1.22 Baudrate Setting Command

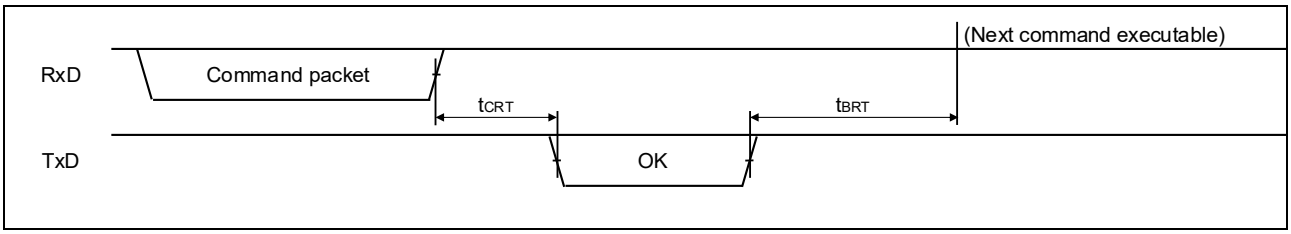


Figure 95. Baudrate Setting Command

Parameter	Symbol	Min	Typ	Max	Unit
Command response time	tCRT	-	-	3	s
Baudrate setting time	tBRT	-	-	1	ms

8.1.23 Erase Command

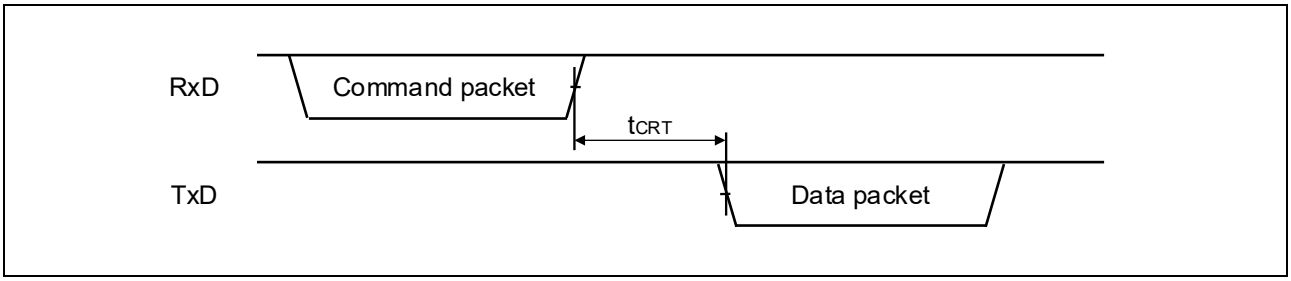


Figure 96. Erase Command

Parameter	Symbol	Min	Typ	Max	Unit
Command response time	tCRT	-	-	3 (*1) (*2)	s
Command response time	tCRT	-	-	TBD (*1)(*3)	s

*1: Note that the response time when accessing external flash area depends on the external flash memory access driver and the external flash memory embedded on the user's system.
*2: Standard Product
*3: SiP Product

8.1.24 Write Command

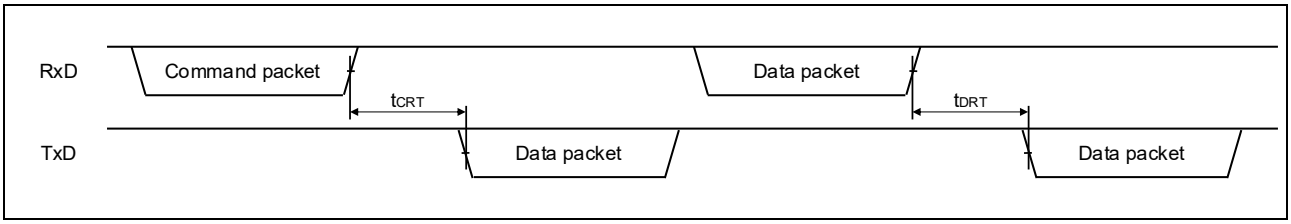


Figure 97. Write Command

Parameter	Symbol	Min	Typ	Max	Unit
Command response time	tCRT	-	-	3 (*2)	s
Data response time	tDRT	-	-	30 (*1) (*2)	s
Command response time	tCRT	-	-	TBD (*3)	s
Data response time	tDRT	-	-	TBD (*1) (*3)	s

*1: Note that the response time when accessing external flash area depends on the external flash memory access driver and the external flash memory embedded on the user's system.
*2: Standard Product
*3: SiP Product

8.1.25 Read Command

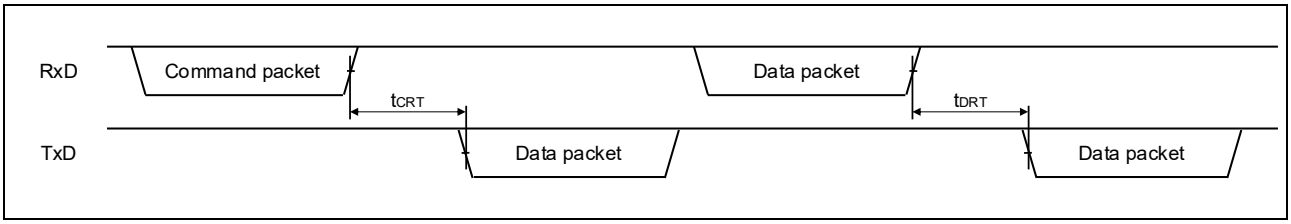


Figure 98. Read Command

Parameter	Symbol	Min	Typ	Max	Unit
Command response time	tCRT	-	-	3 (*2)	s
Data response time	tDRT	-	-	30 (*1) (*2)	s
Command response time	tCRT			TBD (*3)	s
Data response time	tDRT			TBD (*1) (*3)	s

*1: Note that the response time when accessing external flash area depends on the external flash memory access driver and the external flash memory embedded on the user's system.
*2: Standard Product
*3: SiP Product

8.1.26 CRC Command

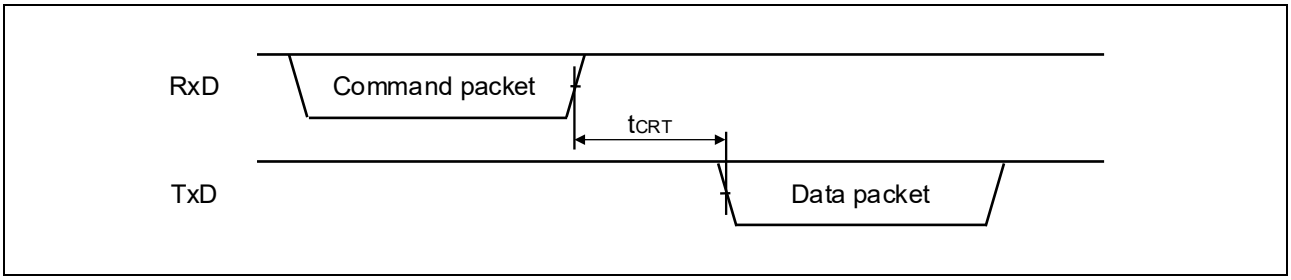


Figure 99. CRC Command

Parameter	Symbol	Min	Typ	Max	Unit
Command response time	tCRT	-	-	3 (*1) (*2)	s
Command response time	tCRT			TBD (*1) (*3)	s

*1: Note that the response time when accessing external flash area depends on the external flash memory access driver and the external flash memory embedded on the user's system.
*2: Standard Product
*3: SiP Product

8.1.27 OEM Root Public Key Setting Command

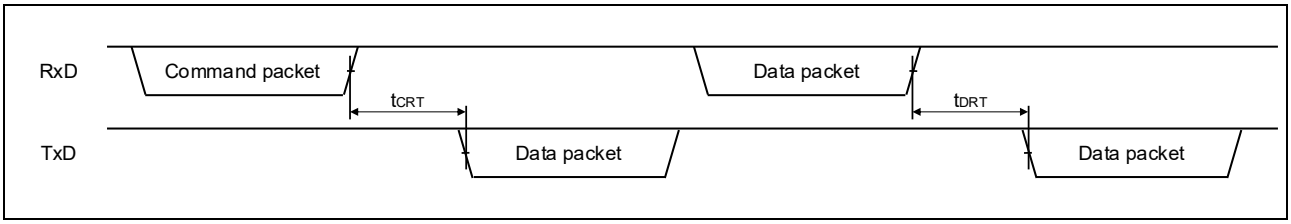


Figure 100. OEM Root Public Key Setting Command

Parameter	Symbol	Min	Typ	Max	Unit
Command response time	tCRT	-	-	3 (*1)	s
Data response time	tDRT	-	-	30 (*1)	s
Command response time	tCRT	-	-	TBD (*2)	s
Data response time	tDRT	-	-	TBD (*2)	s

*1: Standard Product
*2: SiP Product

8.1.28 OEM Root Public Key Verify Command

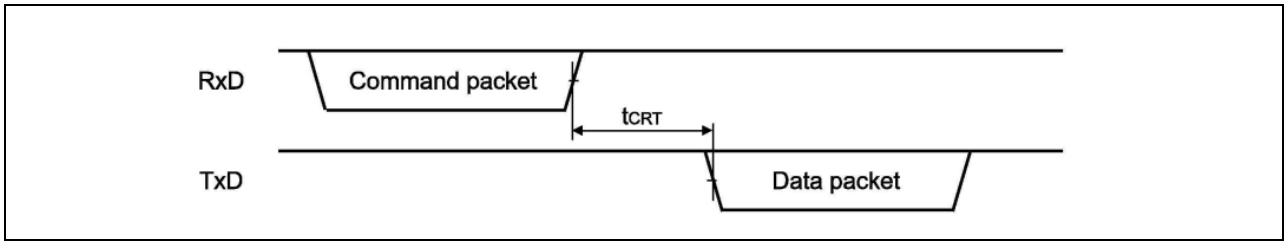


Figure 101. OEM Root Public Key Verify Command

Parameter	Symbol	Min	Typ	Max	Unit
Command response time	tCRT	-	-	3 (*1)	s
Command response time	tCRT	-	-	TBD (*2)	s

*1: Standard Product
*2: SiP Product

8.1.29 Code Certificate Update Command

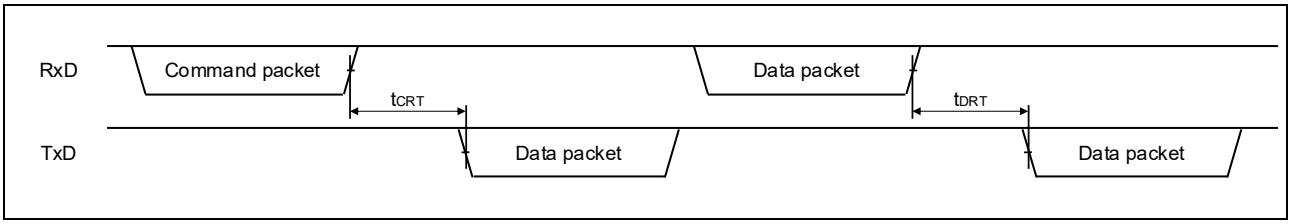


Figure 102. Code Certificate Update Command

Parameter	Symbol	Min	Typ	Max	Unit
Command response time	tCRT	-	-	3 (*1)	s
Data response time	tDRT	-	-	30 (*1)	s
Command response time	tCRT			TBD (*2)	s
Data response time	tDRT			TBD (*2)	s

*1: Standard Product
*2: SiP Product

8.1.30 Code Certificate Check Command

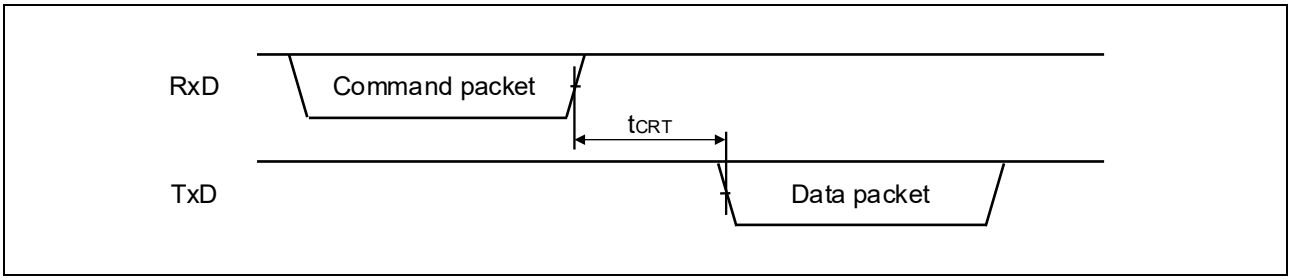


Figure 103. Code Certificate Check Command

Parameter	Symbol	Min	Typ	Max	Unit
Command response time	tCRT	-	-	3(*1)	s
Command response time	tCRT	-	-	TBD (*2)	s

*1: Standard Product
*2: SiP Product

8.1.31 SiP Flash Various Setting Command

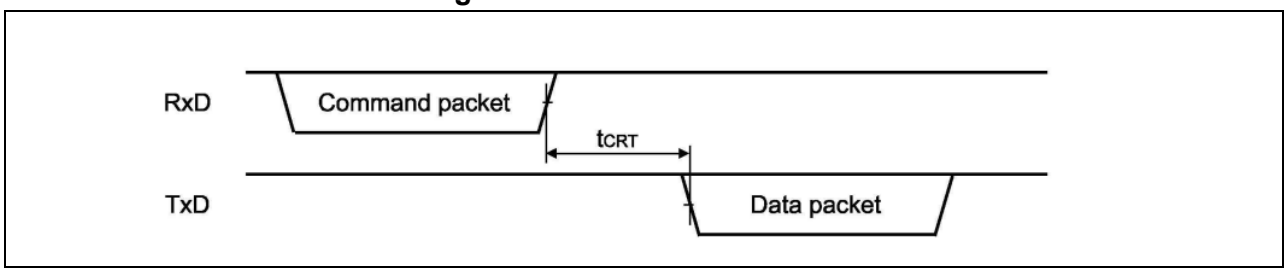


Figure 104. SiP Flash Various Setting Command

Parameter	Symbol	Min	Typ	Max	Unit
Command response time	tCRT	-	-	TBD	s

8.1.32 SiP Flash Various Request Command

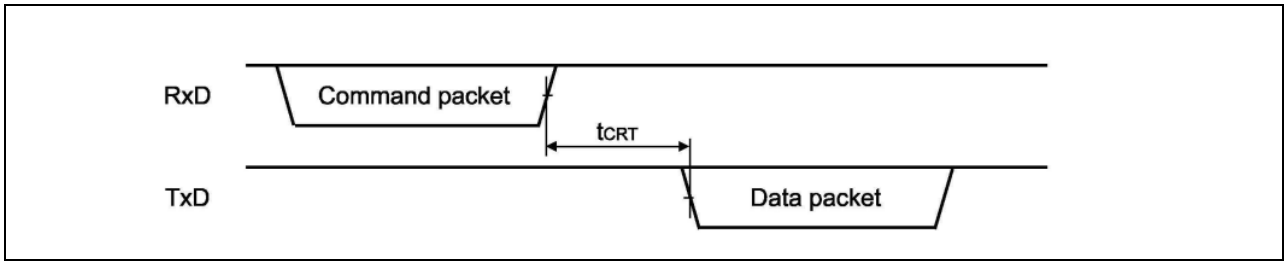


Figure 105. SiP Flash Various Request Command

Parameter	Symbol	Min	Typ	Max	Unit
Command response time	tCRT	-	-	TBD	s

8.1.33 External Flash Memory Setting Command

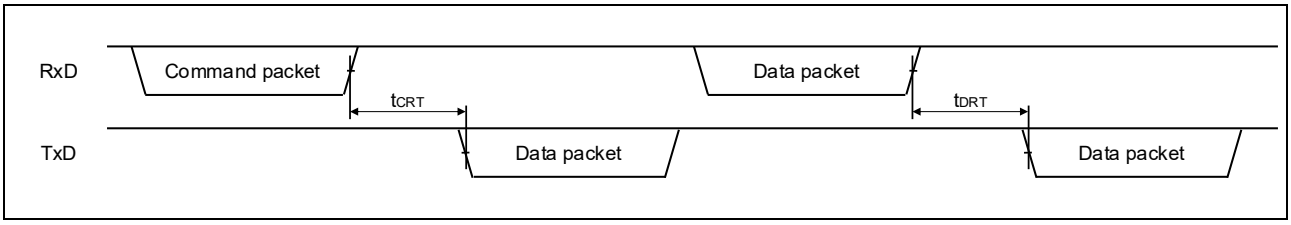


Figure 106. External Flash Memory Setting Command

Parameter	Symbol	Min	Typ	Max	Unit
Command response time	tCRT	-	-	3 (*2)	s
Data response time	tDRT	-	-	3 (*1) (*2)	s
Command response time	tCRT			TBD (*3)	s
Data response time	tDRT	-	-	TBD (*1) (*3)	s

*1: Note that the response time of the last data packet depends on the external flash memory access driver and the external flash memory embedded on the user's system.
*2: Standard Product
*3: SiP Product

8.1.34 Encrypted Data Write Command

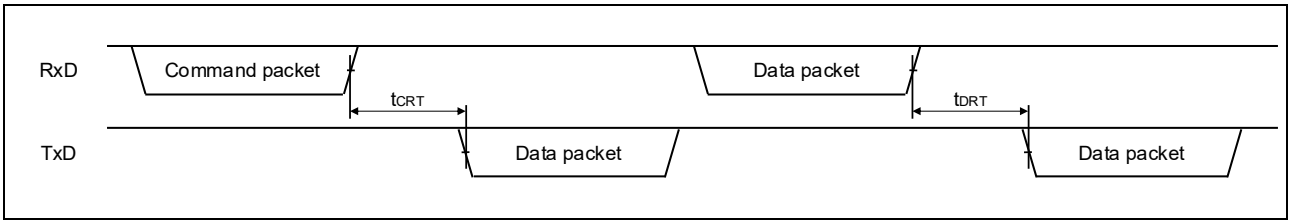


Figure 107. Encrypted Data Write Command

*) tDRT specifies the longest time among all the kinds of data packets of this command

Parameter	Symbol	Min	Typ	Max	Unit
Command response time	tCRT	-	-	30 (*2)	s
Data response time	tDRT	-	-	30 (*1) (*2)	s
Command response time	tCRT	-	-	TBD (*3)	s
Data response time	tDRT	-	-	TBD (*1)(*3)	s

*1: Note that the response time when accessing external flash area depends on the external flash memory access driver and the external flash memory embedded on the user's system.
*2: Standard Product
*3: SiP Product

9. Sequencer Command List

Table 25 shows the sequencer commands executed by each communication command.

Table 25. Sequencer Command List

Communication command	Sequencer command	Number of issue times
DLM state transit command	Configuration set	1 time
	Program	1 time
	Forced stop *) Use to clear error status.	1 time
Protection level transit command	Configuration set	1 time
	Forced stop *) Use to clear error status.	1 time
Authentication command	Program	Depends on the DDLM and the result of all erasure at RMA_REQ transition
	Configuration set	Depends on the DDLM and the result of all erasure at RMA_REQ transition
	Forced stop *) Use to clear error status	Depends on the DDLM and the result of all erasure at RMA_REQ transition
Key setting command	Configuration set	4 times
	Forced stop *) Use to clear error status	1 time
Initialize command	Program	12 times
	Configuration set	126 times
	Forced stop *) Use to clear error status	Depends on the result of all erasure
Boundary setting command	Configuration set	1 time
	Forced stop *) Use to clear error status	1 time

Communication command	Sequencer command	Number of issue times
Parameter setting command	Program	1 time
	Forced stop *) Use to clear error status	1 time
ARC configuration setting command	Program	1 time
	Forced stop *) Use to clear error status	1 time
Write command	Program	Depends on designated address
	Configuration set	Depends on designated address
	Forced stop *) Use to clear error status	1 time
OEM root public key setting command	Program	3 times
	Forced stop *) Use to clear error status	1 time
Code certificate update command	Read counter	Depends on current and new OEM boot loader version
	Increment counter	Depends on current and new OEM boot loader version
	Forced stop *) Use to clear error status	Depend on the result of writing “Code certificate” and “MAC value of Code certificate and OEM boot loader”
Encrypted data write command	Program	Depends on designated address
	Configuration set	Depends on designated address
	Forced stop *) Use to clear error status	1 time

10. Precaution List

10.1.1 Initialize Command

1. The following parameters are not initialized by this command. For details on each parameter, refer to Parameter setting command.
   - Disable of authentication using AL1_KEY
   - Disable transition to LCK_BOOT
   - Master CPU selection
   - Disable security extension of CPU1
2. The following areas are not initialized by this command.
   - Anti-rollback counter area
   - External flash area
3. Only Status register and Main Flash areas are erased for SiP Flash. However, areas where protection cannot be released are not erased. As a result, after executing this command, the following data remains not erased and the device's protection level is changed back to PL2.
   - Value of Status register, when permanent lock of Status register is set
   - Data in Main Flash where protection by BP is set, when permanent lock of Status register is set and also protection by BP is set
   - Data in Main Flash where protection by other than BP is set
   - Data in other than Status register and Main Flash

   Data leakage, etc, by this can be prevented by disabling initialization using Parameter setting command.

10.1.2 Erase Command

1. When accessing the external flash area, the driver function for access is called. Therefore, send the driver code with the “External flash memory setting command” in advance. In this command, “EraseChip driver” is called when the entire area of external flash area 0 is specified. Otherwise, the “EraseSector driver” is called every time a sector is erased.
   Also, access to addresses to which external flash memory is not allocated is not guaranteed.

10.1.3 Write Command

1. If permanent block protection or OFSPS in the Config OTP area is set, the protected area cannot be rewritten. Therefore, rewrite the protected area before setting them.
2. When accessing the External flash area, the driver function for access is called, so send the driver code with the “External flash memory setting command” in advance. This command calls the “Program Data driver”. Also, access to addresses to which external flash memory is not allocated is not guaranteed.
3. Config OTP area cannot be written back to 1 after 0 is written. For Config OTP area which has ECC, ECC bits cannot be written back from 0 to 1. Therefore, rewriting is not possible even when the data bits are to be changed only from 1 to 0. However, rewriting is possible as an exception in the following cases:
   - Rewriting the same value
   - The data already written is all“F”.

10.1.4 Read Command

1. “External flash memory setting command” must be executed in advance to access the External flash area. Boot firmware does not call any External flash memory access drivers in this command; When External flash area is specified as SAD/EAD, boot firmware issues read access to the specified address on assumption that Octal SPI has been initialized to Memory mapping mode. Also, access to addresses to which external flash memory is not allocated is not guaranteed.

10.1.5 CRC Command

1. ”External flash memory setting command” must be executed in advance to access the External flash area. Boot firmware does not call any External flash INI I X memory access drivers in this command; When External flash area is specified as SAD/EAD, boot firmware issues read access to the specified address on assumption that Octal SPI has been initialized to Memory mapping mode. Also, access to addresses to which external flash memory is not allocated is not guaranteed.

10.1.6 Code Certificate Update Command

1. Use this command after writing “OEM boot loader” to the User area and “Code certificate start address” to the Config OTP area with the Write command or Encrypted data write command in advance.
2. Use this command after saving the “OEM root public key encrypted Hash value” in the device in advance with the OEM root public key setting command.
3. Verification fails if data of received Key certificate or Code certificate does not conform to device specifications.
   Refer to the user's manual of the device for certificates' specifications.
4. Key certificate is not necessary when MAC type = None. Specify KCS = 0 and do not send any data as Key certificate data in this case.

10.1.7 SiP Flash Various Setting Command

(1) It is not possible to execute neither setting or reading the functions below which SiP Flash has via boot firmware.
   - Configuration register
   - Nonvolatile lock bit

- Password
- Sector protection register

In addition, when these settings are changed from the shipped state by a software other than boot firmware, operations regarding SiP Flash via boot firmware are not guaranteed. In this case it is possible again to execute operations for SiP Flash via boot firmware, that is, the user's program sets these settings back to shipped state.

(2) Programming of OTP array is permanently disabled after OTP permanent lock mapped in OTP array itself is set.
Programming of PMR is permanently disabled after PMR permanent lock mapped in PMR itself is set.
Programming of Status register is permanently disabled after Status register permanent lock mapped in PMR is set.
(At this time, programming of SiP Flash area protected by BP is also permanently disabled, when BP is set by Status register).

(3) When SiP Flash programming fails by this command or by other commands, that is, in other words when SiP Flash access error is returned or the programmed data is not reflected, it may be that an operation that is against a protection function of SiP Flash was executed. The table below shows protection functions of SiP Flash which may cause failure of SiP Flash programming via boot firmware. Check the settings of these protection functions when programming SiP Flash fails.

Protected area	Protection function	
	Name	Can be enabled by
Main Flash (referred to as SiP Flash area in area information)	Block protect	Status register
	Lock bit	Nonvolatile lock bit
OTP array	OTP permanent lock	OTP control byte placed in OTP array itself
Status register	Status register lock	PMR register
PMR register	PMR lock	PMR register itself

10.1.8 External Flash Memory Setting Command

(1) This command does not affect SiP Flash area programming.
Programming to External flash area and programming to SiP Flash area can be executed independently of each other.

In addition, even after OCTACLK is changed by this command, boot firmware changes OCTACLK to appropriate frequency before SiP Flash programming and then changes it back after the programming finishes. Therefore, SiP Flash programming and External flash memory programming does not also affect programming speed of each other.

10.1.9 Encrypted Data Write Command

(1) This command becomes inexecutable once permanent block protection is set.

(2) This command becomes inexecutable if any of the followings is set and also OFSPS is set:
- SAS.BTFLG = 0b
- BANKSEL.BANKSWP[2:0] ≠ 111b (only for dual mode supported devices)
- BANKSEL_SEC.BANKSWP[2:0] ≠ 111b (only for dual mode supported devices)

(3) If permanent block protection or OFSPS in the Config OTP area is written before the protected area, this command abnormally finishes at writing of the protected area.
To avoid this, Data packet [encrypted user data] for the protected areas must be sent earlier than those for permanent block protection area or OFSPS area.

(4) Do not set permanent block protection of the area where user keys are to be written when User key setting command is used.
Do not set permanent block protection of the area where Code certificate is to be written when Code certificate update command is used.
If they are set, both commands become inexecutable due to Protection error.

(5) When accessing the External flash area, the driver function for access is called, so send the driver code with the “External flash memory setting command” in advance. This command calls the “Program Data driver”.
Also, access to addresses to which external flash memory is not allocated is not guaranteed.

(6) Config OTP area cannot be written back to 1 after 0 is written.

For Config OTP area which has ECC, ECC bits cannot be written back from 0 to 1. Therefore, note that, re-writing is not possible even when the data bits are only to be changed from 1 to 0. However, rewriting is possible as an exception in the following cases:
- Rewriting the same value
- The data already written is all“F”

(7) The theoretical maximum data size that can be written with this command is 16,519,088 bytes (*1), because the maximum value of LOD is 00FFFFF0h (16,777,200) as described in the explanation of LOD.

However, because the write data cannot be sent by one “Data packet [encrypted user data]” when the write destination address is not consecutive as explained, the actual maximum data size that can be written with this command depends on the area information of the product and the write destination address of the data to be written.

*1) 16,519,088-byte data can be sent by 16132 packets of “Data packet [encrypted user data]”; for which the 16131 packets send 1024-byte write data each, and the remaining one packet sends 944-byte write data. Accordingly, when the write data size is 16,519,088-byte, LOD = (1024×16131) + 944 + (16×16132) = 16,777,200 = 00FFFFF0h = Maximum.

11. Causes for Operation Stop

The boot firmware enters an infinite loop in the following cases:

11.1 Initialization Phase

- When the following CPU exceptions occur:
  NMI / HardFault / MemManage / BusFault / UsageFault / SecureFault / SVCall / DebugMonitor / PendSV / SysTick.
- When Trusted system goes into an abnormal state.

11.2 Communication Setting Phase

- When the USB cable is disconnected with the USB status as “Configured”.
- When the following CPU exceptions occur:
  NMI / HardFault / MemManage / BusFault / UsageFault / SecureFault / SVCall / DebugMonitor / PendSV / SysTick.

11.3 Command Acceptable Phase

- When the USB cable is disconnected with the USB status as “Configured”.
- When the following CPU exceptions occur:
  NMI / HardFault / MemManage / BusFault / UsageFault / SecureFault / SVCall / DebugMonitor / PendSV / SysTick.

11.4 DLM State Transit Command

- When transition to LCK_BOOT is complete.
- When transition to RMA_RET is complete.
- When Hardware error occurred.

11.5 Protection Level Transit Command

- When Hardware error occurred.

11.6 Authentication Command

- When Trusted system goes into an abnormal state.
- When DLM state transition is complete.
- When Hardware error occurred.

11.7 Key Setting Command

- When Trusted system goes into an abnormal state.

11.8 User Key Setting Command

- When Trusted system goes into an abnormal state.

11.9 Key Verify Command

- When Trusted system goes into an abnormal state.

11.10 User Key Verify Command

- When Trusted system goes into an abnormal state.

11.11 Initialize Command

- When the command completes successfully.
- When Hardware error occurred.

11.12 OEM Root Public Key Setting Command

- When Trusted system goes into an abnormal state.

11.13 Code certificate update command

- When Trusted system goes into an abnormal state.
- When the OEM_BL address pointed to by the Code certificate is invalid.

11.14 Code Certificate Check Command

- When Trusted system goes into an abnormal state.
- When the OEM_BL address pointed to by the Code certificate is invalid.
- When the value of the Code certificate start address is invalid.
- When the Code certificate check command is executed even though the Code certificate update command has not completed normally.

11.15 Encrypted Data Write Command

- When the Trusted system goes into an abnormal state
- When Hardware error occurred

12. Causes for Software Reset

Boot firmware performs software reset in the following cases:

12.1 Communication Setting Phase

- When all of the conditions below are met:
- MD = 1.
- Not JTAG/SWD mode.
- Top 8 bytes of User area are not all-F.

Website and Support

Visit the following URLs to learn about key elements of the RA family, download components and related documentation, and get support:

RA Product Information	renesas.com/ra
RA Product Support Forum	renesas.com/ra/forum
RA Flexible Software Package	renesas.com/FSP
Renesas Support	renesas.com/support

Revision History

Rev.	Date	Description	
		Page	Summary
1.00	Oct.01.25	-	First release document

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

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

1. Precaution against Electrostatic Discharge (ESD)
   A strong electrical field, when exposed to a CMOS device, can cause destruction of the gate oxide and ultimately degrade the device operation. Steps must be taken to stop the generation of static electricity as much as possible, and quickly dissipate it when it occurs. Environmental control must be adequate. When it is dry, a humidifier should be used. This is recommended to avoid using insulators that can 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 must be grounded. The operator must also be grounded using a wrist strap. Semiconductor devices must not be touched with bare hands. Similar precautions must be taken for printed circuit boards with mounted semiconductor devices.
2. Processing at power-on
   The state of the product is undefined at the time when power is supplied. The states of internal circuits in the LSI are indeterminate and the states of register settings and pins are undefined at the time when power is supplied. In a finished product where the reset signal is applied to the external reset pin, the states of pins are not guaranteed from the time when power is supplied until the reset process is completed. In a similar way, the states of pins in a product that is reset by an on-chip power-on reset function are not guaranteed from the time when power is supplied until the power reaches the level at which resetting is specified.
3. Input of signal during power-off state
   Do not input signals or an I/O pull-up power supply while the device is powered off. 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. Follow the guideline for input signal during power-off state as described in your product documentation.
4. Handling of unused pins
   Handle unused pins in accordance with the directions given under handling of unused pins in the manual. The input pins of CMOS products are generally in the high-impedance state. In operation with an unused pin in the open-circuit state, extra electromagnetic noise is induced in the vicinity of the LSI, an associated shoot-through current flows internally, and malfunctions occur due to the false recognition of the pin state as an input signal become possible.
5. Clock signals
   After applying a reset, only release the reset line after the operating clock signal becomes stable. When switching the clock signal during program execution, wait until the target clock signal is stabilized. When the clock signal is generated with an external resonator or from an external oscillator during a reset, ensure that the reset line is only released after full stabilization of the clock signal. Additionally, when switching to a clock signal produced with an external resonator or by an external oscillator while program execution is in progress, wait until the target clock signal is stable.
6. 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 V_{IL} (Max.) and V_{IH} (Min.) due to noise, for example, 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 V_{IL} (Max.) and V_{IH} (Min.).
7. Prohibition of access to reserved addresses
   Access to reserved addresses is prohibited. The reserved addresses are provided for possible future expansion of functions. Do not access these addresses as the correct operation of the LSI is not guaranteed.
8. Differences between products
   Before changing from one product to another, for example to a product with a different part number, confirm that the change will not lead to problems. The characteristics of a microprocessing unit or microcontroller unit products in the same group but having a different part number might differ in terms of internal memory capacity, layout pattern, and other factors, which can affect the ranges of electrical characteristics, such as characteristic values, operating margins, immunity to noise, and amount of radiated noise. When changing to a product with a different part number, implement a system-evaluation test for the given product.

Notice

1. Descriptions of circuits, software and other related information in this document are provided only to illustrate the operation of semiconductor products and application examples. You are fully responsible for the incorporation or any other use of the circuits, software, and information in the design of your product or system. Renesas Electronics disclaims any and all liability for any losses and damages incurred by you or third parties arising from the use of these circuits, software, or information.
2. Renesas Electronics hereby expressly disclaims any warranties against and liability for infringement or any other claims involving patents, copyrights, or other intellectual property rights of third parties, by or arising from the use of Renesas Electronics products or technical information described in this document, including but not limited to, the product data, drawings, charts, programs, algorithms, and application examples.
3. No license, express, implied or otherwise, is granted hereby under any patents, copyrights or other intellectual property rights of Renesas Electronics or others.
4. You shall be responsible for determining what licenses are required from any third parties, and obtaining such licenses for the lawful import, export, manufacture, sales, utilization, distribution or other disposal of any products incorporating Renesas Electronics products, if required.
5. 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.
6. 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.
7. No semiconductor product is absolutely secure. Notwithstanding any security measures or features that may be implemented in Renesas Electronics hardware or software products, Renesas Electronics shall have absolutely no liability arising out of any vulnerability or security breach, including but not limited to any unauthorized access to or use of a Renesas Electronics product or a system that uses a Renesas Electronics product. RENESAS ELECTRONICS DOES NOT WARRANT OR GUARANTEE THAT RENESAS ELECTRONICS PRODUCTS, OR ANY SYSTEMS CREATED USING RENESAS ELECTRONICS PRODUCTS WILL BE INVULNERABLE OR FREE FROM CORRUPTION, ATTACK, VIRUSES, INTERFERENCE, HACKING, DATA LOSS OR THEFT, OR OTHER SECURITY INTRUSION (“Vulnerability Issues”). RENESAS ELECTRONICS DISCLAIMS ANY AND ALL RESPONSIBILITY OR LIABILITY ARISING FROM OR RELATED TO ANY VULNERABILITY ISSUES. FURTHERMORE, TO THE EXTENT PERMITTED BY APPLICABLE LAW, RENESAS ELECTRONICS DISCLAIMS ANY AND ALL WARRANTIES, EXPRESS OR IMPLIED, WITH RESPECT TO THIS DOCUMENT AND ANY RELATED OR ACCOMPANYING SOFTWARE OR HARDWARE, INCLUDING BUT NOT LIMITED TO THE IMPLIED WARRANTIES OF MERCHANTABILITY, OR FITNESS FOR A PARTICULAR PURPOSE.
8. 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.
9. 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.
10. 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.
11. 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.
12. 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.
13. This document shall not be reprinted, reproduced or duplicated in any form, in whole or in part, without prior written consent of Renesas Electronics.
14. Please contact a Renesas Electronics sales office if you have any questions regarding the information contained in this document or Renesas Electronics products.

(Note1) “Renesas Electronics” as used in this document means Renesas Electronics Corporation and also includes its directly or indirectly controlled subsidiaries.

(Note2) “Renesas Electronics product(s)” means any product developed or manufactured by or for Renesas Electronics.

(Rev.5.0-1 October 2020)

Corporate Headquarters

TOYOSU FORESIA, 3-2-24 Toyosu,
Koto-ku, Tokyo 135-0061, Japan

www.renesas.com

Trademarks

Renesas and the Renesas logo are trademarks of Renesas Electronics Corporation. All trademarks and registered trademarks are the property of their respective owners.

Contact information

For further information on a product, technology, the most up-to-date version of a document, or your nearest sales office, please visit:
www.renesas.com/contact/.