Renesas RA Family

Renesas Boot Firmware for RA4L1 MCU Group

Introduction

This application note describes the communication protocol, command set, and usage of the boot firmware provided with Renesas RA4L1 MCU Group.

Target Device

RA4L1 MCU Group

Contents

1. Terminology

1.1 Boot Firmware

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

1.2 Flash Memory

The following areas are collectively called flash memory:

- Code Flash: The ROM area where program code is written (FLP/FLI)
- Data Flash: The ROM area where data is written (EEP)

The Code Flash area used by the user is called the "User area"; the Data Flash area used by the user is called the "Data area"; and the area to store configuration data is called the "Config area". The boot firmware rewrites and reads the User area, Data area, and Config area according to commands given by the user.

Also, in this document, the flash memory area that can be rewritten by boot firmware may be generically referred to as "memory".

Figure 1 shows an example of flash memory structure. Memory structure differs from device to device.

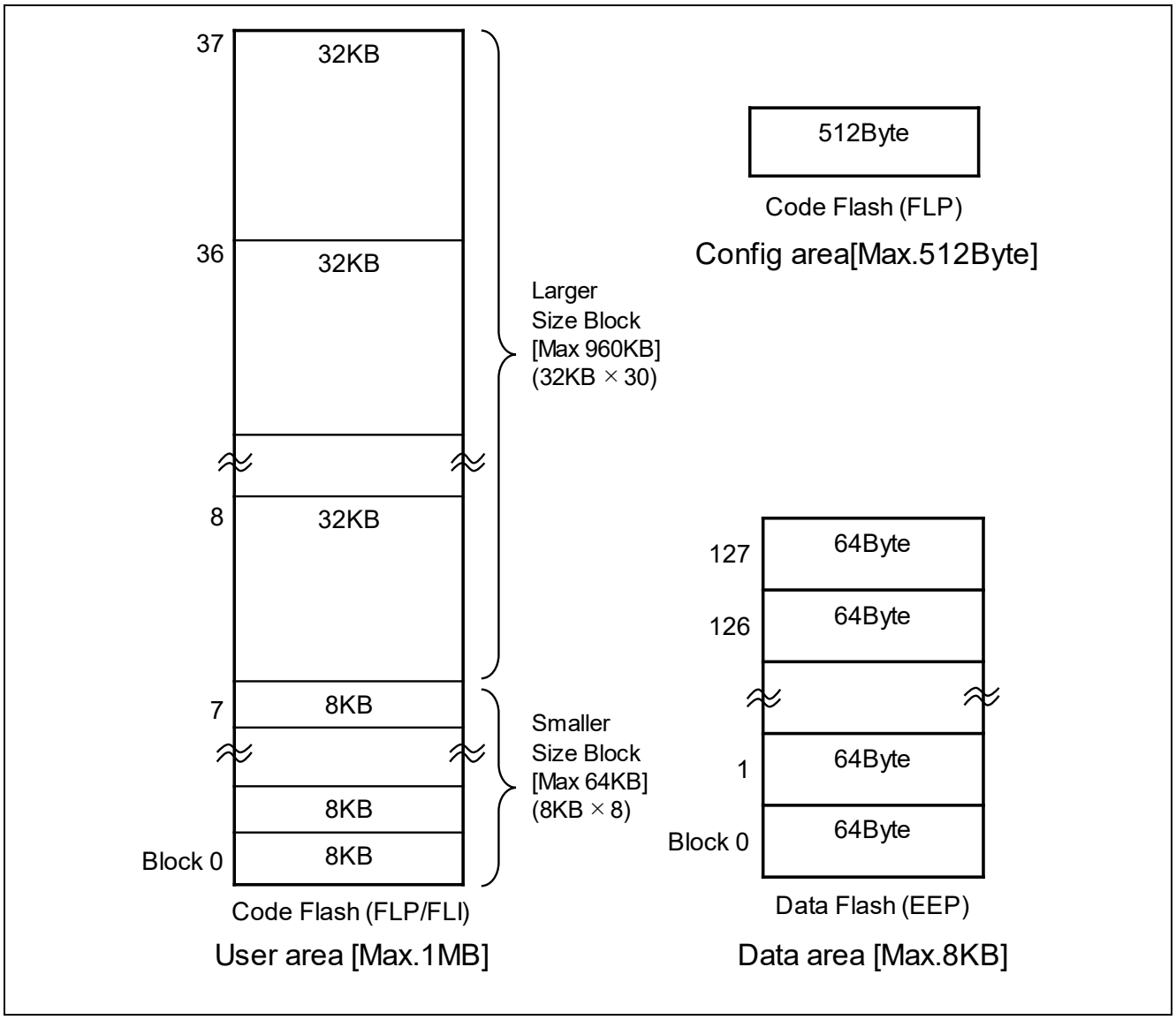


Figure 1. Flash Memory Structure Example

1.3 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.

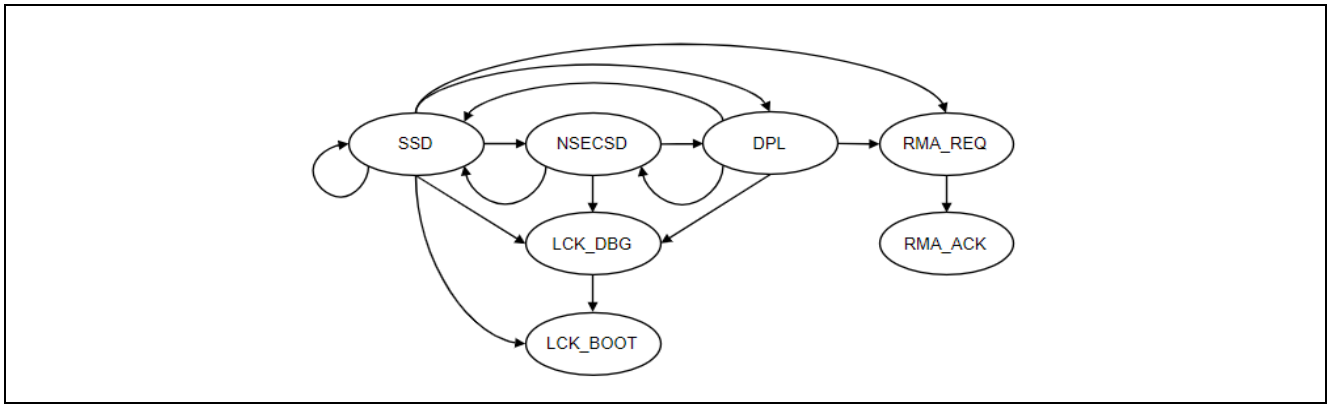


Figure 2. Device Lifecycle States

Table 1. DLM States

DLM State Name	Description
SSD	Secure Software Development
NSECSD	Non-SECure Software Development
DPL	DePLoyed
LCK_DBG	LoCKed DeBuG
LCK_BOOT	LoCKed BOOT interface
RMA_REQ	Return Material Authorization REQuest
RMA_ACK	Return Material Authorization ACKnowledged

1.4 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.

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

1.5 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 listed in Table 2.

Table 2. Block Protection Types

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

1.6 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.

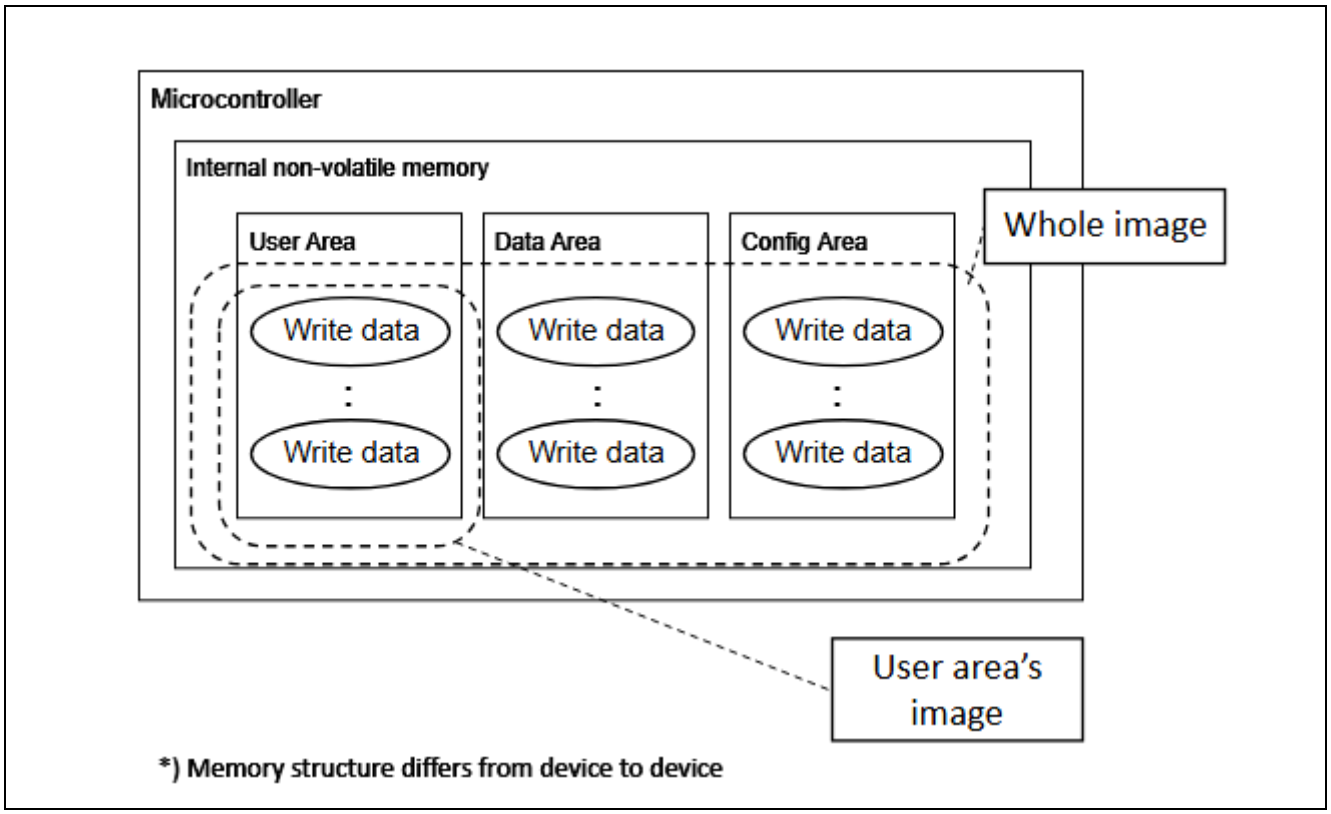


Figure 3. Memory Image Concept

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 RA4L1 MCU Group

This chapter describes the system architecture of RA4L1 MCU Group regarding the flash memory control.

Table 3. Operating Environment

<table>
<tr><th>CPU core</th><th>Arm Cortex-M33 with TrustZone</th></tr>
<tr><td>Max CPU operating frequency</td><td>80 MHz (Maximum by HW specifications)
Boot firmware operating frequency is as below:
- 48MHz when VCC is higher than 1.8V
- 4MHz when VCC is lower than or equal to 1.8V</td></tr>
<tr><td>Main-OSC</td><td>4, 6, 8, 12, 16 MHz
*If neither is set, operate with HOCO.
*However, a Main-OSC whose frequency is around plus-minus 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 use a Main-OSC of the very frequency listed above when using USB communications.</td></tr>
<tr><td>Operating voltage</td><td>VCC = 1.6~3.6 V
*When using USB communication: VCC = 3.0~3.6 V</td></tr>
<tr><td>Operating mode</td><td>Boot mode</td></tr>
<tr><td>Flash memory</td><td>
<table>
<tr><td rowspan="2">Code Flash</td><td>User area</td><td>Max. 512 KB [2MAT]</td></tr>
<tr><td>User boot area</td><td>None</td></tr>
<tr><td rowspan="2">Data Flash</td><td>Data area</td><td>Max. 8 KB</td></tr>
<tr><td>Extended data area</td><td>None</td></tr>
</table>
</td></tr>
<tr><td>RAM</td><td>SRAM: 64 KB (used by the Boot firmware: 32 KB)</td></tr>
<tr><td>Communication method</td><td>[2-wire UART communication]
(Initial / Min) 9600 bps (Max) 2 Mbps(*)
*) Max when the VCC is lower than or equal to 1.8V is 115200bps
*) The max bps is determined by the VCC level at boot firmware’s startup processing, and returned by RMB of Signature request command

[USB communication]
12 Mbps
*USB communication operation is only guaranteed for use with Windows 10 as the host OS. Use with other host OS systems is not guaranteed.

[SWD communication]
(Max) 6 MHz(*)
*) Max when the VCC is lower than or equal to 1.8V is also 6 MHz, but communication handshake cannot be omitted.
See “3. Communication Method” for details.
*) Host can determine the current VCC with RMB value returned by Signature request command. When the returned RMB is 115200bps, the VCC is lower than or equal to 1.8V for this product.</td></tr>
</table>

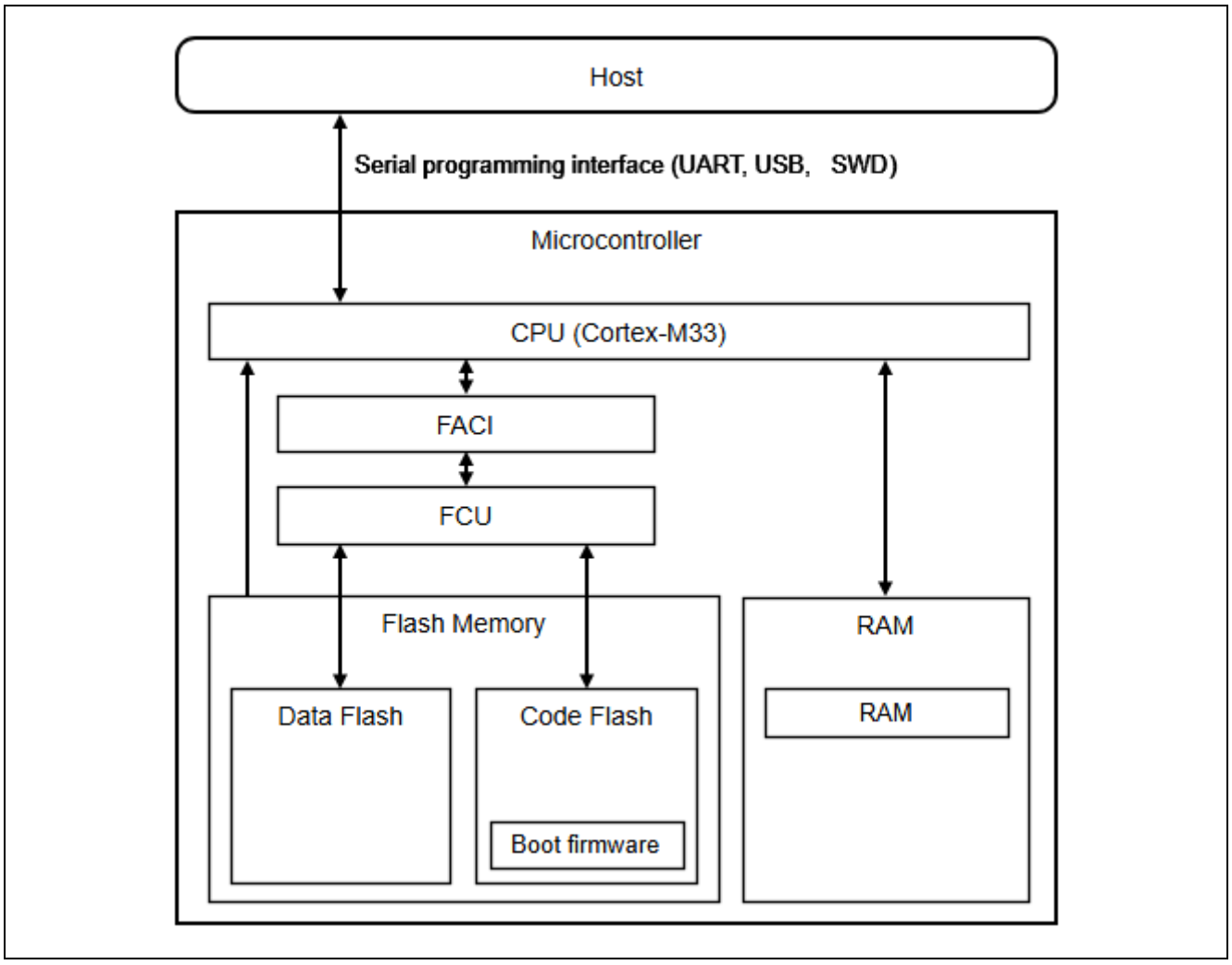


Figure 4. Block Diagram

3. Communication Methods

Boot firmware has interfaces for the following communication methods:

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

3.1 2-wire UART communication

Boot firmware supports the 2-wire UART communication, as shown in Figure 5.

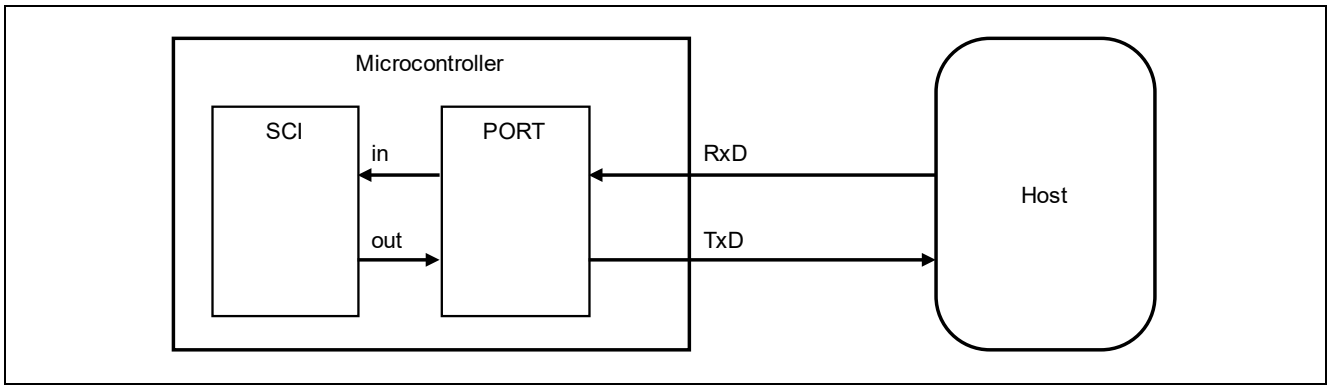


Figure 5. 2-Wire UART Communication

Table 4. UART Settings

General settings	
Interface	SCIM-7 ch9
RxD	P110, Input mode
TxD	P109, Output mode
Transfer rate	9600 bps (Min, until the Baudrate setting command) 2 Mbps (Max)
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 are not guaranteed.

3.2 Universal Serial Bus (USB) Communication

Boot firmware supports USB communication as shown in Figure 6.

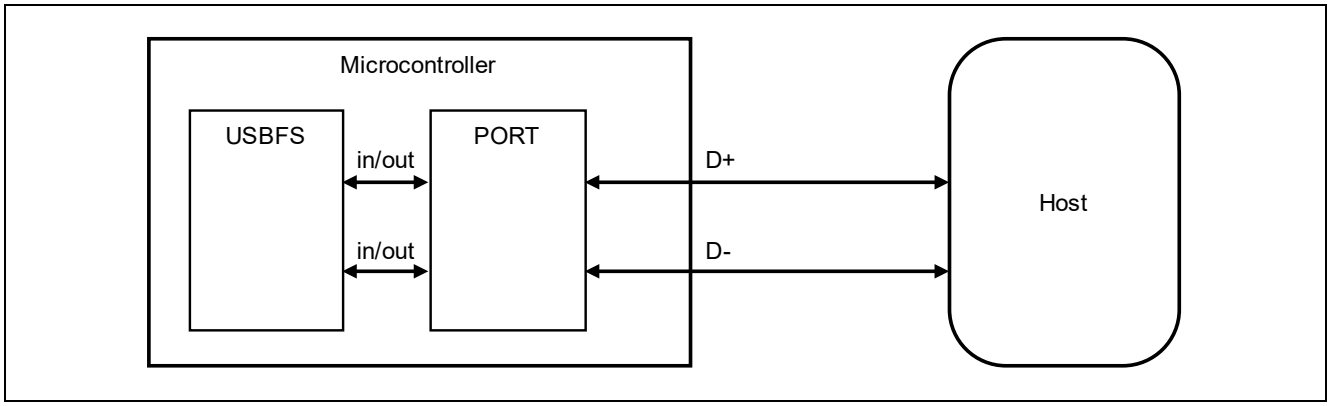


Figure 6. USB Communication

Table 5. USB Settings

Interface	USBFS
VBUS	P407, input mode
D+	Input-output mode
D-	Input-output mode
Transfer rate	12 Mbps (USB2.0 Full-Speed)
Device class	Communication device class (CDC) • SubClass: 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 SWD Communication

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

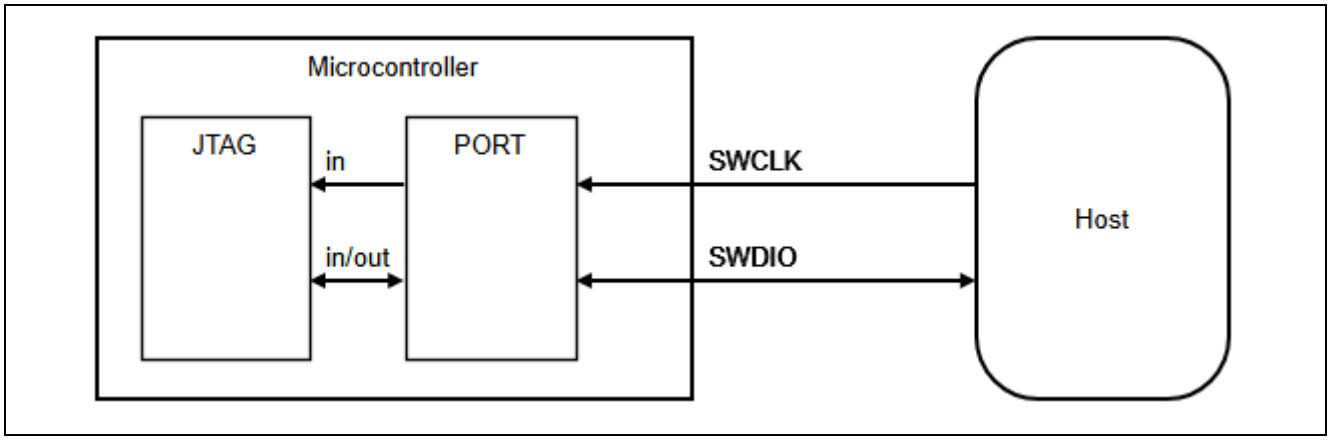


Figure 7. SWD Communication

Table 6. SWD Settings

General Settings	
[SWD] SWCLK	P300, Input mode
[SWD] SWDIO	P108, Input-Output mode
Transfer rate	6 MHz (Max)
Data length	32 bit
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 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

3.3.3 SWD Communication

It is prohibited to omit the Communication handshake above when operating in a low VCC environment. This is because boot firmware changes its operating clock depending on the VCC, as described in 2. System Architecture. Host can determine the VCC with RMB returned by the Signature request command. When the returned RMB is smaller than the maximum UART baudrate, do not omit the Communication handshake above. See the specifications of the Baudrate setting command for the maximum UART baudrate.

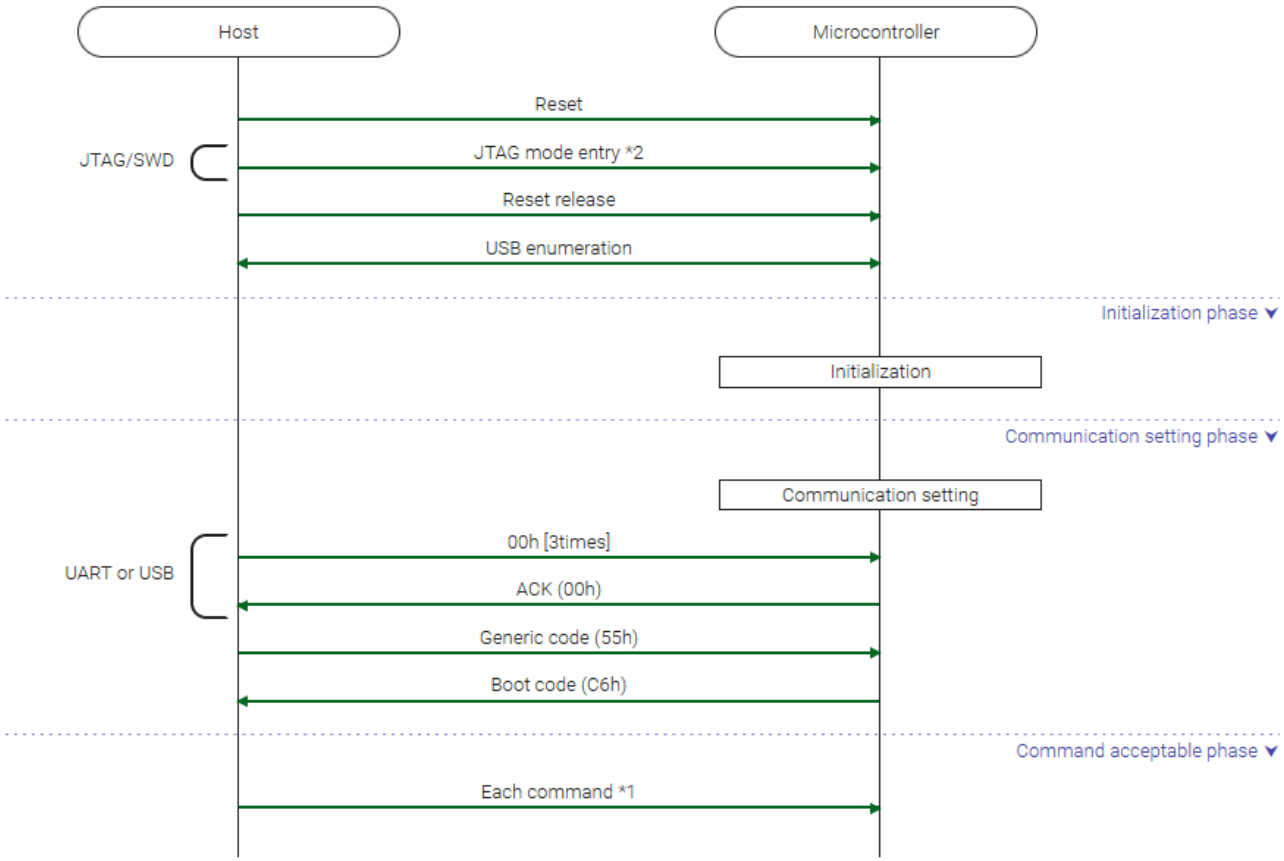
4. General Procedure

Boot firmware transits phases in the following order after the reset release:

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

The above sequence cannot be altered.

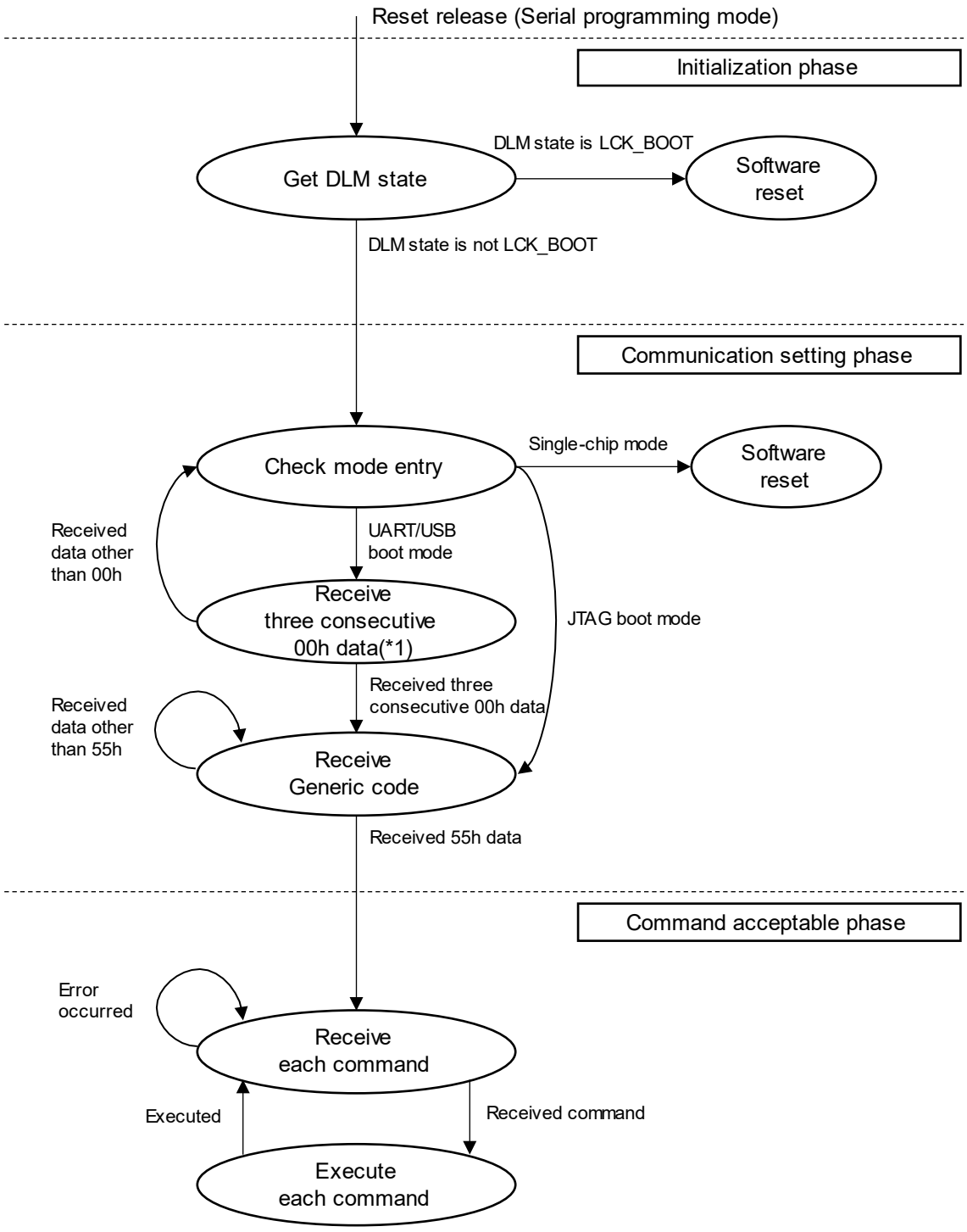
4.1 Sequence Diagram (Generic Sequence)



*1: From the Command acceptable phase, the host and microcontroller send data in turn unless otherwise noted. The host shall execute 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 will boot into mode.

Figure 8. Sequence Diagram (Generic Sequence)

4.2 State Transition Diagram (Generic State Transition)



*1: Boot firmware adopts IP (SCI or USBFS) that received 00h consecutively three times earlier as the communication method.

Figure 9. State Transition Diagram (Generic State Transition)

4.3 Initialization Phase

Boot firmware initializes hardware modules in this phase. After that, boot firmware transits to the "Communication setting phase".

4.3.1 Processing Procedure

Boot firmware initializes after reset release.

Boot firmware initializes hardware modules, then transits 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 7. After receiving the generic code using the established communication method, the boot firmware transitions to the "Command acceptable phase".

Table 7. 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 the terminal reset. MD pin level is high.	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
- When all the following conditions are met, SWD communication is determined to be selected.
  * When SWD communication is selected, boot firmware waits for the generic code without waiting for 00h.
  - MD=1
  - JBMDR=A5h
- When SWD communication is not selected, 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 AC Characteristics.
- After that, when the generic code is received, boot firmware sends a "Boot code".
  If a code other than the generic code is received, the boot firmware waits to receive the generic code again.
  The time from when reset is released until the generic code can be received is shown in AC Characteristics.
- 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 operating mode is serial programming mode, the boot firmware initializes 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 USB, the count value is reset.

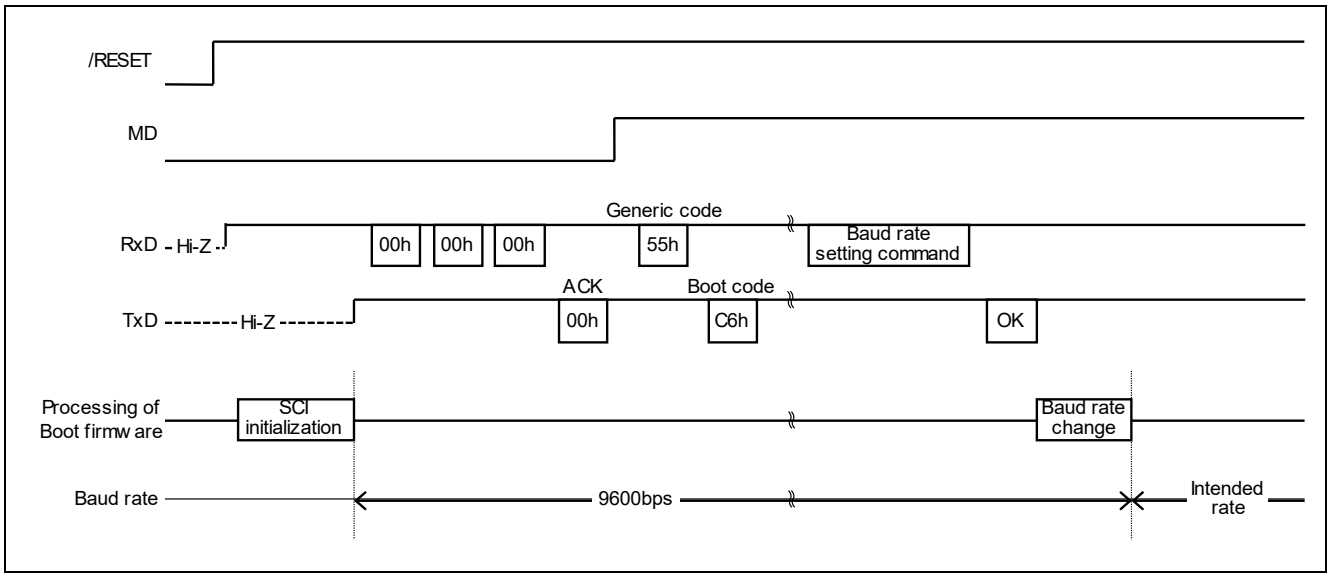


Figure 10. 2-wire UART Communication Setting

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. After SCI initialization, the boot firmware outputs High from TxD.

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 the reset release.

4.4.3 Settings of the USB Communication

When the device's operating mode 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.

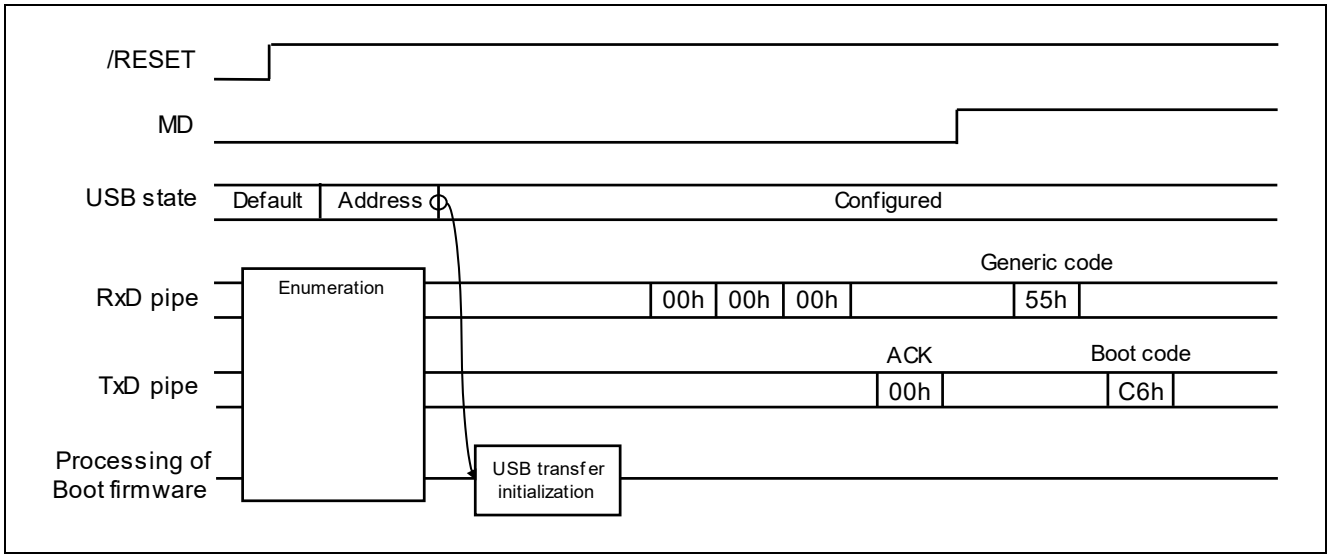


Figure 11. USB Communication Setting

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 SWD communication

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

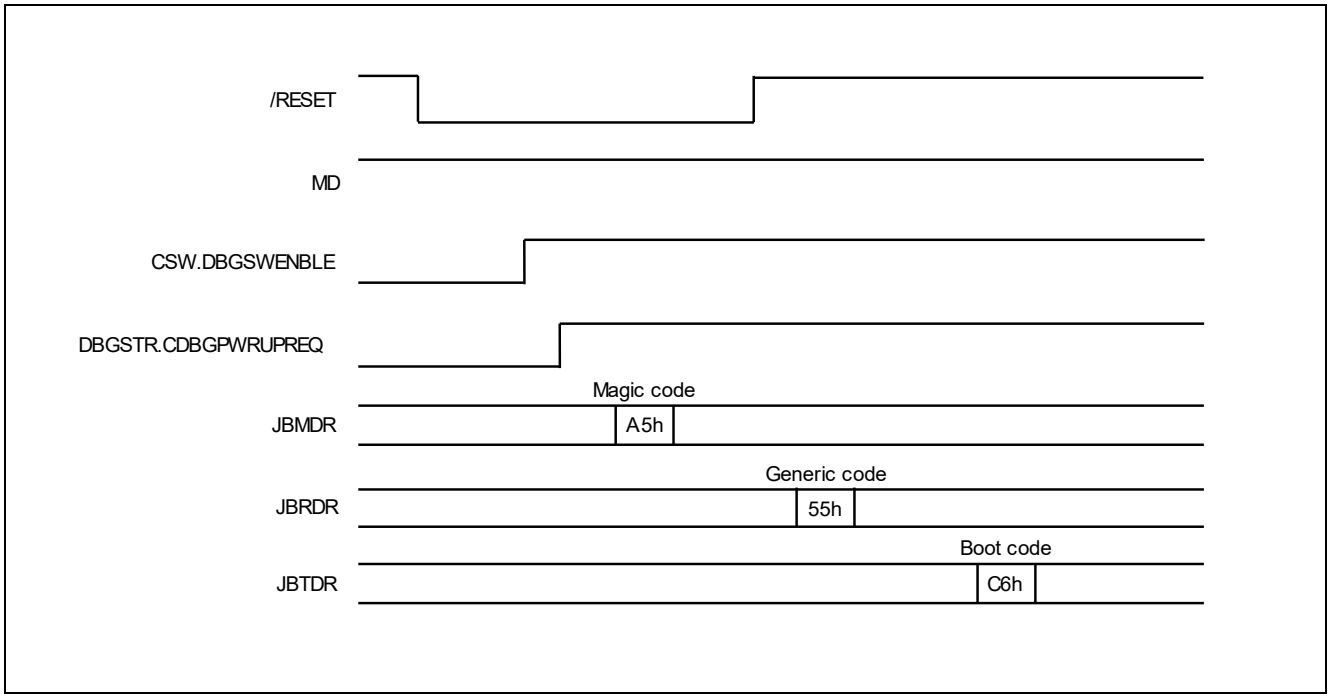


Figure 12. SWD Communication Setting

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 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 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 received command packet's LNH and LNL 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 received command packet's LNH and LNL 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 8. 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 9. 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 10. Command Codes

Value	Name	Description
71h	DLM state transit command	Authentication-free DLM transition.
2Ch	DLM state request command	Get the current DLM state.
30h	Authentication command	Authentication-required DLM 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.
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).
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.
1Ah	Encrypted data write command	Write encrypted data to target area.

5.1.5 RES: Response Code

Table 11. Response Codes

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

5.1.6 STS: Status Code

Table 12. Status Codes

Value	Name	Description
00h	Communication is normal [OK]	-
C0h	Unsupported command error	(*1) Received an unsupported command.
C1h	Packet error	(*1) Abnormality of packet format.
C2h	Checksum error	(*1) Abnormality of packet's checksum value.
D0h	Parameter error	(*1) Abnormality of packet parameter.
D5h	Command acceptance error	(*1) A command cannot execute in its current state.
D6h	DLM state unmatched error	(*1) Device reset is not asserted after the DLM state is changed.
D7h	Hardware error	(*1) Abnormality of memory value.
DAh	Protection error	(*1) Accessing protected areas or performing prohibited actions.
DBh	Trusted system error	(*1) Abnormality from the Trusted system.
E4h	Secure error	(*1) Access to an area that is inaccessible with current privilege.
E5h	Flash access error	(*1), (*2), (*3) Abnormality from the Flash sequencer.
E8h	Verify error	(*1) Verify that the written data fails.

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

5.1.7 ST2: Status Details

Table 13. Status Details

Value	Name	Description
FSTATR[31:0]	Flash status	When a Flash access error occurs, boot firmware returns the value of the FSTATR register. When not, boot firmware returns FFFFFFFFh. Boot firmware clears the FSTATR register after the status sending, so even when error(s) occur, the host can retry the next command without reset release.

5.1.8 ADR: Failure Address

Table 14. Failure Address

Value	Name	Description
00000000h–FFFFFFFFh	Failure address	When a Flash access error occurs, boot firmware returns the value of the start address of the flash sequencer command. When not, boot firmware returns FFFFFFFFh.

5.1.9 DLM: Device Lifecycle Management State Code

Table 15. DLM State Codes

Value	Name	Description
02h	SSD	Secure Software Development
03h	NSECSD	Non-SECure Software Development
04h	DPL	DePLoyed
05h	LCK_DBG	LoCKed DeBuG
06h	LCK_BOOT	LoCKed BOOT interface
07h	RMA_REQ	Return Material Authorization REQuest
08h	RMA_ACK	Return Material Authorization ACKnowledged

6. Command List

Table 16. Command List

Name	Communication Method	DLM State					
		SSD	NSECSD	DPL	LCK_DBG	LCK_BOOT	RMA_REQ
DLM State Transit Command	2-wire UART, USB, SWD	◎	◎	◎	◎	(*1)	
DLM State Request Command	2-wire UART, USB, SWD	◎	◎	◎	◎	(*1)	◎
Authentication Command	2-wire UART, USB, SWD	◎	◎	◎		(*1)	◎
Key Setting Command	2-wire UART, USB, SWD	◎	◎			(*1)	
User Key Setting Command	2-wire UART, USB, SWD	◎	◎			(*1)	
Key Verify Command	2-wire UART, USB, SWD	◎	◎	◎	◎	(*1)	◎
User Key Verify Command	2-wire UART, USB, SWD	◎	◎	◎	◎	(*1)	◎
Initialize Command	2-wire UART, USB, SWD	◎	◎	◎		(*1)	
Boundary Setting Command	2-wire UART, USB, SWD		◎			(*1)	
Boundary Request Command	2-wire UART, USB, SWD	◎	◎	◎	◎	(*1)	◎
Parameter Setting Command	2-wire UART, USB, SWD	◎	◎	◎		(*1)	
Parameter Request Command	2-wire UART, USB, SWD	◎	◎	◎	◎	(*1)	◎
Inquiry Command	2-wire UART, USB, SWD	◎	◎	◎	◎	(*1)	◎
Signature Request Command	2-wire UART, USB, SWD	◎	◎	◎	◎	(*1)	◎
Area Information Request Command	2-wire UART, USB, SWD	◎	◎	◎	◎	(*1)	◎
Baudrate Setting Command	2-wire UART, USB, SWD	◎	◎	◎	◎	(*1)	◎
Erase Command	2-wire UART, USB, SWD	◎	◎			(*1)	
Write Command	2-wire UART, USB, SWD	◎	◎			(*1)	
Read Command	2-wire UART, USB, SWD	◎	◎			(*1)	
CRC Command	2-wire UART, USB, SWD	◎	◎	◎	◎	(*1)	◎
Encrypted Data Write Command	2-wire UART, USB, SWD	◎				(*1)	

Notes:
◎ : 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 the 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:

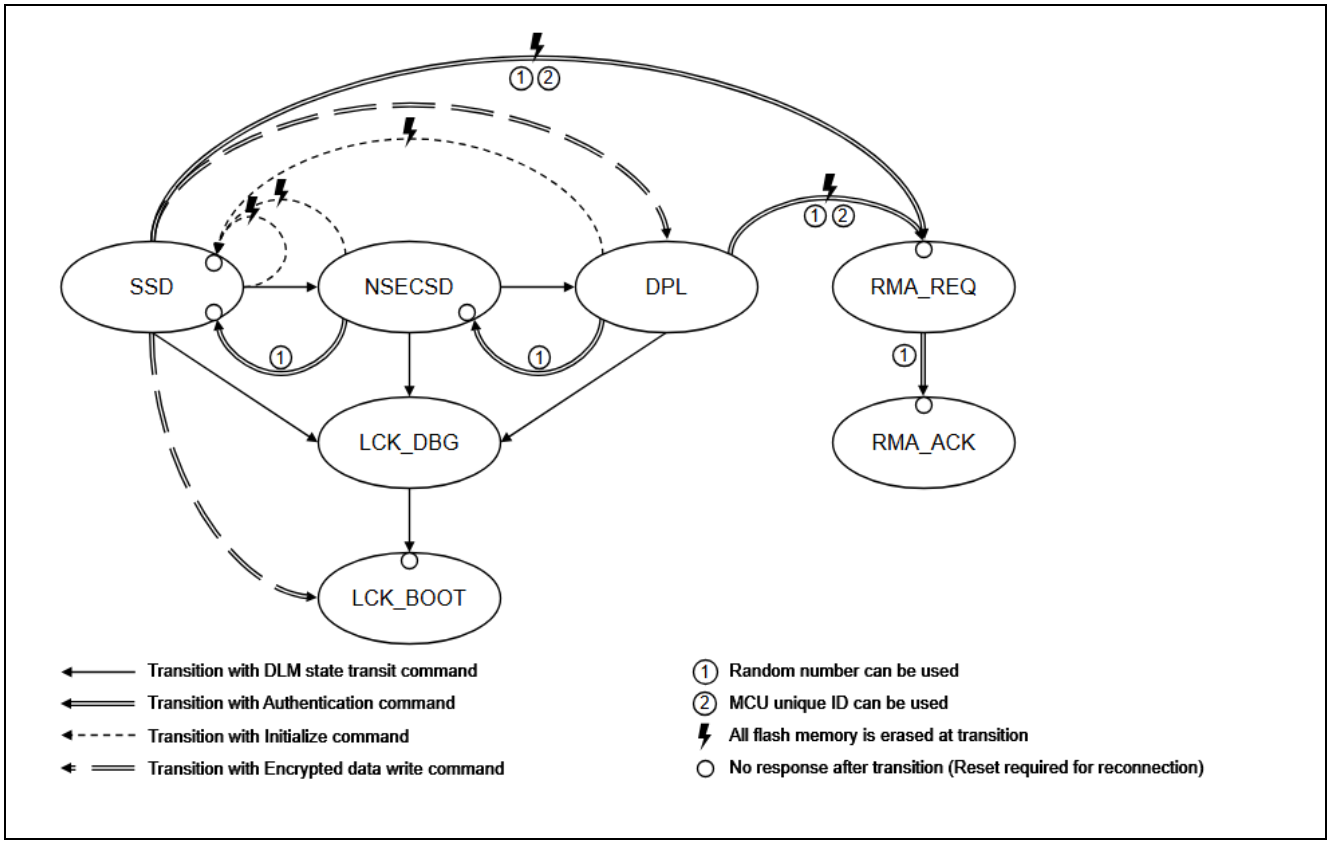


Figure 13. DLM State Transitions

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.

This command requires adherence to conditions described in Command List.

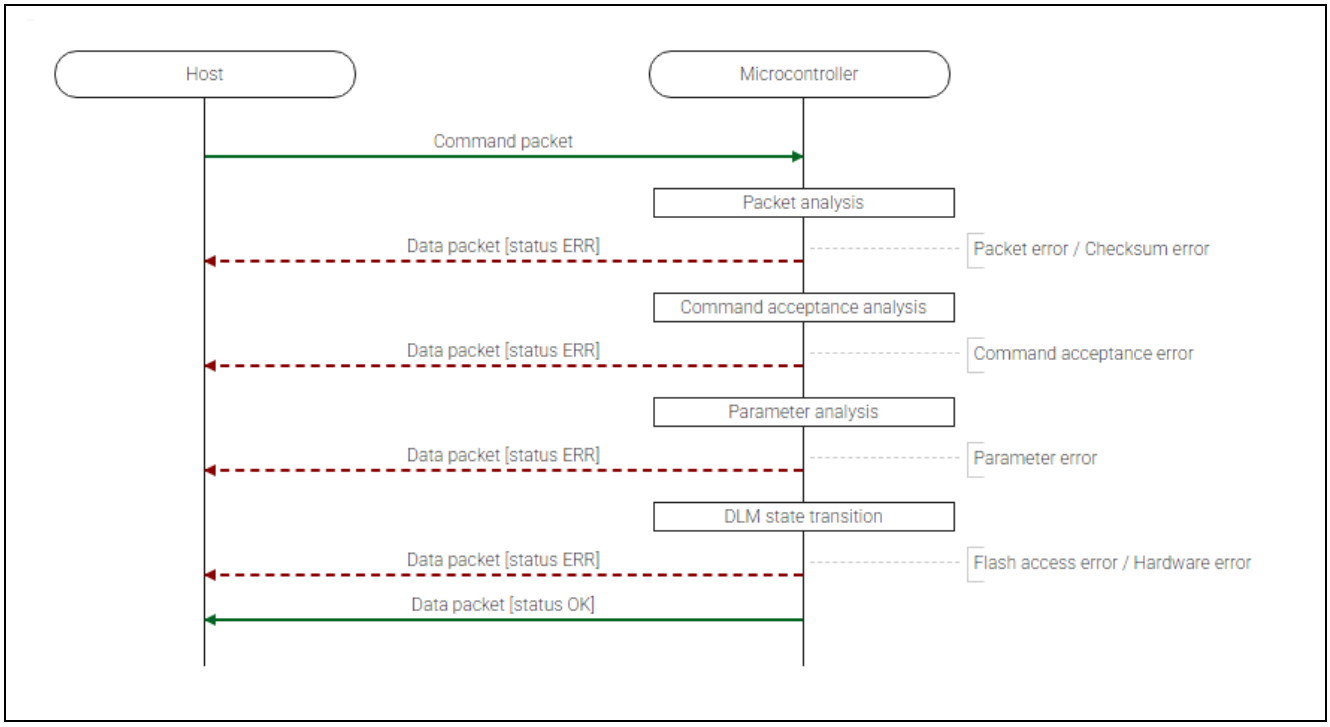


Figure 14. DLM State Transit Command Sequence Diagram

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: • 02h: SSD • 03h: NSECSD • 04h: DPL • 05h: LCK_DBG
DDLM	(1 byte)	Destination DLM state code: • 03h: NSECSD • 04h: DPL • 05h: LCK_DBG • 06h: LCK_BOOT
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 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 received command packet's LNH and LNL are different from the values specified in the packet format, the boot firmware sends a "Packet error".
- If the received command packet's LNH and LNL 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, 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 a "Flash access error" and waits for the next command.
  * Check the DLM state after the Flash access 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, the boot firmware will send "OK" and will not respond.
- When DLM state transit is successfully completed, "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 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
Source DLM state code is different from the current DLM state.	Parameter error	FFFFFFFFh	FFFFFFFFh
Destination DLM state code is not a transitionable DLM state.	Parameter error	FFFFFFFFh	FFFFFFFFh
FACI detected an error after the command execution in not disclosed area.	Flash access error	Flash status	FFFFFFFFh
DLM state is abnormal.	Hardware error	FFFFFFFFh	FFFFFFFFh
Successful completion.	OK	FFFFFFFFh	FFFFFFFFh

6.2.4 DLM State Transition

Figure 15 shows the DLM states that can be transited by the DLM State Transit command.

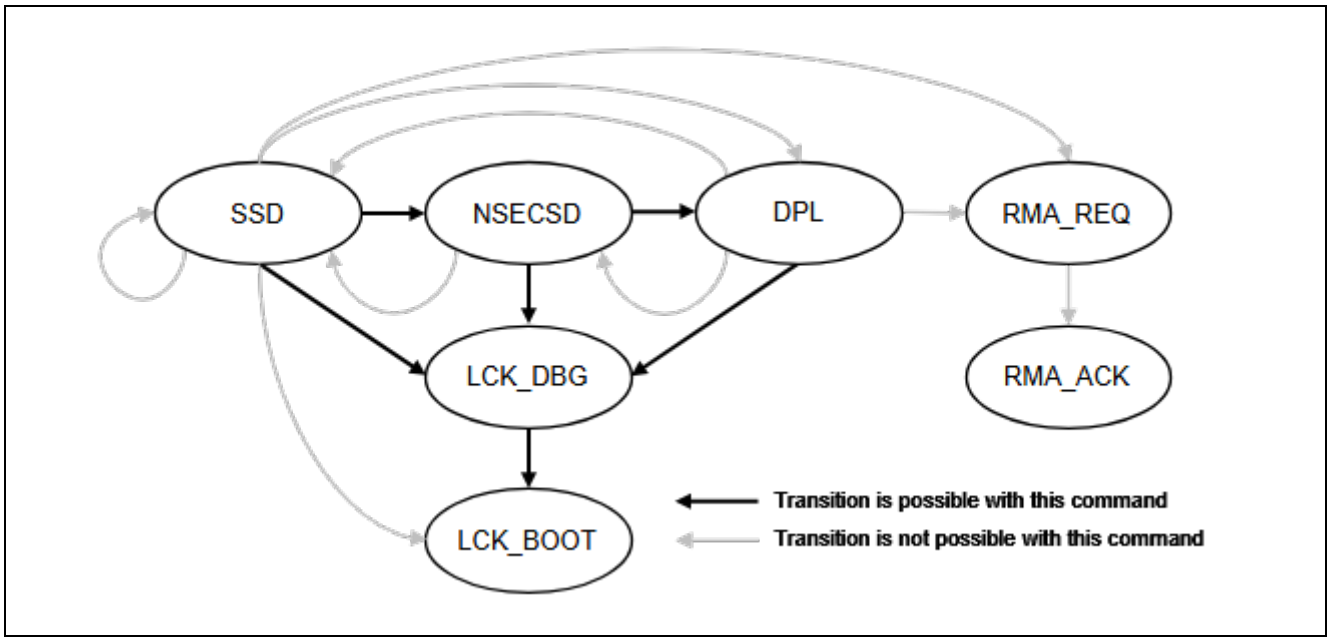


Figure 15. Valid State Transitions for DLM State Transit Command

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

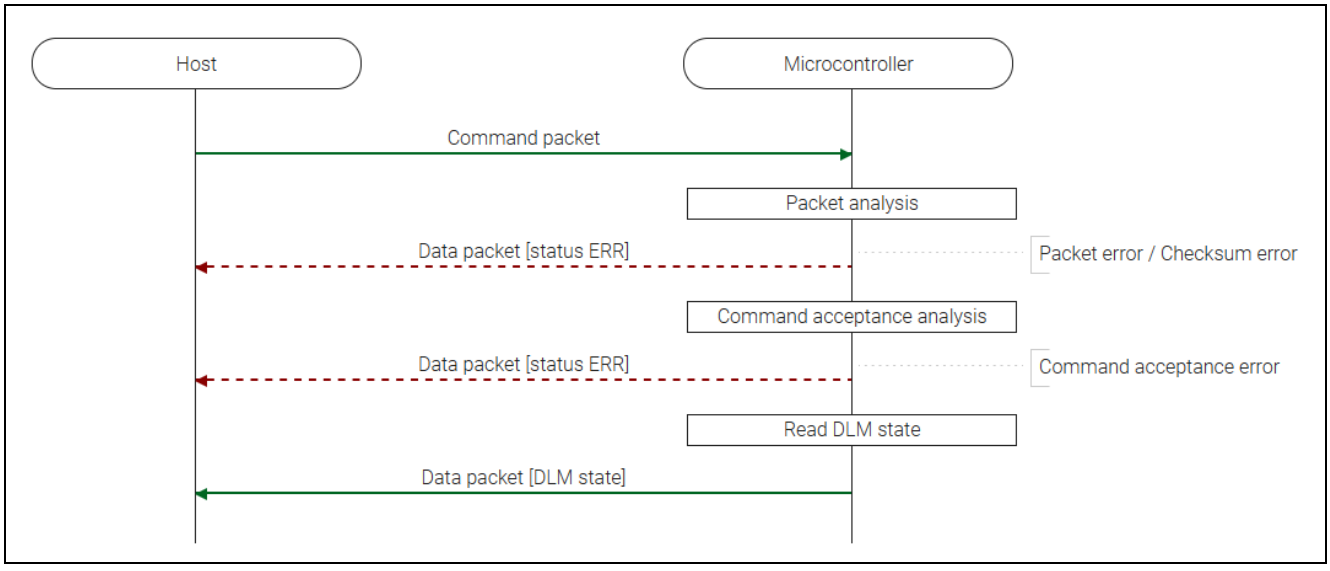


Figure 16. DLM State Request Command Sequence Diagram

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 received command packet's LNH and LNL are different from the values specified in the packet format, the boot firmware sends a "Packet error".
- If the received command packet's LNH and LNL 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 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 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

6.4 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 flash 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).

This command requires adherence to conditions described in Command List.

6.4.1 Sequence Diagram

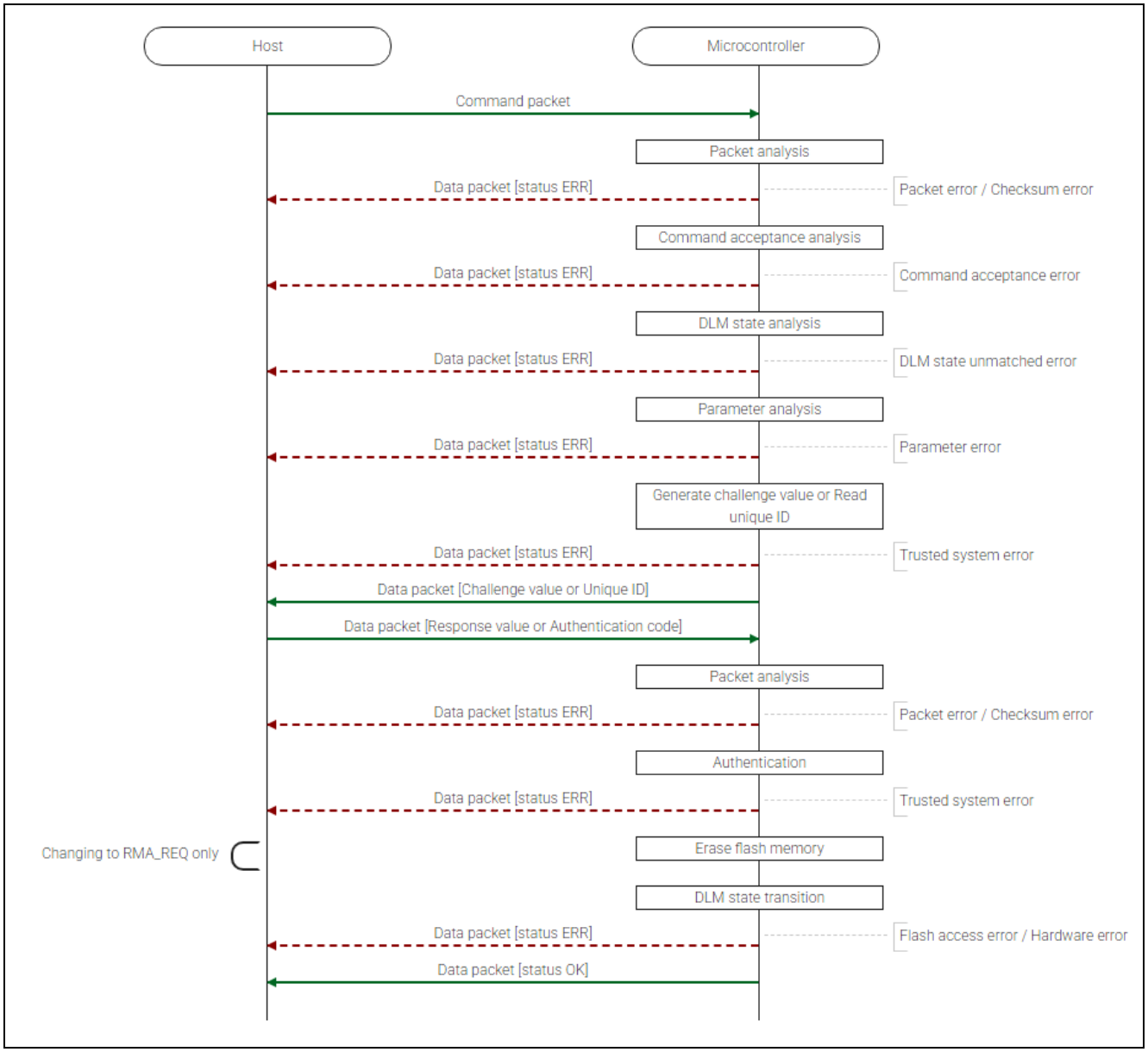


Figure 17. Authentication Command Sequence Diagram

6.4.2 Packets

6.4.2.1 Command Packet

SOH	(1 byte)	01h
LNH	(1 byte)	00h
LNL	(1 byte)	04h
CMD	(1 byte)	30h (Authentication command)
SDLM	(1 byte)	• 02h: SSD • 03h: NSECSD • 04h: DPL • 07h: RMA_REQ
DDLM	(1 byte)	• 02h: SSD • 03h: NSECSD • 07h: RMA_REQ • 08h: RMA_ACK
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

6.4.2.2 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

6.4.2.3 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 details of the Response value, refer to Response Value Calculation.
SUM	(1 byte)	Sum data
ETX	(1 byte)	03h

6.4.2.4 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

6.4.2.5 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

6.4.3 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 received command packet's LNH and LNL are different from the values specified in the packet format, the boot firmware sends a "Packet error".
- If the received command packet's LNH and LNL 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 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 a "Parameter error".
- When SDLM and DDLM are not a transitional combination, boot firmware returns a "Parameter error".
- When CHCT is not a challenge type that can be used to transition the DLM state; boot firmware returns 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.

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, a "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 the 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.

Note: This command erases the memory even if initialization is disabled. (Refer to the Parameter request command.)

- If an error occurs during erasure in the Block, protect the setting, the boot firmware sends a "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 does not respond.
  * The value of the Block protect setting is undefined.
- If an error occurs during erasure in the User area, the boot firmware sends a "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 does not respond.
  * The value of the area after ADR (Failure address) of the User area is undefined.
- If an error occurs during erasure in the Data area, the boot firmware sends a "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 does not respond.
  * The value of the Data area is undefined.
- If an error occurs during erasure in the Config area, the boot firmware sends a "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 does not respond.
  * The value of the Config area is undefined.
- If an error occurs during erasure in the EEP Config area, the boot firmware sends a "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 does not respond.
  * The value of the EEP Config area is undefined.
- If an error occurs during boundary setting and Key index (Wrapped key) erasure in the User area, the boot firmware sends a "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 does not respond.

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

- If the Trusted system becomes abnormal during the 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 the transition, boot firmware returns "Flash access 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 erased except below:
  - Blocks for which "0" is set for permanent block protection setting (PBPS, PBPS_SEC).
    * Not affected by block protection settings (BPS, BPS_SEC).
  - All Data areas are erased.
  - The Config area is written the value when shipped except the following:
    - Permanent block protection setting (PBPS, PBPS_SEC).
    - Block protection setting (BPS, BPS_SEC) for blocks on which "0" is set for permanent block protection setting (PBPS, PBPS_SEC).

- Secure Attribute setting for block protection (BPS_SEL).
- FSPR and BTFLG when FSPR = 0.

6.4.4 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 currently active DLM state does not match the stored DLM state.	DLM states unmatched error	FFFFFFFFh	FFFFFFFFh
SDLM is different from the current DLM state or AL.	Parameter error	FFFFFFFFh	FFFFFFFFh
SDLM and DDLM are not a transitionable combination.	Parameter 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
FACI detected an error after the command execution in the disclosed area.	Flash access error	Flash status	Failure address
FACI detected an error after the command execution in an undisclosed area.	Flash access error	Flash status	FFFFFFFFh
DLM state is abnormal.	Hardware error	FFFFFFFFh	FFFFFFFFh
Successful completion.	OK	FFFFFFFFh	FFFFFFFFh

6.4.5 Authentication Level Transition

Figure 18 shows the Authentication level that can be transit by this command.

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

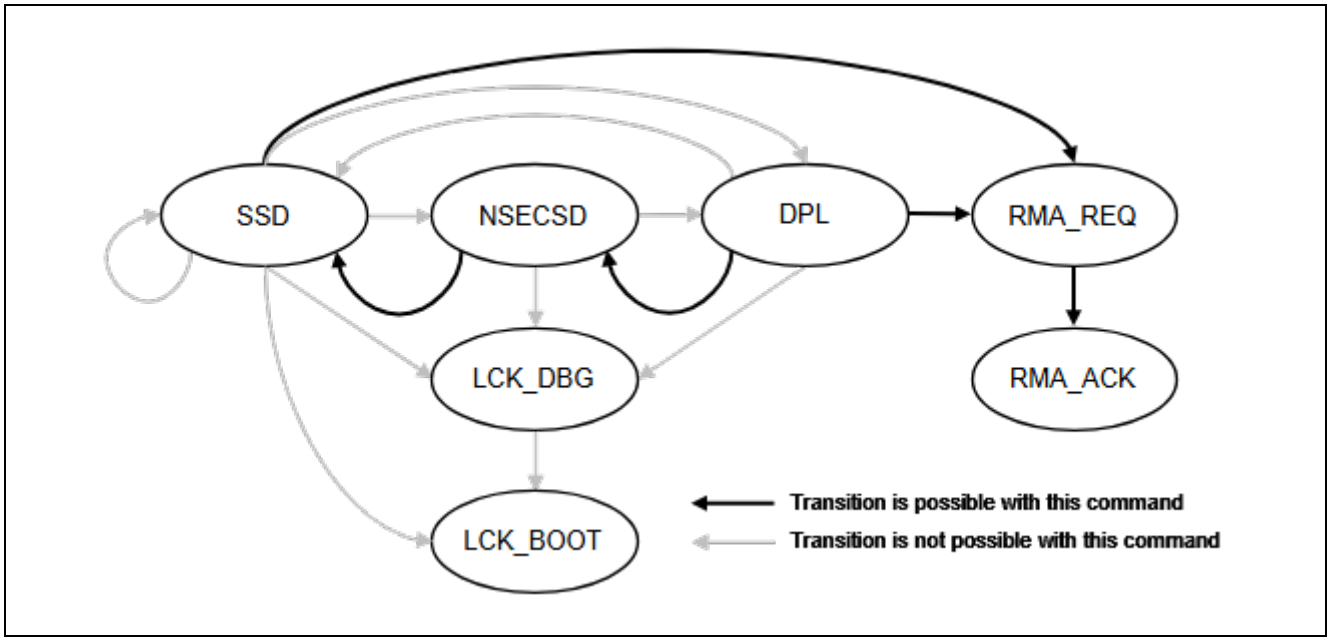


Figure 18. Valid Authentication Level Transitions

Source DLM state	Destination DLM state	Required key
SSD	RMA_REQ	RMA_Key
NSECSD	SSD	SECDBG_KEY
DPL	NSECSD	NONSECDBG_KEY
	RMA__REQ	RMA_KEY
RMA_REQ	RMA_ACK	RMA_ACK_KEY

6.4.6 Response Value Calculation

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

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

6.5 Key Setting Command

This command sets the authentication key to the device. The authentication key must be specified in the DLM state to be able to set the key.

This command requires adherence to conditions described in Command List.

6.5.1 Sequence Diagram

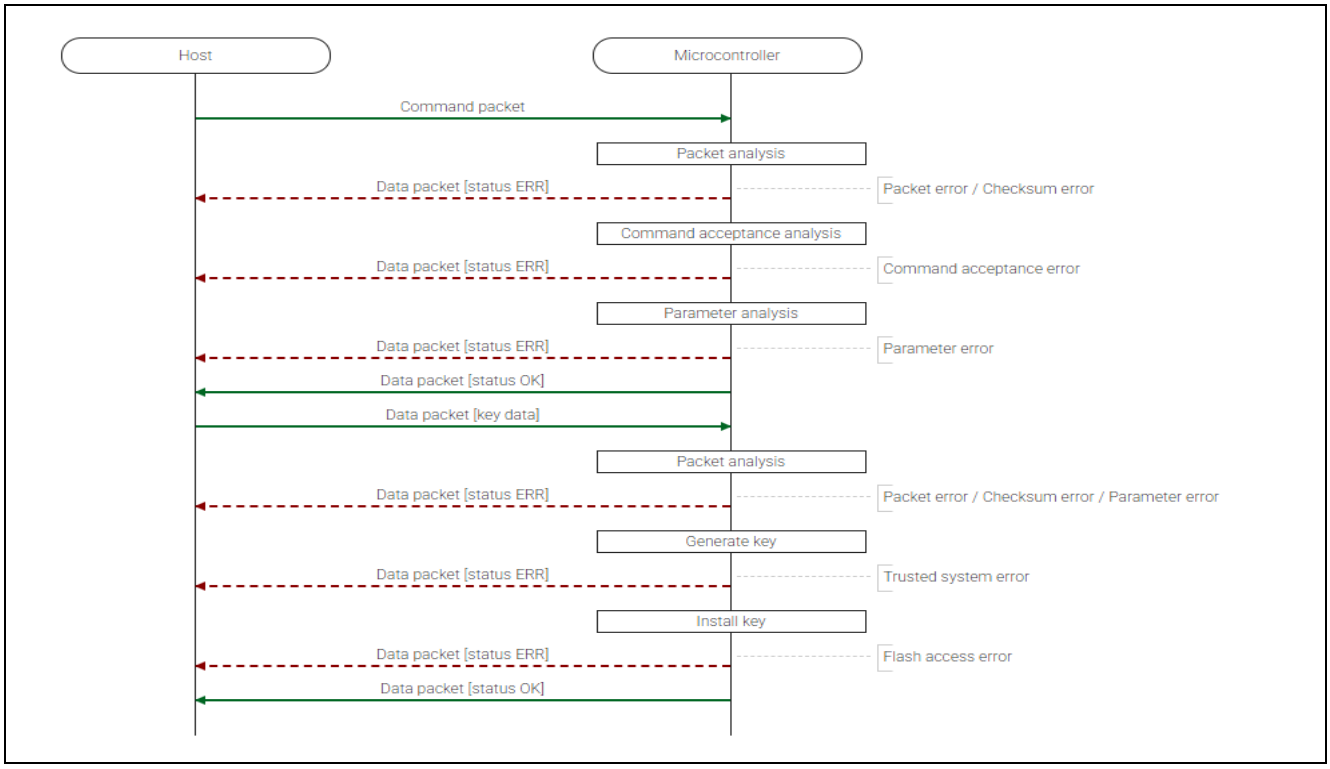


Figure 19. Key Setting Command Sequence Diagram

6.5.2 Packets

6.5.2.1 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: SECDBG_KEY • 02h: NONSECDBG_KEY • 03h: RMA_KEY
SUM	(1 byte)	Sum data
ETX	(1 byte)	03h

6.5.2.2 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 (bytes 0–15) + MAC (bytes 16-31)
For example: If 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>

6.5.2.3 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

6.5.2.4 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

6.5.3 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 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 received command packet's LNH and LNL are different from the values specified in the packet format, the boot firmware sends a "Packet error".
- If the received command packet's LNH and LNL 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, boot firmware executes the parameter analysis:

- KYTY is unspecified value
- KYTY is not able to be set in the current DLM state
  * 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, 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, a "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 in the command in the received data packet, "Parameter 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 generates the 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 the Key index to memory:

- If an error occurs while writing the Key index (Wrapped key), the boot firmware sends a "Flash access error" and returns to the command wait state.
  * Use the Key verify command to check the status of the Key index (Wrapped key) after the Flash access error occurs.
- When the authentication key setting is successfully completed, boot firmware returns "OK" and waits for the next command.

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 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 specified Key type is an unspecified value.	Parameter error	FFFFFFFFh	FFFFFFFFh
The specified Key type cannot be inserted at the current Authentication level.	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 value specified in the command.	Parameter error	FFFFFFFFh	FFFFFFFFh
Authentication key generate failed.	Trusted system error	FFFFFFFFh	FFFFFFFFh
FACI detected an error after the command execution in not disclosed area.	Flash access error	Flash status	FFFFFFFFh
Successful completion.	OK	FFFFFFFFh	FFFFFFFFh

6.5.5 Key type that can be set in each DLM state

Table 177 shows the following DLM state that can be set to Key type.

Table 17. Key Types for each Authentication Level

DLM state	Key type
SSD	SECDBG_KEY NONSECDBG_KEY RMA_KEY
NSECSD	NONSECDBG_KEY

6.6 User Key Setting Command

This command generates a Key index (Wrapped key) using the 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).

The storage area must be erased in advance.

This command requires adherence to conditions described in Command List.

6.6.1 Sequence Diagram

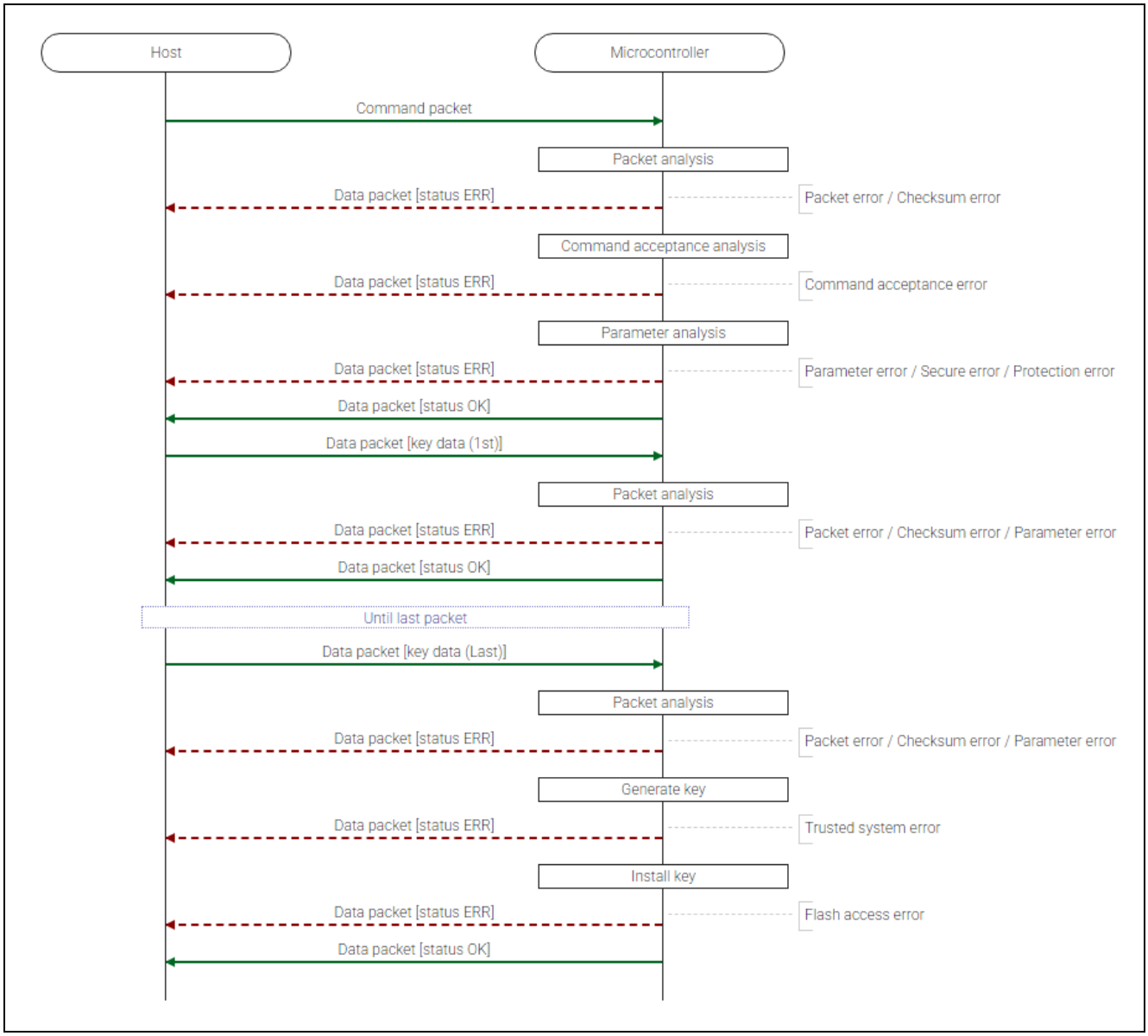


Figure 20. User Key Setting Command Sequence Diagram

6.6.2 Packets

6.6.2.1 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

6.6.2.2 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

*) Do not send SKR, ESKY, IVEC, and ENKY separately with multiple packets, except RSA-4096 Private key.

For RSA-4096 Private key, send first 972 bytes of the Install data with the first packet and send the remaining 68 bytes with the second packet.

6.6.2.3 Data Packet [Key Data (2nd~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

6.6.2.4 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

6.6.2.5 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

6.6.3 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 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 received command packet's LNH and LNL are different from the values specified in the packet format, the boot firmware sends a "Packet error".
- If the received command packet's LNH and LNL 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:

- If ENTY is not specified as the Key type, the boot firmware will send 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 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, a "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 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 the dedicated area:

- If an error occurs while writing Key index (Wrapped key), the boot firmware sends a "Flash access error" and returns to the command wait state.
  * WAU size from failure address (ADR) of memory area are undefined.
- If the key index (Wrapped key) is successfully saved to the device, the boot firmware sends "OK" and returns to the command wait state.

6.6.4 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 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
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 Kinds 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 current DLM state is NSECSD and Key setting address contains a Secure region..	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
FACI detected an error after the command execution.	Flash access error	Flash status	Failure address
Successful completion.	OK	FFFFFFFFh	FFFFFFFFh

6.6.4.1 User Key List

The list of user keys specified by this command is shown in Table 18.

Table 18. User Key List

Key type	Installation key	Install data size (bytes)	Key index size (bytes)
05h	AES-128	32	36
07h	AES-256	48	52
08h	AES-128 XTS	48	52
09h	AES-256 XTS	80	84
16h	ECC P256 Public key	80	84
17h	ECC P256 Private key	48	52
1Ah	HMAC-SHA224	48	52
1Bh	HMAC-SHA256	48	52
FFh	Key update key	48	52

6.7 Key Verify Command

This command verifies the authentication key that setting to device.

This command require adherence to conditions described in Command List.

6.7.1 Sequence Diagram

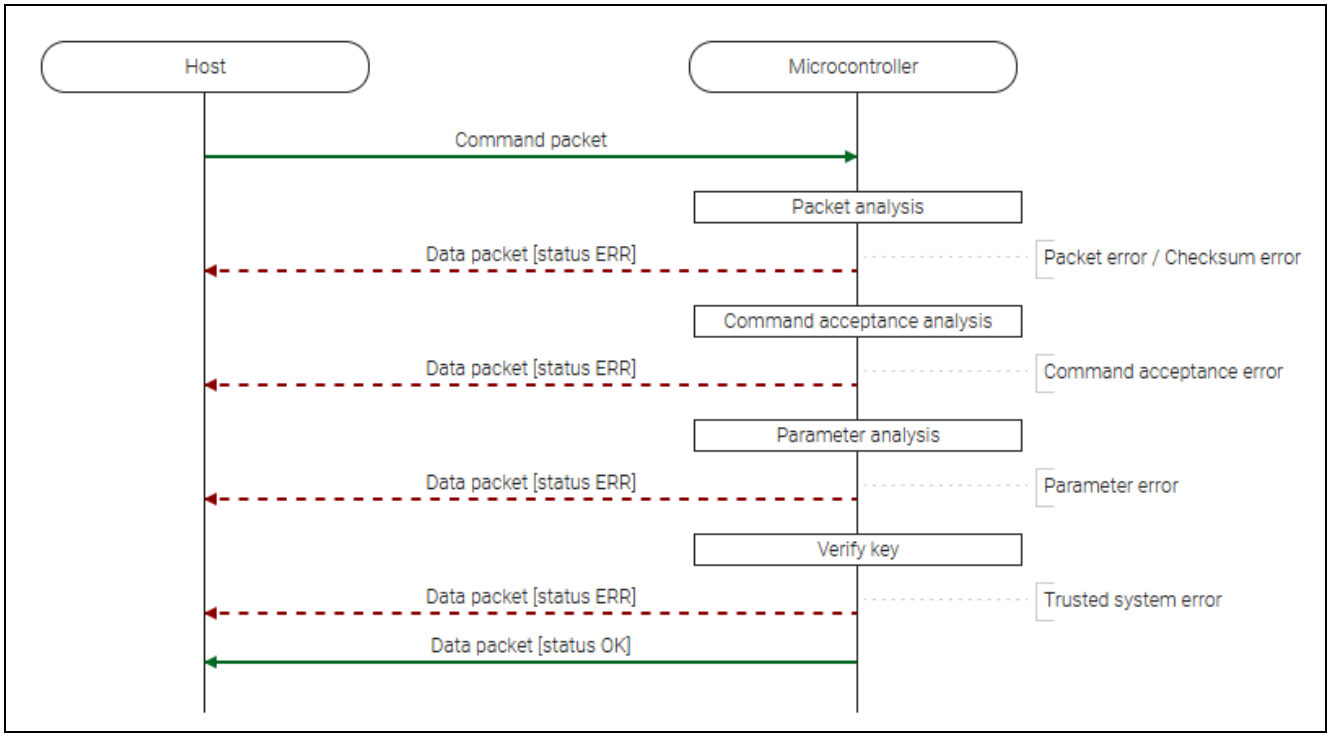


Figure 21. Key Verify Command Sequence Diagram

6.7.2 Packets

6.7.2.1 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: SECDBG_KEY • 02h: NONSECDGB_KEY • 03h: RMA_KEY
SUM	(1 byte)	Sum data
ETX	(1 byte)	03h

6.7.2.2 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

6.7.2.3 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

6.7.2.4 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 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 received command packet's LNH and LNL are different from the values specified in the packet format, the boot firmware sends a "Packet error".
- If the received command packet's LNH and LNL 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 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 the Key index (Wrapped key).

- If verification of the 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 the verification of the
  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.

6.7.3 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 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
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

6.8 User Key Verify Command

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

This command requires adherence to conditions described in Command List.

6.8.1 Sequence Diagram

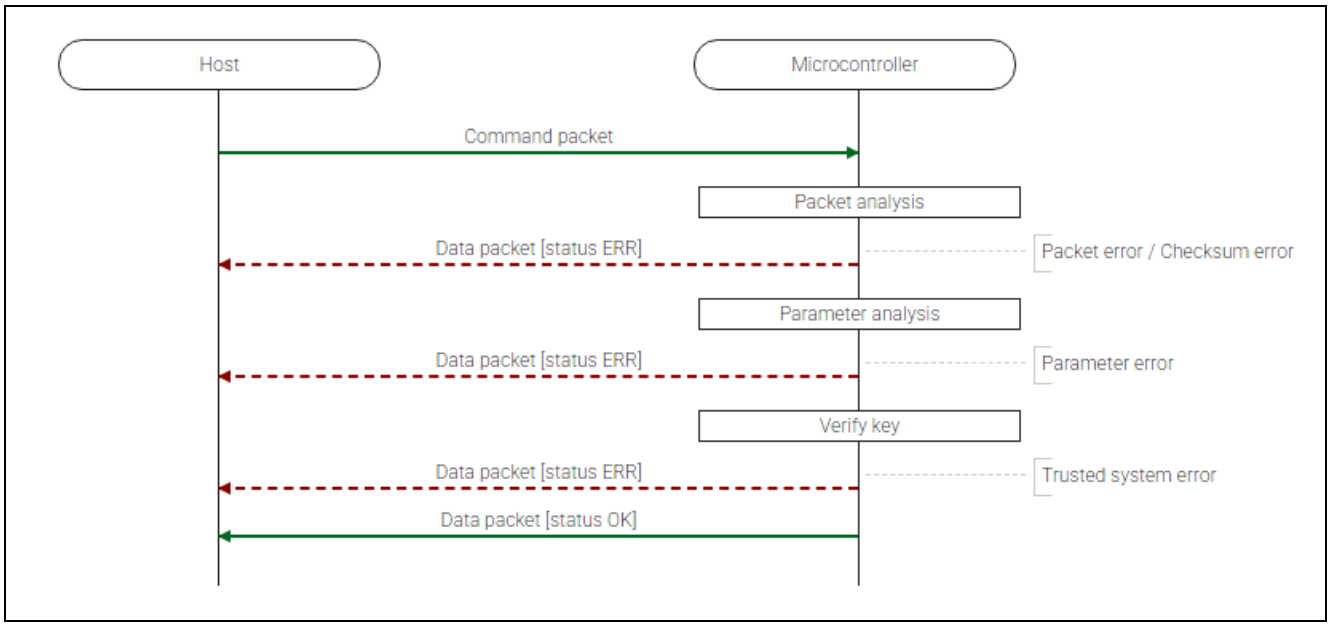


Figure 22. User Key Verify Command Sequence Diagram

6.8.2 Packets

6.8.2.1 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

6.8.2.2 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)	D6h
ETX	(1 byte)	03h

6.8.2.3 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

6.8.3 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 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 received command packet's LNH and LNL are different from the values specified in the packet format, the boot firmware sends a "Packet error".
- If the received command packet's LNH and LNL 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 parameter analysis:

- If ENTY is not specified as the Key type, the boot firmware will send 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".
- 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 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.

6.8.4 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 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
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 Data area specified by area information.	Parameter error	FFFFFFFFh	FFFFFFFFh
The area from the Key address to the Key index size spans different Kinds 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
Key index verify failed.	Trusted system error	FFFFFFFFh	FFFFFFFFh
Successful completion.	OK	FFFFFFFFh	FFFFFFFFh

6.9 Initialize Command

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

- User area
- Data area
- Config area
- 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).

This command require adherence to conditions described in Command List.

6.9.1 Sequence Diagram

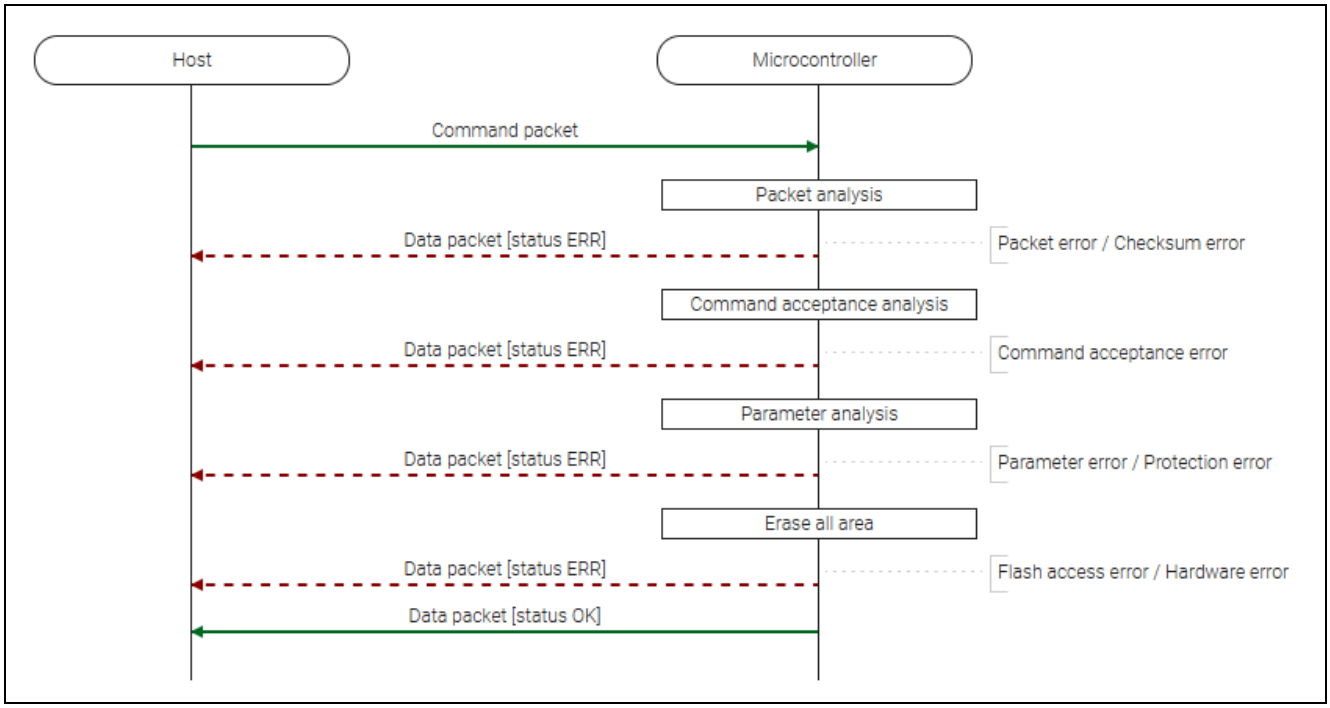


Figure 23. Initialize Command Sequence Diagram

6.9.2 Packets

6.9.2.1 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: • 02h: SSD • 03h: NSECSD • 04h: DPL
DDLM	(1 byte)	Destination DLM state code: • 02h: SSD
SUM	(1 byte)	Sum data
ETX	(1 byte)	03h

6.9.2.2 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

6.9.2.3 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

6.9.3 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 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 received command packet's LNH and LNL are different from the values specified in the packet format, the boot firmware sends a "Packet error".
- If the received command packet's LNH and LNL 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 SSD, "Parameter error" is returned.
- When initialization is disabled, "Protection error" is returned.
- When Permanent protected block exists (There is a bit that is "0" in PBPS[139:0] and PBPS_SEC[139:0]), "Protection error" is returned.
- When the FSPR bit is 0, the "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 contents do not change before command reception.

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

- If an error occurs while initializing the Block, protect the setting, the boot firmware sends a "Flash access error" and returns to the command wait state.
  * The value of the Block protect setting is undefined.
- If an error occurs while initializing the User area, the boot firmware sends a "Flash access error" and returns to the command wait state.
  * The value of the area after ADR (Failure address) of the User area is undefined.
- If an error occurs while initializing the Data area, the boot firmware sends a "Flash access error" and returns to the command wait state.
  * The value of the Data area is undefined.
- If an error occurs while initializing the Config area, the boot firmware sends a "Flash access error" and returns to the command wait state.
  * The value of the Config area is undefined.
- If an error occurs while initializing boundary setting and Key index (Wrapped key), the boot firmware sends a "Flash access error" and returns to the command wait state.
- If an error occurs during the transition Protection level, boot firmware returns "Flash access error" and waits for the next command.
  * Check the Protection level after the Flash access error has occurred with the Protection level request command.
- 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: Erased.
    - Data area: Erased.
    - Config area: Value when shipped, except for that reserved area, is not changed before command execution.

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 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
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
There is a permanently protected block.	Protection error	FFFFFFFFh	FFFFFFFFh
The FSPR bit is set. (FSPR = 0)	Protection error	FFFFFFFFh	FFFFFFFFh
FACI detected an error after the command execution in disclosed area.	Flash access error	Flash status	Failure address
FACI detected an error after the command execution in not disclosed area.	Flash access error	Flash status	FFFFFFFFh
DLM state is abnormal.	Hardware error	FFFFFFFFh	FFFFFFFFh
Successful completion.	OK	FFFFFFFFh	FFFFFFFFh

6.9.5 Protection Level Transition

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

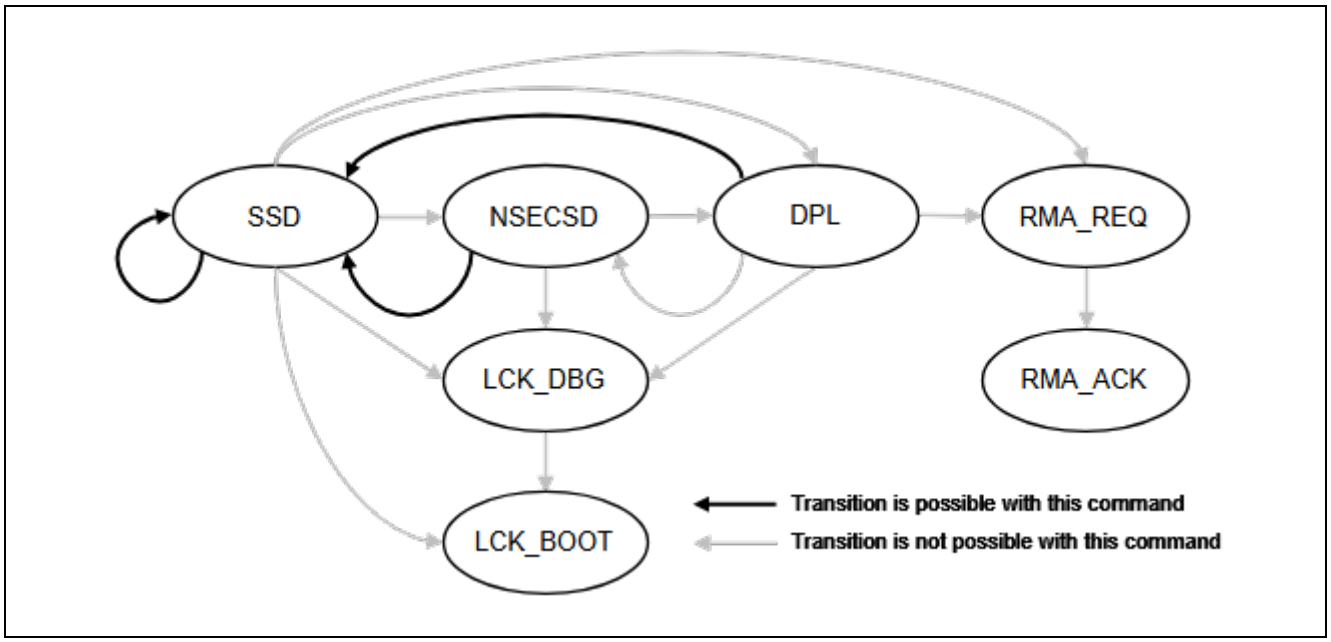


Figure 24. Protection Level Transitions

6.10 Boundary Setting Command

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

This command requires adherence to conditions described in Command List.

6.10.1 Sequence Diagram

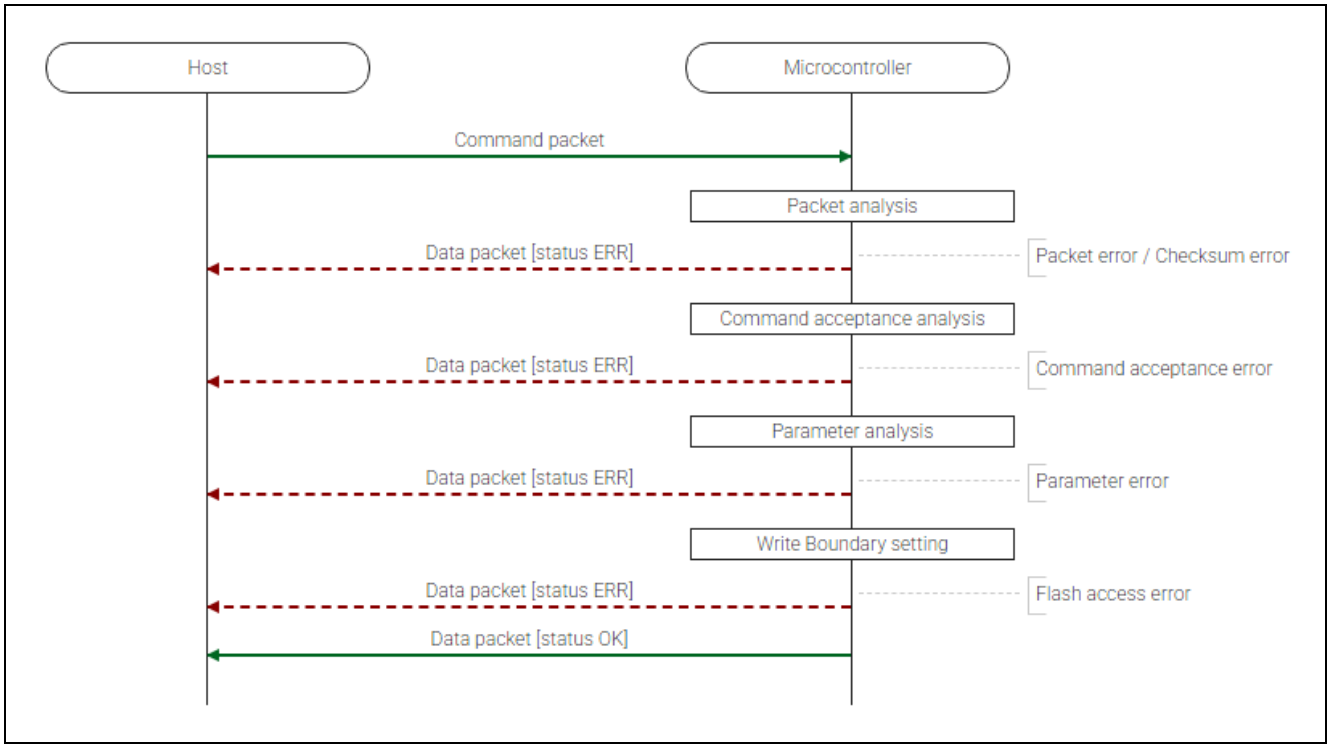


Figure 25. Boundary Setting Command Sequence Diagram

6.10.2 Packets

6.10.2.1 Command Packet

SOH	(1 byte)	01h
LNH	(1 byte)	00h
LNL	(1 byte)	0Bh
CMD	(1 byte)	4Eh (Boundary setting command)
CFS1	(2 byte)	Size of Code Flash Secure region (without NSC) [KB] e.g.) 0100h -> 01h, 00h (256KB)
CFS2	(2 byte)	Size of Code Flash Secure region [KB] e.g.) 0100h -> 01h, 00h (256KB) * 32KB align
DFS1	(2 byte)	Size of Data Flash Secure region [KB] e.g.) 0004h -> 00h, 04h (4KB)
SRS1	(2 byte)	Size of SRAM Secure region (without NSC) [KB] e.g.) 0040h -> 00h, 40h (64KB)
SRS2	(2 byte)	Size of SRAM Secure region [KB] e.g.) 0040h -> 00h, 40h (64KB) * 8KB align
SUM	(1 byte)	Sum data
ETX	(1 byte)	03h

NSC: Non-secure callable regions
* If CFS does not comply with alignment, boot firmware rounds down them.

6.10.2.2 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

6.10.2.3 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

6.10.3 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 received command packet's LNH and LNL are different from the values specified in the packet format, the boot firmware sends a "Packet error".
- If the received command packet's LNH and LNL 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, boot firmware analyzes the command parameters.

- When CFS1 is bigger than CFS2, “Parameter error” is returned.
- When SRS1 is bigger than SRS2, “Parameter 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 writes the boundary setting.

- If an error occurs while writing, the boot firmware sends a “Flash access 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.

6.10.4 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
CFS1 is larger than CFS2.	Parameter error	FFFFFFFFh	FFFFFFFFh
SRS1 is larger than SRS2.	Parameter error	FFFFFFFFh	FFFFFFFFh
FACI detected an error after the command execution in an undisclosed area.	Flash access error	Flash status	FFFFFFFFh
Successful completion.	OK	FFFFFFFFh	FFFFFFFFh

6.10.5 Example of Use

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

Example: CFS=0200h, DFS=0004h, SRS1=001Fh, SRS2=0020h

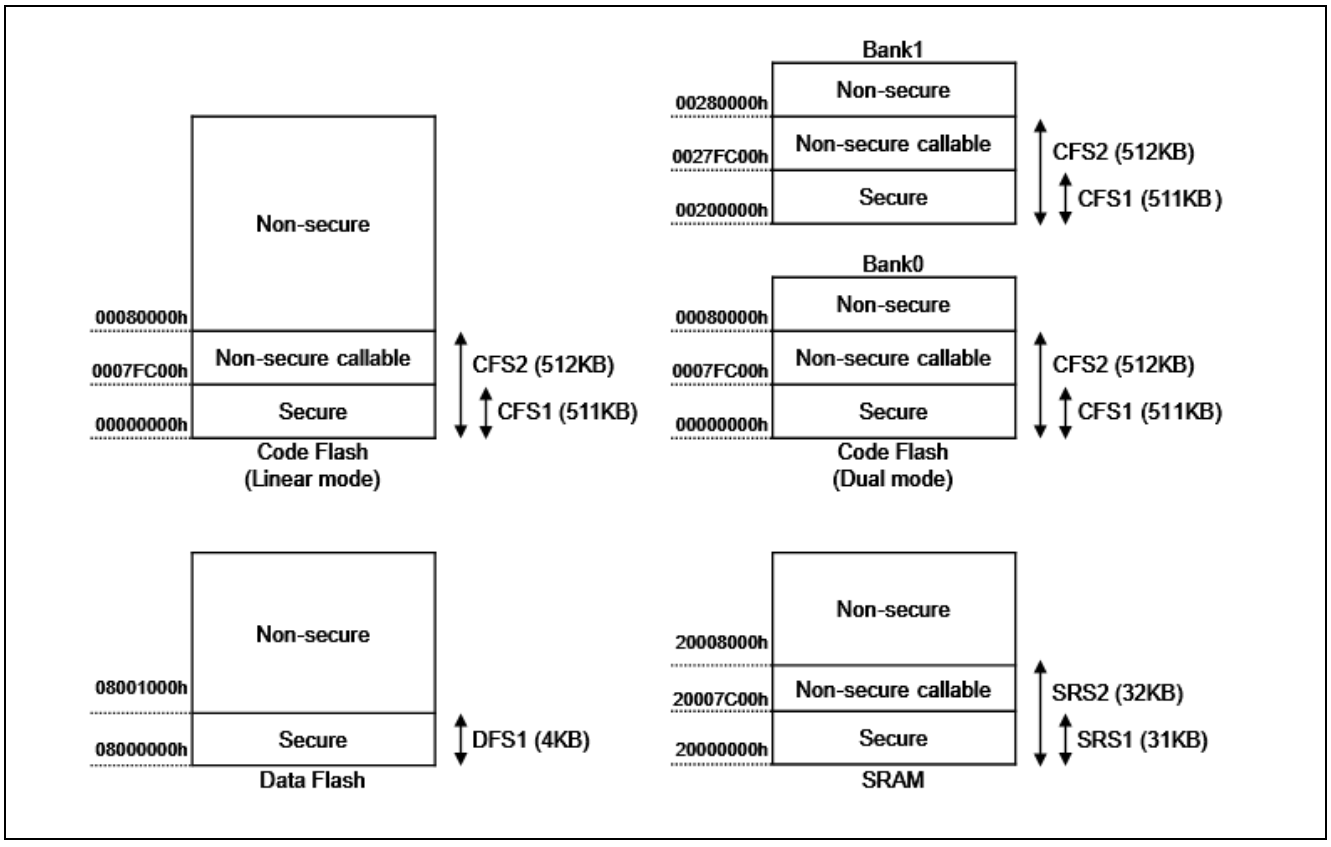


Figure 26. Boundary Setting Example

6.11 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.

6.11.1 Sequence Diagram

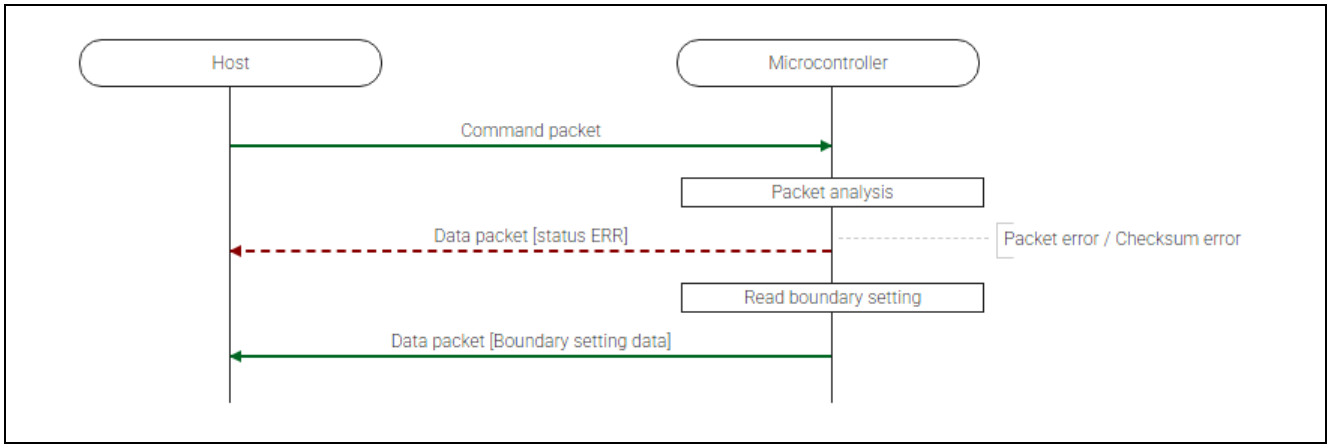


Figure 27. Boundary Request Command Sequence Diagram

6.11.2 Packets

6.11.2.1 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

6.11.2.2 Data packet [Boundary Setting Data]

SOD	(1 byte)	81h
LNH	(1 byte)	00h
LNL	(1 byte)	0Bh
RES	(1 byte)	4Fh (OK)
CFS1	(2 bytes)	Size of Code Flash Secure region (without NSC) [KB] For example: 0100h -> 01h, 00h (256 KB)
CFS2	(2 bytes)	Size of Data Flash Secure region [KB] For example: 00h -> 04h (4 KB)
DFS1	(2 bytes)	Size of Data Flash Secure region [KB] For example: 0004h -> 00h, 04h (4 KB)
SRS1	(2 bytes)	Size of SRAM Secure region (without NSC) [KB] For example: 0040h -> 00h, 40h (64 KB)
SRS2	(2 bytes)	Size of SRAM Secure region [KB] For example: 0040h -> 00h, 40h (64 KB)
SUM	(1 byte)	Sum data
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)	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

6.11.3 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 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 received command packet's LNH and LNL are different from the values specified in the packet format, the boot firmware sends a "Packet error".
- If the received command packet's LNH and LNL 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 returns boundary setting.

- Boot firmware sends "Boundary information" and waits for the next command.
  * 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 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

6.12 Parameter Setting Command

This command stores the received parameter in the device.

This command requires adherence to conditions described in Command List.

6.12.1 Sequence Diagram

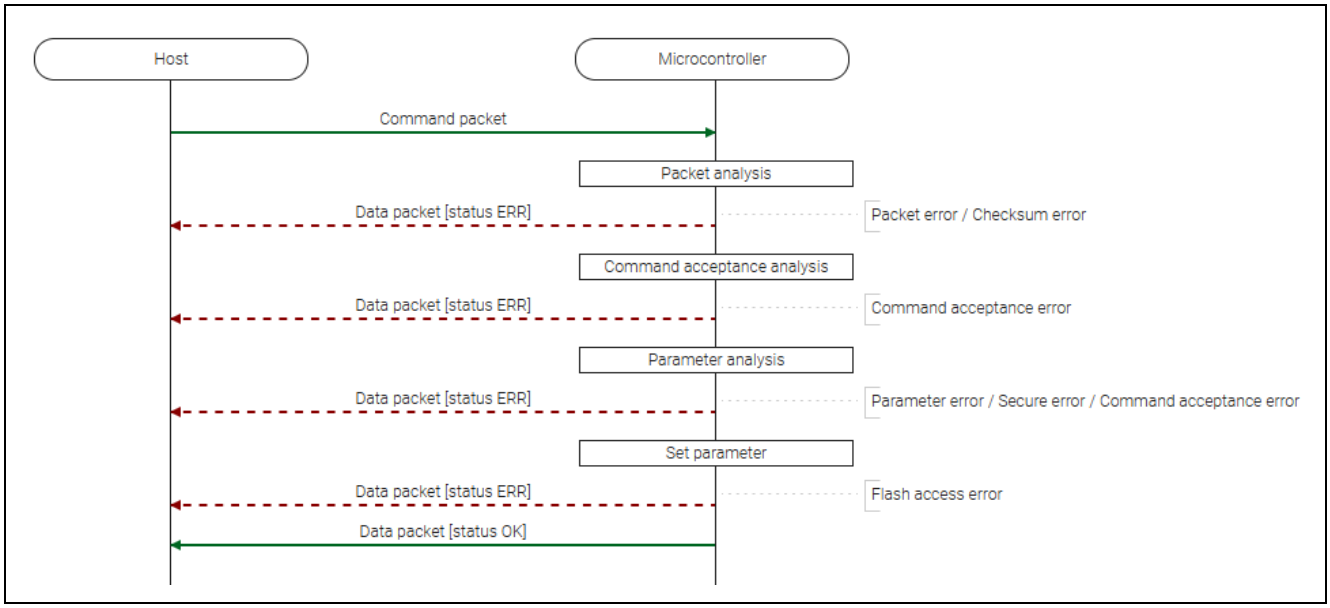


Figure 28. Parameter Setting Command Sequence Diagram

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)	51h (Parameter setting command)
PMID	(1 byte)	Parameter ID Specifiable parameter: **PMID** | **Parameter description** 01h | Disable initialization
PRMT	(1 byte)	Parameter data: • [PMID=01h] — 00h: Disable initialization
SUM	(1 byte)	Sum data
ETX	(1 byte)	03h

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

6.12.2.2 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

6.12.2.3 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

6.12.3 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 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 received command packet's LNH and LNL are different from the values specified in the packet format, the boot firmware sends a "Packet error".
- If the received command packet's LNH and LNL 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.
- If PRMT is not the specified value, the boot firmware sends a "Parameter error" and returns to the command wait state.
- When 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 writes parameter setting.

- If an error occurs while writing, the boot firmware sends a "Flash access 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.

6.12.4 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 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
The specified Parameter ID is an unsupported value.	Parameter error	FFFFFFFFh	FFFFFFFFh
Parameter data is not the specified value.	Parameter error	FFFFFFFFh	FFFFFFFFh
FACI detected an error after the command execution in not disclosed area.	Flash access error	Flash status	FFFFFFFFh
Successful completion.	OK	FFFFFFFFh	FFFFFFFFh

6.12.5 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.

6.13 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 require adherence to conditions described in Command List.

6.13.1 Sequence Diagram

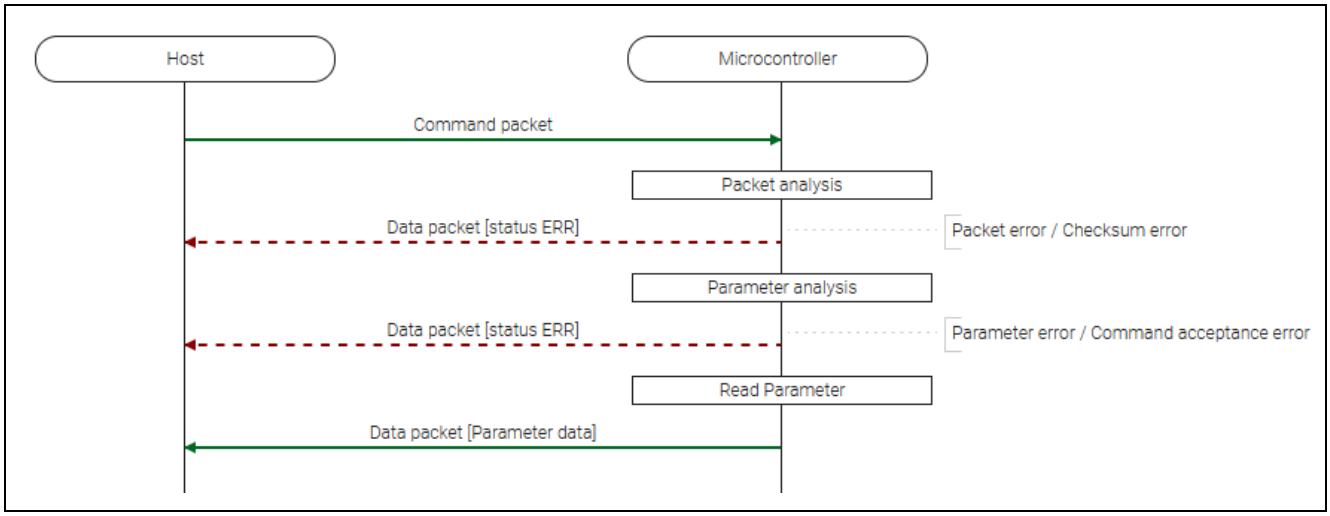


Figure 29. Parameter Request Command Sequence Diagram

6.13.2 Packets

6.13.2.1 Command Packet

SOH	(1 byte)	01h
LNH	(1 byte)	00h
LNL	(1 byte)	02h
CMD	(1 byte)	52h (Parameter request command)
PMID	(1 byte)	Parameter ID Specifiable parameter: **PMID** / **Parameter description** 01h / Disable initialization
SUM	(1 byte)	Sum data
ETX	(1 byte)	03h

6.13.2.2 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.
SUM	(1 byte)	Sum data
ETX	(1 byte)	03h

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

6.13.2.3 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

6.13.3 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 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 received command packet's LNH and LNL are different from the values specified in the packet format, the boot firmware sends a "Packet error".
- If the received command packet's LNH and LNL 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":
- When 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 returns the parameter setting:

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

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 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
The specified Parameter ID is an unsupported value.	Parameter error	FFFFFFFFh	FFFFFFFFh

6.13.5 Parameter Details

The following shows the parameter data (PRMT) details.

[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

6.14 Inquiry Command

This command is used to check if boot firmware is in the "Command acceptable phase" or not.

This command requires adherence to conditions described in Command List.

6.14.1 Sequence Diagram

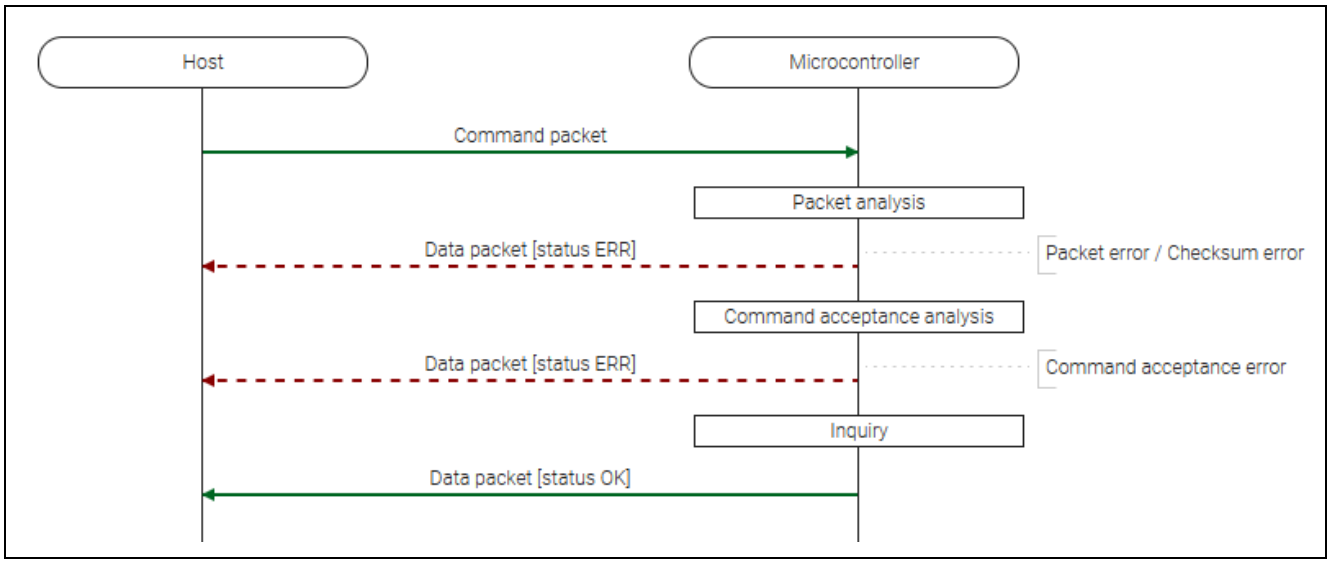


Figure 30. Inquiry Command Sequence Diagram

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)	00h (Inquiry command)
SUM	(1 byte)	FFh
ETX	(1 byte)	03h

6.14.2.2 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

6.14.2.3 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

6.14.3 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 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 received command packet's LNH and LNL are different from the values specified in the packet format, the boot firmware sends a "Packet error".
- If the received command packet's LNH and LNL 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, boot firmware executes the inquiry processing:

- The boot firmware sends "OK".
  * Memory status does 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
The process has ended normally.	OK	FFFFFFFFh	FFFFFFFFh

6.15 Signature Request Command

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

This command require adherence to conditions described in Command List.

6.15.1 Sequence Diagram

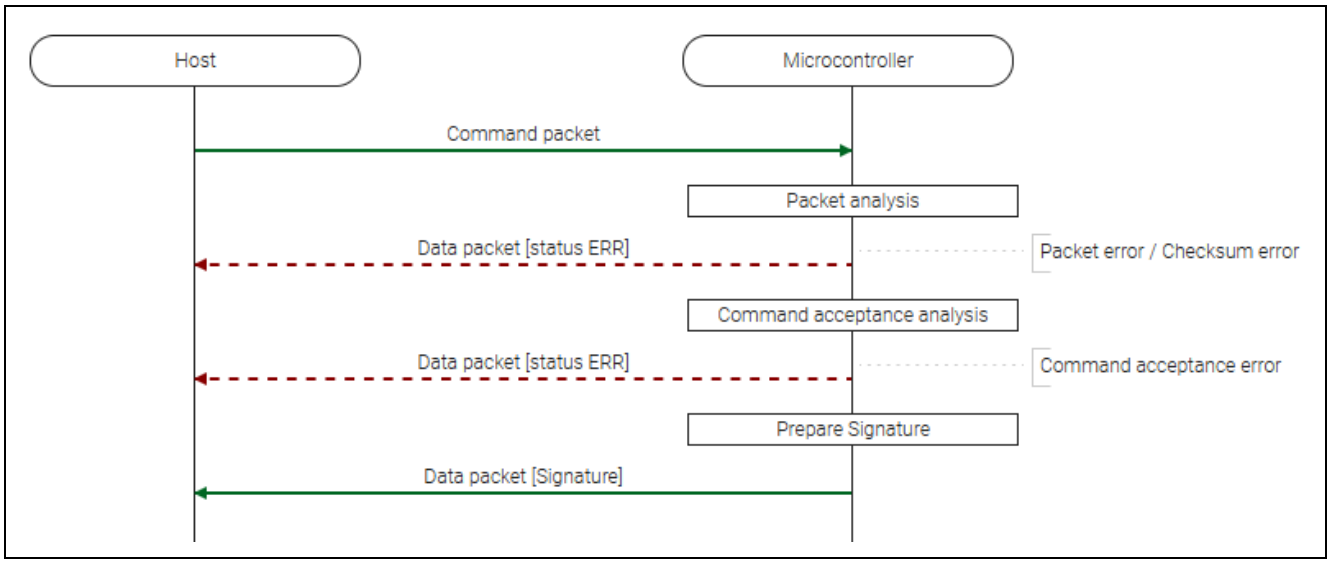


Figure 31. Signature Request Command Sequence Diagram

6.15.2 Packets

6.15.2.1 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

6.15.2.2 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): 08h: RA4L1,RA4C1 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 the 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

6.15.2.3 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

6.15.3 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 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 received command packet's LNH and LNL are different from the values specified in the packet format, the boot firmware sends a "Packet error".
- If the received command packet's LNH and LNL 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, boot firmware returns the signature.

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

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 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

6.16 Area Information Request Command

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

This command requires adherence to conditions described in Command List.

6.16.1 Sequence Diagram

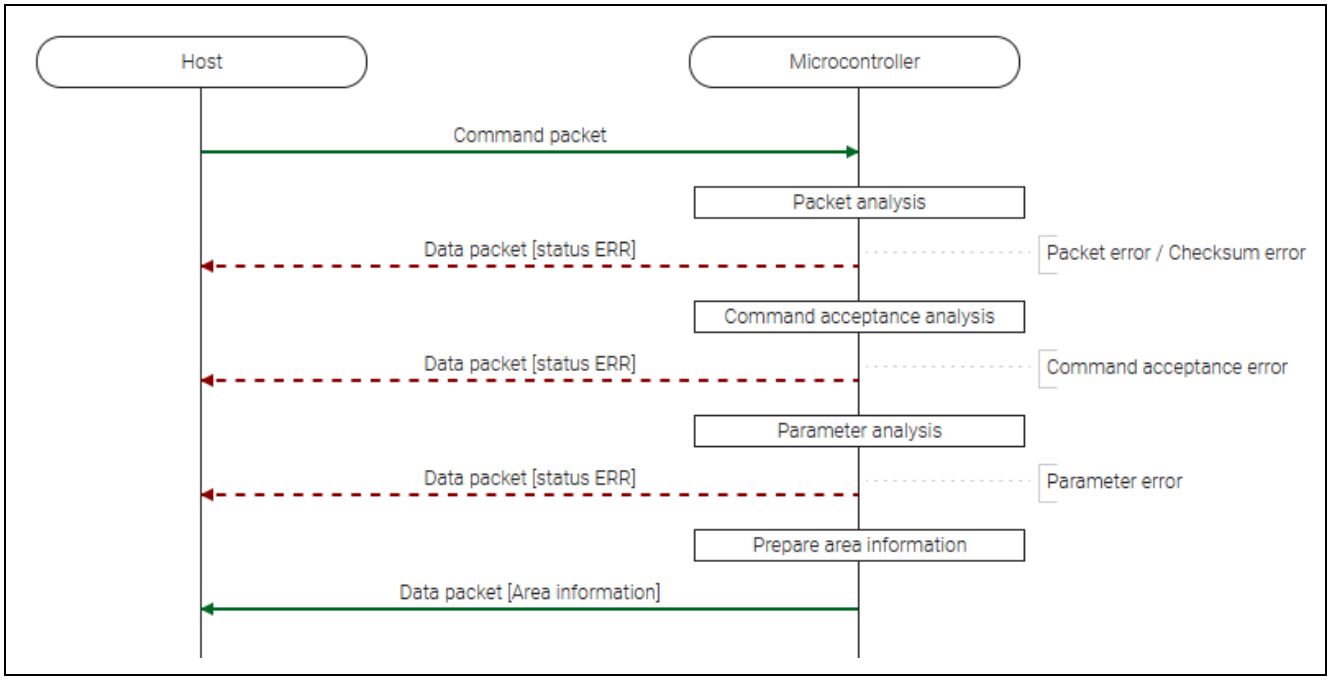


Figure 32. Area Information Request Command Sequence Diagram

6.16.2 Packets

6.16.2.1 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

6.16.2.2 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) • 1Nh: Data area N (*2) • 2Nh: Config 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: 32KB (32768byte) -> 00h, 00h, 80h, 00h Target command: Erase command.
WAU	(4 bytes)	Write access unit (alignment) [byte] (*1) Order of sending: High -> ... -> Low For example: 128byte -> 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: 1byte -> 00h, 00h, 00h, 01h Target command: Read command.
CAU	(4 bytes)	CRC access unit (alignment) [byte] (*1) Order of sending: High -> ... -> Low For example: 4byte -> 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

6.16.2.3 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

6.16.3 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 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 received command packet's LNH and LNL are different from the values specified in the packet format, the boot firmware sends a "Packet error".
- If the received command packet's LNH and LNL 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 will be returned:

- Send area information of specified NUM and return to command waiting status.
  * Memory status does 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 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

6.16.5 Example of Area Information

Example: RA4L1,RA4C1 (Linear mode)

NUM	Area	KOA	SAD	EAD	EAU	WAU	RAU	CAU
0	User area0	00h	00000000h	0007FFFFh(*3)	2KB	8B	1B	32KB
1	Data area	10h	08000000h	08001FFFH(*3)	256KB	1B	1B	1KB
2	Config area	20h	0100A100h	0100A2FFh	0 (*1)	16B	1B	256B(*2)

*1: When the Access unit is 0, it indicates that the corresponding operation is not supported.
*2: CRC command to Config area is executable only for the entire area, in other words, it is only possible to specify the area's SAD/EAD as CRC command's SAD/EAD. See CRC command for details.
*3: These addresses can change depending on the capacity of the flash of each product.

Example: RA4L1,RA4C1 (Dual mode)

NUM	Area	KOA	SAD	EAD	EAU	WAU	RAU	CAU
0	User area0	00h	00000000h	0003FFFFh(*3)	2KB	8B	1B	32KB
1	User area1	01h	00200000h	023FFFFh(*3)	2KB	8B	1B	32KB
2	Data area	10h	08000000h	08001FFFh(*3)	256KB	1B	1B	1KB
3	Config area	20h	0100A100h	0100A2FFh	0 (*1)	16B	1B	256B(*2)

*1: When the Access unit is 0, it indicates that the corresponding operation is not supported.
*2: CRC command to Config area is executable only for the entire area; in other words, it is only possible to specify the area's SAD/EAD as CRC command's SAD/EAD. See CRC command for details.
*3: These addresses can change depending on the capacity of the flash of each product

6.17 Baudrate Setting Command

This command receives baudrate data and changes 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 require adherence to conditions described in Command List.

6.17.1 Sequence Diagram

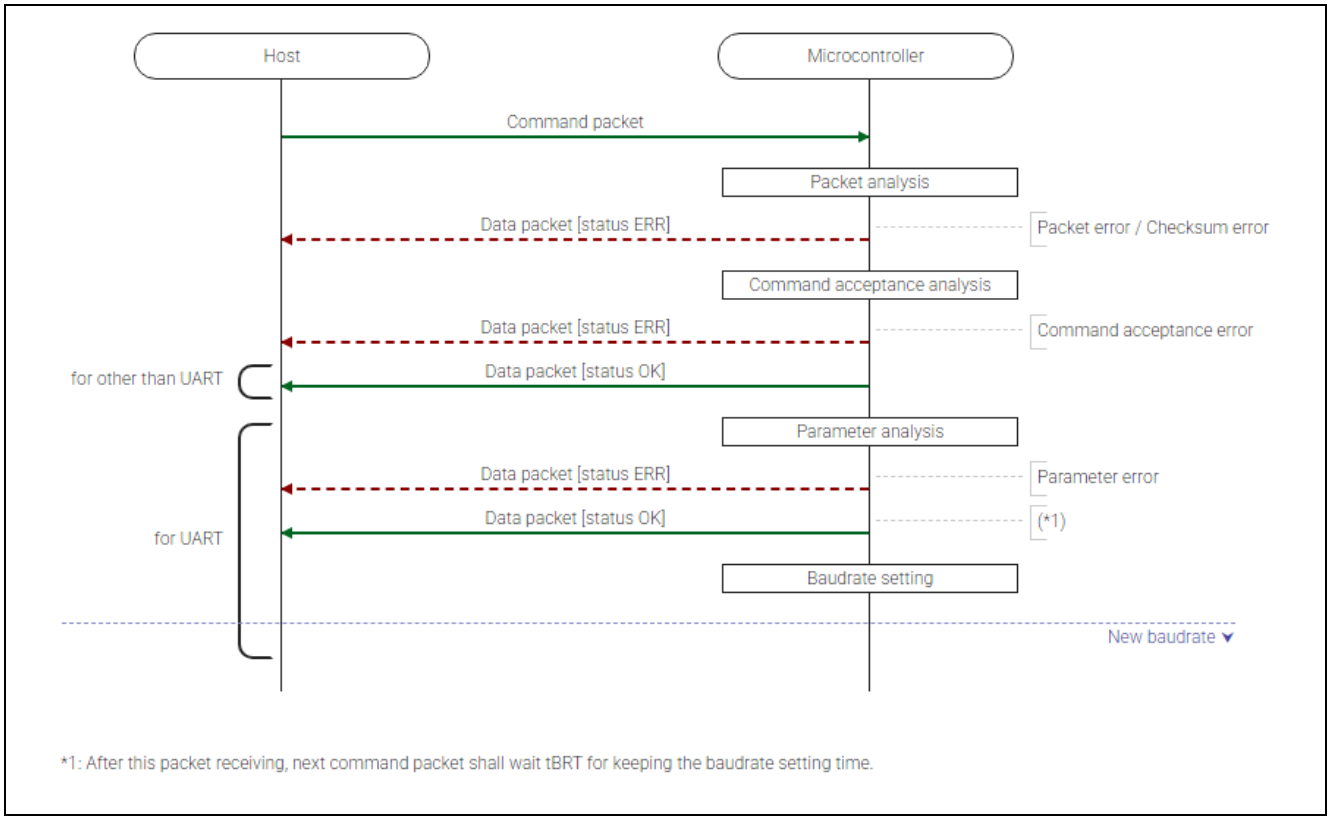


Figure 33. Baudrate Setting Command Sequence Diagram

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)	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
9600bps	00	00	25	80
115200bps	00	01	C2	00
500Kbps	00	07	A1	20
1.0Mbps	00	0F	42	40
1.5Mbps	00	16	E3	60
2.0Mbps	00	1E	84	80

6.17.2.2 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

6.17.2.3 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

6.17.3 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 received command packet's LNH and LNL are different from the values specified in the packet format, the boot firmware sends a "Packet error".
- If the received command packet's LNH and LNL 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 communication mode is not asynchronous 2-wire communication, a response will be 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 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
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 19. Baudrate Setting Values

When VCC is 1.8V or lower

Intended Baudrate	ABCS	CKS[1:0]	BRR[7:0]	MDDR[7:0]	Accuracy
9600bps	0	00b	07h	C9h	-0.2%
115200bps	0	00b	00h	ECh	±0.0%
Other	unavailable	unavailable	unavailable	unavailable	-

When VCC is higher than 1.8V

Intended Baudrate	ABCS	CKS[1:0]	BRR[7:0]	MDDR[7:0]	Accuracy
9600bps	0	00b	99h	FCh	-0.2%
115200bps	0	00b	0Ch	(Disabled)	+0.2%
500Kbps	0	00b	01h	ABh	+0.2%
1.0Mbps	0	00b	00h	ABh	+0.2%
1.5Mbps	0	00b	00h	(Disabled)	±0.0%
2.0Mbps	1	00b	00h	ABh	+0.2%
Other	unavailable	unavailable	unavailable	unavailable	-

6.18 Erase Command

This command erases data in the specified area of the flash memory. The alignment of the target addresses shall follow 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.

6.18.1 Sequence Diagram

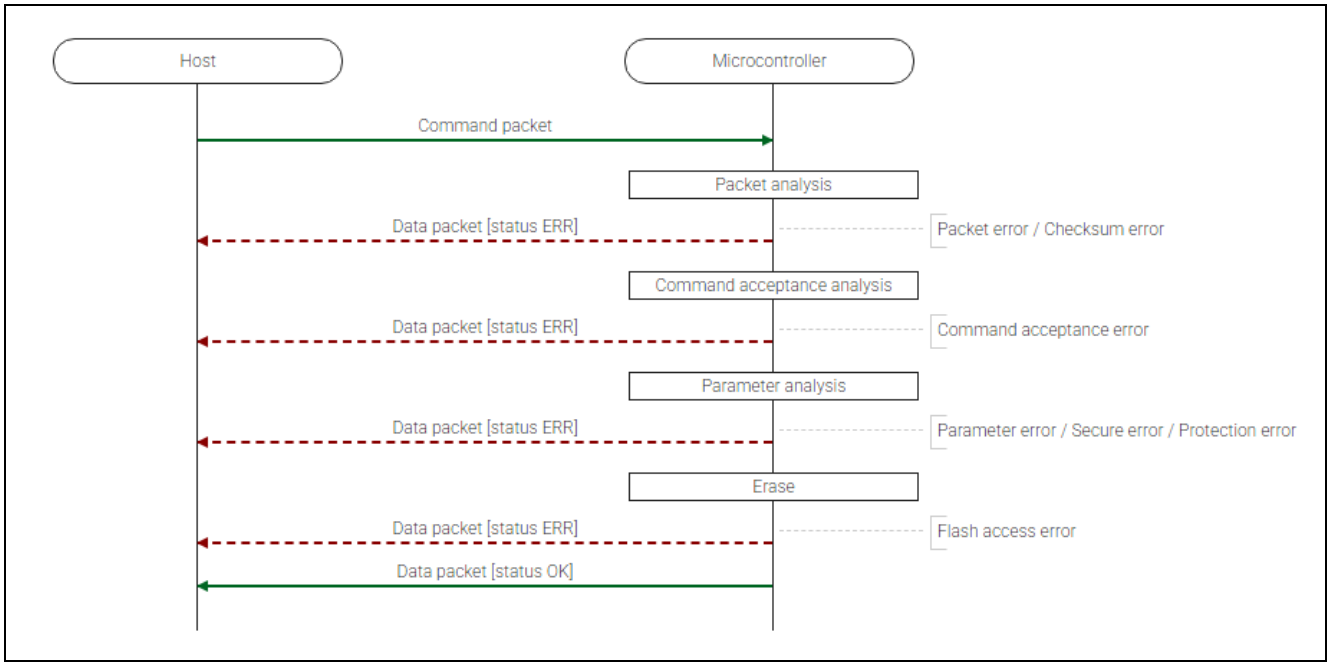


Figure 34. Erase Command Sequence Diagram

6.18.2 Packets

6.18.2.1 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

6.18.2.2 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

6.18.2.3 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

6.18.3 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 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 received command packet's LNH and LNL are different from the values specified in the packet format, the boot firmware sends a "Packet error".
- If the received command packet's LNH and LNL 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".
- 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 an address that is inaccessible with the current boundary setting, the boot firmware sends an "Invalid address error".
- When the designated erasure range includes a permanently protected block, "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.

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

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

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
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 Kinds 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
Current DLM state is NSECSD and designated erasure range includes Secure region.	Secure error	FFFFFFFFh	FFFFFFFFh
Designated erasing range includes permanent protected blocks.	Protection error	FFFFFFFFh	FFFFFFFFh
FACI detected an error after the command execution.	Flash access error	Flash status	Failure address
Successful completion.	OK	FFFFFFFFh	FFFFFFFFh

6.19 Write Command

This command receives data from the host and writes those data to the specified area. The alignment of the target address shall follow 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 the Command List.

6.19.1 Sequence Diagram

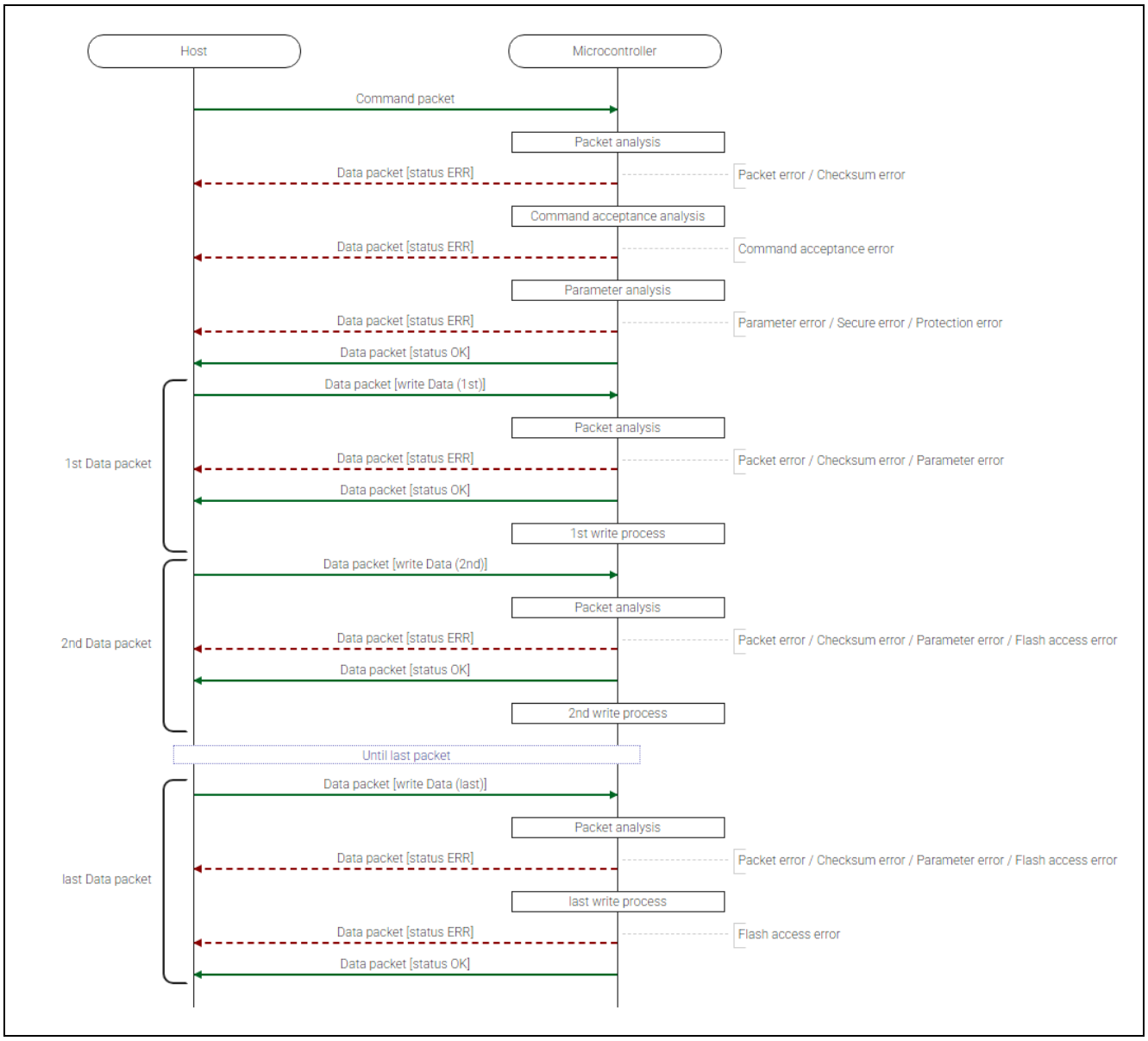


Figure 35. Write Command Sequence Diagram

6.19.2 Packets

6.19.2.1 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

6.19.2.2 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.

6.19.2.3 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

6.19.2.4 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

6.19.3 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 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 received command packet's LNH and LNL are different from the values specified in the packet format, the boot firmware sends a "Packet error".
- If the received command packet's LNH and LNL 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".
- 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 will send 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 an address that is inaccessible with the current boundary setting, the boot firmware sends an "Invalid address error".
- If the current DLM state is NSECSD and the specified range includes a secure area, the boot firmware sends a “Secure error”.
- When designated writing range includes PBPS block, "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 will wait 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 the total length of the received data of data packets exceeds the size of a specified area, a "Parameter error" is returned.
- If the 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 a "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 the status:

- Boot firmware executes the write processing.
- If an error occurs while writing, the boot firmware sends a "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.

6.19.4 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
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 Kinds 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
Current DLM state is NSECSD and designated writing range includes Secure region.	Secure error	FFFFFFFFh	FFFFFFFFh
Designated writing range includes permanent protected blocks.	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
FACI detected an error after the command execution.	Flash access error	Flash status	Failure address
Successful completion.	OK	FFFFFFFFh	FFFFFFFFh

6.19.5 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.

6.20 Read Command

This command reads data from a specified area and sends those data to host. The alignment of the target addresses shall 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.

6.20.1 Sequence Diagram

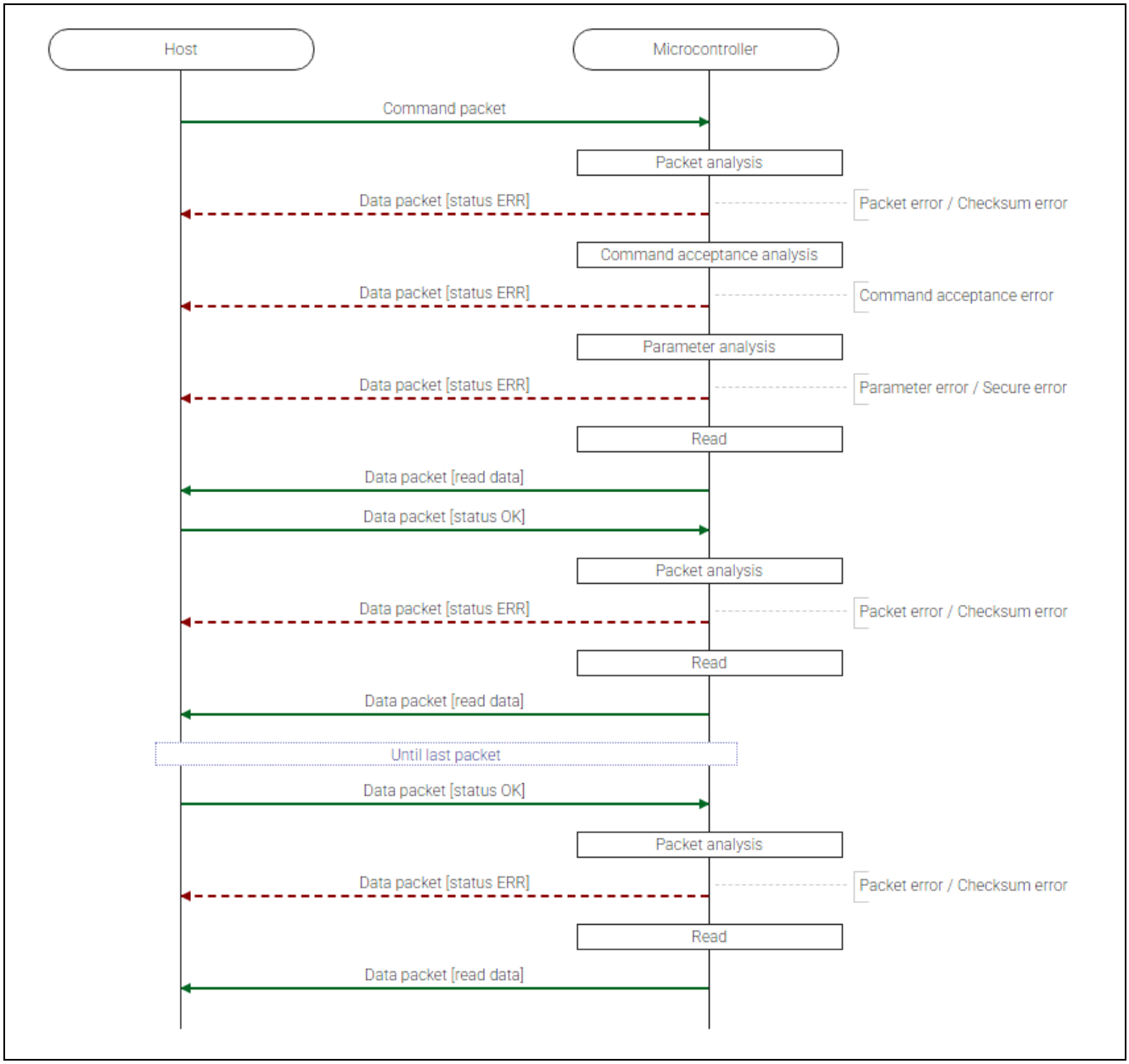


Figure 36. Read Command Sequence Diagram

6.20.2 Packets

6.20.2.1 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

6.20.2.2 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

6.20.2.3 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

6.20.2.4 Data Packet [Read Data]

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

N = 1–1024

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 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 received command packet's LNH and LNL are different from the values specified in the packet format, the boot firmware sends a "Packet error".
- If the received command packet's LNH and LNL 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".
- 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 will send 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 an "Invalid address 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 read processing:

- Boot firmware returns the data stored in the internal buffer (packet length: Max.1024bytes).
- When all the data have been sent, 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, a "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 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 errors do not occur, the boot firmware continues to read and send data.

6.20.3.1 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 Kinds 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	FFFFFFFFh	FFFFFFFFh
Current DLM state is NSECSD, designated reading range is User area or Data area and includes Secure region.	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

6.21 CRC Command

This command calculates CRC data from a specified area and sends it to host. The alignment of the target addresses shall follow 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 require adherence to conditions described in Command List.

Boot firmware use the following CRC method:

Name	CRC-32-IEEE-802.3
Default value	FFFFFFFFh
Shift direction	Left shift
Polynomial representations	(MSB first) 04C11DB7h

6.21.1 Sequence Diagram

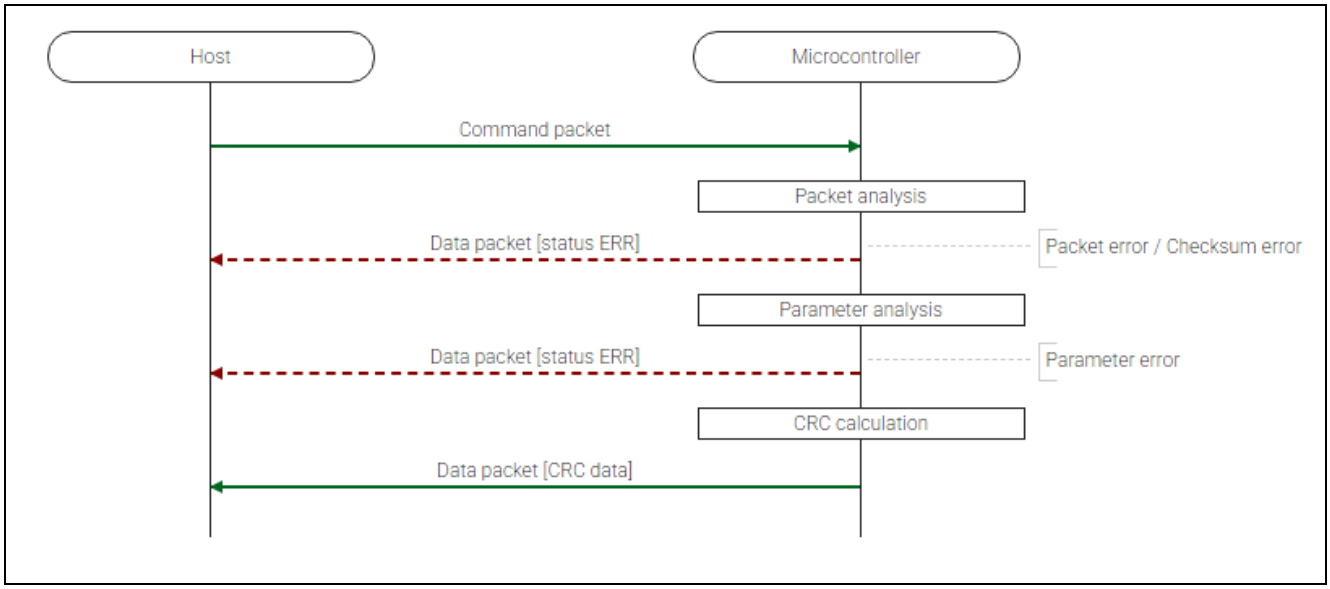


Figure 37. CRC Command Sequence Diagram

6.21.2 Packets

6.21.2.1 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

6.21.2.2 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). For example: 01234567h -> 01h, 23h, 45h, 67h
SUM	(1 byte)	Sum data
ETX	(1 byte)	03h

6.21.2.3 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

6.21.3 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 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 received command packet's LNH and LNL are different from the values specified in the packet format, the boot firmware sends a "Packet error".
- If the received command packet's LNH and LNL 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, 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 will send 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".
- When KOA is 20h (Config area) and SAD/EAD does not specify the entire Config area, a “Parameter 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.

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

- After the CRC calculation, boot firmware returns "CRC data" and waits for the next command.
  * 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
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 Kinds of area.	Parameter error	FFFFFFFFh	FFFFFFFFh
The access unit "CAU" of the specified area is 0.	Parameter error	FFFFFFFFh	FFFFFFFFh
Start address or End address does not comply with CAU of the area.	Parameter error	FFFFFFFFh	FFFFFFFFh
The specified area is Config area and Start address and End address do not specify entire Config area.	Parameter error	FFFFFFFFh	FFFFFFFFh

6.22 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 requires adherence to conditions described in Command List.

Only the following commands are executable after boot firmware sends the 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	–
Boundary request command	–
Parameter setting command	–
Parameter request command	–
CRC command	–

6.22.1 Sequence Diagram

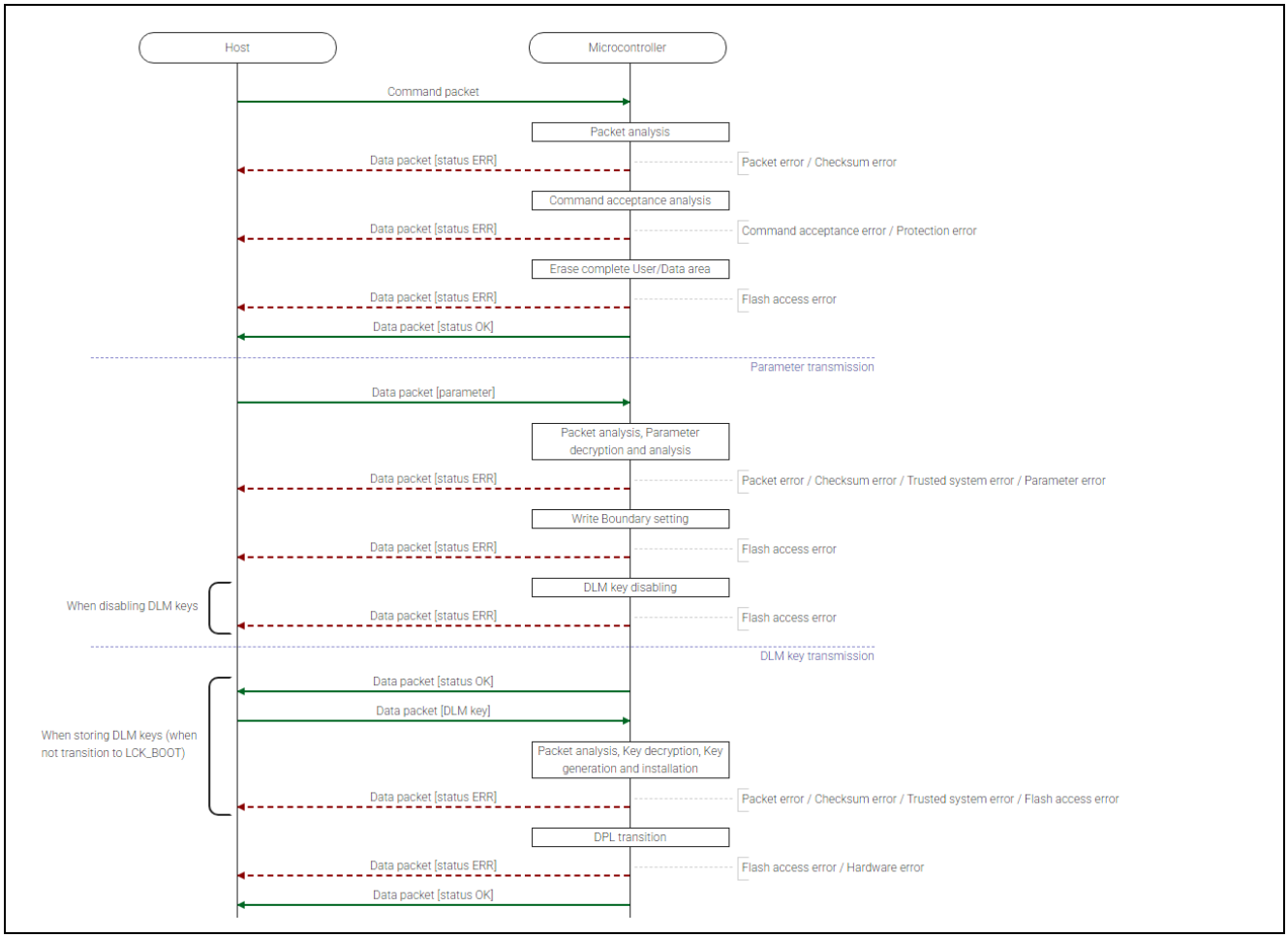


Figure 38. Encrypted Data Write Command Sequence Diagram (Part 1)

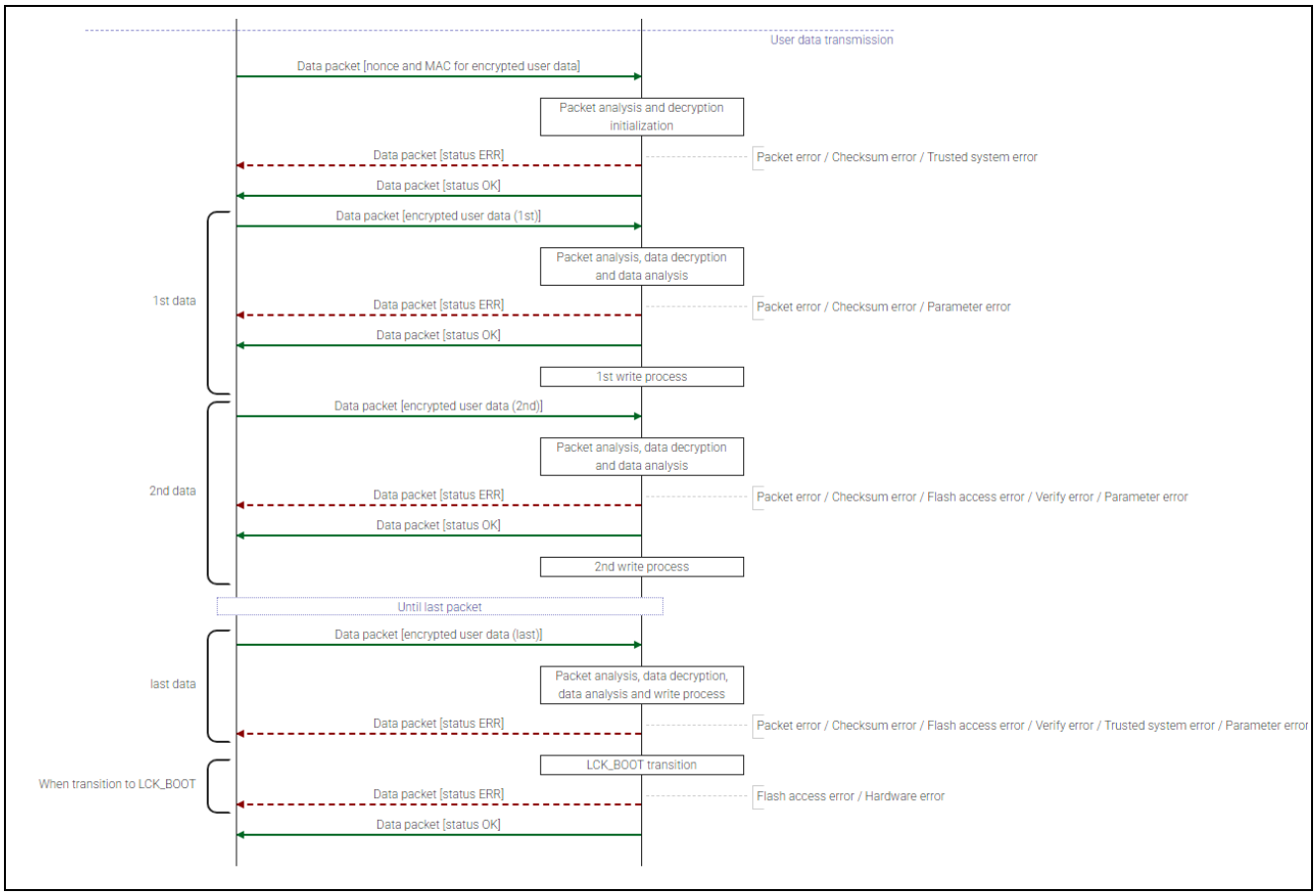


Figure 39. Encrypted Data Write Command Sequence Diagram (Part 2)

6.22.2 Packets

6.22.2.1 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

6.22.2.2 Data Packet [Parameter]

<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>3Dh</td></tr>
<tr><td>RES</td><td>(1 byte)</td><td>1Ah (OK)</td></tr>
<tr><td>NCE</td><td>(12byte)</td><td>Nonce is used for encrypting parameters.
Nonce length is 12 bytes and counter length is 4 bytes.</td></tr>
<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–16th bytes</td></tr>
<tr><td colspan="5">(reserved: FFh)</td></tr>
<tr><td colspan="5">17th-24th bytes</td></tr>
<tr><td colspan="5">Boundary Setting Parameter (CFS1, CFS2, DFS1, SRS1)</td></tr>
<tr><td colspan="3">25th-26th bytes</td><td colspan="2">27th-32th bytes</td></tr>
<tr><td colspan="3">Boundary Setting Parameter (SRS2)</td><td colspan="2">(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
- 16bytes * 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: 00000000h ~ 0007FFFFh (524,288 byte)
Area2: 08000000h ~ 080001FFh (512 byte)
Area3: 08000300h ~ 0800037Fh (128 byte)

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

*1) 512 packets are necessary to send 524,288-bytewrite data since up to 1024-byte data can be sent</td></tr>
</table>
</td></tr>
</table>

<table>
<tr><td></td><td></td><td><table>
<tr><td></td><td>by one *Data packet [encrypted user data]”.

*2) One “Data packet [encrypted user data]” is necessary each for write data of these two area; cannot send write data for these two area by single “Data packet [encrypted user data]”. This is because even when the total size of the write data is 1024 bytes 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: DPL
02h: LCK_BOOT</td></tr>
<tr><td>KNUM2
(1 byte)</td><td>Number of SECDBG_key to store, specifiable 1 only.
This parameter is ignored when transmitting to LCK_BOOT (TRN=02h)</td></tr>
<tr><td>KNUM1
(1 byte)</td><td>Number of NONSECDBG_key to store, specifiable 1 only.
This parameter is ignored when transitting to LCK_BOOT (TRN=02h)</td></tr>
<tr><td>CFS1
(2 byte)</td><td rowspan="5">Boundary setting parameters.
See the Boundary Setting command for details of each parameter.</td></tr>
<tr><td>CFS2
(2 byte)</td></tr>
<tr><td>DFS1
(2 byte)</td></tr>
<tr><td>SRS1
(2 byte)</td></tr>
<tr><td>SRS2
(2 byte)</td></tr>
<tr><td>(reserved)</td><td>Reserved data is ignored. Although any value is specifiable, using random numbers or similarly complex values is recommended for security reasons, rather than using simple values 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>

6.22.2.3 Data Packet [DLM Key]

<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>7Dh</td></tr>
<tr><td>RES</td><td>(1 byte)</td><td>1Ah (OK)</td></tr>
<tr><td>NCE</td><td>(12 byte)</td><td>Nonce used for encrypting DLM key data
Nonce length is 12 byte and the counter length is 4 byte</td></tr>
<tr><td>KEY</td><td>(96 byte)</td><td>Encrypted DLM key data
Encryption method is AES128-CCM mode (NIST SP800-38C)

DLM key data before encryption is shown in the two tables below.

DLM key data before encryption:

<table>
<tr><td rowspan="2">SECDBG 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">NONSECDBG key data</td><td>49th ~ 64th bytes</td><td>65th ~ 96th bytes</td></tr>
<tr><td>IVEC</td><td>ENKY</td></tr>
</table>

Where IVEC and ENKY mean:
<table>
<tr><td>IVEC</td><td>Initialization Vector</td></tr>
<tr><td>ENKY</td><td>Encrypted DLM key (0-15byte) + MAC (16-31 byte)</td></tr>
</table>
</td></tr>
<tr><td>MAC</td><td>(16 byte)</td><td>MAC for Encrypted DLM 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>

6.22.2.4 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

6.22.2.5 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><td>1st–4th bytes</td><td>5th–6th bytes</td><td>7th–16th bytes</td></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 followings:
— 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

6.22.2.6 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

6.22.2.7 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

6.22.3 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 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 received command packet's LNH and LNL are different from the values specified in the packet format, the boot firmware sends a "Packet error".
- If the received command packet's LNH and LNL 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 device reset is not asserted after the Encrypted data write command execution, the boot firmware sends a "Command acceptance error".
- If any of the following conditions is met, boot firmware sends a "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 a Permanently 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 erasure fails, the boot firmware sends a "Flash access error" and returns to the 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 the 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 completes, boot firmware checks PRM:
  - Boot firmware sends "Parameter error" if PRM is unspecified value.
- 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 DLM keys depending on the parameters:

- Boot firmware disables DLM keys depending on TRN, as shown in the following table:

TRN	KNUM2	KNUM1	SECDBG_KEY	NONSECDBG_KEY
DPL	1	1	-	-
LCK_BOOT	ignore	ignore	X	X

X: Disable

-: Not disable

- If an error occurs while disabling DLM keys, boot firmware sends a "Flash access error" and returns to the command wait state.

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

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

TRN	KNUM2	KNUM1	SECDBG_KEY	NONSECDBG_KEY
DPL	1	1	X	X
LCK_BOOT	ignore	ignore	-	-

X: Receive and write
-: Not receive nor write

- Boot firmware sends "OK".
- Boot firmware receives data packet [DLM key].
  * Refer to "Data Packet Reception" below for data packet reception processing.
- Boot firmware decrypts the received DLM key.
  If decryption fails, boot firmware sends a "Trusted system error".
  However if the Trusted system becomes abnormal during decryption, boot firmware does not send any packet and also will not respond to the data sent from the host after that.
- Boot firmware generates Key index (Wrapped DLM key).
  If the generation of the Key index (Wrapped DLM 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 DLM key) to the device.
  If an error occurs while writing the Key index (Wrapped DLM key), the boot firmware sends a "Flash access error" and returns to the command wait state.

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

- If an error occurs during the DPL transition, boot firmware returns a "Flash access 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, boot firmware sends "OK".

When the processing above is successfully completed, boot firmware receives the 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 the 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 Kinds of area.
    - SAD specifies the area whose WAU=0.
    - SAD is not a multiple of WAU.
    - SAD is not multiple of encryption block size.
    - SIZE is not multiple of WAU.

  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 a data packet, sends a "Flash access error", and returns to the command wait state.
- If the write value and write result do not match at writing to the Config area or EEP config area, boot firmware sends a "Verify error" and returns to the 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 "Flash access error" and returns to the command wait state.
- If the write value and write result do not match at writing to the Config area or EEP config area, boot firmware sends a "Verify error" and returns to the command waiting state.
- When the write processing is successfully completed, and TRN is "LCK_BOOT," boot firmware executes the LCK_BOOT transition.
- If an error occurs during the LCK_BOOT transition, boot firmware returns a "Flash access 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.

6.22.3.1 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 will wait 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 received the 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.
- If the received data packet's LNH and LNL are different from the values specified in each command, the boot firmware sends a "Packet error".
- If 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.

6.22.4 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 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
Any of the following conditions is met: • 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).	Protection error	FFFFFFFFh	FFFFFFFFh
There are many blocks protected by permanent block protection (PBPS).	Protection error	FFFFFFFFh	FFFFFFFFh
FACI detected an error after the command execution in the disclosed area.	Flash access error	Flash 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
DLM state is abnormal	Hardware error	FFFFFFFFh	FFFFFFFFh

Condition	STS	ST2	ADR
Any of the following conditions 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 Kinds of 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 bytes for AES128	Parameter error	FFFFFFFFh	FFFFFFFFh
The written value and the write result do not match at writing at Config area.	Verify error	FFFFFFFFh	FFFFFFFFh
Successful completion.	OK	FFFFFFFFh	FFFFFFFFh

6.22.5 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.

(3) If permanent block protection in the Config 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.

(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. If set, User key setting command become inexecutable due to Protection error.

(5) 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 note that 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 1 packet sends 944-bybte write data. Accordingly, when the write data size is 16,519,088-byte, LOD = (1024*16131) + 944 + (16*16132) = 16,777,200 = 00FFFFF0h = Maximum.

6.22.6 Device State after Command Execution

Table 20 shows the state of the device after this command is executed.

Table 20. 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/Data area</th><th>Boundary setting</th><th>DLM key</th><th>DLM</th><th>Area specified by SAD</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/Data area</td><td>Undefined</td><td colspan="4" rowspan="2">No change</td></tr>
<tr><td>Decrypting or analyzing the parameter</td><td rowspan="8">Erased</td></tr>
<tr><td>Setting boundary</td><td>Undefined</td><td colspan="3">No change</td></tr>
<tr><td>Disabling DLM key</td><td rowspan="9">Changed</td><td>Undefined</td><td colspan="2" rowspan="2">No change</td></tr>
<tr><td>Decrypting, generating or writing DLM key</td><td>Undefined or disabled depending on TRN</td></tr>
<tr><td>Transiting to DPL</td><td rowspan="3">Written or disabled depending on TRN</td><td>Undefined</td><td rowspan="2">No change</td></tr>
<tr><td>Initializing decryption</td><td rowspan="3">DPL</td></tr>
<tr><td>Decrypting, checking or writing user data</td><td>Erased (Areas specified by SAD are undefined.)</td><td>Undefined</td></tr>
<tr><td>Transiting to LCK_BOOT</td><td>User data written</td><td>DPL or Undefined depending on TRN</td><td rowspan="2">User data written</td></tr>
<tr><td colspan="2">Successful completion</td><td>DPL or LCK_BOOT depending on TRN</td></tr>
</table>

6.22.7 DLM State Transitions

Figure 40 shows the DLM states that can be transited by this command.

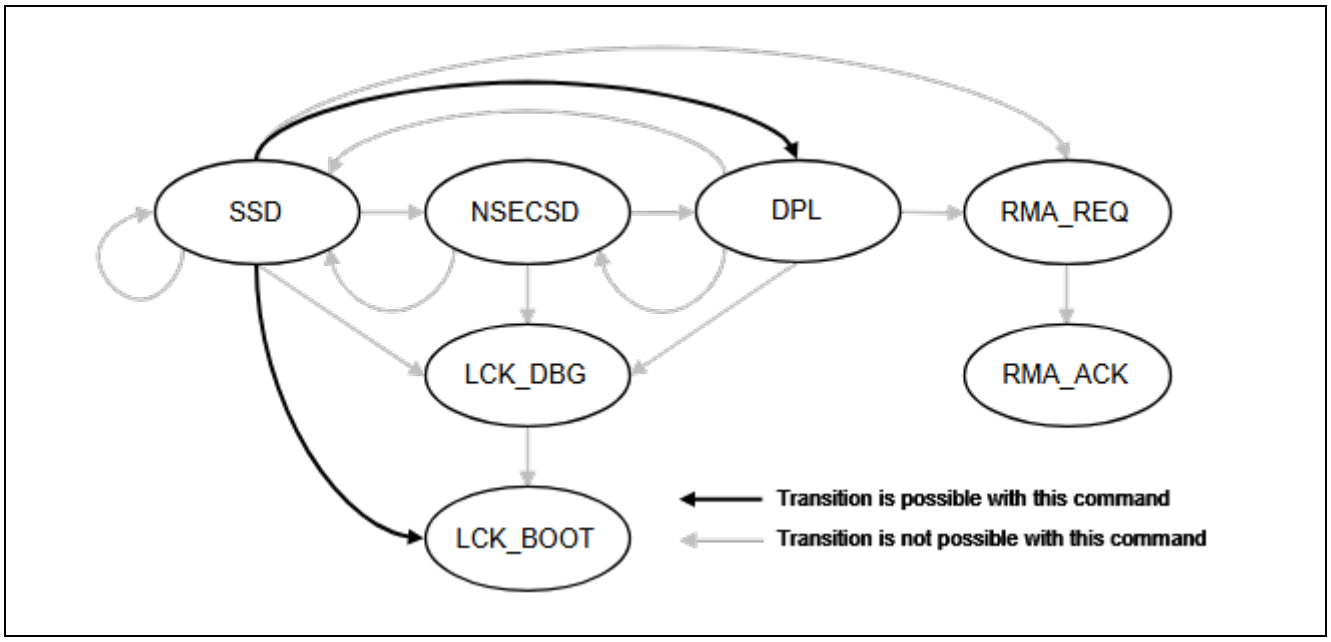


Figure 40. Valid DLM State Transitions for Encrypted Data Write Command

7. Flow Examples

7.1 Beginning Communication

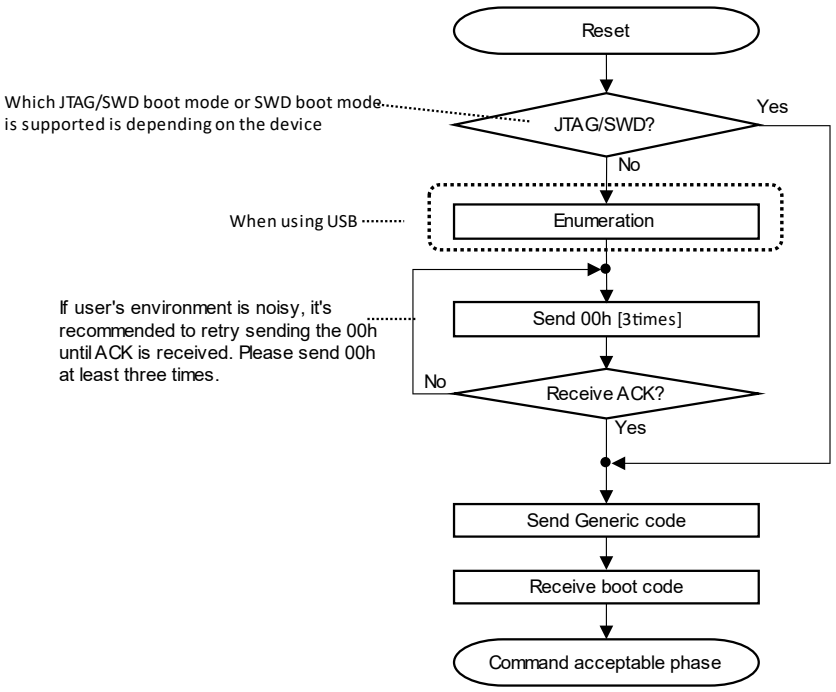


Figure 41. Beginning Communication

7.2 Acquisition of Device Information / Baudrate Settings

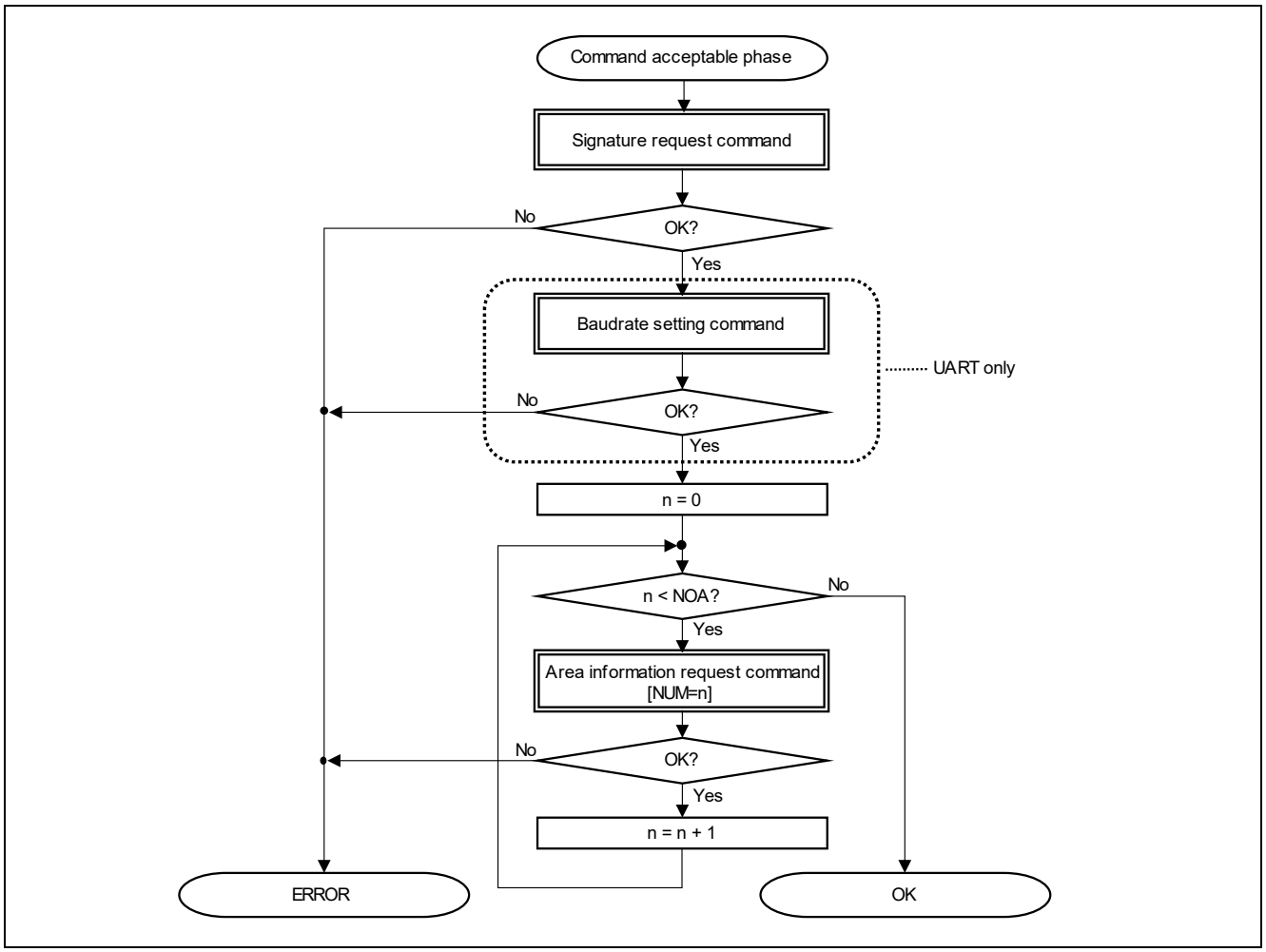


Figure 42. Acquisition of Device Information / Baudrate Settings

7.3 Transiting DLM State

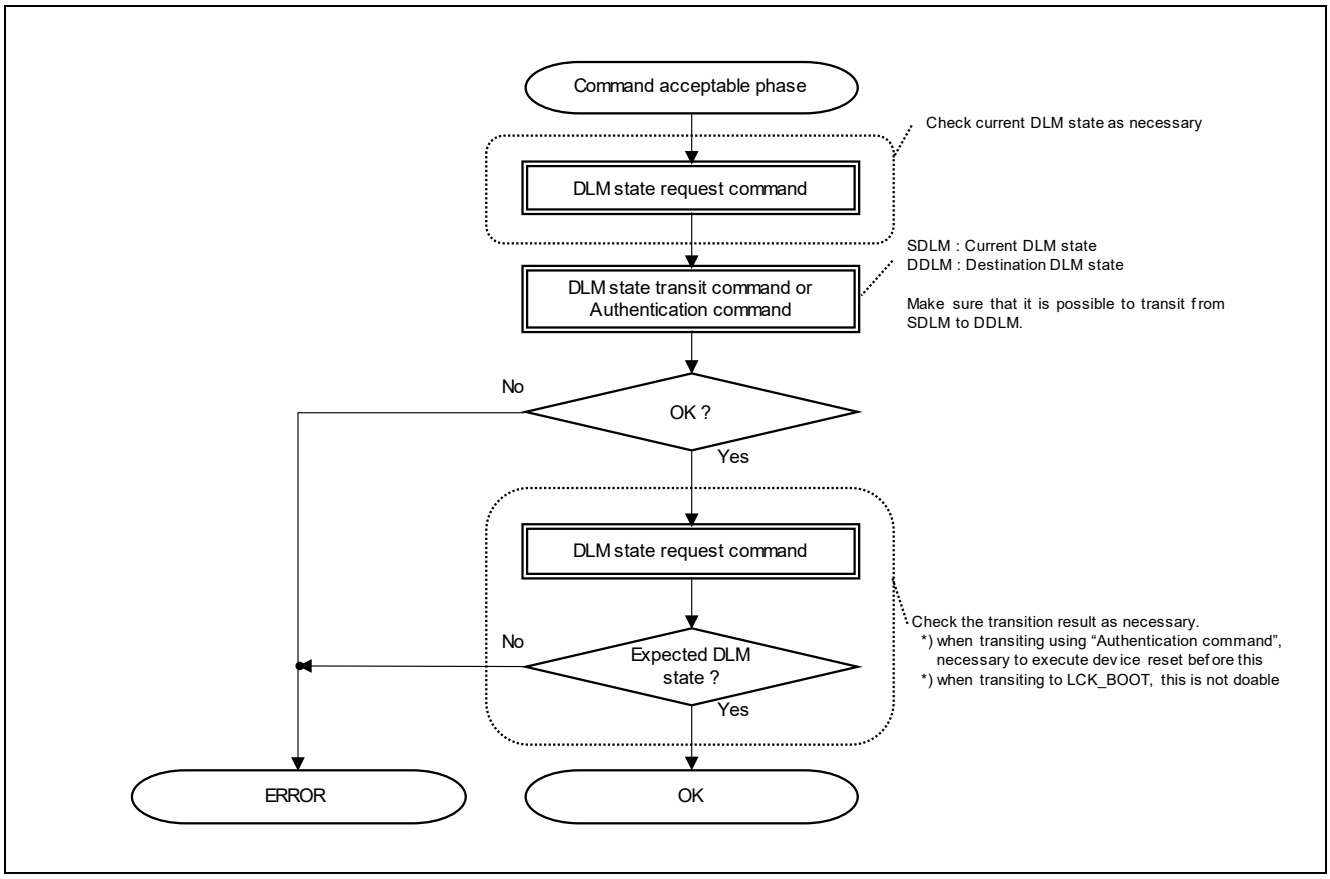


Figure 43. Transiting DLM State

7.4 Data Programming

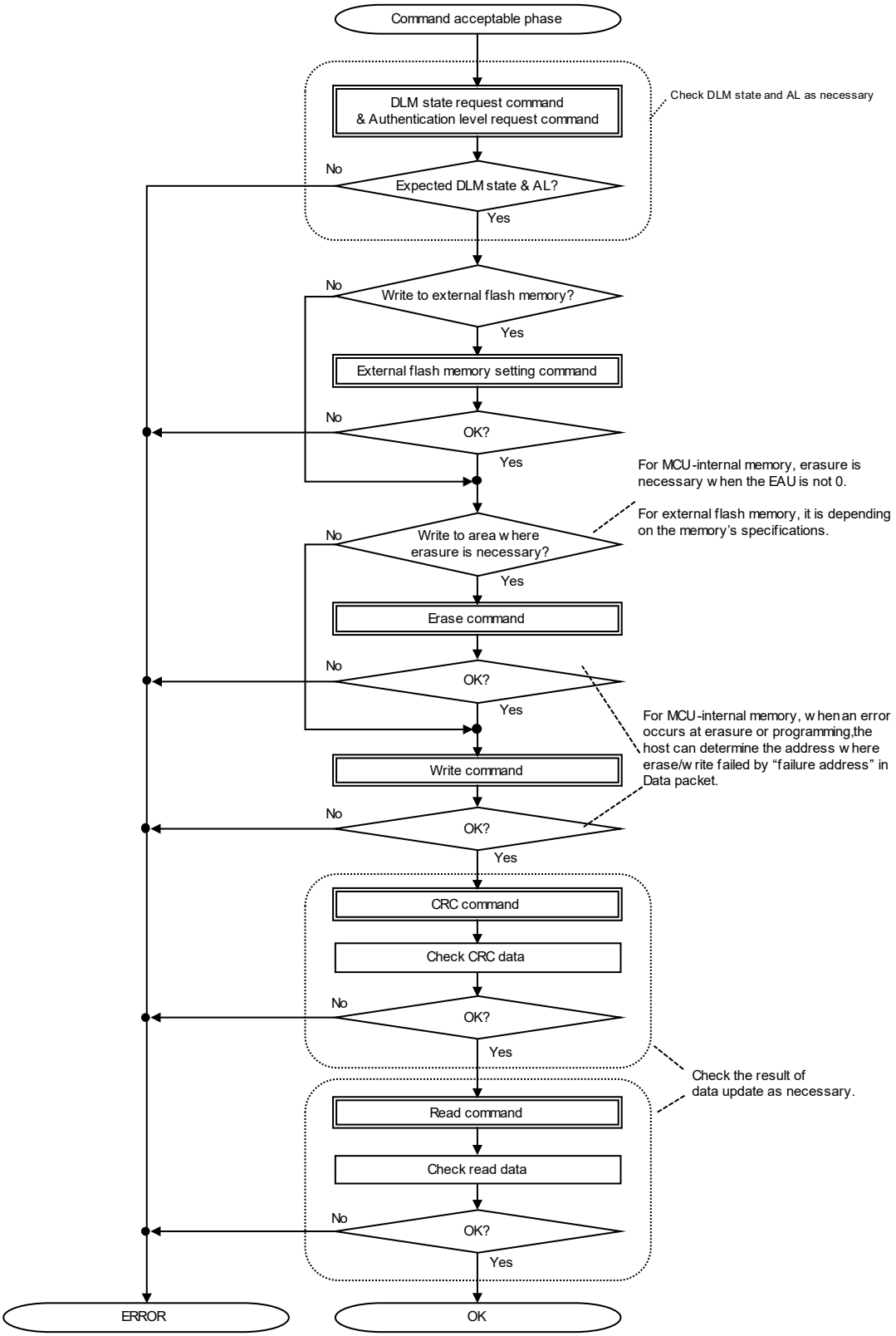


Figure 44. Data Programming

7.5 Encrypted Data Programming

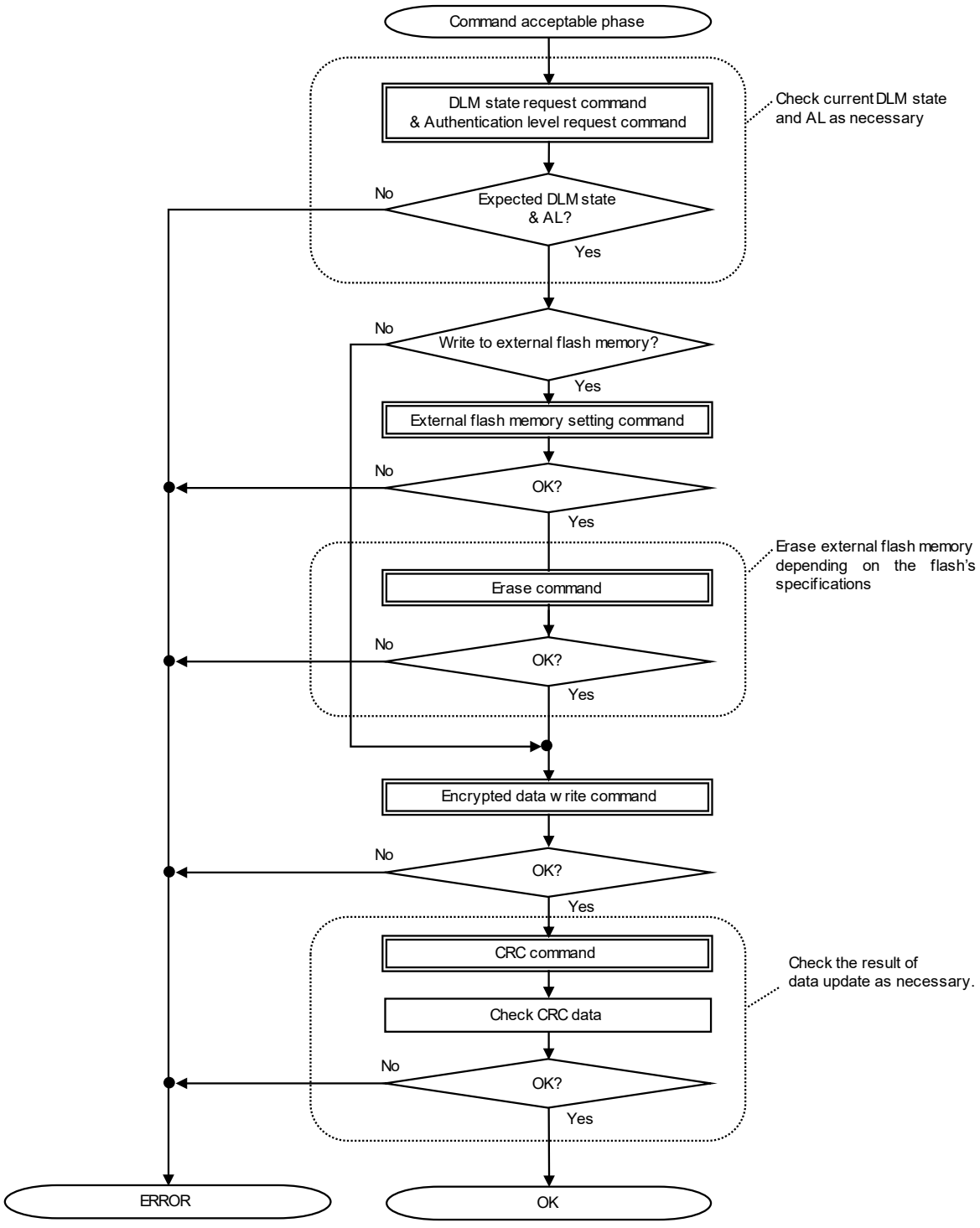


Figure 45. Encrypted Data Programming

7.6 Initializing Memory

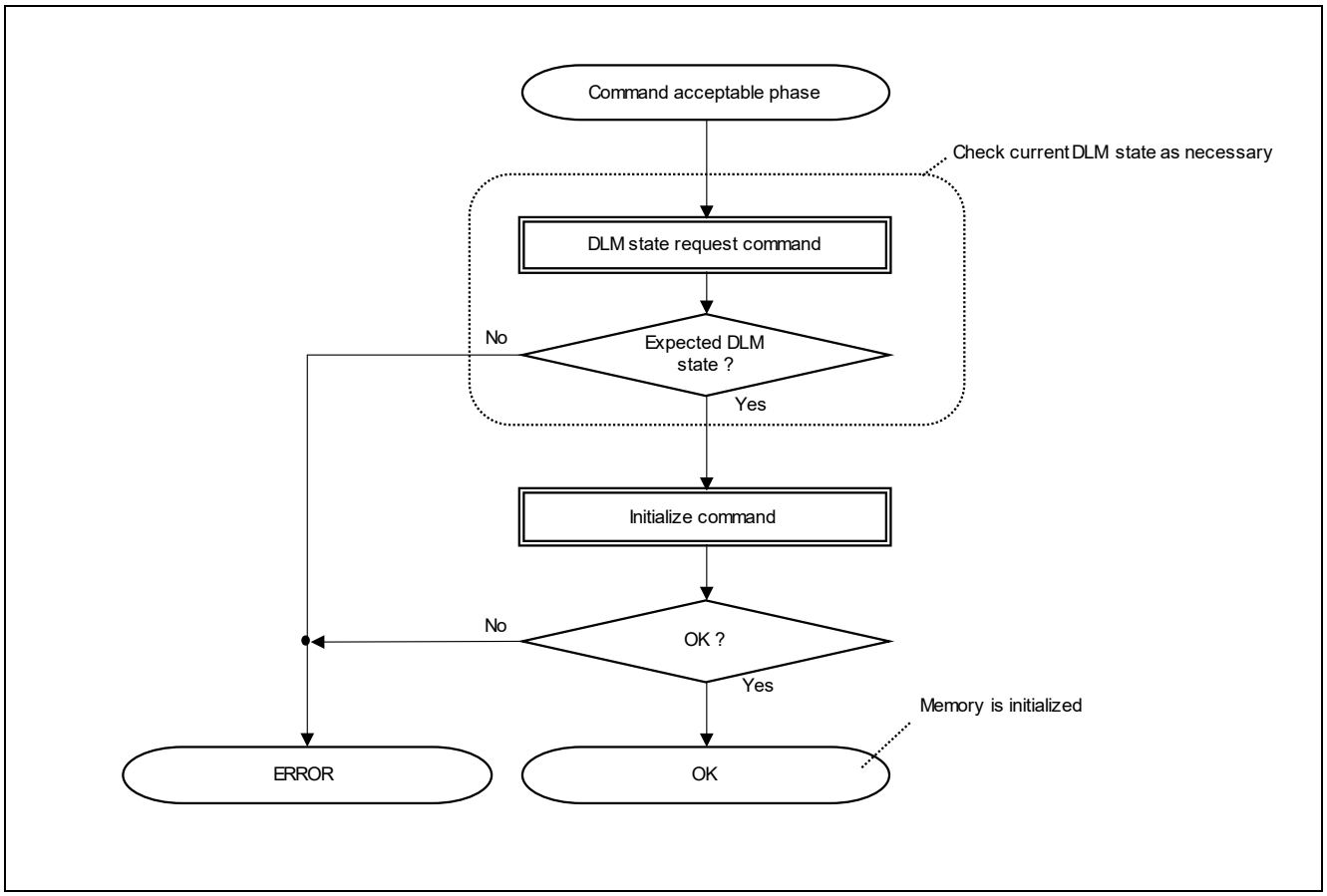


Figure 46. Initializing Memory

7.7 Storing Keys

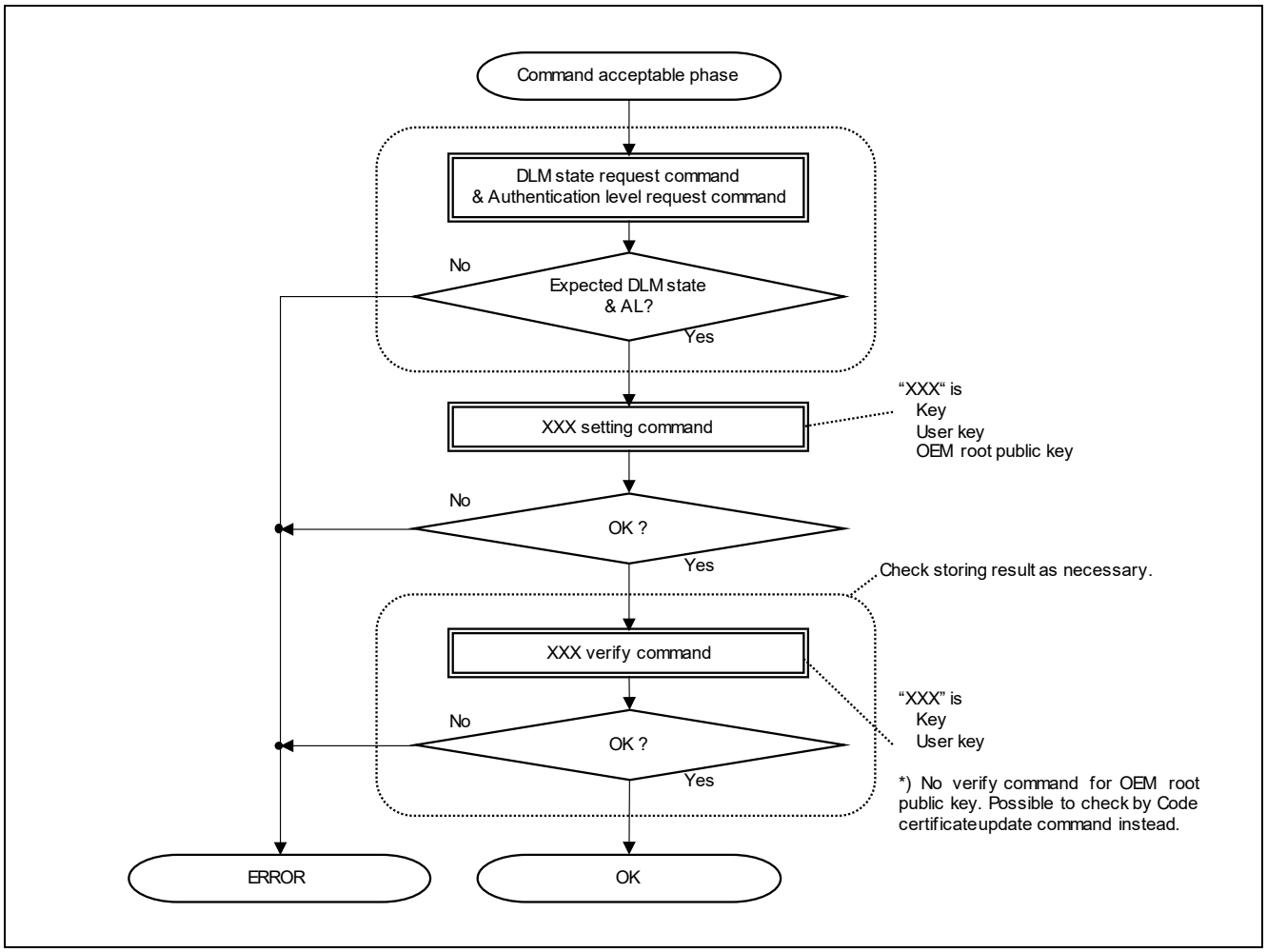


Figure 47. Storing Keys

7.8 Updating Boundary, Parameter, Lock Bit, or ARC Configuration Setting

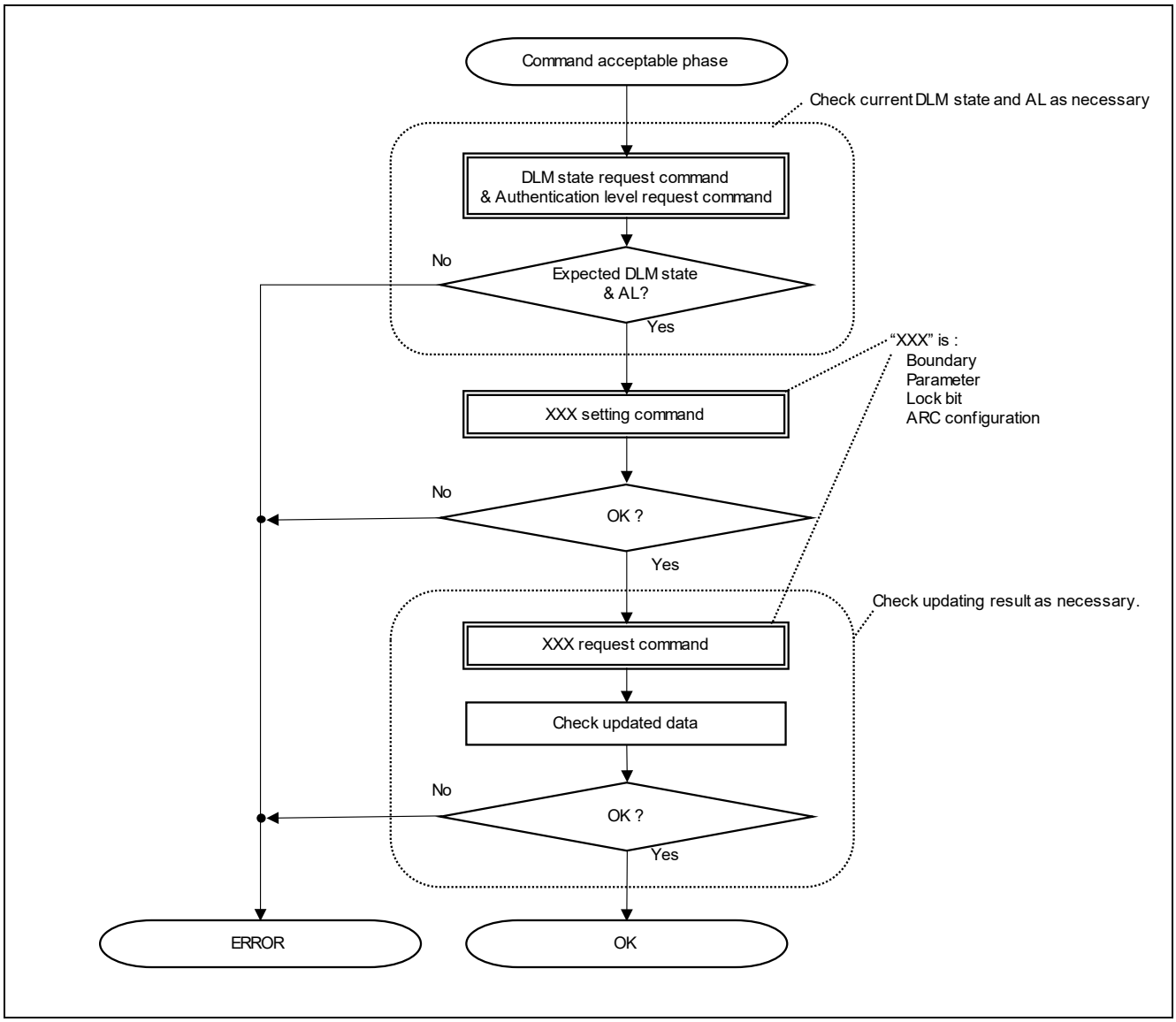


Figure 48. Updating Boundary, Parameter, Lock Bit, or ARC Configuration Setting

7.9 Downloading Whole Image

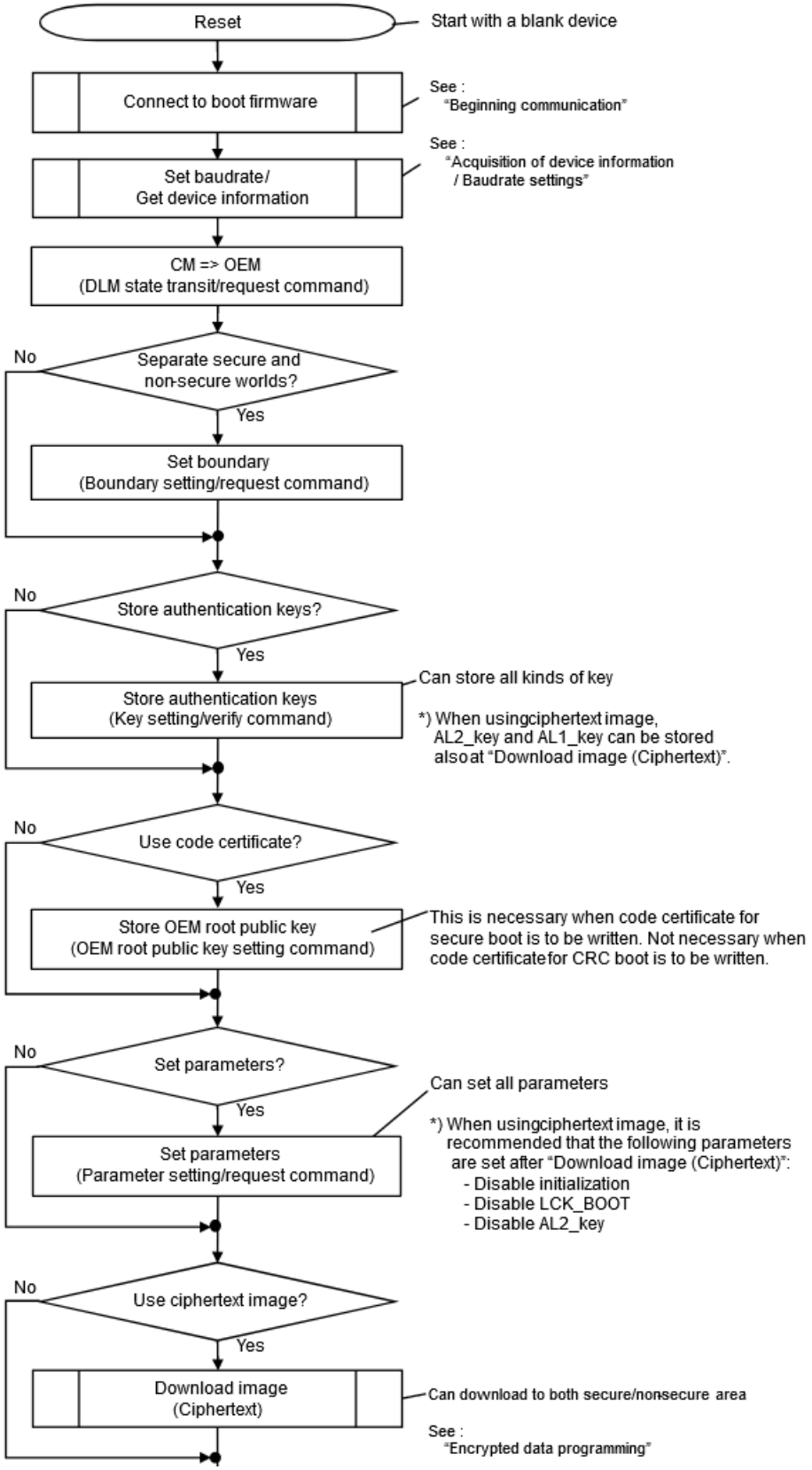


Figure 49. Downloading Whole Image (Part 1)

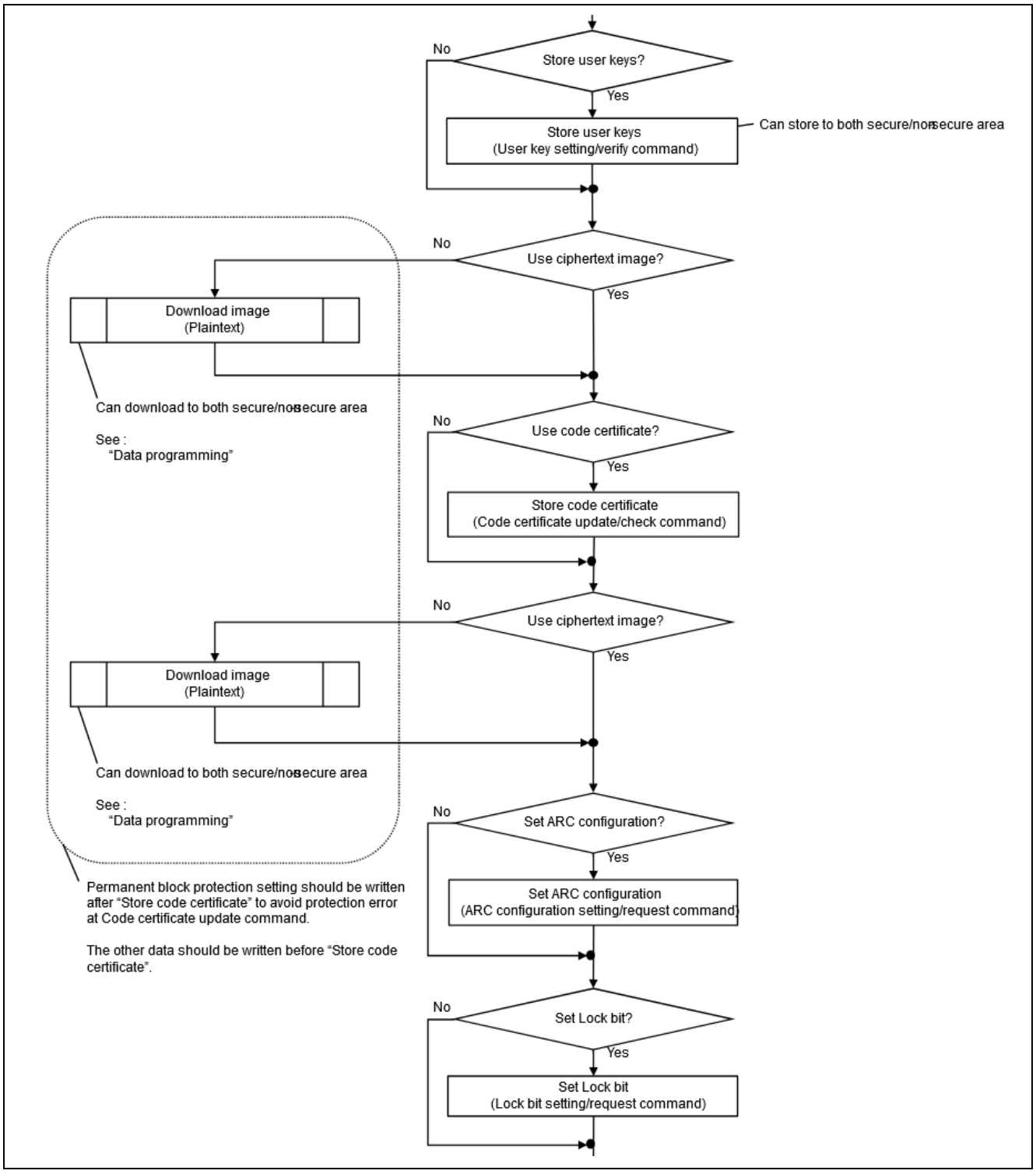


Figure 50. Downloading Whole Image (Part 2)

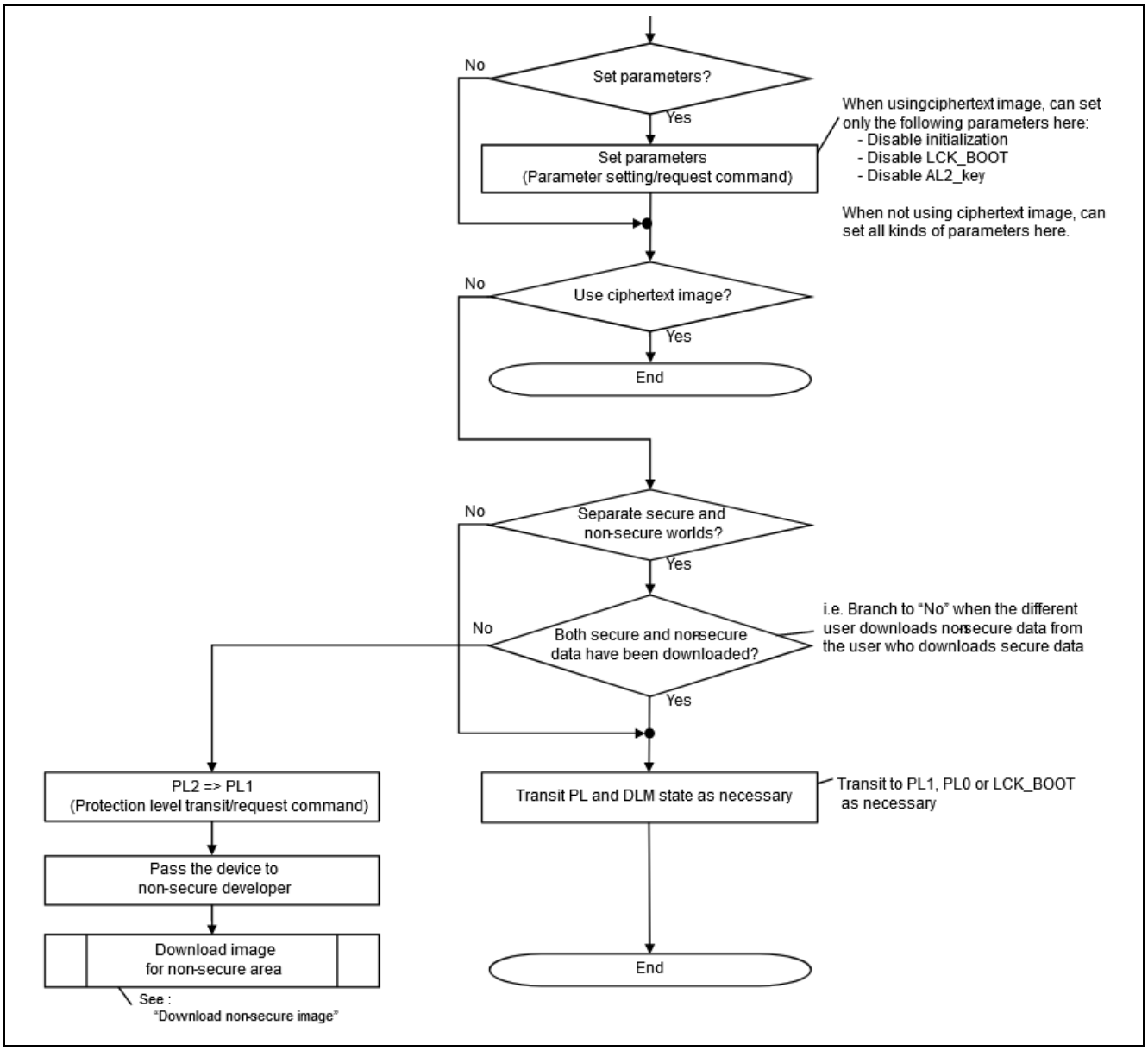


Figure 51. Downloading Whole Image (Part 3)

7.10 Downloading Non-secure Image

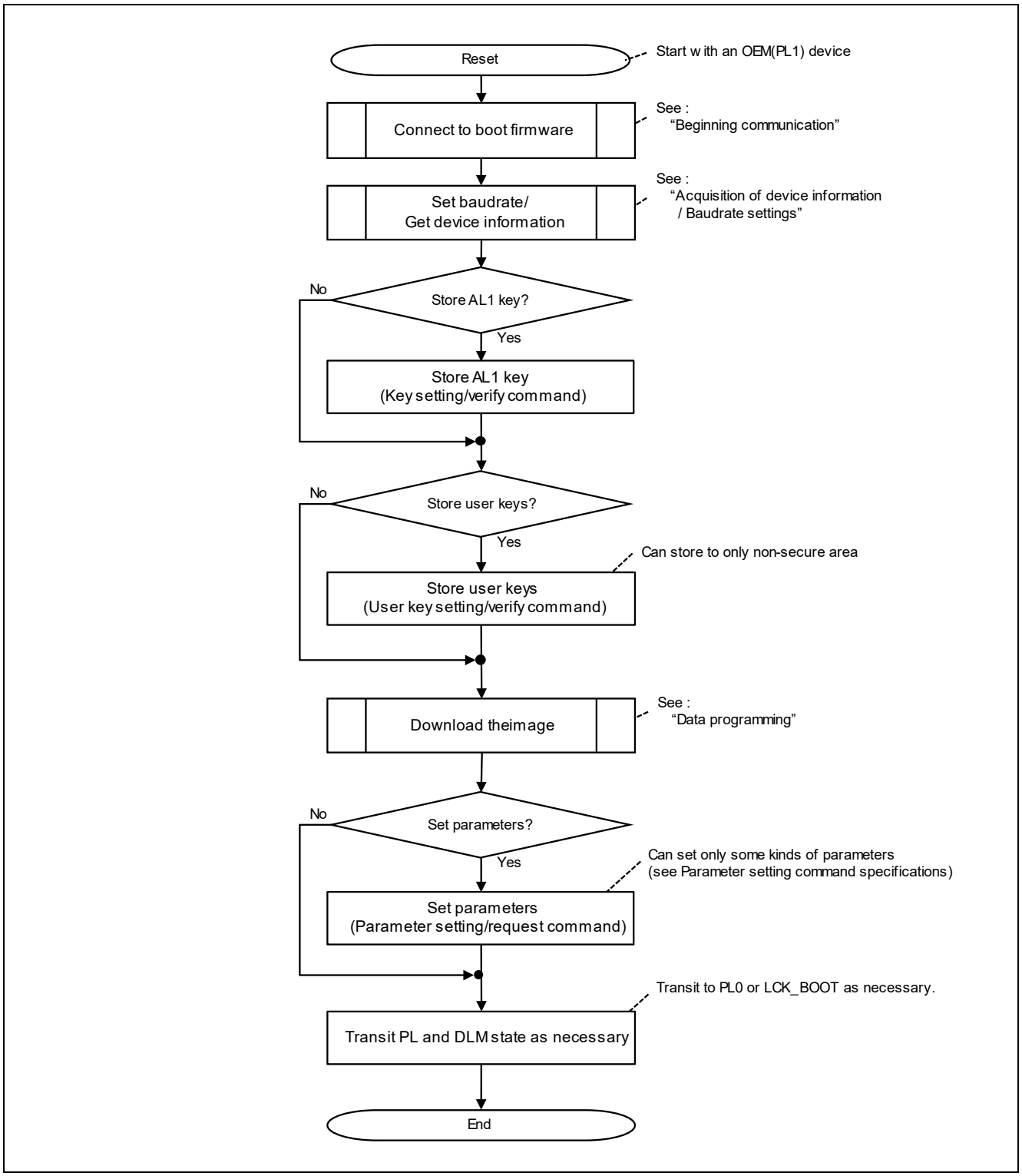


Figure 52. Downloading Non-secure Image

7.11 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.

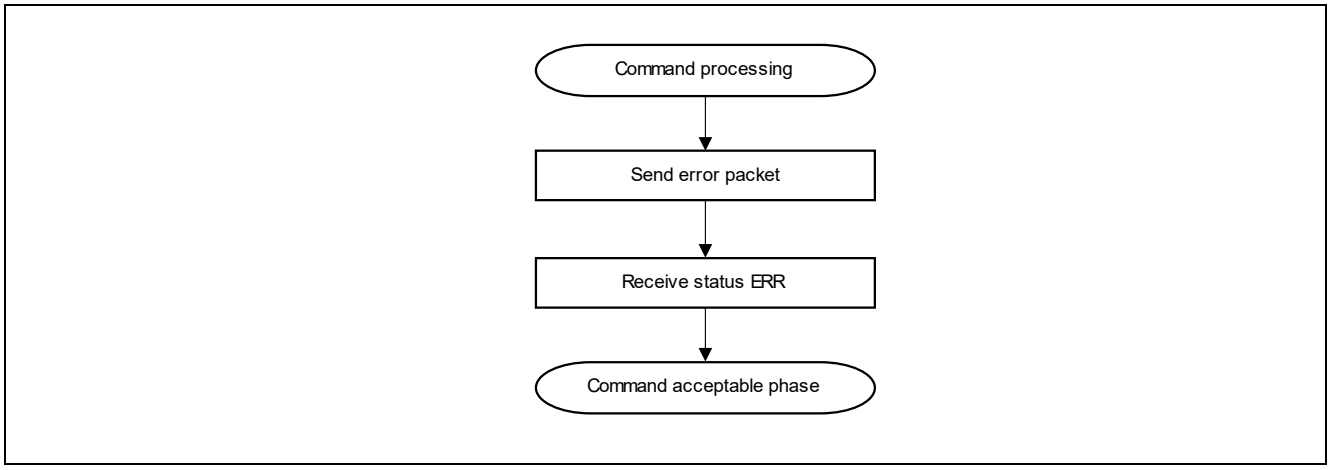


Figure 53. Command Cancel

Example: 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]	SOD	(1 byte)	81h
Key setting command	Data packet [key data]	LNH	(1 byte)	00h
User key setting command	Data packet [key data]	LNL	(1 byte)	01h
Write command	Data packet [write data]	RES	(1 byte)	FFh (ERR)
Read command	Data packet [status OK]	SUM	(1 byte)	00h
Encrypted data write command	Data packet [Key/encrypted user data]	ETX	(1 byte)	03h

8. AC Characteristics

8.1.1 Communication Setting Phase

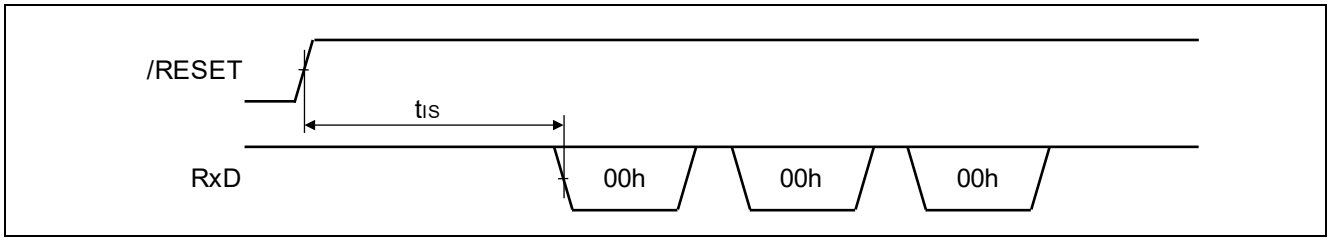


Figure 54. 2-wire UART Communication

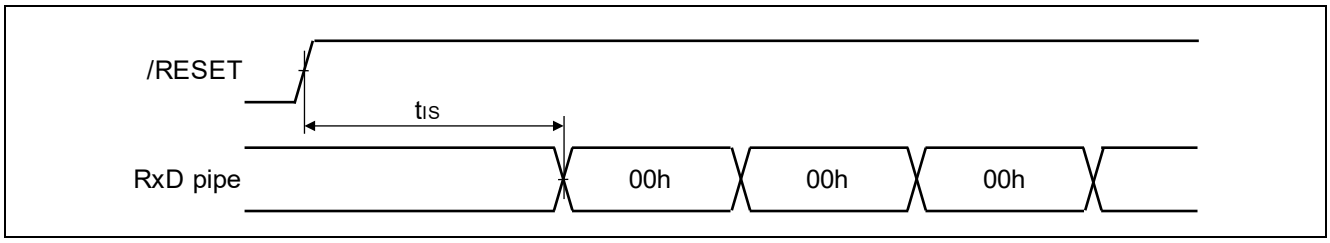


Figure 55. USB Communication

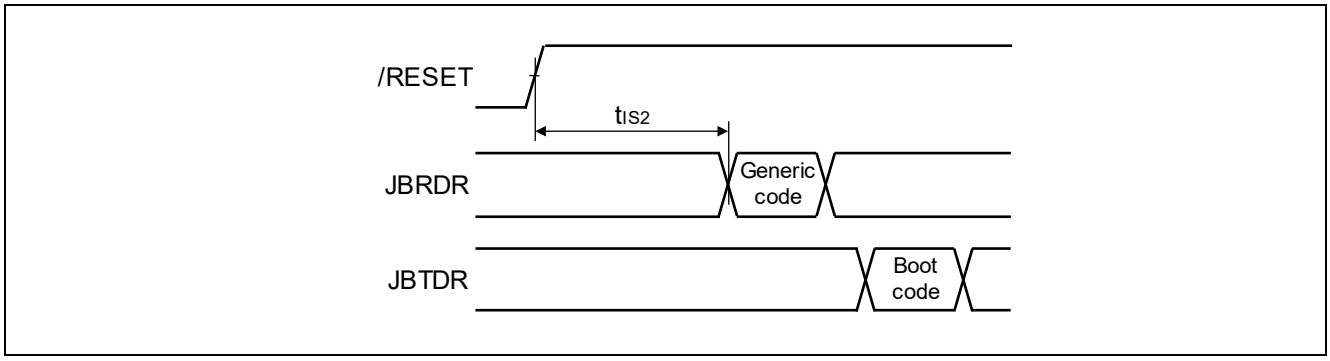


Figure 56. JTAG/SWD Communication Initial Setting Time

Parameter	Symbol	Min	Typ	Max	Unit
Initial setting time (when using Main-OSC (External clock))	tIS	-	-	326	ms
Initial setting time (when using Main-OSC (Crystal resonator)	tIS	-	-	505	ms
Initial setting time (when using HOCO)	tIS	-	-	774	ms
Initial setting time 2	tIS2	-	-	200	ms

8.1.2 DLM State Transit Command

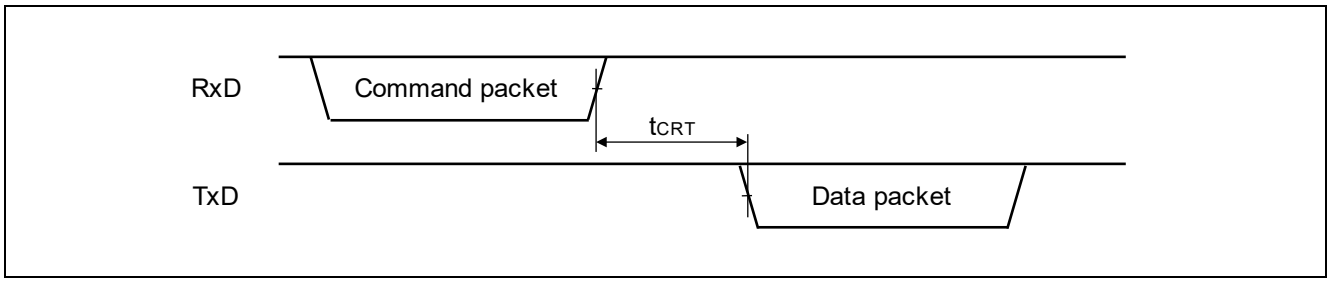


Figure 57. DLM State Transit Command

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

8.1.3 DLM State Request Command

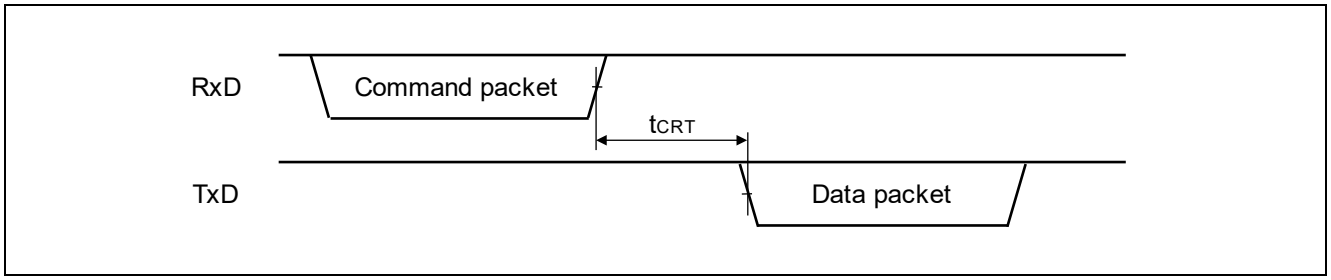


Figure 58. DLM State Request Command

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

8.1.4 Authentication Command

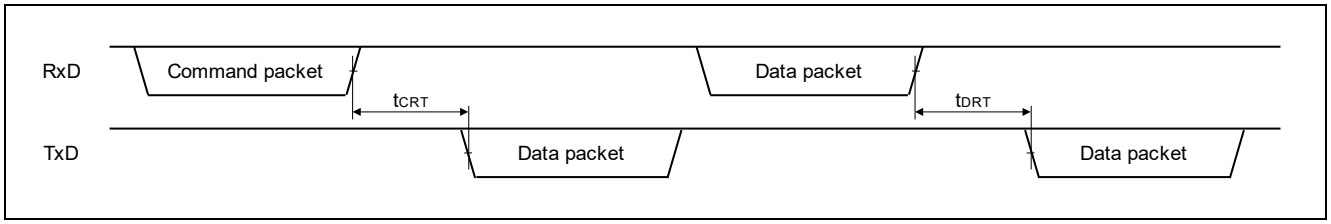


Figure 59. Authentication Command

Parameter	Symbol	Min	Typ	Max	Unit
Command response time	tCRT	-	-	3	s
Data response time	tDRT	-	-	180	s

8.1.5 Key Setting Command

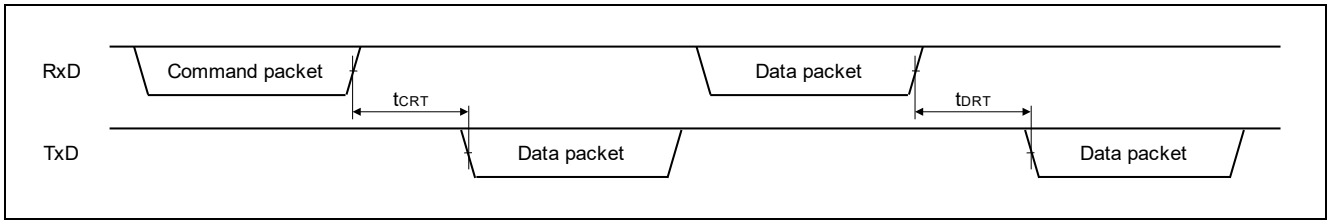


Figure 60. Key Setting Command

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

8.1.6 User Key Setting Command

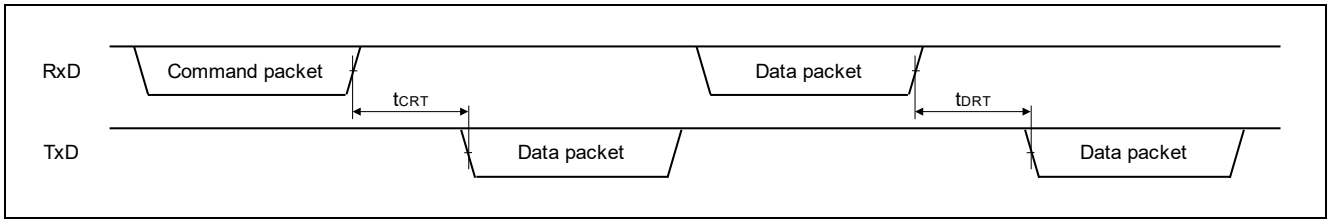


Figure 61. User Key Setting Command

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

8.1.7 Key Verify Command

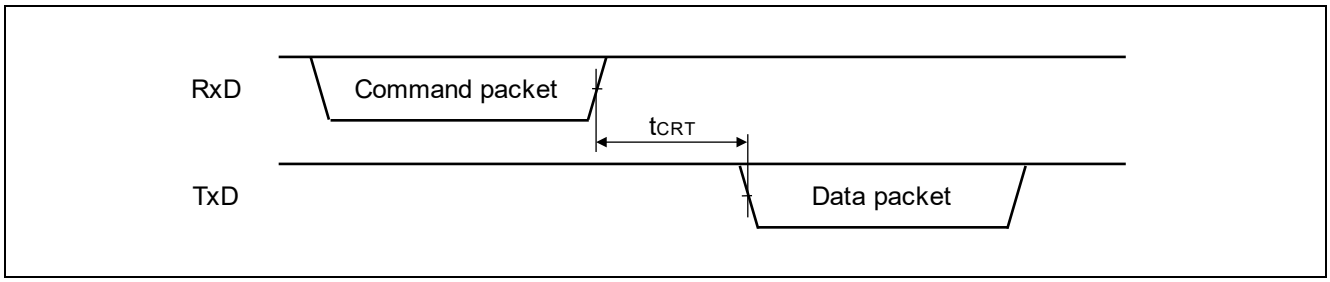


Figure 62. Key Verify Command

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

8.1.8 User Key Verify Command

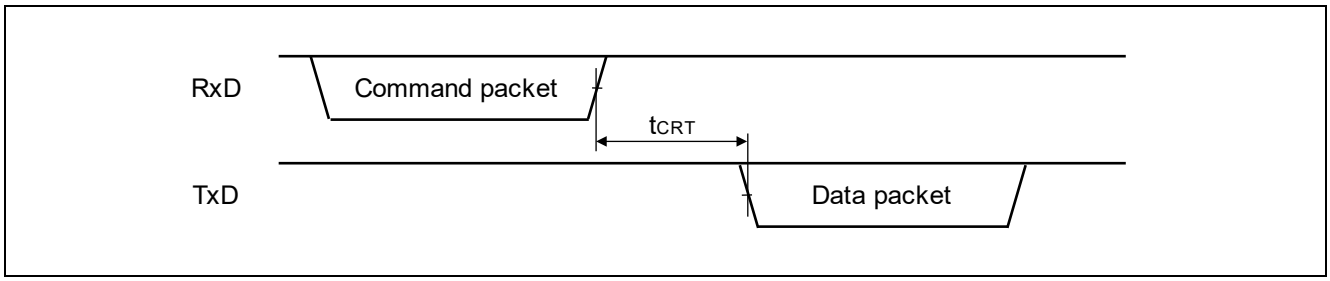


Figure 63. User Key Verify Command

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

8.1.9 Initialize Command

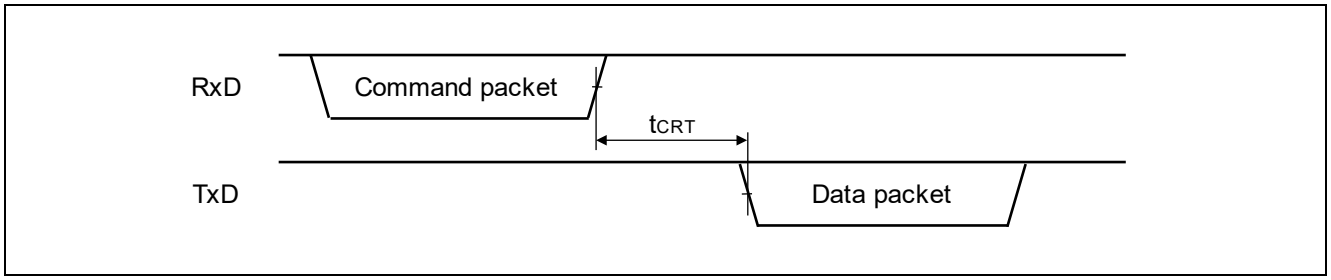


Figure 64. Initialize Command

Parameter	Symbol	Min	Typ	Max	Unit
Command response time	tCRT	-	-	180	s

8.1.10 Boundary Setting Command

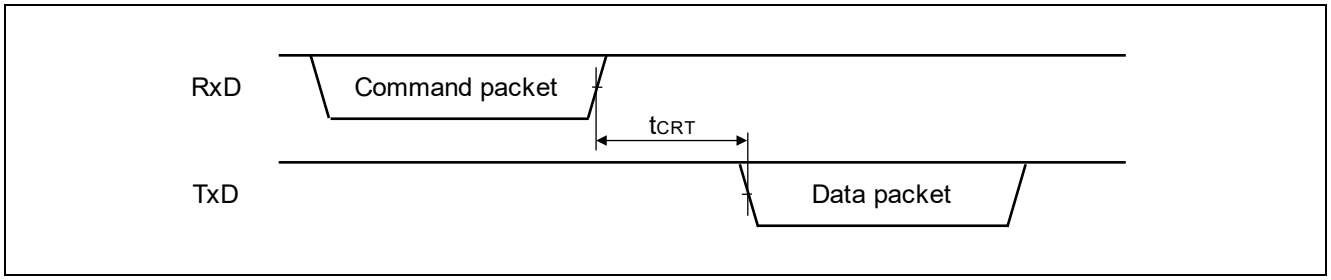


Figure 65. Boundary Setting Command

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

8.1.11 Boundary Request Command

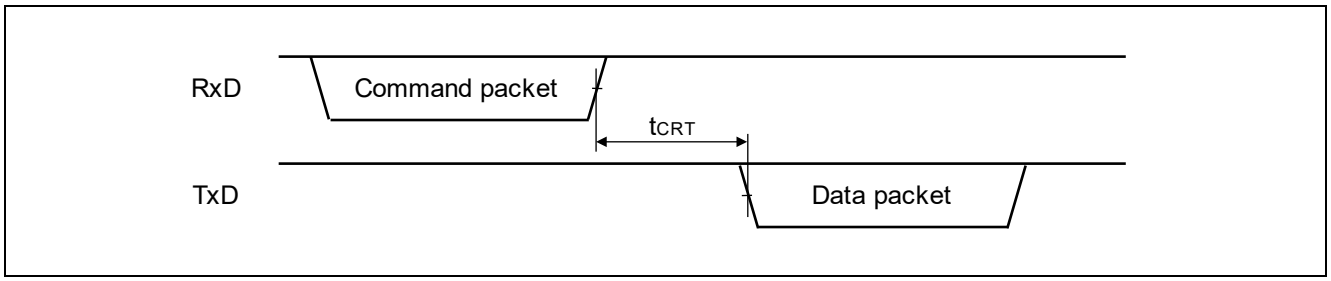


Figure 66. Boundary Request Command

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

8.1.12 Parameter Setting Command

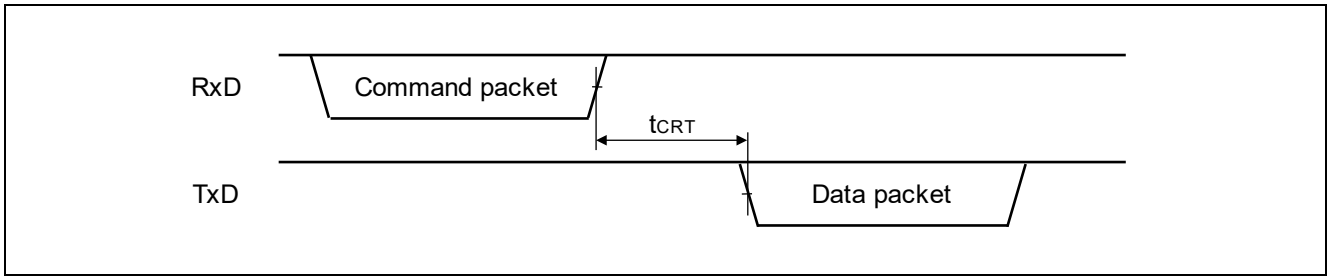


Figure 67. Parameter Setting Command

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

8.1.13 Parameter Request Command

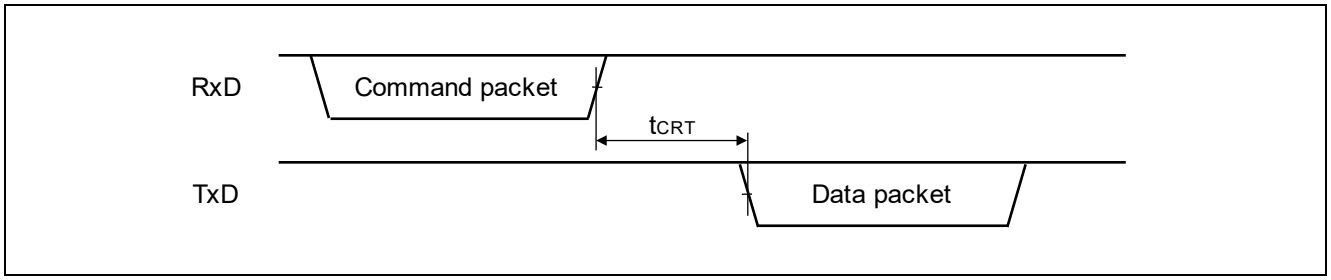


Figure 68. Parameter Request Command

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

8.1.14 Inquiry Command

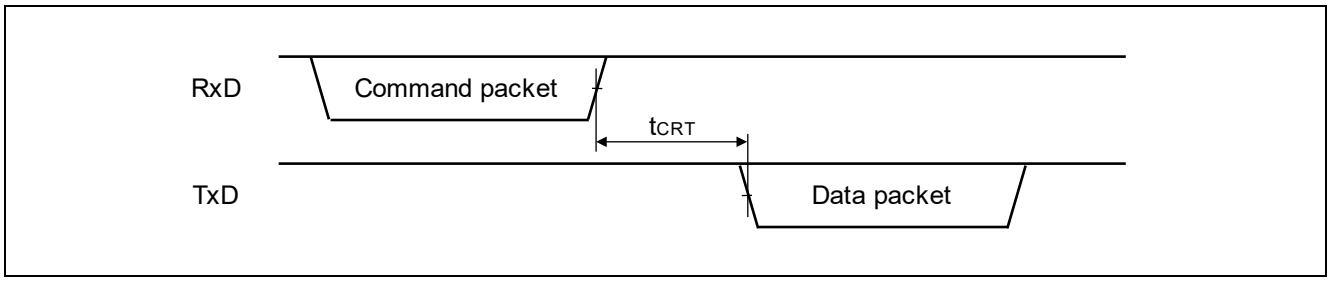


Figure 69. Inquiry Command

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

8.1.15 Signature Request Command

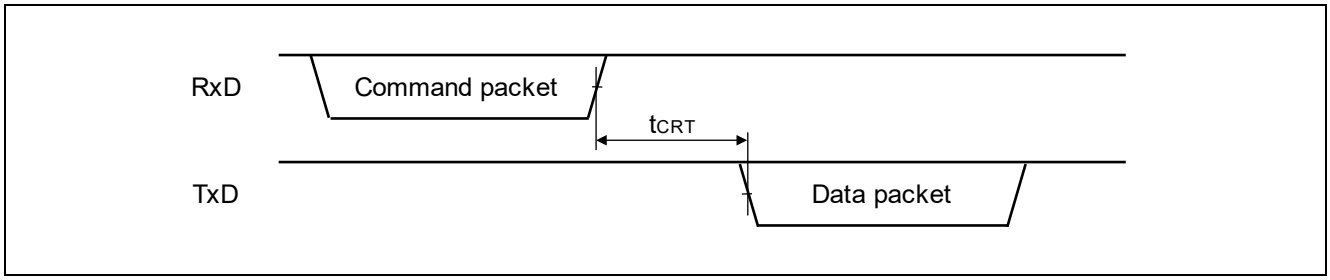


Figure 70. Signature Request Command

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

8.1.16 Area Information Request Command

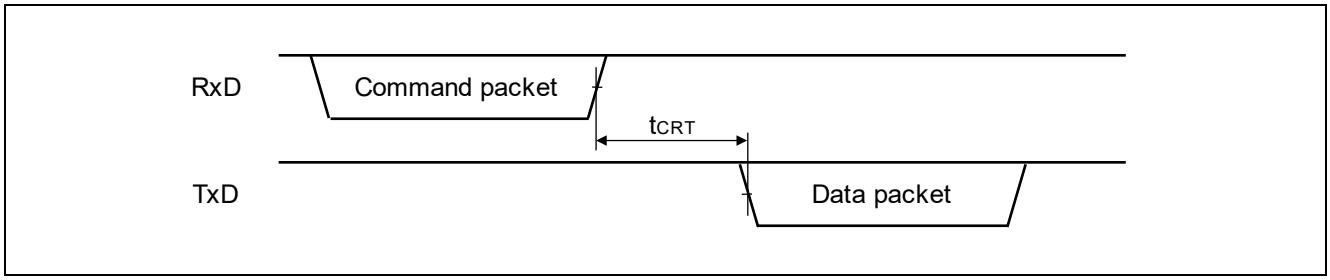


Figure 71. Area Information Request Command

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

8.1.17 Baudrate Setting Command

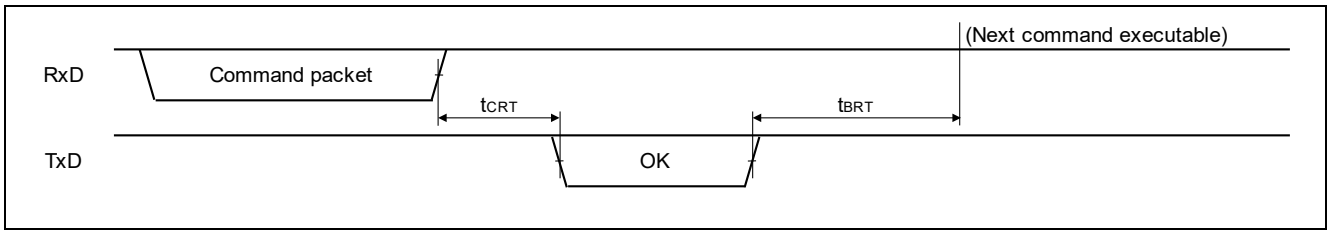


Figure 72. Baudrate Setting Command

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

8.1.18 Erase Command

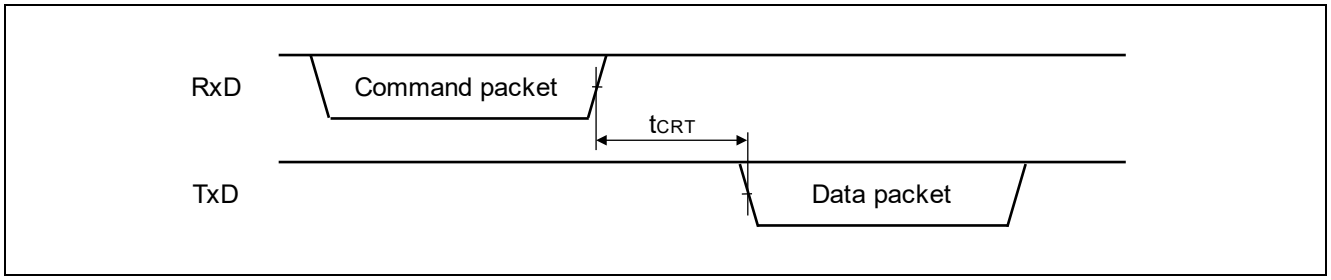


Figure 73. Erase Command

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

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

8.1.19 Write Command

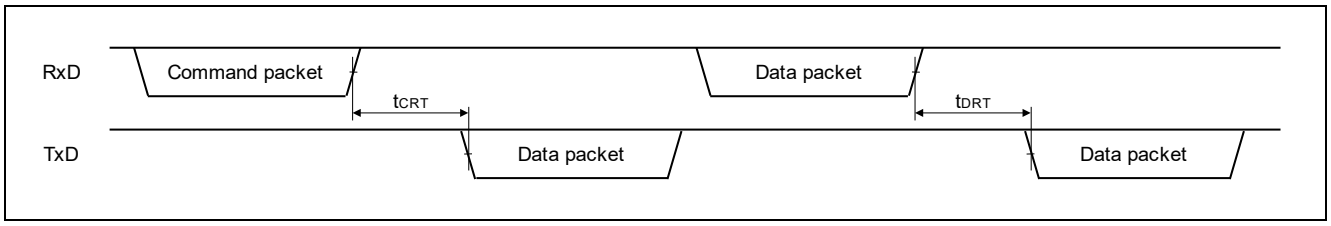


Figure 74. Write Command

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

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

8.1.20 Read Command

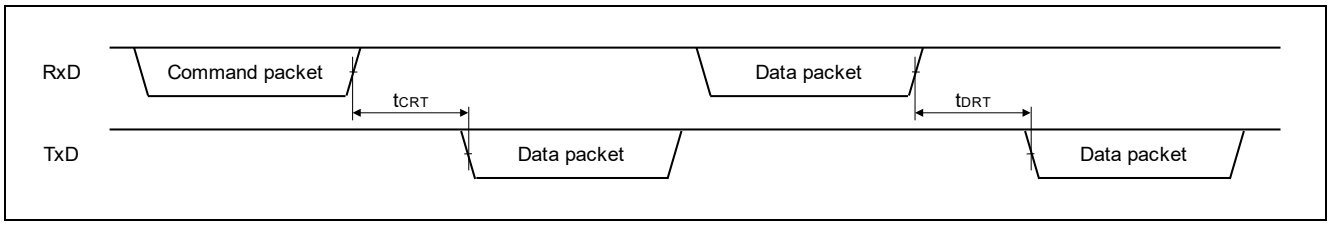


Figure 75. Read Command

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

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

8.1.21 CRC Command

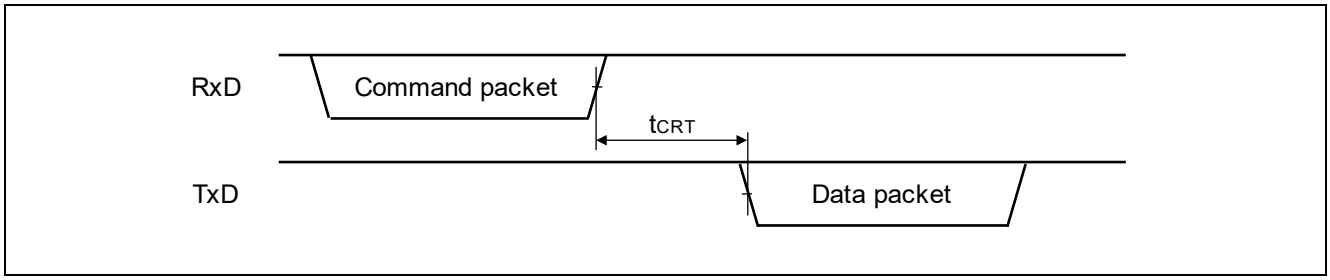


Figure 76. CRC Command

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

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

8.1.22 Encrypted Data Write Command

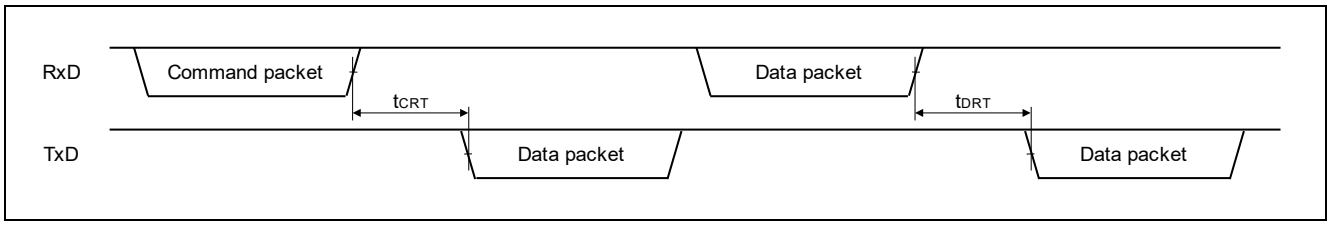


Figure 77. Encrypted Data Write Command

*) 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	-	-	180	s
Data response time	tDRT	-	-	60	s

9. Sequencer Command List

Table 21 shows the sequencer commands executed by each communication command.

Table 21. Sequencer Command List

Communication command	Sequencer command	Number of issue times
DLM state transit command	Configuration set	1 time
	Forced stop *) Use to clear error status.	1 time
Authentication command	Block Erase	[Transiting to RMA_REQ] [Size of User area / 2K] – [Number of Blocks in which PBPS is set] + [Size of Data area / 256] times
	Configuration set	Transiting to state other than RMA_REQ: 1 time • 4 times at BPS initialization • 9–10 times at Config area initialization (depending on FSPR state) • 13 times at Boot region initialization • 1 time at DLM transition
	Forced stop *) Use to clear error status	Depends on the DDLM and the result of all erasure at the RMA_REQ transition
	Configuration set	4 times

Communication command	Sequencer command	Number of issue times
Key setting command	Forced stop *) Use to clear error status	1 time
User key setting command	Program	Depends on the designated address and key type
	Forced stop *) Use to clear error status	1 time
Initialize command	Block Erase	[Size of User area / 2K] + [Size of Data area / 256K]
	Configuration set	30 times: • 4 times at BPS initialization • 12 times at Config area initialization • 13 times at Boot region initialization 1 time at PL transition
	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
Parameter setting command	Configuration set	1 time
	Forced stop *) Use to clear error status	1 time
Erase command	Block Erase	Depends on the designated address
	Forced stop *) Use to clear error status	1 time
Write command	Program	Depends on the designated address
	Configuration set	Depends on the designated address
	Forced stop *) Use to clear error status	1 time
Encrypted data write command	Program	Depends on the designated address
	Block Erase	[Size of User area / 2K] + [Size of Data area 256] times
	Configuration set	Depends on the designated address
	Forced stop *) Use to clear error status	1 time

10. Write Command

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.

10.1.1 Encrypted Data Write Command

1. This command becomes inexecutable once after permanent block protection is set.
2. This command becomes inexecutable if SAS.BTFLG=0b and SAS.FSPR=0b.
3. If permanent block protection in the Config area is written before the protected area, this command abnormally finishes at the writing of the protected area.
   To avoid this, Data packets [encrypted user data] for the protected areas must be sent earlier than Data packets for the permanent block protection 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.
5. 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 note that 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 1 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 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 when the USB status is "Configured".
- When 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 when the USB status is "Configured".
- When 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 Hardware error occurred.

11.5 Authentication Command

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

11.6 Key Setting Command

- When Trusted system goes into an abnormal state.

11.7 User Key Setting Command

- When Trusted system goes into an abnormal state.

11.8 Key Verify Command

- When Trusted system goes into an abnormal state.

11.9 User Key Verify Command

- When Trusted system goes into an abnormal state.

11.10 Initialize Command

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

11.11 Encrypted Data Write Command

- When 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 Initialization Phase

- When the DLM state is LCK_BOOT after startup.
- When the DLM state is abnormal after startup.

12.2 Communication Setting Phase

- When MD=1 is detected, and also SWD boot mode is not requested during communication mode judgement.

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	Feb.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/.