Renesas RA Family

Renesas Boot Firmware for RA2L2 MCU Group

Introduction

This application note describes the communication protocol, command set, and usage of the boot firmware provided with the Renesas RA2L2 MCU Group.

Target Device

RA2L2 MCU Group

Contents

1. Terminology

1.1 Boot Firmware

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

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 Flash Memory Structure Example shows an example of a flash memory structure. Memory structure differs from device to device.

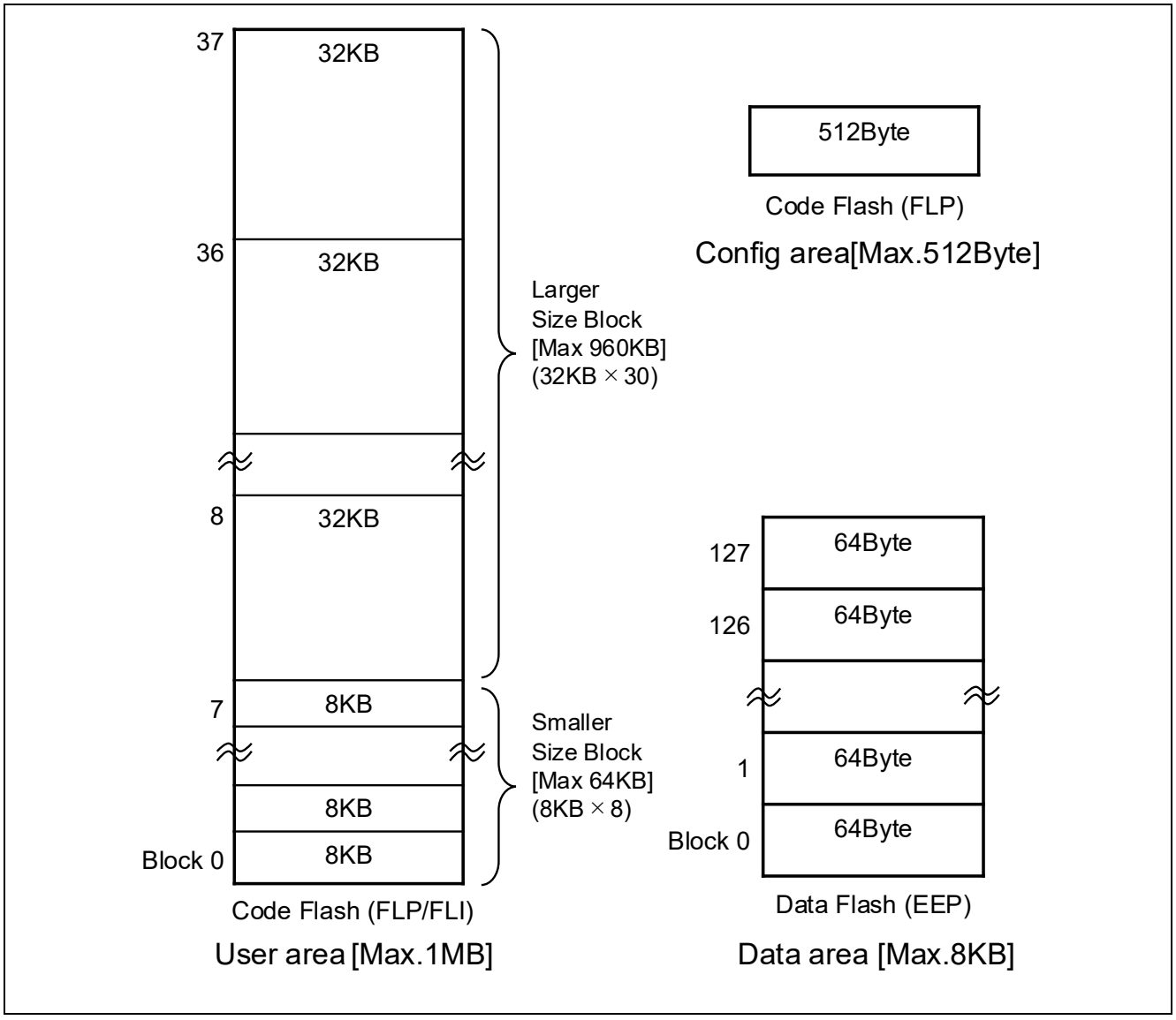


Figure 1. Flash Memory Structure Example

1.3 AW: Access Window

The function for assigning accessible sectors to erase and write in code flash is called Access Window (AW). This function allows access from the start sector to the end sector and disallows access to other areas. The user set a value corresponding to "start sector" to the FAWS, and also set a value corresponding to "end sector + 1" to the FAWE. If FSPR is 0, FAWS and FAWE cannot be changed. For details, please refer to the Flash Memory User's Manual.

Table 1. An Example of Flash Memory Structure

Sector No.	Base address	Size	Setting value
0	00000000h	2KB	000h
1	00000800h	2KB	001h
:	:	:	:
62	0001F000h	2KB	03Eh
63	0001F800h	2KB	03Fh

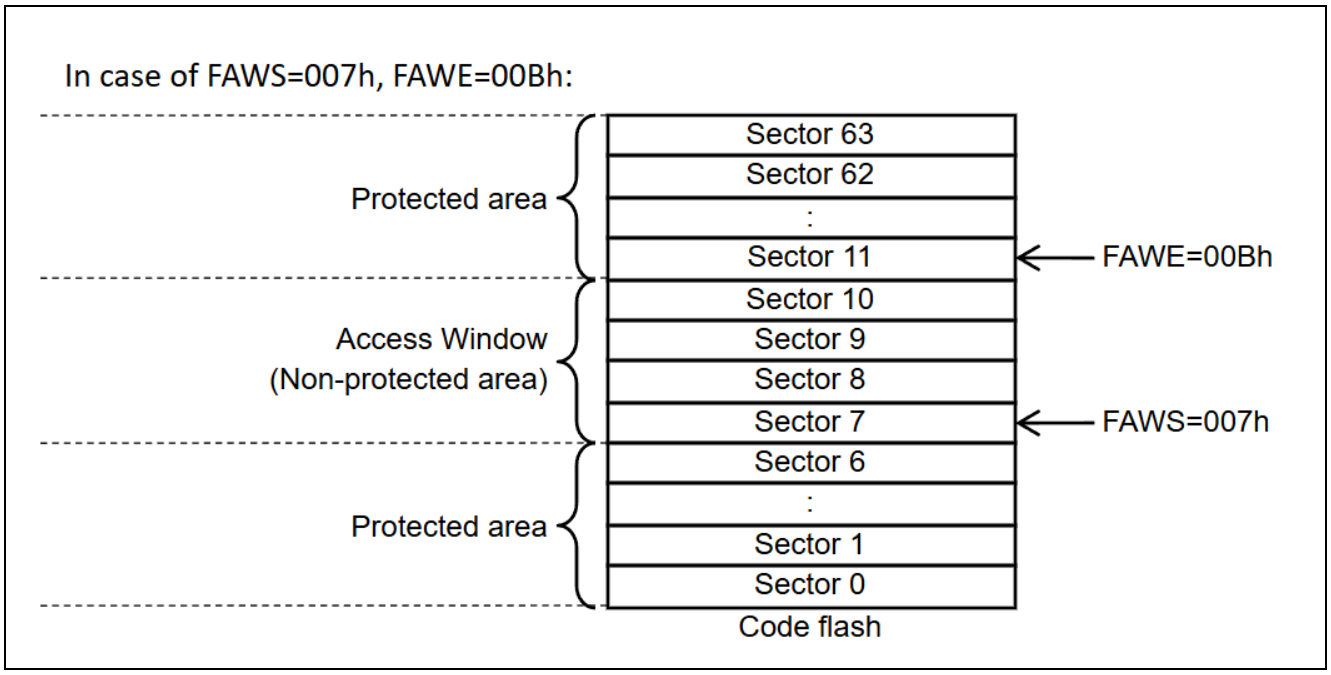


Figure 2. Flash Access Window

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 RA2L2 MCU Group

This chapter describes the system architecture of RA2L2 MCU Group regarding the flash memory control.

Table 2. Operating Environment

CPU core	Arm Cortex-M23
Max CPU operating frequency	48MHz Boot firmware operating frequency is as follows: - 32 MHz when VCC is higher than 1.8 V - 4 MHz when VCC is lower than or equal to 1.8 V
Main-OSC	Unnecessary

<table>
<tr><td>Operating voltage</td><td>VCC = 1.6 - 5.5 V
*When using I2C communication, VCC must be 2.7 V or higher</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. 128 KB</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. 4 KB</td></tr>
<tr><td>Extended data area</td><td>None</td></tr>
</table>
</td></tr>
<tr><td>RAM</td><td>SRAM: Max. 16 KB (used by the Boot firmware: 8 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 1.8 V is 115200 bps
*) The max bps is determined by the VCC level at the boot firmware’s startup processing, and returned by the RMB of the Signature request command

[Inter-Integrated Circuit communication]
400 Kbps (Max)
*) Gang programming operation, programming operation under the condition that one host is connected to more than one RA2L2 microcontroller, is not supported.</td></tr>
</table>

2.1.1 Block Diagram

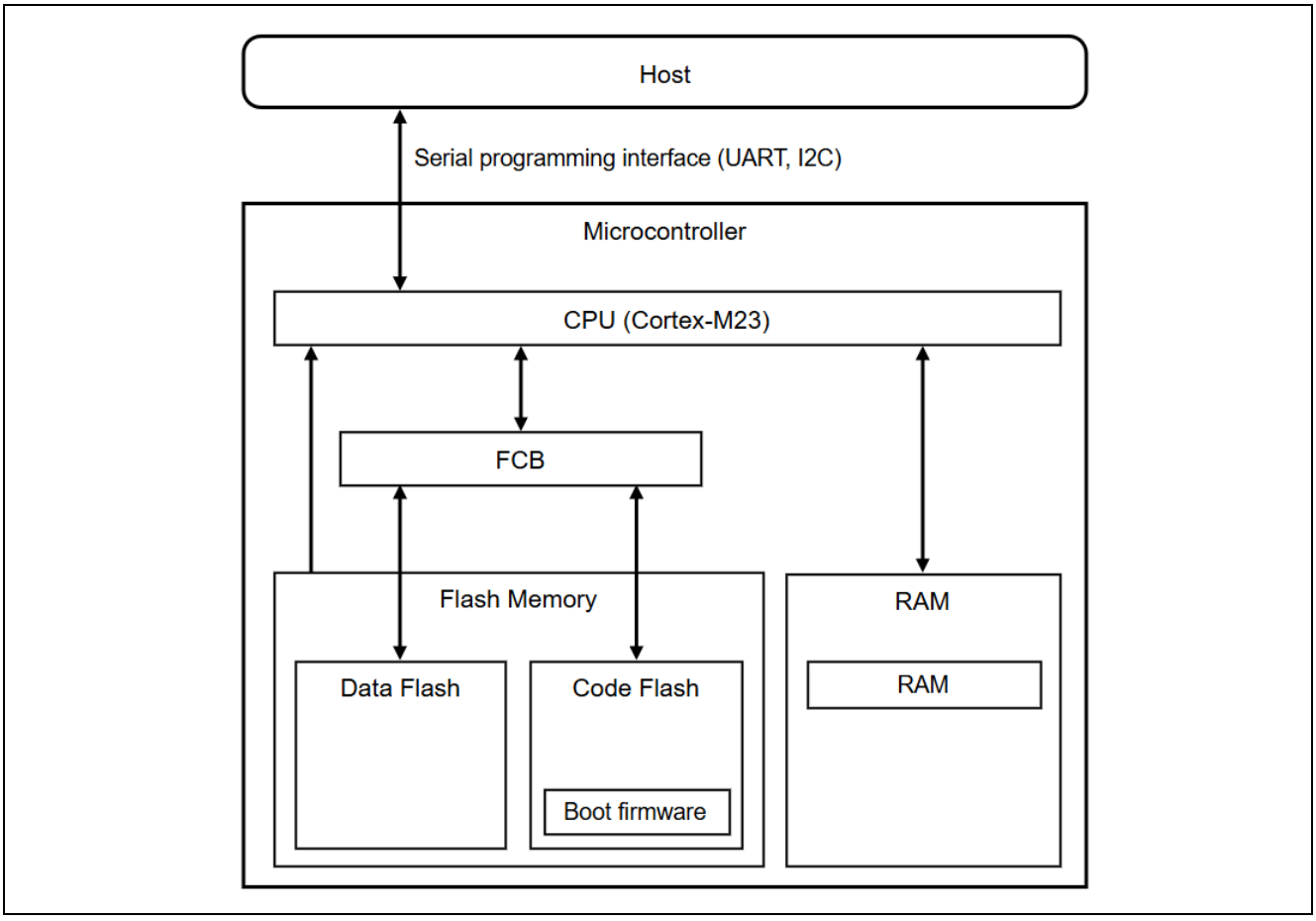


Figure 3. Block Diagram

3. Communication Method

Boot firmware has interfaces for the following communication methods:

- 2-wire UART communication
- Inter integrated Circuit(I2C) communication

3.1 2-wire UART communication

Boot firmware supports 2-wire UART communication.

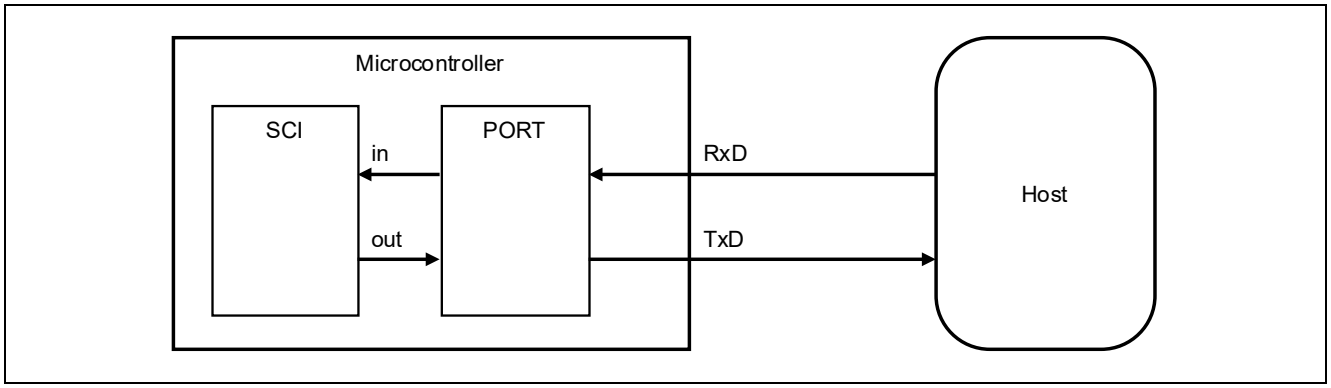


Figure 4. 2-Wire UART Communication

Table 3. UART Settings

General settings	
Interface	SCI ch9
RxD	P110, Input mode
TxD	P109, Output mode
Transfer rate	9600bps (Min, Until the baudrate setting command) 2Mbps (Max)
Data length	8bit (LSB first)
Parity bit	none
Stop bit	1 bit

Communication is performed at 9600bps until the baudrate setting command. After the baudrate setting command is completed normally, 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.
* If the communication cable is disconnected during communication, subsequent operations are not guaranteed.

3.2 Inter Integrated Circuit (I2C) Communication

Boot firmware supports I2C communication.

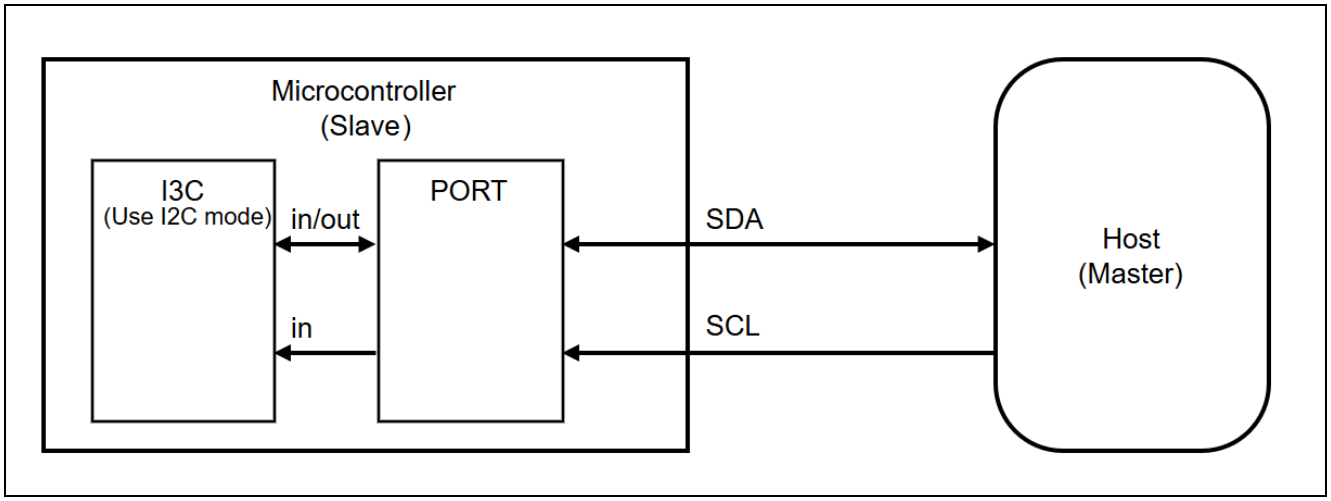


Figure 5. Inter Integrated Circuit (I2C) Communication

Table 4. I2C Communication

General settings	
Interface	I3C (I2C mode)
SDA	P912
SCL	P913
Transfer rate	Max 400 Kbps (Fast mode)
Slave address	0b001 0001
General call address	Not supported
Device ID	Not supported
Software reset	Not supported
Address frame	7-bit address
Clock stretch	Enabled but not used (See [Clock stretch] below)

3.3 Samples of I2C frame

Samples of the I2C frame used for the I2C boot are shown below.

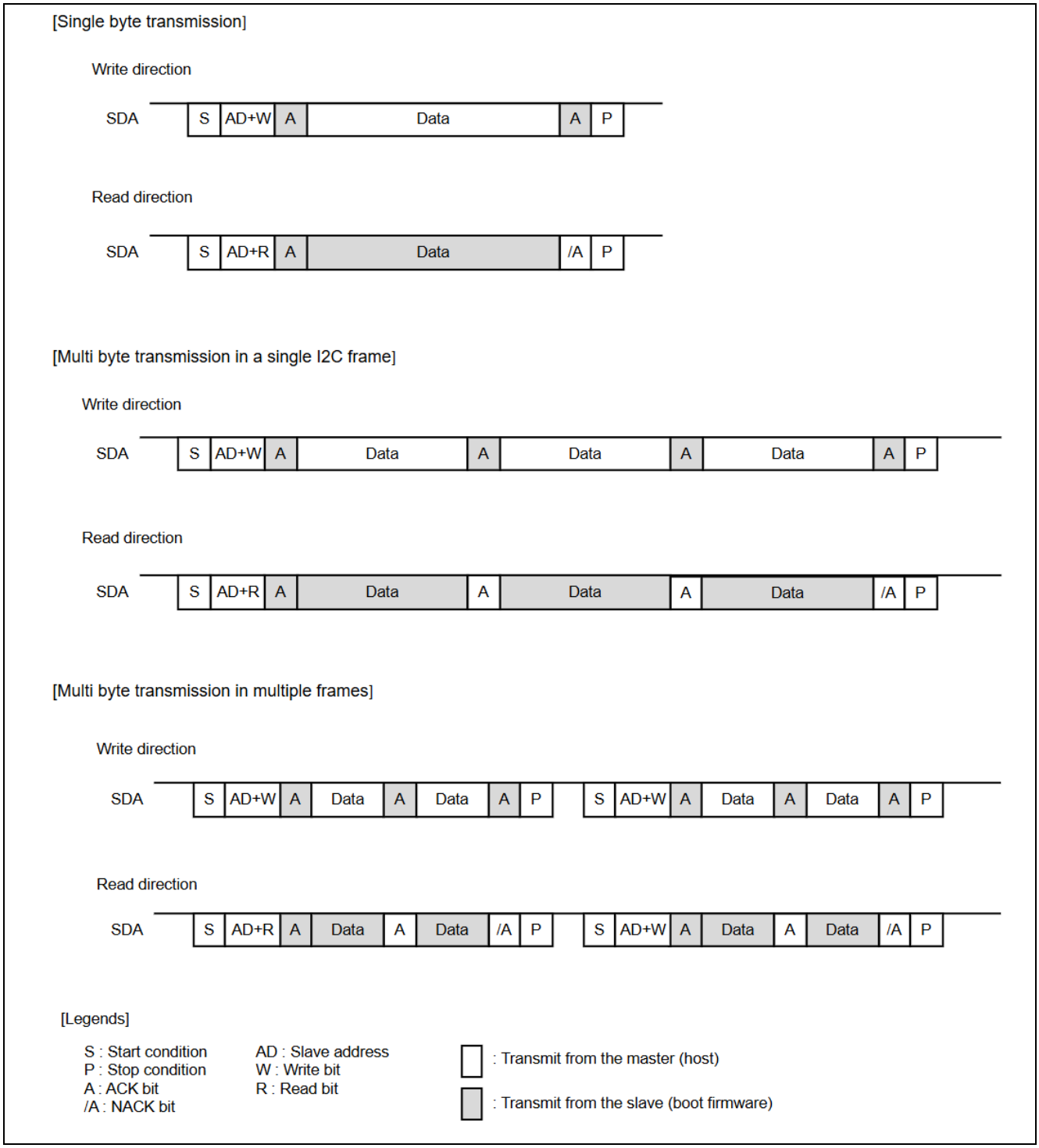


Figure 6. Samples of I2C Frame

3.3.1 ACK Bit Polling

Boot firmware returns not the ACK bit but the NACK bit when not being ready to receive/send data, such as the following cases:

- When boot firmware has not finished I2C initialization.

- After boot firmware received data or a packet from the host, and is not yet ready to return the data to the host, e.g., during command processing.

The host can determine whether an I2C communication can be started by checking the ACK bit.

When a NACK bit is detected, the host must issue a stop condition and try again from a start condition.

By repeating these steps, the host can perform polling to determine when the boot firmware becomes ready to start I2C communication.

3.4 Example of ACK Bit Polling

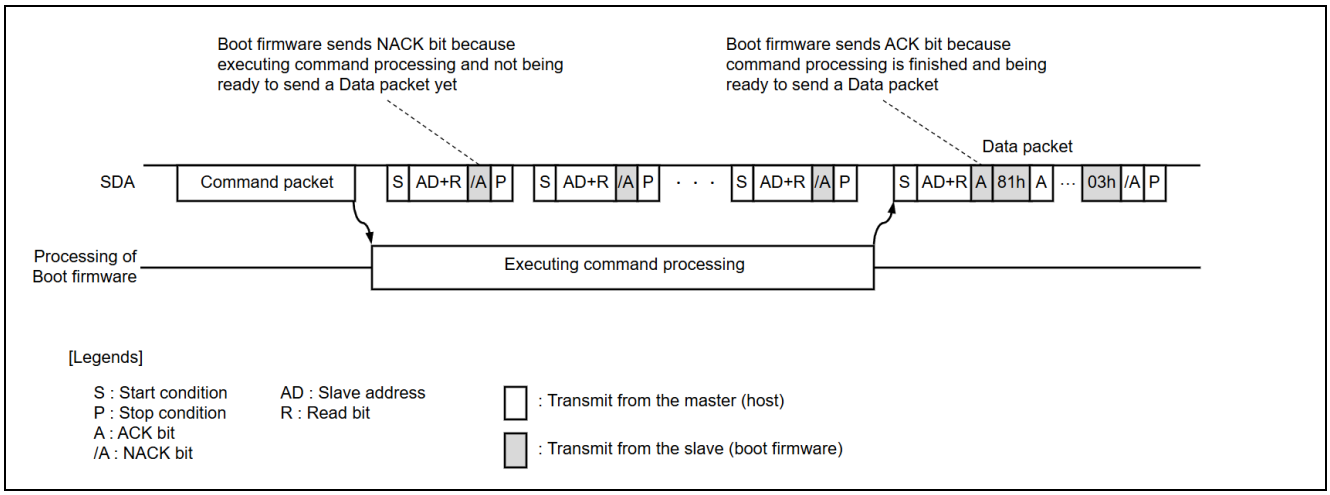


Figure 7. Example of ACK Bit Polling

3.4.1 Timing of NACK Bit

Boot firmware returns the NACK bit only at the timing described above (ACK polling).

The host can return NACK only before a Stop Condition, to notify the boot firmware that the read direction I2C frame finishes.

3.4.2 Clock Stretch

Clock stretch, a low hold operation of SCL by a slave device, is enabled since it cannot be disabled because of HW specifications.

Although this low hold occurs after an AD+R/W bits when the slave device is not ready to start an I2C communication in general, boot firmware does not hold SCL low in such cases but returns NACK instead as ACK polling, considering that the host does not support a slave device that does clock stretching.

However, if an I2C communication that does not comply with the communication protocol of the I2C boot is issued, a clock stretch occurs unintentionally since it is not actually disabled as described.

For example, a read direction I2C frame is started even when a command packet is NOT sent to boot firmware, and a clock stretch occurs after the AD+R bits of the I2C frame.

3.4.3 Other Precautions

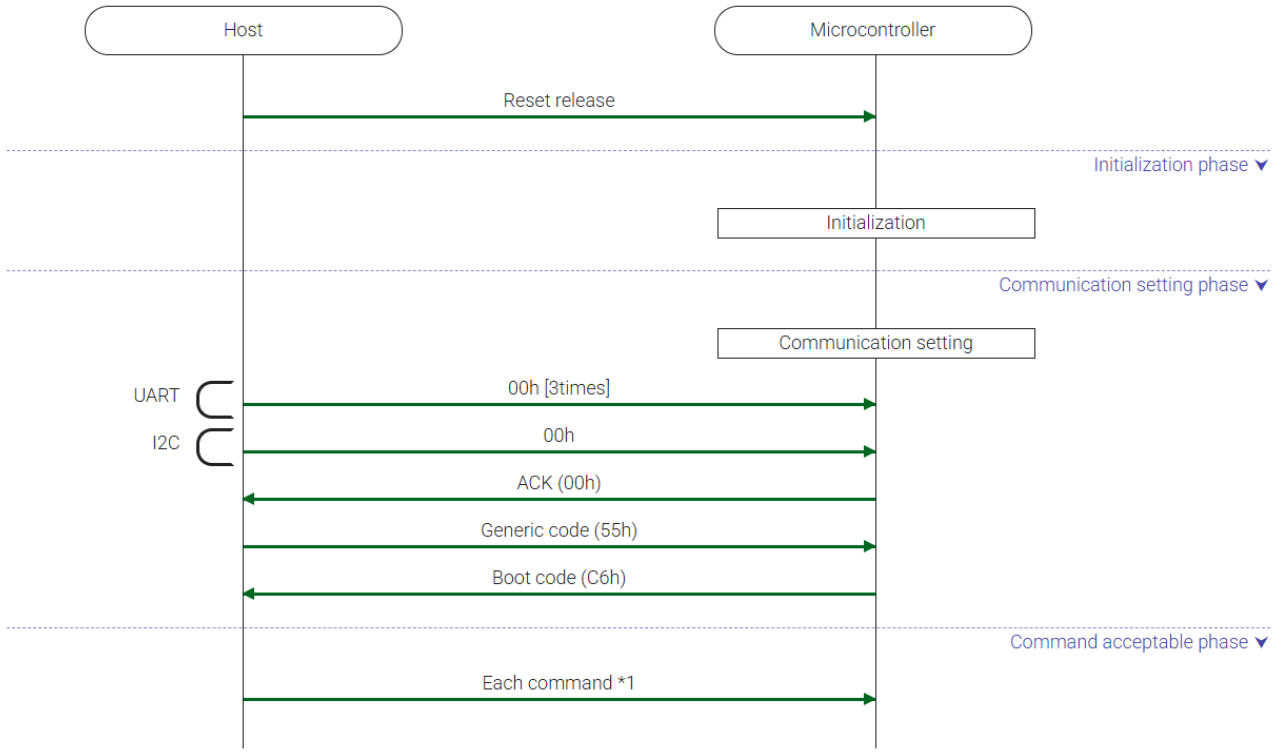
- A repeated start condition is not used for the communication protocol of the I2C boot. Operation when a repeated start condition is sent is not guaranteed (not tested).
- Therefore, a start condition can be issued only when the bus is free.
- A stop condition can be issued only after an ACK/NACK bit. Do not issue a stop condition at other timing, during a data bit, for example.
- Multi master operation, operation under an environment where multi masters communicate on the same bus, is not guaranteed.

- If the communication cable is disconnected during communication, subsequent operations are not guaranteed.

4. General Procedure

Boot firmware transitions in the following order after the reset release. This sequence is non-invertible.

4.1 Sequence Diagram (Generic Sequence)

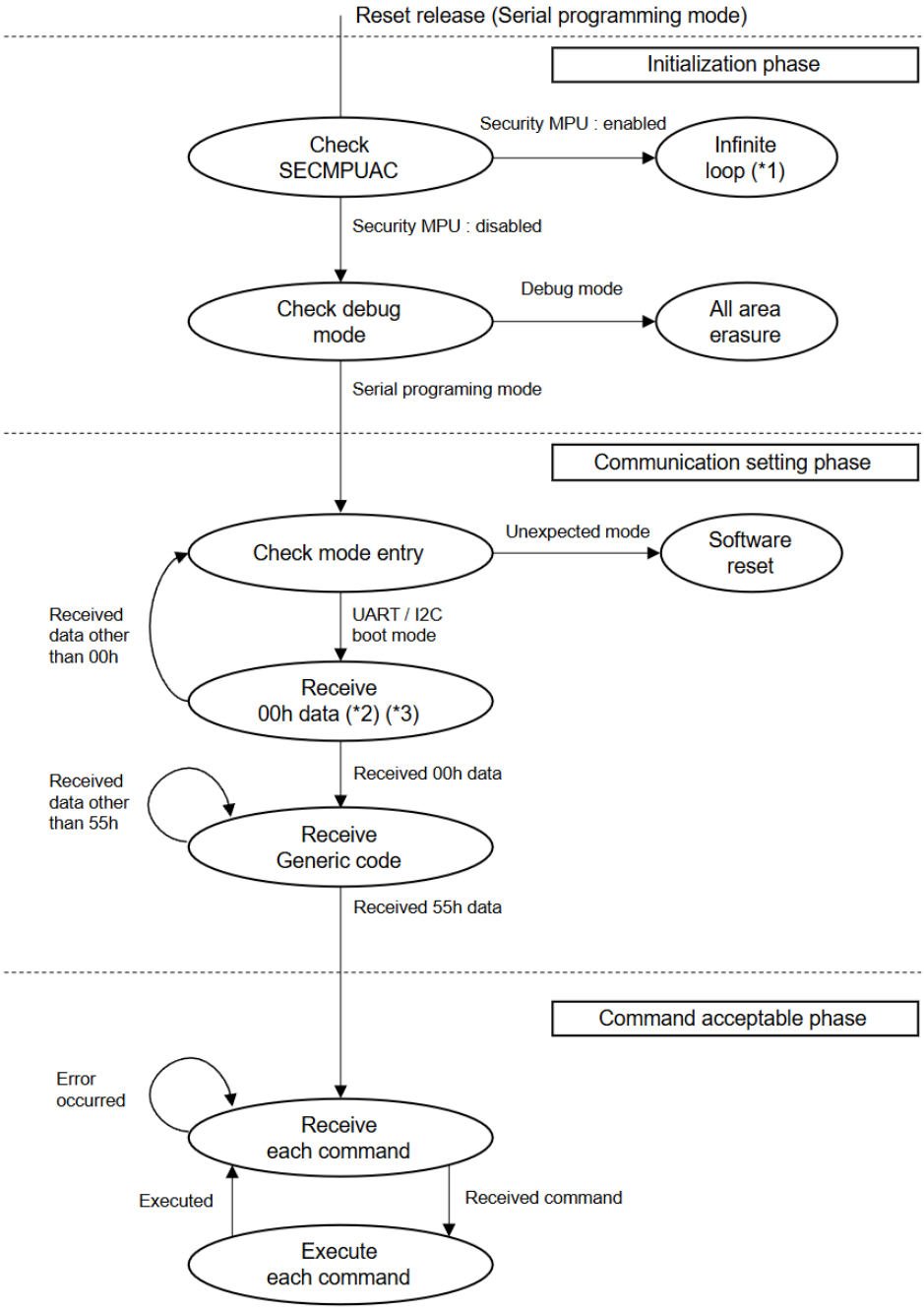


*1: From the Command acceptable phase, Host and Microcontroller send data in turn unless otherwise noted. The host shall execute the data transmission after the data reception from the microcontroller.

Figure 8. State Transition Diagram (Generic State Transition)

4.2 State Transition Diagram (Generic State Transition)

Boot firmware selects Serial programming mode or Debug mode based on the MD pin level at reset timing. If the operating mode is Serial programming mode, the boot firmware transitions to the Communication setting phase. If the operating mode is not Serial programming mode, the boot firmware erases all of the User area and Config area, and then it goes into an infinite loop. If the Security MPU is valid, the boot firmware enters an infinite loop without transitioning to either mode.



*1: Boot firmware performs a software reset for re-entry to Single-chip mode when detecting MD=1, even when the Security MPU is enabled and executing an infinite loop.
*2: Boot firmware adopts SCI-IP when it receives three consecutive 00h in 2-wire UART communication.
*3: Boot firmware adopts I3C-IP(use I2C mode) when it receives one time 00h in I2C communication.

Figure 9. State Transition Diagram (Generic State Transition)

Table 5. Check SECMPUAC

Condition	Next operation
SECMPUAC = FFFFh	Check Debug mode
SECMPUAC != FFFFh	Infinite loop

Table 6. Check Debug Mode

Condition	Next operation
MD = 1(*1)	Debug mode (Accepts all area erasure processing)
MD = 0(*1)	Serial programming mode

4.3 Initialization Phase

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

4.3.1 Processing Procedure

Boot firmware initializes after reset release:

- Boot firmware initializes hardware modules and transitions to the "Communication setting phase".
- If the operation mode is debug mode, debug mode processing is performed.
  If the stored ID[127:126] in the device is not 11b, the boot firmware sets to standby mode and enters an infinite loop.
  If the stored ID[127:126] in the device is 11b, boot firmware erases all the flash memory, and if the erasure succeeds, sets sleep mode and enters an infinite loop.
  Also, if erasing fails, the boot firmware sets to standby mode and enters an infinite loop.

  If the FSPR in the config area is set to 0, the boot firmware sets to standby mode without executing all erasure and enters an infinite loop.
  * Flash memory status does not change before command reception.

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 the table below. After receiving the generic code using the established communication method, the boot firmware transitions to the "Command acceptable phase".

Table 7. Check Debug Mode

Condition	Communication method
- Data "00h" was continuously received 3 times by 2-wire UART communication	2-wire UART communication
- Data "00h" was received 1 time by I2C communication	I2C communication

4.4.1 Processing Procedure

Boot firmware performs communication settings:

- If MD=1, the boot firmware will perform a software reset.

- When either of the following conditions is met, "ACK" is transmitted. (* data is received until the communication mode is determined)
  - The boot firmware received 00h consecutively three times in 2-wire UART communication.
  - The boot firmware received 00h in I2C communication.

  * The time from when the reset is released until 00h can be received is shown in the AC characteristics.

* When the boot firmware receives a generic code after sending an ACK, it sends a "Boot code". If a code other than the Generic code is received, it will wait to receive the Generic code again.
* 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 from USB, the count value will be reset.

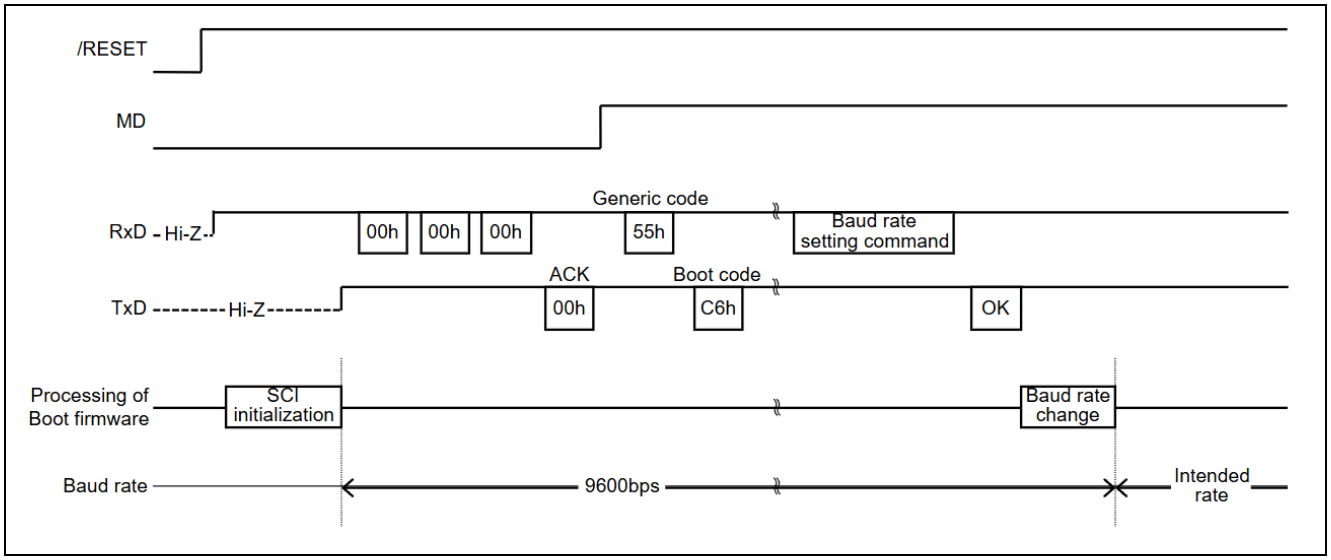


Figure 10. 2-wire UART Communication

* 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 of data (Generic code) from the host.
4. Send C6h data (Boot code) from boot firmware.

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

4.4.3 Settings of the I2C Communication

When the device operating mode is serial programming mode, the boot firmware initializes I2C and waits for reception. By receiving 00h, it is determined that I2C communication is selected as the communication method.

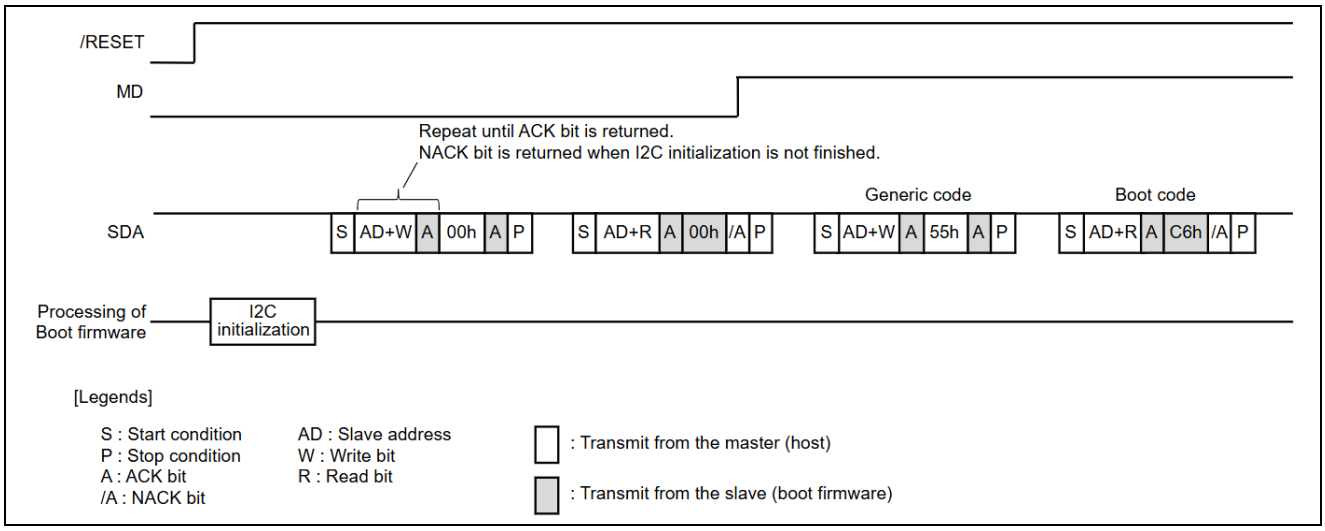


Figure 11. Settings of the I2C Communication

By performing the following procedure, communication establishment is completed, and the process moves to the "Command acceptable phase".

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

* If the ACK bit is not returned at 00h transmission, or even though the ACK bit is returned at 00h transmission but 00h data (ACK) is not returned from the boot firmware, check the communication environment and try again from reset release.
* Follow the transfer rate that can be communicated, refer to "3. Communication Method".

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. (Refer to the explanation of each command for details).

- When a command is normally finished, the boot firmware stays in 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

5.1.2 Command Packet

The host sends data of 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 For example: For Write command: Start/End address For example: 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.

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

5.1.3 Data Packet

The 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 For example: For Write data transmission: Write data For example: 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.

*1: If the host sends a packet whose length is 0 bytes or over 1025 bytes, the microcontroller returns a packet with an 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.

5.1.4 CMD: Command Code

Table 10. Command Codes

Value	Name	Description
30h	Authentication command	Authenticate ID for connection with the device
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 the target area.
13h	Write command	Write data to the target area.
15h	Boundary setting command	Set the boundary.
18h	CRC command	Cyclic Redundancy Check of the 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 the 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.
DAh	Protection error	(*1) Accessing protected areas or performing prohibited actions.
DDh	ID discord error	(*1), ID authentication failed.
DEh	Serial programming disable error	(*1) If serial programming is disabled. (stored ID [127] = 0)
E5h	Flash access error	(*1), (*2), (*3) Abnormality from the Flash sequencer.

*1: When this error occurs, the 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.
*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
FFFF0000h | FSTATR2 [15:0]	Flash status	When a Flash access error occurs, boot firmware returns the value of the FSTATR2 register. When not, boot firmware returns FFFFFFFFh. Boot firmware clears the FSTATR2 register after the status sending, so even when error(s) occur, the host can retry the next command without a reset release. The upper 16-bit is always FFFFh, the 32-bit format is for compatibility with other RA series boot firmware.

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.

6. Command List

Table 15. Command List

Name	Communication Method	Phase			Prerequisite Command
		Initialization	Communication setting	Command Acceptable	
Authentication Command	2-wire UART, I2C	NG	NG	OK	-
Inquiry Command	2-wire UART, I2C	NG	NG	OK	➢ (*1)
Signature request command	2-wire UART, I2C	NG	NG	OK	-
Area information request command	2-wire UART, I2C	NG	NG	OK	-
Baudrate setting command	2-wire UART, I2C	NG	NG	OK	-
Erase command	2-wire UART, I2C	`	NG	OK	➢ (*1)
Write command	2-wire UART, I2C	NG	NG	OK	➢ (*1)
Read command	2-wire UART, I2C	NG	NG	OK	➢ (*1)
CRC command	2-wire UART, I2C	NG	NG	OK	-

*1): [When the ID code is not stored in the device (stored ID code is all "1")]
There are no other commands that have to be executed before executing this command.
*2): [When the ID code is stored in the device (stored ID code is not all "1")]
The authentication command has to be executed before executing this command.

*3): Communication method: That command is available in the communication. (Even if an unavailable command is sent, boot firmware doesn't return an error.)

*4) NG: That command is not available in the phase. (Boot firmware doesn't return any data packet.)

6.1 Authentication Command

This command compares the ID-code stored in the microcontroller with the ID-code received from the flash programmer and sends the result to the flash programmer. If the received ID code is "ALeRASE (*1)", all areas of the flash memory are erased without ID authentication. This command cannot be executed if the ID code stored in the microcontroller is all "1". Also, this command cannot be executed after successful authentication until reset if the ID-code stored in the microcontroller is NOT all "1".
This command requires adherence to conditions described in the Command List.

*1: "ALeRASE" = 41h, 4Ch, 65h, 52h, 41h, 53h, 45h, FFh, FFh, FFh, FFh, FFh, FFh, FFh, FFh, FFh.

Condition	Processing	
stored ID [127:0] = All "1"	Non-secure device	-> Return to command acceptance. (Other commands become executable)
stored ID [127] = 0b	Serial programming disable	-> Serial programming disable error (Enter an infinite loop)

stored ID [127:126] = 10b		ID authentication	When ID mismatch -> ID discord error (Enter an infinite loop) When ID matches -> Return to command acceptance (Other commands become executable)
stored ID [127:126] = 11b	received ID !="ALeRASE"		
	received ID = "ALeRASE"	All area erasure	When FSPR = 0 -> Protection error (Enter an infinite loop) When an Error occurred -> Flash access error (Enter an infinite loop) When successful erasure -> Return to command acceptance (Other commands become executable)

6.1.1 Authentication Command Sequence Diagram

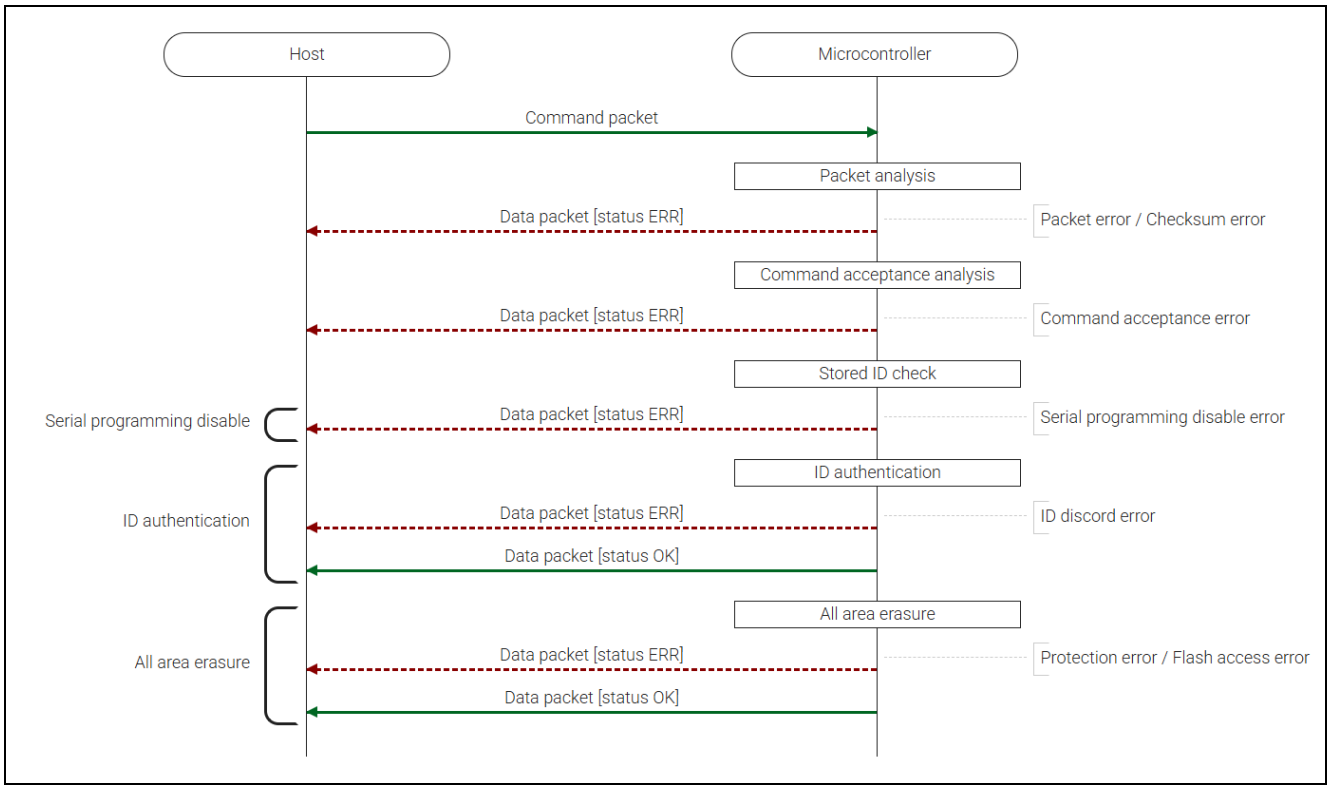


Figure 12. Authentication Command Sequence Diagram

6.1.2 Packets

6.1.2.1 Command Packet

<table>
<tr><td>SOH</td><td>(1 byte)</td><td>01h</td></tr>
<tr><td>LNH</td><td>(1 byte)</td><td>00h</td></tr>
<tr><td>LNL</td><td>(1 byte)</td><td>11h</td></tr>
<tr><td>CMD</td><td>(1 byte)</td><td>30h (Authentication command)</td></tr>
<tr><td>IDC</td><td>(16 byte)</td><td>ID code
For example, if the stored ID is as follows, the Host should send IDC in the order shown in the lower table.

Stored ID:
<table>
<tr><td colspan="4">ID [127:96]</td><td colspan="4">ID [95:64]</td></tr>
<tr><td>F0</td><td>F1</td><td>F2</td><td>F3</td><td>E4</td><td>E5</td><td>E6</td><td>E7</td></tr>
<tr><td colspan="4">ID [63:32]</td><td colspan="4">ID [31:0]</td></tr>
<tr><td>D8</td><td>D9</td><td>DA</td><td>DB</td><td>CC</td><td>CD</td><td>CE</td><td>CF</td></tr>
</table>

Order of sending IDC for ID authentication:
<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>F0</td><td>F1</td><td>F2</td><td>F3</td><td>E4</td><td>E5</td><td>E6</td><td>E7</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>D8</td><td>D9</td><td>DA</td><td>DB</td><td>CC</td><td>CD</td><td>CE</td><td>CF</td></tr>
</table>

For example: For all area erasure, the Host should send "ALeRASE" as IDC.

Order of sending IDC for all area erasure:
<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>41</td><td>4C</td><td>65</td><td>52</td><td>41</td><td>53</td><td>45</td><td>FF</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>FF</td><td>FF</td><td>FF</td><td>FF</td><td>FF</td><td>FF</td><td>FF</td><td>FF</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>

6.1.2.2 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 byte)	FFFFFFFFh (unused code)
ADR	(4 byte)	FFFFFFFFh (unused code)
SUM	(1 byte)	CEh
ETX	(1 byte)	03h

6.1.2.3 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

6.1.3 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:

- No setting OCD/Serial ID, or already authenticated, 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.

Boot firmware checks for "Serial programming disable error" after the processing above.

- When bit 127 of the ID code stored in the device is 0, the boot firmware sends a "Serial programming disable error" and finishes command processing.
  Boot firmware does not accept any commands or responses for the commands after sending this error.
  * Memory contents do not change before command reception.

When no error occurs, the boot firmware decides whether to execute ID code authentication.

- Boot firmware executes ID code authentication when at least one of the following conditions is met.
  - Bit 126 of the ID code stored in the device is 0b.
  - Bit 126 of the ID code stored in the device is 1b, but the received IDC is not "ALeRASE".
- When ID code authentication fails, the boot firmware sends "ID discord error" and finishes command processing.
  Boot firmware does not accept any commands or responses for the commands after sending this error.
  * Memory contents do not change before command reception.
- When ID code authentication passes, the boot firmware sends "OK" and finishes command processing.
  Then boot firmware transitions to the Command acceptable phase and waits for the next command.
  * Memory contents do not change before command reception.

When no error occurs and also ID code authentication is not executed, the boot firmware executes a full erasure of flash memory.

- When FSPR in the Config area is set, e.g., the value is 0, the boot firmware sends "Protection error" and does not execute all erasure but finishes command processing.
  Boot firmware does not accept any commands or responses for the commands after sending this error.
  * Memory contents do not change before command reception.
- If the above error does not occur, the boot firmware executes all erasure.
- When all erasure fails, boot firmware receives a data packet and returns "Erase error", "Write error" or "Sequencer error" and finishes command processing.
  Boot firmware does not accept any commands or responses for the commands after sending this error.
  * Refer to the Status code for the conditions of the errors.
  * Memory contents are undefined.
- When all erasure passes, the boot firmware sends "OK" and finishes command processing.
  Then boot firmware transitions to the Command acceptable phase and waits for the next command.*
  Memory contents do not change before command reception.
  * All memory areas are in an erased state.

6.1.4 Status Information from the Microcontroller

(Listed in descending order of priority.)

Condition	STS	ST2	ADR
The received packet does not have ETX.	Packet error	FFFFFFFFh	FFFFFFFFh
Sum data in the packet received is different from the value calculated by the boot firmware.	Checksum error	FFFFFFFFh	FFFFFFFFh
Packet length in the received packet does not comply with the packet format.	Packet error	FFFFFFFFh	FFFFFFFFh
Packet length in the received packet does not comply with the specifications of this command.	Packet error	FFFFFFFFh	FFFFFFFFh
OCD/Serial ID is not stored in the device.	Command acceptance error	FFFFFFFFh	FFFFFFFFh
OCD/Serial ID is stored in the device, but this command has already been successfully completed after device reset.	Command acceptance error	FFFFFFFFh	FFFFFFFFh
If the Highest-order bit of the stored ID is "0".	Serial programming disable error	FFFFFFFFh	FFFFFFFFh
The ID authentication fails.	ID discord error	FFFFFFFFh	FFFFFFFFh
When the received ID is "ALeRASE", the stored ID [127:126] = 11b, and also FSPR is set (FSPR=0).	Protection error	FFFFFFFFh	FFFFFFFFh
FCB detected an error after the command execution.	Flash access error	Flash status	Failure address
Successful completion.	OK	FFFFFFFFh	FFFFFFFFh

6.2 Inquiry Command

This command is used to check if boot firmware is "Command acceptable phase" or not.
This command require adherence to conditions described in Command List.

6.2.1 Inquiry Command Sequence Diagram

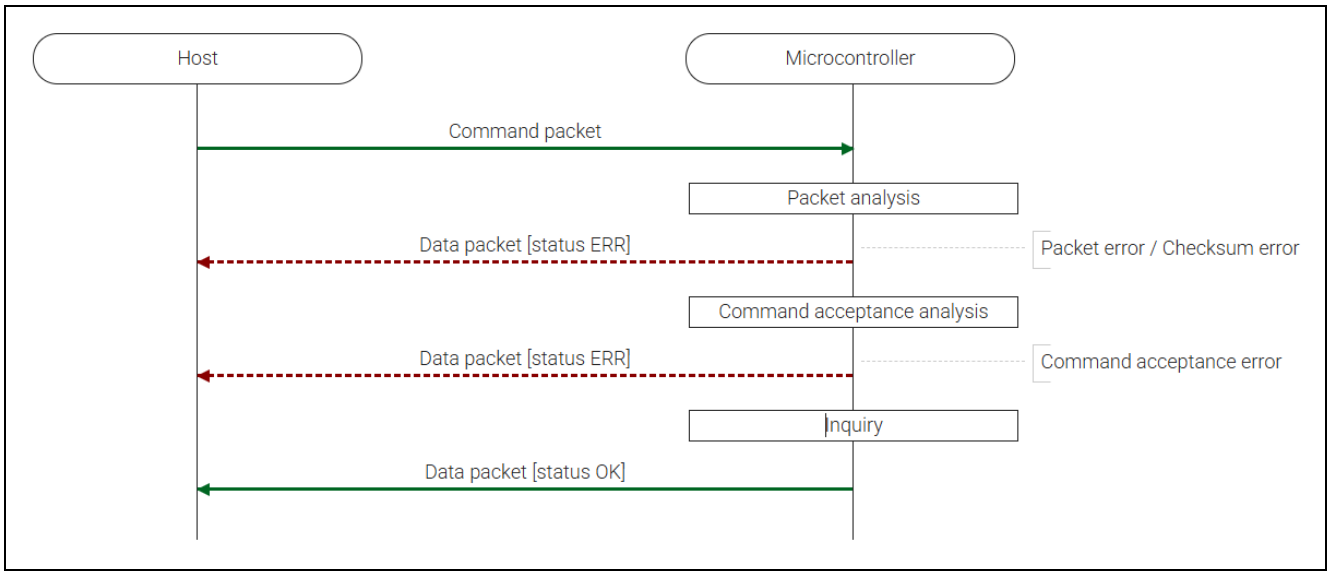


Figure 13. Inquiry Command Sequence Diagram

6.2.2 Packets

6.2.2.1 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

6.2.2.2 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

6.2.2.3 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

6.2.3 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, the boot firmware executes the acceptance analysis.

- If OCD/Serial ID is set and not authenticated, 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 executes the inquiry processing.

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

6.2.4 Status Information from the Microcontroller

(Listed in descending order of priority.)

Condition	STS	ST2	ADR
The received packet does not have ETX.	Packet error	FFFFFFFFh	FFFFFFFFh
Sum data in the packet received is different from the value calculated by the boot firmware.	Checksum error	FFFFFFFFh	FFFFFFFFh
Packet length in the received packet does not comply with the packet format.	Packet error	FFFFFFFFh	FFFFFFFFh
Packet length in the received packet does not comply with the specifications of this command.	Packet error	FFFFFFFFh	FFFFFFFFh
OCD/Serial ID is set, and ID authentication by the Authentication command has not been performed.	Command acceptance error	FFFFFFFFh	FFFFFFFFh
The process has ended normally.	OK	FFFFFFFFh	FFFFFFFFh

6.3 Signature Request Command

This command sends the information of the device signature to the host and requires adherence to the conditions described in the Command List.

6.3.1 Signature Request Command Sequence Diagram

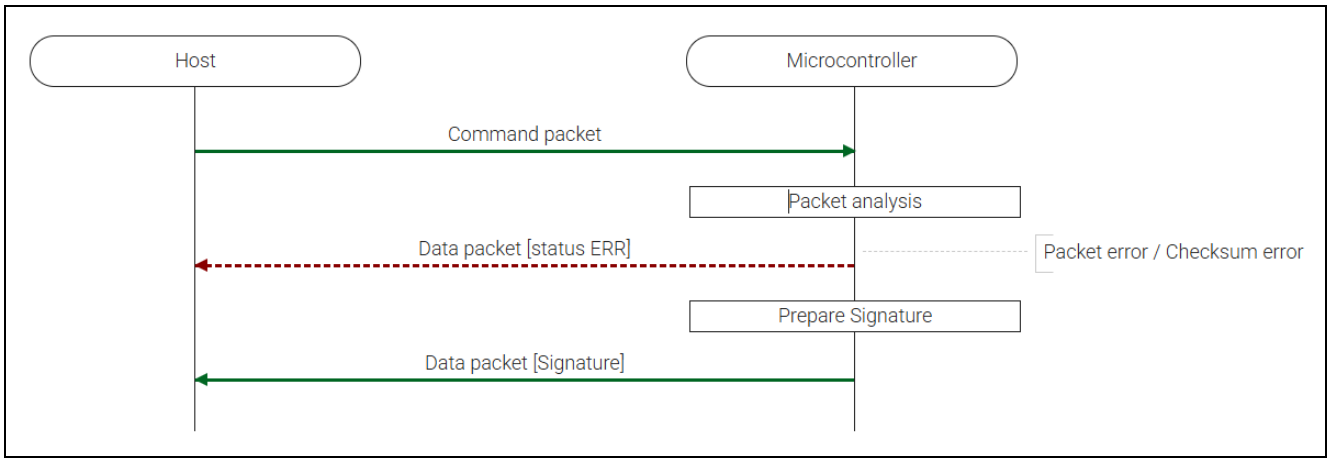


Figure 14. Signature Request Command Sequence Diagram

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)	3Ah (Signature request command)
SUM	(1 byte)	C5h
ETX	(1 byte)	03h

6.3.2.2 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 e.g.) 6Mbps (6000000bps) -> 00h, 5Bh, 8Dh, 80h. *The returned value changes depending on the current operating VCC.
NOA	(1 byte)	Number of accessible areas e.g.) If the device has four areas. -> 04h
TYP	(1 byte)	Type code (features and functions of the device) 0Ah: RA2L2 Group
BFV	(3 bytes)	Boot firmware version. *Order of sending: Major version -> Minor version -> Build For example: v2.4.16 -> 02h, 04h, 10h
DID	(16 bytes)	Device ID 16-byte ID code (unique ID) for identifying the individual MCU.
PTN	(16 bytes)	Product type name Values of PNRn (n=0-3) registers are returned. The order of sending is as follows: PNR3 [31:24], PNR3 [23:16], ... PNR0 [15:8], PNR0 [7:0]
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)	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

6.3.3 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, the boot firmware returns the signature.

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

6.3.4 Status Information from the Microcontroller

(Listed in descending order of priority.)

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

6.4 Area Information Request Command

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

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

6.4.1 Area Information Request Sequence Diagram

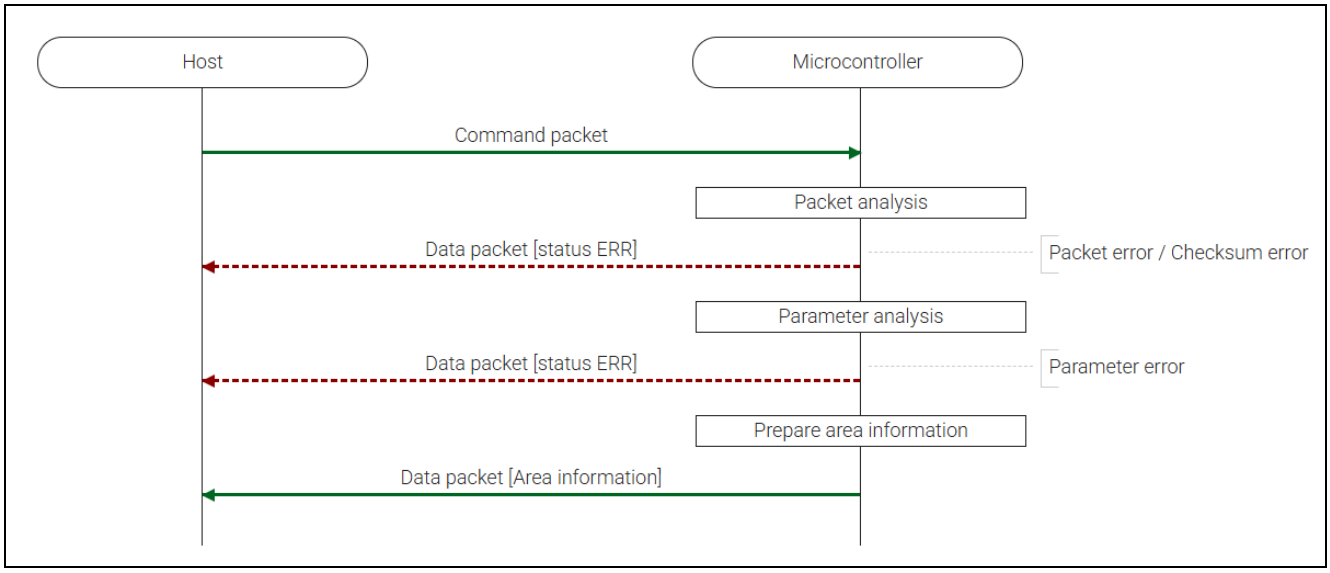


Figure 15. Area Information Request Command Sequence Diagram

6.4.2 Packets

6.4.2.1 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

6.4.2.2 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
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, the target command isn't available for the area.
*2: N = 0–F

6.4.2.3 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

6.4.2.4 Processing Procedure

Boot firmware receives and analyzes a command packet:

- The boot firmware recognizes the start of the command packet by receiving SOH.
  If the boot firmware receives something other than SOH, it 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, the boot firmware analyzes the command parameters.

- If the specified NUM is "NOA" returned by "signature requests 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 contents do 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 contents do not change before command reception.

6.4.3 Status Information from the Microcontroller

(Listed in descending order of priority.)

Condition	STS	ST2	ADR
The received packet does not have ETX.	Packet error	FFFFFFFFh	FFFFFFFFh
Sum data in the packet received is different from the value calculated by the boot firmware.	Checksum error	FFFFFFFFh	FFFFFFFFh
Packet length in the received packet does not comply with the packet format.	Packet error	FFFFFFFFh	FFFFFFFFh
Packet length in the received packet does not comply with the specifications of this command.	Packet error	FFFFFFFFh	FFFFFFFFh
If the Area number in the received packet is a nonexistent area number.	Packet error	FFFFFFFFh	FFFFFFFFh

6.4.4 Specific Value of Area Information

(For RA2L2)

NUM	Area	KOA	SAD	EAD	EAU	WAU	RAU	CAU
0	User area	00h	00000000h	0001FFFFh (*3)	2 KB	4B	1B	32 KB
1	Data area	10h	40100000h	40100FFFh (*3)	1 KB	1B	1B	1 KB
2	Config area	20h	01010010h	01010033h	0 (*1)	4B	1B	1B (*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, only possible to specify the area's SAD/EAD as CRC command's SAD/EAD. See CRC command for details.

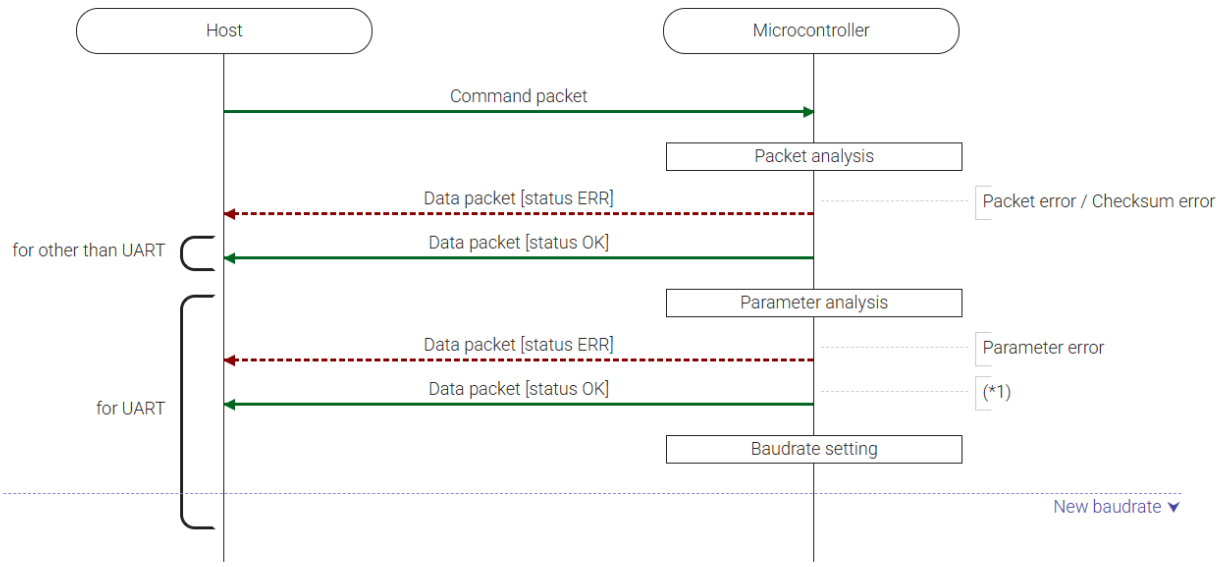
*3: These addresses can change depending on the ROM size of each product.

6.5 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 the Command List.

6.5.1 Baudrate Setting Command Sequence Diagram



*1: After receiving this packet, the next command packet shall wait for tBRT to keep the baudrate setting time.

Figure 16. Baudrate 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)	05h
CMD	(1 byte)	34h (Baudrate setting command)
KADR	(4 bytes)	UART Baudrate [bps] You can set one of the following values.
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

6.5.2.2 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

6.5.2.3 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

6.5.3 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 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.

6.5.4 Status Information from the Microcontroller

(Listed in descending order of priority.)

Condition	STS	ST2	ADR
The received packet does not have ETX.	Packet error	FFFFFFFFh	FFFFFFFFh
Sum data in the packet received is different from the value calculated by the boot firmware.	Checksum error	FFFFFFFFh	FFFFFFFFh
Packet length in the received packet does not comply with the packet format.	Packet error	FFFFFFFFh	FFFFFFFFh
Packet length in the received packet does not comply with the specifications of this command.	Packet error	FFFFFFFFh	FFFFFFFFh
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 16. Baudrate Setting Values

When VCC is 1.8 V or lower

Intended Baudrate	ABCS	CKS [1:0]	BRR [7:0]	MDDR [7:0]	Accuracy
9600bps	0	00b	07h	9Dh	-0.2%
115200bps	0	00b	00h	ECh	+0.1%
Other	unavailable	unavailable	unavailable	unavailable	-

Intended Baudrate	ABCS	CKS [1:0]	BRR [7:0]	MDDR [7:0]	Accuracy
9600bps	0	00b	67h	(Disabled)	+0.2%
115200bps	0	00b	07h	ECh	+0.1%
500Kbps	0	00b	01h	(Disabled)	±0.0%
1.0Mbps	0	00b	00h	(Disabled)	±0.0%
1.5Mbps	1	00b	00h	C0h	±0.0%
2.0Mbps	1	00b	00h	(Disabled)	±0.0%
Other	unavailable	unavailable	unavailable	unavailable	-

6.6 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.

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

6.6.1 Erase Command Sequence Diagram

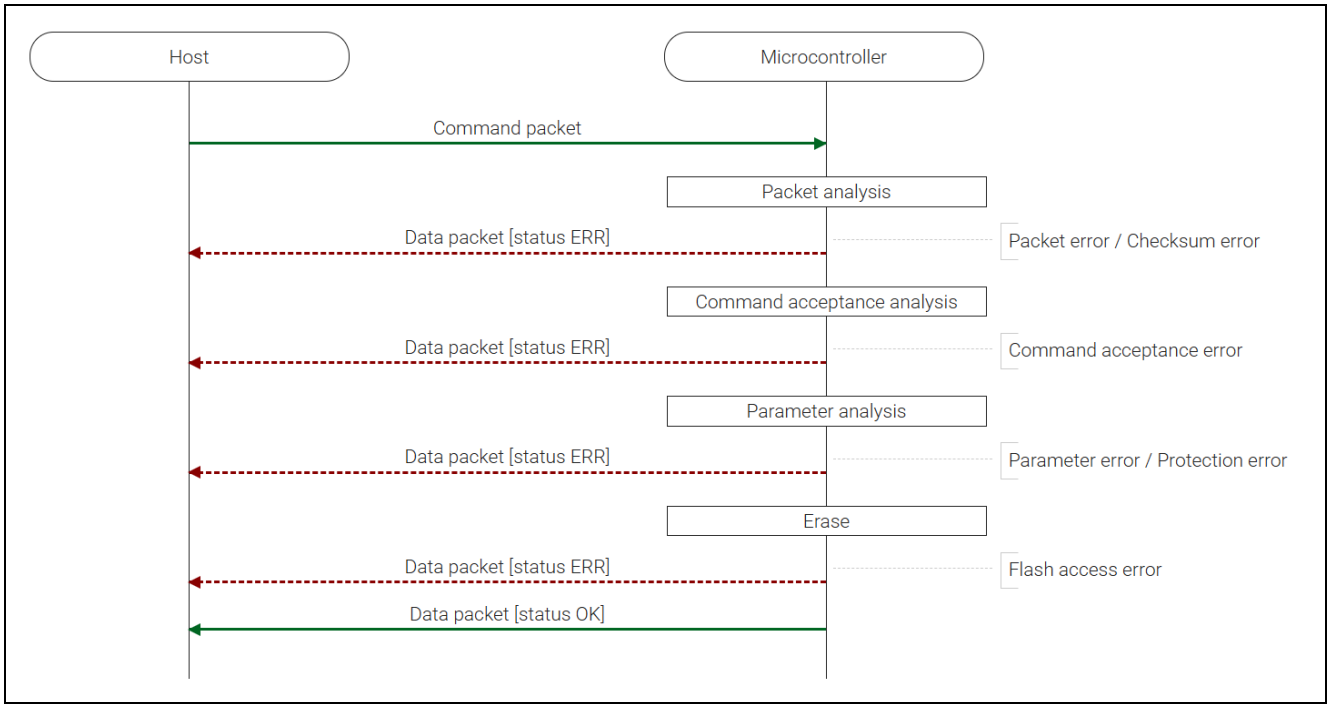


Figure 17. Erase Command Sequence Diagram

6.6.2 Packets

6.6.2.1 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

6.6.2.2 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

6.6.2.3 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

6.6.3 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 status does not change before command reception.

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

- If OCD/Serial ID is set and not authenticated, 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, the 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".
- When the designated erasure range includes an area outside of the Access window, "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, the 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.
- When the erase processing is normally finished, boot firmware returns "OK" and waits for the next command.
  * Specified areas in memory are in an erased 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 packet received is different from the value calculated by the boot firmware.	Checksum error	FFFFFFFFh	FFFFFFFFh
Packet length in the received packet does not comply with the packet format.	Packet error	FFFFFFFFh	FFFFFFFFh
Packet length in the received packet does not comply with the specifications of this command.	Packet error	FFFFFFFFh	FFFFFFFFh
OCD/Serial ID is set, and ID authentication by the Authentication command has not been performed.	Command acceptance error	FFFFFFFFh	FFFFFFFFh
Start address is bigger than the End address.	Parameter error	FFFFFFFFh	FFFFFFFFh
Start address or End address is outside the scope of the user area specified in the area information.	Parameter error	FFFFFFFFh	FFFFFFFFh
Start address and the End address belong to different kinds of areas.	Parameter error	FFFFFFFFh	FFFFFFFFh
The access unit "EAU" of the specified area is 0.	Parameter error	FFFFFFFFh	FFFFFFFFh
Start address or the End address does not comply with the EAU of the area.	Parameter error	FFFFFFFFh	FFFFFFFFh
Designated erasing range includes outside of the access window.	Protection error	FFFFFFFFh	FFFFFFFFh
FCB detected an error after the command execution.	Flash access error	Flash status	Failure address
Successful completion.	OK	FFFFFFFFh	FFFFFFFFh

6.7 Write Command

This command receives data from the host and writes that 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.

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

6.7.1 Write Command Sequence Diagram

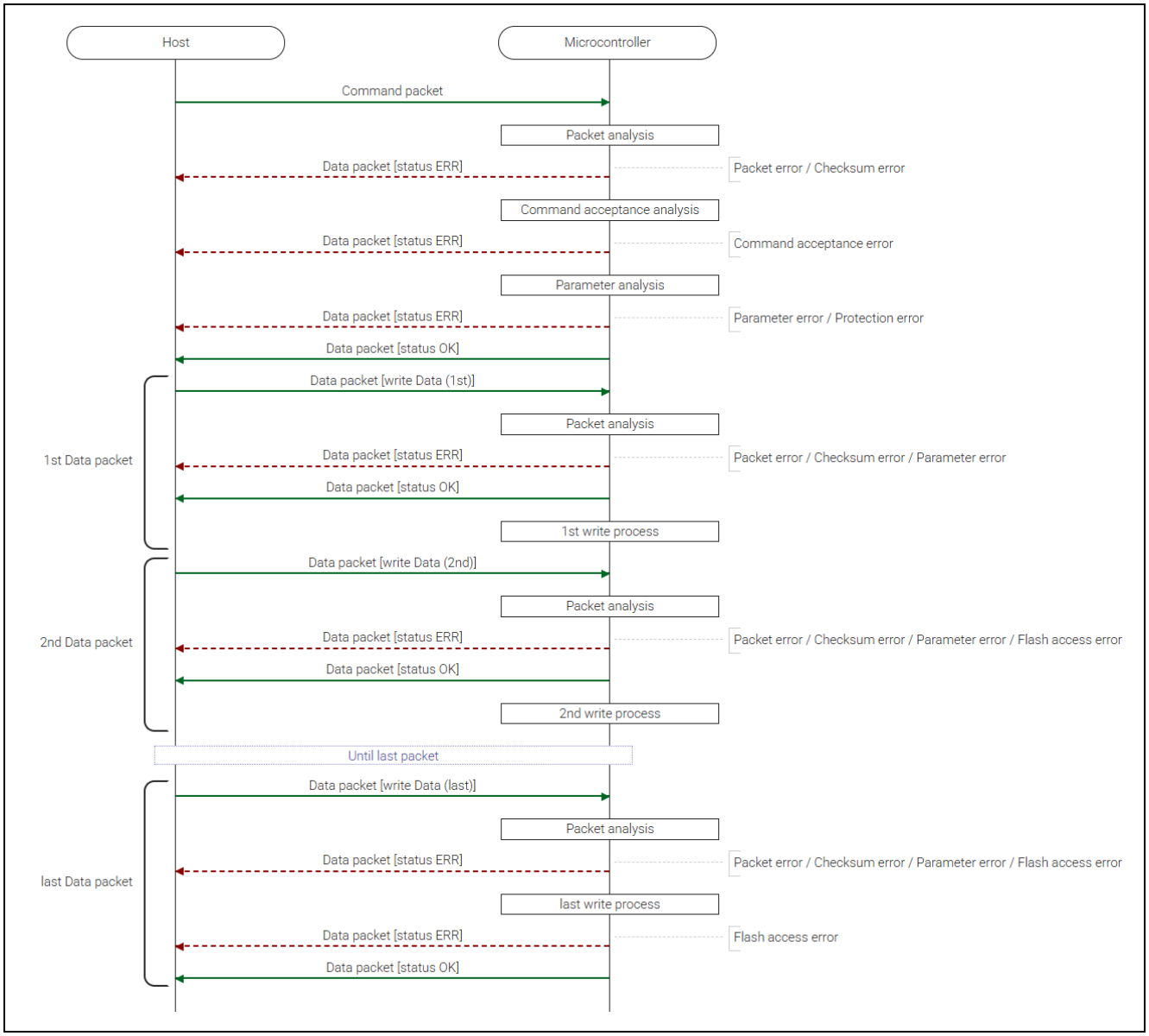


Figure 18. Write Command Sequence Diagram

6.7.2 Packets

6.7.2.1 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

6.7.2.2 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

6.7.2.3 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

6.7.2.4 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

6.7.3 Processing Procedure

Boot firmware receives and analyzes a command packet:

- The boot firmware recognizes the start of the command packet by receiving SOH.
  If the boot firmware receives something other than SOH, it 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 executes the acceptance analysis:

- If OCD/Serial ID is set and not authenticated, 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, the 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".
- When the designated writing range includes an area outside of the Access window, "Protection error" is returned.
- When the designated writing range is protected by FSPR and FSPR is set (=0), "Protection error" is returned.
- When any of the above errors occur, 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, the 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 the 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, the 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 the failure address (ADR) of the memory area is undefined.
- When the write processing is normally finished, the boot firmware receives the next data packet.

When the received data packet is the last write data, the 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 the failure address (ADR) of the memory area is undefined.
- When the write processing is normally finished, boot firmware returns "OK" and waits for the next command.
  * Sent data is written to the specified area in memory.

6.7.4 Status Information from the Microcontroller

(Listed in descending order of priority.)

Condition	STS	ST2	ADR
The received packet does not have ETX.	Packet error	FFFFFFFFh	FFFFFFFFh
Sum data in the packet received is different from the value calculated by the boot firmware.	Checksum error	FFFFFFFFh	FFFFFFFFh
Packet length in the received packet does not comply with the packet format.	Packet error	FFFFFFFFh	FFFFFFFFh
Packet length in the received packet does not comply with the specifications of this command.	Packet error	FFFFFFFFh	FFFFFFFFh
OCD/Serial ID is set, and ID authentication by Authentication command has not been performed.	Command acceptance error	FFFFFFFFh	FFFFFFFFh
Start address is bigger than the End address.	Parameter error	FFFFFFFFh	FFFFFFFFh
Start address or End address is outside the scope of the accessible area specified in the area information.	Parameter error	FFFFFFFFh	FFFFFFFFh
Start address and the End address belong to different kinds of areas.	Parameter error	FFFFFFFFh	FFFFFFFFh
The access unit "WAU" of the specified area is 0.	Parameter error	FFFFFFFFh	FFFFFFFFh
Start address or the End address does not comply with WAU of the area.	Parameter error	FFFFFFFFh	FFFFFFFFh
Designated writing range includes outside of the access window.	Protection error	FFFFFFFFh	FFFFFFFFh
Designated writing range includes the FSPR protected area, and also the FSPR bit is set (FSPR=0).	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 packets exceeds the specified end address.	Parameter error	FFFFFFFFh	FFFFFFFFh
The data size of the data packet does not comply with the writing unit of the area.	Parameter error	FFFFFFFFh	FFFFFFFFh
FCB detected an error after the command execution.	Flash access error	Flash status	Failure address
Successful completion.	OK	FFFFFFFFh	FFFFFFFFh

6.7.5 Precautions

(1) The following cannot be executed when 0b is written to FSPR:

- Writing to access the start/end address setting window.
- All erasure by the Authentication command with ALeRASE ID.
- All erasure by debug mode.

FSPR cannot be written back to 1 once after 0 is written.

(2) When data that is not 0xFFFF is written to SECMPUAC, the boot firmware does not operate but goes into an infinite loop after the device reset.

(3) The following cannot be executed when 0b is written to ID [127] in the Config area:

- ID authentication with the Authentication command.
- All area erasure with Authentication command ("ALeRASE" ID).
- All area erasure in the debug mode (all erasure with a debugger).

(4) The following cannot be executed when 0b is written to ID [126] in the Config area:

- All areas erasure with Authentication command ("ALeRASE" ID).
- All areas erasure in the debug mode (all erasure with a debugger).

6.8 Read Command

This command reads data from a specified area and sends data to the 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 the conditions described in the Command List.

6.8.1 Sequence Diagram

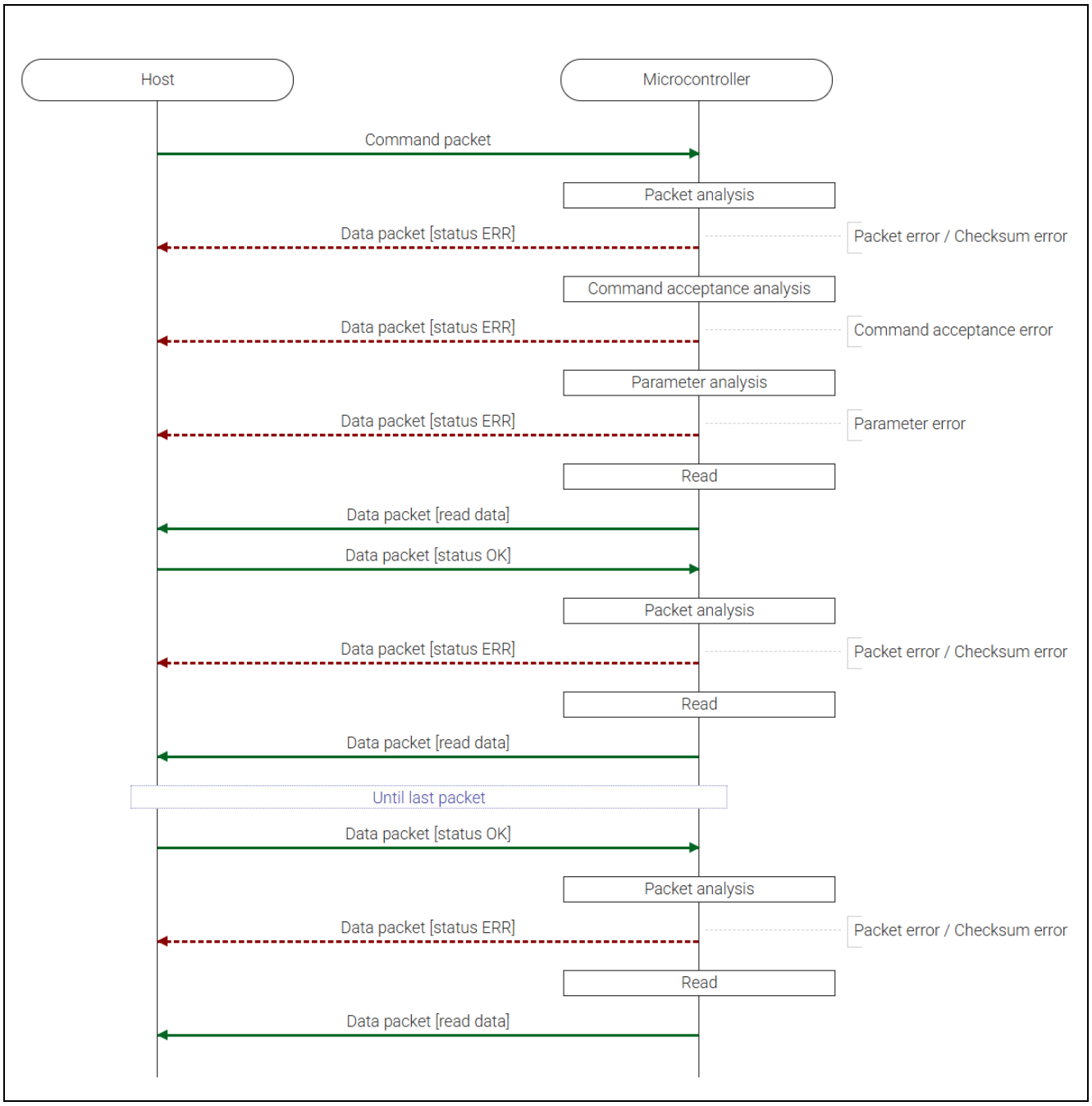


Figure 19. Read Command Sequence Diagram

6.8.2 Packets

6.8.2.1 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

6.8.2.2 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

6.8.2.3 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

6.8.2.4 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

6.8.3 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 status does not change before command reception.

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

- If OCD/Serial ID is set and not authenticated, 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, the boot firmware will send a "Parameter error".
- If the RAU for the specified area is 0, 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.

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 data other 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 the SUM in the received data packet is different from the value calculated by the 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 the defined values, "Packet error" is returned.
- When LNH and LNL in the received data packet do not comply with the format 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 status does not change before command reception.
- If the above errors do not occur, the boot firmware continues to read and send data.

6.8.3.1 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 packet received is different from the value calculated by the boot firmware.	Checksum error	FFFFFFFFh	FFFFFFFFh
Packet length in the received packet does not comply with the packet format.	Packet error	FFFFFFFFh	FFFFFFFFh
Packet length in the received packet does not comply with the specifications of this command.	Packet error	FFFFFFFFh	FFFFFFFFh
OCD/Serial ID is set, and ID authentication by the Authentication command has not been performed.	Command acceptance error	FFFFFFFFh	FFFFFFFFh
Start address is bigger than the End address.	Parameter error	FFFFFFFFh	FFFFFFFFh
Start address or End address is outside the scope of the accessible area specified in the area information.	Parameter error	FFFFFFFFh	FFFFFFFFh
Start address and the End address belong to different Kinds of areas.	Parameter error	FFFFFFFFh	FFFFFFFFh
The access unit "RAU" of the specified area is 0.	Parameter error	FFFFFFFFh	FFFFFFFFh
Start address or the End address does not comply with the RAU of the area.	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

6.9 CRC Command

This command calculates CRC data from a specified area and sends it to the 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. As an exception, when specifying the Config area, it can specify only the entire Config area regardless of the CRC access unit returned by the Area information command.

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

Boot firmware uses the following CRC method:

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

6.9.1 CRC Command Sequence Diagram

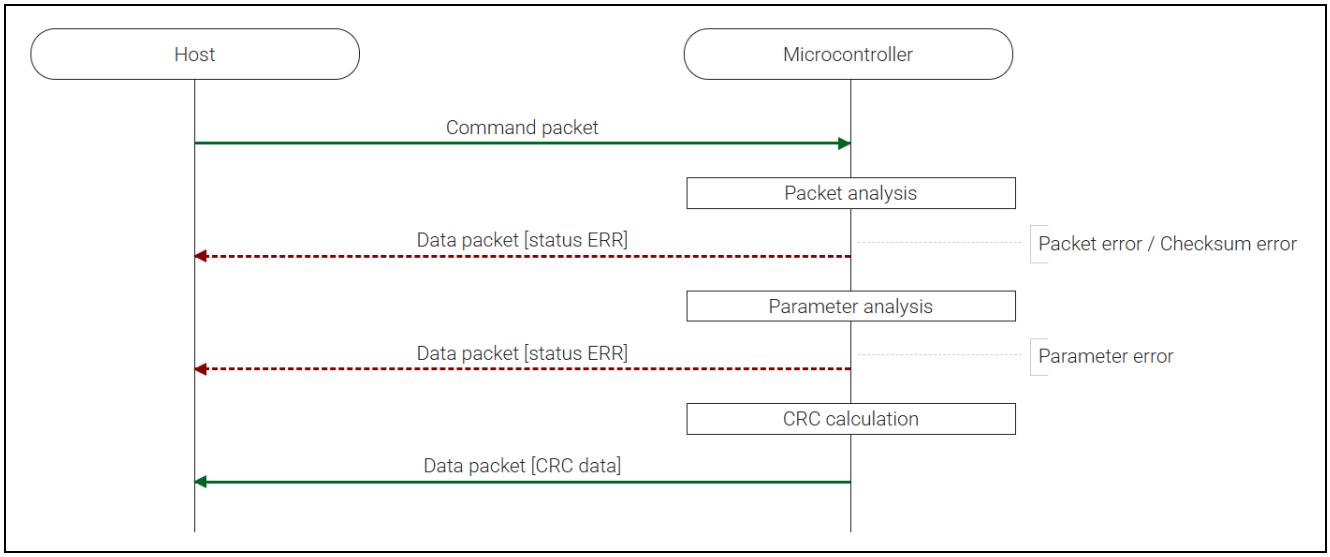


Figure 20. CRC Command Sequence Diagram

6.9.2 Packets

6.9.2.1 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

6.9.2.2 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

6.9.2.3 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

6.9.3 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 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, the 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 occur, 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 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.

6.9.4 Status Information from the Microcontroller

(Listed in descending order of priority.)

Condition	STS	ST2	ADR
The received packet does not have ETX.	Packet error	FFFFFFFFh	FFFFFFFFh
Sum data in the packet received is different from the value calculated by the boot firmware.	Checksum error	FFFFFFFFh	FFFFFFFFh
Packet length in the received packet does not comply with the packet format.	Packet error	FFFFFFFFh	FFFFFFFFh
Packet length in the received packet does not comply with the specifications of this command.	Packet error	FFFFFFFFh	FFFFFFFFh
Start address is bigger than the End address.	Parameter error	FFFFFFFFh	FFFFFFFFh
Start address or End address is outside the scope of the accessible area specified in the area information.	Parameter error	FFFFFFFFh	FFFFFFFFh
Start address and the End address belong to different Kinds of areas.	Parameter error	FFFFFFFFh	FFFFFFFFh
The access unit "CAU" of the specified area is 0.	Parameter error	FFFFFFFFh	FFFFFFFFh
Start address or the End address does not comply with the CAU of the area.	Parameter error	FFFFFFFFh	FFFFFFFFh
The specified area is the Config area, and the Start address and End address do not specify the entire Config area.	Parameter error	FFFFFFFFh	FFFFFFFFh

7. Flow Examples

7.1 Beginning Communication

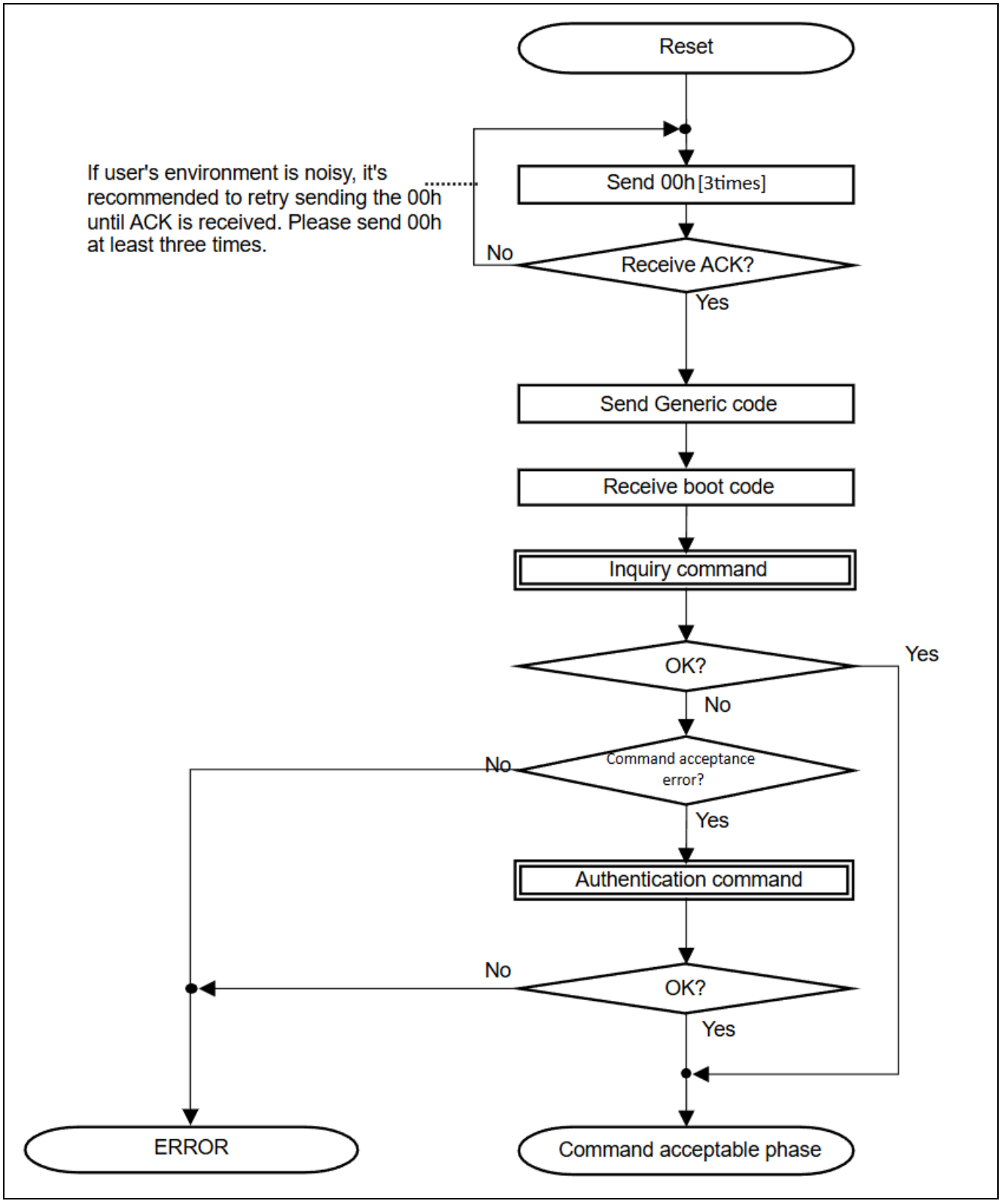


Figure 21. Beginning Communication

7.2 Acquisition of Device Information / Baudrate Settings

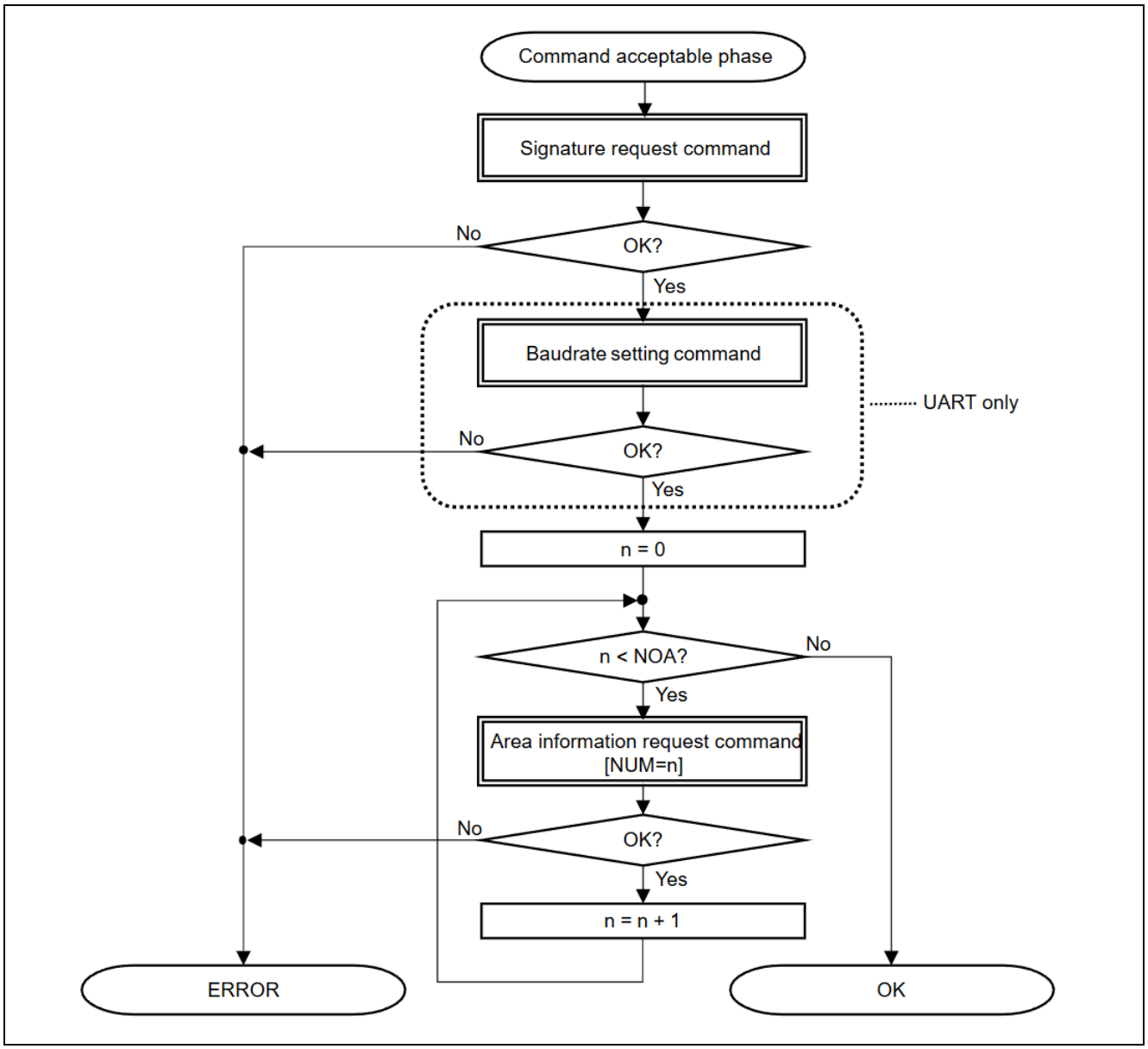


Figure 22. Acquisition of Device Information / Baudrate Settings

7.3 ID Authentication

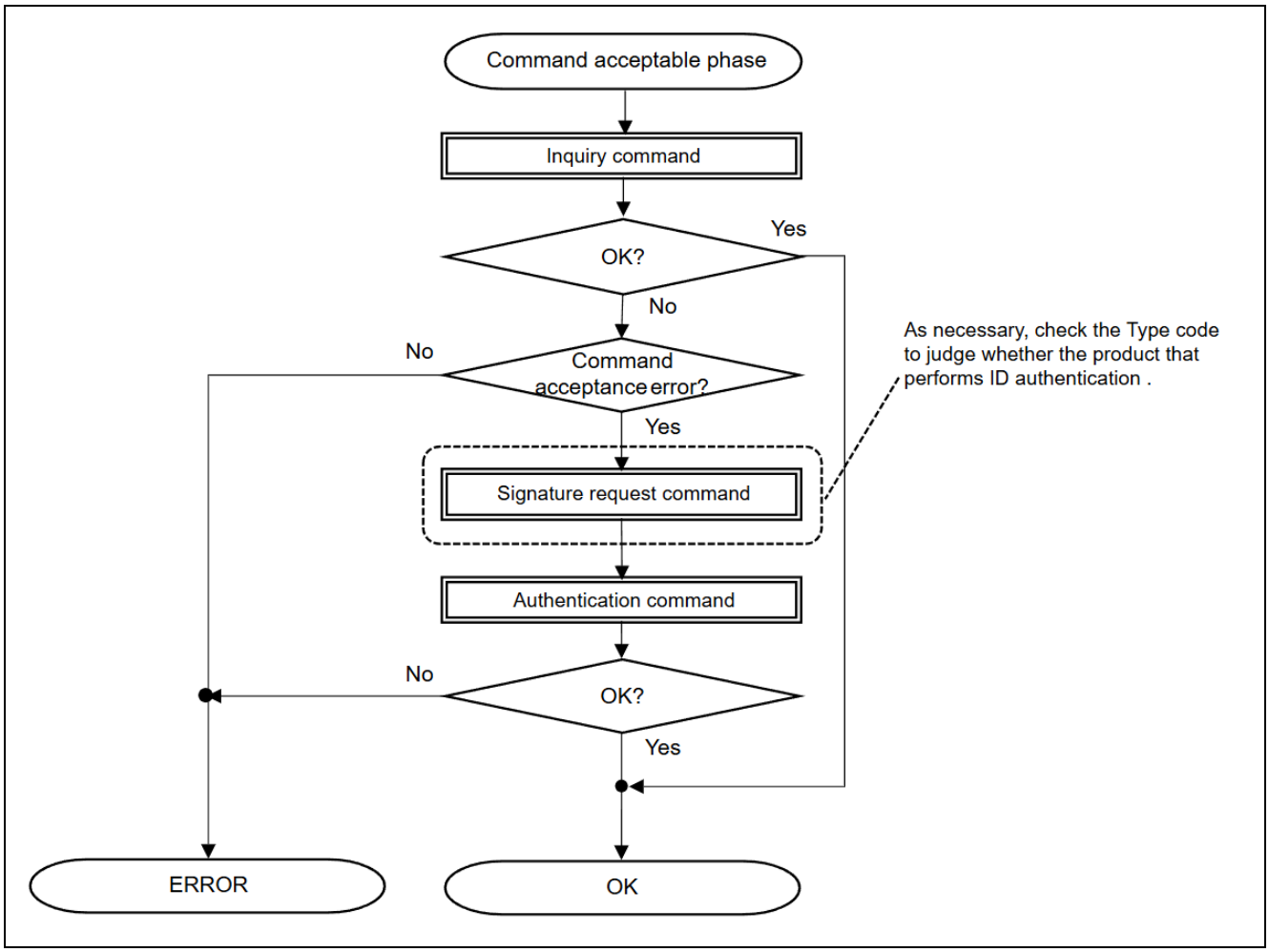


Figure 23. ID Authentication

7.4 Data Programming

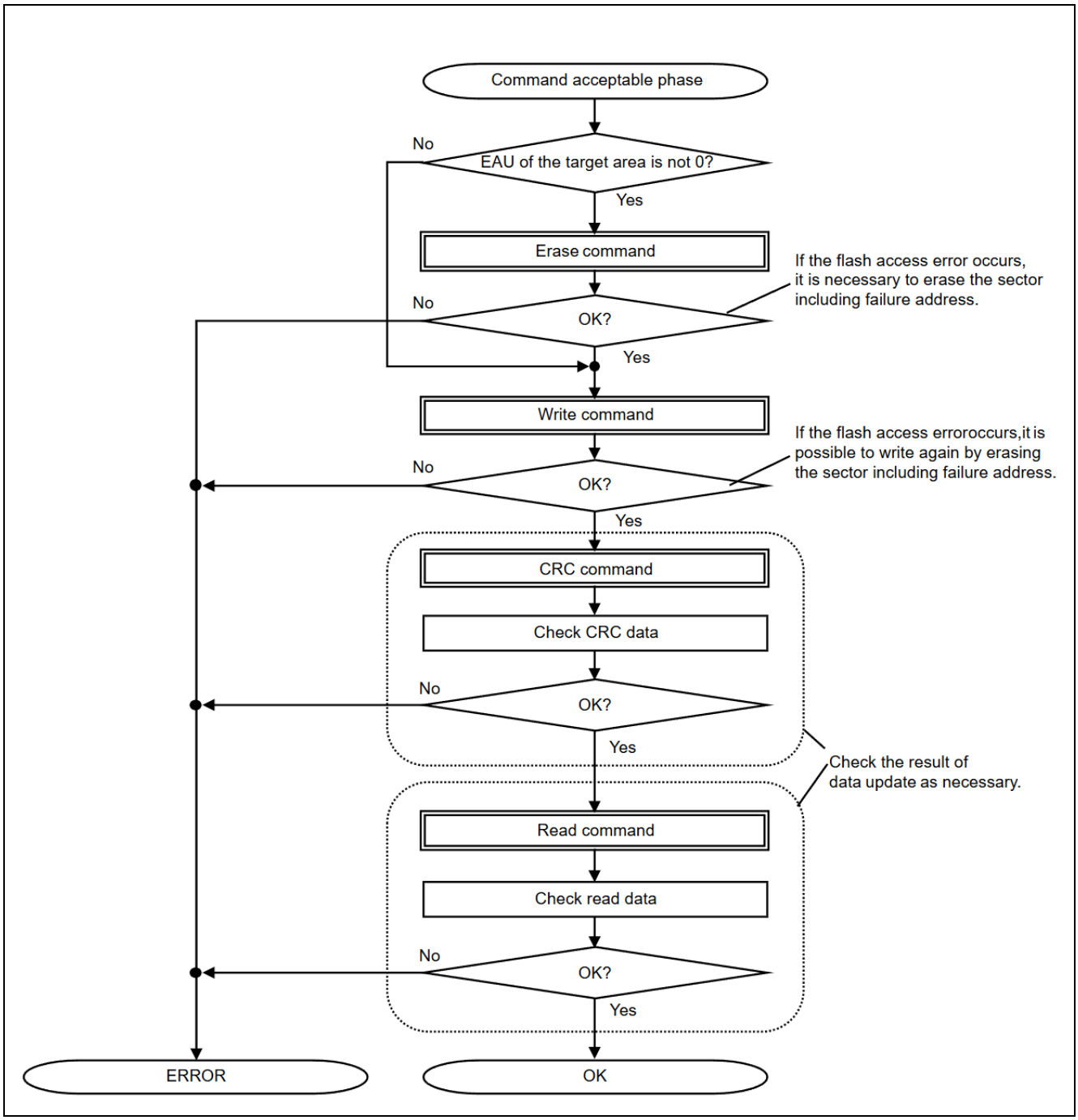


Figure 24. Data Programming

7.5 Initializing Memory

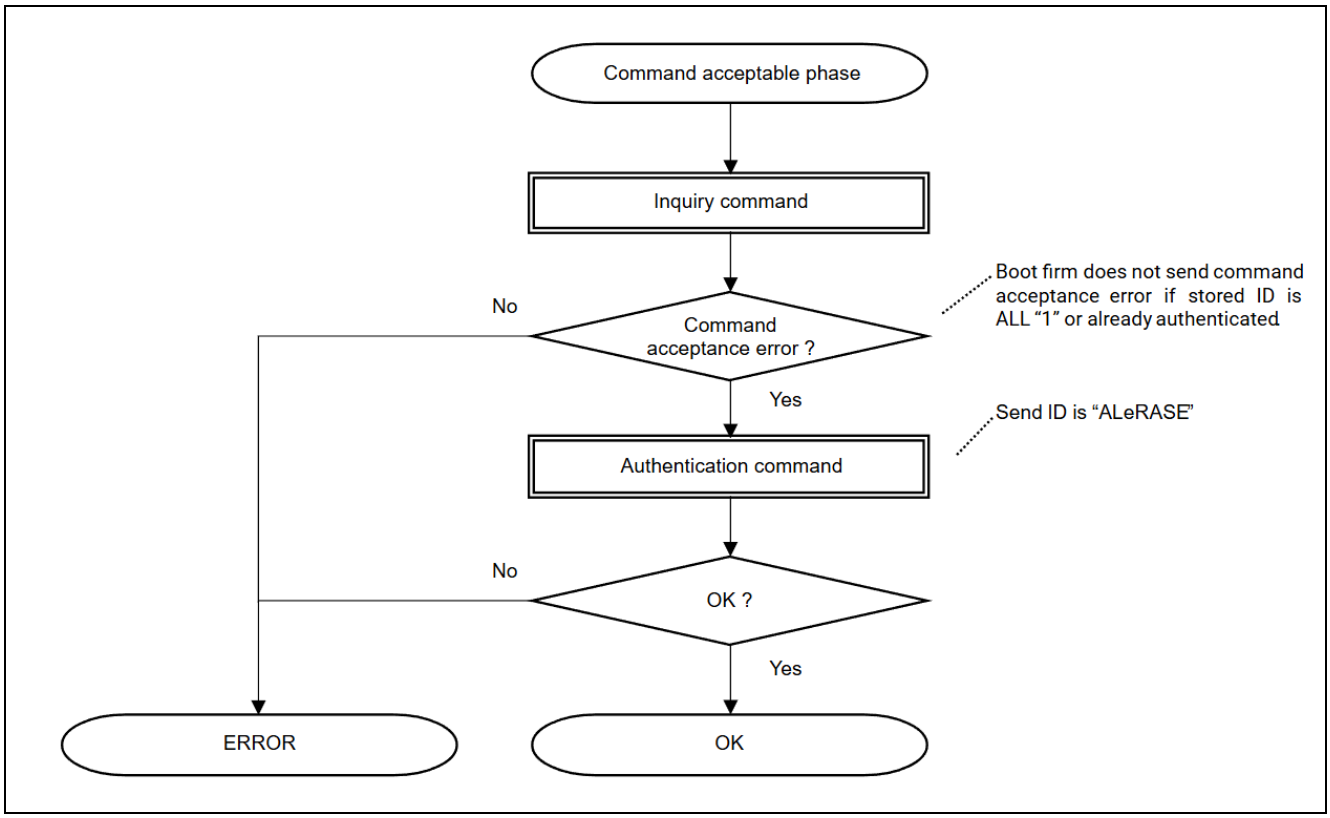


Figure 25. Initializing Memory

7.6 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 acceptable phase.

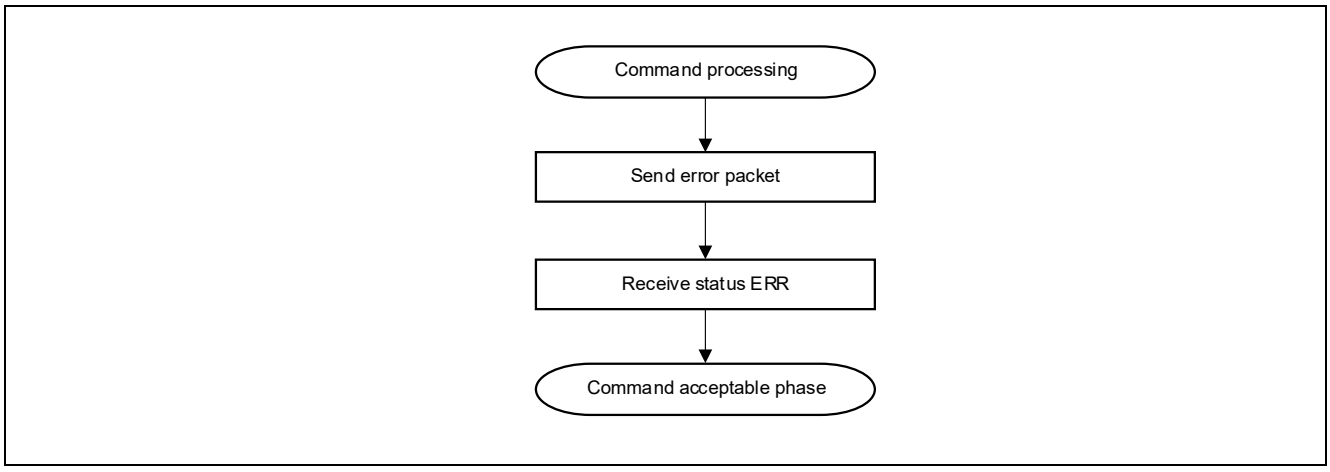


Figure 26. Command Cancel

Example: Error packets to end the command:

Command	When to send error packets	Example of the error packet		
Write command	Data packet [write data]	SOD	(1 byte)	81h
Read command	Data packet [status OK]	LNH	(1 byte)	00h
		LNL	(1 byte)	01h
		RES	(1 byte)	FFh (ERR)
		SUM	(1 byte)	00h
		ETX	(1 byte)	03h

8. AC Characteristics

8.1.1 Communication Setting Phase

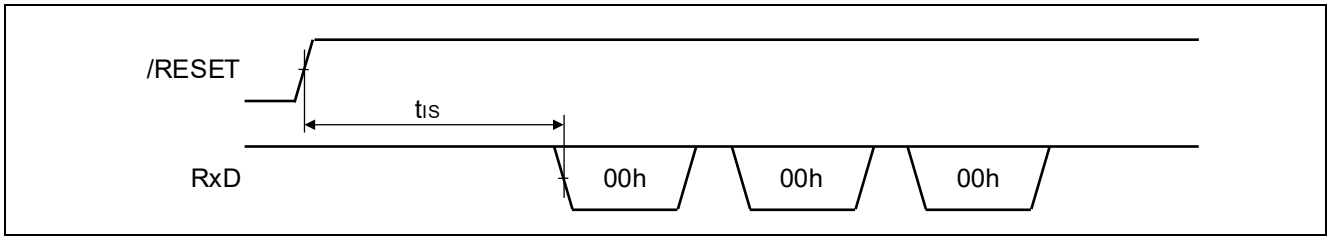


Figure 27. 2-wire UART Communication

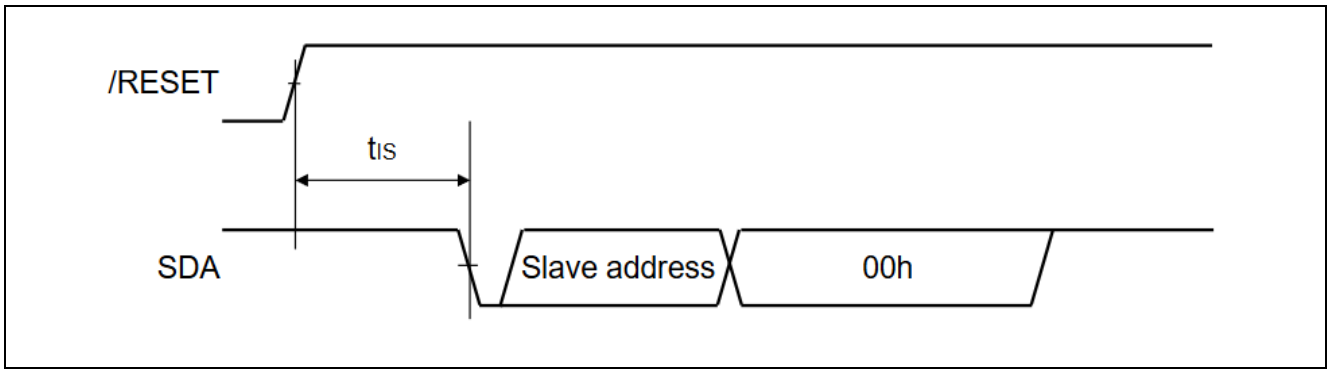


Figure 28. I2C communication

Parameter	Symbol	Min	Typ	Max	Unit
Initial setting time	tIS	-	-	42	ms

8.1.2 Authentication Command

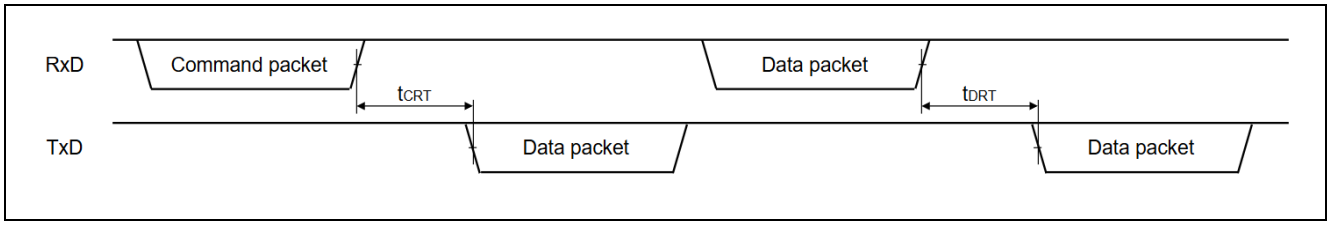


Figure 29. Authentication Command

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

8.1.3 Inquiry Command

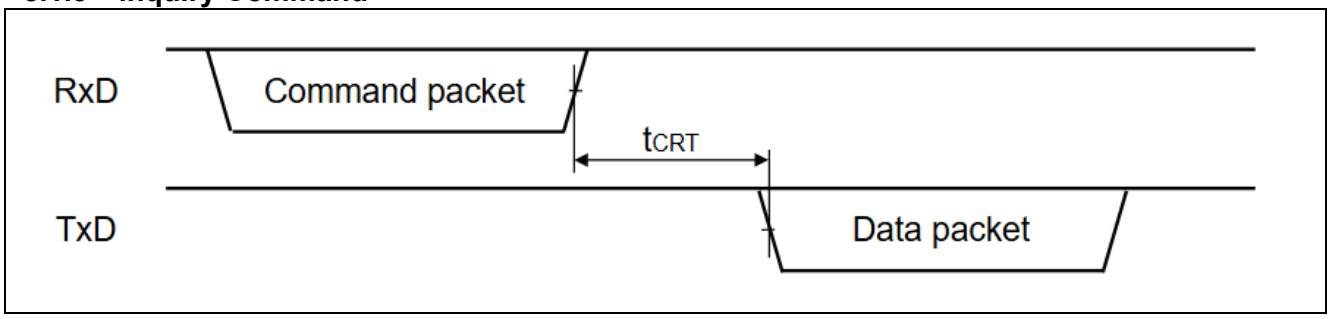


Figure 30. Inquiry Command

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

8.1.4 Signature Request Command

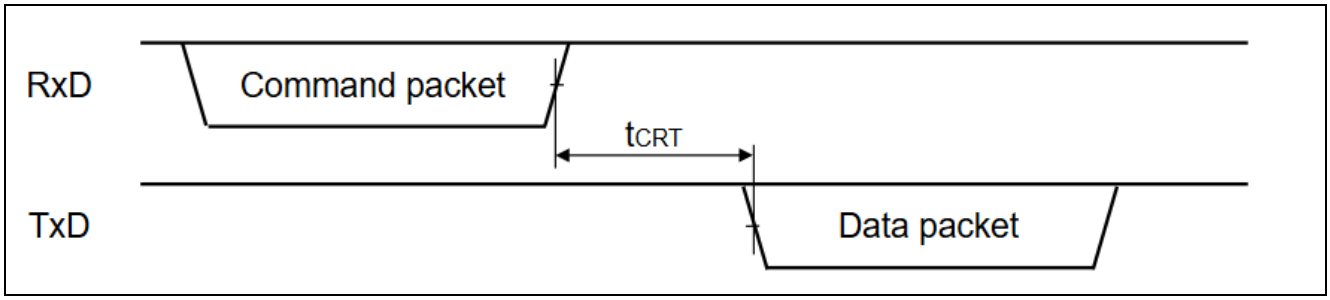


Figure 31. Signature Request Command

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

8.1.5 Area Information Request Command

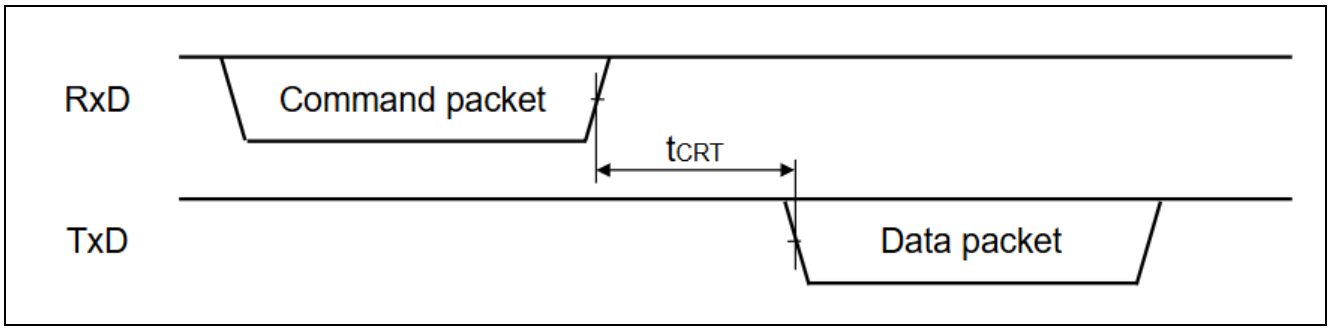


Figure 32. Area Information Request Command

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

8.1.6 Baudrate Setting Command

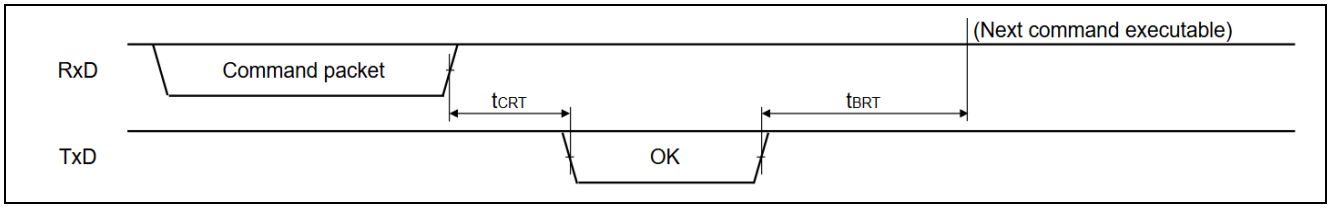


Figure 33. Baudrate Setting Command

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

8.1.7 Erase Command

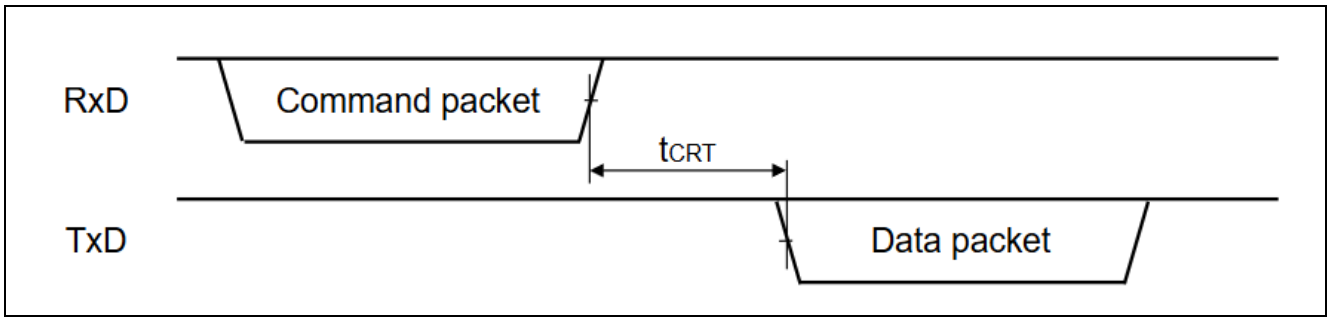


Figure 34. Erase Command

Parameter	Symbol	Min	Typ	Max	Unit
Command response time	tCRT	-	-	60	s

8.1.8 Write Command

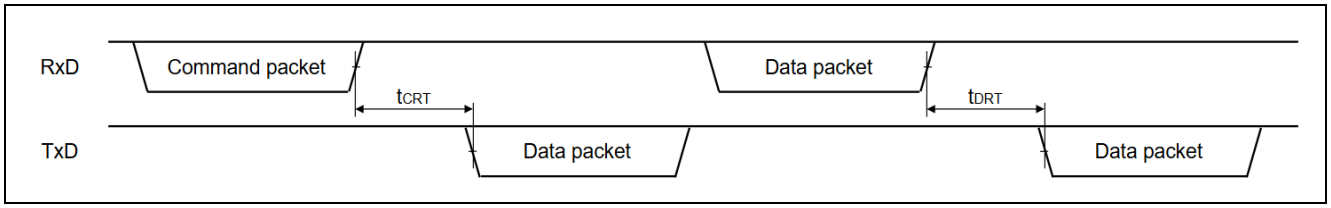


Figure 35. Write Command

Parameter	Symbol	Min	Typ	Max	Unit
Command response time	tCRT	-	-	3	s
Data response time	tDRT	-	-	30	s

8.1.9 Read Command

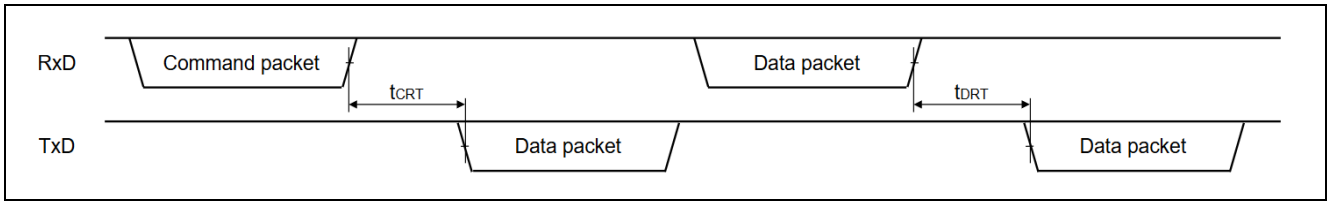


Figure 36. Read Command

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

8.1.10 CRC Command

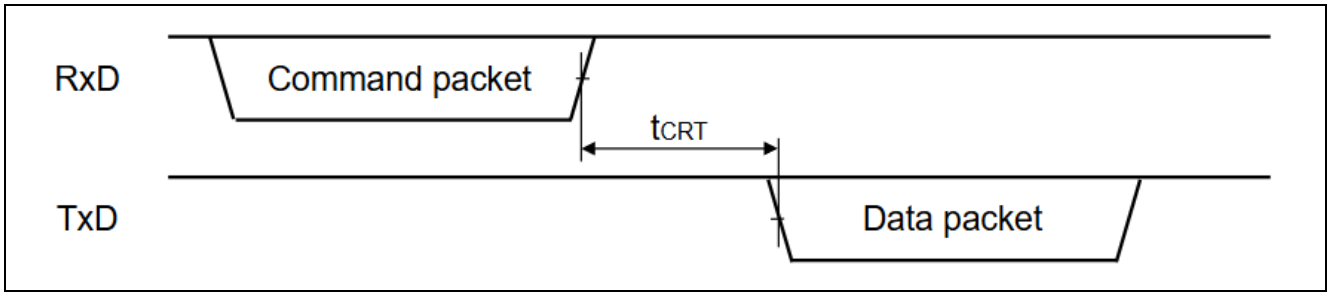


Figure 37. CRC Command

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

9. Sequencer Command List

The sequencer commands executed by each communication command are shown below.

Table 17. Sequencer Command List

Communication Command	Sequencer Command	Number of Issue Times
Authentication command	Access window information program	[Access Window Erasure] 1 time
	Block Erase	[Code Flash Erasure] [Size of User Area / 2K] times [Data Flash Erasure] [Size of Data area / 1K] times
	Config program	[Config Erasure] 5 times
Erase command	Block Erase	Depends on the designated address
Write command	Program	Depends on the designated address
	Access window information program	1 time
	Startup area information and security program	1 time
	OCDID program	Max 4 times

10. Precaution List

10.1 Write Command

(1) The following cannot be executed when 0b is written to FSPR:

- Writing to access the start/end address setting window.
- All erasure by the Authentication command with ALeRASE ID.
- All erasure by debug mode.

FSPR cannot be written back to 1 once after 0 is written.

(2) When data that is not 0xFFFF is written to SECMPUAC, the boot firmware does not operate but goes into an infinite loop after the device reset.

(3) The following cannot be executed when 0b is written to ID [127] in the Config area:

- ID authentication with the Authentication command.

- All area erasure with Authentication command ("ALeRASE" ID).
- All area erasure in the debug mode (all erasure with a debugger).

(4) The following cannot be executed when 0b is written to ID [126] in the Config area:

- All area erasure with Authentication command ("ALeRASE" ID).
- All area erasure in the debug mode (all erasure with a debugger).

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 SECMPUAC is not 0xFFFF.

11.2 Communication Setting Phase

- When following CPU exceptions occur: NMI / HardFault / MemManage / BusFault / UsageFault / SecureFault / SVCall / DebugMonitor / PendSV / SysTick.

11.3 Command Acceptable Phase

- When following CPU exceptions occur: NMI / HardFault / MemManage / BusFault / UsageFault / SecureFault / SVCall / DebugMonitor / PendSV / SysTick.

12. Causes for Software Reset

Boot firmware performs a software reset in the following cases.

12.1 Initialization Phase

- When SECMPUAC is not 0xFFFF and MD=1 is detected.

12.2 Communication Setting Phase

- When MD=1 is detected during the 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	Aug.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/.