User's Manual

E1/E20 Emulator

Additional Document for User's Manual (Notes on Connection for 78K0)

Supported Devices:

78K0

Renesas Electronics
www.renesas.com

Rev.1.00 Aug 2011

Notice

1. All information included in this document is current as of the date this document is issued. Such information, however, is subject to change without any prior notice. Before purchasing or using any Renesas Electronics products listed herein, please confirm the latest product information with a Renesas Electronics sales office. Also, please pay regular and careful attention to additional and different information to be disclosed by Renesas Electronics such as that disclosed through our website.
2. Renesas Electronics does not assume any liability for infringement of 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. No license, express, implied or otherwise, is granted hereby under any patents, copyrights or other intellectual property rights of Renesas Electronics or others.
3. You should not alter, modify, copy, or otherwise misappropriate any Renesas Electronics product, whether in whole or in part.
4. 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 of these circuits, software, and information in the design of your equipment. Renesas Electronics assumes no responsibility for any losses incurred by you or third parties arising from the use of these circuits, software, or information.
5. When exporting the products or technology described in this document, you should comply with the applicable export control laws and regulations and follow the procedures required by such laws and regulations. You should not use Renesas Electronics products or the technology described in this document for any purpose relating to military applications or use by the military, including but not limited to the development of weapons of mass destruction. Renesas Electronics products and technology may 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.
6. Renesas Electronics has used reasonable care in preparing the information included in this document, but Renesas Electronics does not warrant that such information is error free. Renesas Electronics assumes no liability whatsoever for any damages incurred by you resulting from errors in or omissions from the information included herein.
7. Renesas Electronics products are classified according to the following three quality grades: "Standard", "High Quality", and "Specific". The recommended applications for each Renesas Electronics product depends on the product's quality grade, as indicated below. You must check the quality grade of each Renesas Electronics product before using it in a particular application. You may not use any Renesas Electronics product for any application categorized as "Specific" without the prior written consent of Renesas Electronics. Further, you may not use any Renesas Electronics product for any application for which it is not intended without the prior written consent of Renesas Electronics. Renesas Electronics shall not be in any way liable for any damages or losses incurred by you or third parties arising from the use of any Renesas Electronics product for an application categorized as "Specific" or for which the product is not intended where you have failed to obtain the prior written consent of Renesas Electronics. The quality grade of each Renesas Electronics product is "Standard" unless otherwise expressly specified in a Renesas Electronics data sheets or data books, etc.

   "Standard": Computers; office equipment; communications equipment; test and measurement equipment; audio and visual equipment; home electronic appliances; machine tools; personal electronic equipment; and industrial robots.

   "High Quality": Transportation equipment (automobiles, trains, ships, etc.); traffic control systems; anti-disaster systems; anti-crime systems; safety equipment; and medical equipment not specifically designed for life support.

   "Specific": Aircraft; aerospace equipment; submersible repeaters; nuclear reactor control systems; medical equipment or systems for life support (e.g. artificial life support devices or systems), surgical implantations, or healthcare intervention (e.g. excision, etc.), and any other applications or purposes that pose a direct threat to human life.
8. You should use the Renesas Electronics products described in this document within the range specified by Renesas Electronics, especially with respect to the maximum rating, operating supply voltage range, movement power voltage range, heat radiation characteristics, installation and other product characteristics. Renesas Electronics shall have no liability for malfunctions or damages arising out of the use of Renesas Electronics products beyond such specified ranges.
9. Although Renesas Electronics endeavors to improve the quality and reliability of its products, semiconductor products have specific characteristics such as the occurrence of failure at a certain rate and malfunctions under certain use conditions. Further, Renesas Electronics products are not subject to radiation resistance design. Please be sure to implement safety measures to guard them against the possibility of physical injury, and injury or damage caused by fire in the event of the failure of a Renesas Electronics product, 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, please evaluate the safety of the final products or system 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. Please use Renesas Electronics products in compliance with all applicable laws and regulations that regulate the inclusion or use of controlled substances, including without limitation, the EU RoHS Directive. Renesas Electronics assumes no liability for damages or losses occurring as a result of your noncompliance with applicable laws and regulations.
11. This document may not be reproduced or duplicated, in any form, in whole or in part, without prior written consent of Renesas Electronics.
12. Please contact a Renesas Electronics sales office if you have any questions regarding the information contained in this document or Renesas Electronics products, or if you have any other inquiries.

(Note 1) "Renesas Electronics" as used in this document means Renesas Electronics Corporation and also includes its majority-owned subsidiaries.

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

CONTENTS

CHAPTER 1 OUTLINE

1.1. Features

E1/E20 Emulator (hereinafter referred to as E1/E20) is an on-chip debug emulator with flash programming function, which is used for debugging and programming a program to be embedded in on-chip flash memory microcontrollers. This product can debug with the target microcontroller connected to the user system, and can write programs to the on-chip flash memory of microcontrollers.

1.2 Cautions on Using E20

The functions used for debugging of the 78K0 device by using the E20 are the same as in the E1. Large trace function, characteristic functions of the E20, cannot be used. The power supply function from the E20 is not supported.

1.3 Configuration of Manuals

Documentation for the E1/E20 emulator manual is in two parts: the E1/E20 Emulator User's Manual and the E1/E20 Emulator Additional Document for User's Manual (this manual). Be sure to read both of the manuals before using the E1/E20 emulator.

(1) E1/E20 Emulator User's Manual

The E1/E20 Emulator User's Manual has the following contents:

- Components of the emulators
- Emulator hardware specification
- Connection to the emulator and the host computer and user system

(2) E1/E20 Emulator Additional Document for User's Manual

The E1/E20 Emulator Additional Document for User's Manual has the following contents:

- For use in hardware design, an example of connection and the interface circuit required to connect the emulator.
- Notes on using the emulator
- Software specifications and so on for using each microcomputers

CHAPTER 2 DESIGNING USER SYSTEM

To connect the E1/E20 emulator, a connector for the user system interface cable must be mounted on the user system. When designing the user system, read this section of this manual and the hardware manual for the MCUs.

2.1 Connecting the Emulator with the User System

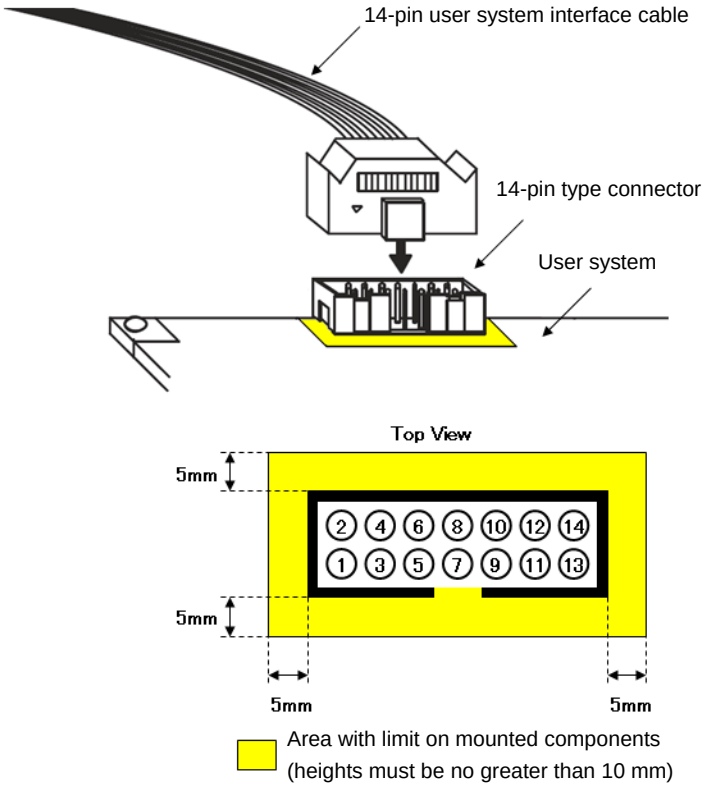
Table 2-1 shows the type numbers of the E1/E20 emulators

Table 2-1. Type Numbers

	Type Number	Manufacturer	Specification
14-pin connector	7614-6002	Sumitomo 3M Limited	14-pin straight type (Japan)
	2514-6002	3M Limited	14-pin straight type (other countries)

Figure 2.1 shows examples of the connection between a user system interface cable of the 14-pin type. Do not mount other components with a height exceeding 10 mm within 5 mm of the connector on the user system. 38-pin of the E20 is not supported. To use the E20, use the 38-pin/14-pin conversion adapter [R0E000200CKA00] that comes with the E20 for connection.

Figure 2-1. Connecting the User System Interface Cable to the 14-pin Connector of the E1 Emulator



2.2 Pin Assignments of the Connector on the User System

Table 2-2 shows the pin assignments of the 14-pin connector.

Table 2-2. Pin Assignments of the Connector on the User System (14-Pin)

Pin No.	Pin Name (——— : Active low)	Input/Output[Note 1]
1	R.F.U	–
2	GND[Note 2]	–
3	CLK	Output
4	FLMD0	Output
5	RxD	Input
6	$\overline{\text{RESET\_IN}}$	Input
7	TxD/DATA	Output/Input and Output
8	VDD	–
9	R.F.U	Output
10	$\overline{\text{RESET\_OUT}}$[Note 3]	Output
11	R.F.U	–
12	GND[Note 2]	–
13	$\overline{\text{RESET\_OUT}}$[Note 3]	Output
14	GND[Note 2]	–

Notes 1. As seen from E1/E20.

2. Securely connect pins 2, 12, and 14 of the connector to GND of the user system. These pins are used for electrical grounding as well as for monitoring of connection with the user system by the E1/E20.

3. Securely connect both pin 10 and pin 13. These pins are also used to monitor the user system.

Table 2-3. Pin Functions

Pin Name	Input/Output[Note]	Description
RESET_IN	Input	Pin used to input reset signal from the user system
RESET_OUT	Output	Pin used to output reset signal to the target device
CLK	Output	Pin used to output clock signal to the target device
FLMD0	Output	Pin used to set the target device to debug mode or programming mode (This pin is used by only the devices that have OCDxA and OCDxB pins.)
RxD	Input	Pin used to receive command/data from the target device
TxD/DATA	Output/Input and Output	Pin used to transmit command/data to the target device
R.F.U.	–	This pin is reserved. For the connection of the reserved pins, see each circuit related to the pins.

Note As seen from E1/E20.

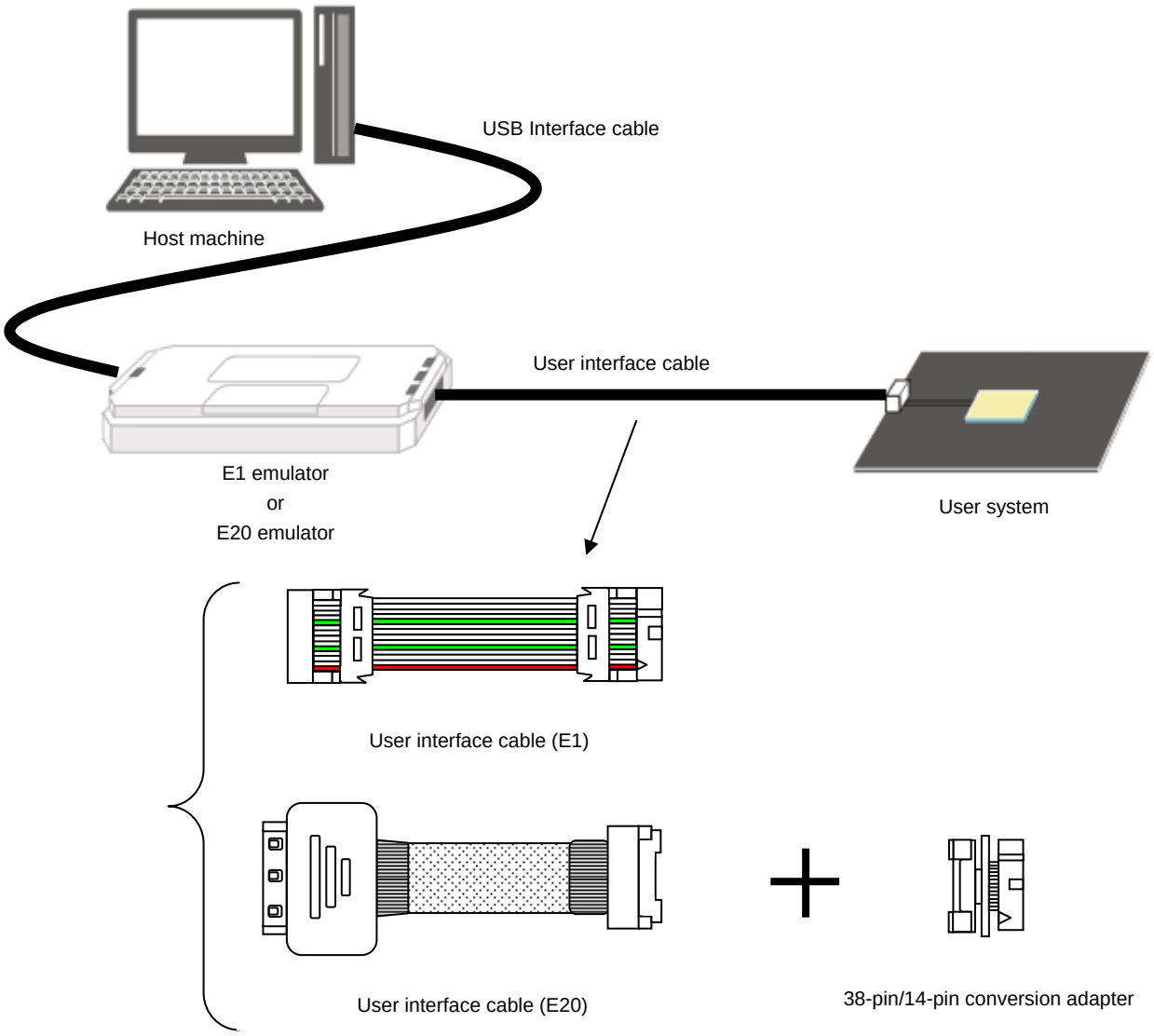
2.3 System Configuration

Figure 2-2 shows the system configuration used for the E1/E20. For cautions on connection, refer to the E1/E20 User's Manual. As software used on the host machine, use the "CubeSuite+" when on-chip debugging is used, or use the "Renesas Flash programmer" for flash programming. For details, refer to the following URL's.

- Integrated development environment "CubeSuite+" website
  http://www.renesas.com/cubesuite+

- Flash writing tool "Renesas Flash Programmer" website
  http://www.renesas.com/rfp

Figure 2-2. Connection Diagram of E1/E20



Remark To use it with the E20, connect the 38-pin/14-pin conversion adapter to the user interface cable (E20). 38-pin is not supported.

2.4 Examples of Recommended Connection Circuits Between Emulator Connection Connector and MCU

2.4.1 Cautions on creating recommended circuits and examples of circuit connection

The following are common cautions on creating recommended circuits.

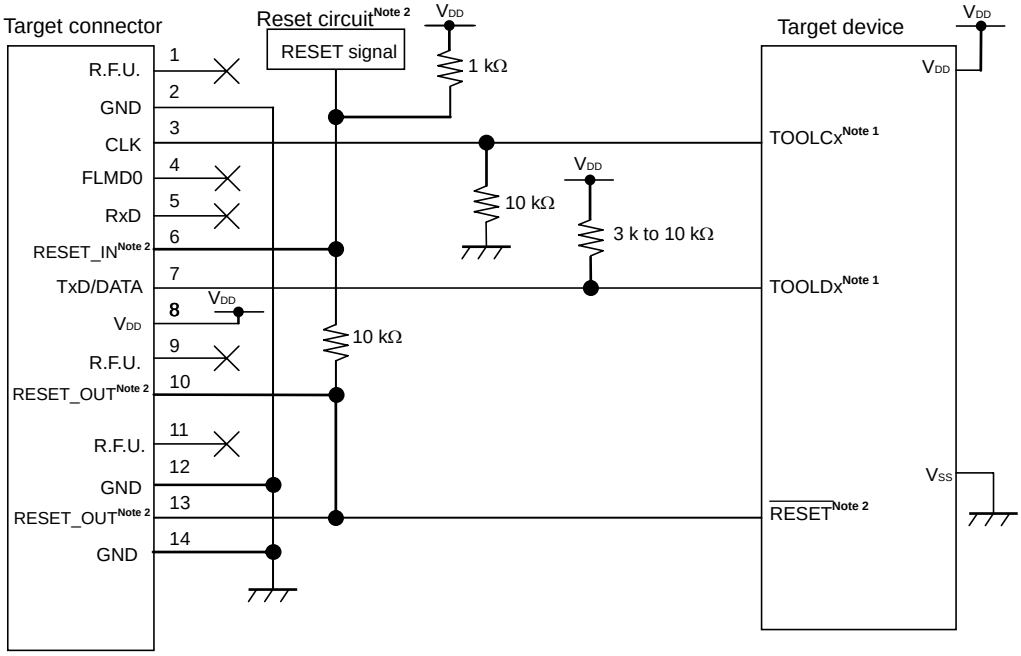
- The circuits and resistance values listed are recommended but not guaranteed. Determine the circuit design and resistance values by taking into account the specifications of the target device and noise. For flash programming for mass production, perform sufficient evaluation about whether the specifications of the target device are satisfied.
- For processing of pins not used by the E1/E20, refer to the user's manual of the device.
- Connect the TxD (transmission side) of the target device to the RxD (reception side) of the target connector. Connect the TxD (transmission side) of the target connector to the RxD (reception side) of the target device.
- Securely connect pins 2, 12, and 14 of the connection to GND of the user system. These pins are used for electrical grounding as well as for monitoring of connection with the user system by the E1/E20.
- Securely connect both pin 10 and pin 13. These pins are also used to monitor the user system.
- Pins for on-chip debugging and programming vary depending on the device. Refer to Table 2-4.
- In a circuit using the external oscillator, debugging with the debug pin connected to the external oscillator cannot be done.
- When using the pins used for debugging or serial programming in your circuit, use a jumper for isolation to avoid conflict of signals. For details, refer to **2.4.3 Cautions on connecting the user circuit to the RxD pin of the target device.**

Table 2-4. Supported Device and Interfaces

<table>
<tr><th>Supported Device</th><th>Programming Interface</th><th>Debug Interface</th><th>Circuit diagram</th></tr>
<tr><td>78K0/Kx2-L</td><td colspan="2" rowspan="3">TOOLCx, TOOLDx (x = 0, 1)</td><td rowspan="3">Figure 2-3</td></tr>
<tr><td>78K0/Ix2</td></tr>
<tr><td>78K0/Fx2-L</td></tr>
<tr><td>78K0/Kx2</td><td rowspan="12">TxD/RxD (UART)Note 1</td><td rowspan="12">OCDxA, OCDxB
(x = 0, 1)</td><td rowspan="12">Figure 2-4
Figure 2-5
Figure 2-6
Figure 2-7</td></tr>
<tr><td>μPD78F8039
(μPD78F8017 to μPD78F8020,
μPD78F8026 to μPD78F8039)</td></tr>
<tr><td>78K0/Kx2-A</td></tr>
<tr><td>78K0/Kx2-C</td></tr>
<tr><td>78K0/Lx2</td></tr>
<tr><td>78K0/Lx3</td></tr>
<tr><td>78K0/Lx3-M</td></tr>
<tr><td>78K0/Dx2</td></tr>
<tr><td>78K0/Fx2</td></tr>
<tr><td>μPD78F0730</td></tr>
</table>

Note Check the available channels in the user's manual of the target device (chapter on flash memory).

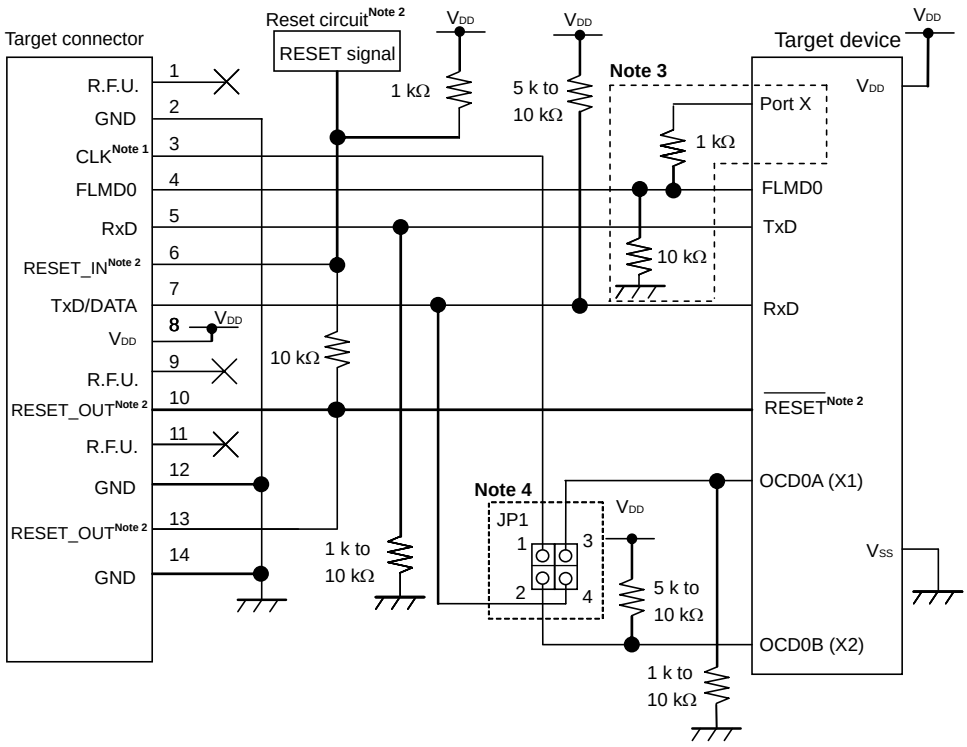
Figure 2-3. Using One Connector for Both Debugging and Programming (TOOLCx/TOOLDx Communication x = 0, 1)



Notes 1. When the external clock is used, use the TOOLC1 and TOOLD1 pins for debugging.

2. This connection is designed assuming that the RESET signal is output from the N-ch open-drain buffer (output resistance: 100 Ω or less). For details, refer to **2.4.2 Connection of reset pin**.

Figure 2-4. Using One Connector for Both Debugging and Programming (OCD0A, OCD0B Communication)



Notes 1. During on-chip debugging, the X1 oscillation circuit cannot be used as the operation clock of the target device.

2. This connection is designed assuming that the RESET signal is output from the N-ch open-drain buffer (output resistance: 100 Ω or less). For details, refer to **2.4.2 Connection of reset pin**.

3. The circuit enclosed by a dashed line is designed for flash self programming, which controls the FLMD0 pin via ports. (This circuit is used to control the port during flash programming.)
Connect any port that can output data to FLMD0 via a resistor.
When not using flash self programming, process the pins according to the device specifications.

4. Table 2-5 shows the JP1 setting.

Table 2-5. JP1 Setting

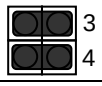
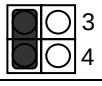
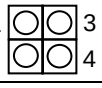
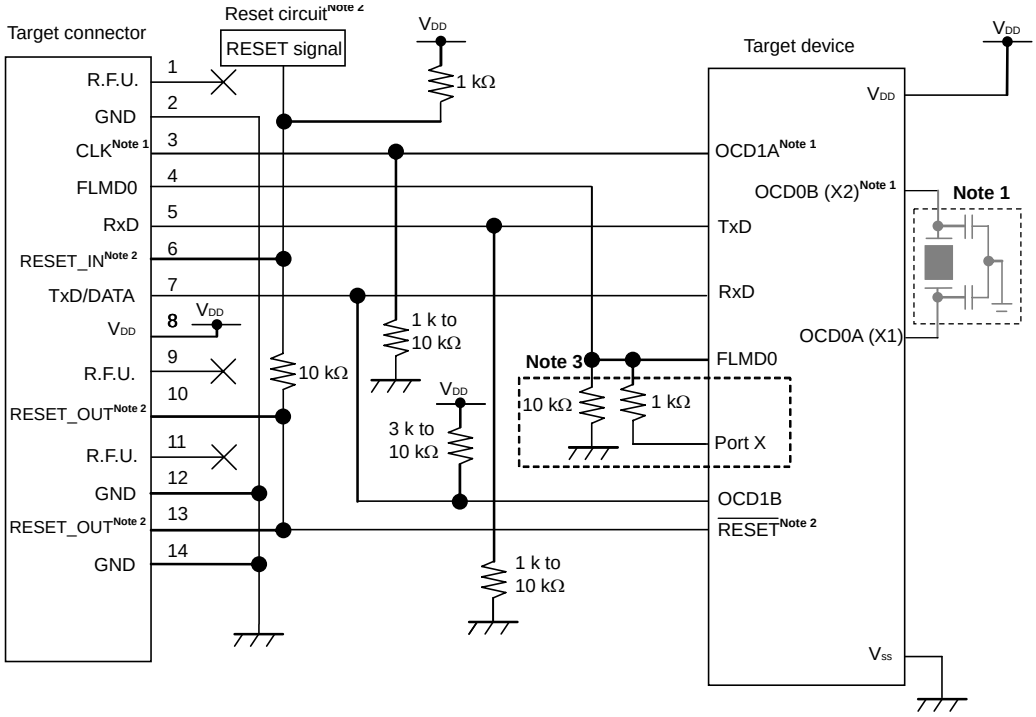
	JP1 Setting
On-chip debugging (1-3 short, 2-4 short)	
Flash programming (1-2 short, 3-4 open)	
E1/E20 not connected (all open)	

Figure 2-5. Using One Connector for Both Debugging and Programming (OCD1A, OCD1B Communication)

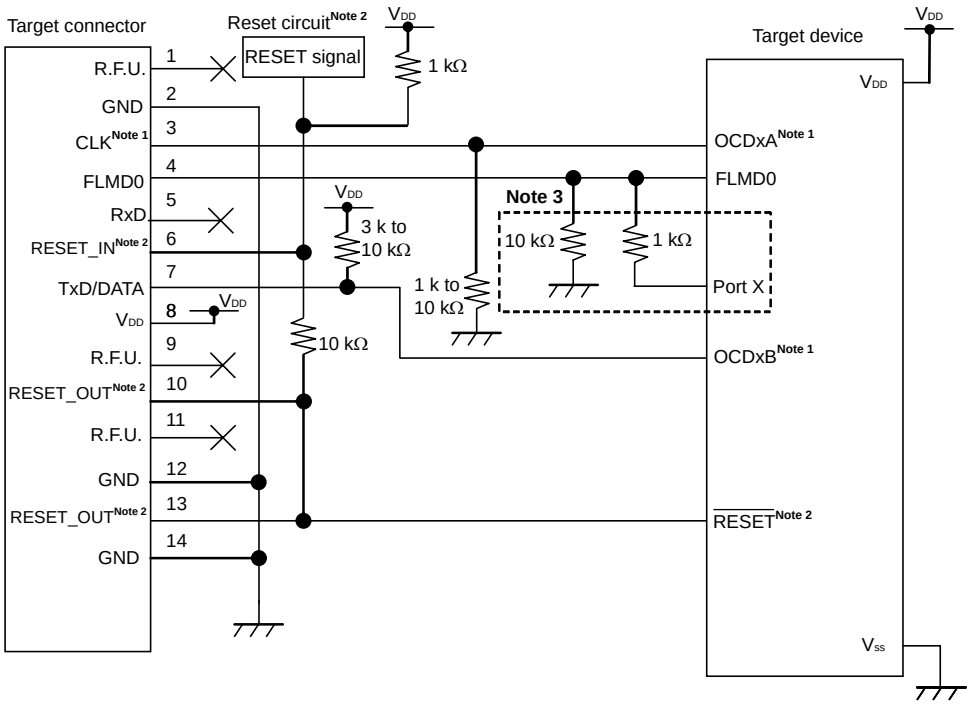


Notes 1. When the external clock is not used, connect the CLK pin of the E1 to the X2 pin of the device.

2. This connection is designed assuming that the RESET signal is output from the N-ch open-drain buffer (output resistance: 100 Ω or less). For details, refer to **2.4.2 Connection of reset pin**.

3. The circuit enclosed by a dashed line is designed for flash self programming, which controls the FLMD0 pin via ports. Connect any port that can output data to FLMD0 via a resistor.
When not using flash self programming, process the pins according to the device specifications.

Figure 2-6. Only Debugging Is Performed (OCDxA, OCDxB Communication x = 0, 1)

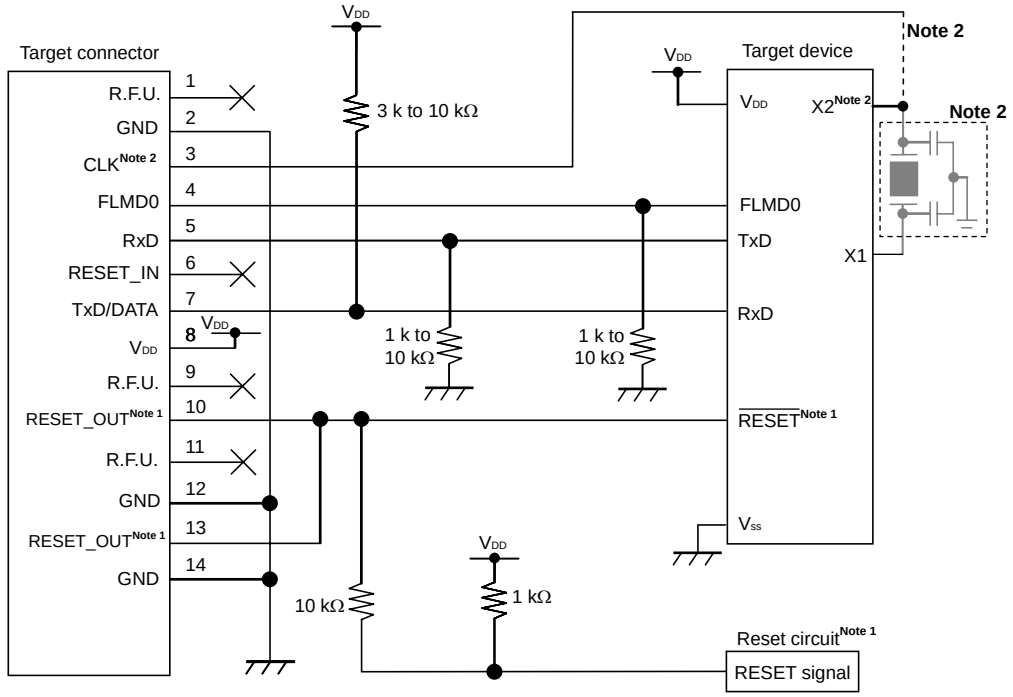


Notes 1. When the external clock is used, use the OCD1A and OCD1B for debugging.

2. This connection is designed assuming that the RESET signal is output from the N-ch open-drain buffer (output resistance: 100 Ω or less). For details, refer to **2.4.2 Connection of reset pin**.

3. The circuit enclosed by a dashed line is designed for flash self programming, which controls the FLMD0 pin via ports. Connect any port that can output data to FLMD0 via a resistor.
When not using flash self programming, process the pins according to the device specifications.

Figure 2-7. Only Programming Is Performed



Notes 1. This connection is designed assuming that the RESET signal is output from the N-ch open-drain buffer (output resistance: 100 Ω or less). For details, refer to **2.4.2 Connection of reset pin**.

2. When the external clock is not used, connect the CLK pin of the E1 to the X2 pin of the device.

2.4.2 Connection of reset pin

This section describes the connection of the reset pin, for which special attention must be paid, in circuit connection examples shown in the previous section.

During on-chip debugging, a reset signal from the target system is input to E1/E20, masked, and then output to the target device. Therefore, the reset signal connection varies depending on whether E1/E20 is connected.

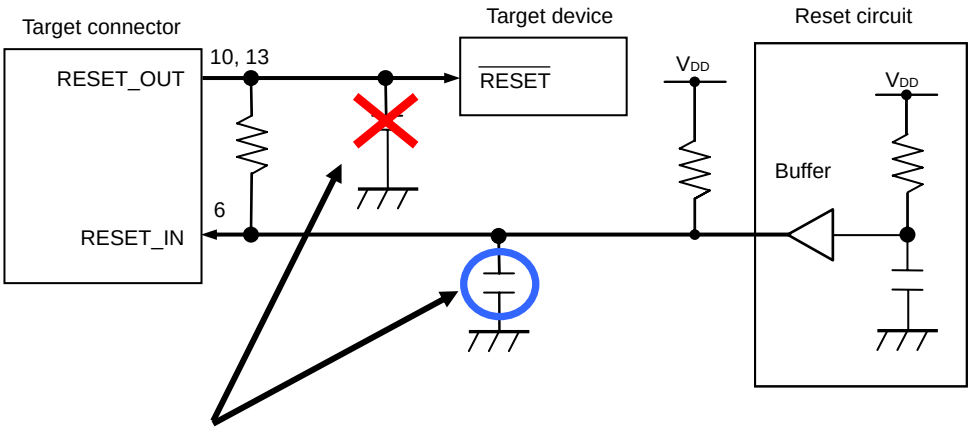
For flash programming, the circuit must be designed so that the reset signals of the user system and E1/E20 do not conflict.

Select one of the following methods and connect the reset signal in the circuit. The details of each method are described on the following pages.

(1) Automatically switching the reset signal via resistor (recommended; described in recommended circuit connection in the previous section)
(2) Automatically switching the reset signal via selector logic
(3) Manually switching the reset signal with jumper
(4) Resetting the target device by power-on clear (POC) only

Caution Do not connect a capacitor to the RESET_OUT pin.

Figure 2-8. Cautions for Using RESET_OUT Pin



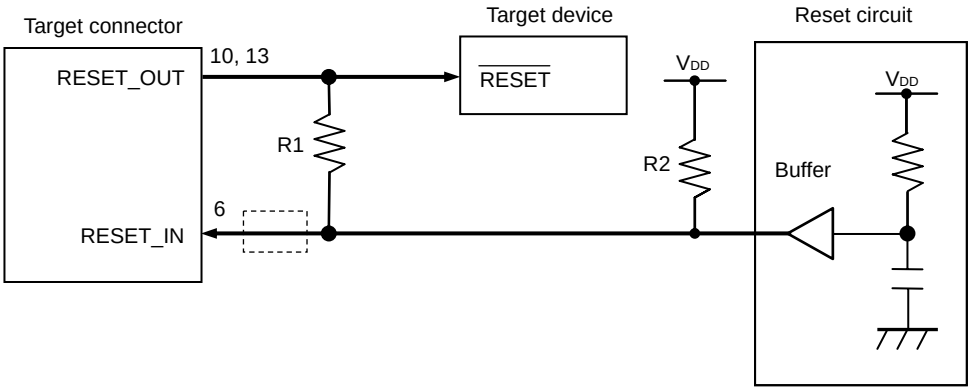
If a capacitance element such as a capacitor or resistor is connected to the RESET_OUT pin, the target device might not be able to communicate with E1. If necessary, connect a capacitor to the RESET_IN pin.

(1) Automatically switching the reset signal via resistor

Figure 2-9 illustrates the reset pin connection described in **2.4.1 Circuit connection examples**.

This connection is designed assuming that the reset circuit on the target system contains an N-ch open-drain buffer (output resistance: 100 Ω or less). The V_{DD} or GND level may be unstable when the logic of RESET_IN/OUT of E1/E20 is inverted, so observe the conditions described below in **Remark**.

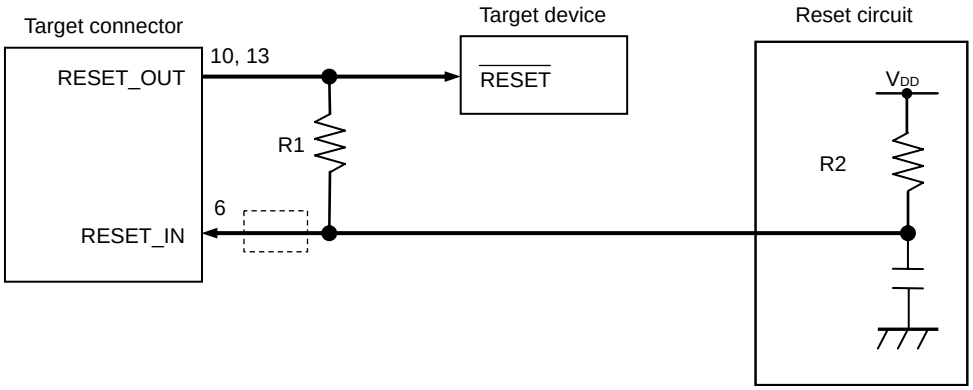
Figure 2-9. Circuit Connection with Reset Circuit That Contains Buffer



Remark Make the resistance of at least R1 ten times that of R2, R1 being 10 kΩ or more.
Pull-up resistor R2 is not required if the buffer of the reset circuit consists of CMOS output.
The circuit enclosed by a dashed line is not required when only flash programming is performed.

Figure 2-10 illustrates the circuit connection for the case where the reset circuit on the target system contains no buffers and the reset signal is only generated via resistors or capacitors. Design the circuit, observing the conditions described below in **Remark**.

Figure 2-10. Circuit Connection with Reset Circuit That Contains No Buffers



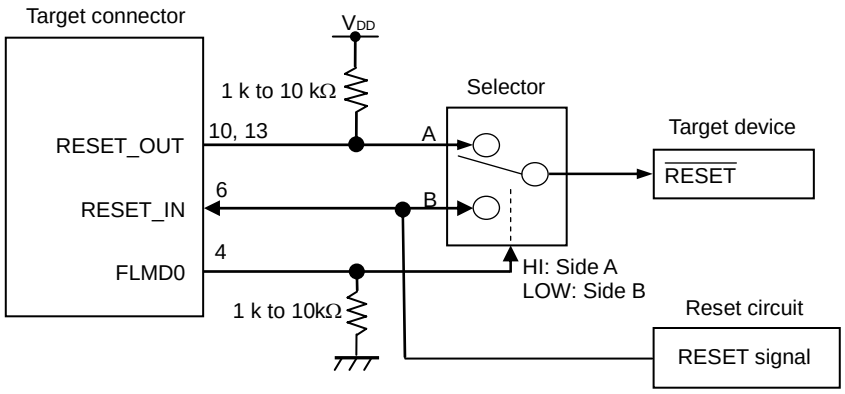
Remark Make the resistance of at least R1 ten times that of R2, R1 being 10 kΩ or more.
The circuit enclosed by a dashed line is not required when only flash programming is performed.

(2) Automatically switching the reset signal via selector logic (devices other than 78K0/Kx2-L, 78K0/Ix2, 78K0/Fx2-L)

Figure 2-11 illustrates the circuit connection for the case where the reset signal is switched automatically using the selector logic, with or without E1/E20 connected. When using flash self programming, refer to Figure 2-12.

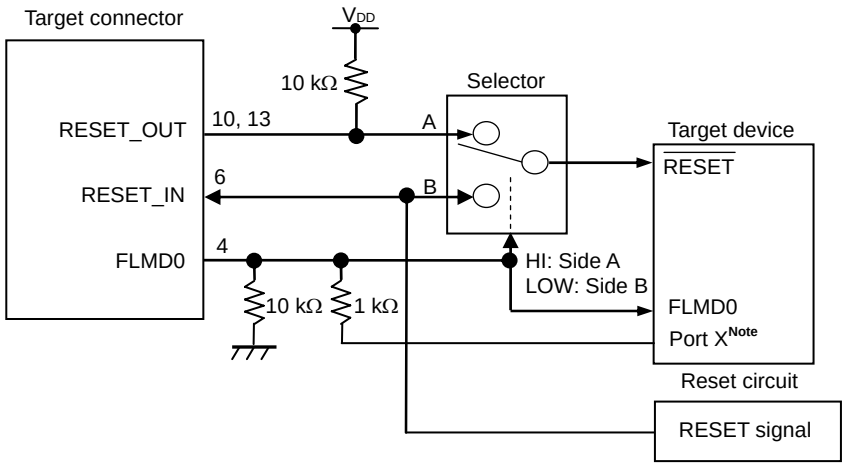
Caution The reset circuit is disconnected if flash self programming is performed (FLMD0 = HI) without E1/E20 connected, so no signals can be input to the _RESET pin.

Figure 2-11. Circuit Connection for Automatically Switching Reset Signal with Selector Logic (When Not Using Flash Self Programming)



Remark FLMD0 is high level when E1/E20 is connected, and FLMD0 is pulled down when MINICUBE2 is not connected.

Figure 2-12. Circuit Connection for Automatically Switching Reset Signal with Selector Logic (When Using Flash Self Programming)

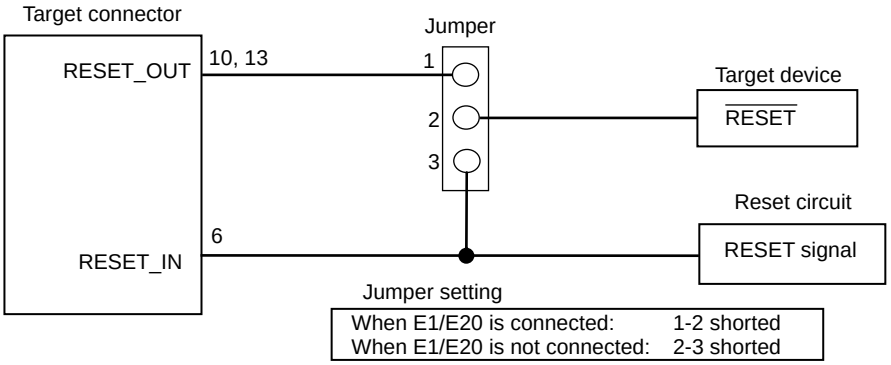


Note This circuit is designed for flash self programming, which controls the FLMD0 pin via ports.

(3) Manually switching the reset signal with jumper

Figure 2-13 illustrates the circuit connection for the case where the reset signal is switched using the jumper, with or without E1/E20connected. This connection is simple, but the jumper must be set manually.

Figure 2-13. Circuit Connection for Switching Reset Signal with Jumper

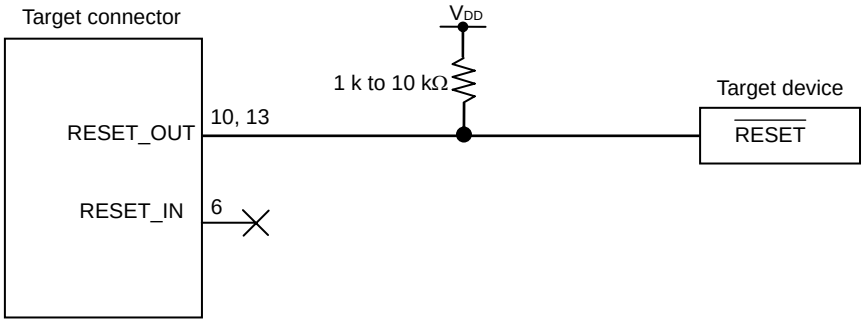


(4) Resetting the target device by power-on clear (POC) only

Figure 2-14 illustrates the circuit connection for the case where the target device is only reset via POC without using the reset pin. RESET_OUT becomes active when power is applied to E1/E20.

Even if power supply to the target system is turned off during debugging, pseudo POC function emulation is available because RESET_OUT becomes active. Note that the POC detection voltage is about 1 V higher than that of the target device.

Figure 2-14. Circuit Connection for the Case Where Target Device Is Only Reset via POC

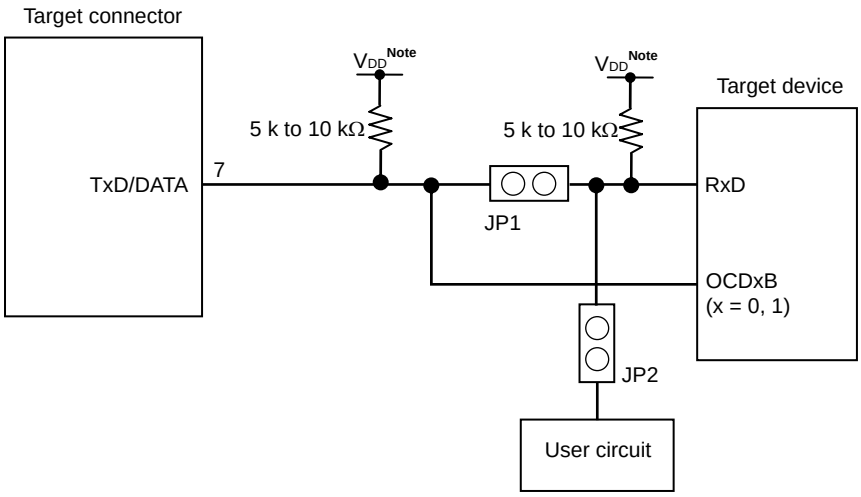


2.4.3 Cautions on connecting the user circuit to the RxD pin of the target device

Figure 2-15 shows an example of circuit when the user circuit is connected to the RxD pin of the target device.

When the user circuit is connected to the RxD pin of the target device, use the jumpers (JP1 and JP2) so that the connection between the RxD and OCDxB (x= 0, 1) of the target device and the connection between the RxD pin of the target device and user circuit can be disconnected. For the jumper (JP1, JP2) settings for various operations, refer to Table 2-6. If the jumper (JP1, JP2) settings are wrong in various operations, it may not function normally due to a signal conflict.

Figure 2-15. Example of Circuit When User Circuit Is Connected to RxD Pin of Target Device



Note Mount a pull-up resistor between TxD/DATA of the target connector and JP1 and between RxD of the target device and JP1 for pin processing for on-chip debugging and for E1/E20 not connected (single unit operation). (They are pin processings for the OCDxB pin and RxD pin of the target device, respectively.)

Table 2-6. Jumper (JP1, JP2) Setting for Various Operations

	JP1	JP2
On-chip debug E1/E20 not connected (single unit operation)	Open	Short
Flash programming	Short	Open

2.5 Cautions on User System Design

Note the following cautions when designing the user system.

If possible, do not create sections in which the communication lines for debugging (OCDxA, OCDxB, TOOLCx, and TOOLDx) run in parallel in the user system. If this cannot be prevented, shorten the sections as much as possible.
When using TOOLC0 and TOOLD0 as communication lines for debugging by using X1 or X2 oscillation, be sure to make the section where TOOLC0 and TOOLD0 run in parallel less than 30 mm.

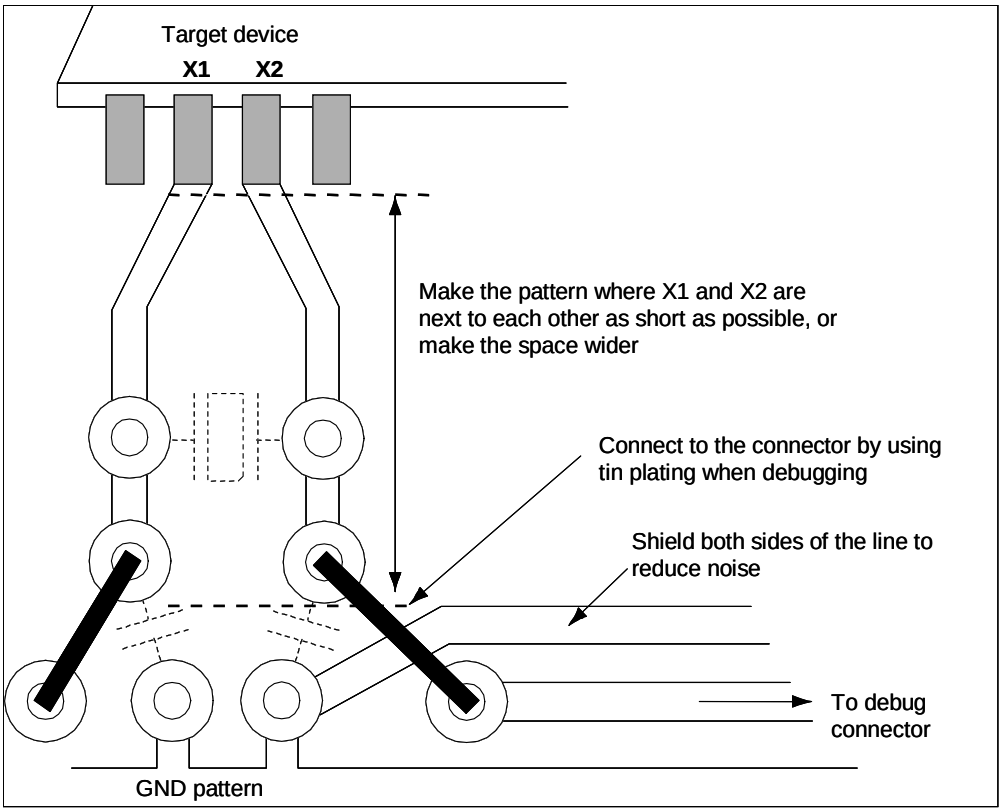
Use a GND pattern to shield the communication lines for debugging (OCDxA, OCDxB, TOOLCx, and TOOLDx) to reduce their capacitive load, because the lines are used for high-speed communication.

Make the distance between the target connector and the target device as short as possible.

Before shipping the product, use jumpers or other means to physically separate the X1 and X2 pins from the target connector in order to ensure normal clock oscillation.

To use OCD0A, OCD0B, TOOLC0, and TOOLD0 as communication pins for debugging, remove elements such as resonator capacitance and feedback resistors, so that the signals do not degrade due to capacitive load.

Figure 2-16. Reference Diagram of User System



2.6 Clock Setting

The clock signal generated by the X1 oscillator, internal high-speed oscillator, or subsystem clock oscillator can be used for the clock signal of the target device during on-chip debugging. Setting up each is described below.

Remark In the E1/E20, operation clock of the target device can be supplied from the emulator instead of the X1 oscillator circuit.
Subsystem clock supply from the emulator is not supported.

(1) Using the X1 oscillator

a. Using the clock on the user system (only when OCD1A/OCD1B or TOOLD1/TOOLC1 is used)
Select “Generate by emulator” for <1> in Figure 2-17. For <2>, select the frequency used for communication between the E1/E20 and target device from 4/8/16MHz. Download time that is dependent on the communication speed is shorter at 16MHz than 4MHz.

b. Using the clock generated in the E1/E20 (only when OCD0A/OCD0B or TOOLD0/TOOLC0 is used)
Select “Generate by emulator” for <1> in Figure 2-17. For <2>, select the frequency used for communication between the E1/E20 and target device from 4/8/16MHz.

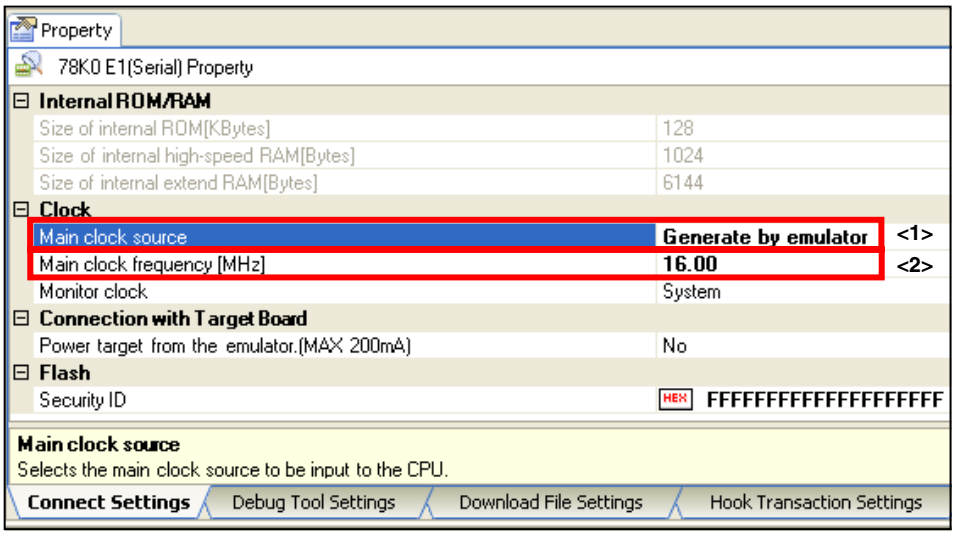
(2) Using the internal high-speed oscillator (independent of the pin used)

Select “Generate by emulator” for <1> in Figure 2-17. For <2>, select the frequency used for program download upon debugger startup from 4/8/16MHz. Download time that is dependent on the communication speed is shorter at 16MHz than 4MHz.

(3) Using the subsystem clock (independent of the pin used)

No setting is required. The subsystem clock on the user system is used only with the program settings.

Figure 2-17. CubeSuite+ Setting Screen



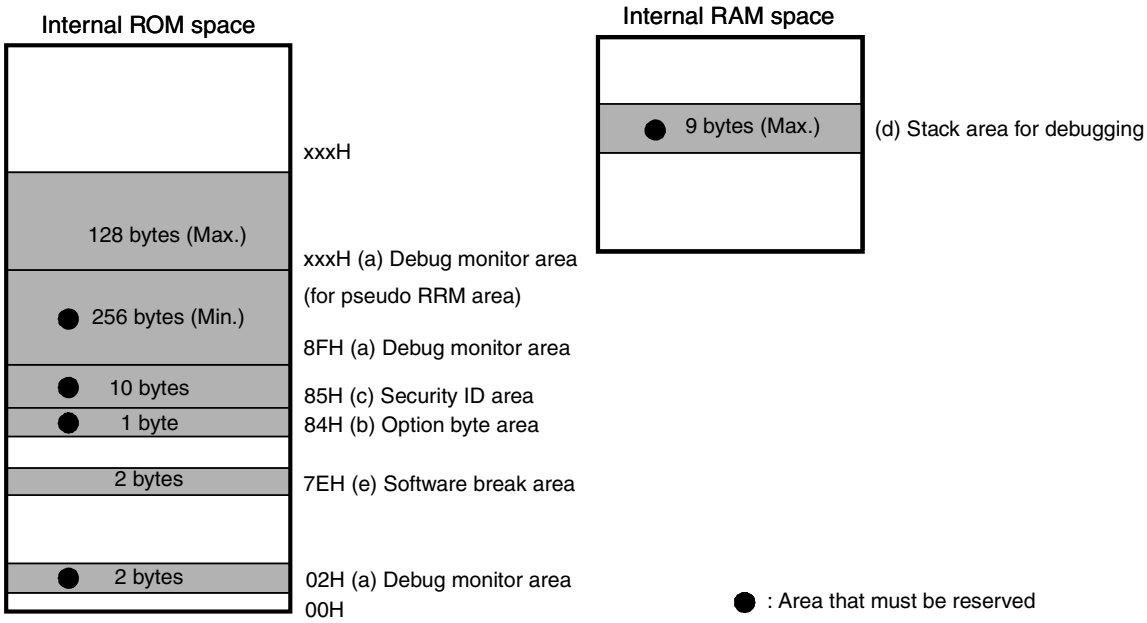
2.7 Securing of User Resources and Setting of Security ID

E1/E20 uses the user memory spaces (shaded portions in Figure 2-18) to implement communication with the target device, or each debug functions. The areas marked with a dot (•) are always used for debugging, and other areas are used for each debug function used. Refer to the descriptions of (a) to (e) on the following pages and secure these spaces in the user program or using the compiler options.

When C-SPY manufactured by IAR Systems is used, read also the following material.

- IAR C-SPY Hardware Debugger Systems User Guide issued by IAR Systems

Figure 2-18. Reserved Area Used by E1/E20



(a) Debug monitor areas (areas must be reserved)

The area of addresses 0x02, 0x03, and 0x8F and on is used to embed e a monitor program for debugging. Be sure to reserve this area. The monitor program performs initialization process for the communication interface for debugging as well as run/break processing of the CPU. If this area is rewritten by user program or flash self-programming, on-chip debugging can no longer be performed.

[Area reservation method]

Figure 2-19 shows an example of reserving an area with the CubeSuite+. Figure 2-19 is the setting dialog for link option of the CubeSuite+. In [Use on-chip debug] in the red rectangle in Figure 2-19, set [Yes (-go)] and specify [Debug monitor area size] (the area to reserve varies depending on the target device and whether the RRM function is used or not. For the required size of the debug monitor area, refer to Table 2-7.). This setting reserves the area of 0x02, 0x03, and 0x8F and on for the debug monitor.

Figure 2-19. Link Option Setting (Debug Monitor Area)

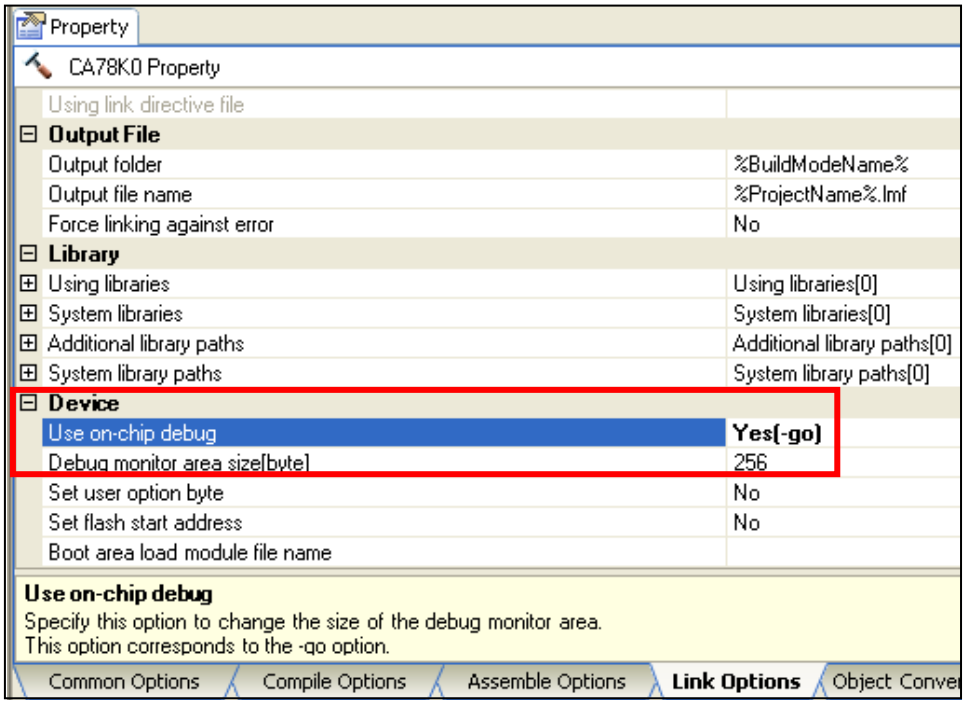


Table 2-7. Size of Debug Monitor Area to Be Reserved

Supported Device	Size of Debug Monitor Area to Be Reserved	
	RRM function is not used (bytes)	RRM function is used (bytes)
78K0/Kx2-L	256	384
78K0/Ix2		
78K0/Fx2-L		
78K0/Kx2	256	
μPD78F8039 (μPD78F8017 to μPD78F8020, μPD78F8026 to μPD78F8039)		
78K0/Kx2-A		
78K0/Kx2-C		
78K0/Lx2		
78K0/Lx3		
78K0/Lx3-M		
78K0/Dx2		
78K0/Fx2		
μPD78F0730		

(b) Option byte area (required)

This is the area for the security setting to prevent the flash memory from being read by an unauthorized person. The target device operates in accordance with the set value, as shown in Table 2-8. For the detailed settings of the option byte area, refer to the user's manual of the device.

Table 2-8. Setting and Description of Option Bytes (84H)

Target Devices	Value[Note 1]	Description
780/Kx2-L 78K0/Ix2 78K0/Fx2-L	00H	The entire internal flash memory area is erased if E1/E20 is connected.
	02H	The internal flash memory is not erased regardless of how many times authenticating the security ID code fails.[Note 2]
	03H	The entire internal flash memory area is erased if authenticating the security ID code fails.[Note 2]
78K0/Kx2 μPD78F8039 (μPD78F8017 to μPD78F8020, μPD78F8026 to μPD78F8039) 78K0/Kx2-A 78K0/Kx2-C 78K0/Lx2 78K0/Lx3 78K0/Lx3-M 78K0/Dx2 78K0/Fx2 μPD78F0730	00H	The debugger cannot be started even if E1/E20 is connected.
	02H	The internal flash memory is not erased regardless of how many times authenticating the security ID code fails.
	03H	The entire internal flash memory area is erased if authenticating the security ID code fails.
Other than the above		Setting prohibited

Notes 1. This value is to be specified in the assembler source code of a user-created program. When the program is downloaded by the debugger, the debugger changes the value as required.

2. If the value at address 83H of the option bytes is illegal, the entire internal flash memory area is erased and then the debugger starts. If authenticating the security ID code fails after the debugger starts, the entire internal flash memory area is specified to be erased.

[Setting method]

There are the following 2 methods to set option bytes to the internal flash memory. If both (1) and (2) are set, the setting of (2) has precedence.

(1) Setting method with the program

Embed option bytes to the user program. Add the code to the assembler source by referring to the following example.

Example) Setting 80H:03, 81H:00 82H:00 83H:00H 84H:02H

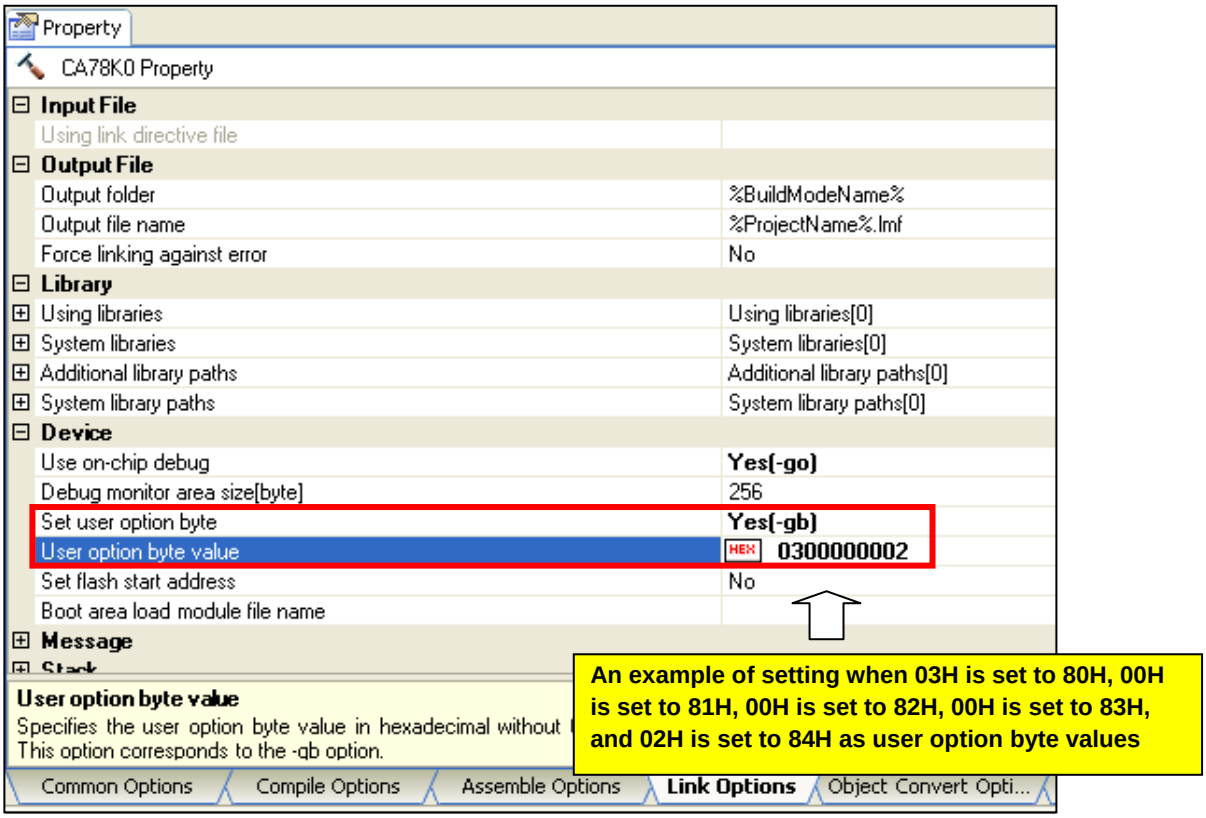
```
SSS     CSEG    AT      080H;   "SSS" is any symbol name (up to 8 characters).
                DB      03H;
                DB      00H;
                DB      00H;
                DB      02H;
```

Caution If the address 0x84 is rewritten to 0x00 in self-programming, communication becomes unavailable and debugging cannot be done. Connection cannot be made even when the debugger is restarted. In this case, use the Renesas Flash Programmer to erase it.

(2) Setting method with the CubeSuite+

You can also set the option byte area with the CubeSuite+ setting. In [Set user option byte] in the red rectangle in Figure 2-20, select [Yes (-gb)], and specify the option byte values for the addresses from 80H to 84H.

Figure 2-20. User Option Byte Setting



(c) Security ID area (required)

This setting prevents a third party from reading the content of the flash memory via the debug interface. The security ID is installed in the address 0x85-0x8E of the internal flash memory. Upon a startup the debugger, the debugger starts up only if the security ID set with the CubeSuite+ and the memory content in 0x85-0x8E match. If they do not match, the debugger operates the target device according to the option byte area setting for on-chip debugging.

If you forget the security ID, erase the flash memory, and set a new security ID.

Caution To set a security ID, use a value other than all F's (FFFFFFFFFFFFFFFFFFFF).

[Setting a security ID to the internal flash memory]

There are the following 2 methods to set a security ID to the internal flash memory. If both (1) and (2) are set, the setting of (2) has precedence.

(1) Embedding the security ID in 0x85-0x8E in the user program

Embed the security ID in 0x85-0x8E in the user program. For example, if the security ID is embedded as described below, the security ID set in the debugger is "0123456789ABCDEF1234" (not case-sensitive).

[Setting method]

(1) Setting method with the program

Embed an option byte to the user program. Add the code to the assembler source by referring to the following example.

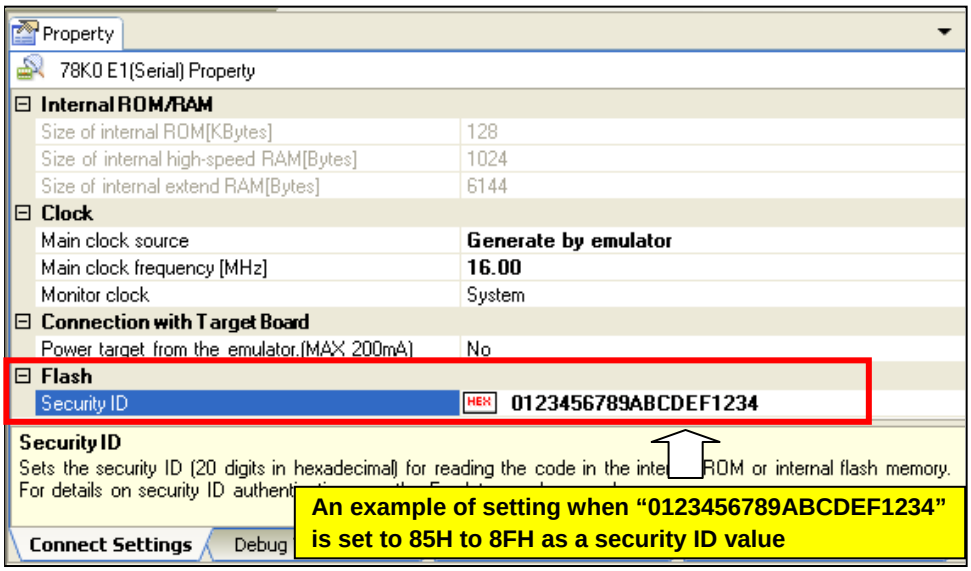
Example) Setting the security ID "0123456789ABCDEF1234" to the addresses 0x85 to 0x8E

```
SSS     CSEG    AT      85H;    "SSS" is any symbol name (up to 8 characters).
                DB      01H;
                DB      23H;
                DB      45H;
                DB      67H;
                DB      89H;
                DB      ABH;
                DB      CDH;
                DB      EFH;
                DB      12H;
                DB      34H;
```

(2) Setting method with the CubeSuite+

You can also set the security ID with the CubeSuite+ setting. In the right field of [Security ID] in the red rectangle in Figure 2-21, specify the values in hexadecimal successively for the addresses from 85H to 8FH.

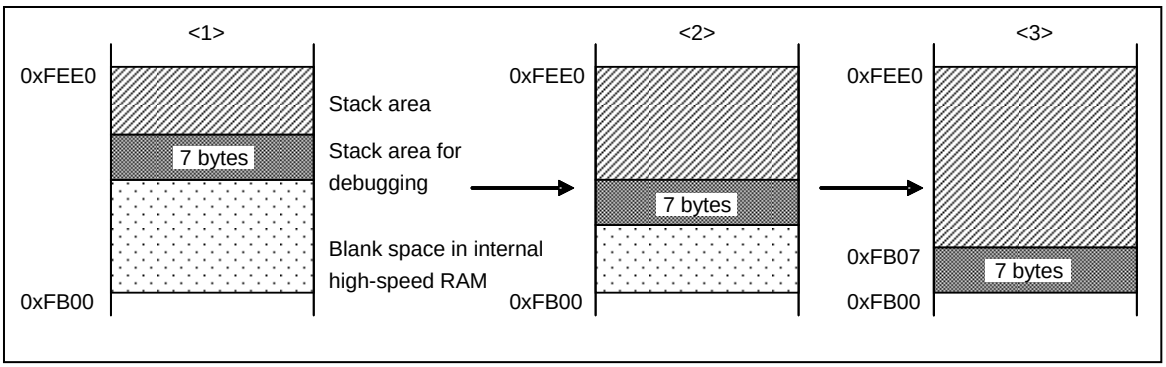
Figure 2-21. Security ID Setting



(d) Stack area for debugging (area must be secured)

This area is used as a stack area for debugging and uses 7 to 9 bytes. Because this area is located immediately below stack area, the addresses of the stack area for debugging change as the stack increases or decreases. Figure 2-22 illustrates an image of an increase in the stack area where the start address of the internal high-speed RAM is 0xFB00.

Figure 2-22. Change in Address of Stack Area for Debugging



The size of the stack area for debugging varies depending on whether software break and/or pseudo real-time RAM monitor are used. Table 2-9 shows the details.

Table 2-9. Size of the Stack Area for Debugging

Software Break	Pseudo Real-Time RAM Monitor	Size of the Stack Area for Debugging
Not used	Not used	7 bytes
Used	Not used	9 bytes
Not used	Used	
Used	Used	

[Area reservation method]

Set the stack pointer by referring to the address ranges shown in the following example.

Example) When start address of internal high-speed RAM is 0xFB00

- When software break is not used
  Within range of 0xFB07 to 0xFEE0
- When software break is used
  Within range of 0xFB09 to 0xFEE0

Also refer to (e) Area for software break below.

(e) Area for software break

This area is used for software break.

[Area reservation method]

Secure the area by referring to the following example.

```
SSS   CSEG   AT     07EH;     "SSS" is any symbol name (up to 8 characters).
      DB     0FFH,  0FFH
```

CHAPTER 3 SPECIFICATIONS

Specifications are below table.

Table 3-1. E1/E20 Specification List

Large Item	Middle Item	Small Item	Specification	
			E1	E20
Hardware Common	Target host machine		Computer equipped with a USB port OS depends on the software.	←
	User system interface		14-pin connector	←
	Host machine interface		USB2.0 (Full speed/High speed)	←
	Connection to the user system		Connection by the provided user-system interface cable	←
	Power supply function		3.3 V or 5.0 V, set in software tool, can be supplied to the user system (with current up to 200 mA)	Cannot supply power.
	Power supply for the emulator		No need (the host computer supplies power through the USB)	←
Related debugging	Break	Software break	2000 points	←
		Hardware break	One point of break before execution (not available when software break is used), one point of access break	←
		Forced break	Available	←
	Event	Number of events	Before execution: 1 point, Access: 1 point	←
		Available function	Hardware break only	←
	Trace		Unavailable	←
	Performance measurement	Measurement item	From run to break	←
		Performance	Resolution 100 μs, Max. measurement time 100 hours	←
	Pseudo realtime RAM monitor (RRM)		Available (CPU is used when monitoring)	←
	Dynamic memory modification (DMM)		Available (CPU is used when changing)	←
	Hot plug-in		Unavailable	←
	Security		10-byte ID code authentication	←
Related programming	Clock supply		16, 8, or 4 MHz clock can be supplied Clock mounted on the user system can be used	←
	Security flag setting		Available	←
	Standalone operation		Unavailable (must be connected to host machine)	←

CHAPTER 4 NOTES ON USAGE

This section describes cautions on use of the E1/E20 emulator. To use the E1/E20 properly, read the cautions thoroughly.

4.1 Lists

Table 4-1. List of Notes on Usage

No.	Item
1	Handling the device used for debugging
2	Rewriting of the flash memory during debugging
3	Software brake
4	Self-programming
5	Boot swapping during self-programming
6	Emulation in the self-programming mode
7	Stack pointer initialization fail-safe function
8	Downloading a HEX file
9	Cautions on using step-in (step execution)
10	Cautions on using the pseudo real-time RAM monitor function
11	Emulation of the POC function
12	Devices with/without the on-chip debug function
13	Cautions on reading locations with a break or reserved areas
14	Operation after a reset

4.2 Details

No.1 Handling of device that was used for debugging

Do not mount a device that was used for debugging on a mass-produced product. (Because the flash memory was rewritten during debugging and the number of rewrites of the flash memory cannot be guaranteed.) Do not embed the monitor program for debugging in a mass-produced product.

No.2 Overwriting flash memory during on-chip debugging

If the following debugger operations <1> to <8> are performed on-chip, the flash memory in the device is overwritten.

If any of the following debugger operations <1> to <8>, which involve flash memory rewriting, is performed while flash memory cannot be rewritten, the debugger automatically changes the register setting so as to enable flash memory rewriting, and restores the register setting after the operation is completed. If any of the following operations <1> to <8> is performed while flash memory rewriting is disabled or operation is performed at a voltage at which flash memory cannot be rewritten, however, the debugger outputs an error and the operation is ignored.

<1> Writing to internal flash memory
<2> Program execution after specifying or canceling software breakpoints
<3> Step-over execution, Return Out execution
<4> Come Here
<5> If Permit is selected in the Target Power off area in the Configuration dialog box, the following operations cannot be performed:
  a) Specifying, changing, or canceling hardware breakpoints
  b) Masking/unmasking internal resets
  c) Switching peripheral breakpoints
  d) Program execution
  e) Software reset (a reset performed by the debugger)
<6> Adding, changing, or deleting the monitor address when using the pseudo real-time RAM monitor function
<7> Performing operations without using breakpoints when software breakpoints are specified
<8> When the debugger is started or terminated

It takes a while from the completion of flash memory programming until control is passed to the GUI.

No.3 Software break

During program running, do not rewrite the data at the address where a software break is set. This includes self programming and rewriting to RAM. If performed, the instruction placed at the address may be invalid.

No.4 Self programming

If the space where the monitor program for debugging is rewritten by flash self programming, the debugger does not operate correctly. This also holds true when boot swapping is executed.

No.5 Boot swapping during self programming

The boot swapping function cannot be emulated. This is because boot swapping moves the memory spaces used for debugging, and thus the debug communication can no longer be performed.

No.6 Emulation in self programming mode

For self programming, the mode is switched from normal mode to self programming mode.

```
MOV PFCMD,#0A5H
MOV FLPMC,#1H       ← (1)
MOV FLPMC,#0FEH
MOV FLPMC,#1H
------------------------------------------------ From this position
CALL !08100H        ← (2)                  ↑
MOV PFCMD,#0A5H
MOV FLPMC,#0H                          Mode A1
MOV FLPMC,#0FFH
MOV FLPMC,#0H       ← (3)                  ↓
------------------------------------------------ To this position
```

The section between (2) and (3) is in self programming mode A1. Step execution or debug operations, such as stopping program execution, cannot be performed in this section. In addition, do not set a software break in the section between (1) and (3); otherwise the execution continues without a break but emulation is not performed normally.

No.7 Break function for stack pointer initialization failure

This function executes a break when an interrupt occurs or a PUSH instruction is executed while the initial setting has not been made for the stack pointer.

If the manipulation or instruction shown below is executed immediately after a reset operation, the break function for stack pointer initialization failure becomes invalid.

- Setting a software break
- Write to the stack pointer from the Register window
- Write to the flash memory from the Memory window, etc

If a software break occurs while the initial setting has not been made for the stack pointer, the message “Uninitialized Stack Pointer” is displayed on the status bar.

The subsequent operations are not performed normally, so make sure to set the SP value in the user program.

No.8 Caution on downloading a HEX file

When downloading a HEX file, do not set a filling value other than 0xFF for the option (-U) of the Renesas object converter.

No.9 Cautions when stepping into code

The value of some SFRs (special function registers) might remain unchanged while stepping into code. If the value of the SFRs does not change while stepping into code, operate the microcontroller by continuously executing the instructions instead of executing them in steps.

Stepping into code: Instructions in the user-created program are executed one by one.

Continuous execution: The user-created program is executed from the current PC value.

No.10 Cautions when using pseudo real-time RAM monitor function

<1> The user program is stopped while the pseudo real-time RAM monitor function processing is being performed.
Example)
When 78K0/Kx2: 30 × (n + 1) [μs] per n bytes when operating at 20 MHz
When 78K0/KX2-L: 120 × (n + 1) [μs] per n bytes when operating at 20 MHz

<2> If the pseudo real-time RAM monitor function is executed during a standby state, the standby state is released.

<3> If the clock signal switches to that generated by the internal low-speed oscillator or to the subsystem clock signal by the user-created program when pseudo real-time RAM monitoring is performed, pseudo real-time RAM monitoring stops. The user-created program is stopped for about 200 clock cycles by pseudo real-time RAM monitoring.

<4> Pseudo real-time RAM monitoring updates RAM information in byte units. Therefore, even if information is displayed in word units in the debugger, there will be a difference of a few ms in updating the higher and lower byte data.

<5> The pseudo real-time RAM monitor function does not operate during execution of self-programming.

No.11 Emulation of POC function

If power supply to the target system is turned off during debugging, the target device enters the reset state by the RESET_OUT pin of E1/E20. Consequently, the target device's POC function performs pseudo emulation. Note that there is an error of ±1 V between the POC detection voltage and the target system voltage. Note that the POC detection voltage might be 1 V higher than that of the target device.

No.12 Device with/without on-chip debug function

Only the devices with the on-chip debug function can be debugged. See the user's manual of the target device for checking whether it is equipped with the on-chip debug function.

To debug a device without the on-chip debug function, perform debugging by using an upper-compatible product. The following shows an example when using the 78K0/KF2.

Target Device	On-Chip Debug Function	Device for Debugging
μPD78F0544	Not equipped	μPD78F0547D (Set the IMS and IXS registers to values according to the target device.)
μPD78F0545		
μPD78F0546		
μPD78F0547		

No.13 Cautions when reading the location and the reserved area where SW/HW breaks are set

When the addresses where the software breaks are set are read, the data read from the addresses differ from the actual data. When the addresses where the hardware breaks are set are read, breaks occur.

When the addresses where are reserved for E1, the operations differ depending on the following usages by the software breaks.

- When software breaks are not used
  The data read from the addresses may differ from the actual data.
- When software breaks are used:
  The data read from the addresses may differ from the actual data, or breaks may occur.

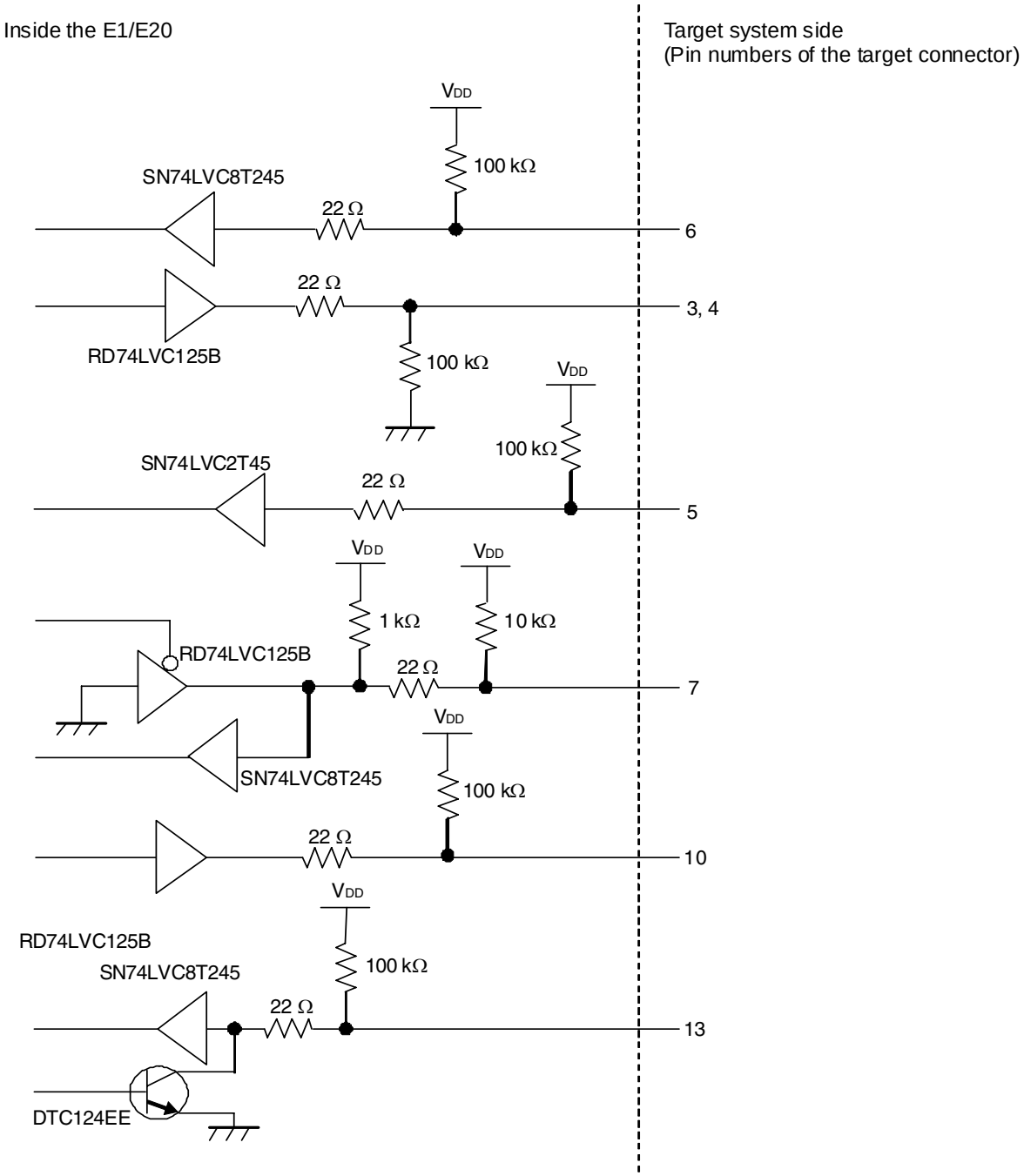
No.14 Operation after a reset

After a reset of the external pin or an internal reset, the time from the occurrence of the reset to the execution of the user program is different from the actual device operation because the debugging monitor program performs initialization processing for debugging.

APPENDIX EQUIVALENT CIRCUIT FOR E1/E20-78K0 CONNECTION

The internal equivalent circuit related to the communication interface between the E1/E20 and user system is shown below. An example of circuit connection for the user system is shown in this document. Please use it as a reference when determining parameters in board design.

Figure A-1. E1/E20 Equivalent Circuit



E1/E20 Emulator
Additional Document for User's Manual
(Notes on Connection for 78K0)

Publication Date: Aug 4, 2011 Rev.1.00

Published by: Renesas Electronics Corporation

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

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

Renesas Electronics America Inc.
2880 Scott Boulevard Santa Clara, CA 95050-2554, U.S.A.
Tel: +1-408-588-6000, Fax: +1-408-588-6130

Renesas Electronics Canada Limited
1101 Nicholson Road, Newmarket, Ontario L3Y 9C3, Canada
Tel: +1-905-898-5441, Fax: +1-905-898-3220

Renesas Electronics Europe Limited
Dukes Meadow, Millboard Road, Bourne End, Buckinghamshire, SL8 5FH, U.K
Tel: +44-1628-585-100, Fax: +44-1628-585-900

Renesas Electronics Europe GmbH
Arcadiastrasse 10, 40472 Düsseldorf, Germany
Tel: +49-211-65030, Fax: +49-211-6503-1327

Renesas Electronics (China) Co., Ltd.
7th Floor, Quantum Plaza, No.27 ZhiChunLu Haidian District, Beijing 100083, P.R.China
Tel: +86-10-8235-1155, Fax: +86-10-8235-7679

Renesas Electronics (Shanghai) Co., Ltd.
Unit 204, 205, AZIA Center, No.1233 Lujiazui Ring Rd., Pudong District, Shanghai 200120, China
Tel: +86-21-5877-1818, Fax: +86-21-6887-7858 / -7898

Renesas Electronics Hong Kong Limited
Unit 1601-1613, 16/F., Tower 2, Grand Century Place, 193 Prince Edward Road West, Mongkok, Kowloon, Hong Kong
Tel: +852-2886-9318, Fax: +852 2886-9022/9044

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

Renesas Electronics Singapore Pte. Ltd.
1 harbourFront Avenue, #06-10, keppel Bay Tower, Singapore 098632
Tel: +65-6213-0200, Fax: +65-6278-8001

Renesas Electronics Malaysia Sdn.Bhd.
Unit 906, Block B, Menara Amcorp, Amcorp Trade Centre, No. 18, Jln Persiaran Barat, 46050 Petaling Jaya, Selangor Darul Ehsan, Malaysia
Tel: +60-3-7955-9390, Fax: +60-3-7955-9510

Renesas Electronics Korea Co., Ltd.
11F., Samik Lavied' or Bldg., 720-2 Yeoksam-Dong, Kangnam-Ku, Seoul 135-080, Korea
Tel: +82-2-558-3737, Fax: +82-2-558-5141

Colophon 1.1

E1/E20 Emulator
Additional Document for User's Manual
(Notes on Connection for 78K0)

Renesas Electronics Corporation

R20UT0691EJ0100