User's Manual

E2 Emulator, E2 Emulator Lite

Additional Document for User's Manual

(Notes on Connection of RA Devices)

Supported Devices:

RA Family

Renesas Electronics
www.renesas.com

Rev.3.20 Mar 2023

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/.

Table of Contents

Terminology

Some specific words used in this user's manual are defined below.

Host machine

This means a personal computer used to control the emulator.

User system

This means a user's application system in which the MCU to be debugged is used.

User program

This means the program to be debugged.

Programming software

In this document, this indicates the Renesas Flash Programmer that can be used with the E2 or E2 Lite.

Emulator

In this document, this refers to the E2 or E2 Lite.

DLM

Device lifecycle management

1. Overview

1.1 Overview of E2 Emulator and E2 Emulator Lite

In this document, we describe ‘E2 Emulator’ as ‘E2’ and ‘E2 Emulator Lite’ as ‘E2 Lite’.

The E2 and E2 Lite are on-chip debugging emulators for Renesas' mainstream MCUs.

The E2 Lite is highly affordable development tools providing basic debugging functions. The E2 handles high-speed downloading at up to twice the rate of the E2 Lite. In addition, the E2 can supply power that is adjustable from 1.8 V to 5.0 V at 0.1-V intervals. As a development tool, the E2 allows more advanced debugging than the E2 Lite. The E2 and E2 Lite can also serve as a Flash Programmer.

1.2 Configuration of E2/E2 Lite Manuals

The E2/E2 Lite manual consists of the following.

- E2 Emulator User’s Manual
- E2 Emulator Lite User’s Manual
- E2 Emulator, E2 Emulator Lite Additional Document for User’s Manual

Be sure to read each user’s manual before using the E2 or E2 Lite.

(1) E2 emulator user’s manual

The E2 emulator user’s manual has the following contents:

— Components of the E2
— E2 hardware specification
— Connection to the E2 and the host machine and user system

(2) E2 Emulator Lite user’s manual

The E2 Emulator Lite user’s manual has the following contents:

— Components of the E2 Lite
— E2 Lite hardware specification
— Connection to the E2 Lite and the host machine and user system

(3) E2 Emulator, E2 Emulator Lite Additional Documents for User's Manual (Notes on Connection of RA Devices) (this document)

The E2 Emulator, E2 Emulator Lite Additional Documents for User's Manual (Notes on Connection of RA Devices) describes information necessary for hardware design such as connection examples and interface circuits.

(4) Renesas Flash Programmer Flash memory programming software User’s Manual

The Renesas Flash Programmer Flash memory programming software User’s Manual describes the specifications of the software and the method of operation for the Renesas Flash Programmer.

- For the debugging configuration of the E2 or E2 Lite emulator debugger, refer to the help system for the e^2 studio.

1.3 Preparation

Obtain an integrated development environment (IDE) and other required software from links at the following URL and install them on the host machine.

https://www.renesas.com/development-tools

If you are using the e^2 studio for Linux, refer to the separate guide “e2studio_setup.md”, which is included in the file for downloading at the “E2 emulator, E2 emulator Lite Linux driver” link on the Web page for the emulator. Note that this is a markdown-format English file.

1.4 Supported Devices

Table 1.1 Supported Device List

Supported Device	E2			E2 Lite		
	SWD I/F	JTAG I/F	SCI	SWD I/F	JTAG I/F	SCI
RA2 series	DBG	—	PRG	DBG	—	PRG
RA4 series	DBG	DBG*1	PRG	DBG	—	PRG
RA6 series	DBG	DBG*1	PRG	DBG	—	PRG

* DBG: Can be used for debugging, PRG: Can be used for flash programming and transition of the DLM state

Note: 1. Some MCUs do not have a JTAG interface. For details, refer to the *User's Manual: Hardware* for the given MCU.

2. Designing the User System

2.1 Connecting the Emulator with the User System

To connect the 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 *User's Manual: Hardware* for the given MCU.

2.2 Installing the Connector on the User System

Table 2.1 and Table 2.2 list the recommended connectors and user system interface cables for the emulator, respectively.

Table 2.1 Recommended Connectors

Connector	Type Number	Manufacturer	Specifications
20-pin (1.27-mm pin pitch) connector	FTSH-110-01-L-DV-K	Samtec	20-pin surface-mount technology (SMT) straight type
10-pin (1.27-mm pin pitch) connector	FTSH-105-01-L-DV-K	Samtec	10-pin SMT straight type
10-pin (1.27-mm pin pitch) connector	FTSH-105-01-L-DV* (without a marking for matching the position of the connector; keying shroud)	Samtec	10-pin SMT straight type

Note: When using a connector without a guide marking (keying-shroud type), take care with regard to the direction for insertion of the cable.

Table 2.2 User System Interface Cables

Cable Type	Type Number	E2	E2 Lite
20-pin to 20-pin cable* (for the 20-pin (1.27-mm pin pitch) connector)	RTE0T00020KCAC0000J	Comes with the product	Separately sold
20-pin to 10-pin cable (for the 20-pin (1.27-mm pin pitch) connector)	RTE0T00020KCAC1000J	Separately sold	Separately sold

Note: The 20-pin to 20-pin cable can be connected to the guideless 10-pin (1.27-mm pin pitch) connector; when doing so, however, check the pin assignments and take care with regard to the direction for insertion of the cable.

Only connect the emulator after confirming that there are no mismatches of alignment on the user system port connector. Incorrect connection will result in the host machine, the emulator, and the user system emitting smoke or catching fire.

2.2.1 Connecting the User System Interface Cable to the 20-Pin Connector

Figure 2.1 shows how to connect the user system interface cable to the 20-pin connector.

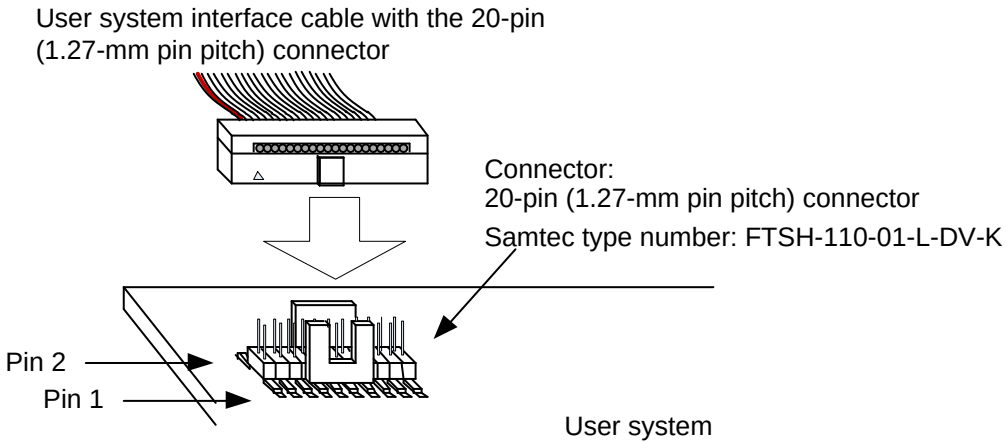


Figure 2.1 Connecting the User System Interface Cable to the 20-Pin Connector

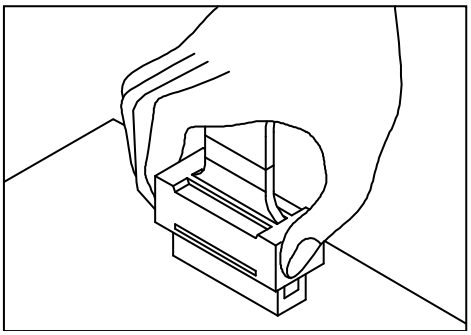
CAUTION

Notes on connector insertion and removal:

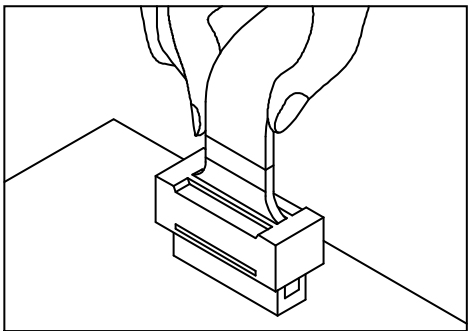
When connecting or disconnecting the user-system interface cable and the emulator or user system, grasp the connector cover at the end of the cable. Pulling the cable itself will damage the wiring.

Also, be aware that the user-system interface cable has the direction in which it must be inserted. If the cable is connected in the wrong direction, it may be damaged.

Correct example



Incorrect example



2.3 Pin Assignments of the Connector on the User System

2.3.1 20-Pin and 10-Pin Connector Specifications

Figure 2.2 shows the specifications of the 20-pin and 10-pin (1.27-mm pitch) connectors.

Table 2.3 and Table 2.4 show the pin assignments for the SWD and JTAG interface connections, respectively.

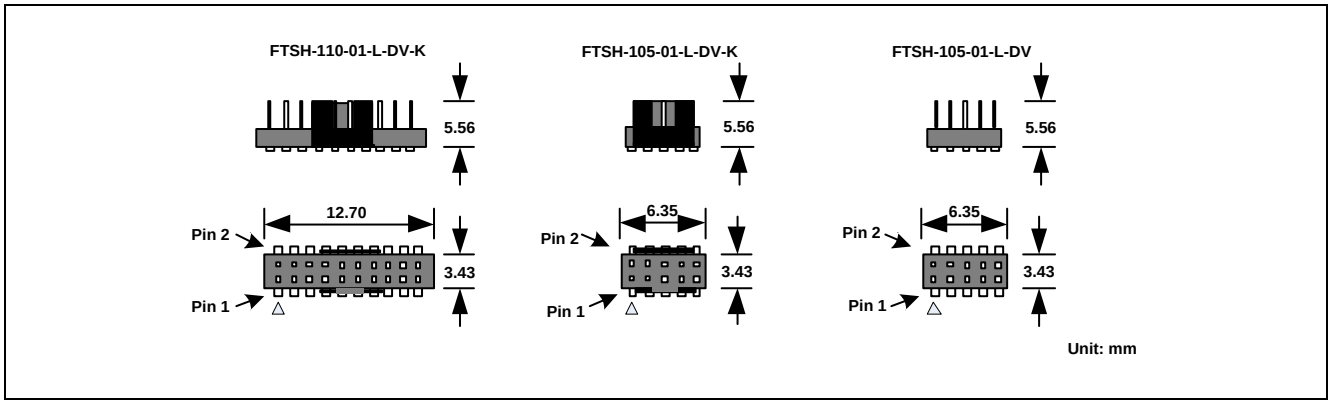


Figure 2.2 20-Pin and 10-Pin Connector Specifications

Table 2.3 Pin Assignments for SWD Interface Connection

Pin No.	Signal	Direction*1	Note
1	VCC	–	Power supply
2	SWDIO	I/O	For debugging communications
3	GND	–	
4	SWCLK	Input	Clock for debugging communications
	MD		For setting the operating mode
5	GND	–	
6	TxD9	Output	For programming flash memory and transition of the DLM state
7	NC	–	
8	RxD9	Input	For programming flash memory and transition of the DLM state
9	UCON	–	Connect this signal to ground on the user system. It is used to confirm the connection between the emulator and user system.
10	RES#	I/O	User system reset
11*2	NC	–	
12*2	NC	–	
13*2	NC	–	
14*2	NC	–	
15*2	GND	–	
16*2	NC	–	
17*2	GND	–	
18*2	NC	–	
19*2	GND	–	
20*2	NC	–	

Notes: 1. Input to or output from the user system.
"Input" refers to input from the emulator to the user system and "output" refers to output from the user system to the emulator.
2. If a 10-pin connector is mounted on the user system, pins 11 to 20 are not used.

Table 2.4 Pin Assignments for JTAG Interface Connection

Pin No.	Signal	Direction*1	Note
1	VCC	–	Power supply
2	TMS	Input	For debugging communications
3	GND	–	
4	TCK	Input	Clock for debugging communications
	MD		For setting the operating mode
5	GND	–	
6	TDO/TxD9	Output	For debugging communications, programming flash memory, and transition of the DLM state
7	NC	–	
8	TDI/RxD9	Input	For debugging communications, programming flash memory, and transition of the DLM state
9	UCON	–	Connect this signal to ground on the user system. It is used to confirm the connection between the emulator and user system.
10	RES	I/O	User system reset
11*2	NC	–	
12*2	NC	–	
13*2	NC	–	
14*2	NC	–	
15*2	GND	–	
16*2	NC	–	
17*2	GND	–	
18*2	NC	–	
19*2	GND	–	
20*2	NC	–	

Notes: 1. Input to or output from the user system.
"Input" refers to input from the emulator to the user system and "output" refers to output from the user system to the emulator.
2. If a 10-pin connector is mounted on the user system, pins 11 to 20 are not used.

2.4 Recommended Circuit between the Connector and the MCU

This section shows recommended circuits for connection between the connector and the MCU. For details on the handling of signals, refer to section 2.5, Notes on Connection.

2.4.1 SWD Interface Connection

Figure 2.3 shows a recommended circuit for connection through the SWD interfaces.

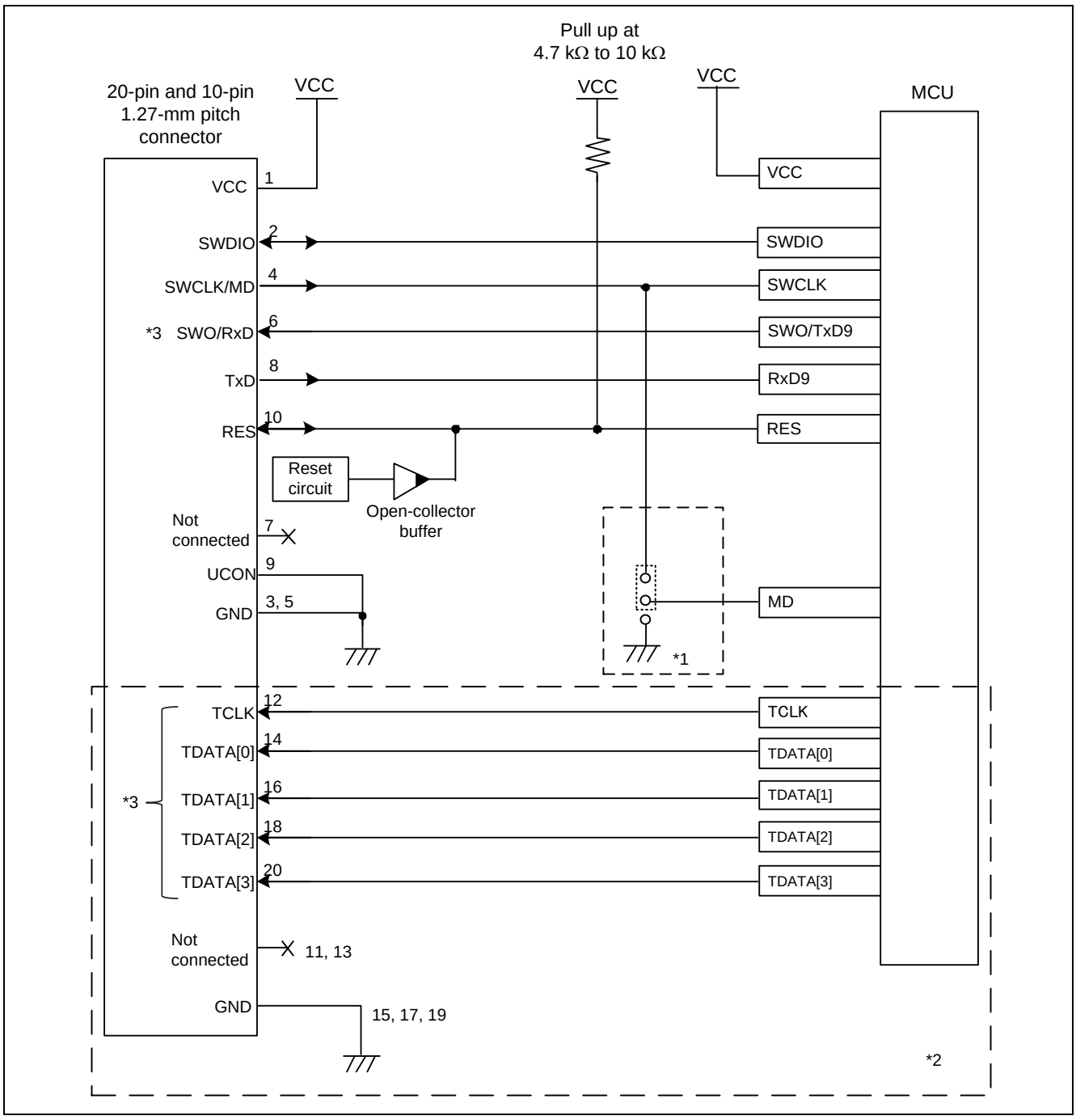


Figure 2.3 Example of a Connection through SWD Interfaces

Notes:
1. The connection depends on the device. Refer to the *User's Manual: Hardware* for the given MCU of the RA family. For the MCUs for which this connection is recommended, connect the MD pin to SWCLK/MD of the emulator. When the MD pin is not connected to SWCLK/MD of the emulator, a circuit to connect the pin must be configured on the user system. For details on handling of the MD pin, refer to section 2.5, Notes on Connection.
2. If a 10-pin connector is mounted on the user system, pins 11 to 20 are not used.
3. The E2 emulator Lite does not support a trace output facility.
   The E2 emulator only supports trace output through the SWO pin.

2.4.2 JTAG Interface Connection

Figure 2.4 shows a recommended circuit for connection through the JTAG interfaces. The E2 Lite does not support JTAG interface connection.

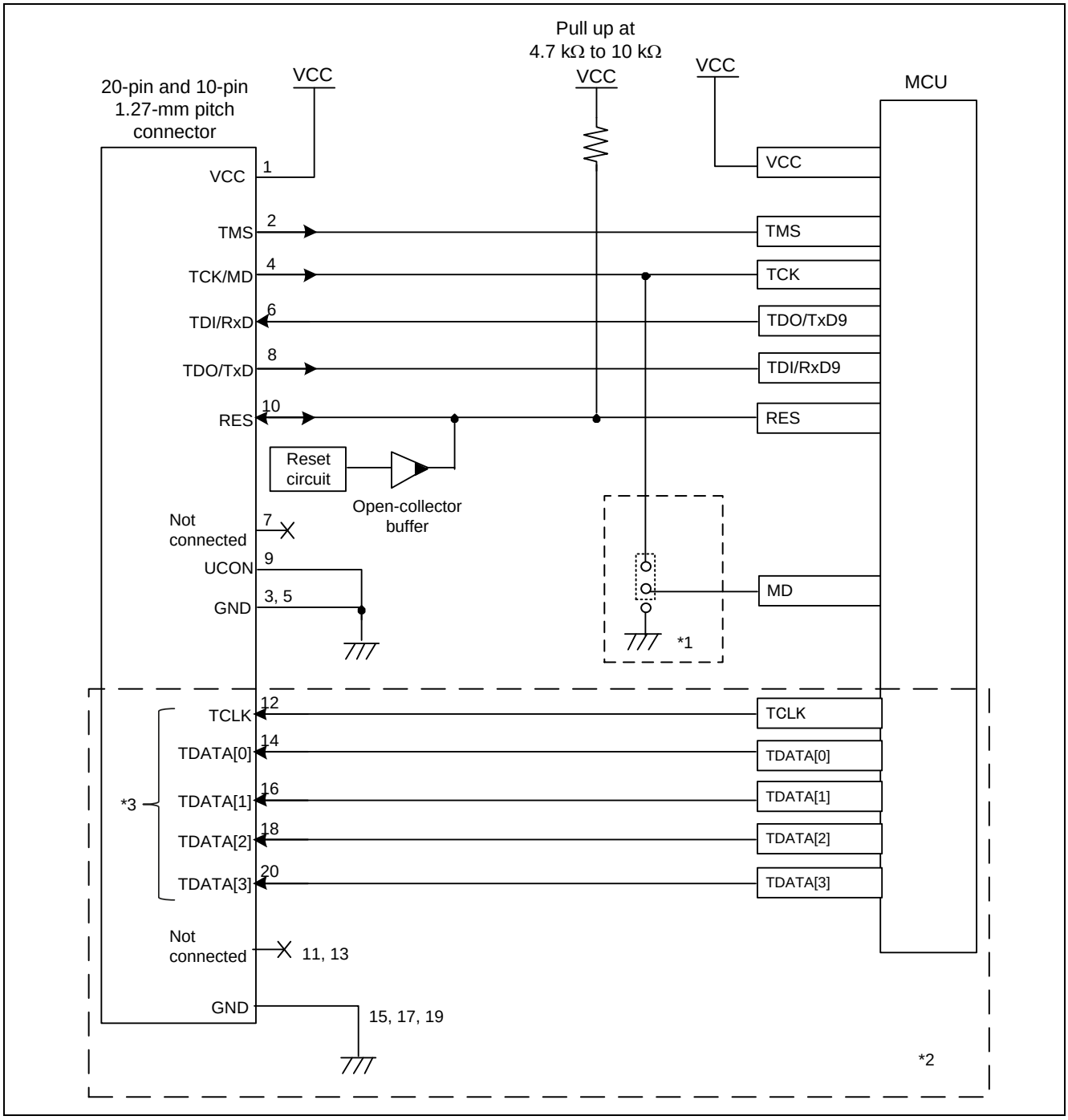


Figure 2.4 Example of a Connection through JTAG Interfaces

Notes:
1. The connection depends on the device. Refer to the *User's Manual: Hardware* for the given MCU of the RA family. For the MCUs for which this connection is recommended, connect the MD pin to TCK/MD of the emulator. When the MD pin is not connected to TCK/MD of the emulator, a circuit to connect the pin must be configured on the user system. For details on handling of the MD pin, refer to section 2.5, Notes on Connection.
2. If a 10-pin connector is mounted on the user system, pins 11 to 20 are not used.
3. The E2 emulator does not support a trace output facility.

2.4.3 SCI Connection

Figure 2.5 shows a recommended circuit for connection to the Renesas Flash Programmer. This circuit is for use in cases where the only operation is writing a program into the flash memory without any debugging operations.

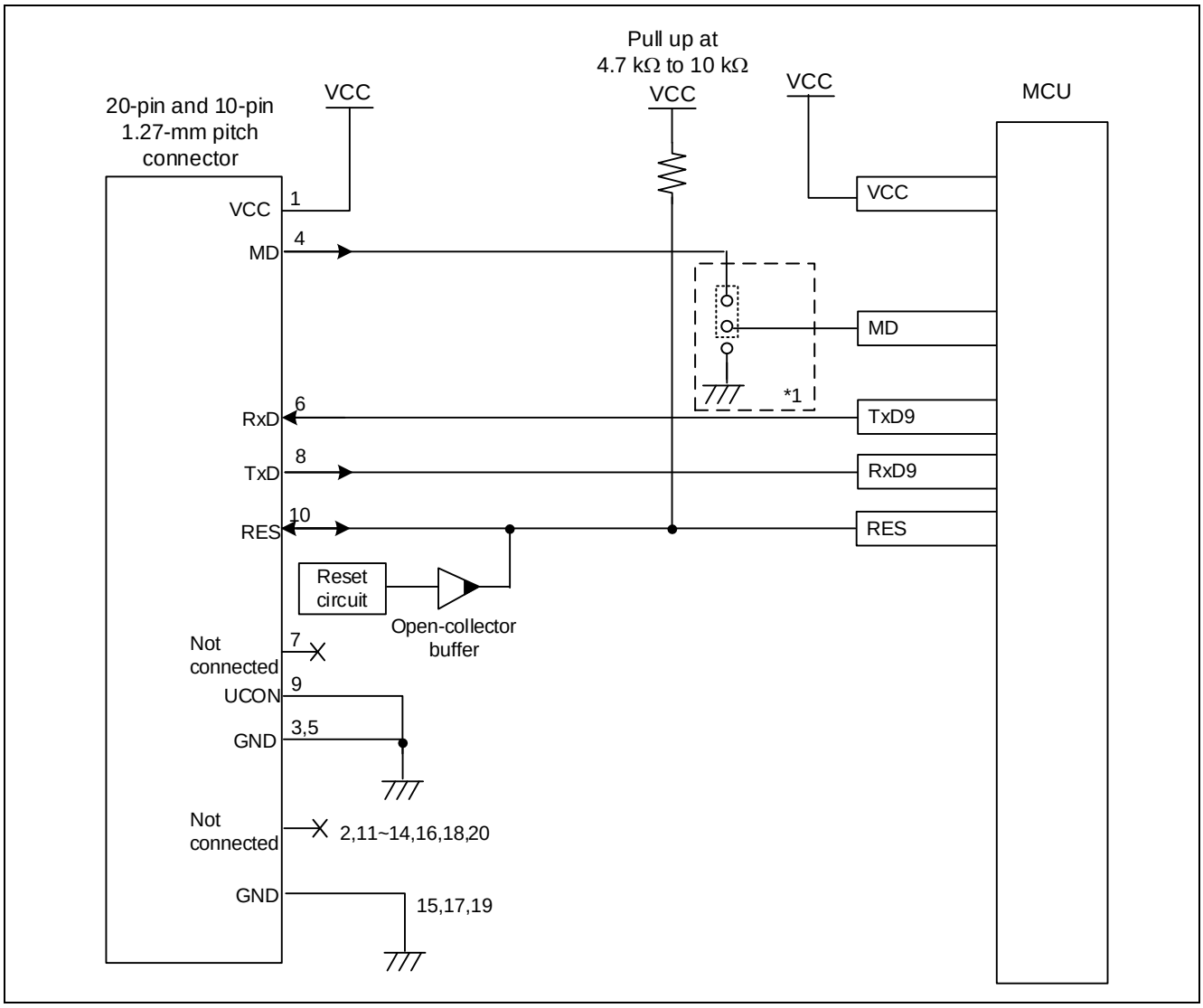


Figure 2.5 Example of a Connection through SCIs

Note: 1. The connection depends on the device. Refer to the *User's Manual: Hardware* for the given MCU of the RA family. For the MCUs for which this connection is recommended, connect the MD pin to MD of the emulator. When the MD pin is not connected to MD of the emulator, a circuit to connect the pin must be configured on the user system. For details on handling of the MD pin, refer to section 2.5, Notes on Connection.

2.5 Notes on Connection

Wiring patterns between the connector and the MCU must be as short as possible (within 50 mm is recommended). Do not connect the signal lines between the connector and MCU to other signal lines on the board.

For the handling of pins while the emulator is not in use, refer to the *User's Manual: Hardware* for the given MCU.

2.5.1 RES Pin

The emulator uses the RES pin.

If the user system includes a user logic reset circuit, the output signal from the reset circuit must be connected to the RES pin of the connector via an open-collector buffer as shown below. If there is no reset circuit, the RES pin from the connector must be directly connected to the RES pin of the MCU.

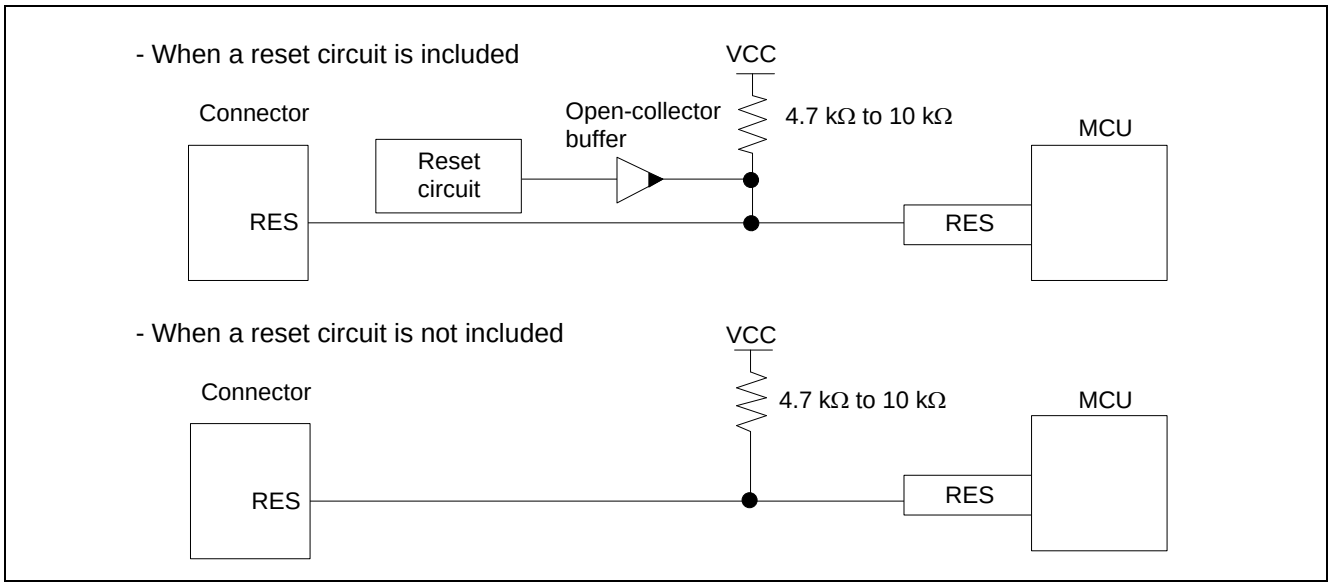


Figure 2.6 Connection of the RES Pin

The voltage on the RES line must satisfy the electrical characteristics of the device and rise in no more than 9 ms.

2.5.2 MD Pin

Refer to the *User's Manual: Hardware* for the given MCU of the RA family and confirm whether connection between the MD pin of the MCU and SWCLK/TCK/MD of the emulator is recommended for the MCU.

When the SWD boot mode is not supported and the CPU core is an M33: Make a wired connection between SWCLK and MD. The emulator uses the MD pin (see Figure 2.7).

When the SWD boot mode is supported and the CPU core is an M33: SWCLK cannot be connected to MD. Place a switch to switch the level on the MD pin as required on the user system (see Figure 2.8). When the MD pin is to be used in SCI or USB boot mode, the MD pin must be handled so that it is switched to the low level.

When the SWD boot mode is not supported and the CPU core is not an M33: Either of the circuit configurations shown in Figure 2.7 and Figure 2.8 may be used.

When the emulator is not to be used and the MD pin is to be used with SCI or USB boot mode: Use the circuit configuration shown in Figure 2.8.

Note that the circuit for the MD pin of the MCU includes a pull-up resistor.

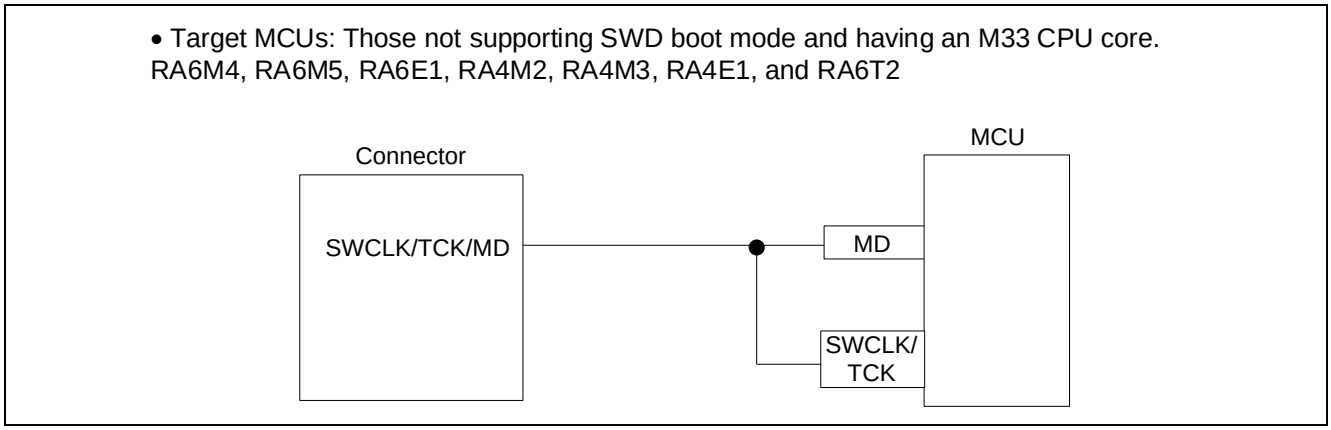


Figure 2.7 Connection of the MCUs that Do Not Support the SWD Boot Mode and Have an M33 CPU Core

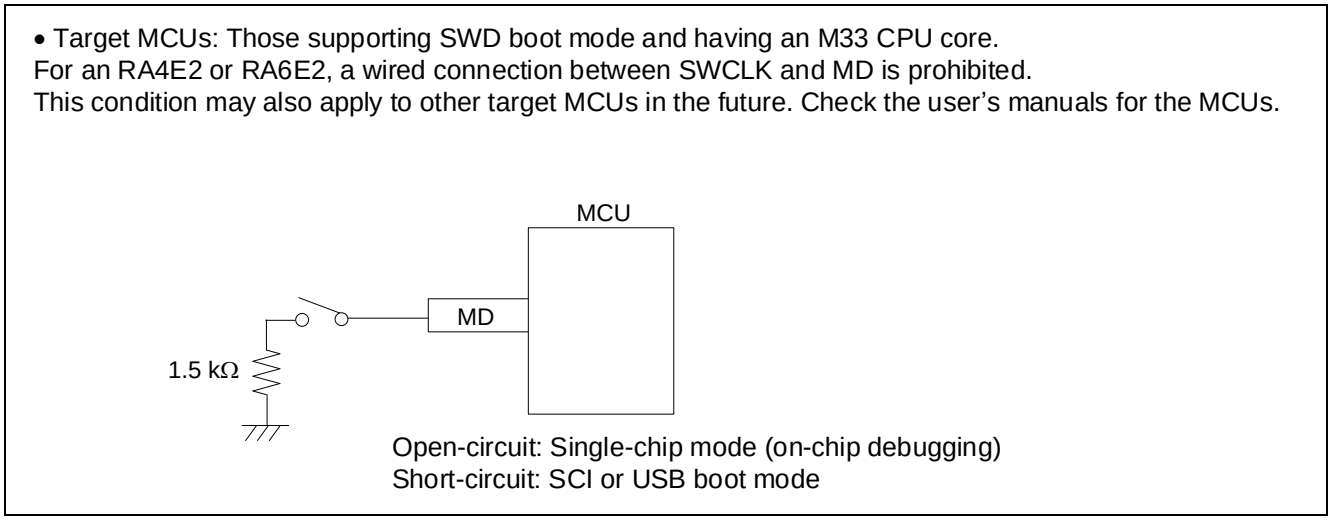


Figure 2.8 Connection of the MCUs that Support the SWD Boot Mode and Have an M33 CPU Core

2.5.3 GND

The pins of the connector marked "GND" must be at the same ground level as the VSS pin of the MCU.

2.5.4 VCC

Connect the VCC of the connector to the VCC (power supply) of the user system.

Use the emulator within the power supply voltage of 1.8 V to 5.5 V and within the operating voltage range of the MCU.

When power is supplied to the user system from other than the emulator, the E2/E2 Lite consumes the power supply for the last output and first input buffers of the emulator.

- E2: 3.3 V: approximately 20 mA, 5.0 V: approximately 40 mA
- E2 Lite: 3.3 V: approximately 20 mA, 5.0 V: approximately 40 mA

The E2/E2 Lite can supply power to a simple evaluation system.

- E2: Can supply power of 1.8 V to 5.0 V, up to 200 mA.
- E2 Lite: Can supply power of 3.3 V, up to 200 mA.

When using the power supply function of the E2 or E2 Lite, check the voltage that is actually being supplied to the user system since this depends on the environment.

Power supply from the E2/E2 Lite depends on the quality of the USB power supply of the host machine, and as such, precision is not guaranteed. When writing a program that requires reliability, do not use the power supply function of the E2/E2 Lite. Use a stable, separate power supply for the user system. As the software when writing a program in a mass-production process, use the Renesas Flash Programmer.

For details on the programming software, refer to the following:

Renesas Flash Programmer: https://www.renesas.com/RFP

When the MCU is changed to low power mode, the internal debugging circuit continues to run. This leads to the MCU drawing more electric current than is listed in the DC characteristics of the target MCU.

WARNING

Warning for Turning the Power On/Off:

When supplying power, ensure that there are no shorts between Vcc and GND. Only connect the E2/E2 Lite after confirming that there are no mismatches of alignment on the user system port connector. Incorrect connection will result in the host machine, the E2/E2 Lite, and the user system emitting smoke or catching fire.

2.5.5 RxD9 and TxD9 Pins (Flash Programming via an SCI)

When flash memory is programmed via an SCI, the RxD9 and TxD9 pins must be connected to the emulator. For MCUs in which the RxD9 and TxD9 pins can be allocated to multiple pins, check the *User's Manual: Hardware* for the given MCU to confirm which of the pins is used in boot mode.

2.6 Internal Circuits of the Emulator

2.6.1 Internal Circuits of the E2

Figure 2.9 and Figure 2.10 respectively show the internal circuits of product revisions C and D of the E2.

The alphabet at the end of the serial No. written on the E2 main unit indicates the product revision.

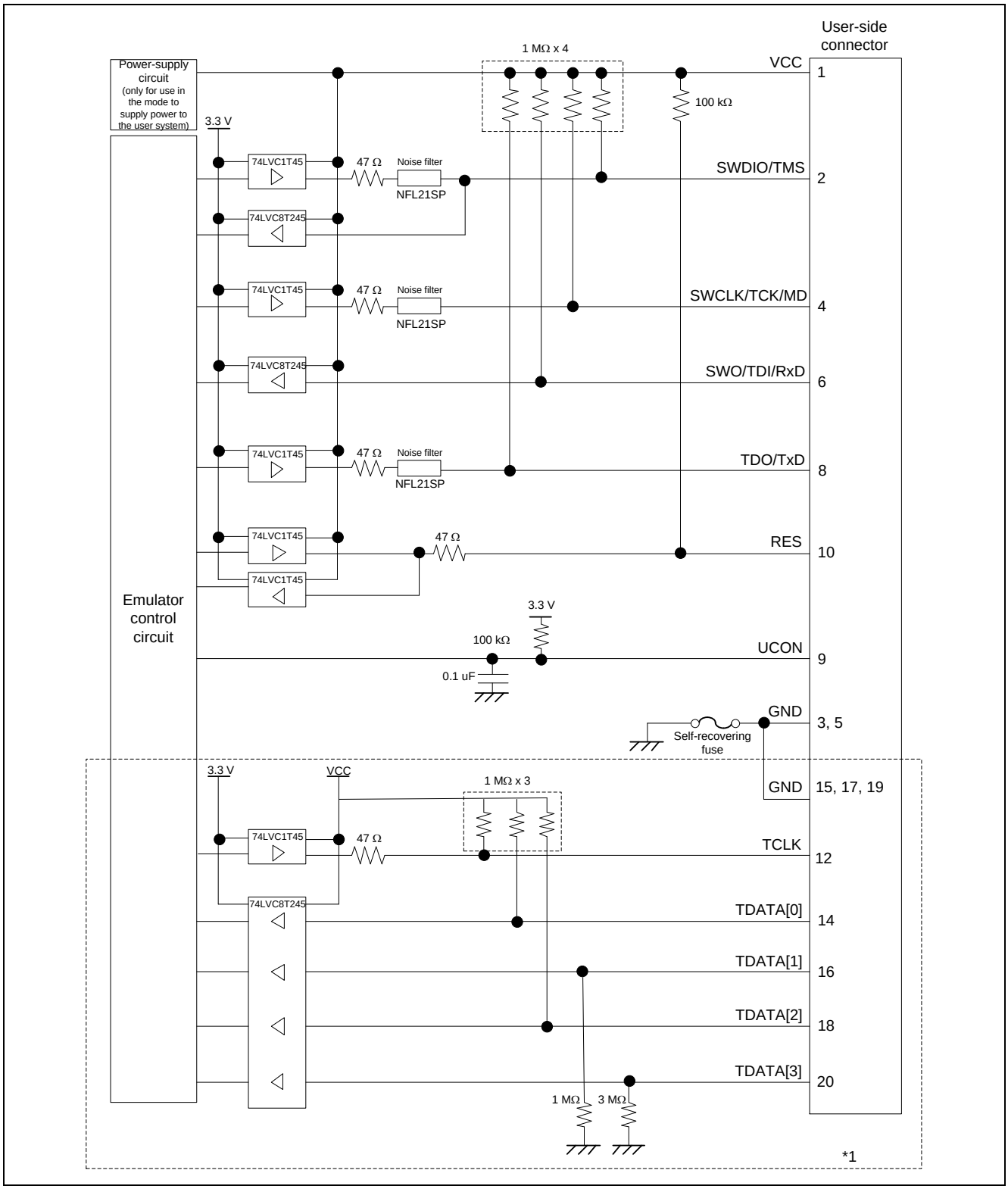


Figure 2.9 Internal Circuits of the E2 (Rev. C)

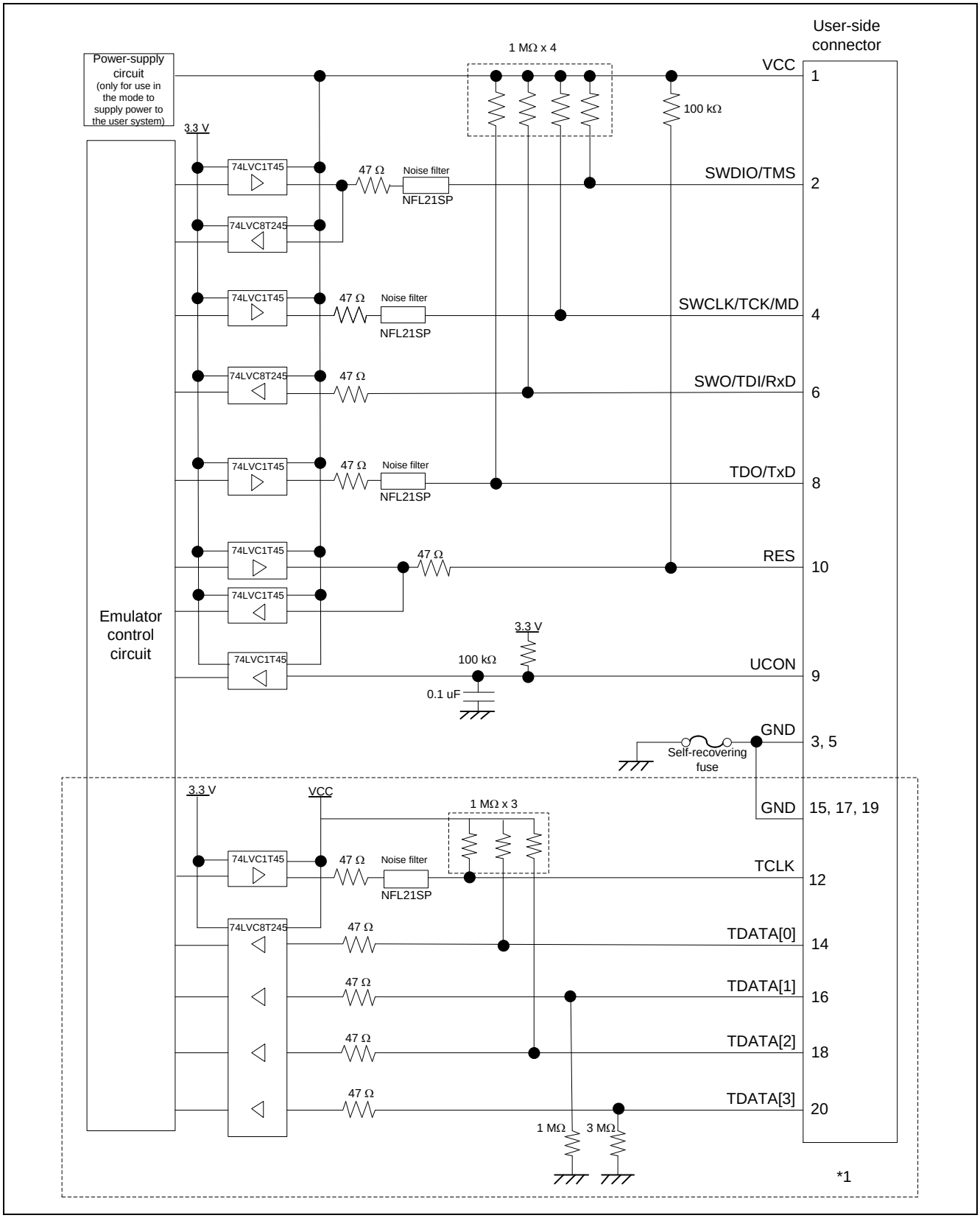


Figure 2.10 Internal Circuits of the E2 (Rev. D)

Note 1: If a 10-pin connector is mounted on the user system, pins 11 to 20 are not used.

2.6.2 Internal Circuits of the E2 Lite

Figure 2.11 shows the internal circuits of the E2 Lite.

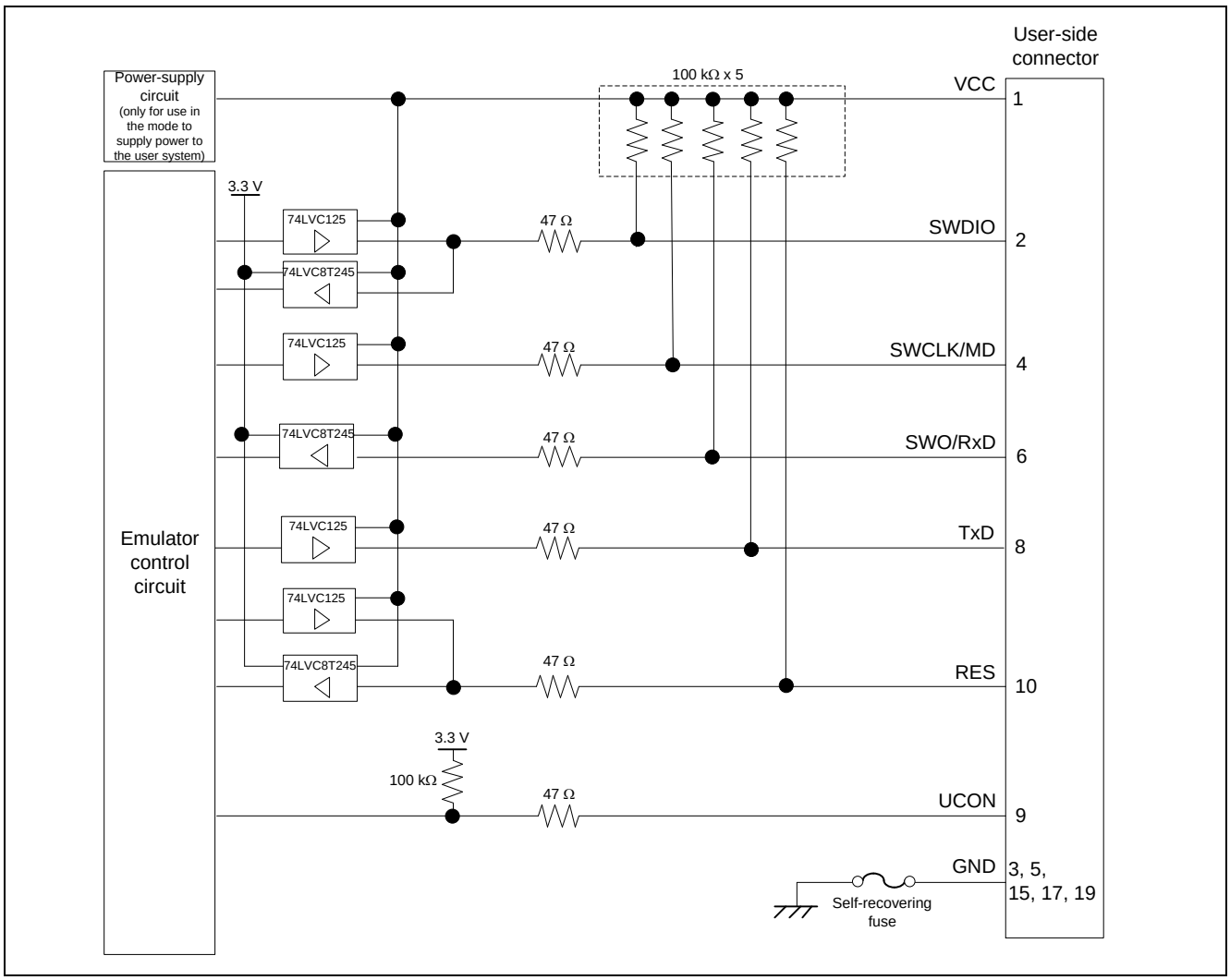


Figure 2.11 Internal Circuits of the E2 Lite

3. Notes on Usage

3.1 Turning the Power On/Off

Turn the power of the emulator and the user system following the procedure below.

3.1.1 When a Separate Power Supply is Used for the User System

<When using the emulator>

(1) Check that the power is off.
Check that the user system is turned off.
(2) Connect the user system.
Connect the emulator and the user system with a user-system interface cable.
(3) Connect the host machine and turn on the emulator.
Connect the emulator and the host machine with a USB interface cable. The E2/E2 Lite is turned on by connecting the USB interface cable.
(4) Launch the emulator debugger or programming software.
Launch the emulator debugger or programming software.
(5) Turn on the user system.
Turn on the user system.
(6) Connect the emulator debugger or programming software to the emulator.
Connections may vary depending on software.

<When finished using the emulator>

(1) Disconnect the emulator from the emulator debugger or programming software.
Disconnections may vary depending on software.
(2) Turn off the user system.
Turn off the user system.
(3) Close the emulator debugger or programming software.
Close the emulator debugger or programming software.
(4) Turn off the emulator and disconnect the emulator.
Disconnect the USB interface cable from the emulator. The E2/E2 Lite is turned off by disconnecting from the USB interface cable.
(5) Disconnect the user system.
Disconnect the user-system interface cable from the user system.

CAUTION

Notes on the User System Power Supply:

While the power of the user system is on, do not turn off the host machine or unplug the USB interface cable.
The user system may be damaged due to leakages current.

3.1.2 When Power is Supplied to the User System from the Emulator

<When using the emulator>

(1) Connect the user system.
Connect the emulator and the user system with a user-system interface cable.
(2) Connect the host machine and turn on the emulator.
Connect the emulator and the host machine with a USB interface cable, then turn on the emulator.
(3) Launch the emulator debugger.
Launch the emulator debugger and select the setting of power supply to the user system.
Under [Power] on the [Connection Settings] tabbed page, select [Yes] for [Power Target From The Emulator (MAX 200mA)]. Refer to section 3.3, Notes on Using the Emulator Debugger, for how to open the [Debug Configurations] window.

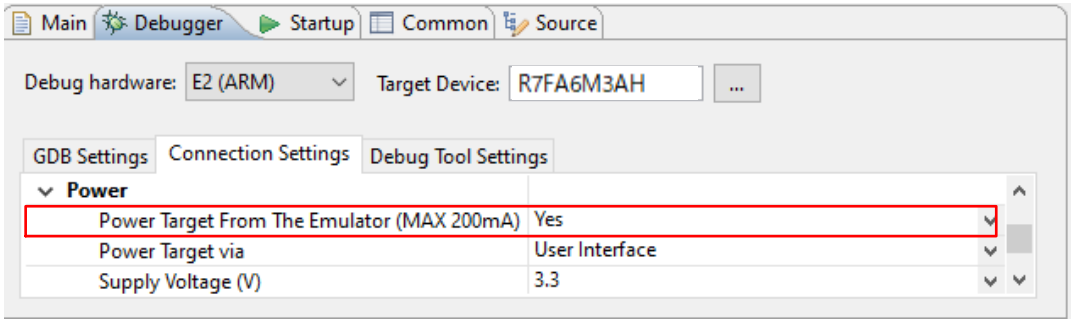


Figure 3.1 Setting for Supplying Power

(4) Connect the emulator debugger or programming software to the emulator.
Connections may vary depending on software.

<When finished using the emulator>

(1) Disconnect the emulator from the emulator debugger or programming software.
Disconnections may vary depending on software.
(2) Close the emulator debugger or programming software.
Close the emulator debugger or programming software.
(3) Turn off the emulator and disconnect the emulator.
Disconnect the USB interface cable from the emulator, then turn off the emulator.
(4) Disconnect the user system.
Disconnect the user-system interface cable from the user system.

3.2 Power Supply Function of the E2/E2 Lite

The E2/E2 Lite can supply power to a simple evaluation system.

- E2: Can supply power of 1.8 V to 5.0 V, up to 200 mA.
- E2 Lite: Can supply power of 3.3 V, up to 200 mA.

When using the power supply function of the E2 or E2 Lite, check the voltage that is actually being supplied to the user system since this depends on the environment.

3.3 Notes on Using the Emulator Debugger

This section describes how to set the [Debug Configurations] window of the e² studio. To open the [Debug Configurations] window, click on [Run] → [Debug Configurations…] or the downward-pointing arrow next to the icon→ [Debug Configurations…].

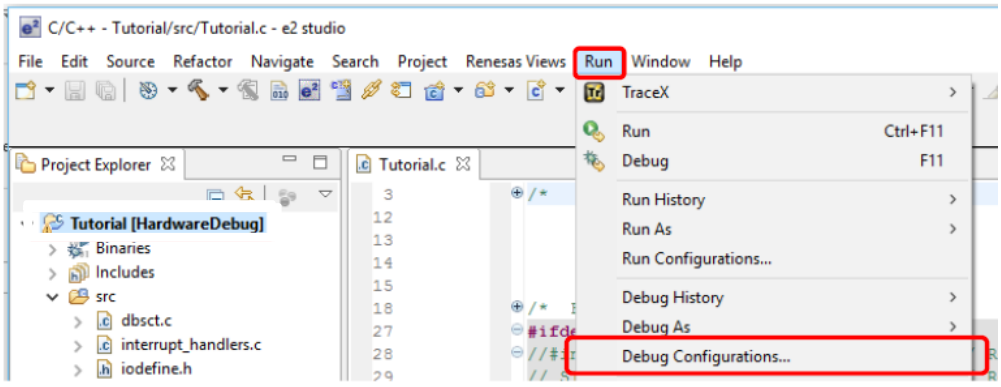


Figure 3.2 Opening the [Debug Configurations] Window

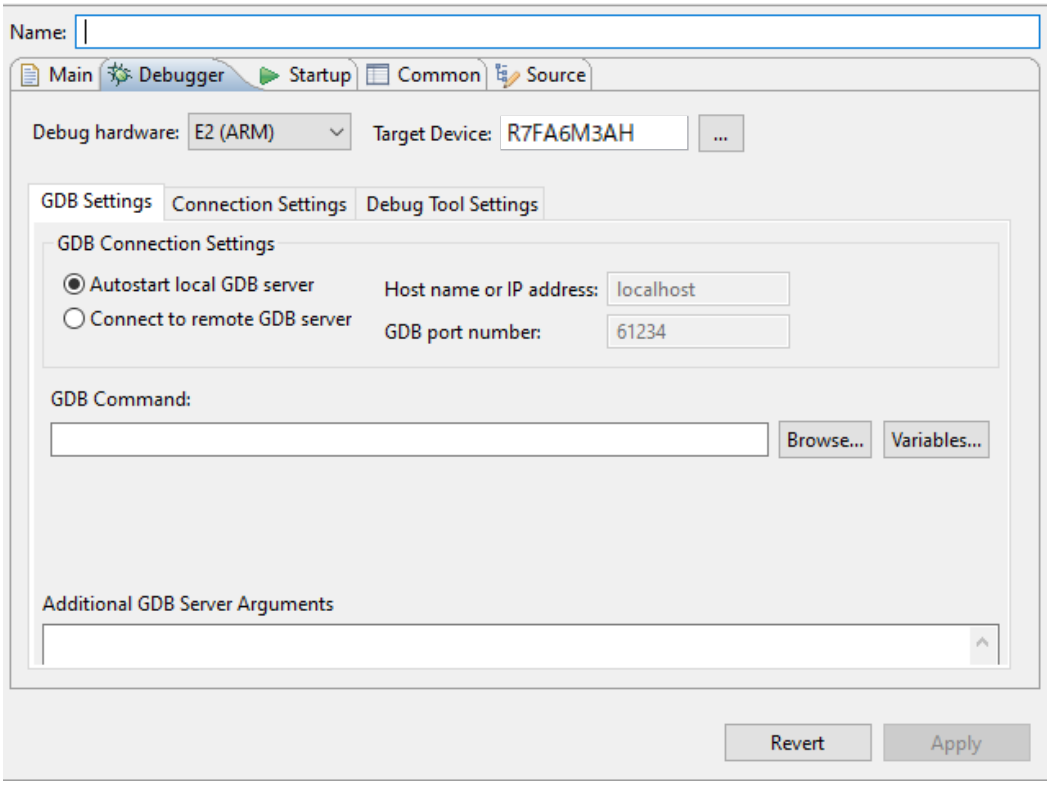


Figure 3.3 [Debug Configurations] Window

3.3.1 Notes on Connecting the Emulator Debugger

(1) Reset state

Under [Connection] on the [Connection Settings] tabbed page, be sure to select [Yes] for [Hold reset during connect].

When [Yes] is selected, during connection to the emulator debugger, the emulator maintains the low-level output on the RES# pin of the MCU and places the MCU in the OCD mode. However, to start the operation of the built-in debugging circuits of the MCU, the emulator releases the reset for about 50 msec while it is connected, and the user program automatically runs during that period.

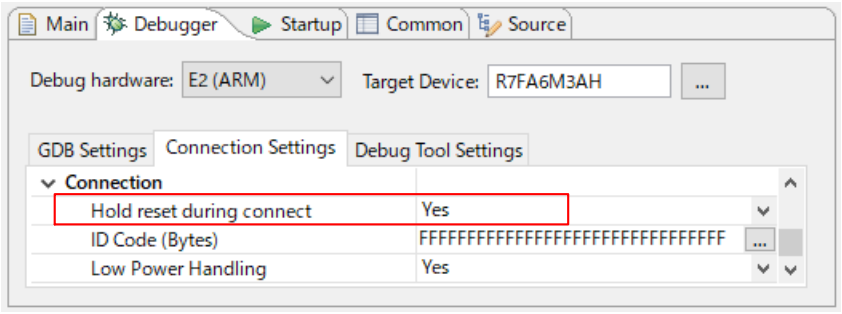


Figure 3.4 Setting of [Hold reset during connect]

(2) Startup mode

When the emulator debugger is connected, handle the pins on the user system so that the startup mode of the MCU is single-chip mode.

In single-chip mode, the MD pin is at the high level. Correct connection of the emulator debugger is not possible in SCI or USB boot mode.

(3) Debugging after rewriting ID code

If the ID code (OSIS register) has been rewritten, enter the new ID code.

Here, the ID code to be entered means the value to be entered in [ID Code (Bytes)] under [Connection] on the [Connection Settings] tabbed page.

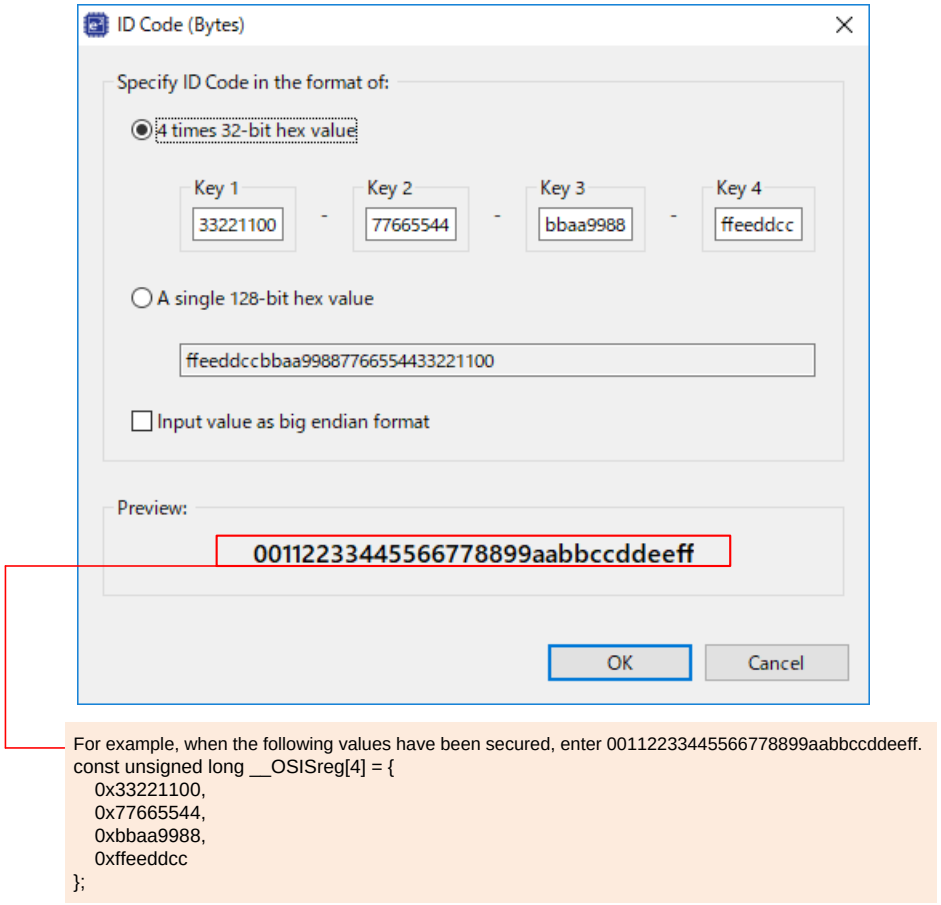


Figure 3.5 Setting of [ID Code (Bytes)]

Devices incorporating DLM facilities, such as those of the RA6M4 group, do not have ID code authentication.

(4) Entering the ALeRASE command for ID code

If an ALeRASE command (FFFFFFFFFFFFFFFFFF45534152654C41) is entered in [ID Code (Bytes)] under [Connection] on the [Connection Settings] tabbed page, the code flash memory and option-setting memory will be erased when the emulator debugger is connected.

For details on entering the ALeRASE command, refer to E2/E2 Lite (RA) Connection Settings in the help system for the e^2 studio.

For the conditions under which the ALeRASE command is usable, refer to the *User's Manual: Hardware* for the given MCU. If the ALeRASE command is entered then the emulator debugger is connected while the command is not usable on the MCU, an error message “Failed to erase all flash memory by the ID for erasing all flash memory.” is displayed and the connection process is suspended.

Since devices such as those of the RA6M4 group have DLM facilities and so do not have ID code authentication, the ALeRASE command cannot be used.

(5) Connection speed

The speed for the connection of the emulator to the target board is specified within the following upper-limit values.

— JTAG (E2 only): 25000 kHz
— SWD (with an E2): 25000 kHz
— SWD (with an E2 Lite): 6000 kHz

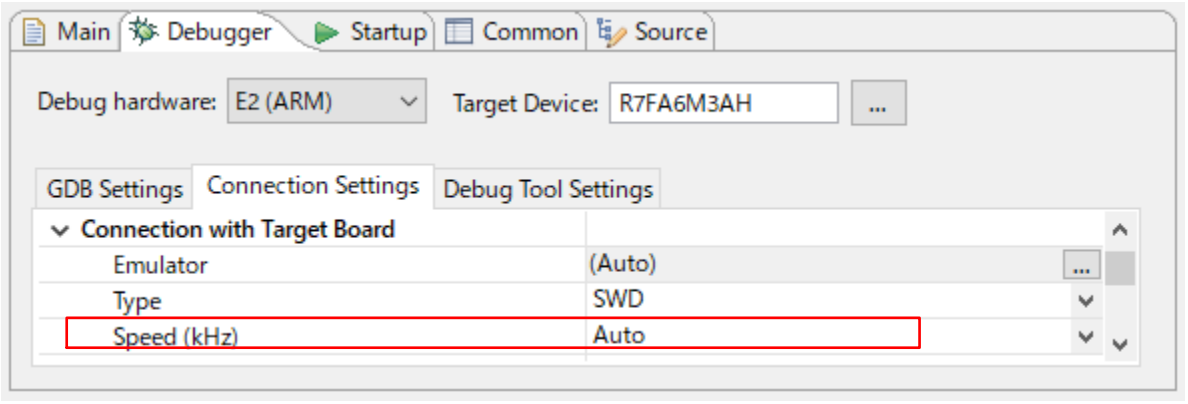


Figure 3.6 Setting the Connection Speed

When [Auto] is selected, the maximum connectable speed will automatically be set.

3.3.2 Notes on a Debugging Operation that Involves Reprogramming of Flash Memory

A “debugging operation that involves reprogramming of flash memory” refers to the following operations of the emulator debugger.

- Downloading data to flash memory
- Using software break functions in flash memory
  (a) Setting and canceling breakpoints
  (b) Executing or step-executing programs from a breakpoint
  (c) Using the “Run to Line” function from the state where a break was set

(1) Program for reprogramming flash memory

Since the emulator debugger enables a debugging operation that involves reprogramming of flash memory, the emulator writes the program for reprogramming flash memory to the on-chip SRAM and executes the program to reprogram the flash memory. After the flash memory has been reprogrammed, the emulator debugger restores the on-chip SRAM to its initial state.

(2) Destination for allocation of the program for reprogramming flash memory

By default, the program for reprogramming the flash memory is allocated to the 5-Kbyte space from the address where the SRAM0 area starts (or the address where the SRAMHS area starts for devices that do not include an SRAM0 area). If the default allocation destination is not available due to the security settings or DMAC/DTC transfer*, enter the start address of an available space in the on-chip RAM in units of 1000h bytes against [Work RAM Start Address] under [Flash] on the [Debug Tool Settings] tabbed page for the emulator debugger.

Note: The DMAC or DTC will continue to operate even during a break. Take care that a transfer source or destination for the DMAC or DTC is not within the address range of the working RAM where the program is to be allocated.

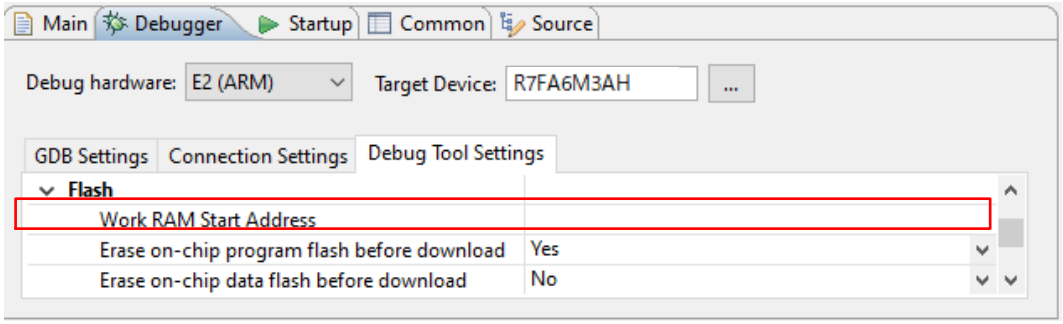


Figure 3.7 Setting of [Work RAM Start Address]

Do not deploy the program for reprogramming flash memory when the device incorporates the TrustZone® facility (for example, those of the RA6M4 group) and the emulator debugger is connected to the device with debugging-access level DBG1.

(3) Interrupts and resets during execution of the program for reprogramming flash memory

Interrupts other than non-maskable interrupts are to be masked while the program for reprogramming flash memory is being executed. Also, so that the program for reprogramming flash memory is correctly executed, all interrupt source flags which have been set before executing the program are cleared.

If a non-maskable interrupt occurs, the emulator continues running the program for reprogramming flash memory. If a reset occurs while the program for reprogramming flash memory is being executed, the emulator shows an error message and stops processing. Since doing so may damage the contents of flash memory, do not apply a reset while the program is running.

(4) Conditions for downloading data to flash memory being available

When the MCU satisfies all the following conditions, the downloading of data to flash memory can proceed.

(a) The code flash memory of the MCU is in read mode.

(b) The frequency of the system clock (ICLK) of the MCU is 1 MHz or higher.*

Note: For [Clock] in the [Connection Settings] tabbed page of the emulator debugger, when [Yes] is selected for [Permit Clock Source change on writing on-chip Flash Memory], condition (b) can be excluded.

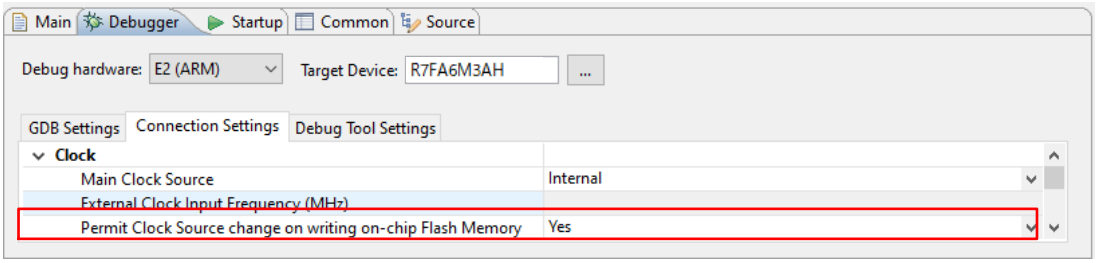


Figure 3.8 Setting of [Permit Clock Source change on writing on-chip Flash Memory]

If data are downloaded to flash memory while any condition is not satisfied, the emulator shows an error message and stops processing. In such a case, restart downloading of data to flash memory after the CPU has been reset or reconnect the emulator debugger after reviewing its settings.

(5) Downloading data to a secure area

When downloading data to a secure area which has been specified for the security MPU, for [Flash] in the [Debug Tool Settings] tabbed page of the emulator debugger, select [Yes] for [Erase on-chip program flash before download].

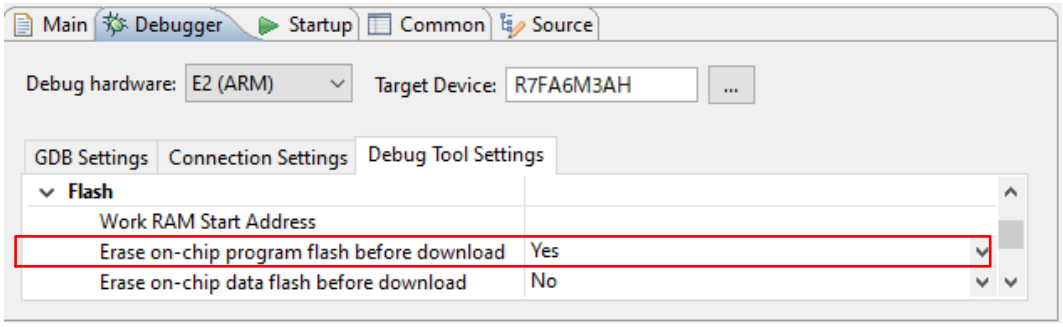


Figure 3.9 Setting of [Erase on-chip program flash before download]

(6) Access window function

When the access window function of the MCU is to be used, only reprogram the flash memory in the area specified for access.

(7) Conditions for using software breaks in flash memory

When the MCU satisfies all the following conditions, the software break function for flash memory is enabled.

(a) The code flash memory of the MCU is in read mode.

(b) The frequency of the system clock (ICLK) of the MCU is 1 MHz or higher.*

(c) For [Break] in the [Debug Tool Settings] tabbed page, [Yes] is selected for [Use Flash Breakpoints].

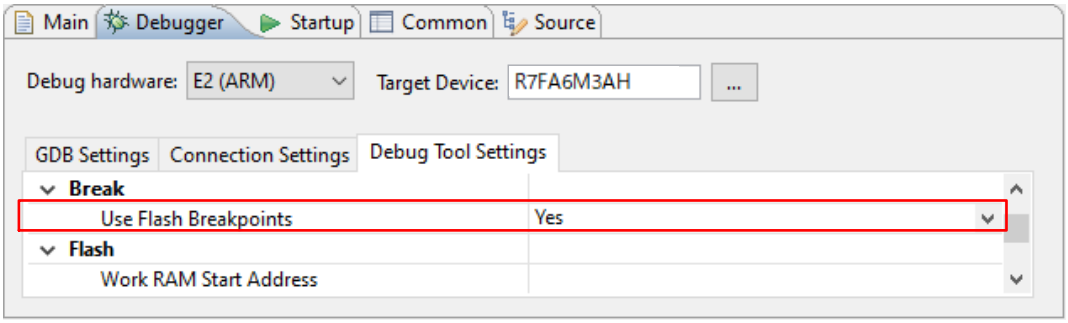


Figure 3.10 Setting of [Use Flash Breakpoints]

Note: For [Clock] in the [Connection Settings] tabbed page of the emulator debugger, when [Yes] is selected for [Permit Clock Source change on writing on-chip Flash Memory], condition (b) can be excluded.

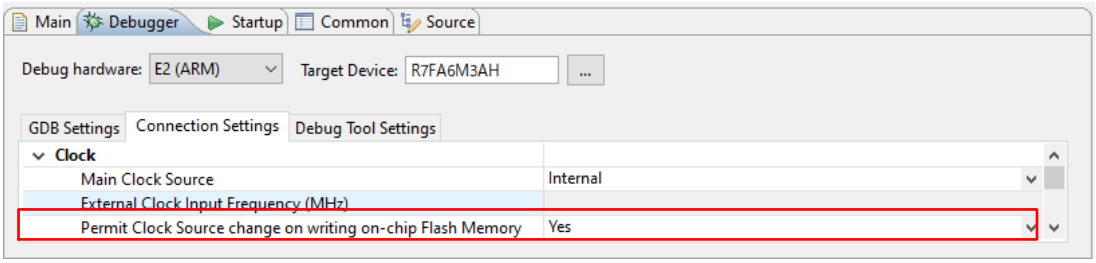


Figure 3.11 Setting of [Permit Clock Source change on writing on-chip Flash Memory]

If the software break function is used with any condition not satisfied, the emulator shows an error message. In such a case, use the hardware break or confirm that conditions (a) to (c) above are satisfied.

(8) Flash memory I/O register

After a debugging operation that involves reprogramming of flash memory, the value of the flash memory I/O register is rewritten by the emulator debugger.

3.3.3 Note on Using Software Breaks in the On-Chip SRAM

(1) Overwriting of software breakpoints by user programs

If a software breakpoint is overwritten by a user program, the program will not stop even if it runs through the address. In such a case, set the software breakpoint after the target on-chip SRAM has been rewritten by the program.

3.3.4 Notes on Using Software Breaks (Common to the On-Chip SRAM and Flash Memory)

(1) Setting software breakpoints in a secure area

Do not set software breakpoints in a secure area which has been specified by the security MPU. If this is attempted, an instruction code may be restored as incorrect data on release of the break.

(2) Reading an address where a software breakpoint has been set

Do not read an address where a user program has set a software breakpoint. Doing so may cause the program to operate in a different way from in the normal state.

(3) Viewing memory in the [Memory] view

During execution of the user program, if a range of memory in which a software breakpoint has been set is displayed in the [Memory] view of the emulator debugger, the value (BKPT instruction code) that is shown is different from that in the actual program data.

(4) Removal of software breakpoints when the emulator debugger is to be disconnected

When the emulator debugger is to be disconnected, remove all software breakpoints that have been set. At this time, reset the CPU since the emulator debugger is certain to reprogram the flash memory.

3.3.5 Note on Peripheral I/O Registers Occupied by the Debugger

(1) Peripheral I/O registers occupied by the debugger

The emulator debugger occupies the following peripheral I/O registers during debugging. Do not change the values of these registers, since continued debugging might not be possible after having done so.

- Debug stop control register (DBGSTOPCR)
- Micro trace buffer (MTB) (SFR area)
- System control OCD control register (SYOCDCR)

3.3.6 Note on Using the MTB Trace Function

When the MTB trace function is in use, the trace recording area is the size of recorded tracing that has been selected for the emulator debugger from the address where the on-chip SRAM starts. When the on-chip SRAM is in use by a user program, do not use the trace recording area. For details on the address where the on-chip SRAM starts, refer to the *User's Manual: Hardware* for the given MCU.

If the size of the trace recording area for the emulator debugger is changed, the linker script must also be changed.

3.3.7 Notes on Using the ETB Trace Function

The following describes points for caution regarding the ETB trace function of MCUs which have a Cortex-M33 core.

(1) When a branch instruction is step-executed, the results of tracing are not correct.
After step-execution of the instruction, the address that is displayed is not that of the branch destination but that of the branch source.

(2) The operation of data comparison events as trace events is not normal.
Even if a data comparison event is set as a trace event, the data are not compared but the event condition is satisfied at the time of access to the location alone.

(3) If a reset is applied during the execution of a program, trace information will not be acquired.

(4) If any of the following exceptions occur, the results of tracing may not be correct.
- HardFault
- MemManage
- BusFault
- Debug Monitor
- PendSV
- SysTick

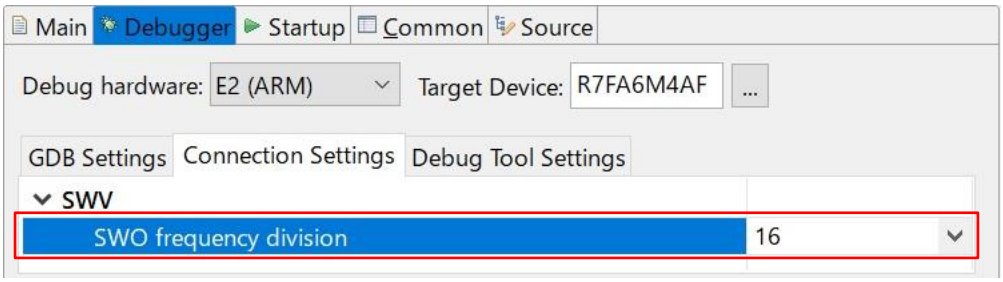
3.3.8 Notes on Using the SWO Trace Function

The E2 emulator supports the acquisition and display of trace data which are output serially via the SWO pin (hereafter referred to as the SWO trace function). Notes on using the SWO trace function are provided below.

(1) Interface clock output from the SWO pin
- The E2 emulator handles up to 15 MHz for the interface clock of data output by the SWO trace function.
- The setting for frequency division* on the emulator debugger for the frequency of the system clock source in the MCU determines the frequency of the interface clock.
- Trace information will not be correctly displayed if the frequency of the interface clock is greater than 15 MHz because the emulator will not be capable of the normal reception of trace data. Therefore, correctly set the frequency division* on the emulator debugger so that the frequency is no greater than 15 MHz.

Example: For an RA6M4 device, ICLK at 100 MHz, and frequency division by two for ICLK (SCKDIVCR.ICK: 001b)
Since the frequency of the system clock source is 200 MHz with the parameters above, select a value of at least 16 (for a frequency of or lower than 12.5 MHz) for the frequency division* on the emulator debugger.

Note: [SWO frequency division] for [SWV] on the [Connection Settings] tabbed page of the emulator debugger



(2) Buffer overflow in the MCU

A trace buffer overflow may occur in the MCU if the output of trace data by the MCU to the emulator as required is not possible.

The display of trace data on the emulator debugger is not possible when the buffer has overflowed.

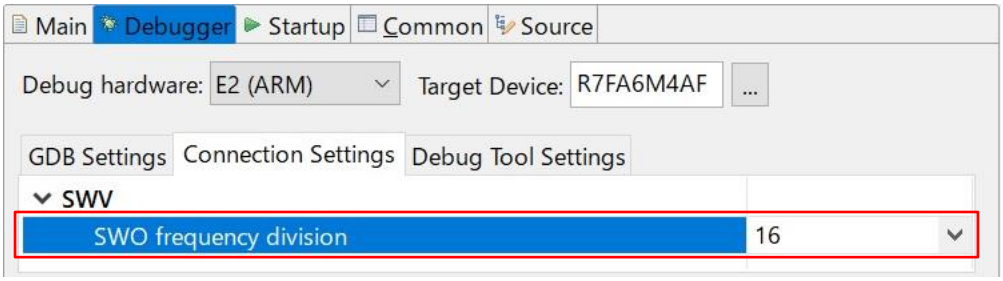
For more details on the conditions for the buffer in the MCU overflowing, refer to the *Architecture Reference Manual* for the target CPU.

(3) Buffer overflow in the emulator

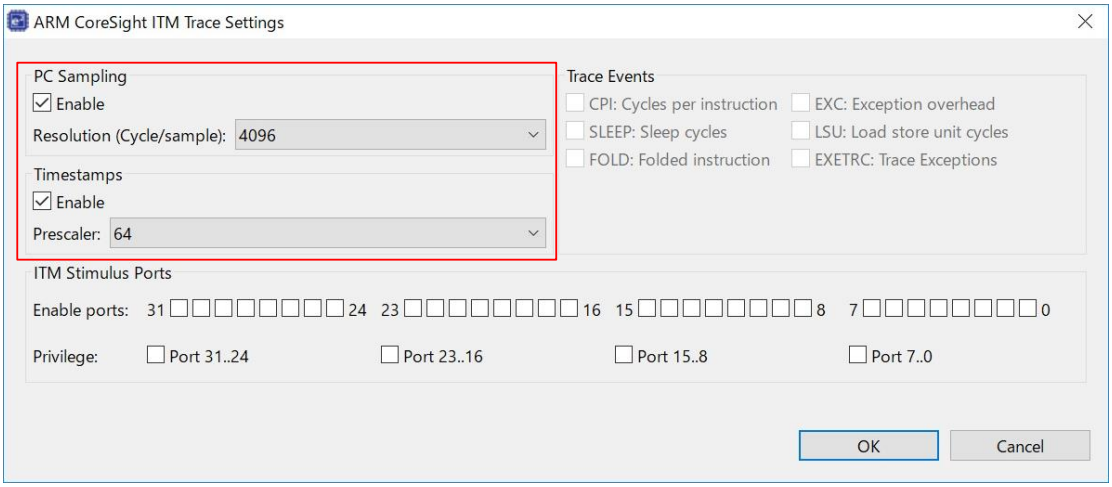
When using the SWO trace function, a reception buffer overflow may occur in the emulator if the reading of buffered trace data by the emulator debugger cannot keep up with the output of trace data.

- If the buffer overflows, the emulator debugger displays a warning message and keeps acquiring and displaying trace data. However, trace data will not be shown during the overflow.
- Various factors such as the performance of the host PC, operating frequency of the MCU, the amount of output trace data, and debugging functions running at the same time may cause an overflow.
- To resolve the overflow, the following measures are effective in reducing the transmission of trace data to the emulator.
  - Set a larger value for [SWO frequency division] on the emulator debugger*1.
  - When [Enable] is set for [PC Sampling], set a larger value for [Resolution]*2.
  - When [Enable] is set for [Timestamp], set a larger value for [Prescaler]*2.
  - Disable any SWO trace functions, such as [PC Sampling] and [Timestamps], which are running at the same time*2.
  - When trace data are output by the user program, adjust the program so that the output interval is large.

Notes: 1. [SWO frequency division] for [SWV] on the [Connection Settings] tabbed page of the emulator debugger



2. [CoreSight ITM Settings] in the [Live Trace Console] view of the emulator debugger



3.3.9 Notes on Low-Power Modes

(1) Debugging in SSTBY or SNOOZE mode

In SSTBY or SNOOZE mode, the emulator debugger does not have access to the system bus of the MCU. While the user program is being executed or during mode transitions of the MCU, setting and viewing of the memory or peripheral I/O registers and setting and changing breakpoints are not possible.

(2) Forcibly stopping a program in SSTBY or SNOOZE mode

When a program is forcibly stopped in SSTBY or SNOOZE mode, proceed with one of the following operations. Each operation leads to release from SSTBY or SNOOZE mode.

— Using [Reset] of the emulator debugger makes the MCU stop the user program and go to the position indicated by the reset vector.

— Using [Suspend] of the emulator debugger stops the MCU at the next instruction after the WFE instruction which led to the mode transition. When [Suspend] is to be used, for [Connection] on the [Connection Settings] tabbed page of the emulator debugger, select [Yes] for [Low Power Handling].

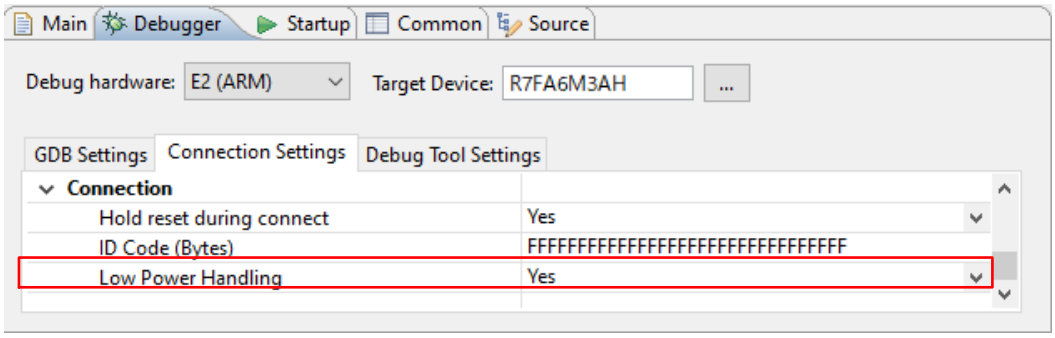


Figure 3.12 Setting of [Low Power Handling]

Even if [Yes] is selected for [Low Power Handling], forcibly stopping the program in SSTBY or SNOOZE mode on a device that incorporates the TrustZone® facility (for example, an RA6M4 device) is not possible.

(3) Debugging in DSTBY mode

Debugging a program that has entered DSTBY is not supported.

3.3.10 Current Drawn during Debugging

Since the debugging circuits within the MCU are always active during connection of the emulator debugger, the MCU draws more current than in the actual user system. Take care on this point when attempting to measure the current drawn in the user system.

3.3.11 Note on the Memory Protection Units (MPUs)

(1) Access to protected areas

Table 3.1 shows the operation of the emulator debugger in response to attempted access to areas protected by each of the MPUs.

Table 3.1 Access to MPU-Protected Areas

Protected Area	Operation of the Emulator Debugger
Arm® MPU	Accessible
Bus master MPU	The protected area is not accessible.
Slave MPU	Accessible
Security MPU	On-chip SRAM/peripheral I/O registers • Reading: A dummy value (0x00) is read. • Writing: Writing is ignored. Flash memory • Reading: A dummy value (0x00) is read. • Writing: Writing is possible if data are downloaded and written.

3.3.12 Notes on the TrustZone® Facility

(1) When the DLM state of the device is NSECSD, the emulator debugger is connected to the device with debugging-access level DBG1.

(2) When the emulator debugger is connected to the device with debugging-access level DBG1, a program must have already been written such that execution by the device correctly enters the non-secure area after release from a CPU reset (after the emulator debugger is connected, the execution will stop at the first instruction where execution by the device enters the non-secure area.)

(3) While the emulator debugger is connected with debugging-access level DBG1, breakpoints cannot be set in a secure area.

(4) Access to secured areas

Table 3.2 shows the operation of the emulator debugger in response to attempted access to areas secured by the TrustZone® facility during connection with debugging-access level DBG1.

Table 3.2 Access to Secured Areas

Secured Area	Operation of the Emulator Debugger
Peripheral I/O registers	Reading: A dummy value (0x00) is read.* Writing: Writing is ignored.
On-chip SRAM	Reading and writing: A security error is indicated.

Note: For the values read from peripheral I/O registers, refer to the specifications of each register.

(5) Use of software break functions at debugging-access level DBG2 (flash memory)

When a program for debugging is allocated to both secure and non-secure areas, do not execute debugging operations related to the following software break functions.

- Setting and canceling software breakpoints in a non-secure callable (NSC) or non-secure area.
- Setting and canceling software breakpoints in a situation where a program is stopped in a non-secure area.
- Moving to a non-secure area by stepping through after a program has stopped at a software breakpoint in a secure area.

If the above operations are executed, the TrustZone® facility may cause faults or failures in the reprogramming of flash memory in further debugging operations, such that continued debugging is not possible.

When the setting of breakpoints in the cases above is necessary, use hardware breakpoints instead.

3.3.13 Note on the Code Flash Dual Mode Facility

When the code flash dual mode (including the bank-swapping facility) and code flash block-swapping facility are to be used, select [No] for [Erase on-chip program flash before download] for [Flash] on the [Debug Tool Settings] tabbed page of the emulator debugger. For [System], select [Yes] for [Debug the program re-writing the on-chip program flash].

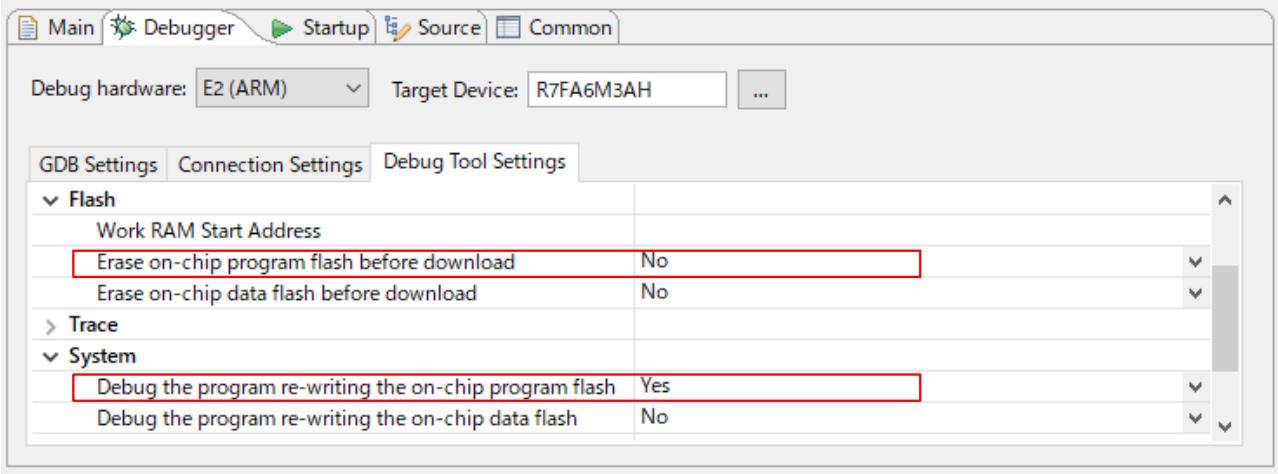


Figure 3.13 Setting of Items under [Flash] and [System]

3.3.14 Notes on the S Cache and C Cache Facilities

When using the S cache and C cache facilities, use hardware breakpoints instead of software breakpoints if both of the following conditions are met.

- The CPU is in the non-secure state.
- Control registers (CCACTL, CCAFCT, SCACTL, and SCAFCT) for the S cache and C cache facilities are in the secure state.

When downloading proceeds while both of the conditions above are met, invalidate the cache contents after downloading because differences between the written contents and cache memory may have been generated.

3.3.15 Notes on the Start/Stop Facility

The start/stop facility is used to execute a specified routine of the user program immediately before executing and after halting the user program. This facility uses a debug monitor interrupt, which is one of the interrupt functions of the Arm® core. Note the following items when using the start/stop facility.

(1) Monitor program for the start/stop facility

The start/stop facility is realized by an emulator debugger writing a monitor program for the start/stop facility to the on-chip SRAM of the MCU. Thus, a part of the SRAM (1 Kbyte) is occupied while the start/stop facility is enabled.

(2) Destination for allocation of the monitor program for the start/stop facility

The destination for allocation of the monitor program for the start/stop facility must not overlap with any areas used for DMAC or DTC transfer and must be within an area for which security settings have been made by the MPU or otherwise. Enter the start address for the program in the on-chip RAM in 1000h-byte units in [Work RAM Start Address] for [Start/Stop Function Settings] on the [Debug Tool Settings] tabbed page of the emulator debugger.

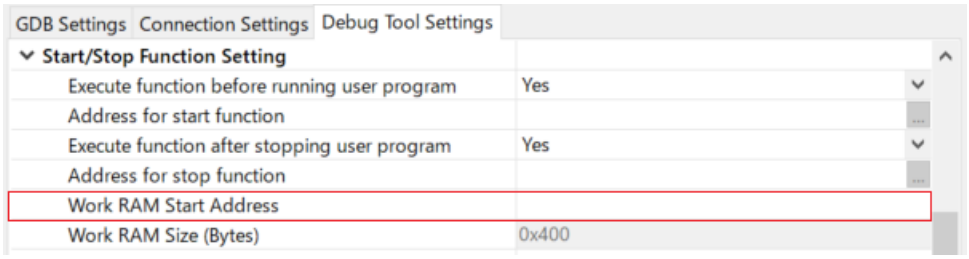


Figure 3.14 Setting of [Work RAM Start Address]

(3) Specifying the start and stop functions

Use [Address for start function] and [Address for stop function] to specify the addresses of the routines to be executed as the start and stop functions immediately before executing the user program and immediately after halting the user program, respectively.

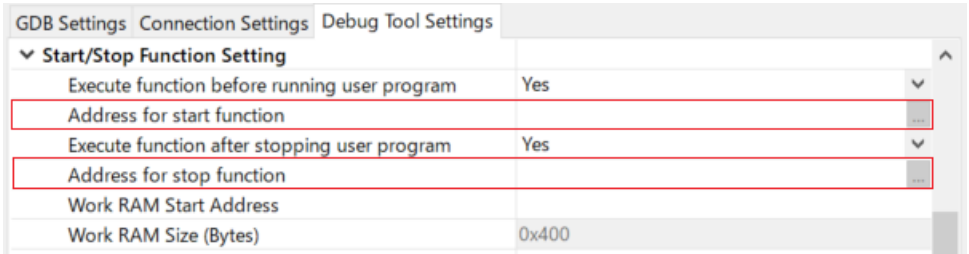


Figure 3.15 Settings of [Address for start function] and [Address for stop function]

Clicking on the […] button to the right of [Address for start function] or [Address for stop function] produces the [Select Symbol] window. You can directly select a symbol from the list of function symbols.

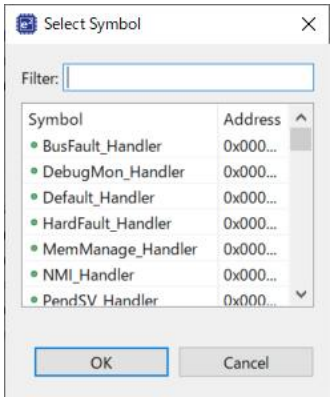


Figure 3.16 [Select Symbol] Window

(4) Writing the start and stop functions

The start and stop functions can be written in the C or even assembly languages. In the C language, be sure to include a return instruction for execution at the end of each function. In assembly language, save the value of the LR register at the beginning of the function and restore the value of the PC at the end of the function. Any names can be used for the functions.

(5) Writing code for the vector table

When using the start/stop facility, specify the jump address for the debug monitor interrupt in the vector table as the value obtained by adding 0x1 to the destination address for allocation of the monitor program for the start/stop facility.

Example: A case where 0x20002000 has been specified as the destination address for the allocation of the monitor program for the start/stop facility

Figure 3.17 shows an example of code for a vector table in the source program (startup.c) that has been automatically generated by the flexible software package (FSP). If you specified 0x20002000 as the destination address for allocation of the monitor program for the start/stop facility, specify 0x20002001 for the jump address for the debug monitor interrupt in the vector table.

```
/* Vector table. */
BSP_DONT_REMOVE const exc_ptr_t __Vectors[BSP_CORTEX_VECTOR_TABLE_ENTRIES] BSP_PLACE_IN_SECTION(
    BSP_SECTION_FIXED_VECTORS) =
{
    (exc_ptr_t) (&g_main_stack[0] + BSP_CFG_STACK_MAIN_BYTES), /*      Initial Stack Pointer     */
    Reset_Handler,                                             /*      Reset Handler             */
    NMI_Handler,                                               /*      NMI Handler               */
    HardFault_Handler,                                         /*      Hard Fault Handler        */
    MemManage_Handler,                                         /*      MPU Fault Handler         */
    BusFault_Handler,                                          /*      Bus Fault Handler         */
    UsageFault_Handler,                                        /*      Usage Fault Handler       */
    SecureFault_Handler,                                       /*      Secure Fault Handler      */
    0,                                                         /*      Reserved                  */
    0,                                                         /*      Reserved                  */
    0,                                                         /*      Reserved                  */
    SVC_Handler,                                               /*      SVCall Handler            */
    0x20002001,                                                /*      Debug Monitor Handler     */
    0,                                                         /*      Reserved                  */
    PendSV_Handler,                                            /*      PendSV Handler            */
    SysTick_Handler,                                           /*      SysTick Handler           */
};
```

Figure 3.17 Example of the Setting for a Vector Table

(6) Note the following points when using the start/stop facility. Any of the listed actions may cause the emulator to operate in an unexpected way.
- Do not use multiple interrupts.
- Ensure that optimization by the compiler does not eliminate the start and stop functions.
- Do not set any breakpoints in the start and stop functions.

In addition, note that the accuracy of run-break times cannot be guaranteed while the start/stop facility is in use.

(7) The start/stop facility cannot be used in either of the following cases.
- The device has a core other than a Coretex-M33 or Coretex-M4.
- The device incorporates a TrustZone® facility (for example, an RA6M4 device) for which the debugging-access level is other than DBG2.

3.4 MCUs that are Used in Debugging

After debugging with the emulator, if the MCU is disconnected from the emulator and run on its own, correct operation cannot be guaranteed. To operate the MCU on its own, use the programming software to re-program the MCU.

MCUs that are connected to the emulator and used in debugging are placed under stress by repeated programming of flash memory during emulation. Do not use MCUs that were used in debugging in mass-production for end users.

3.5 Final Evaluation of the User Program

Before entering the mass-production phase, be sure to perform a final evaluation of the program which has been written to the flash ROM by the programming software, without the emulator connected.

Revision History	E2 Emulator, E2 Emulator Lite Additional Document for User's Manual (Notes on Connection of RA Devices)

Rev.	Date	Description	
		Page	Summary
1.00	Mar.16.20	—	First Edition issued.
2.00	Oct.01.20	—	With the addition of support for TrustZone® and DLM, relevant information was included. Section 3.3.7, Notes on Using the ETB Trace Function, was added.
3.00	Aug.16.21	—	The following sections were added: 2.4.3, SCI Connection, 3.3.8, Notes on Using the SWO Trace Function, and 3.3.14, Notes on the S Cache and C Cache Facilities. A new item, (5) Use of software break functions at debugging-access level DBG2 (flash memory), was added to section 3.3.12. The statements in (5) Connection speed, in section 3.3.1, were modified.
3.10	Jun.16.22	12 to 14	The description of note 1 was changed.
3.20	Mar.01.23	16	The statements in section 2.5.2, MD Pin, were modified.
		36	Section 3.3.15, Notes on the Start/Stop Facility, was added.

E2 Emulator, E2 Emulator Lite
Additional Document for User's Manual
(Notes on Connection of RA Devices)

Publication Date: Rev.3.20 Mar.01.23

Published by: Renesas Electronics Corporation

E2 Emulator, E2 Emulator Lite
Additional Document for User's Manual
(Notes on Connection of RA Devices)

Renesas Electronics Corporation

R20UT4686EJ0320