User's Manual

E1/E20 Emulator, E2 Emulator

Additional Document for User's Manual (Notes on Connection of RH850/F1H and RH850/F1M)

Supported Devices:
RH850 Family RH850/F1x Series

Renesas Electronics
www.renesas.com

Rev.8.00 Oct 2020

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 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.
5. 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.
6. 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.
7. 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.
8. 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.
9. 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.
10. 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.
11. This document shall not be reprinted, 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.

(Note 1) “Renesas Electronics” as used in this document means Renesas Electronics Corporation and also includes its directly or indirectly controlled subsidiaries.

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

(Rev.4.0-1 November 2017)

Contents

1. Outline

1.1 Features of an E1, E20 or E2 emulator

An E1, E20 or E2 emulator is an on-chip debugging emulator that includes a flash programming function, which is used for debugging and programming programs to be embedded in microcontrollers that have on-chip flash memory. That is, either product can debug a program while the target microcontroller is connected to the user system, and can write programs to the on-chip flash memory of microcontrollers.

1.2 Caution on using the E20 emulator

The functions used for debugging of the RH850 family by using the E20 emulator are the same as in the E1 emulator. Large trace function, characteristic function of the E20 emulator, cannot be used.

1.3 Configuration of manuals

When using the E1, E20 or E2 emulator in debugging with an RH850 family product, be sure to read the manuals (1) and (2) below. Also read the application note (3) if required.

(1) E1 or E20 emulator user's manual, E2 emulator user's manual

The E1/E20 Emulator User's Manual, E2 Emulator User's Manual describes hardware specifications including the following items:

- Components of the emulators
- Emulator hardware specifications
- Connecting the emulator to a host computer and user system

(2) E1 or E20 emulator, E2 emulator additional document for user's manual

An E1 or E20 Emulator, E2 Emulator Additional Document for User's Manual describes functions of a debugger, and its contents depend on the given set of MCUs. In general, an additional document has notes on items including the following:

- For use in hardware design, an example of connection and the interface circuits required to connect the emulator.
- Notes on using the emulator

(3) E2 emulator application note

The E2 Emulator Application Note includes an explanation, descriptions of usage, and notes on the extended functions of the E2 emulator.

2. Connecting the Emulator and User System

To connect the E1, E20 or E2 emulator, a connector for the user system interface cable must be mounted on the user system. When designing the user system, read this chapter of this manual and the hardware manual for the MCUs to be used.

2.1 Connector mounted on the user system

Table 2-1 shows the recommended connectors for connection of the E1, E20 or E2 emulator. If you intend to use the 14-pin connector, do not mount components with heights exceeding 10 mm within 5 mm of the connector on the user system as shown in Figure 2-1.

Table 2-1 Recommended Connectors

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

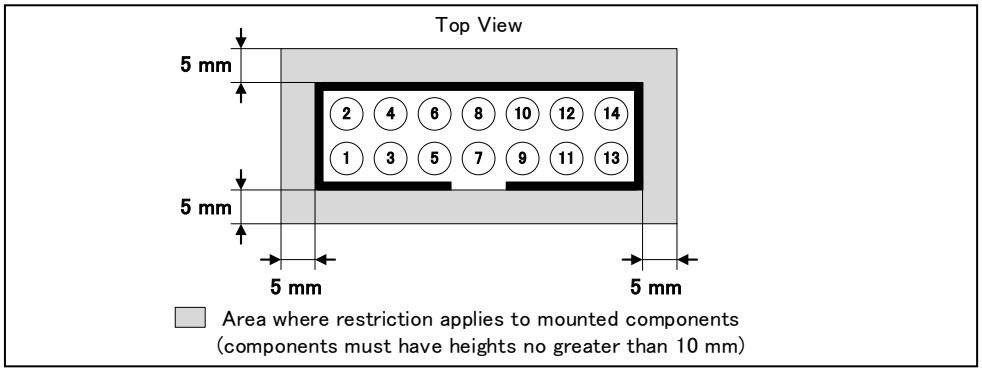

Figure 2-1 Area where Restriction Applies to Mounted Components

• For the connection of an E1 emulator
Figure 2-2 shows an example of the connection of the user system interface cable of an E1 emulator to a 14-pin connector.

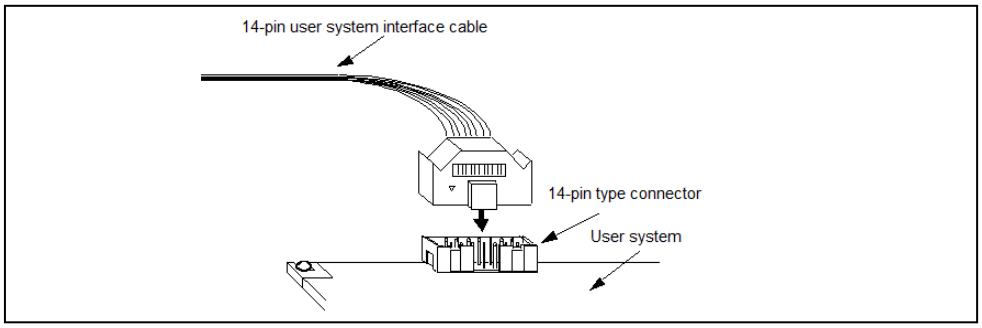

Figure 2-2 Connecting the User System Interface Cable to the 14-pin Connector in the E1 Emulator

• For the connection of an E20 emulator
To use an E20 emulator with a 14-pin connector, use the 38-pin/14-pin connector conversion adapter [R0E000200CKA00] that comes with the E20.

• For the connection of an E2 emulator

To use an E2 emulator with a 14-pin connector, use the connector conversion adapter that comes with the E2. Figure 2-3 shows an example of the connection.

The connector conversion adapter is provided with a switch. Setting for the switch must be on the “1” side for the RH850. Operation is not guaranteed if the switch is on the “3” side. For setting the switch, refer to Table 2-2.

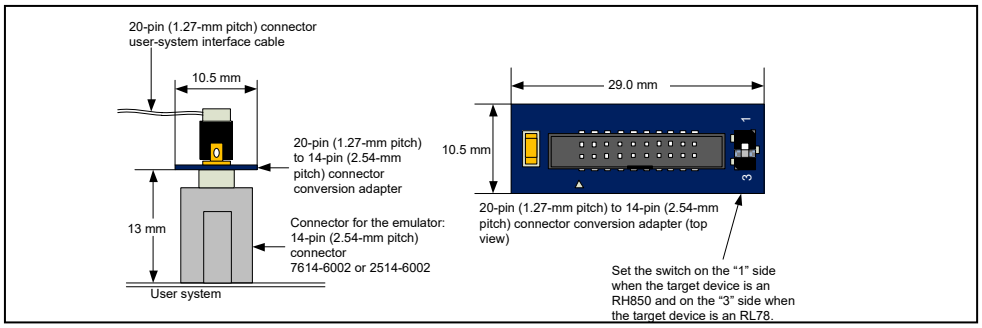

Figure 2-3 Connecting the User System Interface Cable to the 14-pin Connector in the E2 Emulator

Table 2-2 Setting of Switches (SW1)

Setting	Description
1	The target device is an RH850 microcontroller (default setting).
3	The target device is an RL78 microcontroller.

CAUTION

Note on connector insertion and removal (1):

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

CAUTION

Note on connector insertion and removal (2):

Be aware that the user-system interface cable or the connector conversion adapter must be inserted with the correct orientation. Connecting the user-system interface cable or the connector conversion adapter with the wrong orientation may cause damage.

2.2 Pin assignments of the connector

Table 2-3 shows the pin assignments of the 14-pin connector.

Table 2-3 Pin Assignments of the 14-pin Connector

Pin No.	Signal name (#: active low, -: unused)				I/O (*3)
	Debugging		Programming		
	4-pin LPD	1-pin LPD	2-wire UART	1-wire UART	
1	LPDCLK	—	—	—	Input
2 (*1)	GND	GND	GND	GND	—
3	—	—	—	—	—
4	FPMD0	FPMD0	FPMD0	FPMD0	Input
5	LPDO	—	FPDT	—	Output
6	—	—	—	—	—
7	LPDIO	LPDIO	FPDR	FPDR	I/O
8	TVDD	TVDD	TVDD	TVDD	—
9	—	—	—	—	—
10	—	—	—	—	—
11	LPDCLKO	—	—	—	Output
12 (*1)	GND	GND	GND	GND	—
13 (*2)	RESET#	RESET#	RESET#	RESET#	Input
14 (*1)	GND	GND	GND	GND	—

Notes 1. Securely connect pins 2, 12, and 14 of the connector to GND of the user system. These pins are used for electrical GND and to monitor connection with the user system by the E1, E20 or E2 emulator.
2. Be particularly sure to connect pin 13 before using the emulator.
3. Input and output are defined from the perspective of the user system.

CAUTION

Unused pins:

Do not apply signals from the user system to unused pins. Doing so may damage the pins.

2.3 Connection interface and modes

The operating mode and the connection interface of an E1, E20 or E2 emulator is switched in the ways shown in Table 2-4 according to whether it is in use for debugging (when a debugger is in use) or programming (when the Flash Programmer is in use). The serial programming mode may still be used even if the debugger is in use. When flash memory is programmed by the downloading function of the debugger, the flash self-programming function is used.

Table 2-4 Modes and Connection Interfaces

		Mode and Connection Interface	
		Normal Operation Mode	Serial Programming Mode
Tool to be Used		Connection Interface	Connection Interface
Flash Programmer (e.g. RFP)		-	1- or 2-wire UART
Debugger (e.g. CS+)	When OPJTAG is automatically set (connected)*	-	When 1-pin LPD is selected, 1-wire UART is used. When 4-pin LPD is selected, 2-wire UART is used.
	During debugging	1- or 4-pin LPD	-

(*)OPJTAG automatic setting function: When a device is debugged, the OPJTAG bit in the option byte register determines the type of connection interface. Debugging will not start if the interface selected by the OPJTAG bit does not match that selected by the debugger. If the OPJTAG automatic setting function is enabled, the emulator makes a transition to the serial programming mode without fail and reads the OPJTAG bit. If the interface differs from that selected by the debugger, the OPJTAG bit is rewritten, the mode is switched to the normal operating mode, and debugging will start.

When this function is enabled to start debugging, since the mode is switched to the serial programming mode, some emulation may be impossible since the initial values in memory and of ECC errors after a reset are undefined. Therefore, only use the OPJTAG automatic setting function when the OPJTAG bit in the option byte register is to be modified. For details on setting this function, refer to the user's manual for the debugger you are using.

With CS+, select "Yes" as the [Set OPJTAG in LPD connection before connecting] property on the [Connect Settings] tabbed page to enable the OPJTAG automatic setting function.

2.4 Examples of recommended connections between the connector and MCU

This section describes examples of recommended connections between the target MCU and interface circuit.

2.4.1 Example of recommended connections

Multiple recommended examples for connection are given in accord with the purposes for which the emulator is to be used. Select the appropriate circuit with reference to the table shown below. Be sure to take the specifications of the target device as well as measures to prevent noise into consideration when designing your circuit.

Purpose	Figure
Both debugging (4-pin LPD) and programming (1-wire UART or 2-wire UART)	Figure 2-4
Both debugging (1-pin LPD or 4-pin LPD) and programming (1-wire UART or 2-wire UART)	Figure 2-5
Both debugging (1-pin LPD) and programming (1-wire UART)	Figure 2-6
Only programming (1-wire UART or 2-wire UART)	Figure 2-7
Only programming (1-wire UART)	Figure 2-8

(1) Connection which allows Both debugging (4-pin LPD) and programming (1-wire UART or 2-wire UART)

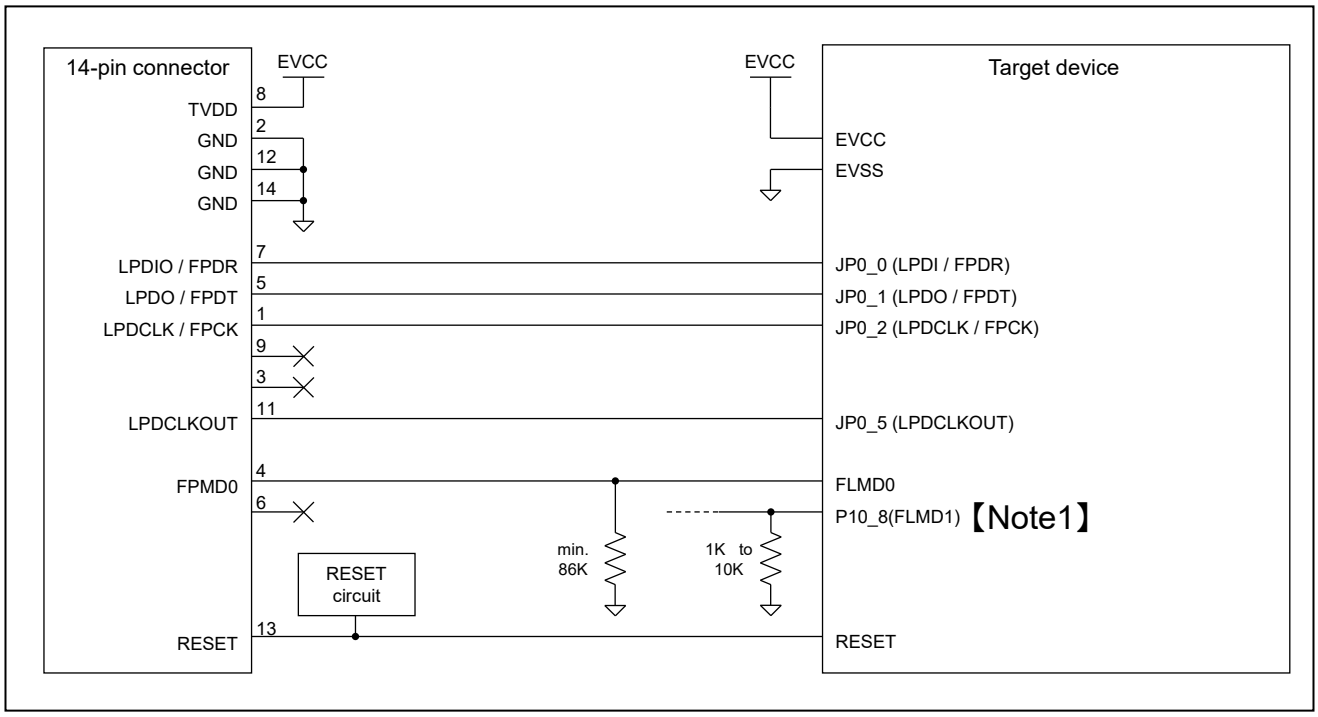

Figure 2-4 Example of Connection

- Refer to section 2.4.2, Connecting the RESET pin, for more information on the reset circuit.
- For details on TVDD, refer to section 2.4.3, Connecting the TVDD pin.
- Make wiring runs between the 14-pin connector and target device as short as possible (within 50 mm is recommended). Do not connect the signal lines between the connector and MCU to other signal lines.
- Use GND to apply a guard ring for the wiring which runs between the 14-pin connector and target device. Do not route high-speed signal lines parallel to each other or allow them to cross each other.
- Pin names may vary among target devices. Refer to the user's manual for the target device you are using for the actual pin names.
- Proceed with appropriate processing for pins of target devices which do not require connection to the emulator in accord with the descriptions in "Handling of Unused Pins" in the user's manual for the target device.

Note 1: Design the circuit so that the FLMD1 pin must be at the low level during programming.

(2) Connection which allows Both debugging (1-pin LPD or 4-pin LPD) and programming (1-wire UART or 2-wire UART)

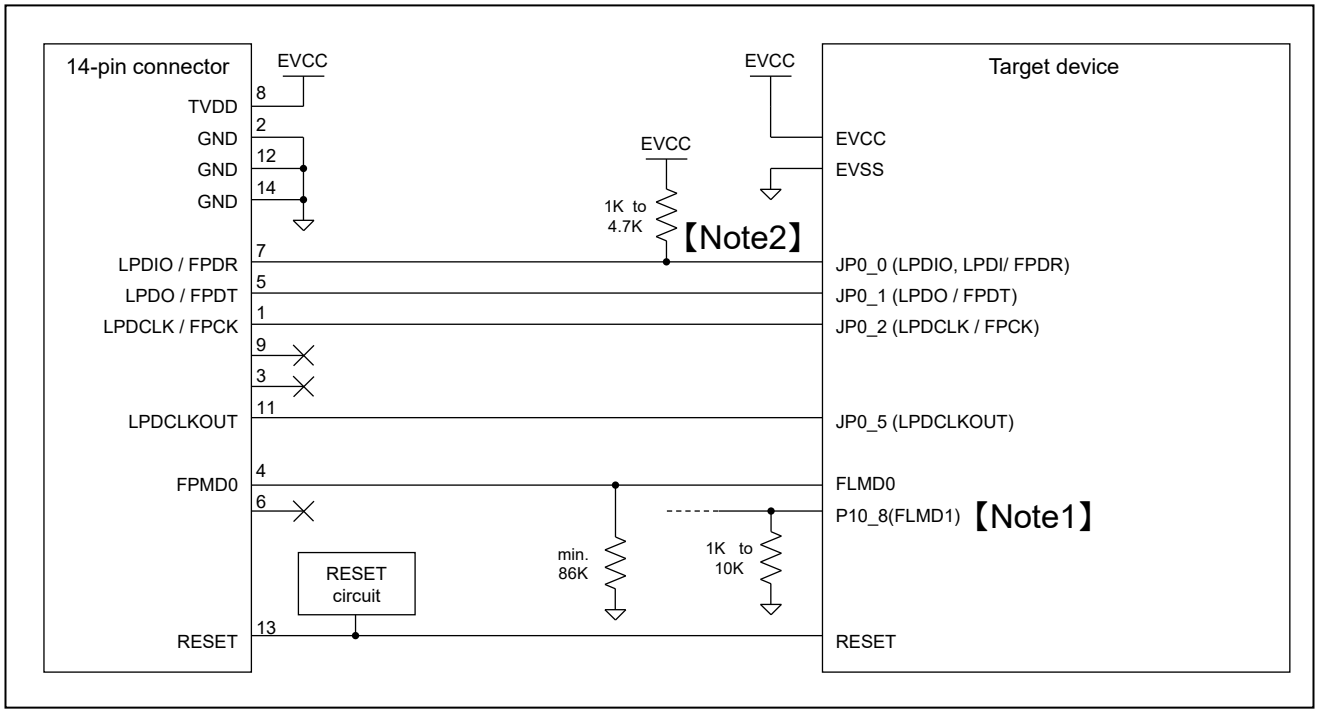

Figure 2-5 Example of Connection

- Refer to section 2.4.2, Connecting the RESET pin, for more information on the reset circuit.
- For details on TVDD, refer to section 2.4.3, Connecting the TVDD pin.
- Make wiring runs between the 14-pin connector and target device as short as possible (within 50 mm is recommended). Do not connect the signal lines between the connector and MCU to other signal lines.
- Use GND to apply a guard ring for the wiring which runs between the 14-pin connector and target device. Do not route high-speed signal lines parallel to each other or allow them to cross each other.
- Pin names may vary among target devices. Refer to the user's manual for the target device you are using for the actual pin names.
- Proceed with appropriate processing for pins of target devices which do not require connection to the emulator in accord with the descriptions in "Handling of Unused Pins" in the user's manual for the target device.

Note 1: Design the circuit so that the FLMD1 pin must be at the low level during programming.

Note 2: Pulling up of JP0_0 is only necessary if you are using 1-pin LPD. If you are not using 1-pin LPD, make the connections shown in Figure 2-4.

(3) Connection which allows Both debugging (1-pin LPD) and programming (1-wire UART)

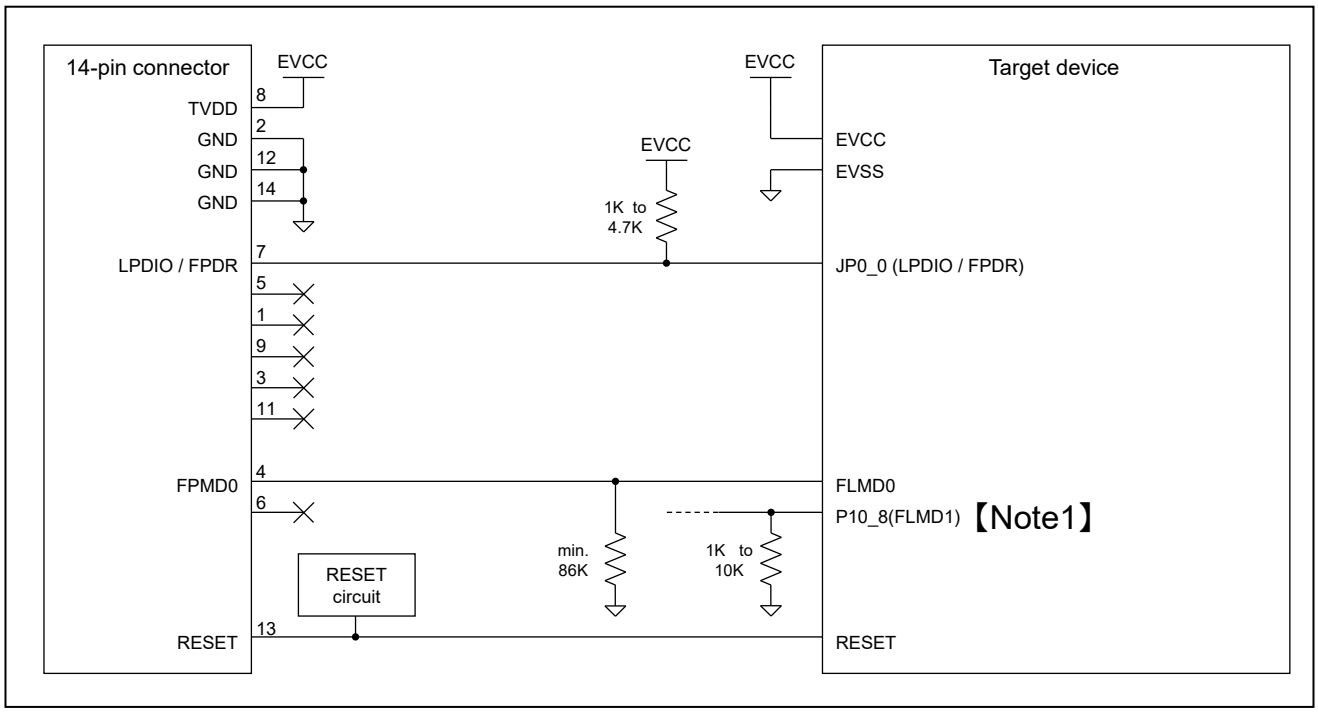

Figure 2-6 Example of Connection

- Refer to section 2.4.2, Connecting the RESET pin, for more information on the reset circuit.
- For details on TVDD, refer to section 2.4.3, Connecting the TVDD pin.
- Make wiring runs between the 14-pin connector and target device as short as possible (within 50 mm is recommended). Do not connect the signal lines between the connector and MCU to other signal lines.
- Use GND to apply a guard ring for the wiring which runs between the 14-pin connector and target device. Do not route high-speed signal lines parallel to each other or allow them to cross each other.
- Pin names may vary among target devices. Refer to the user's manual for the target device you are using for the actual pin names.
- Proceed with appropriate processing for pins of target devices which do not require connection to the emulator in accord with the descriptions in "Handling of Unused Pins" in the user's manual for the target device.

Note 1: Design the circuit so that the FLMD1 pin must be at the low level during programming.

(4) Connection which allows Only programming (1-wire UART or 2-wire UART)

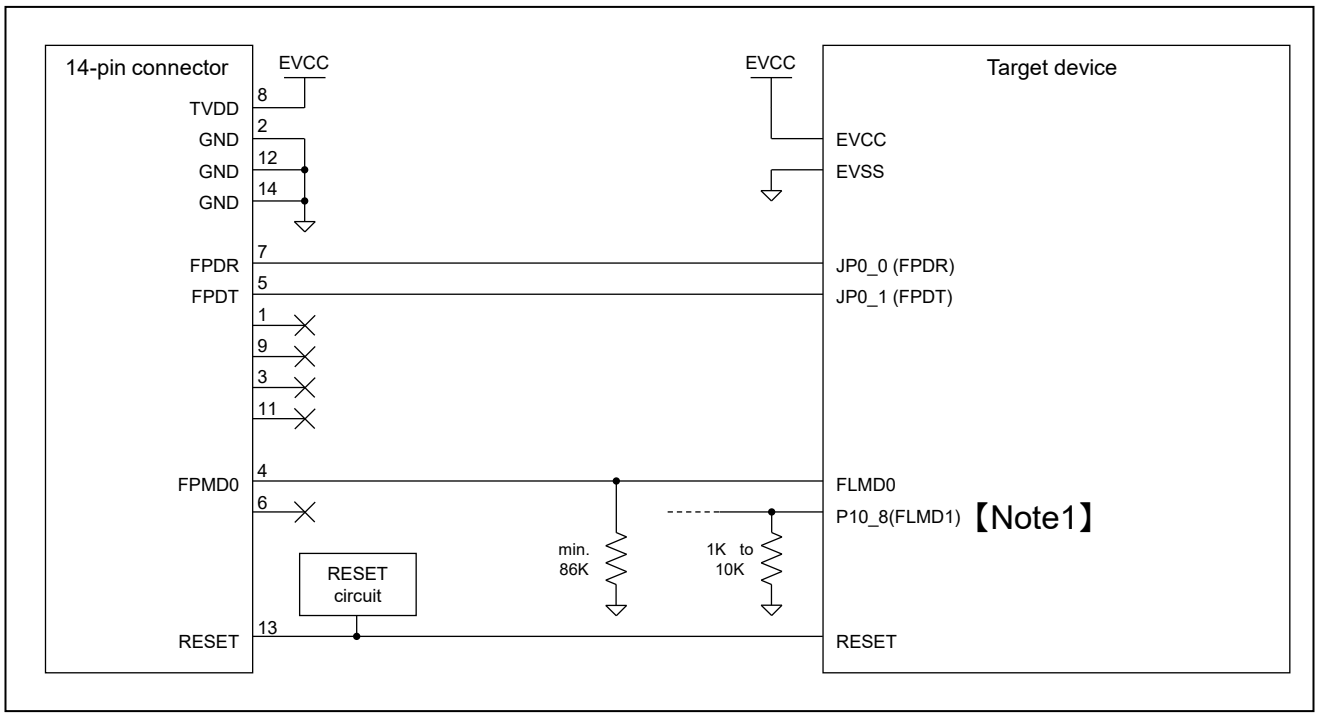

Figure 2-7 Example of Connection

- Refer to section 2.4.2, Connecting the RESET pin, for more information on the reset circuit.
- For details on TVDD, refer to section 2.4.3, Connecting the TVDD pin.
- Make wiring runs between the 14-pin connector and target device as short as possible (within 50 mm is recommended). Do not connect the signal lines between the connector and MCU to other signal lines.
- Use GND to apply a guard ring for the wiring which runs between the 14-pin connector and target device. Do not route high-speed signal lines parallel to each other or allow them to cross each other.
- Pin names may vary among target devices. Refer to the user's manual for the target device you are using for the actual pin names.
- Proceed with appropriate processing for pins of target devices which do not require connection to the emulator in accord with the descriptions in "Handling of Unused Pins" in the user's manual for the target device.

Note 1: Design the circuit so that the FLMD1 pin must be at the low level during programming.

(5) Connection which allows Only programming (1-wire UART)

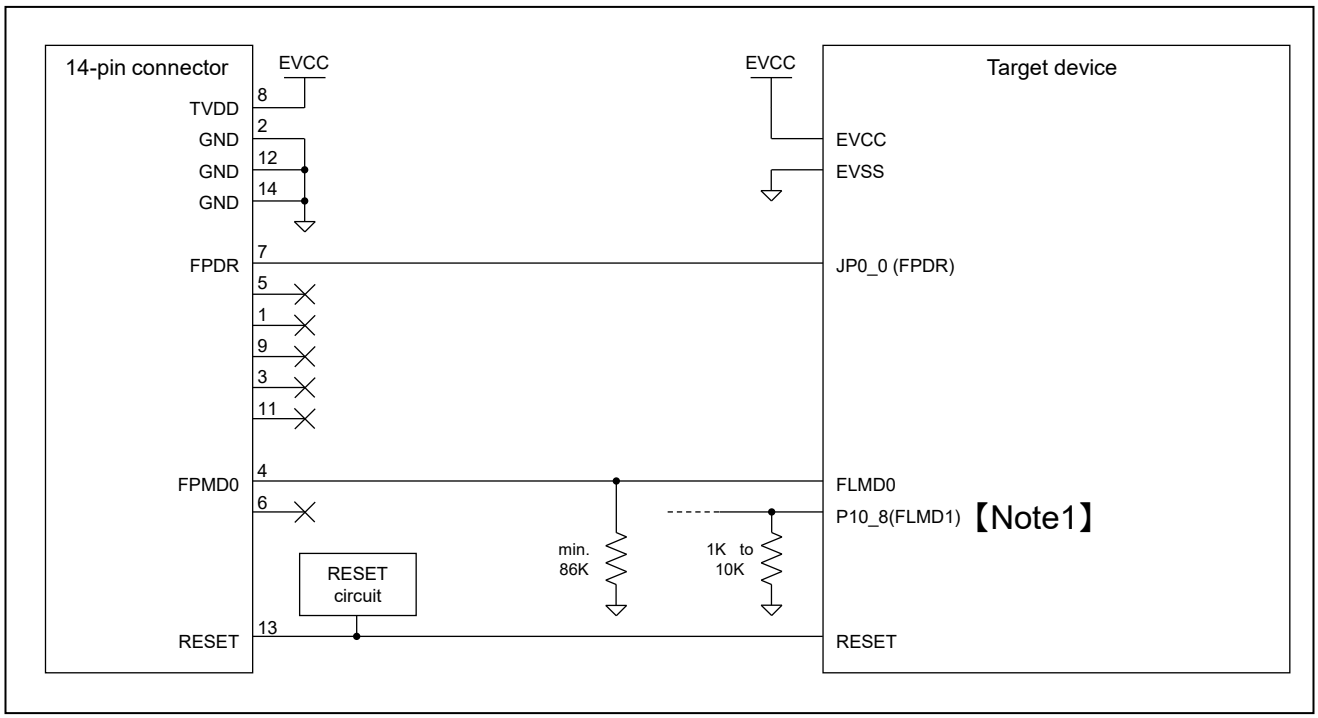

Figure 2-8 Example of Connection

- Refer to section 2.4.2, Connecting the RESET pin, for more information on the reset circuit.
- For details on TVDD, refer to section 2.4.3, Connecting the TVDD pin.
- Make wiring runs between the 14-pin connector and target device as short as possible (within 50 mm is recommended). Do not connect the signal lines between the connector and MCU to other signal lines.
- Use GND to apply a guard ring for the wiring which runs between the 14-pin connector and target device. Do not route high-speed signal lines parallel to each other or allow them to cross each other.
- Pin names may vary among target devices. Refer to the user's manual for the target device you are using for the actual pin names.
- Proceed with appropriate processing for pins of target devices which do not require connection to the emulator in accord with the descriptions in "Handling of Unused Pins" in the user's manual for the target device.

Note 1: Design the circuit so that the FLMD1 pin must be at the low level during programming.

2.4.2 Connecting the RESET pin

While you are using the E1, E20 or E2 emulator, pin 13 (RESET pin) of the 14-pin connector must be connected to the reset pin of the target device. Figure 2-9 below shows an example.
The E1, E20 or E2 emulator fixes the RESET pin to the low level before the debugger is activated. After the debugger is activated, the emulator either keeps the pin at the low level or places it in the high-impedance state in accord with the operation of the debugger.

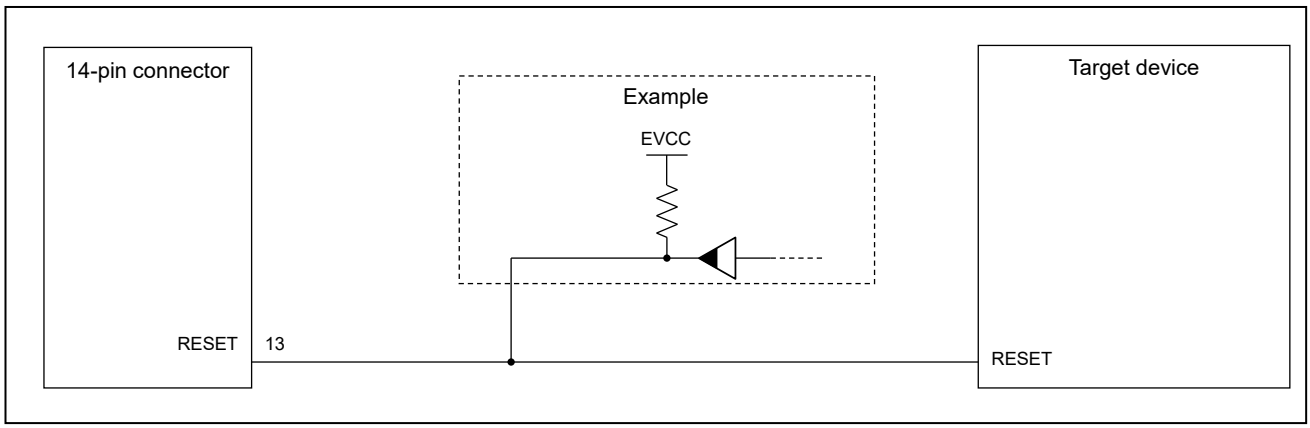

Figure 2-9 Example of Connecting a Reset Circuit

- Output of the reset circuit should be either n-channel open drain or be a signal generated solely by a resistor and capacitor (and possible other components).
- For the target device in this user's manual, pull the RESET signal up to the EVCC voltage.
- Pin 13 (RESET) of the E1, E20 or E2 emulator is pulled up (by a 100-kΩ resistor) within the emulator (refer to section 5, Internal Circuits of the Emulator).
- The RESET pin of the target device may be pulled up or down within the device. On this point, refer to the user's manual for the target device.
- The maximum sink current accepted by the RESET pin of the E1, E20 or E2 emulator is 2 mA. Select an appropriate pull-up resistance which does not surpass this value.
- Adjust the time constant of the reset circuit so that the time elapsing before the signal reaches 80% of the high level from the low level is within 900 μs.
- When you use hot plug-in, consider installation of a capacitor between the reset signal and GND in order to suppress a noise. In this case, however, the specifications of the time described above must be satisfied.

2.4.3 Connecting the TVDD pin

(1) Power source monitoring function

Connect the power source on the user system to pin 8 (TVDD pin) of the 14-pin connector. For the target device in this user's manual, this will be the source of the EVCC voltage.
The power source connected to the TVDD pin provides power to the final stage output buffer and first stage input buffer on the E1/E20/E2 emulator circuit. When the E1, E20 or E2 emulator is connected, it will draw current as described below in addition to the current drawn by the user system.

- E1/E2 emulator: Approx. 20 mA when TVDD is 3.3 V, and approx. 40 mA when TVDD is 5.0 V
- E20 emulator: Approx. 40 mA when TVDD is 3.3 V, and approx. 100 mA when TVDD is 5.0 V

If there is a possibility you will be using hot plug-in connection, you will need to configure the circuit as shown below.
Pin 8 of the E1 emulator is connected to a 4.7-uF capacitor as shown in (1) in Figure 2-10, so hot plug-in connection of the emulator may lead to a momentary drop in the power-supply voltage on the user system. This might cause the MCU to be reset.
As shown in (2) in Figure 2-10, this effect can be reduced by placing a ferrite bead (or inductor) and relatively large capacitor with low equivalent series resistance near the TVDD line of the connector for connection of the emulator. Note that this measure will not completely eliminate the voltage drop. Note that hot plug-in connection is only for use during debugging, and a separately sold hot plug-in adapter is necessary to use this function otherwise.

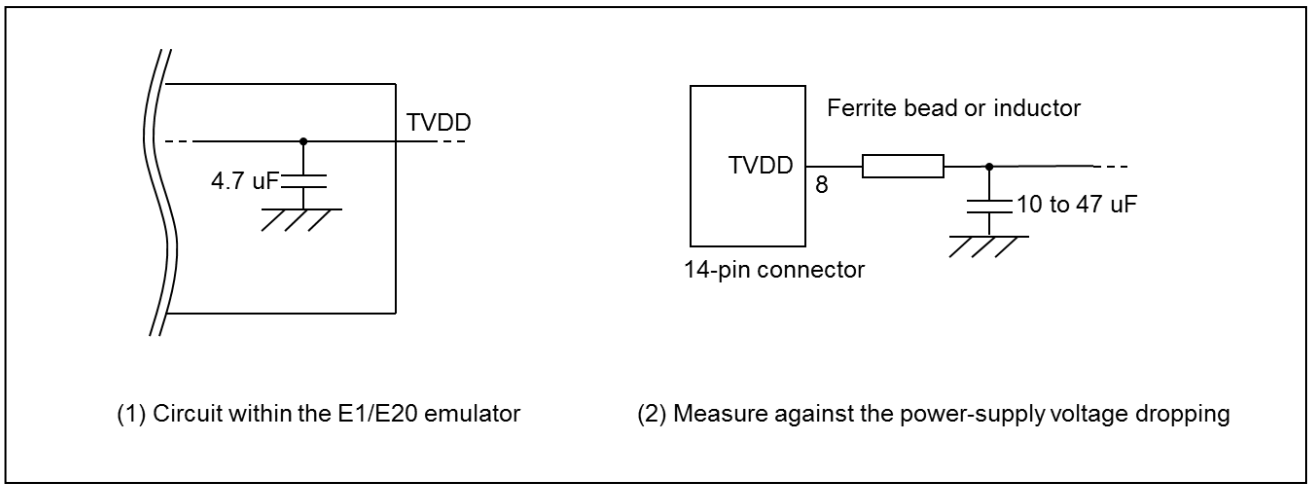

Figure 2-10 Circuit Configuration for Hot Plug-in

(2) Power supply function (applies only to the E1 or E2 emulator)

The E1 or E2 emulator can also supply power at 3.3 V or 5.0 V from the TVDD pin to the user system (at a current of up to 200 mA). When using this function, take care of the following points.

- Do not use this function if power is being separately supplied to the user system. Attempting to do so might break the E1 or E2 emulator.
- Do not use this function for a user system which draws a current of 200 mA or more. The E1 or E2 emulator or USB interface of the host machine might be broken.
- Make sure that the supplied voltage is within the voltage range required by the user system.
- E1 emulator : The 5.0-V supply depending on the environment of the host machine in use, the voltage might be lower than 5.0 V by 0.5 V or more.
- E2 emulator : The 5.0-V supply depending on the environment of the host machine in use, the voltage might be lower than 5.0 V by 0.3 V or more.

Power supply from the E1 or E2 emulator 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 E1 or E2 emulator. Use a stable, separate power supply for the user system. When writing a program for mass production processes, use the Renesas Flash Programmer.

⚠WARNING

Warning for Turning the Power On/Off:

When supplying power, ensure that there are no shorts between the user system and power circuit. Only connect the E1, E20 or E2 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.

2.4.4 Hot plug-in adapter for the E1 emulator

For hot plug-in connection to the E1 emulator, use the hot plug-in adapter for the E1 emulator (R0E000010ACB00) that is separately available from Renesas.

2.4.5 Isolator for the E1 emulator

For a debugging environment where there is a difference in potential between the GND of the user system and that of the host PC, use the isolator for the E1 emulator (R0E000010ACB20) which is separately available from Renesas.

2.4.6 Small connector conversion adapter for the E1 emulator

A small connector conversion adapter for the E1 emulator (R0E000010CKZ11) is separately available from Renesas for user system boards which are too small to mount the 14-pin connector that is the standard connector for the E1 emulator. By using the adapter, you can reduce the area taken up by the connector mounted on your system. However, when you use the small connector conversion adapter for the E1 emulator, be aware that the pin assignments of the connector differ from those of the standard interface connector for the E1 emulator.

3. Specifications

Table 3-1 shows specifications common to the E1, E20, and E2 emulators.

Table 3-2 shows specifications specific to the E2 emulator.

Support for some debugging-related functions also depends on the debugger. Refer to the user's manual, etc. for the debugger you are using.

Table 3-1 Specifications Common to the E1, E20, and E2 Emulators

Broad Category	Medium Category	Narrow Category	Specification
Hardware in general	Corresponding host machine		Computer equipped with a USB port, OS depends on the debugger
	User system interface		14-pin connector
	Host machine interface		USB 2.0 (full speed or high speed)
	Connection to the user system		Connection by the provided user system interface cable
	Power supply function (only when the emulator is an E1 or E2)		3.3 V or 5.0 V (with current up to 200 mA) can be supplied from TVDD to the user system (make settings with the debugger)
	Power supply for the emulator		No need (the host computer supplies power through the USB)
Debugging-related items	Break	Software break	In ROM and RAM areas combined: 2000 points
		Hardware break	12 points including those used for both execution and CPU access conditions (8 points only for execution conditions, and 4 points for either execution or access conditions)
		Event break	Available
		Forced break	Available
		Trace-full break	Available (internal trace memory and E2 storage)
		External trigger input break	Available (E2 emulator only)
	Event	Number of events that can be set	8 points for execution, 8 points for CPU access, 4 points for DMA access, and 4 points for GRAM access
		Available function	Break, trace, performance measurement
		Combination of events	OR, sequential
	Tracing (only for devices including an internal trace RAM)	Destination for storage	Internal trace memory
		Size	Branch only: 1,000 branches Data trace only: 1,000 cycles of access Software trace only: 1,000 to 2,000 instructions
		Traced data	Branches, cycles of data access, cycles of DMA access, cycles of GRAM access, and software trace
		Conditions to start and stop recording of data	Stopping of program execution, event condition settings
		Data-trace conditions	Event conditions
		Priority of trace acquisition	Real-time trace mode (priority given to speed) Non-real-time trace mode (priority given to data)
		Recording of trace memory	Ring mode (overwriting mode) Trace-full stop mode Trace-full break mode Halting tracing due to the input of an external trigger (E2 emulator only)

Broad Category	Medium Category		Narrow Category	Specification
	Perfor-mance mea-sure-ment	Time (1)	Measurement section	From run to break
			Item measured	Execution time*4
			Performance	32-bit counters
		Time (2)	Measurement section	From run to break, or between two event points
			Items measured	Execution time, total execution time, pass count, maximum execution time, minimum execution time*4
			Performance	32-bit counters (for three sections)
		Other than time	Items measured	Number of instructions executed (all or branches only), number of interrupts accepted (EI level or FE level), number of exceptions accepted (instruction asynchronous or instruction synchronous), clock cycles (all, while interrupts are inhibited, or other than for the processing of interrupts), number of instruction fetches requested, number of hits on the instruction cache
			Measurement section	From run to break, or between two event points
			Items measured	Latest value, total value, pass count, maximum value, minimum value
			Performance	32-bit counters (for four sections)
	Pseudo real-time RAM monitoring			Available (occupies a bus (steals cycles))*1
	Direct memory modification			Available (occupies a bus (steals cycles))*1
	Debugging console			Not available
	Downloading of the external flash memory			Not possible
	Hot plug-in			Possible (To use with the E1 emulator, requires a separately sold hot plug-in adapter)
	Peripheral breaks			Available*2
	Emulator detection by user programs			Available*3 Debugging startup register Initial value: 0000 0000_H Address: FA00 2078_H (CPU1) FA00 2078_H (CPU2)
	Security			16-byte ID code authentication
	Security ID settings			Not available
	Security flag settings			Not available
	Activating the settings of the Intelligent Cryptographic Unit (Master/Slave type) (ICUM/ICUS)			Not possible
	Connection interface			1-pin LPD (500 Kbps/1 Mbps/2 Mbps) or 4-pin LPD (5.5 MHz/11 MHz)
Programming-related items	Security ID settings			Available
	Security flag settings			Available
	Activating the settings of the Intelligent Cryptographic Unit (Master/Slave type) (ICUM/ICUS)			Possible
	Connection interface			2-wire UART, 1-wire UART

Note 1: Only available for the local RAM, retention RAM, and global RAM areas.

Note 2: The function to stop peripheral I/O operation in a break is called the peripheral break function.
Whether peripheral emulation functions are set or not is determined by the debugger.
Refer to the manual for the debugger you are using for how to set them.
Refer to the manual for the MCU you are using to check whether peripheral emulation functions are set.

Note 3: For this function, any 32-bit value which is debugging information from the debugger is specified and held in the debugging startup register while the emulator is connected. This function can be used to determine the state of the emulator being connected or not from within user programs (refer to cautionary note No.43 of section 4.2).

Note 4: The resolution of the measured times depends on the interface used for the connection (e.g., 90.9-nsec resolution for a 4-pin LPD connection running at 11 MHz).

Table 3-2 Specifications Specific to the E2 Emulator

Broad Category	Medium Category	Narrow Category	Specification
Debugging-related items	Software tracing (LPD output)*1	Target CPU	Selection of a single CPU. For multiple-core devices: When the debugger is connected to the emulator, a single target CPU is selected. If the target CPU is changed, the debugger must be re-connected to the emulator (only available in the synchronous debugging mode).
		Destination for storage	"E2 storage": memory for storage in the E2 emulator
		Internal buffer	Eight stages*4
		Traced data	Software trace data + timestamps (given by the E2 emulator)*2 Resolution: 8.333 ns, maximum 27 days
		Conditions to start and stop recording of data	Starting and stopping of program execution (breaks)
		Priority of trace acquisition	Real-time trace mode (priority given to speed)
		Recording of trace memory	Ring mode (overwriting mode) Trace-full stop mode Trace-full break mode
	External trigger input/output*1	Input signal channels	E2 expansion interface: 2 ch. 0: pin 11, ch. 1: pin 12
		Output signal channels	E2 expansion interface: 2 ch. 0: pin 9, ch. 1: pin 10
		Interface voltage	When the emulator is not supplying power to the user system: TVDD voltage Any voltage between 1.8 V to 5.0 V When the emulator is supplying power to the user system: Voltage being supplied to the user system
		Conditions for detection of trigger inputs	Edge detection (rising, falling, or both edges) Level detection (low or high)
		Operation when a trigger is input	When software tracing (LPD output) is in use: Break When software tracing (LPD output) is not in use: Break or stopping of recording in internal trace memory
		Condition for detection of trigger outputs	Break detection*3
		Operation when a trigger is output	Output of a low or high pulse (for from 1 μsec to 65535 μsec) can be specified.

Note 1: When the software tracing (LPD output), external trigger input, or external trigger output functions are in use, access to memory during the execution of a program, changes to event conditions, the reading of internal trace memory, and the display of state indicators such as STOP are disabled.

Note 2: A timestamp indicates the time that the E2 emulator acquires the software tracing data, not the time the instruction in the software being debugged was executed. The E2 emulator requires execution of the program by the MCU to start only after it has started counting its timestamp values. Since the start of counting of timestamp values cannot be precisely synchronized with the start of program execution, the timestamps which have been

added to the software tracing data stored from the head of the E2 storage may include some errors.

Note 3: In general, when the software tracing (LPD output) function is not in use, breaks are not detectable during the 10-µsec period after a program has started to run, but the period becomes 100-µsec when the low-speed internal oscillator (LS IntOSC) is the operating clock for the emulator.

Note 4: The output of the combination of a PC value and the corresponding immediate or register value uses one stage of the internal buffer. When software tracing data have been stored up to the seventh stage of the internal buffer, an overflow message is stored in the eighth stage.

3.1 Overview of specifications specific to the E2 emulator

3.1.1 Software tracing (LPD output)

Devices of the RH850 family support debugging instructions for the output of software trace data. Software trace data are stored in the internal trace memory of the device and output to the emulator via the LPD pins, which is the debugging-connection interface. Unlike conventional tracing, the software tracing function does not cater for the setting of events or conditions so that trace data are output when the settings match the results of program execution; instead, this function helps the user to embed debugging instructions in the program to be executed as checkpoints or for the purpose of the output of specific information or register values and output of the history execution to the emulator side as trace data. Make use of this function as a new way of debugging. The debugger of CS+ provides useful functionality for applying this software tracing function (via the LPD interface). For details, refer to the user's manual and application note for CS+.

For details of the debugging instructions, refer to the RH850 G3M/G3MH/G3K/G3KH User's Manual: Debugging Instructions. Table 3-3 gives an overview of these instructions.
When the emulator is not connected and the debugging instructions embedded in a program are executed, Software trace data are not output from the LPD interface.

Table 3-3 Debugging Instructions for Software Tracing

Debugging Instruction	Function	Interval between Execution of the Embedded Instruction and the LPD Output (*)	
		4-pin LPD (11 MHz)	1-pin LPD (2 Mbps)
DBCP	Outputs the current PC value as software trace data.	5.182 usec	28.500 usec
DBTAG imm10	Outputs a 10-bit immediate (imm10) value as software trace data. Output of the PC value is also selectable.	1.727 usec (without the PC value)	9.500 usec (without the PC value)
DBPUSH rh-rt (General-purpose registers are specified as rh ≤ rt (in ascending order).)	Outputs the register numbers and values of general-purpose registers from rh to rt as software trace data. Output of the PC value is also selectable.	5.182 usec (Output one register without the PC value)	28.500 usec (Output one register without the PC value)

(*) This item indicates the time required for the LPD output of software trace data generated by executing a debugging instruction. When this interval follows the execution of a debugging instruction, overflows (losses) of software trace data can be avoided. Even if the debugging instruction is executed with a short interval, the device has an internal buffer for tracing and an overflow (a loss of data) will not occur immediately; however, note that an overflow occurs if the internal buffer becomes full. For DBPUSH instruction, set the total number of the registers to less than 5 to avoid an overflow.

3.1.2 External trigger input and output

Using the expansion interface of the E2 emulator (the connector for the interface can be found by removing the cover on which SELF CHECK is printed) enables the input and output of external triggers. For details on the function, refer to Table 3-2. For details on the expansion interface, refer to the E2 Emulator User's Manual.

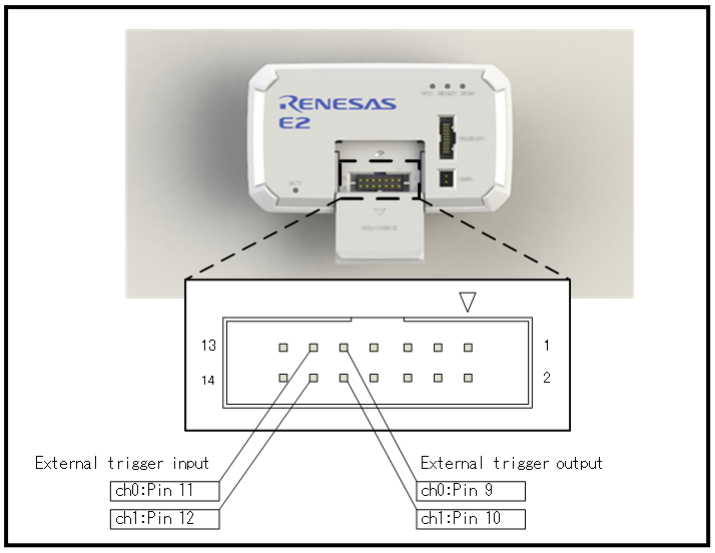

Figure 3-1 Expansion Interface of the E2 Emulator

4. Notes on Usage

Cautionary notes on using the E1, E20 or E2 emulator are given below.

4.1 Notes on differences in operation between the actual device and the E1, E20 or E2 emulator

No.1 DBTRAP instruction

The DBTRAP instruction is used for software break functions and thus cannot be used in programs with the emulator.

No.2 Serial programming function

The serial programming function cannot be used with the emulator during debugging.

No.3 HALT mode (skipped number)

The information that was previously under this number has been integrated into section 4.2 No.16.

No.4 Current drawn

More current is drawn when an emulator is connected than when it is not connected. That is, the target device consumes more power during debugging than in normal operation since the debugging functions are operating.

No.5 Initialization of RAM areas

When an emulator is connected, local RAM, retention RAM, global RAM, and FCU-RAM areas are initialized to 0000 0000_H. This leads to the following differences from the actual device.

- The initial values in the RAM area after starting an emulator are different from the initial values (undefined values).
- ECC errors due to non-initialization of RAM are not detected with the emulator. If the emulator is not connected and the operation is incorrect, check that the RAM areas have been initialized.

To emulate ECC errors, set the following options.

- The RAM area is not initialized when the emulator is started.
- OPJTAG is not set for an LPD connection before the emulator is connected.

However, if a RAM area is not initialized, the following functions are not available.

- Downloading to on-chip flash memory
- Changes to on-chip flash memory by using the [Memory] panel or the [Disassemble] panel
- Setting of software breaks
- Rewriting of the option byte

No.6 OTP flag

Do not set the one-time programming (OTP) flag in self-programming with the emulator. Note that setting of the flag makes downloading from the debugger to flash memory impossible.

No.7 Operations in response to resets and interrupts when an emulator is in use (skipped number)

The information that was previously under this number has been integrated into section 4.2 No.14 and No.15.

No.8 Option byte register

The debugger cannot write new values to the bits of the option byte register indicated below since they are used by the emulator. Also, do not attempt self-programming to write new values to these bits.

- OPJTAG1 and OPJTAG0 bits (bits 30 and 29 of the OPBT0 register)

The value of the OPJTAG1 and OPJTAG0 bits while an emulator is connected is "01B" if the 4-pin LPD interface is selected and "10B" if the 1-pin LPD interface is selected.

4.2 Cautionary notes on debugging

No.1 Handling of devices which were used for debugging

Do not use devices that were used for debugging in mass-production. This is because writing to the flash memory of such devices has already proceeded during debugging, so we cannot guarantee the number of times rewriting of the flash memory can proceed. Debugger errors occur when programming of the flash memory is no longer possible. Replace the device in such situations.

No.2 Power to the target system while debugging

Do not turn the power to the target system off during debugging. Doing so will require reconnection of the debugger.

No.3 Hardware break (access) function (the timing of a break occurring)

When the hardware break (access) function is in use, a break in response to the reading or writing of specified data by a read–modify–write instruction will occur after the instruction. Other hardware breaks (access) occur before the instruction.

No.4 Hardware break (access) function (forms of read or write access that are not detectable)

In general, no break occurs even if a read or write access condition is satisfied by the instructions below.

(1) CAXI, SET1, CLR1, NOT1, and TST1

(2) PREPARE, DISPOSE, PUSHSP, POPSP, SWITCH, CALLT, and SYSCALL

However, the detection of reading by the instructions in (1) is possible, but only if no data condition is specified.

No.5 Hardware break (access) function (EIINT table)

Do not set the address of the EIINT table as a hardware break condition. If a break occurs, it will not be possible, in some cases, to return from the interrupt processing even if EIRET is executed.

No.6 Multiplexed functions of pins used for OCD signals

Multiplexed functions of pins used for on-chip debugging (OCD) cannot be used during debugging.

No.7 Connection interface during debugging

The E1, E20 and E2 emulators support both 1- and 4-pin LPD interfaces.

Operation is as follows if the setting of the OPJTAG1 and OPJTAG0 bits of the option byte 0 register is "11B" (JTAG: the JTAG interface is selected in the case of a blank chip).

a. When starting (connecting) the E1, E20 or E2 emulator

Settings of the option byte 0 register are changed from the setting for JTAG to that for 1- or 4-pin LPD by the debugger on connection to an emulator.

Therefore, the OPJTAG1 and OPJTAG0 bits of the option byte 0 register are either “10B” (1-pin LPD) or "01B" (4-pin LPD) during emulator operation.

b. When exiting from a session with (disconnecting) the E1, E20 or E2 emulator

Settings of the option byte 0 register can be changed by the debugger.

- The value of the OPJTAG1 and OPJTAG0 bits of the option byte 0 register can be changed to "11B" (for JTAG), which requires rewriting of the flash memory.
- The setting of the OPJTAG1 and OPJTAG0 bits of the option byte 0 register can be left as "01B" (for 4-pin LPD) or "10B" (for 1-pin LPD).

When LPD interface is also used the next time the emulator is connected, we recommend exit from the program without changing the settings from that for the LPD interface.

If power to the target system is turned off because of an abnormal end to the emulator session, the OPJTAG1 and OPJTAG0 bits of the option byte 0 register are not rewritten and so retain the value "01B" (for 4-pin LPD) or "10B" (for 1-pin LPD). If you wish to change the OPJTAG1 and OPJTAG0 bits of the option byte 0 register to "11B" (for JTAG), please do so at the end of the E1, E20 or E2 emulator session.

No.8 Initialization of RAM areas

All RAM areas for use by a program must be initialized when an emulator is in use. Before the emulator is used, if any setting is made to initialize the RAM area when the emulator is started, ECC errors are not generated since the debugger initializes the RAM area. However, when the actual device is operated with a program which does not initialize the RAM area, ECC errors will be generated, preventing normal program operation.

ROMization is also required because any data downloaded from the emulator to the RAM area before program execution will also be initialized. For details, refer to the user’s manual for the compiler you are using.

No.9 Reset of pins (skipped number)

The information that was previously under this number has been integrated into section 4.2 No.14.

No.10 Trace function [when a device with a trace function is in use]

The following restrictions apply to the trace function.

- Writing of data in the form of pushing by executing the PUSHSP or PREPARE instruction might not be traced.
- In the case of section trace, for example, the instruction immediately before the fetched instruction that actually caused tracing to start might be included in trace data.
- In some cases, acquired trace information will be lost. This depends on the program being executed. The lost information cannot be restored, but the fact of the loss is indicated (displayed). Information is lost when access to data by the CPU is continuous and frequent.
- When priority in tracing is given to non-realtime operation, the functions to stop tracing when the trace memory becomes full (trace-full stop function) and when a specified number of trace messages have been acquired following an event (trace delay-stop function) are not available. To use these functions, give priority to realtime operation.
- When data-qualified tracing (point tracing), i.e. tracing only of data in access to a specific address, is specified, tracing proceeds with any data conditions ignored, even if read access conditions are set. Tracing is still governed by conditions other than data conditions.

No.11 Power-saving modes

When a power-saving mode is in use, the restrictions below apply.

- For debugging of a program, ensure that the program sets WUFMSK0[31] to 0.
- Release from the STOP, DeepSTOP, or CyclicSTOP mode follows any of the following operations or conditions while the user program is being executed.
  - Break
  - Memory access
  - Forcibly stopping tracing
  - Setting an event
  - Enabling of the trace-full stop or trace-delay stop functions
- The power supply to the Iso area (CPU, RAM, peripheral module, etc.) is not stopped in the DeepSTOP mode during debugging. For this reason, RAM or registers which have undefined initial values retain these values. Be sure to initialize them after returning to the RUN mode.

No.12 Quality of flash programming

To improve the quality, follow the guidelines below.

- Circuits are designed as described in the user's manuals for the MCU and E1, E20 or E2 emulator.
- The MCU, E1, E20 or E2 emulator, and the software are used as described in respective user's manuals.
- The supply of power to the user system is stable.

No.13 Turing the power on/off

Turn the power of the E1, E20 or E2 emulator and the user system following the procedure below.

- When a separate power supply is used for the user system

<When using the emulator>

(1) Check the power is off.
Check that the user system is turned off. When using the E20 emulator, check its power switch is 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 E1 or E2 emulator is turned on by connecting the USB interface cable. When using the E20 emulator, turn on its power switch.

(4) Turn on the user system.
Turn on the user system.

(5) Launch the debugger.
Launch the debugger.

<When finished using the emulator>

(1) Close the debugger.
Close the debugger.

(2) Turn off the user system.
Turn off the user system.

(3) Turn off the emulator and disconnect the emulator.
When using the E20 emulator, turn off its power switch. Disconnect the USB interface cable from the E1, E20 or E2 emulator. The E1 or E2 emulator is turned off by disconnecting from the USB interface cable.

(4) Disconnecting 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, unplug the USB interface cable, or turn off the power switch of the E20 emulator.
The user system may be damaged due to leakage current.

- When power is supplied to the user system from the emulator (E1 or E2 emulator)

<When using the emulator>

(1) Check the power is off.
Check that the user system is turned off.

(2) Connect the user system.
Connect the emulator and user system with a user-system interface cable.

(3) Connect the host machine and turn on the emulator.
Connect the emulator and host machine with a USB interface cable, then turn on the emulator.

(4) Launch the debugger.
Launch the debugger and select the setting of power supply to the user system.

<When finished using the emulator>

(1) Close the debugger.
Close the debugger.

(2) Turn off the emulator and disconnect the emulator.
Disconnect the USB interface cable from the emulator, then turn off the emulator.

(3) Disconnecting the user system.
Disconnect the user-system interface cable from the user system.

No.14 Resets when an emulator is in use

Table 4-1 shows the states of the device when an emulator is in use and the operation in response to a reset (i.e. a user-system reset) issued by the user system or the user program. During single-stepped execution, the emulator masks the user-system reset so that it can continue to emulate the source code of the program line-by-line rather than in realtime. For C-source-level stepped execution, the reset is masked in different ways depending on the debugger; single-stepped execution is used or the user program is executed by setting temporary breakpoints. Accordingly, this user's manual cannot define whether a reset is masked by the emulator or not, so refer to the user's manual for the debugger you are using.

Table 4-1 State of the Device and Masking of User-system Resets by the Emulator

		State of the device			
		During a break	**Single stepping**	**Executing the user program**	**Stepping at C-source level**
Reset mask specification of the debugger	**Not masked**	Masked*		Not masked	Depends on the debugger
	Masked	Masked*			

- When a reset is issued by the debugger (by using a reset button of the debugger, etc.), the CPU is always reset regardless of whether the masking of resets is currently enabled or disabled. After a reset from the debugger, breaks are generated in all CPUs.
- Resets generated in the states marked (*) in table 4-1 will be discarded. For example, when a setting for software-reset processing is made during single-stepped execution or a software reset by setting a register is applied by the debugger during a break, the reset is not applied.
- Do not allow the generation of a reset in the form of a pin reset from the target system other than while a program is in execution regardless of the presence of masking as described above. A reset generated while the program is running may cause the debugger to hang.

No.15 Interrupts when an emulator is in use

Table 4-2 shows the states of the device when an emulator is in use and the operation in response to an interrupt. During single-stepped execution, the emulator masks interrupts so that it can continue to emulate the source code of the program line-by-line rather than in realtime. For stepped execution of interrupt processing, set a breakpoint at the start of the interrupt service routine, then generate an interrupt during the execution of a user program so that the break at the start of the interrupt service routine is applied. For C-source-level stepped execution, interrupts are masked in different

ways depending on the debugger; single-stepped execution is used or the user program is executed by setting temporary breakpoints. Accordingly, this user's manual cannot define whether interrupts are masked by the emulator or not, so refer to the user's manual for the debugger you are using.

Table 4-2 State of the Device and Masking of Interrupts by the Emulator

State of the device			
During a break	**Single stepping**	**Executing the user program**	**Stepping at C-source level**
Interrupts masked*		Interrupts not masked (operation is according to the settings of the user system)	Depends on the debugger

- Interrupts (EIINT, FEINT, and FPI) which have been generated in the state marked (*) in table 4-2 are held pending and interrupt processing is performed after the interrupt mask is removed.

No.16 HALT mode and stepped execution of the HALT instruction

If a break occurs, the device is released from HALT mode.

When a HALT instruction is encountered during single-stepped execution (execution in units of assembly instruction), a break is set at the next instruction following the HALT instruction, and the mode does not change to the HALT mode.

When a HALT instruction is encountered during C-source-level stepped execution, whether or not the transition to the HALT mode proceeds depends on the facilities of the debugger.

No.17 Stepped execution of an instruction which would lead to a transition to the DeepSTOP mode on the actual device

Stepped execution has two variants: Single step execution (execution in units of assembly instructions) and C-source-level stepped execution (execution in units of statements or functions in C language source code). When an instruction which would lead to a transition to the DeepSTOP mode on the actual device is executed during single step execution, the program breaks at the address at the time of the reset and does not switch to the DeepSTOP mode. When an instruction which would lead to a transition to the DeepSTOP mode on the actual device is executed during C-source-level stepped execution, whether or not the transition to the DeepSTOP mode proceeds depends on the facilities of the debugger.

No.18 Cautionary note on connecting an emulator (pin reset)

The reset signal continuing to be asserted while communications between the emulator and MCU are being prepared when the emulator is connected causes incorrect communications. Thus, ensure that the reset signal does not remain asserted when the emulator is connected.

No.19 Cautionary note on connecting an emulator (time required for preparing to communicate)

When an emulator is connected, a program which was written to the MCU is executed from the reset vector before the emulator and MCU become able to communicate. Take care on this point.

When debugging of a program written to the MCU creates a problem, eliminate the problem by inserting a waiting time* indicated below before executing the program following release from the reset state.

For the 1-pin LPD interface, waiting for at least 140 ms is required.

For the 4-pin LPD interface, waiting for at least 12 ms is required.

Note: Time required for preparing communications depends on the host PC environment of the E1, E20 or E2 emulator and the operating frequency of the MCU.

No.20 Cautionary note on connecting an emulator (internal reset)

When the stored program generates an internal reset (software reset or reset caused by the watchdog timer overflowing) immediately after a release from the initial reset state, the internal reset may be generated before communications between the emulator and MCU have been established after the emulator is connected, causing incorrect communications.

Accordingly, insert a waiting time* indicated below before applying an internal reset after a release from the initial reset state when debugging a program which includes an internal reset.

For the 1-pin LPD interface, waiting for at least 140 ms is required.

For the 4-pin LPD interface, waiting for at least 12 ms is required.

Note: Time required for preparing communications depends on the host PC environment of the E1, E20 or E2 emulator and the operating frequency of the MCU.

No.21 Cautionary note on connecting an emulator (DeepSTOP mode)

When the program for a mode transition to the DeepSTOP mode is stored immediately after a release from the initial reset state, the MCU switches to the DeepSTOP mode before communications between the emulator and MCU are established when the emulator is connected. This causes incorrect communications. Accordingly, insert a waiting time* indicated below before executing an instruction which changes the mode to the DeepSTOP mode after a release from the initial reset state when debugging a program that changes the mode to the DeepSTOP mode.

For the 1-pin LPD interface, waiting for at least 140 ms is required.

For the 4-pin LPD interface, waiting for at least 12 ms is required.

Note: Time required for preparing communications depends on the host PC environment of the E1 or E20 emulator and the operating frequency of the MCU.

No.22 Access to I/O resources in the MCU

Access to I/O resources (registers and RAM) in the MCU by the debugger (i.e. access through the memory or I/O register window) proceeds in the same way as access from a user program.

Examples (for the actual operation of I/O resources, refer to the manual of the MCU you are using)

- Access to FCU-RAM resources

  Normal access will not proceed unless the FCU-RAM enable bit is set.
- Access to the PBG area

  Attempted access to the PBG area will not proceed while the guard is enabled.

No.23 Cyclic run mode and cyclic stop mode

This item only applies to the F1H group.

When the microcontroller enters the cyclic run mode or cyclic stop mode, the core other than CPU1 is stopped. Debugging must only proceed as asynchronous debugging. That is, synchronous debugging is not usable in this situation. Also, since the core other than CPU1 is stopped during asynchronous debugging, it cannot be debugged.

No.24 Cautionary point regarding hot plug-in connection

- When the OPJTAG [1:0] bits of the option byte register are not set for the LPD operation mode at the time of hot plug-in connection, a connection error occurs. Thus, before proceeding with hot plug-in connection, set the OPJTAG [1:0] bits for the LPD operation mode.
- Allowing hot plug-in connection prevents usage of the optional isolator for the E1 emulator (the isolator is only for use with the RH850 and RL78 groups).
- Allowing hot plug-in connection prevents the supply of power to the user system by the E1 or E2 emulator.
- If hot plug-in connection is not to be used, the RAM area will be initialized* when the emulator is started. Allowing hot plug-in connection prevents this initialization and makes the masking of pins impossible. Thus, when the emulator is started without initializing the RAM area to be used by a program, ECC errors occur. Therefore, make sure to initialize the RAM area to be used by a program before setting up the system for hot plug-in connection.
- After completing hot plug-in connection, the user program will be running. At this time, only the emulator functions listed below are available.

  Read or write access to the internal RAM area

  Forced break

  CPU reset

  Apply a forced break if you wish to return to using all functions supported by the emulator. After the forced break, functions equivalent to those that can be used after normal starting of a program become available.

Note: The RAM area is only initialized if the setting is made to initialize the RAM area when the emulator is started.

No.25 Cases where hot plug-in connection is not possible

Hot plug-in connection cannot be used when the microcontroller is in any of states listed below.

Reset input state

Cyclic run mode

Cyclic stop mode

No.26 Standby mode released by hot plug-in connection

Hot plug-in connection releases the microcontroller from standby when the microcontroller is in any of states below.

Stop mode

DeepSTOP mode (A reset will occur due to the specifications of the device.)

No.27 Cautionary note on asynchronous debugging mode (peripheral break function)

This item only applies to the F1H group.

In the asynchronous debugging mode, peripheral break functions cannot be used. Even if peripheral break functions are enabled, peripheral functions are not stopped.

No.28 Cautionary note on asynchronous debugging mode (reset)

This item only applies to the F1H group.

In the asynchronous debugging mode, when any of CPUs is in the break state, no resets are acceptable.

No.29 Cautionary note on asynchronous debugging mode (watchdog timer)

This item only applies to the F1H group.

In the asynchronous debugging mode, when any of CPUs is in the break state, counters are stopped in WDTA0, WDTA1, and WDTA2.

No.30 Cautionary note on asynchronous debugging mode (ECC error)

This item only applies to the F1H group.

During execution of a user program, there may be a case that the ECC error function does not normally operate for flash memory.

Example: When any CPU accesses flash memory during execution of a user program causing an ECC error and another CPU which is in the break state accesses the same resources in the memory window at the same timing, the debugger temporarily controls the ECC error and no ECC error occurs in any CPU.

No.31 Cautionary note on asynchronous debugging mode (specific sequence)

This item only applies to the F1H group.

During execution of a user program, there may be a case that the specific sequence is not satisfied.

Example: When any CPU accesses the specific I/O register during execution of a user program and another CPU which is in the break state accesses the same peripheral function in the I/O register window at the same timing, the specific sequence from any CPU is not satisfied and normal accessing is disabled.

No.32 Performance measurement

In the case of measuring a specific section, if the intervals between the start and the end of one measurement, and between the end of that measurement and the start of the next is short, the measurement is not possible. To obtain correct measurements, the interval* should be long enough.

*: The required detection interval depends on the operating frequency and the LPD communications frequency of the MCU.

No.33 Reset

If a CPU reset is generated while a program is running, debugging functions (breaks, events, traces, and timer) and the operation of the MCU may become unstable.

No.34 Rewriting of on-chip flash memory (Working RAM)

When the debugger performs any operation that involves programming of the flash memory* during a break, part of the internal RAM area is used as a working RAM area. The 4-KB area (for the E2 emulator) or the 9-KB area (for the E1 or E20 emulator) from the last address of the local RAM area of CPU1 are initially set as the working RAM area. If a device has no local RAM area, the retention RAM area is used.

The debugger can change the working RAM area. After the debugger has saved the values from the working RAM area and rewrites the flash memory, it restores the saved values to the working RAM area. To guarantee the values, it is required to set an area to which there will be no access by the DMAC or any external master to the working RAM area so that operation may continue even if the device enters the break state.

Note: Rewriting of flash memory proceeds in response to any of the operations below.

- Downloading to on-chip flash memory
- Changes to on-chip flash memory by using the [Memory] panel or the [Disassemble] panel
- Setting or cancellation of software breaks
- Re-execution after a software break is encountered (including stepped execution)

No.35 Rewriting of on-chip flash memory (clock monitor)

The debugger changes the PLL multipliers as shown below when the flash memory is rewritten*. Thus, rewriting the flash memory raises a possibility of the frequency becoming higher than that currently in use. If the frequency surpasses the upper limit which was set by the clock monitor (CLMA), this prevents rewriting of the flash memory. If the change in the clock frequency due to the debugger is a problem, set [Change the clock to flash writing] in the [Property] panel to [No].

[Changes in the PLL multiplier]
When MainOSC = 8 MHz, PLL0 is multiplied by 15 with settings for no frequency division (CPU clock at 120 MHz).
When MainOSC = 10 MHz, PLL0 is multiplied by 12 with settings for no frequency division (CPU clock at 120 MHz).
When MainOSC = 12 MHz, PLL0 is multiplied by 10 with settings for no frequency division (CPU clock at 120 MHz).
When MainOSC = 16 MHz, PLL0 is multiplied by 7.5 with settings for no frequency division (CPU clock at 120 MHz).
When MainOSC = 20 MHz, PLL0 is multiplied by 6 with settings for no frequency division (CPU clock at 120 MHz).
When MainOSC = 24 MHz, PLL0 is multiplied by 5 with settings for no frequency division (CPU clock at 120 MHz).

Note: Rewriting of flash memory proceeds in response to any of the operations below.

- Downloading to on-chip flash memory
- Changes to on-chip flash memory by using the [Memory] panel or the [Disassemble] panel
- Setting or cancellation of software breaks
- Re-execution after a software break is encountered (including stepped execution)

No.36 Breaks during execution of code for making clock settings

The flash memory cannot be programmed if a break occurs while the MCU is running code written to memory for making clock settings (setting of the main oscillator or PLL frequency divider and so on).
If you wish either of the following types of operation to proceed when a break has occurred during clock settings, set [Change the clock to flash writing] in the [Property] panel to [No].

a. Any operation that involves programming of the flash memory (e.g. re-downloading)
b. Setting or deleting software breakpoints

Also, do not set software breakpoints within code for making clock settings.

No.37 Event functions (64-bit access)

Do not set any access events with the condition in 64-bit units. The emulator may detect access in a unit other than 64 bits as satisfying such conditions or other events may not operate normally.

No.38 Event functions (in the order of event detection)

In the following cases, since the orders of instructions and event detection may not operate as set, to measure the time or performance in sequential events, section tracing, and desired sections may not be possible.

- An event is set for consecutive instructions but the two instructions are executed at the same time.
- An access event detects adjacent read and write instructions, since the timing of event detection differs in write and read access and the timing may be detected in the order of reading then writing, even though the instructions are executed in the order of writing then reading.
- Access events may be detected at the same time since bank A and bank B of the global RAM can be accessed simultaneously and up to two access events are detectable.

No.39 Event functions (bit-manipulation instructions)

When a read or write access condition is set for an event, the writing cycle of read-modify-write generated by a bit-manipulation instruction is not detected as an event. This condition cannot be used as a trigger for a break, trace acquisition, or performance measurement in the case of such instructions.

No.40 Satisfaction of two break conditions before a single break

If another read-access event is detected immediately before a transition to the break state due to a forced break or an event break, a further break occurs immediately after execution of the program is resumed because the break request was accepted as a read-access event at the time of resumption.

No.41 External wait function

The signal on the $\overline{\text{MEMC0WEIT}}$ pin for the external wait function of the external memory controller (MEMC) is not masked even if masking of it is set in the debugger.

No.42 Software break function (RAM areas)

The software break function is implemented by replacing instructions. Thus, note that no break will occur if the value at an address where a software break has been set is rewritten by a user program which is running.

No.43 Emulator detection by user programs

Even if debugging information is specified, note that the value will be initialized to 0000 0000H if a reset is generated. Debugging information is specified again when a break occurs in all CPUs and when the user program is re-executed after a reset.

No.44 Software tracing (LPD output) function when the 1-pin LPD interface is selected (only for the E2 emulator)

When the 1-pin LPD is selected and a break is generated as a forced break, a trace-full break from the E2 storage, or a break due to the input of an external trigger, software tracing (LPD output) cannot be used when execution of the program is subsequently resumed. To proceed with software tracing (LPD output) again, re-connect the emulator to the debugger.

No.45 Cautionary point regarding trace data acquired by software tracing (LPD output) (only for the E2 emulator)

When a break is generated as a forced break, a trace-full break from the E2 storage, or a break due to the input of an external trigger, information from a debugging instruction that was executed immediately before the break will not be stored in the E2 storage.

When a debugging instruction is executed during single-stepped execution and a software break or hardware break is specified and executed by the debugging instruction, software trace data are not output through the LPD interface.

When trace acquisition is stopped due to a break generated by a software break, hardware break, event break, or trace-full break from internal trace memory, the history of execution from a DBCP instruction executed in the debugging area is stored as the final trace data in the E2 storage and internal trace memory after the break in execution.

No.46 Breakpoints in the code flash P/E mode or data flash P/E mode

During debugging of a user program which makes the target device enter the code flash P/E mode or data flash P/E mode, we recommend using hardware breakpoints rather than software breakpoints.
Since flash memory cannot be programmed while the target device is in the code flash P/E mode or data flash P/E mode, software breakpoints can neither be added to nor deleted from the code flash memory. Accordingly, actually adding or deleting them on the target device is not possible. Only add or delete software breakpoints in the code flash memory after the target device is in a mode other than the code flash P/E mode or data flash P/E mode.
If a break is generated at a software breakpoint in the code flash memory while the target device is in the code flash P/E mode or data flash P/E mode, the break will be generated at the current address (that of the software breakpoint), so attempting to execute the user program will again lead to a break and the program will not run beyond the current address. In such cases, apply a reset.

No.47 Software breakpoints in the code flash memory in the cyclic run mode

Since flash memory cannot be programmed while the target device is in the cyclic run mode, software breakpoints can neither be added to nor deleted from the code flash memory. Accordingly, actually adding or deleting them on the target device is not possible. Only add or delete software breakpoints in the code flash memory after the target device is in a mode other than the cyclic run mode.

5. Internal Circuits of the Emulator

The internal interface circuits related to the communications interface between the E1 or E20 emulator, the E2 emulator and user system are shown in figures A, B, C and D below. Please refer to these figures when determining parameters in board design.

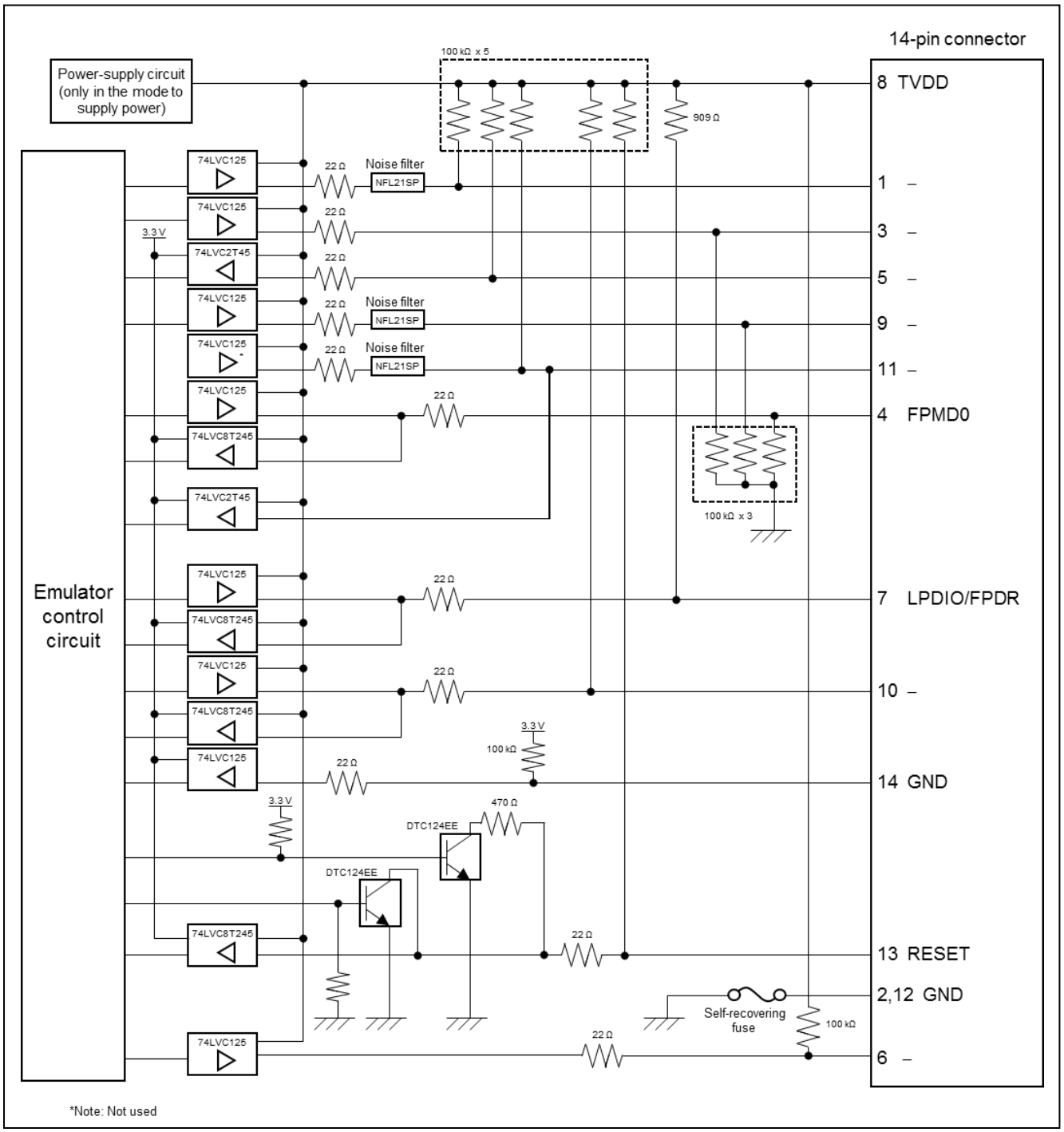

Figure A Interface Circuits in the E1 or E20 Emulator (1-Pin LPD, 1-Wire UART)

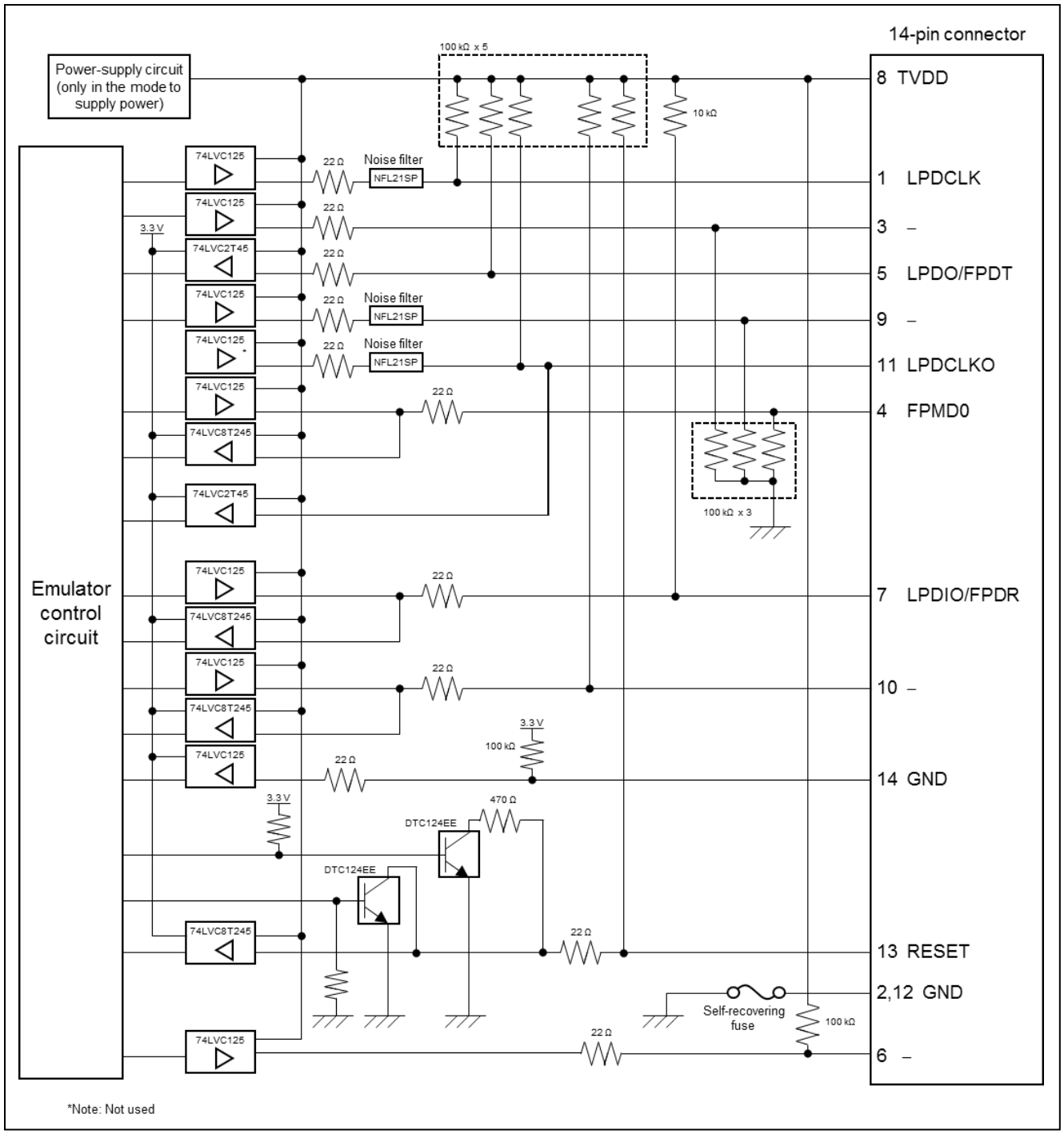

Figure B Interface Circuits in the E1 or E20 Emulator (4-Pin LPD, 2-Wire UART)

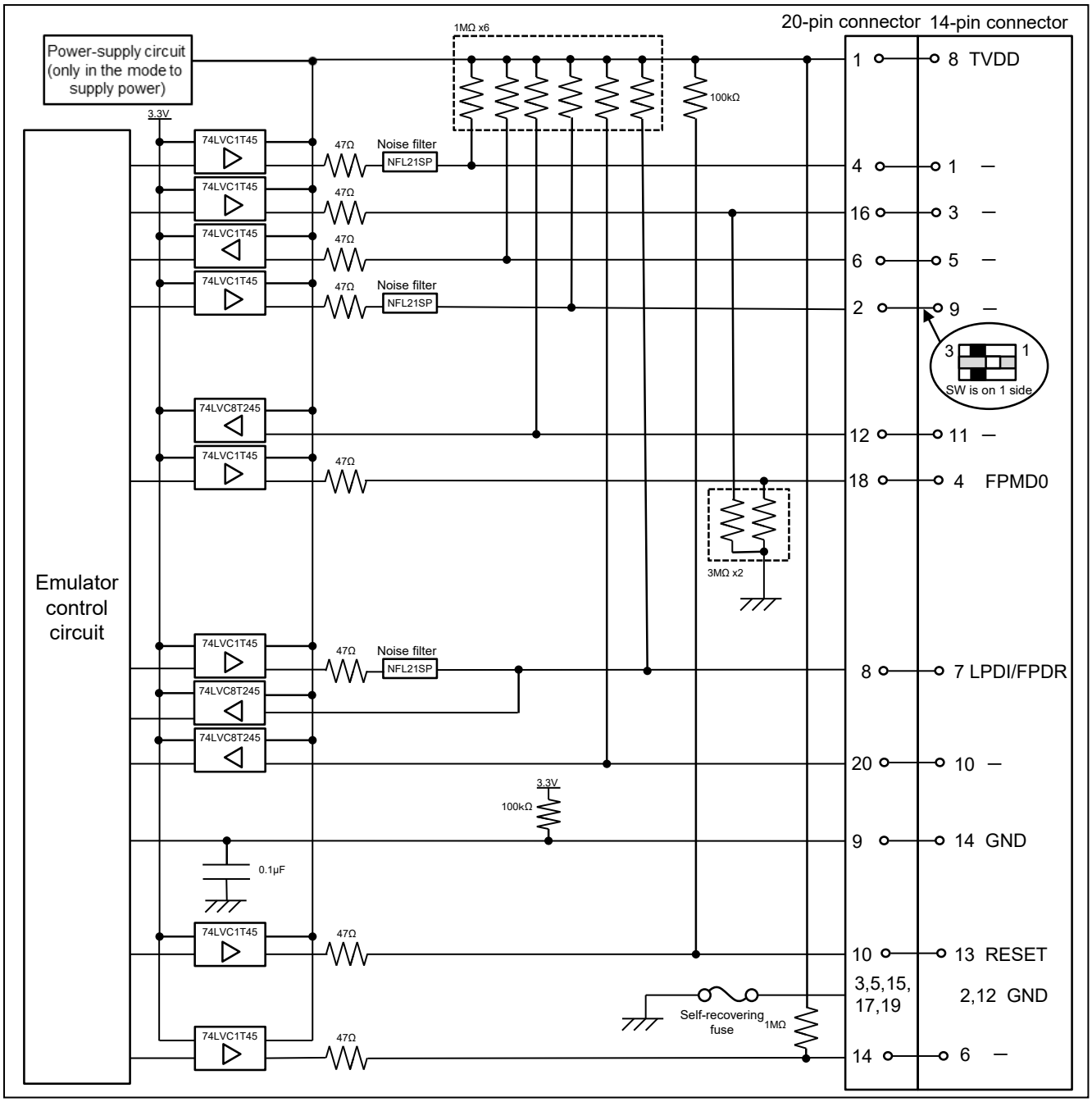

Figure C Interface Circuits in the E2 Emulator (1-Pin LPD, 1-Wire UART)

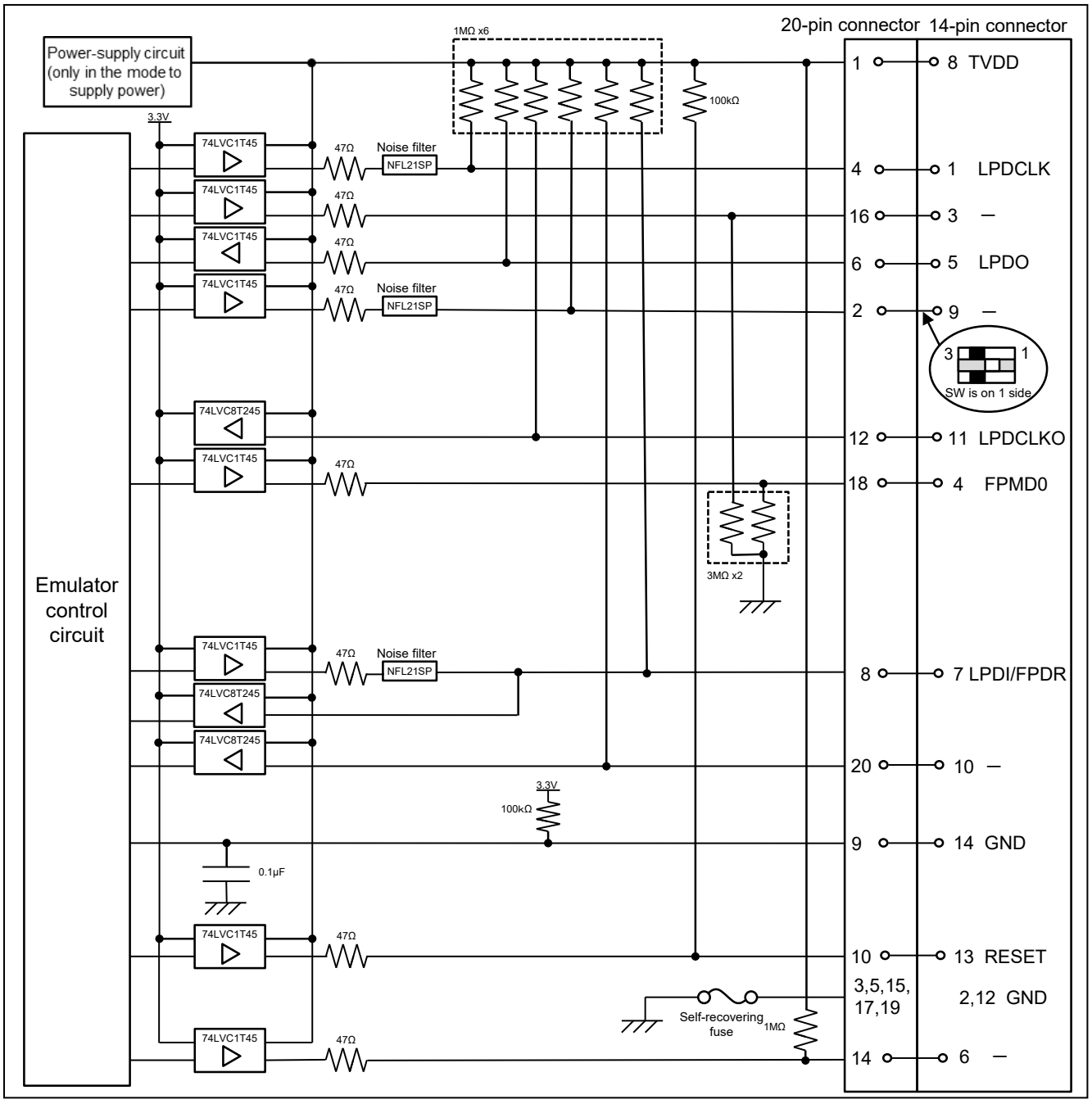

Figure D Interface Circuits in the E2 Emulator (4-Pin LPD, 2-Wire UART)

6. Troubleshooting

This chapter gives examples of problems that may arise while the E1, E20 or E2 emulator is being used in combination with a debugger and of remedies for these problems. Also read the sections of the E1 or E20 emulator user's manual, E2 emulator user's manual on the Renesas homepage, and in user's manuals for debuggers which include FAQs or information on troubleshooting. The error codes for CS+ are also listed below. If you are using a debugger other than that of CS+, refer to the user's manual for the given debugger.

6.1 Problems when the emulator is connected

Table 6-1 Problems when the Emulator is Connected (1/2)

Problem	Remedy	Error Code in CS+
Inability to connect with the debugging tool (emulator) This error may occur in the serial programming mode.	When OPJTAG automatic setting is enabled for the setting of the debugger, switch the device to the serial programming mode when it is connected and check and change the value of the OPJTAG bit in the option byte (see section 2.3). If this fails, the error message shown at right will appear. Check the following items. • Control of pin resets for the transition to the serial programming mode may be wrong. When an emulator is connected, do not input a reset signal to the pin on the circuit other than from the emulator. Check the notes (e.g. the time the signal takes to reach the high level from the low level) given in section 2.4.2 or whether the electrical characteristics requirements of the reset pin of the device are satisfied. • The connection between the emulator and the target device may be wrong. Refer to section 2.4.1, Example of recommended connections, and check the circuit between the emulator and the target device. • Check that mode pins such as FLMD1, which are not controlled by the emulator, are being handled in ways that allow transitions to the serial programming mode.	E1203237
	• The value set for MainOSC may be wrong. Check whether the frequency of MainOSC on the board matches the value set for connecting the debugger.	E1203275
	• The connection between the emulator and the target device (particularly that of the FLMD0 pin) may be wrong. Refer to section 2.4.1, Example of recommended connections, and check the circuit between the emulator and the target device.	E1203276

Table 6-2 Problems when the Emulator is Connected (2/2)

Problem	Remedy	Error Code in CS+
Inability to connect with the debugging tool (emulator) Error in LPD connection	• The OPJTAG bit in the option byte may not be specifying the correct connection interface (LPD). Enable OPJTAG automatic setting as the setting for the debugger to allow rewriting of the option byte when the emulator is started or use a flash programmer (e.g. the RFP) to change the value of the OPJTAG bit before connecting the debugger. • The condition in cautionary note No.20 in section 4.2 on the time required for preparing communications before the emulator is connected to the target device may not be being satisfied. Use a flash programmer (e.g. the RFP) to erase the code flash memory and check whether this makes the emulator connectable to the target device. • When the emulator is connected other than with a hot plug-in connection, although the emulator controls the pin reset, this may fail. Check the notes (e.g. the time the signal takes to reach the high level from the low level) given in section 2.4.2 or whether the electrical characteristics requirements of the reset pin of the device are satisfied. • The connection between the emulator and the target device may be wrong. Refer to section 2.4.1, Example of recommended connections, and check the circuit between the emulator and the target device. • The specifications for communications may not be being satisfied due to the state of the target board. Set the LPD transfer rate to a low rate and check whether the emulator can then be re-connected. • The value of the option byte may not be correct. Check that the value of the option byte has been specified with suitable settings according to the hardware manual for the MCU in use by using a Flash Programmer (RFP, etc.).	E1203240
	• The RESET pin of the target device may be active. Make sure that the RESET pin is at the inactive level during connection of the emulator.	E1203274
Inability to connect with the debugging tool (emulator) Non-matching of security IDs	• ID authentication may fail when the debugger is connected. Check that the entered ID code is correct. • The condition in cautionary note No.21 in section 4.2 on the time required for preparing communications before the emulator is connected to the target device may not be being satisfied. Use a flash programmer (e.g. the RFP) to erase the code flash memory and check whether this makes the emulator connectable to the target device.	C0602202

6.2 Problems after the emulator is connected

Table 6-3 Problems after the Emulator is Connected

Problem	Remedy	Error Code in CS+
Inability to generate breaks	• The reset signal may have been at the active level for a long time. If a reset is input for more than 8 seconds, forced breaks will be disabled. Wait for the end of the reset input or change the setting for masking resets. • Supply of a clock signal to the CPU may have stopped. Do not stop supply of a clock signal to the CPU. A forced reset may restart supply of a clock signal to the CPU and cause a break (to do this with CS+, set the [Use the force reset] option to [Yes]). • The condition in cautionary note No.11 in section 4.2 on the power-saving modes may not be being satisfied. Wait for the target device to be switched to the normal mode or cancel masking of resets and input a reset to the pin on the target board. During debugging, be sure to set a wake-up source in the WUFMSK0 register.	E1200674

Revision History	E1/E20 Emulator, E2 Emulator Additional Document for User's Manual (Notes on Connection of RH850/F1H and RH850/F1M)

Rev.	Date	Description	
		Page	Summary
4.00	Oct. 01, 2015	—	First Edition issued
5.00	Jul. 01, 2016	6	A point for caution on unused pins was added.
		7	Descriptions were added to section 2.3, Connection interface and modes.
		9-13	The value of the pull-down resistor for FLMD0 was changed (to relax the specification of the target device).
		14	Indications of pull-up resistors in figure 2-7, Example of Connecting a Reset Circuit, were deleted.
		14	Descriptions related to pull-up resistors in the emulator and pull-up or pull-down resistors in the target device were added.
		17	The addition of support for software breaks in the RAM area was stated.
		18	A note on areas supporting pseudo real-time RAM monitoring and direct memory modification was added.
		18	Items on security ID settings, security flag settings, and activating the settings of the ICUM/ICUS were added.
		21	Item No. 4 was newly added.
		21	Item No. 5 was newly added.
		23	The PREPARE instruction, for which writing of data is not traced, was added to the description of No. 10.
		23	In the description of No. 10, the trace-delay stop function, i.e. a function prohibited for used in non-realtime tracing, was added.
		23	In the description of No. 10, a note on data-qualified tracing was added.
		23	A restriction on release from the STOP, DeepSTOP, and CyclicSTOP modes was added to item No. 11 (transcribed from CS+ Release Note R20UT3117EJ0400).
		26	The mention of the DTC-RAM among the examples in No. 22 was deleted. Wrong notation was modified (PBG guard area -> PBG area).
		27	The statement of hot plug-in connection not being usable for the DeepSTOP mode in the F1H group was removed from No. 25.
		27	A statement that products of the F1H group enter DeepSTOP mode as the standby mode on release due to a hot plug-in connection was added to No. 26. This meant that there is no difference between the F1H and F1M groups on this point, so the descriptions were made the same.
		28	Item No. 33 was newly added.
		29	Item No. 34 was newly added.
		29	Item No. 35 was newly added (transcribed from CS+ Release Note R20UT3117EJ0400).
		30	Item No. 36 was newly added (transcribed from CS+ Release Note R20UT3117EJ0400).
		30	Item No. 37 was newly added.
		30	Item No. 38 was newly added.
		30	Item No. 39 was newly added.
		30	Item No. 40 was newly added.
		30	Item No. 41 was newly added.
		30	Item No. 42 was newly added.

		31	A power-supply circuit module was added in figure A.
		32	A power-supply circuit module was added in figure B.
		33-35	Chapter 6 was newly added.
5.10	Oct. 05, 2016	17	The addition of support for software tracing was stated.
		18	A note on internal tracing was added.
		19	The heading of No. 5 was changed from ‘ECC error’ to ‘Initialization of RAM areas’. In addition, an item to check and the functions which are not available when the RAM area is not initialized were stated.
		30	The note on the working RAM area in No. 34 was changed.
		30-31	Conditions for the rewriting of flash memory were added to No. 34 and No. 35.
		36	A remedy was added for cases of inability to connect with the debugging tool in section 6.1.
		37	A remedy was added for cases of inability to generate breaks in section 6.2.
6.00	Jan. 20, 2017	overall	Descriptions for the E2 Emulator were added. Changed “E1 or E20” to “E1,E20 or E2”.
		1	E2 Emulator was added.
		4	Descriptions for the E2 Emulator were added in section 1.1 and 1.3.
		5	Description for the 20-pin/14-pin converter was added in section 2.1.
		16	Description for USB interface was deleted in section 2.4.3. “to the E1 emulator” was added in section 2.4.4.
		19	Note for Hot plug-in using the E1 emulator was added.
		19	Emulator detection by user programs was added.
		32	Item No.43 was newly added.
		35-36	Internal circuit in the E2 emulator was added.
		37	E2 emulator user’s manual was added.
		42	E2 Emulator was added.
		44	E2 Emulator was added.
7.00	Jul. 01, 2017	4	Section 1.3 (3) was newly added.
		8	Descriptions were added to section 2.3.
		18	Table 3-1 was updated.
		21	Table 3-2 was newly added.
		23	Section 3.1 was newly added.
		25-39	Section 4.1 No.3, No.7 and section 4.2 No.9 were skipped. Section 4.1 No.6 was modified. The title of section 4.2 No.3 was changed. The titles were changed and the descriptions were added in section 4.2 No.14, No.15 and No.16. Section 4.2 No.44 and No.45 were newly added.
		42-43	Correct mistakes in Interface Circuits in the E2 Emulator (74LVC8T45 → 74LVC8T245)
7.10	Jan. 15, 2018	5-6	Descriptions were added in section 2.1.
		18	Table 3-1 was updated.
		24	“Top cover” was changed to “the connector for the interface can be found by removing the cover”.
		36	Descriptions were added to item No. 34 in section 4.2.
8.00	Oct. 09, 2020	20	Note 4 was added to table 3-1.
		22	Note 4 was added to table 3-2.
		34	Descriptions were added to item No. 24 in section 4.2.
		39	Section 4.2 No.46 and No.47 were newly added.

		44	Remedies for two types of error (E1203275 and E1203276) were added to table 6-1.
		45	A remedy for one type of error (E1203274) was added to table 6-2.

E1/E20 Emulator, E2 Emulator
Additional Document for User's Manual (Notes on Connection of RH850/F1H and RH850/F1M)

Publication Date:	Oct. 01, 2015	Rev.4.00
	Jul. 01, 2016	Rev.5.00
	Oct. 05, 2016	Rev.5.10
	Jan. 20, 2017	Rev.6.00
	Jul. 01, 2017	Rev.7.00
	Jan. 15, 2018	Rev.7.10
	Oct. 09, 2020	Rev.8.00

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 Corporation
TOYOSU FORESIA, 3-2-24 Toyosu, Koto-ku, Tokyo 135-0061, Japan

Renesas Electronics America Inc. Milpitas Campus
1001 Murphy Ranch Road, Milpitas, CA 95035, U.S.A.
Tel: +1-408-432-8888, Fax: +1-408-434-5351

Renesas Electronics America Inc. San Jose Campus
6024 Silver Creek Valley Road, San Jose, CA 95138, USA
Tel: +1-408-284-8200, Fax: +1-408-284-2775

Renesas Electronics Canada Limited
9251 Yonge Street, Suite 8309 Richmond Hill, Ontario Canada L4C 9T3
Tel: +1-905-237-2004

Renesas Electronics Europe GmbH
Arcadiastrasse 10, 40472 Düsseldorf, Germany
Tel: +49-211-6503-0, Fax: +49-211-6503-1327

Renesas Electronics (China) Co., Ltd.
Room 101-T01, Floor 1, Building 7, Yard No. 7, 8th Street, Shangdi, Haidian District, Beijing 100085, China
Tel: +86-10-8235-1155, Fax: +86-10-8235-7679

Renesas Electronics (Shanghai) Co., Ltd.
Unit 301, Tower A, Central Towers, 555 Langao Road, Putuo District, Shanghai 200333, China
Tel: +86-21-2226-0888, Fax: +86-21-2226-0999

Renesas Electronics Hong Kong Limited
Unit 1601-1611, 16/F., Tower 2, Grand Century Place, 193 Prince Edward Road West, Mongkok, Kowloon, Hong Kong
Tel: +852-2265-6688, Fax: +852 2886-9022

Renesas Electronics Taiwan Co., Ltd.
13F, No. 363, Fu Shing North Road, Taipei 10543, Taiwan
Tel: +886-2-8175-9600, Fax: +886 2-8175-9670

Renesas Electronics Singapore Pte. Ltd.
80 Bendemeer Road, #06-02 Singapore 339949
Tel: +65-6213-0200, Fax: +65-6213-0300

Renesas Electronics Malaysia Sdn.Bhd.
Unit No 3A-1 Level 3A Tower 8 UOA Business Park, No 1 Jalan Pengaturcara U1/51A, Seksyen U1, 40150 Shah Alam, Selangor, Malaysia
Tel: +60-3-5022-1288, Fax: +60-3-5022-1290

Renesas Electronics India Pvt. Ltd.
No.777C, 100 Feet Road, HAL 2nd Stage, Indiranagar, Bangalore 560 038, India
Tel: +91-80-67208700

Renesas Electronics Korea Co., Ltd.
17F, KAMCO Yangjae Tower, 262, Gangnam-daero, Gangnam-gu, Seoul, 06265 Korea
Tel: +82-2-558-3737, Fax: +82-2-558-5338

Colophon 7.0

E1/E20 Emulator, E2 Emulator

Additional Document for User's Manual (Notes on Connection of RH850/F1H and RH850/F1M)

Renesas Electronics Corporation

R20UT3116EJ0800