User's Manual

# Pod for IE850 In-circuit Emulator

RTE7701202EPA00000J

User's Manual

Target Devices:
RH850/E1M-S
RH850/E1L

Renesas Electronics
www.renesas.com

Rev.2.00 Apr 2015

# Important

Before using this product, be sure to read this user’s manual carefully.
Keep this user’s manual, and refer to it when you have questions about this product.

Emulator:
"Emulator" in this document collectively refers to the following products manufactured by Renesas Electronics Corp.
(1) IE850 emulator main unit
(2) Pod

"Emulator" herein encompasses neither the customer's user system nor the host machine.

Purpose of use of the emulator:
This emulator is a device to support the development of systems that use products from the RH850/E1x series of Renesas microcontrollers. It provides support for system development in both software and hardware.
Be sure to use this emulator correctly according to said purpose of use. Please avoid using this emulator other than for its intended purpose of use.

For those who use this emulator:
This emulator can only be used by those who have carefully read the user’s manual and know how to use it.
Use of this emulator requires basic knowledge of electric circuits, logical circuits, and MCUs.

When using the emulator:
(1) This product is a development-support unit for use in your program development and evaluation stages. When a program you have finished developing is to be incorporated in a mass-produced product, the judgment as to whether it can be put to practical use is entirely your own responsibility, and should be based on evaluation of the device on which it is installed and other experiments.
(2) In no event shall Renesas Electronics Corp. be liable for any consequence arising from the use of this product.
(3) Renesas Electronics Corp. strives to provide workarounds for and correct trouble with products malfunctions, with some free and some incurring charges. However, this does not necessarily mean that Renesas Electronics Corp. guarantees the provision of a workaround or correction under any circumstances.
(4) The product covered by this document has been developed on the assumption that it will be used for program development and evaluation in laboratories. Therefore, it does not fall within the scope of applicability of the Electrical Appliance and Material Safety Law and protection against electromagnetic interference when used in Japan.
(5) Renesas Electronics Corp. cannot predict all possible situations and possible cases of misuse that carry a potential for danger. Therefore, the warnings in this user's manual and the warning labels attached to the emulator do not necessarily cover all such possible situations and cases. The customer is responsible for correctly and safely using this emulator.
(6) The product covered by this document has not been through the process of checking conformance with UL or other safety standards and IEC or other industry standards in countries other than Japan. This fact must be taken into account when the product is taken from Japan to another country.
(7) Renesas Electronics Corp. will not assume responsibility for direct or indirect damage caused by an accidental failure or malfunction of this emulator.

When disposing of the emulator:
Penalties may be applicable for incorrect disposal of this waste, in accordance with your national legislation.

Usage restrictions:
The emulator has been developed as a means of supporting system development by users. Therefore, do not use it as an embedded device in other equipment. Also, do not use it to develop systems or equipment for use in the following fields.

(1) Transportation and vehicular

(2) Medical (equipment that has an involvement in human life)

(3) Aerospace

(4) Nuclear power control

(5) Undersea repeaters

If you are considering the use of the emulator for one of the above purposes, please be sure to consult your local distributor.

About product changes:
We are constantly making efforts to improve the design and performance of this emulator. Therefore, the specification or design of this emulator, or this user's manual, may be changed without prior notice.

About rights:

(1) We assume no responsibility for any damage or infringement on patent rights or any other rights arising from the use of any information, products or circuits presented in this user’s manual.

(2) The information or data in this user’s manual does not implicitly or otherwise grant a license to patent rights or any other rights belonging to Renesas or to a third party.

(3) This user’s manual and this emulator are copyrighted, with all rights reserved by Renesas. This user’s manual may not be copied, duplicated or reproduced, in whole or part, without prior written consent from Renesas.

About diagrams:
Some diagrams in this user’s manual may differ from the objects they represent.

# Precautions for Safety

This chapter, by showing the relevant diagrammatic symbols and their meanings, describes the precautions which should be taken in order to use this product safely and properly. Be sure to read and understand this chapter before using this product. Contact us if you have any questions about the precautions described here.

| | |
|---|---|
| ⚠ **WARNING** | **WARNING** indicates a potentially dangerous situation that will cause death or heavy wound unless it is avoided. |
| ⚠ **CAUTION** | **CAUTION** indicates a potentially dangerous situation that will cause a slight injury or a medium-degree injury unless it is avoided. |

In addition to the two above, the following are also used as appropriate.

△ means WARNING or CAUTION.

Example: ⚠ CAUTION AGAINST AN ELECTRIC SHOCK

🛇 means PROHIBITION.

Example: 🛇 DISASSEMBLY PROHIBITED

● means A FORCIBLE ACTION.

Example: ● UNPLUG THE POWER CABLE FROM THE RECEPTACLE.

## ⚠ WARNING

Warnings for AC Power Supply:

If the separately sold AC power cable for the power adaptor does not fit the receptacle, do not alter the AC power cable and do not plug it in forcibly. Failure to comply may cause electric shock and/or fire.

Use an AC power cable which complies with the safety standard of the country.

Do not touch the plug of the AC power cable when your hands are wet. This may cause electric shock.

This product is connected signal ground with frame ground. If your developing product is transformless (not having isolation transformer of AC power), this may cause electric shock. Also, this may give an unrepairable damage to this product and your developing one.
While developing, connect AC power of the product to commercial power through isolation transformer in order to avoid these dangers.

Connect the plug of the AC power cable to the outlet when connecting this emulator and the user system in order to eliminate differences in potential between the grounds of the emulator and of the user’s system.

If other equipment is connected to the same branch circuit, care should be taken not to overload the circuit.

When installing this equipment, insure that a reliable ground connection is maintained.

If you smell a strange odor, hear an unusual sound, or see smoke coming from this product, then disconnect power immediately by unplugging the AC power cable from the outlet.
Do not use this as it is because of the danger of electric shock and/or fire. In this case, contact your local distributor.

Before setting up this emulator and connecting it to other devices, turn off power or remove a power cable to prevent injury or product damage.

## ⚠ WARNING

Warnings to Be Taken for This Product:

Do not disassemble or modify this product. Personal injury due to electric shock may occur if this product is disassembled and modified. Disassembling and modifying the product will void your warranty.

Make sure nothing falls into the cooling fan on the top panel, especially liquids, metal objects, or anything combustible.

Note the following point on products which have a cooling fan.

When the fan does not operate due to, for example, a fault, the temperature of the emulator may be high enough to potentially cause injuries (such as burns) on contact. Accordingly, if the fan does not operate after the emulator is turned on, turn the emulator off immediately and send it to be repaired.

Warning for Installation:

Do not set this product in water or areas of high humidity. Make sure that the product does not get wet. Spilling water or some other liquid into the product may cause unrepairable damage.

Warning for Use Environment:

Care should be taken to ensure that the emulator is not used at temperatures exceeding the maximum ambient temperature.

## ⚠ CAUTION

Point for Caution Regarding the Power Adaptor:

Use only the dedicated power adaptor which is separately sold. Also, do not use the power adaptor for other equipment.

Cautions to Be Taken for Turning On the Power:

Take the steps below to turn the power to the emulator ON or OFF. Not following the order might cause damage to the user system or emulator.

When turning on the power: (1) Turn on the emulator; (2) turn on the user system; (3) connect the debugger (emulator software)

When turning off the power: (1) Disconnect the debugging session (emulator software); (2) turn off the user system; (3) turn off the emulator

Cautions to Be Taken for Handling This Product:

Use caution when handling the emulator. Be careful not to apply a mechanical shock.

Do not touch the connector pins of the emulator and the user system directly. Doing so may lead to the discharge of static electricity and so damage the internal circuits.

When attaching and removing the pod cable, hold a fixture (such as a connector) to avoid pulling the pod cable. Do not pull the emulator and board by the communications interface cable or the cable for connecting the user system since this might lead to the breaking of wires in the pod cable. Also, do not flex the pod cable excessively when installing the cable since this might lead to the breaking of wires in the cable.

Do not use inch-size screws for this equipment. The screws used in this equipment are all ISO (meter-size) type screws.

Caution to Be Taken for System Malfunctions:

If the emulator malfunctions because of interference like external noise, do the following to remedy the trouble.

(1) Exit the debugger (emulator software), and shut OFF the emulator and the user system.

(2) After a lapse of 10 seconds, turn ON the power of the emulator and the user system again, then launch the debugger (emulator software).

## ⚠ CAUTION

Note on Heat Generation While the Product is in Use:

Using this product for a long time might cause the product to have a high temperature. If this is the case, care must be taken to avoid injuries due to the heat such as low temperature burns.

Note on the Cover of the Emulation Pod:

Be sure to use the product with the cover of the emulation pod in place.

Note on Transporting the Product:

When sending your product for repair, use the packing box and cushioning material supplied with the product when it was delivered to you and specify caution in handling (handling as precision equipment). If packing of your product is not complete, it may be damaged during transportation. When you pack your product in a bag, make sure to use the conductive plastic bag supplied with the product. If you use a different bag, it may lead to further trouble with your product due to static electricity.

Caution to Be Taken for Disposal:

Penalties may be applicable for incorrect disposal of this waste, in accordance with your national legislation.

European Union Regulatory Notices:

The WEEE (Waste Electrical and Electronic Equipment) regulations put responsibilities on producers for the collection and recycling or disposal of electrical and electronic waste. Return of WEEE under these regulations is applicable in the European Union only. This equipment (including all accessories) is not intended for household use. After use the equipment cannot be disposed of as household waste, and the WEEE must be treated, recycled and disposed of in an environmentally sound manner.
Renesas Electronics Europe GmbH can take back end of life equipment, register for this service at “http://www.renesas.eu/weee”.

# How to Use This Manual

**Readers**

This manual is intended for users who wish to perform debugging using the RTE7701202EPA00000J (generic name: pod). The readers of this manual are assumed to be familiar with the device functions and usage, and to have knowledge of debuggers.

**Purpose**

This manual is intended to give users an understanding of the basic specifications and correct usage of the pod.

**Organization**

This manual is divided into the following sections.

- Overview
- Names and Functions of Hardware
- Setup Procedure
- Notes
- Optional Product
- Maintenance and Warranty

**How to Read This Manual**

It is assumed that the readers of this manual have general knowledge in the fields of electrical engineering, logic circuits, and microcontrollers. This manual describes the basic setup procedures and how to set switches.

To understand the overall functions and usages of the IE850
→ Read this manual in the order of Contents.

To know the manipulations, command functions, and other software-related settings of the IE850
→ See the user's manual of the debugger to be used.

**Conventions**

| | |
|---|---|
| Note: | Footnote for item marked with Note in the text |
| Caution: | Information requiring particular attention |
| Remark: | Supplementary information |
| Numeric representation: | Binary ... xxxx or xxxxB |
| | Decimal ... xxxx |
| | Hexadecimal ... xxxxH |

Prefix indicating power of 2 (address space, memory capacity):

K (kilo): $2^{10}$ = 1,024

M (mega): $2^{20}$ = $1,024^2$

**Terminology**

The meanings of the terms used in this manual are described in the table below.

| Term | Meaning |
|---|---|
| Target device | This is the device to be emulated. |
| Target system | This is the system to be debugged (system provided by the user). This includes the hardware and software provided by the user. |
| IE850 | Name for IE850 emulator systems in general. |
| IE850 main | QB-V850E2 |
| Pod | The pod is a peripheral of the IE850 main unit and serves as the interface with the target system. |
| Emulator | This is the product to emulate the target device. Refers to the IE850 main unit and pod in this document. |

**Related Documents**

Please refer to the related documents listed below in addition to this manual before using the product.

| Name of documents | | Document No. |
|---|---|---|
| Release note | RTE7701202EPA00000J Release Note (Restrictions on Using CS+) | R20UT2981E |
| IE850 main unit | IE850 main unit for IE850 In-circuit Emulator QB-V850E2 User's Manual | R20UT0824E |
| Exchange adapter (EA) | Exchange adapter for adapting the RTE7701202EPA00000J to 304-Pin BGA connection | R20UT2989E |
| | Exchange adapter for adapting the RTE7701202EPA00000J to 252-Pin BGA connection | R20UT3037E |
| | Exchange adapter for adapting the RTE7701202EPA00000J to 176-Pin QFP connection | R20UT2990E |
| | Exchange adapter for adapting the RTE7701202EPA00000J to 144-Pin QFP connection | R20UT3176E |

Caution: Related documents listed above are subject to change without prior notice. Thus, make sure that you have the latest documents for use in design.

# Table of Contents

# 1. Overview

The RTE7701202EPA00000J (pod) is used together with the QB-V850E2 (IE850 main unit) in order to emulate RH850/E1x series microcontrollers from Renesas Electronics Corp.

The IE850 can be used to debug hardware and software efficiently when developing systems using the target device.

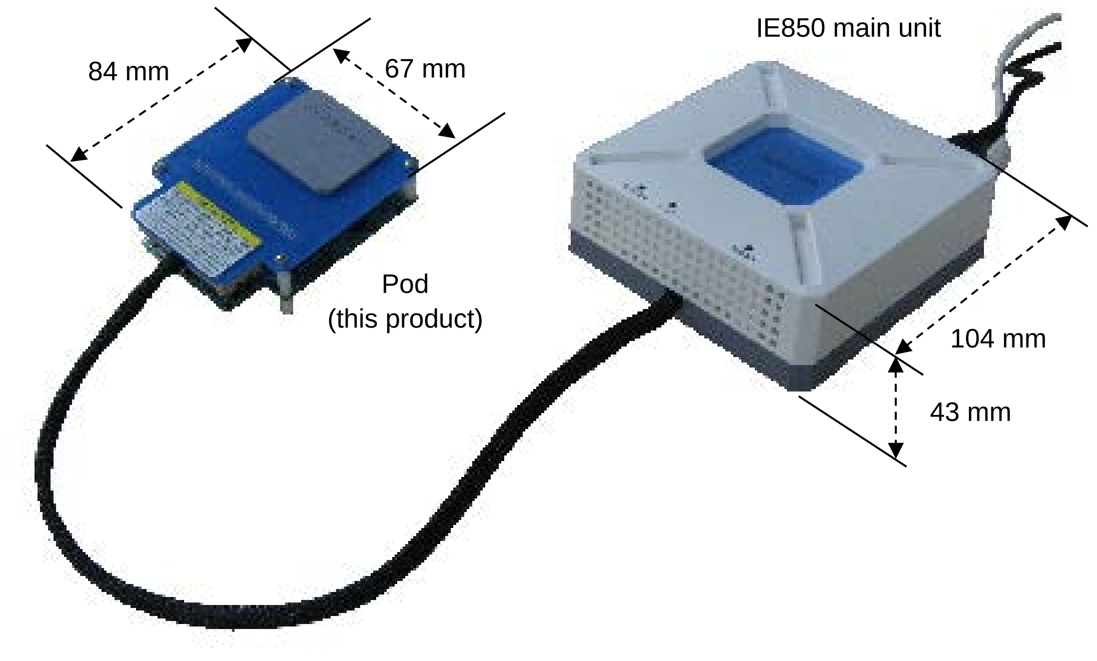


**Figure 1-1 External Appearance**

## 1.1 Hardware specification

The following table describes hardware specifications of the pod.

**Table 1-1 Pod Hardware Specifications**

| Parameter | | Specification |
|---|---|---|
| Target device | | RH850/E1M-S, RH850/E1L |
| Target system interface voltage* | SYSVCC, VCC, PLLVCC, LVDVCC | 3.3 V |
| | TTLVCC | 3.3 V or 5.0 V |
| | E1VCC, E2VCC, A0VCC, A1VCC, ADSVCC, A0VREFH, A1VREFH, ADSVREFH | 5.0 V |
| | VDD | 1.25 V |
| Maximum operating frequency | | 320 MHz |
| Input frequency of the oscillator | | 20 MHz (the oscillator on the pod is used) |
| Operating temperature range | | 0 to 40 °C (No condensation) |
| Storage temperature range | | −15 to 60 °C (No condensation) |

Note: The other terminals are connected to the internal power of the IE850.

## 1.2 System overview

The system configuration is described below. The pod cannot be used on its own. The IE850, an AC adaptor, and sockets are also required. These are sold separately.

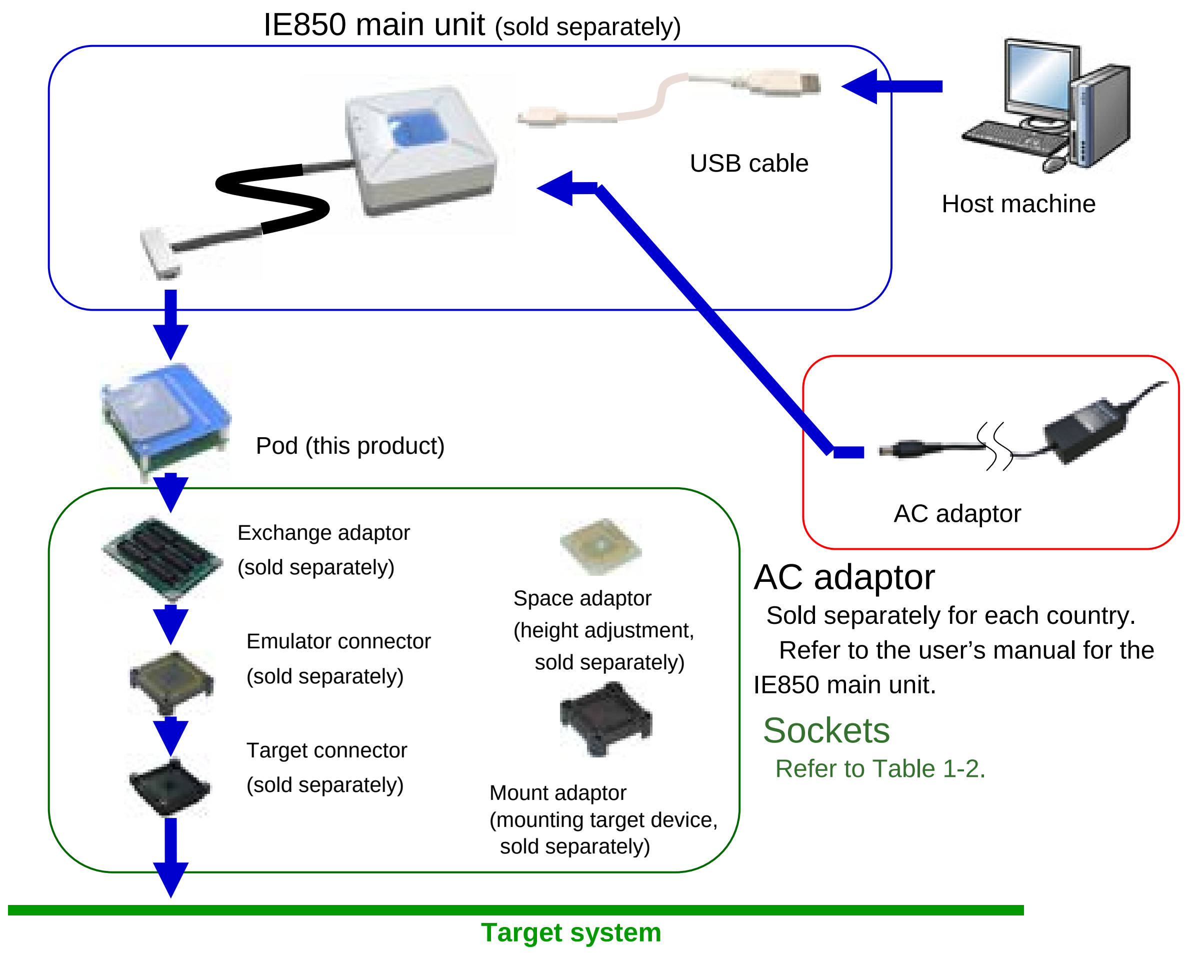


**Figure 1-2 System Configuration (When the QFP Package is Selected)**

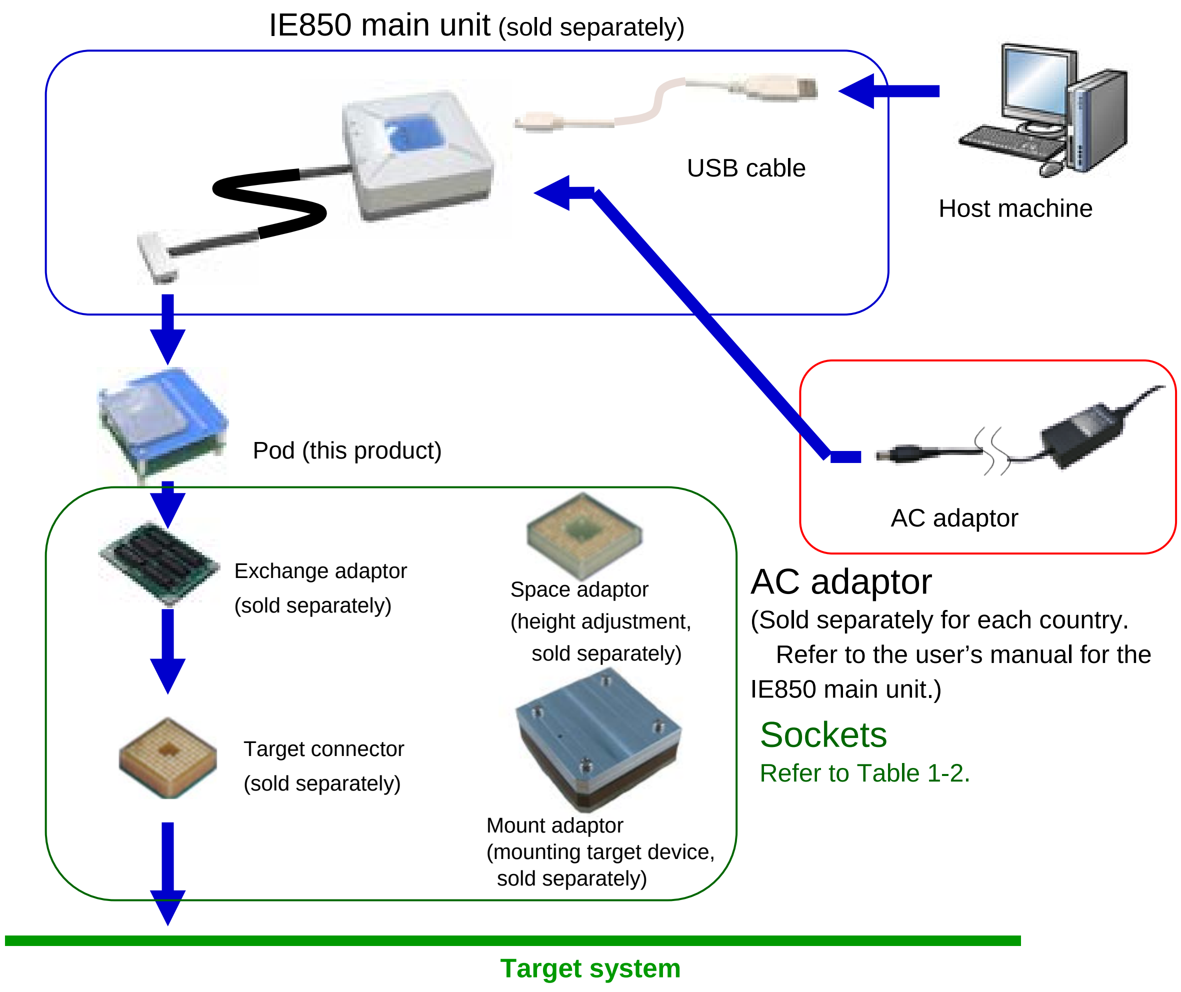


**Figure 1-3 System Configuration (When the BGA Package is Selected)**

The following table describes the corresponding sockets for the target device.

**Table 1-2 Sockets for the Target Device**

| Socket | Target Device | |
|---|---|---|
| | RH850/E1M-S | |
| | Package: BGA<br>BGA304 (PRBG0304GB-A) | Package: BGA<br>BGA252 (PRBG0252GB-A) |
| Exchange Adaptor | RTE7701202CBG304T000J (*1) | RTE7701202CBG252T000J (*1) |
| Emulator Connector | — | — |
| Target Connector | BSSOCKET304A2219RE21N (*2) | BSSOCKET252A2017RE21N (*2) |
| Space Adaptor | CSSOCKET304A2219RE03 (*2)<br>CSSOCKET304A2219RE04 (*2) | CSSOCKET252A2017RE03 (*2)<br>CSSOCKET252A2017RE04 (*2) |
| Mount Adaptor | LSPACK304A2219RE01 (*2) | LSPACK252A2017RE01 (*2) |

| Socket | Target Device | | |
|---|---|---|---|
| | RH850/E1L | | |
| | Package: QFP<br>QFP144 (PLQP0144KA-A) | Package: QFP<br>QFP176 (PLQP0176KB-A) | Package: BGA<br>BGA252 (PRBG0252GB-A) |
| Exchange Adaptor | RTE7701201CFK144T000R (*1) | RTE7701201CFK176T000J (*1) | RTE7701202CBG252T000J (*1) |
| Emulator Connector | QB-144GJ-YQ-01T (*1) | QB-176GM-YQ-01T (*1) | — |
| Target Connector | QB-144GJ-NQ-01T (*1) | QB-176GM-NQ-01T (*1) | BSSOCKET252A2017RE21N (*2) |
| Space Adaptor | QB-144GJ-HQ-01T (*1) | QB-176GM-HQ-01T (*1) | CSSOCKET252A2017RE03 (*2)<br>CSSOCKET252A2017RE04 (*2) |
| Mount Adaptor | QB-144GJ-YS-01T (*1) | QB-176GM-YS-01T (*1) | LSPACK252A2017RE01 (*2) |

Notes *1: Separately sold product (can be purchased from Renesas Electronics Corporation)
*2: Separately sold product (can be purchased from Tokyo Eletech Corporation)

## 1.3 Functional overview

IE850 is provided with a wealth of debugging functions to enable efficient program debugging, in addition to being used to emulate the operation of a target device. An overview of the functions is provided in this section.

Some functions are not supported, depending on the debugger to be used. See also the manual of the debugger to be used to confirm.

**Table 1-3 System Specifications (1/2)**

| Parameter | | Specification |
|---|---|---|
| Emulation memory capacity | On-board flash memory | Same as target devices. |
| | On-board RAM | Same as target devices. |
| | External memory | None |
| Program execution functions | Step execution | Available |
| Reset functions | Forced reset | Available |
| Event functions | Hardware breakpoints | 12 points |
| | Post-execution events | 8 points |
| | CPU access events | 8 points |
| | DMA access events | 4 points |
| | Global RAM access events | 4 points |
| | CPU sequential events | 4 steps<br>One event is created by combining the functions below (in up to 4 steps).<br>Post-execution event or CPU access event or number of passes (measured by the 32-bit counter) |
| | DMA sequential events | 3 steps<br>One event is created by combining the functions below (in up to 3 steps).<br>DMA access event or number of passes (measured by the 32-bit counter) |
| | Global RAM sequential events | 3 steps<br>One event is created by combining the functions below (in up to 3 steps).<br>Global RAM access event or number of passes (measured by the 32-bit counter) |
| Break functions | Hardware break | Available (refer to the "Hardware breakpoints" and "Post-execution events" items in the Event functions category.) |
| | Software break | Available<br>On-board flash memory: 2000 points<br>On-board RAM: Setting is not allowed |
| | Forced break | Available |
| | Other | Trace-full break, delay trigger break |

**Table 1-3 System Specifications (2/2)**

| Parameter | | | Specification |
|---|---|---|---|
| Trace functions | Real-time trace mode | | Available |
| | Non-real-time trace mode | | Available |
| | Type of trace data | | Branch-source PC, branch-destination PC, access PC, access address, accessed data, DMA access cycle, global RAM access cycle, time stamp |
| | Trace events | | Section trace, Qualified trace, Delay trigger trace |
| | Memory capacity | | 9 Mbytes (512 K frames)<br>*One frame = one set of branch information = two cycles (branch source address and branch destination address) |
| | | | 2.25 Gbytes (Approx. 128 M frames)<br>*One frame = one set of branch information = two cycles (branch source address and branch destination address)<br>(When using the long term trace option) |
| | Conditions for recording trace memory | | Non-stop mode, trace-full stop mode, trace-full break mode, delay trigger stop mode, delay trigger break mode |
| Performance measurement functions | Measurement clock | | Clock for debugging (20 MHz or 10 MHz) |
| | | Measurement objects | Beginning through end of program execution, two points between events (three channels) |
| | | Maximum measurement time | Approx. 214 or 428 seconds |
| | | Minimum resolution | 50 or 100 ns |
| | | Measurement results | Execution time |
| | Measurement clock | | CPU clock |
| | | Measurement objects | Two points between events (four channels) |
| | | Maximum measurement time | Dependent on the CPU clock |
| | | Minimum resolution | Dependent on the CPU clock |
| | | Measurement results | Number of CPU cycles |
| Other functions | | | Memory register access function, peripheral debugging function, reset mask function |

### 1.3.1 Program execution function

The program execution function enables program execution equivalent to that of the target device. The executed program can be stopped under various conditions by using the break functions (**1.3.4 Break functions (program execution stop)**). The operation of only a function can be checked by executing a program, because a program can be executed from any address.

(1) Step execution function

The step execution function can be used to execute instructions one by one, in units of assembly instructions (single stepping) or lines of C language source code (source level stepping). In the case of single stepping, only instructions that are purely in the stepwise flow can be executed, because interrupts are not acknowledged during single step execution. Take care of the following point, however, in the case of source level stepping.

**Caution** **Step execution to be performed at the source level is performed by a debugger using the break function. In this case, interrupts are acknowledged in step execution. Consequently, if processing at the interrupt destination cannot be completed, step execution may not be completed. For handling such a case, see the manual of the debugger.**

### 1.3.2 Reset function

The reset function is for resetting the CPU from the debugger. The function is used to rewrite the flash memory (after a program is downloaded followed by rewriting a part of the flash memory), start program execution from the reset vector, or initialize the CPU during debugging

### 1.3.3 Event function (Detection of specific operations)

The event function is used to detect specific fetching or other access by monitoring the bus cycle of the CPU and the external master such as DMA. Operations where the instruction at an address is executed by the CPU or access to a variable is made by the CPU or the external master can be detected. Such specific operations are referred to as events. The event function is used by the hardware break function, trace function, and performance measurement function. Events that can be registered by the event function are shown below.

(1) Hardware breakpoint event

The CPU is equipped with an event function. Events take the form of executing the instruction at an address or access to an address. Events can only be used with the hardware break function. Up to 12 points can be specified as hardware break events. Four of these are shared with the access address event. Eight points are for dedicated use with the execution of instructions at particular addresses.

[Function to be used]
— Hardware break function

[Specifiable detection condition]
— Execution of instruction at an address
— Access to an address and data

**Caution Hardware break events are in principle detected before executing the corresponding instruction unless a data condition for read access is specified as a detection condition.**

(2) Post-execution events

Post-execution events are detected when the instruction at an address is executed. These events are shared by the trace and time measurement functions. Address ranges are specifiable for post-execution events. Up to eight points can be specified as post-execution events, but two points are used for an event where the execution address is specified as a range. In other words, if all events are specified with ranges of execution addresses, only four events can be specified.

[Functions for use with these events]
— Trace function
— Time measurement function

[Specifiable detection condition]
— Execution address (a range can be specified)

**Caution Specifications of the hardware and supported functions might not be consistent depending on the debugger in use. Check the manual and other documents for the debugger you will be using.**

### (3) CPU access event

An access event is detected as access (reading or writing) to an address made by the CPU or the PCU. These events are shared by the trace and time measurement functions. Conditions for detection listed below can be specified. Up to eight points can be specified as an access event, but two points are used for an event where the access address is specified as a range. In other words, if all events are specified with ranges of addresses for access, only four events can be specified.

[Functions for use with these events]

— Trace function
— Time measurement function

[Specifiable detection conditions]

— Access to an address (a range can be specified)
— Accessed data
— Access size
— State of access (reading, writing, or both reading and writing)

**Caution Specifications of the hardware and supported functions might not be consistent depending on the debugger to be used. Thus, also check the manual and other related documents for the debugger you will be using.**

### (4) DMA access event

An access event is detected as access (reading or writing) to an address made by the DMA. These events are not available for the time measurement function. Conditions for detection listed below can be specified. Up to four points can be specified as an access event, but two points are used for an event where the access address is specified as a range. In other words, if all events are specified with ranges of addresses for access, only two events can be specified.

[Function for use with these events]

— Trace function

[Specifiable detection conditions]

— Access to an address (a range can be specified)
— Accessed data
— Access size
— State of access (reading, writing, or both reading and writing)

**Caution Specifications of the hardware and supported functions might not be consistent depending on the debugger to be used. Thus, also check the manual and other related documents for the debugger you will be using.**

### (5) Global RAM access event

An access event is detected as access (reading or writing) to a global RAM. These events are not available for the time measurement function. Conditions for detection listed below can be specified. Up to four points can be specified as an access event, but two points are used for an event where the access address is specified as a range. In other words, if all events are specified with ranges of addresses for access, only two events can be specified.

[Function for use with these events]
— Trace function

[Specifiable detection conditions]
— Access to an address (a range can be specified)
— Accessed data
— Access size
— State of access (reading, writing, or both reading and writing)
— Access source (external masters such as CPU, PCU, and DMA)

**Caution Specifications of the hardware and supported functions might not be consistent depending on the debugger to be used. Thus, also check the manual and other related documents for the debugger you will be using.**

### (6) CPU sequential event

The CPU sequential event function is used to combine post-execution events (items described in (2) above) and events registered as CPU access events (items described in (3) above) with a pass count to make a single event. Linked events are used to detect specific sequences such as cases where the instruction at an address is executed after access to a variable.

**Caution This function might not be supported depending on the debugger in use. Thus, also check the manual and other related documents for the debugger you will be using.**

### (7) DMA sequential event

The DMA sequential event function is used to combine events registered as DMA access events (items described in (4) above) with a pass count to make a single event. Linked events are used to detect specific sequences such as cases where variables A, B, and C are executed in the order of A, B, and C.

**Caution This function might not be supported depending on the debugger in use. Thus, also check the manual and other related documents for the debugger you will be using.**

### (8) Global RAM sequential event

The global RAM sequential event function is used to combine events registered as global RAM access events (items described in (5) above) with a pass count to make a single event. Linked events are used to detect specific sequences such as cases where variables A, B, and C are executed in the order of A, B, and C.

**Caution This function might not be supported depending on the debugger in use. Thus, also check the manual and other related documents for the debugger you will be using.**

### 1.3.4 Break functions (Program execution stop)

The break functions are used to stop program execution. With the IE850, program execution can be stopped under the following various conditions. See (1) to (4) for an overview of each break function.

**Table 1-4 Conditions for Breaks and their Types**

| Condition for break | Type of break |
|---|---|
| Stopping after executing the instruction at an address | Hardware break function<br>Software break function |
| Stopping after access to a variable | Hardware break function |
| Forcibly stopping | Forced break |
| Stopping if trace acquisition matches a certain condition | Trace-full break function<br>Trace delay break function |

Variable values can be checked during a break and a program can be executed again by changing register values, because the CPU operates even during a break (while the program is stopped). Interrupts generated during the break are suspended, because basically peripheral functions also operate during the break. Use the peripheral debugging function (**1.3.8 Peripheral debugging function**) to stop peripheral functions during the break.

(1) Hardware break

The hardware break function is used to observe the CPU bus cycles and set a break for a specific fetch or access operation. For example, a break is set by detecting a state where an address has been executed or a variable has been accessed. For states that can be set and cautionary notes, see **1.3.3 Event function (specific operation detection)**.

(2) Software break

The software break function is used to set a break when a specific address has been executed (fetched). A break is set by temporarily replacing the instruction code of the specified address with the instruction for a break.

**Caution Using the software break function leads to rewriting of the on-board flash memory whenever a break is set, deleted, or executed. This reduces the number of cycles of rewriting relative to cases where the software break function is not used.**

(3) Forced break

This function is used to forcibly stop a program when a user wants to stop a program.

(4) Other types of breaks

- Trace-full break function

  This function is used to stop a program when the trace memory is full.
- Delay trigger break

  This function is used to stop a program after acquiring a certain amount of trace data by hitting a certain event condition.

**Caution This function might not be supported depending on the debugger in use. Thus, also check the manual and other related documents for the debugger you will be using.**

### 1.3.5 Trace function (Program execution history)

The trace function is used to check the execution history (trace information) of external masters such as the CPU and DMAC. Various types of functions shown from (1) to (4) can be used in the IE850.

(1) Priority of trace data acquisition

This feature enables selecting the priority for the acquisition of trace data.

**Table 1-5 Priority of Trace Acquisition**

| Priority of Trace Acquisition | Descriptions |
|---|---|
| Real-time trace mode | Mode that gives priority to the program execution. If trace output is not fast enough, output of the trace information is temporarily halted. Thus, although the program runs in real time, trace information might be lost. This depends on the program being executed. |
| Non-real-time trace mode | Mode that gives priority to trace acquisition. If trace output is not fast enough, operation of the CPU is temporarily halted. Thus, the loss of trace information is suppressed* but the program loses the characteristic of real-time operations.<br>**Caution *: Using this mode will always decrease the degree of information loss compared with the real-time trace mode. However, losing information might still not be avoidable. Whether this is so depends on the program being run.** |

(2) Types of trace data

Types of trace data (trace information that can be acquired) are listed below.

**Table 1-6 List of Types of Trace Data**

| Type of trace data | Descriptions |
|---|---|
| Branch source PC value<br>Branch destination PC value<br>(PC stands for program counter) | Branch source and branch destination PC values can be recorded in the history. The debugger complements the program with the instructions that would otherwise be executed between branch points and displays the complemented data. Thus, a program that was practically executed could be confirmed.<br>**Caution: Execution history that can be displayed depends on the number of branches acquired.** |
| Access PC<br>Access address<br>Accessed data | History of the PC values for executed access instructions for the memory or peripheral I/O register, access addresses, and accessed data from the CPU can be recorded. A history of reading and writing can also be recorded.<br>**Caution: Access to the program registers of the CPU (such as r1 and r2) and system registers (PSW and EIPC) cannot be recorded in the history.** |
| DMA access cycle | History of access addresses and accessed data for the memory or peripheral I/O register from the DMA can be recorded. A history of reading and writing can also be recorded. |
| Global RAM access cycle | History of accesses to the global RAM from the CPU and the external master such as DMA, access addresses, accessed data, and access sources can be recorded. A history of reading and writing can also be recorded. |
| Time stamp | Elapsed time after starting a trace can be added to each item of trace information. The clock used in measurement for the time stamps is based on the CPU clock. |

(3) Types of trace event

Types of trace event that can be specified as conditions for acquisition are listed below. Trace information can be acquired in response to satisfying specified conditions for acquisition in combination with the function described in section **1.3.5**, **Trace function (Program execution history).**

**Table 1-7 List of Types of Trace Event**

| Type of trace event | Descriptions |
|---|---|
| Section trace (Acquire the history of a specific section) | History can only be recorded for the specified section. For example, the execution history of a function only from its beginning to its end can be recorded. |
| Qualified trace (Acquire the history of a specific event occurring) | History can be recorded only when a specific event occurs. For example, access only to a certain variable can be recorded. |
| Delay trigger trace (Acquire the history prior to and subsequent to the occurrence of a specific event) | History after a specific event occurs can be recorded. This resembles the measurement of a signal waveform assuming the edge as a trigger when a signal is measured with the oscilloscope. For example, the history of program execution prior to and subsequent to write access to a variable can be confirmed. Also, the amount of trace information acquired after a generation of the specific trigger can be selected from three levels (i.e., small medium, or large).<br>**Caution: This function might not be supported depending on the debugger in use. Thus, also check the manual and other related documents for the debugger you will be using.** |

(4) Conditions for recording in trace memory

This feature allows you to select how the trace memory is utilized.

**Table 1-8 List of Conditions for Recording in Trace Memory**

| Conditions for recording in trace memory | Descriptions |
|---|---|
| Non-stop mode | New information is written over old information so that the latest information is consistently acquired. Trace acquisition continues until a break is reached. |
| Trace-full stop mode | When trace memory becomes full, further trace information is not acquired. The program continues to run. |
| Trace-full break mode | When trace memory becomes full, further trace information is not acquired and the program is stopped. |
| Delay trigger stop mode | The latest information is consistently acquired by writing new information over old information until a specific condition occurs. After the specific condition occurs, the selected amount of trace information is acquired, after which further trace information is not acquired. The program continues to run. **Caution: This function might not be supported depending on the debugger in use. Thus, also check the manual and other related documents for the debugger you will be using.** |
| Delay trigger break mode | The latest information is consistently acquired by writing new information over old information until a specific condition occurs. After the specific condition occurs, the selected amount of trace information is acquired, after which further trace information is not acquired. The program is also stopped. **Caution: This function might not be supported depending on the debugger in use. Thus, also check the manual and other related documents for the debugger you will be using.** |

### 1.3.6 Performance measurement

Performance measurement is for measuring execution time from the time a program is started until it is stopped or the execution cycle of a specific segment. The time taken from the start of the program until it is stopped is consistently measured.

In addition, the maximum, minimum, latest, or accumulated time and the number of passes can be selected and measured when measurement between two events is executed. For example, it is possible to measure the execution time only for a specific function.

**Caution Specifications of the hardware and supported functions might not be consistent depending on the debugger to be used. Thus, also check the manual and other related documents for the debugger you will be using.**

### 1.3.7 Memory register access function

Memory register access allows the forms of access to memories and registers listed below.

(1) Register access

Access to general-purpose and system registers of the CPU during breaks is possible.

(2) Real-time RAM monitor

Display of on-board RAM areas during program execution is possible.

(3) Change of direct memory

On-board RAM can be changed to a desired value during program execution.

### 1.3.8 Peripheral debugging function

This function is used for debugging peripheral functions. The features shown below can be utilized.

(1) Peripheral break function

When the break function has been used to stop program execution, peripheral functions other than the watchdog timer continue to operate in general, but some peripheral functions can be stopped by using the peripheral break function. Refer to the user’s manual of the target device for information on the peripheral functions.

(2) Access protection reset sequence debugging function

Some peripheral IO registers require an access protection reset sequence to prevent illegal writing due to runaway execution of a program or for other reasons. Access to the corresponding group of registers is possible without affecting the reset sequence even when a break is set while the access protection reset sequence is being executed for a register where this is required. The subsequent part of the reset sequence can be continued without causing an error even when execution of the program is restarted.

### 1.3.9 Reset masking

Two levels of reset masking are selectable on the IE850.

- Masking only the pin reset
  The pin reset is masked by fixing the pin reset for the debug chip (microcontroller) on the pod to high level.
- Masking the system reset as a whole (pin reset and internal reset are masked)
  All resets are masked with the setting in the debug chip (microcontroller).

## 1.4 Regulatory notices

### ●European Union regulatory notices

This product complies with the following EU Directives. (These directives are only valid in the European Union.)

#### CE Certifications:

This product complies with the following European EMC standards.

・EMC Directive (2004/108/EC)

EN 55022 Class A

---

**WARNING:** This is a Class A product. This equipment can cause radio frequency noise when used in the residential area. In such cases, the user/operator of the equipment may be required to take appropriate countermeasures under his responsibility.

---

EN 55024

#### Information for traceability:

・Authorised representative

| | |
|---|---|
| Name: | Renesas Electronics Corporation |
| Address: | 1753, Shimonumabe, Nakahara-ku, Kawasaki, Kanagawa, 211-8668, Japan |

・Manufacturer

| | |
|---|---|
| Name: | Renesas System Design Co., Ltd. |
| Address: | 5-20-1, Josuihon-cho, Kodaira-shi, Tokyo, 187-8588, Japan |

・Person responsible for placing on the market

| | |
|---|---|
| Name: | Renesas Electronics Europe GmbH |
| Address: | Arcadiastrasse 10, 40472 Dusseldorf, Germany |

・Trademark and Type name

| | |
|---|---|
| Trademark: | Renesas |
| Product name: | IE850 Emulator |
| Type name: | RTE7701202EPA00000J |

#### Environmental Compliance and Certifications:

・Waste Electrical and Electronic Equipment (WEEE) Directive 2012/19/EC

### ●United States Regulatory notices on Electromagnetic compatibility

This product complies with the following EMC regulation. (This is only valid in the United States.)

#### FCC Certifications:

This equipment has been tested and found to comply with the limits for a Class A digital device, pursuant to Part 15 of the FCC Rules. These limits are designed to provide reasonable protection against harmful interference when the equipment is operated in a commercial environment. This equipment generates, uses, and can radiate radio frequency energy and, if not installed and used in accordance with the instruction manual, may cause harmful interference to radio communications. Operation of this equipment in a residential area is likely to cause harmful interference in which case the user will be required to correct the interference at his own expense.

This device complies with Part 15 of the FCC Rules. Operation is subject to the following two conditions: (1) this device may not cause harmful interference, and (2) this device must accept any interference received, including interference that may cause undesired operation.

---

**CAUTION:** Changes or modifications not expressly approved by the party responsible for compliance could void the user's authority to operate the equipment.

---

## 1.5 Block overview

An internal block overview of the functions is described below.

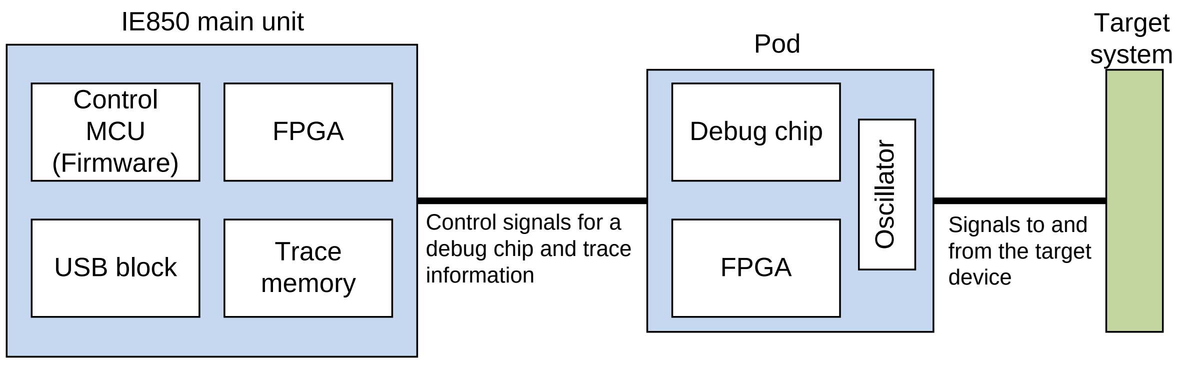


**Figure 1-4 Overview of Internal Blocks**

## 1.6 Package contents

RTE7701202EPA00000J package includes the items below. Confirm the items in the attached contents of the package.

Products supplied with RTE7701202EPA00000J:

— Pod
— Table of Toxic and Hazardous Substance and Elements

# 2. Names and Functions of Hardware

The following shows the names of pod and IE850 hardware units and their features.

## 2.1 Pod

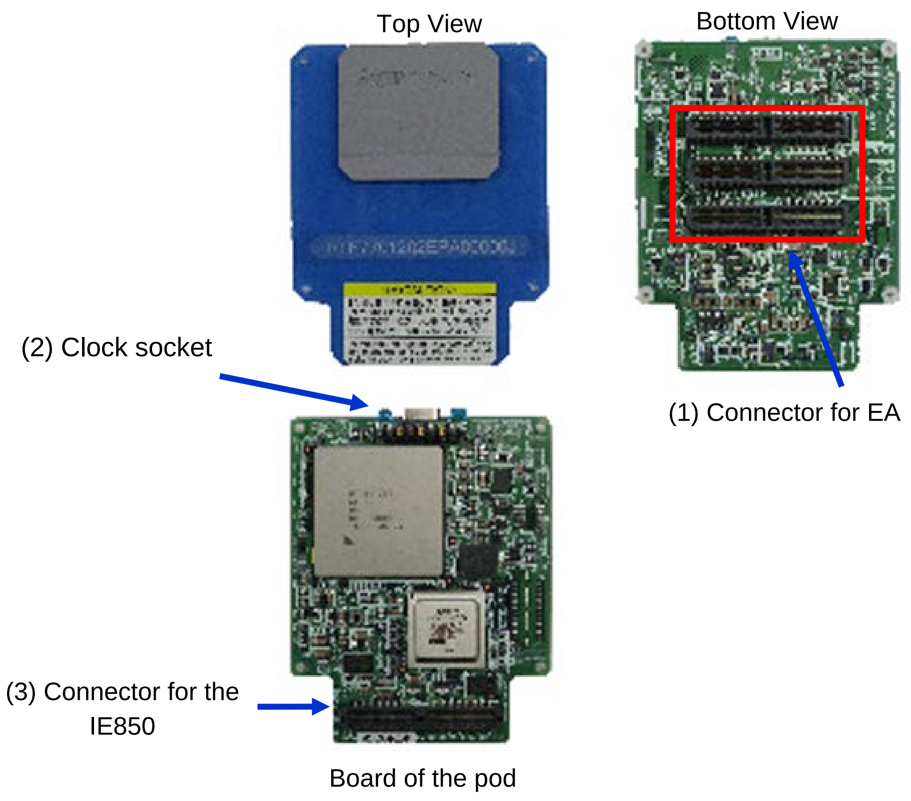


**Figure 2-1 Names of Parts of Pod**

(1) Connector for EA
This is the connector for connecting to the exchange adaptor (EA).

(2) Clock Socket
This is the socket for the main oscillator. A 20 MHz oscillator is mounted upon shipment.

(3) Connectors for the IE850
This is the connector for connecting the pod cable of the IE850 main unit.

## 2.2 IE850 main unit

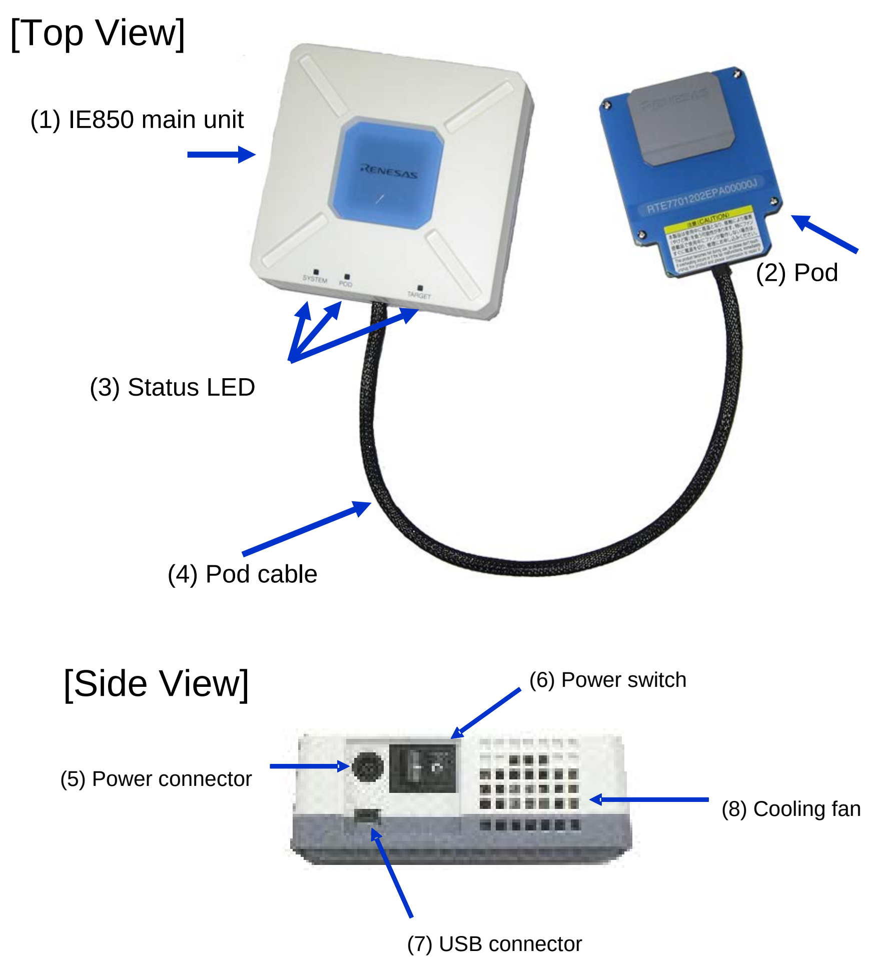


**Figure 2-2 Names of Pars of IE850**

(1) IE850 main unit

The IE850 main unit is the main controller of debugging. The IE850 main unit is sold separately. The control program (firmware) and FPGA data will need to be rewritten in accord with the pod to be connected. For more information on rewriting, refer to http://www.renesas.com/ie850.

(2) Pod

Please refer to section 2.1.

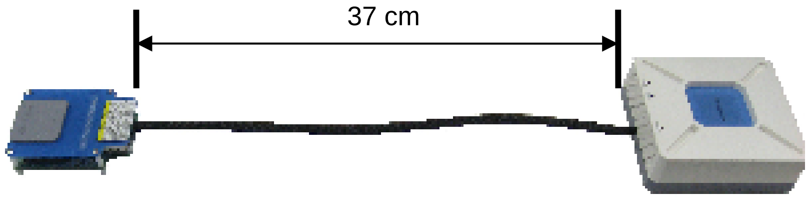
(3) Status LED

The status LEDs turn on or blink according to specific causes as described in the table below. If any LED does not turn on or not blink, IE850 main unit might be broken. In this case, contact a Renesas Electronics sales representative or distributor.

| LED name | Description |
|---|---|
| SYSTEM | This LED turns on when the power switch is turned on.<br>This LED blinks if the FPGA in the IE850 main unit is not running correctly. In this case, the IE850 main unit might be broken. |
| POD | This LED turns on when communication with the emulation pod is established. |
| TARGET | This LED turns on when the target system is turned on. |

(4) Pod cable

This coaxial cable is used to connect the IE850 main unit and emulation pod. The cable length is shown below. Be careful not to excessively bend this cable because doing so might break wires in the cable.



(5) Power connector

This connector is for the power supply cable.

(6) Power switch

This switch turns the power on and off. Press the “|” side to turn on the power or the “O” side to turn off the power.

(7) USB Connector

This connector is for a USB cable.

(8) Cooling fun

This fan cools down the IE850 main unit internal units. Be careful not to obstruct the vents.

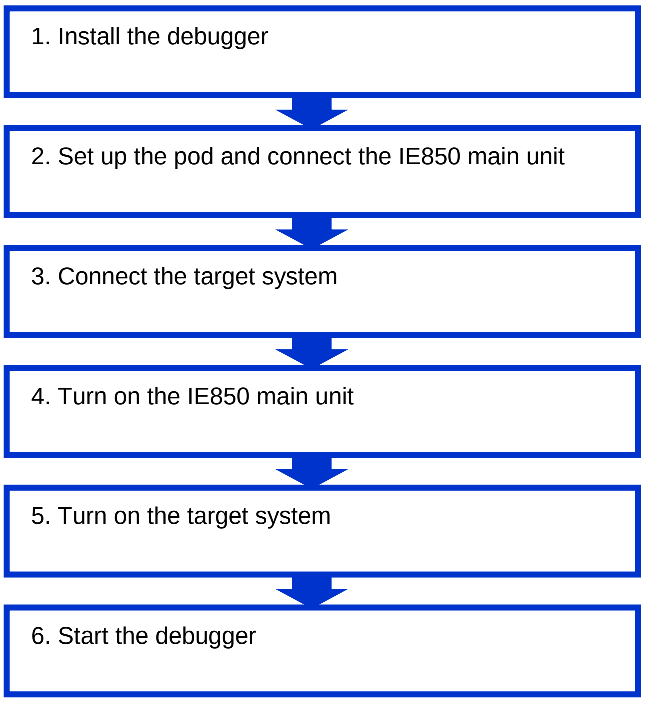
# 3. Setup Procedure

This chapter explains the IE850 main unit and the procedures for setting up the pod and for connecting the main unit via the pod to the target system. Since connecting only the IE850 main unit and pod allows you to start a debugger, you are then able to start developing a user program. Proceed with setting up to suit your field of application.

Setup can be completed by performing installation/setup in the order in which it appears in this chapter.

Perform setup along the lines of the following procedure.

To shut down the system, refer to **3.7 Shut Down Procedure**


1. Install the debugger
2. Set up the pod and connect the IE850 main unit
3. Connect the target system
4. Turn on the IE850 main unit
5. Turn on the target system
6. Start the debugger

## 3.1 Install the debugger

Install the necessary debugger before setting up the hardware.

See the user's manual of the debugger for installation instructions.

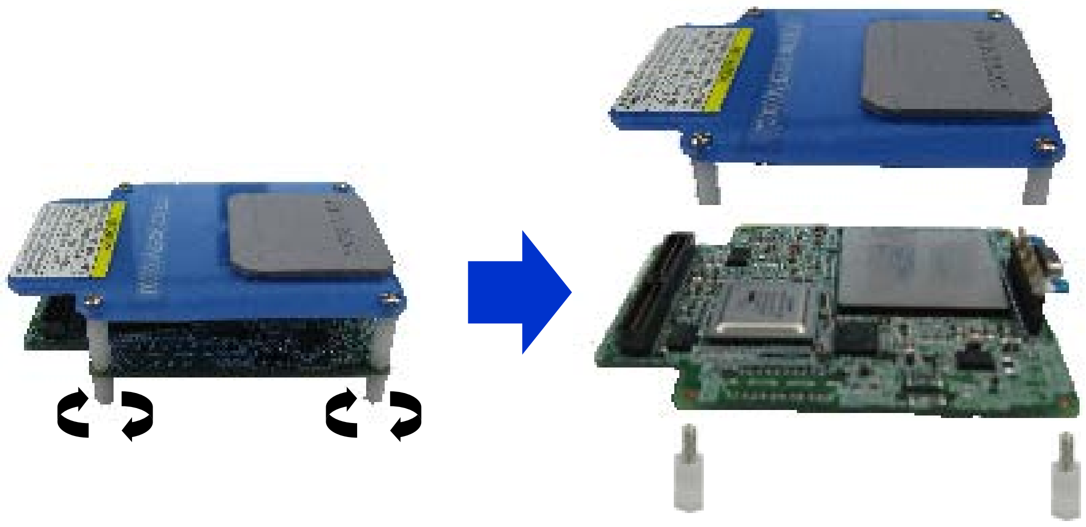
"Debugger" here refers to an integrated development environment such as CS+ or a compatible product from Green Hills Software, a US company.

## 3.2 Set up the pod and connect the IE850 main unit

Set up the clock on the pod and connect the IE850 main unit.

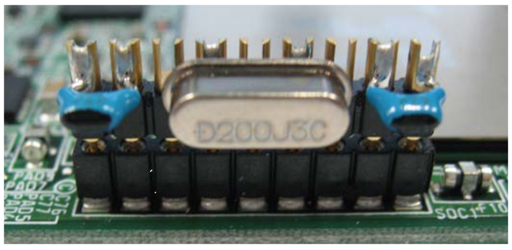
### 3.2.1 Removing pod cover

Remove the pod cover as shown below.



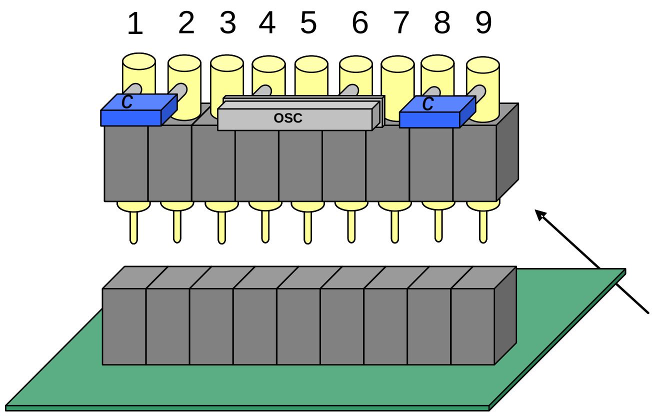
### 3.2.2 Clock Settings

The main-oscillator clock is generated by the oscillator on the pod, as shown in the figure below.



This product has a 20-MHz resonator. To change the resonator, connect a replacement to the IE850 main unit with reference to the positions of the resonator and capacitors as shown below.

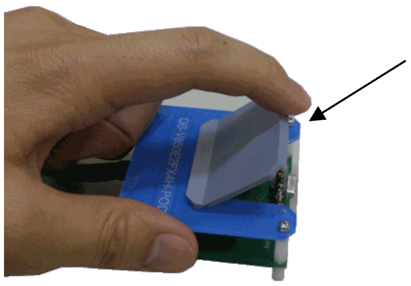
**Caution: This product does not support clock input from an oscillator on the target system.**



1-2: Capacitor
3: Power for the resonator (Unavailable when the resonator is not in use)
4-6: Resonator
8-9: Capacitor
Pins 5 and 7 are open circuit

To remove the module, use a tool such as a precision screwdriver.

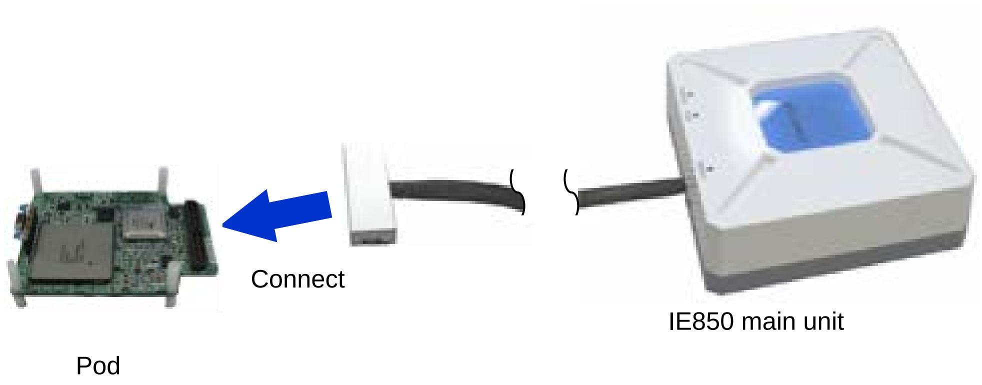
**Additional information: You can open the lid on the top of the pod, as shown in the figure below to set the clock.**



Clock settings can be made after opening the pod cover.

### 3.2.3 Connecting the IE850 main unit to the pod

Connect the IE850 main unit to the pod as shown in the figure below.



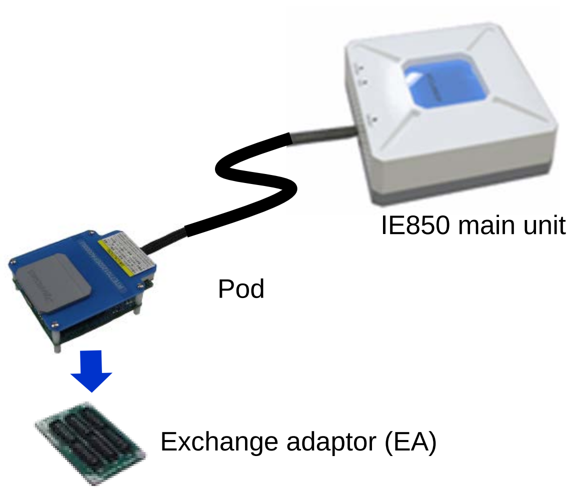
Finally, close the cover of the pod as the last step in connecting it to IE850 main unit.

## 3.3 Connect the target system

This section describes the overall process of connection to the point where the target system is connected.

### 3.3.1 Connection of the pod and target system

Connect sockets such as the exchange adaptor (EA), emulator connector (EC), and target connector (TC) to the pod for connection to the target system.
Refer to the user's manual for the exchange adaptor (EA) supporting the target device for more information on the connection of the emulator connector (EC) and target connector (TC).
Once the pod and target system are connected, make sure that the pod cable is not excessively bent. An image of connecting the IE850 main unit and pod to the exchange adaptor (EA) is shown below for reference.



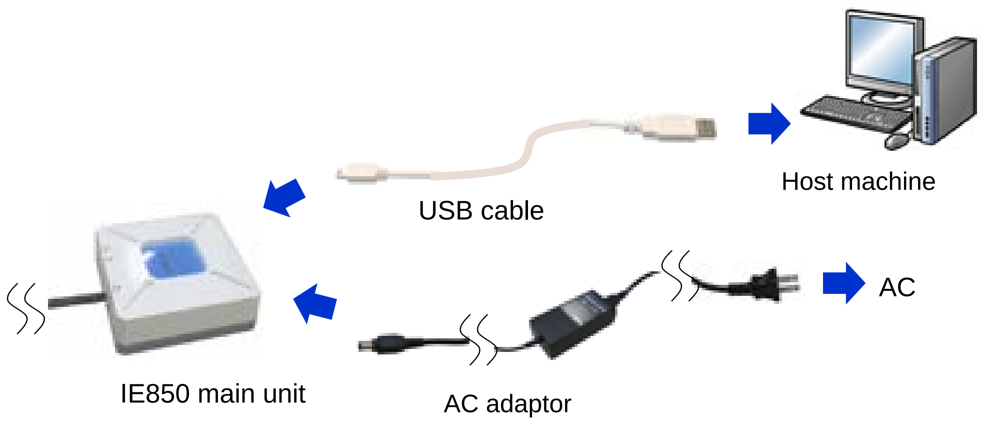
### 3.3.2 Handling precautions for sockets

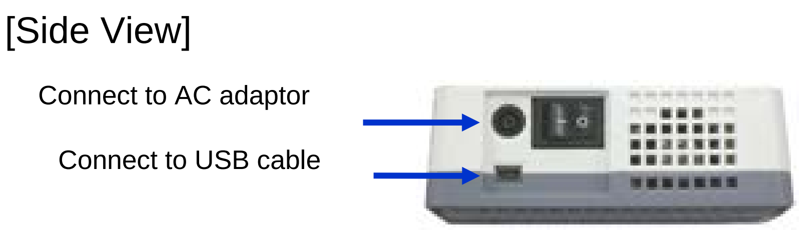
Although a set of descriptions for connecting the socket has been completed, take the precautions below when handling the sockets.

- When removing a socket from the case, pull out the cushion first, holding down the main unit.
- Handle the pins of the emulator connector (EC) and space adaptor (SA) with care, as they are thin and easily bent. Make sure that the pins are not bent before connecting these parts to the target connector (TC).
- When screwing down an EC to a TC soldered to the board of the user system, screw in each of the four screws partway, and then tighten each of the screws in turn, using a #0 or #1 Phillips-head (cross-head) precision screwdriver or torque screwdriver. Lock the torque at 0.054 Nm (maximum). If only one screw is too tight, then it could cause a bad contact. The board that the EC is connected to must have component holes at the prescribed locations (four locations, with diameter of 2.3 or 3.3 mm). Wiring is not permitted in the area of the screw heads (3.8/4.3 mm diameter.).
- When pulling out or pressing in the EC, do not twist or wiggle the EC, as this could cause the EC's pins to bend or fall out. Instead, use a flat-head screwdriver to pry off the EC from each of the four sides, a little at a time. Before connecting the EC and SA, first screw together the TC and EC with the YQ guides (supplied with the EC), using a 2.3-mm-flat-head screwdriver. Lock the torque at 0.054 Nm (maximum). If only one screw is too tight, then it could cause a bad contact.
- Do not clean with solvents, as residue of the cleaning solution could remain in the contact.
- An actual device cannot be mounted in combination with the TC and EC. Use the mount adaptor (MA) to do this.
- Do not use sockets in environments subject to shocks or vibration.

### 3.3.3 Connecting the USB cable and AC adaptor

Connect the USB cable and power supply adaptor as shown below. At this time, make sure that the IE850 main unit is not on.

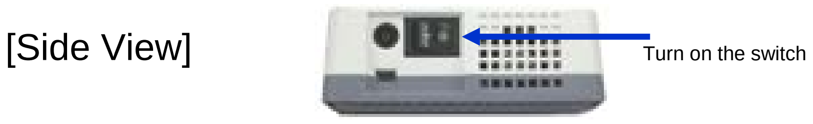


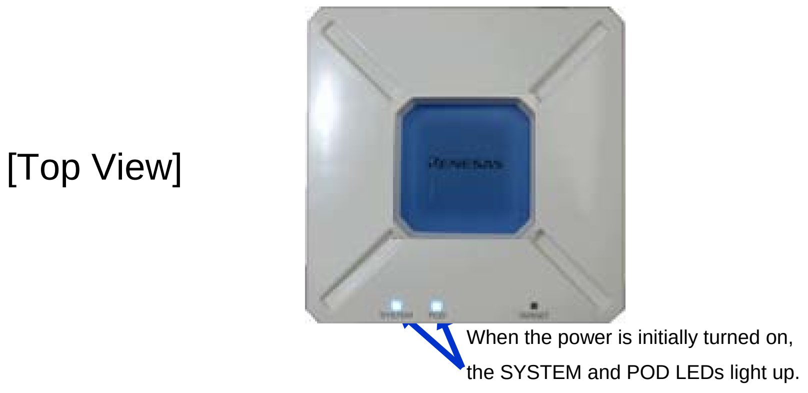


## 3.4 Turn on the IE850 main unit

Turn on the IE850 main unit. At this time, make sure that the target system is not on.

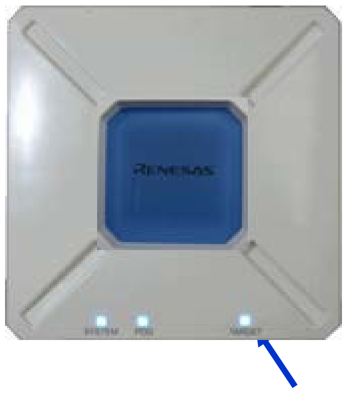
After the power is initially turned on, the SYSTEM and POD LEDs light up. If these LEDs blink or remain off, the IE850 main unit might be broken. In this case, contact a Renesas Electronics sales representative or distributor.




[Top View]
When the power is initially turned on,
the SYSTEM and POD LEDs light up.

## 3.5 Turn on the target system

After the power is on, the TARGET LED lights up. If the LED remains off, connectors might be connected poorly, or might be broken. Confirm if the connectors are poorly connected. If this is not the case, contact a Renesas Electronics sales representative or distributor.



## 3.6 Start the debugger

After the above procedure, start the debugger.

For details about debugging procedures, see the user's manual of the debugger.

## 3.7 Shut down procedure

Shut down the development system according to the procedure below.

1. Exit the debugger
2. Turn off the target system
3. Turn off the IE850 main unit
4. Shut down the development system

# 4. Notes

This chapter explains the common notes of the IE850.

## 4.1 Notes for differences between actual device and emulator

When connecting the emulator and target system for debugging, the emulation duplicates the behavior of an actual device on the target system to the greatest extent possible, but the following differences do exist between the behavior of the emulation and actual device. We therefore urge you to perform an evaluation on the actual device as the final evaluation step before mass production. It is your responsibility to ensure that the target system is suitable.

### 4.1.1 Behavior after target system is powered on

After turning on power to the device mounted on the target system, the program runs following release from the reset state. The emulator, however, will not start the program until the debugger has downloaded the program and performed a start-execution operation.

Also, although the emulator is able to download and run object code before the initial variable values and other information have been ROMized, the actual device will not function normally if the object code is not ROMized.

### 4.1.2 DBTRAP instruction

The DBTRAP instruction is not available in user programs because it is used for software breakpoints.

### 4.1.3 Function of On-chip debugging emulator

Debugging cannot proceed while an on-chip debugging emulator is connected to the user system.

### 4.1.4 AUDR function

Emulation of the advanced user debugger RAM monitor (AUDR) function which helps debugging of programs running on actual system is not possible.

### 4.1.5 Serial programming

Serial programming cannot be used.

### 4.1.6 Downloaded programs

Programs are downloaded to the flash memory of a debugging chip mounted on a pod. In order to run the program successfully, however, you should always download the program before starting debugging.

### 4.1.7 Halt mode

Release from the halt mode follows a break.

### 4.1.8 Power shutoff standby

Emulation of the power shutoff standby is not supported. Confirm the behavior of the power shutoff standby by mounting the target device while the emulator is not connected.

### 4.1.9 Oscillator

The emulator does not support clock input from an oscillator on the target system. The clock for the emulator is from an oscillator on the pod.
Proceed with final evaluation to confirm operation of the target system after mounting the actual device and while using the oscillator on the target system.

### 4.1.10 Current drawn

The emulator might in fact draw less current than the actual device since a part of power is provided by an emulator. Thus, the customer should use the actual device in a final evaluation before mass production and judge the suitability of adopting the product.

### 4.1.11 ECC errors

When the emulator is turned on, the local, global, and FCU-RAM areas are initialized to H’0000 0000. This leads to the following difference from the actual device.

- The initial values in RAM immediately after the startup may be different from the initial value (undefined) of the actual device.
- Errors caused by the RAM values not being initialized are not detected.

To emulate ECC errors, do not make any setting to initialize the RAM area when the emulator is started.

### 4.1.12 VDD and EPTVOUT pins

VDD (core power source) is provided from the emulator on the pod. VDD cannot be supplied from the target board. Also, EPT control for VDD from EPTVOUT (EPT control pin) is not possible.

### 4.1.13 RAMVCL and ADSVCL pins

A stabilizing capacitor is mounted between the stabilizing capacitor connection pins (RAMVCL and ADSVCL) on the pod. These pins are not connected on the target board.

### 4.1.14 VSS pins for each power subsystem

The following VSS pins are connected to the common GND in the emulator.

VSS, A0VSS, A1VSS, ADSVSS, PLLVSS, LVDVSS, ADSVREFL

### 4.1.15 OTP flag

The one-time programming (OTP) flag cannot be emulated. Even if the OTP flag is set by self-programming, it is ignored.

### 4.1.16 A/D converter

Results from the A/D converter differ from those on the actual device because of the exchange adapter etc. between the debug chip and the user system.

## 4.2 Cautionary Notes on Debugging

### 4.2.1 Power to the target system during debugging

Do not turn off the power to the target system during debugging. Turning off the power requires reconnection of the debugger.

### 4.2.2 Access break function

In the access break function, breaks follow reading of specified data and writing of specified data by read-modify-write commands. For other types of access, the break precedes execution of the function.

### 4.2.3 Initialization of the RAM area

Be sure to initialize all RAM areas to be used by a program. If any setting is made to initialize the RAM area when the emulator is started, ECC errors will not be generated while the emulator is in use since the debugger initializes the RAM area. However, operating the actual device with a program without having initialized the RAM area will lead to ECC errors so that the program does not operate normally.

Also, when initializing the RAM, be sure to apply ROMization since the data downloaded to RAM before the program is executed are also initialized.

Refer to the user’s manual of the compiler you will be using for more information on ROMization.

### 4.2.4 Reset of pins

Do not allow the generation of a reset in the form of a pin reset other than while the user program is in execution. Otherwise, the debugger might hang. Caution is also needed in the case where the reset mask setting in the debugger is for masking. For details, refer to section 4.2.11, Operations in response to resets and interrupts when an emulator is in use.

### 4.2.5 Trace function

The following restrictions apply to the trace function.

- The pushing of write data by the PUSHSP instruction cannot 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 might be lost when access to data by the CPU is continuous and frequent.

### 4.2.6 DEB bit of the MGDGRPROTn register

While you are using an emulator, do not change the DEB bit of the MGDGRPROTn register and PROTDEB bit of the FSGDxxDPROTn register from the setting "1" (allowing access by debug masters). Settings different from these initial values may prevent access to memory proceeding normally.

### 4.2.7 Software resets and debugging

Resets are always masked during single-step execution and breaks. Whether resets are masked during C-source-level stepped execution depends on the facilities of the debugger. Software resets will not be generated during single step execution in response to processing for setting a software reset, or when the debugger writes to the setting register for a software reset during a break.

### 4.2.8 Interrupts when stepped execution is in use

EIINT, FEINT, and FPI are held pending if they occur during single step execution. Other interrupts are always accepted. Acceptance of interrupts during C-source-level step execution depends on the facilities of the debugger.

### 4.2.9 Stepped execution of the HALT instruction

When a HALT instruction is encountered during single step 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.

### 4.2.10 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 DTC-RAM resources

  Normal access will not proceed unless a master (i.e. the CPU or PCU) is allocated to use the channel. When access is attempted while a master has not been allocated, an error will be detected on the ECM side.
- Access to FCU-RAM resources

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

  Attempted access to the PBG guard area will not proceed while the guard is enabled. Also, this is within the scope of error detection.

### 4.2.11 Operations in response to resets and interrupts when an emulator is in use

When an emulator is in use, operation in response to resets and interrupts differs according to the specifications of the reset mask and interrupt settings, respectively, as shown in the tables below.

**Table 4-1 Relation between the State of Emulation and Presence of a Reset Mask**

| Reset mask specification with an emulator | State of emulation and presence of a reset mask | | | |
|---|---|---|---|---|
| | In user program execution | In single stepping | In C-source-level stepping | In breaks |
| **Mask specified** | Masked | Masked | Masked | Masked |
| **Mask not specified** | Not masked | Masked | Depends on the debugger | Masked |

Note*: Do not allow the generation of a reset in the form of a pin reset other than while the program is in execution regardless of presence of a mask above. The debugger may hang.
However, when masking of the pin reset only is selected for the reset mask setting in an emulator, the debugger will not hang since the pin reset input for the microcontroller on the pod is masked even if the pin reset is generated on the target system.

**Table 4-2 Relation between the State of Emulation and Acceptance of Interrupts**

| Settings for interrupts | State of emulation and acceptance of interrupts | | | |
|---|---|---|---|---|
| | In user program execution | In single stepping | In C-source-level stepping | In breaks |
| **DI** | Not accepted | Not accepted | Not accepted | Not accepted |
| **EI** | Accepted | Some held pending* | Depends on the debugger | Held pending |

Note: Exceptions that are held pending are EIINT, FEINT, and FPI, and the other exceptions are all accepted.

### 4.2.12 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.

OPJTAG[1-0] (OPBT2[30:29]) bits
The value of the OPJTAG[1-0] bits is always 01B.

STMSEL1 (OPBT0[1]) bits
While an emulator is connected, the value of STMSEL1 is always read as 1. However, the MCU operates as if STMSEL1=0.

### 4.2.13 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 of at least 5 ms* before executing the program following release from the reset state.
Note: Time required for preparing communications depends on the host PC environment of the IE850 emulator and the operating frequency of the MCU.

### 4.2.14 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 started, raising the possibility of incorrect communications.
Accordingly, insert a waiting time of at least 5 ms* before applying an internal reset after a release from the initial reset state when debugging a program which includes an internal reset immediately after a release from the initial reset state.
Note: Time required for preparing communications depends on the host PC environment of the IE850 emulator and the operating frequency of the MCU.

### 4.2.15 Cautionary note on asynchronous debugging mode (peripheral break function)

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

### 4.2.16 Cautionary note on asynchronous debugging mode (reset)

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

### 4.2.17 Cautionary note on asynchronous debugging mode (watchdog timer)

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

### 4.2.18 Cautionary note on asynchronous debugging mode (ECC error)

During execution of a user program, there may be a case that the ECC error function does not normally operate for flash memory resources.
Example: When any CPU accesses flash memory resources 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.

### 4.2.19 Cautionary note on asynchronous debugging mode (specific sequence)

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 window at the same timing, the specific sequence from any CPU is not satisfied and normal accessing is disabled.

# 5. Optional Product

## 5.1 Long term trace option

This chapter explains an optional product QB-V850E2-SP for extending the trace memory.

### 5.1.1 General

The QB-V850E2-SP is an option product that can expand trace memory for the IE850. Please confirm the supported versions of the debugger.

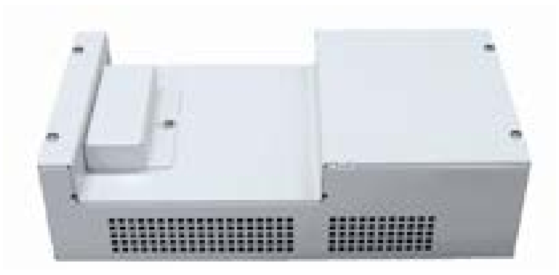
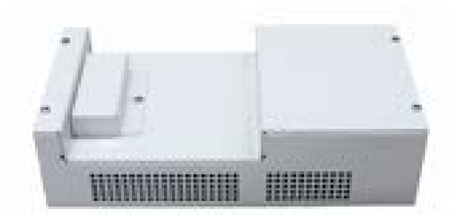
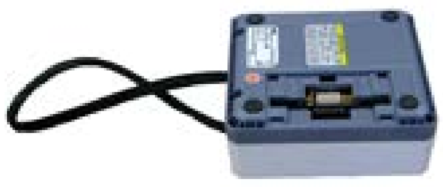
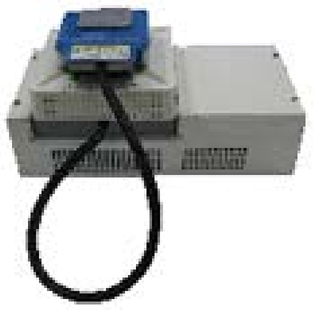


**Figure 5-1 QB-V850E2-SP**

### 5.1.2 Setup procedure

This section describes how to connect the QB-V850E2-SP to the IE850 main unit.

| | |
|---|---|
| 1. Remove the cover from the connector on the top side of the QB-V850E2-SP module. A screwdriver will be necessary to remove the cover. |  |
| 2. Remove the USB cable and power supply adaptor from the IE850 main unit, then remove the cover on the bottom side of the IE850 main unit. |  |
| 3. Connect the IE850 main unit on the QB-V850E2-SP as shown in the picture. Now connect the USB cable and power supply adaptor to the IE850 main unit. |  |

The IE850 automatically detects the expanded trace memory when the QB-V850E2-SP is connected. Please set the capacity of the trace memory in the debugger.

### 5.1.3 Cautionary note when using the QB-V850E2-SP

(1) Support of QB-V850E2-SP and debugger

The QB-V850E2-SP can only be used in the Multi integrated environment provided by Green Hills Software, a U.S. company. It cannot be used in the integrated environments produced by Renesas for developing microcontroller software.

(2) Break when QBV850E2-SP is in use

When the QB-V850E2-SP is in use, the trace-full break function is not available.

# 6. Maintenance and Warranty

This chapter covers basic maintenance, warranty information, provisions for repair and the procedures for requesting a repair.

## 6.1 Maintenance

(1) If dust or dirt collects on this product, wipe it off with a dry soft cloth. Do not use thinner or other solvents because these chemicals can cause the surface coating to separate.

(2) When you do not use this product for a long period, disconnect it from the power supply, host machine, and user system.

## 6.2 Warranty

(1) This product comes with a one-year warranty after purchase.

Should the product break down or be damaged while you're using it under normal conditions in accord with its user's manual, it will be repaired or replaced free of cost.

(2) However, if the following types of failure or damage to the product occur during the term of the warranty, repairing or replacing the product will incur a cost.

a) Failure or damage attributable to the misuse or abuse of the product or its use under other abnormal conditions.

b) Failure or damage attributable to improper handling of the product after purchase, such as dropping the product while it is being transported or otherwise moved.

c) Failure or damage to the product caused by other pieces of equipment connected to it.

d) Failure or damage attributable to fire, earthquakes, thunderbolts, floods, or other natural disasters, or to abnormal voltages, etc.

e) Failure or damage attributable to modifications, repairs, adjustments, or other acts in relation to the product by parties other than Renesas Electronics Corp.

(3) Consumables (e.g., sockets and adaptors) are beyond the scope of repair and replacement.

In the above cases, contact your local distributor. If you are renting the product, consult the company you are renting it from or the owner.

## 6.3 Repair provisions

(1) Repairs not covered by warranty

Problems arising in products for which more than one year has elapsed since purchase are not covered by warranty.

(2) Replacement not covered by warranty

If your product's fault falls into any of the following categories, the fault will be corrected by replacing the entire product instead of repairing it, or you will be advised to purchase a new product, depending on the severity of the fault.

- Faulty or broken mechanical portions
- Flaws, separation, or rust in coated or plated portions
- Flaws or cracks in plastic portions
- Faults or breakage caused by improper use or unauthorized repair or modification
- Heavily damaged electric circuits due to overvoltage, overcurrent or shorting of power supply
- Cracks in the printed circuit board or burnt-down patterns
- A wide range of faults that make replacement less expensive than repair
- Faults that are not locatable or identifiable

(3) Expiration of the repair period

When a period of one year has elapsed after production of a given model ceased, repairing products of that model may become impossible.

(4) Carriage fees for sending your product to be repaired

Carriage fees for sending your product to us for repair are at your own expense.

## 6.4 How to request repairs

If your product is found faulty, fill in a Repair Request Sheet downloadable from the following URL and email the sheet and send the product to your local distributor.

http://www.renesas.com/repair

⚠ **CAUTION**

Note on Transporting the Product:

❗ When sending your product for repair, use the packing box and cushioning material supplied with the MCU unit when it was delivered to you and specify caution in handling (handling as precision equipment). If packing of your product is not complete, it may be damaged during transportation. When you pack your product in a bag, make sure to use the conductive plastic bag supplied with the MCU unit (usually a blue bag). If you use a different bag, it may lead to further trouble with your product due to static electricity.

# Appendix A Characteristics of Target Interface

The target interface (interface carrying the signals which connect the in-circuit emulator and target system) behaves as if the actual device is connected in terms of functionality. However, in terms of characteristics, the behavior of the target interface sometimes differs from the behavior when the actual device is connected. The various equivalent circuits below apply to the target interface of this product.

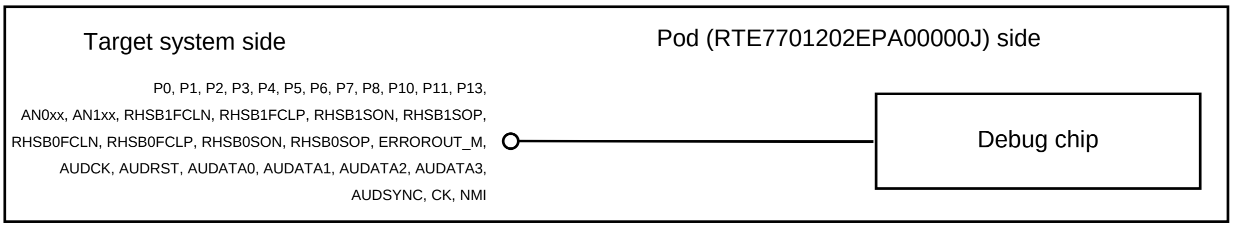

Target system side
Pod (RTE7701202EPA00000J) side
P0, P1, P2, P3, P4, P5, P6, P7, P8, P10, P11, P13,
AN0xx, AN1xx, RHSB1FCLN, RHSB1FCLP, RHSB1SON, RHSB1SOP,
RHSB0FCLN, RHSB0FCLP, RHSB0SON, RHSB0SOP, ERROROUT_M,
AUDCK, AUDRST, AUDATA0, AUDATA1, AUDATA2, AUDATA3,
AUDSYNC, CK, NMI
Debug chip

**Figure A-1 Equivalent Circuit A**

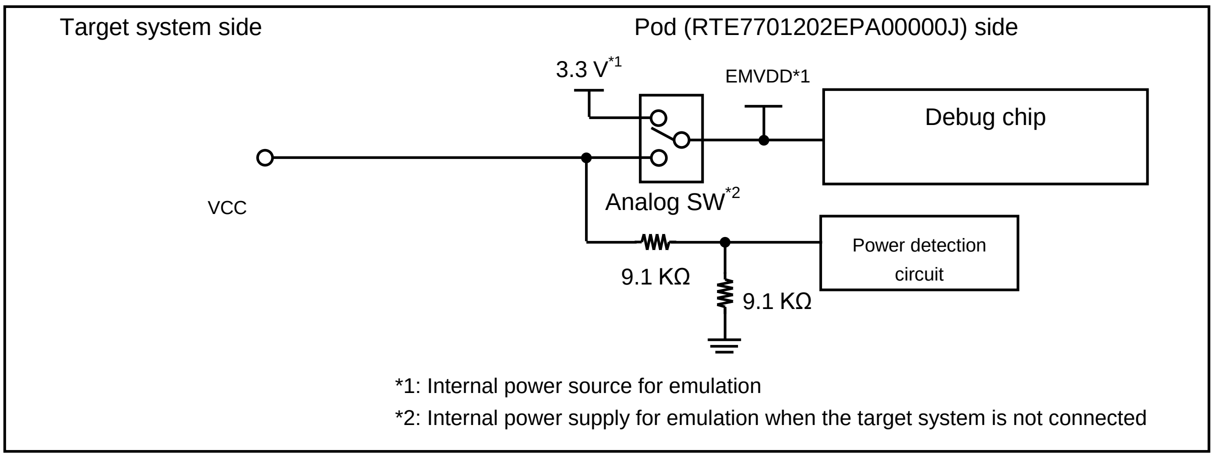

Target system side
Pod (RTE7701202EPA00000J) side
3.3 V*1
EMVDD*1
Debug chip
VCC
Analog SW*2
Power detection circuit
9.1 KΩ
9.1 KΩ
*1: Internal power source for emulation
*2: Internal power supply for emulation when the target system is not connected

**Figure A-2 Equivalent Circuit B**

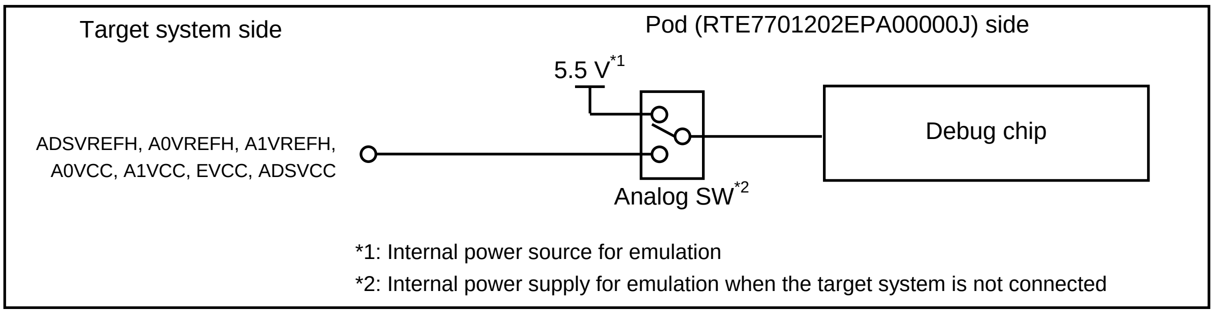

Target system side
Pod (RTE7701202EPA00000J) side
5.5 V*1
ADSVREFH, A0VREFH, A1VREFH,
A0VCC, A1VCC, EVCC, ADSVCC
Debug chip
Analog SW*2
*1: Internal power source for emulation
*2: Internal power supply for emulation when the target system is not connected

**Figure A-3 Equivalent Circuit C**

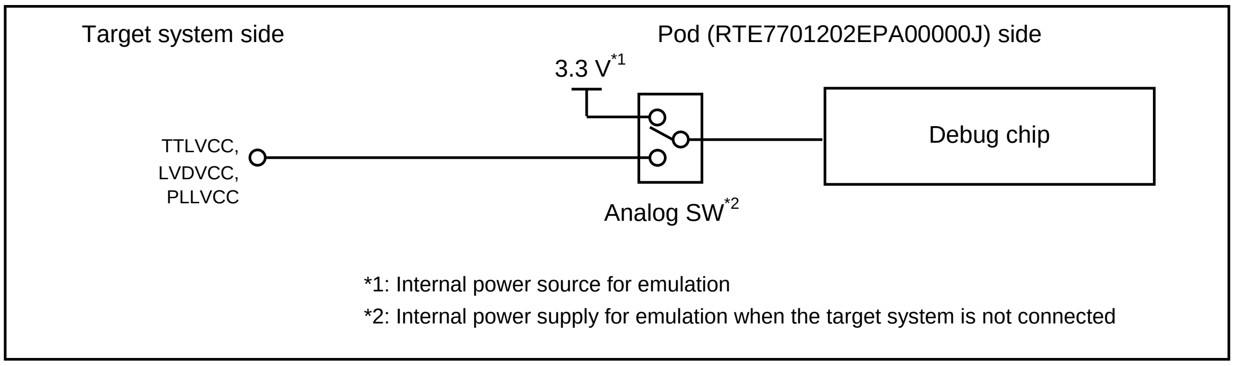

Target system side
Pod (RTE7701202EPA00000J) side
3.3 V*1
TTLVCC,
LVDVCC,
PLLVCC
Analog SW*2
Debug chip
*1: Internal power source for emulation
*2: Internal power supply for emulation when the target system is not connected

**Figure A-4 Equivalent Circuit D**

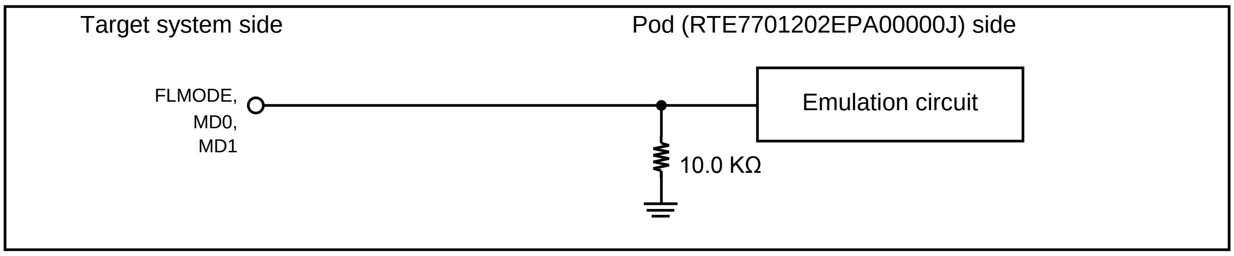


**Figure A-5 Equivalent Circuit E**

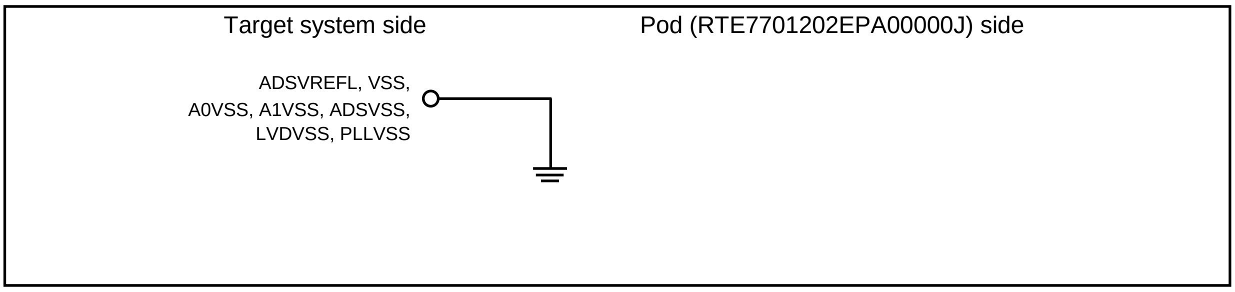


**Figure A-6 Equivalent Circuit F**

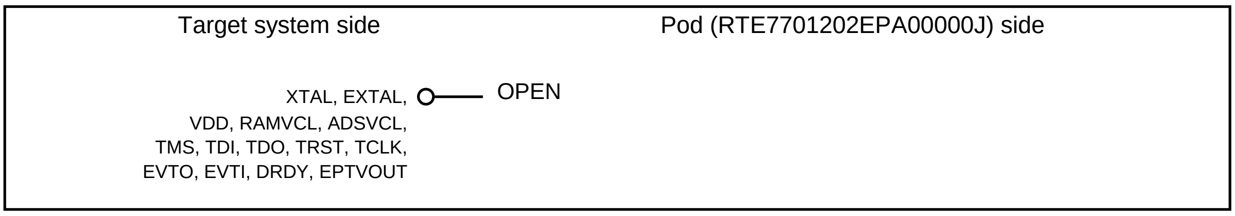
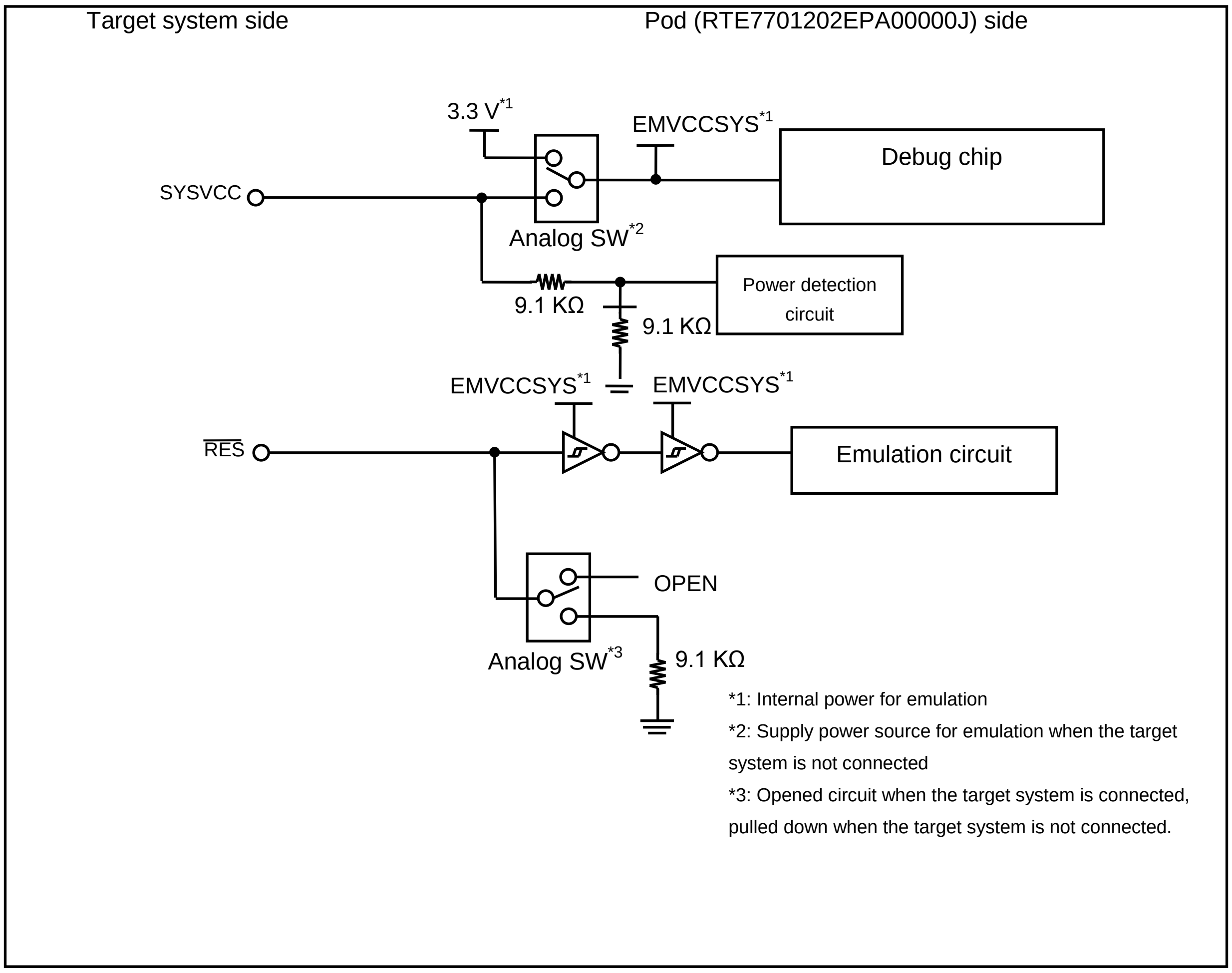

Target system side
Pod (RTE7701202EPA00000J) side
XTAL, EXTAL,
VDD, RAMVCL, ADSVCL,
TMS, TDI, TDO, TRST, TCLK,
EVTO, EVTI, DRDY, EPTVOUT
OPEN

**Figure A-7 Equivalent Circuit G**



*1: Internal power for emulation
*2: Supply power source for emulation when the target system is not connected
*3: Opened circuit when the target system is connected, pulled down when the target system is not connected.

**Figure A-8 Equivalent Circuit H**

| Revision History | Pod for IE850 In-circuit Emulator<br>RTE7701202EPA00000J User's Manual |
|---|---|

| Rev. | Date | Description | |
|---|---|---|---|
| | | Page | Summary |
| 1.00 | Sep 05, 2014 | — | First Edition issued |
| 2.00 | Apr 01, 2015 | 19 and 22 to 24 | With the support of DMA and global RAM access events, the following descriptions were added.<br>Table 1-3, System Specifications (1/2)<br>Section 1.3.3, Event function (Detection of specific operations) |
| | | 20 and 26 | With the addition of DMA and global RAM access cycles to the type of trace data, the following descriptions were added.<br>Table 1-3, System Specifications (2/2)<br>Table 1-6, List of Types of Trace Data |
| | | 20 | The following descriptions of performance measurement functions were added.<br>Table 1-3, System Specifications (2/2) |
| | | 28 | The following descriptions were added.<br>Section 1.3.6, Performance measurement |
| | | 29 | The following descriptions were added.<br>Section 1.3.9, Reset masking |
| | | 43 and 45 | Since initialization of the RAM area can be selected while the emulator is in use, the following descriptions were added.<br>Section 4.1.11, ECC errors<br>Section 4.2.3, Initialization of the RAM area |
| | | 45 | The title of section 4.2.4 was changed: Point for caution during breaks -> Reset of pins<br>The description on caution in the case when the debugger setting is for reset masking was changed. |
| | | 47 | A note was added to Table 4-1, Relation between the State of Emulation and Presence of a Reset Mask, describing the condition for avoiding the generation of a pin reset. |
| | | 47 | The following section was added.<br>4.2.13, Cautionary note on connecting an emulator (time required for preparing to communicate) |
| | | 48 | The following sections were added.<br>4.2.14, Cautionary note on connecting an emulator (internal reset)<br>4.2.15, Cautionary note on asynchronous debugging mode (peripheral break function)<br>4.2.16, Cautionary note on asynchronous debugging mode (reset)<br>4.2.17, Cautionary note on asynchronous debugging mode (watchdog timer)<br>4.2.18, Cautionary note on asynchronous debugging mode (ECC error)<br>4.2.19, Cautionary note on asynchronous debugging mode (specific sequence) |

Pod for IE850 In-circuit Emulator
RTE7701202EPA00000J
User’s Manual

Publication Date: Apr 01, 2015 Rev.2.00

Published by: Renesas Electronics Corporation

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

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

**Renesas Electronics America Inc.**
2801 Scott Boulevard Santa Clara, CA 95050-2549, U.S.A.
Tel: +1-408-588-6000, Fax: +1-408-588-6130

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

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

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

**Renesas Electronics (China) Co., Ltd.**
Room 1709, Quantum Plaza, No.27 ZhiChunLu Haidian District, Beijing 100191, P.R.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, P. R. China 200333
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, Unit #06-02 Hyflux Innovation Centre, Singapore 339949
Tel: +65-6213-0200, Fax: +65-6213-0300

**Renesas Electronics Malaysia Sdn.Bhd.**
Unit 1207, Block B, Menara Amcorp, Amcorp Trade Centre, No. 18, Jln Persiaran Barat, 46050 Petaling Jaya, Selangor Darul Ehsan, Malaysia
Tel: +60-3-7955-9390, Fax: +60-3-7955-9510

**Renesas Electronics India Pvt. Ltd.**
No.777C, 100 Feet Road, HALII Stage, Indiranagar, Bangalore, India
Tel: +91-80-67208700, Fax: +91-80-67208777

**Renesas Electronics Korea Co., Ltd.**
12F., 234 Teheran-ro, Gangnam-Gu, Seoul, 135-080, Korea
Tel: +82-2-558-3737, Fax: +82-2-558-5141

Colophon 4.0

# Pod for IE850 In-circuit Emulator

RTE7701202EPA00000J

User's Manual

Renesas Electronics Corporation

R20UT2980EJ0200