E2 Emulator

Additional Document for User’s Manual

(Setting up Interlocked Debugging of the DFP in an RH850/U2B-Series Device)

Supported Devices:

RH850 Family
RH850/U2B Series
DFP-Equipped Devices

All information contained in these materials, including products and product specifications, represents information on the product at the time of publication and is subject to change by Renesas Electronics Corp. without notice. Please review the latest information published by Renesas Electronics Corp. through various means, including the Renesas Electronics Corp. website (http://www.renesas.com).
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 systems. Renesas Electronics hereby expressly disclaims any and all liability for any losses and damages incurred by you or third parties arising from the use of these circuits, software, or information.

2. Renesas Electronics hereby expressly disclaims any warranties against and liability for infringement or any other claims involving patents, copyrights, or other intellectual property rights of third parties, by or arising from the use of Renesas Electronics products or technical information described in this document, including but not limited to, the product data, drawings, charts, programs, algorithms, and application examples.

3. No license, express, implied or otherwise, is granted hereby under any patents, copyrights or other intellectual property rights of Renesas Electronics or others.

4. You shall be responsible for determining which licenses are required from any third parties, and obtaining such licenses for the lawful import, export, manufacture, sales, utilization, distribution or other disposal of any products incorporating Renesas Electronics products, if required.

5. You shall not alter, modify, copy, or reverse engineer any Renesas Electronics product, whether in whole or in part. Renesas Electronics disclaims any and all liability for any losses or damages incurred by you or third parties arising from such alteration, modification, copying or reverse engineering.

6. Renesas Electronics products are classified according to the following two quality grades: “Standard” and “High Quality”. The intended applications for each Renesas Electronics product depends on the product’s quality grade, as indicated below.

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

   "High Quality": Transportation equipment (automobiles, trains, ships, etc.); traffic control (traffic lights); large-scale communication equipment; key financial terminal systems; safety control equipment; etc.

   Unless expressly designated as a high reliability product or a product for harsh environments in a Renesas Electronics data sheet or other Renesas Electronics document, Renesas Electronics products are not intended or authorized for use in products or systems that may pose a direct threat to human life or bodily injury (artificial life support devices or systems; surgical implantations; etc.), or may cause serious property damage (space system; undersea repeaters; nuclear power control systems; aircraft control systems; key plant systems; military equipment; etc.). Renesas Electronics disclaims any and all liability for any losses or damages incurred by you or third parties arising from the use of any Renesas Electronics product that is inconsistent with any Renesas Electronics data sheet, user’s manual or other Renesas Electronics document.

7. No semiconductor product is absolutely secure. Notwithstanding any security measures or features that may be implemented in Renesas Electronics hardware or software products, Renesas Electronics shall have absolutely no liability arising out of any vulnerability or security breach, including but not limited to any unauthorized access to or use of a Renesas Electronics product or a system that uses a Renesas Electronics product. RENESAS ELECTRONICS DOES NOT WARRANT OR GUARANTEE THAT RENESAS ELECTRONICS PRODUCTS, OR ANY SYSTEMS CREATED USING RENESAS ELECTRONICS PRODUCTS WILL BE INVULNERABLE OR FREE FROM CORRUPTION, ATTACK, VIRUSES, INTERFERENCE, HACKING, DATA LOSS OR THEFT, OR OTHER SECURITY INTRUSION (“Vulnerability Issues”). RENESAS ELECTRONICS DISCLAIMS ANY AND ALL RESPONSIBILITY OR LIABILITY ARISING FROM OR RELATED TO ANY VULNERABILITY ISSUES. FURTHERMORE, TO THE EXTENT PERMITTED BY APPLICABLE LAW, RENESAS ELECTRONICS DISCLAIMS ANY AND ALL WARRANTIES, EXPRESS OR IMPLIED, WITH RESPECT TO THIS DOCUMENT AND ANY RELATED OR ACCOMPANYING SOFTWARE OR HARDWARE, INCLUDING BUT NOT LIMITED TO THE IMPLIED WARRANTIES OF MERCHANTABILITY, OR FITNESS FOR A PARTICULAR PURPOSE.

8. When using Renesas Electronics products, refer to the latest product information (data sheets), user’s manuals, application notes, “General Notes for Handling and Using Semiconductor Devices” in the reliability handbook, etc.), and ensure that usage conditions are within the ranges specified by Renesas Electronics with respect to maximum ratings, operating power supply voltage range, heat dissipation characteristics, installation, etc. Renesas Electronics disclaims any and all liability for any malfunctions, failure or accident arising out of the use of Renesas Electronics products outside of such specified ranges.

9. Although Renesas Electronics endeavors to improve the quality and reliability of Renesas Electronics products, semiconductor products have specific characteristics, such as the occurrence of failure at a certain rate and malfunctions under certain use conditions. Unless designated as a high reliability product or a product for harsh environments in a Renesas Electronics data sheet or other Renesas Electronics document, Renesas Electronics products are not subject to radiation resistance design. You are responsible for implementing safety measures to guard against the possibility of bodily injury, injury or damage caused by fire, and/or danger to the public in the event of a failure or malfunction of Renesas Electronics products, such as safety design for hardware and software, including but not limited to redundancy, fire control and malfunction prevention, appropriate treatment for aging degradation or any other appropriate measures. Because the evaluation of microcomputer software alone is very difficult and impractical, you are responsible for evaluating the safety of the final products or systems manufactured by you.

10. Please contact a Renesas Electronics sales office for details as to environmental matters such as the environmental compatibility of each Renesas Electronics product. You are responsible for carefully and sufficiently investigating applicable laws and regulations that regulate the inclusion or use of controlled substances, including without limitation, the EU RoHS Directive, and using Renesas Electronics products in compliance with all these applicable laws and regulations. Renesas Electronics disclaims any and all liability for damages or losses occurring as a result of your noncompliance with applicable laws and regulations.

11. Renesas Electronics products and technologies shall not be used for or incorporated into any products or systems whose manufacture, use, or sale is prohibited under any applicable domestic or foreign laws or regulations. You shall comply with any applicable export control laws and regulations promulgated and administered by the governments of any countries asserting jurisdiction over the parties or transactions.

12. It is the responsibility of the buyer or distributor of Renesas Electronics products, or any other party who distributes, disposes of, or otherwise sells or transfers the product to a third party, to notify such third party in advance of the contents and conditions set forth in this document.

13. This document shall not be reprinted, reproduced or duplicated in any form, in whole or in part, without prior written consent of Renesas Electronics.

14. Please contact a Renesas Electronics sales office if you have any questions regarding the information contained in this document or Renesas Electronics products.

(Note1) “Renesas Electronics” as used in this document means Renesas Electronics Corporation and also includes its directly or indirectly controlled subsidiaries.

(Note2) "Renesas Electronics product(s)" means any product developed or manufactured by or for Renesas Electronics.

(Rv.5.0-1  October 2020)

Contact information
For further information on a product, technology, the most up-to-date version of a document, or your nearest sales office, please visit: www.renesas.com/contact/

Corporate Headquarters
TOYOSU FORESIA, 3-2-24 Toyosu, Koto-ku, Tokyo 135-0061, Japan
www.renesas.com

Trademarks
Renesas and the Renesas logo are trademarks of Renesas Electronics Corporation. All trademarks and registered trademarks are the property of their respective owners.

© 2022 Renesas Electronics Corporation. All rights reserved.
# Table of Contents

1. Outline .......................................................................................................................... 1  
   1.1 This manual .................................................................................................................. 1  
   1.2 Configuration of manuals .......................................................................................... 1  

2. System Configuration ........................................................................................................ 2  
   2.1 Hardware configuration ............................................................................................... 2  
   2.2 Software configuration ............................................................................................... 3  

3. Setting up the Environment for Interlocked Debugging of the G4MH Cores and DFP ........ 4  
   3.1 Setting up RH850/U2B-series devices ......................................................................... 4  
   3.2 Setting up the CS+ debugger ...................................................................................... 5  
   3.3 Setting up VirtualBox .................................................................................................. 7  
   3.4 Setting up the open OCD ............................................................................................ 10  
   3.5 Setting up the NSI IDE ............................................................................................... 15  
   3.6 Procedure for establishing a system for interlocked debugging of the G4MH cores and DFP 19  

4. Interlocked Debugging of the G4MH Cores and DFP ...................................................... 20  
   4.1 Interlocked execution of the G4MH cores and DFP ...................................................... 20  
   4.2 Interlocked breaks of the G4MH cores and DFP ......................................................... 21  

5. Inquiries ........................................................................................................................... 22
1. Outline

1.1 This manual

This manual explains how to set up the environment for interlocked debugging of an RH850/U2B-series device based on G4MH cores that also incorporates a DFP IP module (hereinafter referred to as the DFP) with both G4MH cores and the DFP as targets for debugging and to proceed with debugging by using an E2 emulator.

1.2 Configuration of manuals

This section describes documents that include information required for debugging an RH850/U2B device by using the E2 emulator.


<table>
<thead>
<tr>
<th>Name of Document</th>
<th>Document No.</th>
</tr>
</thead>
<tbody>
<tr>
<td>E2 Emulator RTE0T00020KCE00000R User’s Manual</td>
<td>R20UT3538E</td>
</tr>
</tbody>
</table>

In the interlocked debugging of G4MH cores and the DFP in a DFP-equipped RH850/U2B-series device, the CS+ integrated development environment (CS+ debugger) is required as the debugger for the G4MH cores. For use of the CS+ debugger, be sure to read its online help.

<table>
<thead>
<tr>
<th>Name of Document</th>
<th>URL</th>
</tr>
</thead>
<tbody>
<tr>
<td>CS+ online help</td>
<td>[CS+ Online Help (renesas.com)]</td>
</tr>
</tbody>
</table>

The interlocked debugging of G4MH cores and the DFP in a DFP-equipped RH850/U2B-series device requires the DR1000C debugger and IDE from NSITEXE for debugging of the DFP. To use the DR1000C debugger and IDE, be sure to read the following documents.

<table>
<thead>
<tr>
<th>Name of Document</th>
</tr>
</thead>
<tbody>
<tr>
<td>NSITEXE DR1000C Debugger User Manual</td>
</tr>
<tr>
<td>NSITEXE DR1000C IDE User Manual</td>
</tr>
</tbody>
</table>
2. System Configuration

This chapter describes the system configuration for interlocked debugging of the G4MH cores and DFP in a DFP-equipped RH850/U2B-series device.

2.1 Hardware configuration

Figure 2-1 and Table 2-1 show the hardware configuration for interlocked debugging of the G4MH cores and DFP in a DFP-equipped RH850/U2B-series device.

![Figure 2-1   Hardware Configuration](image)

### Table 2-1 Hardware Configuration

<table>
<thead>
<tr>
<th>Name</th>
<th>Product Name</th>
<th>Remarks</th>
</tr>
</thead>
<tbody>
<tr>
<td>Emulator debugger</td>
<td>CS+ debugger from Renesas</td>
<td>—</td>
</tr>
<tr>
<td></td>
<td>Debugger and IDE from NSITEXE</td>
<td>Included in the DR1000C SDK from NSITEXE.</td>
</tr>
<tr>
<td>Emulator</td>
<td>E2 emulator</td>
<td>—</td>
</tr>
<tr>
<td>User-system interface cable</td>
<td>User-system interface cable</td>
<td>Provided with the E2 emulator main unit.</td>
</tr>
<tr>
<td>Debugging interface</td>
<td>JTAG interface</td>
<td>LPD interface is not available.</td>
</tr>
<tr>
<td>User system</td>
<td>RH850/U2B Piggyback Board, etc.</td>
<td>Board having a DFP-equipped RH850/U2B-series device</td>
</tr>
</tbody>
</table>
2.2 Software configuration

Table 2-2 lists the software configuration for interlocked debugging of the G4MH cores and DFP in a DFP-equipped RH850/U2B-series device.

**Table 2-2 Software Configuration**

<table>
<thead>
<tr>
<th>Name</th>
<th>Product Name</th>
<th>Remarks</th>
</tr>
</thead>
<tbody>
<tr>
<td>Emulator debugger for the G4MH cores in the RH850 device</td>
<td>CS+ debugger from Renesas</td>
<td></td>
</tr>
<tr>
<td>Emulator debugger for the DFP in the RH850 device</td>
<td>Debugger and IDE from NSITEXE</td>
<td>Use the IDE and open OCD included in the DR1000C SDK from NSITEXE.</td>
</tr>
<tr>
<td>Virtual OS environment</td>
<td>Oracle VM VirtualBox</td>
<td>Use the Ubuntu 20.04 LTS environment to use the DR1000C debugger and IDE under Linux.</td>
</tr>
</tbody>
</table>
3. Setting up the Environment for Interlocked Debugging of the G4MH Cores and DFP

This chapter describes how to set up the environment for interlocked debugging of the G4MH cores and DFP.

3.1 Setting up RH850/U2B-series devices


This section describes setting up of an RH850/U2B-series device and points for caution on interlocked debugging of the G4MH cores and DFP.

3.1.1 DFP-related option bytes

The following option byte must be set to debug the DFP installed in an RH850/U2B-series device.

- S_OPBT0: MPSELECT (Set the bit in DFP chain mode.)

The DFP states and reset vectors at the time power is supplied must be set with the following option bytes.

- OPBT38: DFP_init_boothart : 0000 0000H (setting value)
- OPBT39: DFP_resetvec: 0030 0000H (setting value)

3.1.2 Address map for the DFP installed in an RH850/U2B-series device

The following shows the address map for debugging the DFP installed in an RH850/U2B-series device.

<table>
<thead>
<tr>
<th>Start Address</th>
<th>End Address</th>
<th>Area</th>
</tr>
</thead>
<tbody>
<tr>
<td>0030 0000H</td>
<td>003F FFFFH</td>
<td>External ROM</td>
</tr>
<tr>
<td>F000 0000H</td>
<td>F000 7FFFH</td>
<td>Control Core Unit (CCU) Local Memory</td>
</tr>
<tr>
<td>F004 0000H</td>
<td>F004 1FFFH</td>
<td>Scalar Processor Unit (SPU) 0 Local Memory</td>
</tr>
<tr>
<td>F004 4000H</td>
<td>F004 5FFFH</td>
<td>SPU 1 Local Memory</td>
</tr>
<tr>
<td>F004 8000H</td>
<td>F004 9FFFH</td>
<td>SPU 2 Local Memory</td>
</tr>
<tr>
<td>F004 C000H</td>
<td>F004 DFFFH</td>
<td>SPU 3 Local Memory</td>
</tr>
<tr>
<td>F005 0000H</td>
<td>F005 005FH</td>
<td>Mutual Exclusion Register Variable (MERV)</td>
</tr>
<tr>
<td>F010 0000H</td>
<td>F013 FFFFH</td>
<td>Vector Processor Unit (VPU) Local Memory (VLM)</td>
</tr>
<tr>
<td>F080 0000H</td>
<td>F0BF FFFFH</td>
<td>DFP Peripheral</td>
</tr>
<tr>
<td>F800 0000H</td>
<td>F81F FFFFH</td>
<td>DFP Debug</td>
</tr>
<tr>
<td>FE04 0000H</td>
<td>FE07 FFFFH</td>
<td>External RAM</td>
</tr>
</tbody>
</table>
3.2 Setting up the CS+ debugger

For details on the CS+ debugger, refer to the CS+ online help system. This section describes setting up of the CS+ debugger and points for caution on interlocked debugging of the G4MH cores and DFP.

3.2.1 Settings for interlocked debugging of the G4MH cores and DFP

Property settings for the E2 emulator must be made in the CS+ debugger to debug the DFP installed in an RH850/U2B-series device. Additional settings are also required for interlocked debugging of the G4MH cores and DFP. Table 3-2 and Table 3-3 list the settings of the properties and Figure 3-1 and Figure 3-2 show the windows for reference.

<table>
<thead>
<tr>
<th>Item</th>
<th>Setting</th>
</tr>
</thead>
<tbody>
<tr>
<td>Communications method</td>
<td>JTAG</td>
</tr>
<tr>
<td>Debug the DFP function</td>
<td>Yes</td>
</tr>
<tr>
<td>IP address of the server</td>
<td>IP address of the GDB server specified by the open OCD</td>
</tr>
<tr>
<td></td>
<td>Example: 192.168.56.1</td>
</tr>
<tr>
<td>Port number of the server</td>
<td>Port number of the GDB server specified by the open OCD</td>
</tr>
<tr>
<td></td>
<td>Example: 9824</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Item</th>
<th>Setting</th>
</tr>
</thead>
<tbody>
<tr>
<td>Use the DFP interlocking Resume/the DFP interlocking Halt function</td>
<td>Yes</td>
</tr>
<tr>
<td>Timeout time for waiting for the DFP execution request [s]</td>
<td>Timeout time until issuing a request for executing the DFP after having issued a request for execution by the G4MH cores Example: 20</td>
</tr>
<tr>
<td>Stop emulation of peripherals when stopping</td>
<td>Yes</td>
</tr>
</tbody>
</table>

Figure 3-1 Reference for Setting the Properties of the E2 Emulator in CS+ for Interlocked Debugging of the G4MH Cores and DFP ([Connect Settings] Tabbed Page)
Figure 3-2  Reference for Setting the Properties of the E2 Emulator in CS+ for Interlocked Debugging of the G4MH Cores and DFP ([Debug Tool Settings] Tabbed Page)
3.3 Setting up VirtualBox

Set up the DR1000C debugger and IDE for the virtual Linux running in VirtualBox to use the DR1000C debugger and IDE for debugging of the DFP under Linux.

3.3.1 Installing VirtualBox

Download the installer for Oracle VM VirtualBox from the Oracle home page. Start the installer and proceed with standard installation according to the installer.

3.3.2 Starting VirtualBox and Setting the Virtual Hard Disk

Start VirtualBox. Select the [Tool] -> [Preference] -> [General] menu and specify the destination for storing the virtual hard disk (virtual HD) for the virtual environment. Then, create the virtual HD from the [Tool] -> [New] menu item. Table 3-4 lists the recommended settings and Figure 3-3 and Figure 3-4 show the windows for reference.

Table 3-4 Recommended Settings of the Virtual HD of VirtualBox for Interlocked Debugging of the G4MH Cores and DFP

<table>
<thead>
<tr>
<th>Item</th>
<th>Setting</th>
</tr>
</thead>
<tbody>
<tr>
<td>Memory size</td>
<td>4 GB (1 GB by default)</td>
</tr>
<tr>
<td>File size</td>
<td>30 GB (because the 10-GB default size is insufficient)</td>
</tr>
<tr>
<td>Hard disk file type</td>
<td>VHD</td>
</tr>
<tr>
<td>Storage on physical hard disk</td>
<td>Fixed size</td>
</tr>
</tbody>
</table>

![Create Virtual Machine](Figure 3-3 Reference for Recommended Settings of the Virtual HD of VirtualBox (Memory Size))
3.3.3 Downloading the Ubuntu image file

Download the ISO image file for Ubuntu 20.04 LTS from the Ubuntu home page.

3.3.4 Preparing for creation of the virtual machine

Select the created virtual HD and [Settings] -> [Storage] and click on the [Empty] disk icon for the storage device. Click on the disk icon at the right of the [Optical Drive] drop-down list in the [Attributes] item and then [Choose a disk file...]. Mount the Ubuntu image file (iso) and click on the [OK] button. Figure 3-5 shows the window for reference.
3.3.5 Installing Ubuntu

Clicking on [Start] in VirtualBox starts the installation of Ubuntu. Select the language for the virtual environment and click on [Install Ubuntu]. Proceed with installation according to the installation window.

After installation, restart the virtual environment, enter the user name and password, and log in to the virtual environment. Then update the software and restart the virtual environment.

3.3.6 Configuring the virtual environment

Select [Open Terminal] to open a terminal and install the build-essential software under Ubuntu; that is, make the use of GCC, MAKE, and GDB available.

3.3.7 Connecting a PC to the virtual machine

To connect a PC (Windows) to the virtual machine (Ubuntu), confirm the IP address of the PC and set an IP address for VirtualBox by clicking on [Settings] -> [Network] -> [Advanced] -> [Port Forwarding] -> [Adds new port forwarding rule.]. Table 3-5 lists the settings and Figure 3-6 shows the window for reference.

Table 3-5 Connecting the PC to the Virtual Machine

<table>
<thead>
<tr>
<th>Item</th>
<th>Setting</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>As desired</td>
</tr>
<tr>
<td>Protocol</td>
<td>TCP</td>
</tr>
<tr>
<td>Host IP</td>
<td>IPv4 address of the PC</td>
</tr>
<tr>
<td>Host port</td>
<td>Port number that does not coincide with the guest port number, a reserved port number, or the number of a port that is in use for another purpose</td>
</tr>
<tr>
<td>Guest IP</td>
<td>IPv4 address of the virtual machine</td>
</tr>
<tr>
<td>Guest port</td>
<td>Port number that does not coincide with a reserved port number or the number of a port that is in use for another purpose</td>
</tr>
</tbody>
</table>

Figure 3-6 Reference for Settings for Connecting the PC to the VirtualBox Virtual Machine
3.4 Setting up the open OCD

For details on the open OCD, refer to the DR1000C Debugger User Manual from NSITEXE. This section describes setting up of the open OCD and points for caution on interlocked debugging of the G4MH cores and DFP.

For the open OCD, use version 1.1.1 which has been pre-released as the next update.

3.4.1 Setting the IP addresses, port numbers and hart to debug

The IP addresses and port numbers set for the open OCD must match those specified in the CS+ debugger. If different IP addresses or port numbers are set, the DFP cannot be debugged.

The following shows an example of the open OCD configuration file when the DFP is to be debugged with five and nine instances of hart. If necessary, change the specification of the hart to be debugged. The hart ID for debugging can be selected from 0 and 16-31. However, the maximum number of harts that can be debugged at the same time is the number listed in 3.5.1. Also, be sure to debug hart0. In the example of the configuration file to be debugged with 9 harts, the difference from the setting to debug with 5 harts is described, so please refer to it when changing the specification of the hart to be debugged.
Open OCD configuration file (example of 5 harts debugging)

```bash
# Debug Adapter Configuration
interface remote_bitbang
remote_bitbang_host 192.168.56.1   ★ Specify [IP address of the server] which was specified in the CS+ debugger.
remote_bitbang_port 9824           ★ Specify [Port number of the server] which was specified in the CS+ debugger.

# Tap Declaration
set _CHIPNAME riscv
set _DAP_TAPID 0x100039df
set _ENDIAN little

jtag newtap $_CHIPNAME dap -irlen 5 -ircapture 0x01 -irmask 0x03 -expected-id $_DAP_TAPID

set _TARGETNAME  $_CHIPNAME.cpu0
set _TARGETNAME_16 $_CHIPNAME.cpu16
set _TARGETNAME_20 $_CHIPNAME.cpu20
set _TARGETNAME_24 $_CHIPNAME.cpu24
set _TARGETNAME_28 $_CHIPNAME.cpu28

target create $_TARGETNAME  riscv -endian $_ENDIAN -chain-position $_CHIPNAME.dap -coreid 0 -rtos hwthread
target create $_TARGETNAME_16 riscv -endian $_ENDIAN -chain-position $_CHIPNAME.dap -coreid 16
target create $_TARGETNAME_20 riscv -endian $_ENDIAN -chain-position $_CHIPNAME.dap -coreid 20
target create $_TARGETNAME_24 riscv -endian $_ENDIAN -chain-position $_CHIPNAME.dap -coreid 24
target create $_TARGETNAME_28 riscv -endian $_ENDIAN -chain-position $_CHIPNAME.dap -coreid 28

target smp $_TARGETNAME $_TARGETNAME_16 $_TARGETNAME_20 $_TARGETNAME_24 $_TARGETNAME_28
$_TARGETNAME configure -event gdb-attach { halt }

riscv set_reset_timeout_sec   1200
riscv set_command_timeout_sec 1200
riscv set_enable_virt2phys off

puts "Before init"
# Server Configuration
gdb_port 3333           ★ Port number for waiting for the connection with the GDB
gdb_report_data_abort enable
poll_period 500
init

# General Commands
bindto 0.0.0.0
```

targets $_TARGETNAME
Open OCD configuration file (example of 9 harts debugging)

```bash
# Debug Adapter Configuration
interface remote_bitbang
remote_bitbang_host 192.168.56.1  ★ Specify [IP address of the server] which was specified in the CS+ debugger.
remote_bitbang_port 9824  ★ Specify [Port number of the server] which was specified in the CS+ debugger.

# Tap Declaration
set _CHIPNAME riscv
set _DAP_TAPID 0x100039df
set _ENDIAN little

jtag newtap $_CHIPNAME dap -irlen 5 -ircapture 0x01 -irmask 0x03 -expected-id $_DAP_TAPID

set _TARGETNAME  $ _CHIPNAME.cpu0
set _TARGETNAME_16 $ _CHIPNAME.cpu16
set _TARGETNAME_17 $ _CHIPNAME.cpu17  ★ Diff with 5 harts: Add hart17
set _TARGETNAME_20 $ _CHIPNAME.cpu20
set _TARGETNAME_21 $ _CHIPNAME.cpu21  ★ Diff with 5 harts: Add hart21
set _TARGETNAME_24 $ _CHIPNAME.cpu24
set _TARGETNAME_25 $ _CHIPNAME.cpu25  ★ Diff with 5 harts: Add hart25
set _TARGETNAME_28 $ _CHIPNAME.cpu28
set _TARGETNAME_29 $ _CHIPNAME.cpu29  ★ Diff with 5 harts: Add hart29

target create $_TARGETNAME  riscv -endian $_ENDIAN -chain-position $_CHIPNAME.dap -coreid 0 -rtos hwthread
target create $_TARGETNAME_16 riscv -endian $_ENDIAN -chain-position $_CHIPNAME.dap -coreid 16

Diff with 5 harts: Add hart17

target create $_TARGETNAME_17 riscv -endian $_ENDIAN -chain-position $_CHIPNAME.dap -coreid 17  ★ Diff with 5 harts: Add hart17

target create $_TARGETNAME_20 riscv -endian $_ENDIAN -chain-position $_CHIPNAME.dap -coreid 20

Diff with 5 harts: Add hart21

target create $_TARGETNAME_21 riscv -endian $_ENDIAN -chain-position $_CHIPNAME.dap -coreid 21  ★ Diff with 5 harts: Add hart21

target create $_TARGETNAME_24 riscv -endian $_ENDIAN -chain-position $_CHIPNAME.dap -coreid 24

Diff with 5 harts: Add hart25

target create $_TARGETNAME_25 riscv -endian $_ENDIAN -chain-position $_CHIPNAME.dap -coreid 25  ★ Diff with 5 harts: Add hart25

target create $_TARGETNAME_28 riscv -endian $_ENDIAN -chain-position $_CHIPNAME.dap -coreid 28

Diff with 5 harts: Add hart29

target create $_TARGETNAME_29 riscv -endian $_ENDIAN -chain-position $_CHIPNAME.dap -coreid 29  ★ Diff with 5 harts: Add hart29

target smp $_TARGETNAME $_TARGETNAME_16 $_TARGETNAME_20 $_TARGETNAME_24 $_TARGETNAME_28
$_TARGETNAME_17 $_TARGETNAME_21 $_TARGETNAME_25 $_TARGETNAME_29  ★ Diff with 5 harts: Add hart17, hart21, hart25, hart29

$_TARGETNAME configure -event gdb-attach { halt }
```
riscv set_reset_timeout_sec 1200
riscv set_command_timeout_sec 1200
riscv set_enable_virt2phys off

puts "Before init"
# Server Configuration
```bash
gdb_port 3333  # Port number for waiting for the connection with the GDB
gdb_report_data_abort enable
poll_period 500
init
```
# General Commands
```bash
bindto 0.0.0.0
targets $_TARGETNAME
```

### 3.4.2 Starting the open OCD

Open [Terminal] in the virtual environment (Ubuntu) and run the following command.

```bash
$ openocd --f (Open OCD configuration file)
```

### 3.4.3 Point for caution on starting the open OCD

When the open OCD is started, confirm that the CS+ debugger starts debugging of the G4MH cores ([Connect to Debug Tool] is enabled). Starting the open OCD will fail if the CS+ debugger does not start debugging of the G4MH cores.
3.5 Setting up the NSI IDE

For details on the NSI IDE, refer to the DR1000C IDE User Manual from NSITEXE. This section describes setting up of the NSI IDE and points for caution on interlocked debugging of the G4MH cores and DFP.

3.5.1 Number of instances of hart to be debugged

Up to nine instances of hart can be debugged when debugging the DFP installed in some RH850/U2B-series devices.

3.5.2 DFP program

When you create a new project for the NSI IDE, specify [RH850U2B] as the item for specifying the target board. When a newly created project is built, a DFP program (*.mot) file is generated. For the procedure for creating and building new projects from the NSI IDE, refer to the NSITEXE DR1000C IDE User Manual.

For interlocked debugging of a generated DFP program (*.mot), specify the file to be downloaded by clicking on [Project Tree] -> [RH850 E2 (Debug Tool)] -> [Download File Settings] -> [Download files] -> [[1]] in the CS+ debugger. Figure 3-7 shows the window for reference.

![Figure 3-7 Reference for Setting the Properties of the E2 Emulator in CS+ for Interlocked Debugging of the G4MH Cores and DFP ([Download File Settings] Tabbed Page)](image-url)
3. Setting up the Environment for Interlocked Debugging of the G4MH Cores and DFP

3.5.3 Connection between the open OCD and GDB

In the NSI IDE, connect the open OCD by specifying the IP address of the server on which the open OCD was started and the port number that is to wait for connection with the GDB.

Display the window with the following steps. Table 3-6 lists the settings and Figure 3-8 shows the window for reference.

1. Right-click on the name of the target project in the [Project Explorer] view of the NSI IDE and click on [Debug As] -> [Debug Configurations...] to display the [Debug Configurations] window.
2. Click on [GDB Hardware Debugging] -> [(project name)_Debug_TCPIP] -> [Debugger].

Table 3-6 Connecting the GDB to the Open OCD

<table>
<thead>
<tr>
<th>Item</th>
<th>Setting</th>
</tr>
</thead>
<tbody>
<tr>
<td>JTAG Device</td>
<td>Open OCD (via socket)</td>
</tr>
<tr>
<td>Host name or IP address</td>
<td>IP address of the server on which the open OCD was started. If the server is the same as that on which the NSI IDE was started, “local host” can also be specified.</td>
</tr>
<tr>
<td>Port number</td>
<td>Port number that waits for connection with the GDB specified by the open OCD</td>
</tr>
</tbody>
</table>

![Figure 3-8 Reference for Settings of [Debug Configurations] on the NSI IDE ([Debugger] Tabbed Page)](image)
3.5.4 Starting debugging of the DFP

After the settings for section 3.5.3,
Connection between the open OCD and GDB, have been made, clicking on [Debug] in the lower left of the [Debug Configurations] window starts debugging.

Confirm the settings in the window before starting debugging. Display the [Main] window by clicking on [GDB Hardware Debugging] -> [(project name)_Debug_TCPIP] in the [Debug Configurations] window. Table 3-7 lists the settings and Figure 3-9 shows the window for reference.

Table 3-7 Setting for Debugging of a Project

<table>
<thead>
<tr>
<th>Item</th>
<th>Setting</th>
</tr>
</thead>
<tbody>
<tr>
<td>C/C++ Application</td>
<td>ELF file. The NSI IDE generates a .run file as an ELF file.</td>
</tr>
<tr>
<td></td>
<td>The .run file is generated during the generation of a DFP program.</td>
</tr>
<tr>
<td></td>
<td>For this setting, select a .run file rather than a .mot file.</td>
</tr>
</tbody>
</table>

Figure 3-9 Reference for Settings of [Debug Configurations] of the NSI IDE ([Main] Tabbed Page)
3.6 Procedure for establishing a system for interlocked debugging of the G4MH cores and DFP

This section describes the procedure for establishing an environment for interlocked debugging of the G4MH cores and DFP.

(1) Make option byte settings for the DFP-equipped RH850/U2B-series device (see section 3.1.1).
(2) Set up interlocked debugging of the G4MH cores and DFP in the CS+ debugger (see section 3.2.1).
(3) Specify the DFP program to be debugged in the CS+ debugger (see section 3.5.2).
(4) Select [Connect to Debug Tool] from the [Debug] menu in the CS+ debugger to start debugging the G4MH cores.
(5) Start the Ubuntu environment in VirtualBox to provide a Linux environment for using the DR1000C debugger and IDE.
(6) Set the same IP address and port number of the open OCD for the DR1000C debugger and IDE as the settings in the CS+ debugger (see section 3.4.1).
(7) Start the open OCD for the DR1000C debugger and IDE (see section 3.4.2).
(8) Start the NSI IDE for the DR1000C debugger and IDE and set the connection between the open OCD and GDB (see section 0).
(9) Start debugging of the DFP in the NSI IDE for the DR1000C debugger and IDE (see section 0).

Debugging of the G4MH cores and DFP is now enabled in the CS+ debugger and NSI IDE, respectively.
4. Interlocked Debugging of the G4MH Cores and DFP

4.1 Interlocked execution of the G4MH cores and DFP

This section describes interlocked execution and points for caution on debugging of the G4MH cores and DFP.

4.1.1 Procedure for interlocked execution of the G4MH cores and DFP

When the G4MH cores and DFP are in break (DB-Halt) states during interlocked debugging, the procedure for interlocked execution of the G4MH cores and DFP is as follows.

1. Click on [Go] in the CS+ debugger.
2. Click on [Resume] in the NSI IDE within the timeout time for waiting for the request for execution by the DFP.

4.1.2 Points for caution on interlocked execution of the G4MH cores and DFP

No. 1 Clicking on [Resume] in the NSI IDE without clicking on [Go] in the CS+ debugger

During interlocked debugging of the G4MH cores and DFP, clicking on [Resume] in the NSI IDE without clicking on [Go] in the CS+ debugger leaves the G4MH cores in the break state and only the DFP starts execution of a program.

While only the DFP is executing a program, clicking on [Suspend] in the NSI IDE or halting the DFP at a breakpoint transfers the DFP to the DB-Halt state. Following the procedure for interlocked execution of the G4MH cores and DFP in this state starts interlocked execution by the G4MH cores and DFP and both the G4MH cores and DFP start execution of the given programs.

No. 2 Clicking on [Go] in the CS+ debugger while the DFP is executing a program

During interlocked debugging of the G4MH cores and DFP, if [Go] is clicked in the CS+ debugger while the DFP executes a program, an error will occur and the G4MH cores will not start execution of the program.

No. 3 Clicking on [Go] in the CS+ debugger but not clicking on [Resume] in the NSI IDE within the timeout time

During interlocked debugging of the G4MH cores and DFP, if the timeout time after [Go] is clicked in the CS+ debugger has elapsed without a click on [Resume] in the NSI IDE, an error will occur and the G4MH cores will not start executing programs.
4.2 Interlocked breaks of the G4MH cores and DFP

This section describes interlocked breaks and a point for caution on breaks in debugging of the G4MH cores and DFP.

4.2.1 Operation of interlocked breaks for the G4MH cores and DFP

When the G4MH cores and DFP are in the interlocked execution state for interlocked debugging of the G4MH cores and DFP and a break (halt) of any of the kinds listed below occurs for a G4MH core or the DFP, interlocked breaks occur in the G4MH cores and DFP so that the break (halt) is generally applicable.

- Clicking on [Stop] in the CS+ debugger (forced break for a G4MH core)
- Clicking on [Suspend] in the NSI IDE (forced halt of the DFP)
- Occurrence of a break for a G4MH core at a breakpoint that has been set in the CS+ debugger (event break for the G4MH core)
- Occurrence of a break for the DFP at a breakpoint that has been set in the NSI IDE (event break for the DFP)
5. Inquiries

If trouble arises or you need more information on usage during the debugging of a DFP-equipped RH850/U2B-series device, contact the following companies. If you do not know which company to ask, contact Renesas Electronics.

- Trouble or information on usage during the debugging of a G4MH core by using the CS+ debugger: Renesas Electronics
- Trouble or information on usage during debugging of the DFP by using the NSI IDE: NSITEXE
<table>
<thead>
<tr>
<th>Rev.</th>
<th>Date</th>
<th>Page</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.50</td>
<td>Apr.25.22</td>
<td>—</td>
<td>First Edition issued</td>
</tr>
<tr>
<td>1.00</td>
<td>Jul.20.22</td>
<td>10</td>
<td>Fixed 3.4.1 chapter title</td>
</tr>
<tr>
<td></td>
<td></td>
<td>10</td>
<td>Add description of 9 harts debugging in 3.4.1</td>
</tr>
<tr>
<td></td>
<td></td>
<td>13</td>
<td>Fixed 3.5.1 Number of instances of hart to be debugged from five to nine</td>
</tr>
</tbody>
</table>
(Setting up Interlocked Debugging of the DFP in an RH850/U2B-Series Device)

Publication Date: Rev.1.00 Jul.20.22

Published by: Renesas Electronics Corporation
E2 Emulator

Additional Document for User’s Manual

(Setting up Interlocked Debugging of the DFP in an RH850/U2B-Series Device)