User's Manual

R32C/100 E30A仿真调试器V.1.01

用户手册

瑞萨单片机开发环境系统

本资料所记载的内容，均为本资料发行时的信息，瑞萨电子对于本资料所记载的产品或者规格可能会作改动，恕不另行通知。
请通过瑞萨电子的主页确认发布的最新信息。

瑞萨电子
www.renesas.com

Rev.1.00　2010.08

Notice

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

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

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

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

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

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

前言

High-performance Embedded Workshop 是瑞萨单片机的图形用户界面，能容易开发和调试 C/C++ 语言和汇编语言编写的应用程序。提供高性能和直观的手段，实现对执行应用程序的仿真器或者模拟器的存取、测量和更改。

在本手册中，主要说明 High-performance Embedded Workshop 的 “调试器”功能。

对象系统

本调试器在仿真器 E30A 系统上运行。

对应 CPU

本手册说明对应以下 CPU 的调试功能。

- R32C/100 系列

  （注）取决于此 CPU 的信息，在本手册中记为 “用于 R32C”。

有关产品和本手册内容的联系方法

如果要就此文档或此产品内容进行咨询，请填写安装程序在以下目录中生成的文本文件，然后通过电子邮件发送给当地的经销商。

\SUPPORT\Product-name\SUPPORT.TXT

瑞萨工具主页：http://www.renesas.com/en/tools

目　录

教程篇 32

参考篇 48

启动 / 设置篇

1. 功能概要

1.1 RAM 监控功能

此功能能在目标程序执行过程中参照存储器的内容变化。
能定期地更新被显示的存储器内容。
但是，失去执行程序的实时性。
为了使 RAM 监控功能有效，必须进行以下的设置：

- 在[Event Setting]（事件设置）窗口或者RAM监控区域设置窗口，给事件E5设置RAM监控事件。
- 选择RAM监控对应的各窗口的菜单[RAM Monitor]（RAM监控）→[Enable RAM Monitor]（RAM监控功能有效化）。

必须在 Init 对话框的仿真器选项卡中指定在程序执行过程中获取存储器数据的间隔。

1.1.1 RAM 监控区域

本调试器有 1K 字节的 RAM 监控区域。此 RAM 监控区域能分配到任意的连续地址，或者能以 16 字节为单位拆分为 64 块区域。

1.1.2 采样周期

采样周期是指显示的更新间隔。
能在 RAM 监控对应的各窗口指定 （默认值：1 秒）。
根据运行情况，会慢于指定的采样周期 （取决于以下情况）。

- 通信接口
- [RAM Monitor]（RAM监控）窗口的显示窗口的个数
- [RAM Monitor]（RAM监控）窗口的显示窗口的大小
- [ASM Watch]（ASM监视）窗口的RAM监控区域内的ASM监视点数
- [C Watch]（C监视）窗口的RAM监控区域内的C监视点数

1.1.3 相关窗口

能使用 RAM 监控功能的窗口如下所示：

- [RAM Monitor]（RAM监控）窗口
- [ASM Watch]（ASM监视）窗口
- [C Watch]（C监视）窗口

1.2 暂停功能

本调试器支持以下的暂停功能。

1.2.1 软件暂停功能

软件暂停功能在执行指定地址的指令前停止执行目标程序。

此暂停的点称为软件断点。在 [Editor(Source)] （编辑器 （源））窗口或者 [S/W Breakpoint Setting] （S/W 断点设置）窗口，设置和解除软件断点。也能使软件断点暂时无效或者有效。

能指定 64 点的软件断点。如果指定多个软件断点，就在到达某个断点时暂停执行。

1.2.1.1 软件断点的设置和解除

在以下窗口设置和解除软件断点：

- [Editor(Source)] （编辑器 （源））窗口
- [S/W Breakpoint Setting] （S/W 断点设置）窗口

在编辑器 （源）窗口，能通过双击来设置和解除软件断点。

在 S/W 断点设置窗口，除了能设置和解除软件断点以外，还能切换软件断点的暂时无效或者有效。

1.2.1.2 能设置软件断点的区域

能设置为软件断点的区域因产品而不同。

能设置的区域请参照 “12.1.2 能设置软件断点的区域”。

1.2.2 硬件暂停功能

此功能在检测到存储器的读写数据时或者检测到执行指令时，停止执行目标程序。事件暂停有以下 2 种：

- 指令的暂停执行
  指令的暂停执行是指在即将执行指定地址的指令前停止执行目标程序。
- 存储器的暂停存取
  存储器的暂停存取是指在检测到指定地址的数据存取后，停止执行目标程序 （执行后暂停）。
  数据比较暂停只能指定 1 点。

能指定的事件数为 8 点，跟踪事件和时间测量事件共用。在执行指令时，能指定函数名。在存取存储器时，作为地址的指定方法，能指定地址和地址范围，而且能指定和读写指定地址的数据进行比较的比较数据，也能对比较数据进行屏蔽。能指定比较数据的事件只能指定 1 点的地址范围。

指定的事件暂停能进行以下的组合：

- 某个事件成立 （[OR] （或）条件）。
- 按指定的顺序事件成立 （[Sequential] （顺序）条件）。

1.3 实时跟踪功能

这是记录目标程序的执行履历的功能。

能记录 8M 周期的执行履历。能参照各周期的总线信息、执行的指令、源程序的执行路径。

在 [Trace] （跟踪）窗口参照执行履历。

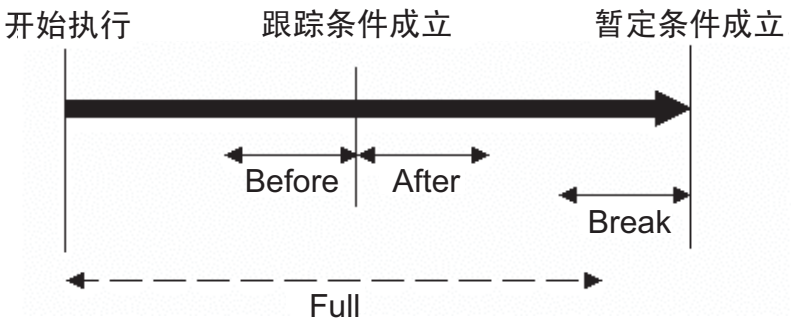
作为执行履历的参照方法，支持以下的显示模式：

- “Bus mode”（总线模式）
  能参照各周期的总线信息。显示的内容取决于所使用的MCU和仿真器系统。除了总线信息以外，能混合显示反汇编信息、源行信息和数据存取信息。
- “Disassemble mode”（汇编模式）
  能参照被执行的指令。除了反汇编信息以外，能混合显示源行信息和数据存取信息。
- “Data access mode”（数据存取模式）
  能参照数据的R/W周期。除了数据存取信息以外，能混合显示源行信息。
- “Source mode”（源模式）
  能在源程序上参照程序的执行路径。

1.3.1 跟踪范围

本调试器能参照 8M 周期的执行履历，跟踪范围支持以下 4 种模式：

- “Break”（暂停）
  目标程序停止前的8M周期
- “Before”（之前）
  经过跟踪点前的8M周期
- “After”（之后）
  经过跟踪点后的8M周期
- “Full”（完整）
  开始执行目标程序后的8M周期



在跟踪测量时，必须设置跟踪范围和跟踪事件。

跟踪范围的默认值设置为 “Break”（暂停）；跟踪事件的默认值设置为以全地址范围为跟踪测量对象的记录转移地址信息的事件。

1.3.2 跟踪条件的设置

本调试器能对跟踪事件进行以下的指定：

- 转移跟踪的指定
- 数据跟踪的指定
- 条件转移跟踪的指定

能进行转移跟踪或者数据跟踪的混和跟踪。
能指定的事件数为 8 点，暂停事件和时间测量事件共用。
各跟踪事件能进行以下的组合：

- 某个事件成立 （[OR] （或）条件）。

1.4 时间测量功能

执行时间的测量
测量全部执行时间。

间隔时间的测量
这是测量指定间隔的最大执行时间、最小执行时间、平均执行时间和执行次数的功能。
本调试器最多能测量 3 点间隔的时间。

1.4.1 测量范围

执行时间的测量
测量程序开始执行到停止的执行时间。

间隔时间的测量
间隔时间的测量条件能进行以下的指定：

- 开始事件成立到结束事件成立的时间

1.5 实时 OS 的调试功能

此功能调试目标程序 （使用实时 OS）的实时 OS 依存部分。
能参照实时 OS 的状态。

1.6 GUI 的输入 / 输出功能

这是在窗口中模拟用户目标系统的键输入面板 （按钮）或者输出面板的功能。
能在输入面板上使用按钮，在输出面板上使用标签 （字符串）和 LED。

2. 仿真器 E30A

仿真器 E30A 是使用 R32C/100 系列 MCU 的内部调试电路 NSD （New Single-wire Debugger）的 on-chip 调试仿真器。

NSD 是指瑞萨独创的 OCD （On Chip Debugger）。

只将目标 MCU 的 BYTE 引脚用作通信线，能以 MCU 的运行和电特性等接近用户最终产品的状态进行调试。

2.1 通信方式

仿真器 E30A 支持的通信方式是 USB 通信。

2.2 功能表

仿真器 E30A 支持的功能如下：

功能	仿真器 E30A
S/W 断点	64 点
H/W 断点	8 点 （跟踪事件和时间测量事件共用）
实时跟踪	8M 周期
RAM 监控	1K 字节 （16 字节 ×64 块）的区域
执行时间的测量	执行时间的测量 （1 点）、间隔时间的测量 （3 点）

3. 启动调试器前

3.1 仿真器的通信方式

仿真器 E30A 系统支持的通信方式如下：

- USB

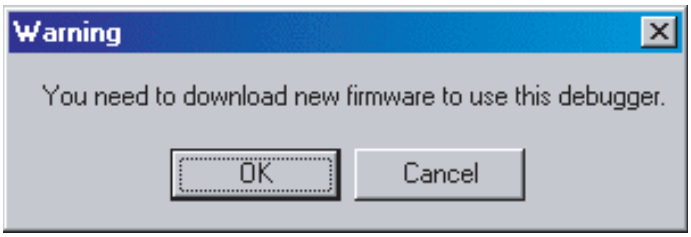
3.1.1 USB 通信

- 符合USB 2.0规格。
- 不支持通过USB集线器的连接。
- 仿真器附带要使用的电缆。

3.2 固件的下载

本产品在启动调试器时检查被下载到仿真器的固件版本。如果下载到仿真器的固件是旧版本，就进入固件下载模式。

如果在仿真器已进入强制下载固件模式的状态下启动调试器，就在启动时打开以下的对话框。请单击 [OK]（确定）按钮，下载固件。



3.3 启动仿真器前的设置

3.3.1 USB 通信

通过 Windows 的即插即用功能检测 USB 设备的连接，自动安装对应的设备驱动程序。

3.3.1.1 USB 设备驱动程序的安装

通过 Windows 的即插即用功能检测 USB 设备，如果检测到 USB 设备，就启动设备驱动程序的安装向导。请按以下的步骤安装 USB 设备驱动程序：

1. 用USB电缆将仿真器和主机连接。
2. 接通仿真器的电源。
3. 如果显示 “新硬件的追加向导”时，请选择 “自动安装软件 （推荐）”。

注意事项

- 在安装 USB 设备驱动程序前，需要先安装要使用的仿真调试器。请先安装仿真调试器。
- 必须由持有 Administrator 权限的用户安装 USB 设备驱动程序。

4. 调试的准备

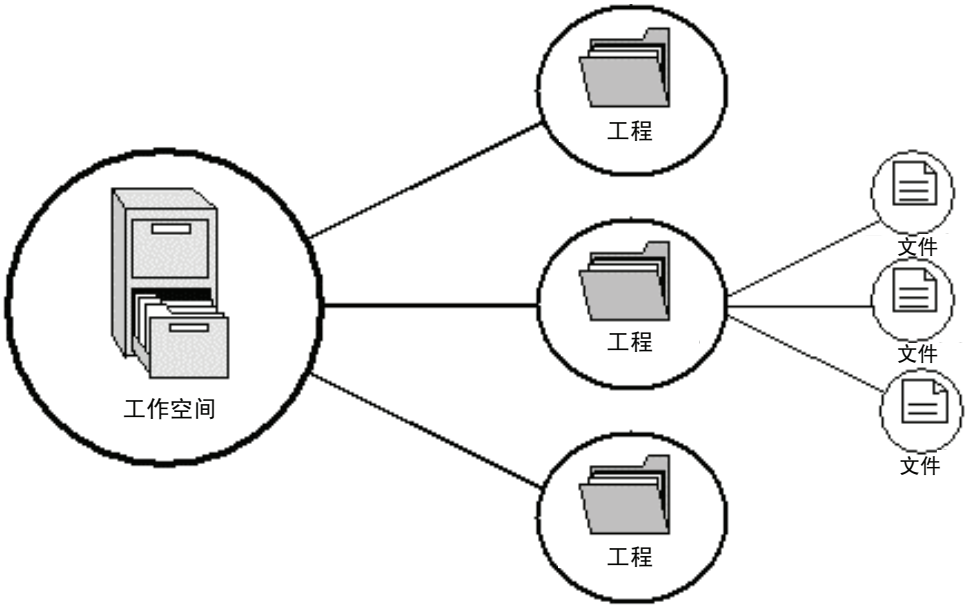
启动本产品并连接仿真器，开始调试。

在通过本产品进行调试时，需要创建工作空间。

4.1 工作空间、工程和文件

如同能通过字处理器创建和修改文档一样，能通过本产品创建和修改工作空间。

工作空间可视为工程的储存箱，工程可视为工程文件的存储箱。因此，各工作空间可包含一个或者多个工程，而每个工程可包含一个或者多个文件。



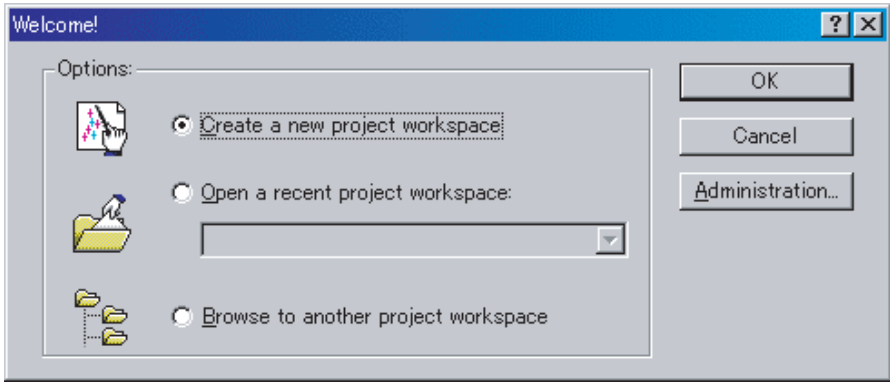
工作空间能将相关的工程汇集成一个。例如，在必须构建一个对应不同处理器的应用程序时，或者在应用程序和程序库要同时开发时，就很方便。能在工作空间内将工程分层连接，也就是说，当构建一个工程时，首先构建该工程的子工程。

为了充分利用工作空间，用户必须首先将工程添加到工作空间，然后将文件添加到该工程。

4.2 High-performance Embedded Workshop 的启动

请从 [Start]（中文系统：开始）菜单的 [Programs]（中文系统：程序）启动 High-performance Embedded Workshop。

显示 [Welcome!]（欢迎！）对话框。



在此对话框中创建和显示工作空间。

- [Create a new project workspace]（创建新的工程工作空间）单选按钮
  在创建新的工作空间时选择。
- [Open a recent project workspace]（打开现有工程工作空间）单选按钮
  在使用现有工作空间时选择。
  显示被打开的工作空间的履历。
- [Browse to another project workspace]（浏览其他工程工作空间）单选按钮
  在使用现有工作空间时选择。
  用于被打开的工作空间没有履历的情况。

要指定现有工作空间时，请选择 [Open a recent project workspace]（打开现有工程工作空间）或者 [Browse to another project workspace]（浏览其他工程工作空间）单选按钮，指定工作空间文件（扩展名 .hws）。

新工作空间的创建方法参照如下：

请参照 “4.2.1 创建新的工作空间（使用工具链）”。

请参照 “4.2.2 创建新的工作空间（不使用工具链）”。

※在通过本产品调试现有加载模块文件时，用此方法创建工作空间。

新的工程工作空间的创建步骤取决于是否使用工具链。本产品不包含工具链，工具链能在安装了对应所用 CPU 的 C/C++ 编译程序包的环境下使用。

有关创建已使用工具链的新工程工作空间的详细内容，请参照 C/C++ 编译程序包附属的手册。

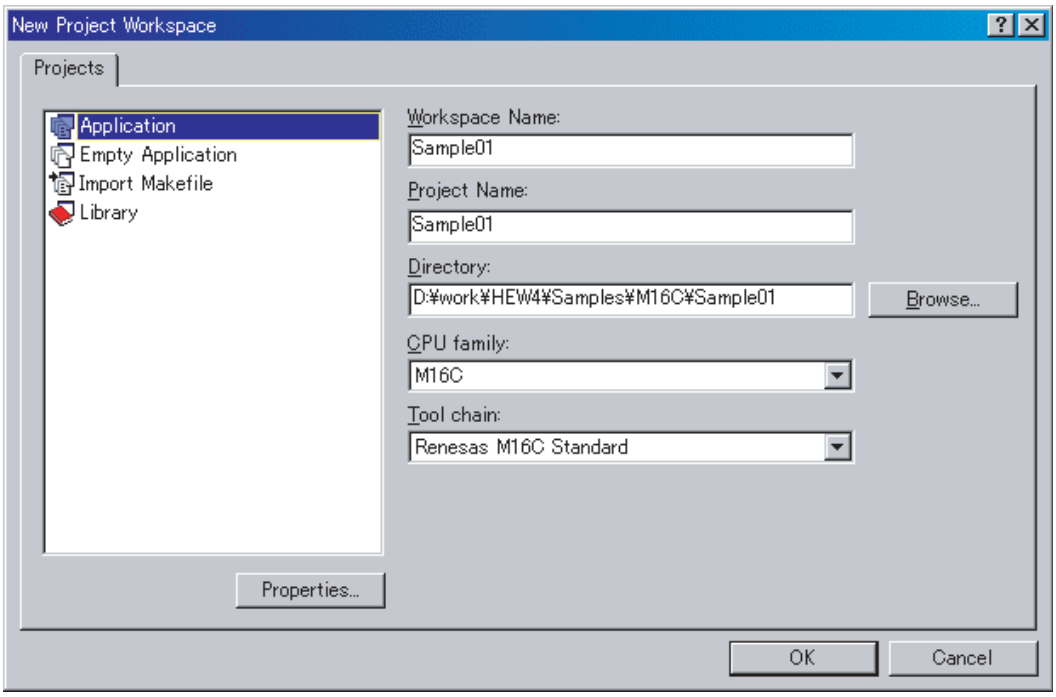
4.2.1 创建新的工作空间 （使用工具链）

4.2.1.1 Step1 ： 新工程工作空间的设置

请在启动 High-performance Embedded Workshop 时显示的 [Welcome!] （欢迎！）对话框中，选择 [Create a new project workspace] （创建新的工程工作空间）单选按钮，单击 [OK] （确定）按钮。

开始创建新工程工作空间。

打开以下的画面。



1. 选择CPU族

   请在[CPU family] （CPU族）下拉式列表框中选择要使用的CPU族。

2. 选择工具链

   请在[Tool chain] （工具链）下拉式列表框中选择相应的工具链名。

3. 选择工程种类

   在左侧的[Project type] （工程种类）列表框中，选择要使用的工程种类。

   在此，请选择 “Application”（应用程序）。

   （有关能选择的工程种类的详细内容，请参照C/C++编译程序包附属的手册。）

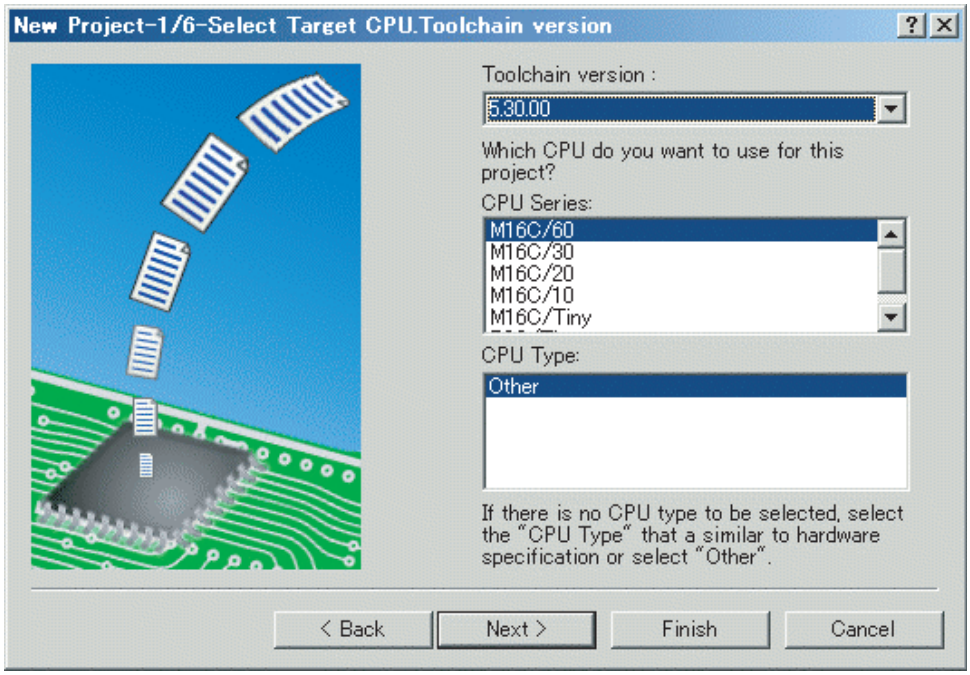
4. 指定工作空间名和工程名

   - 请在[Workspace Name] （工作空间名）编辑框中输入要创建新的的工作空间名。
   - 请在[Project Name] （工程名）编辑框中输入工程名。如果和工作空间同名，就不需要输入。
   - 请在[Directory] （目录）编辑框中输入要创建工作空间的目录。也能单击[Browse...] （浏览...）按钮，选择要创建工作空间的目录。

请在输入后单击 [OK] （确定）按钮。

4.2.1.2 Step2 ：工具链的设置

启动工程创建向导。



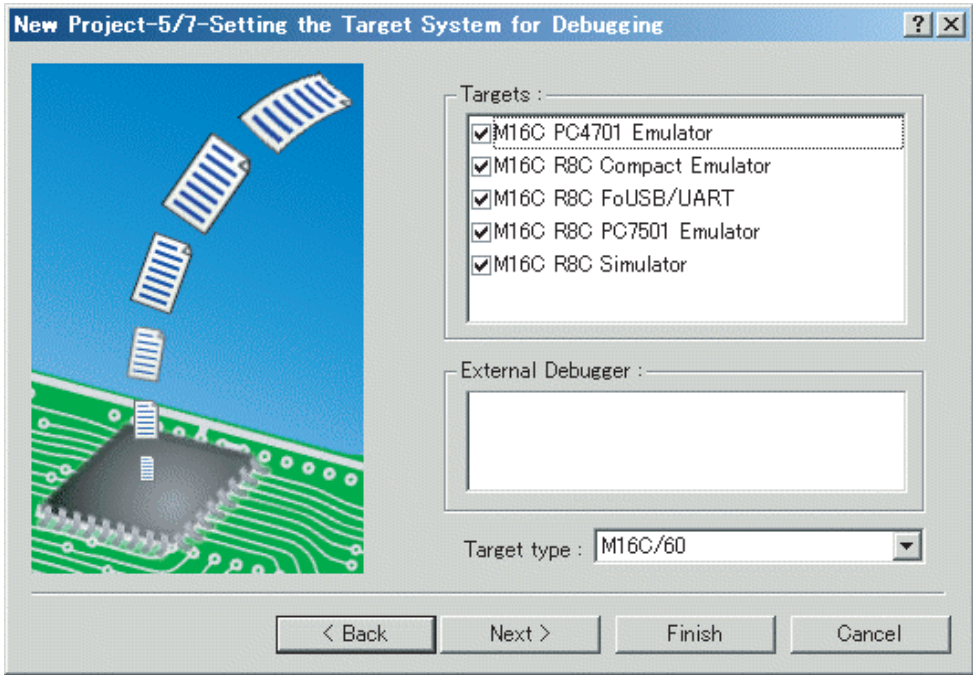
在向导的最初进行以下的设置：

- 工具链的设置
- 有关实时OS的设置 （使用的情况）
- 生成文件、堆区和堆栈区等的设置

请输入必要的信息，然后单击 [next] （下一步）按钮。
设置的内容因所使用的 C/C++ 编译程序包而不同，请参照 C/C++ 编译程序包附属的手册。

4.2.1.3 Step3：目标平台的选择

在向导的最后设置要使用的目标 （仿真器、模拟器）。
在设置完工具链后，显示以下的画面。

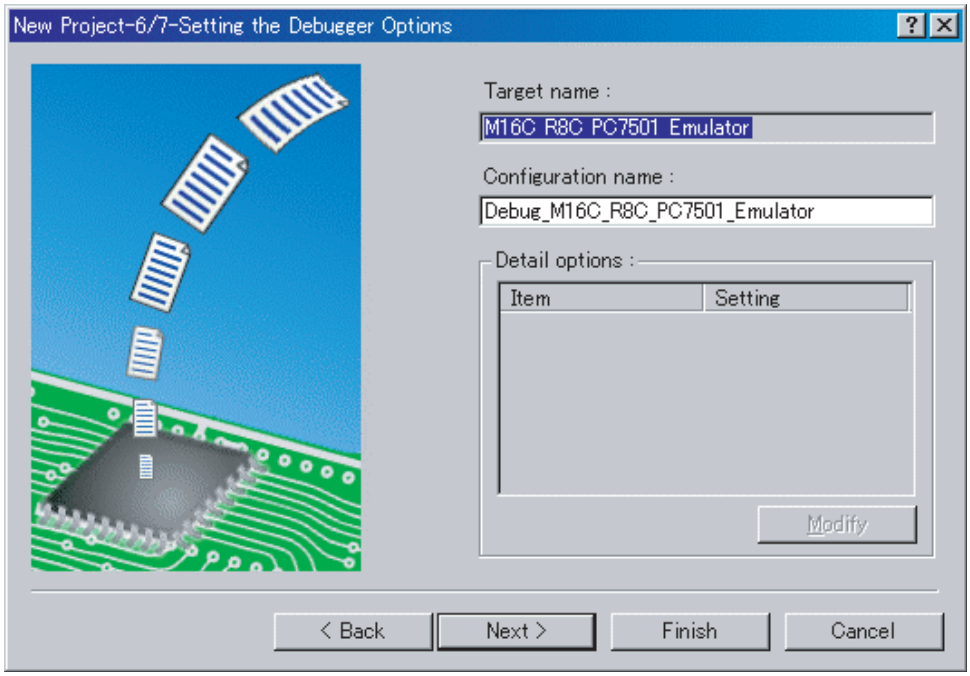


1. 目标种类的选择
   请在[Target type] （目标类型）下拉式列表框中选择所用目标的CPU种类。
2. 目标平台的选择
   在[Targets] （目标）栏显示能使用的目标。
   请选择要使用的目标 （可指定多个）。

请在输入后单击 [next] （下一步）按钮。

4.2.1.4 Step4 ：配置文件名的设置

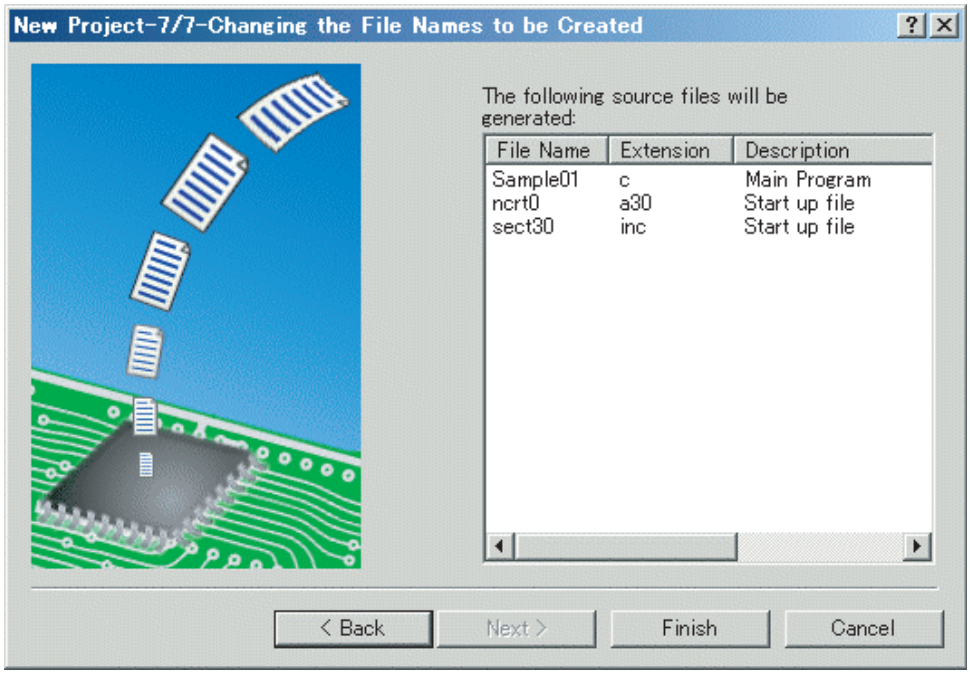
按所选的目标，设置配置文件名。
配置文件是指目标除外的保存 High-performance Embedded Workshop 状态的文件。



已设置了默认的文件名，如果不需要更改，就请单击 [next]（下一步）按钮。

4.2.1.5 Step5：生成文件的确认

根据现在为止的设置，显示本产品生成的文件。如果要更改文件名，请在选择文件名并单击后输入文件名。



在此，结束有关仿真器的设置。
请按照画面的指示结束工程创建向导。

4.2.2 创建新的工作空间 （不使用工具链）

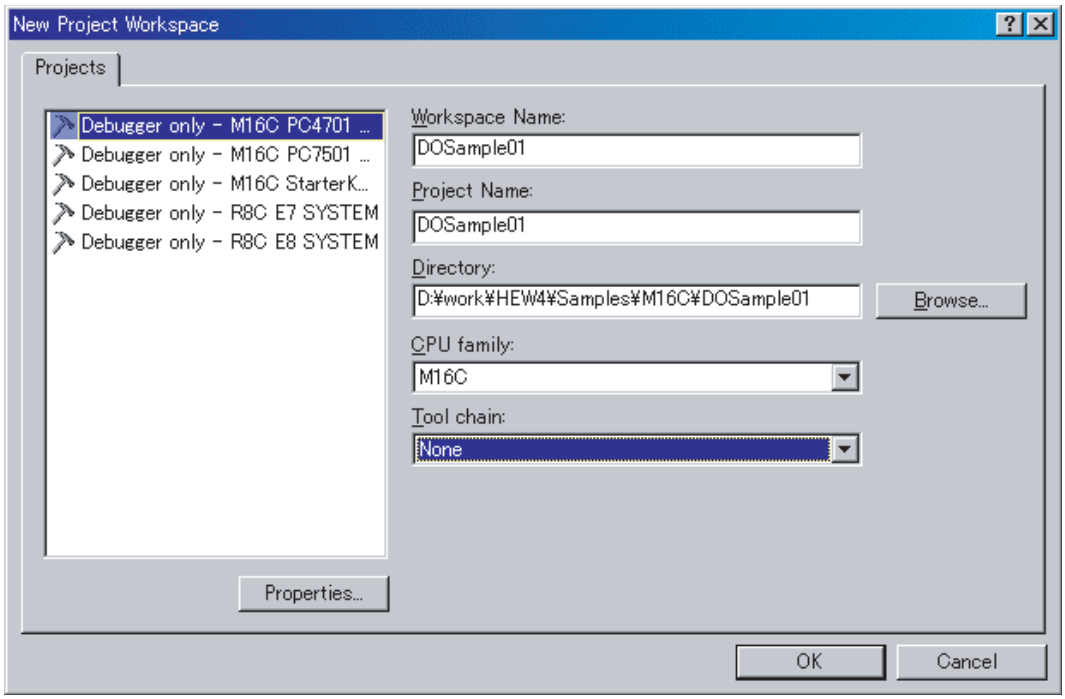
在通过本产品调试现有的加载模块文件时，用此方法创建工作空间。（即使没有安装工具链也可以。）

4.2.2.1 Step1：新工程工作空间的设置

请在启动 High-performance Embedded Workshop 时显示的 [Welcome!] （欢迎！）对话框中，选择 [Create a new project workspace] （创建新的工程工作空间）单选按钮，然后单击 [OK] （确定）按钮。

开始创建新工程工作空间。

打开以下的画面。

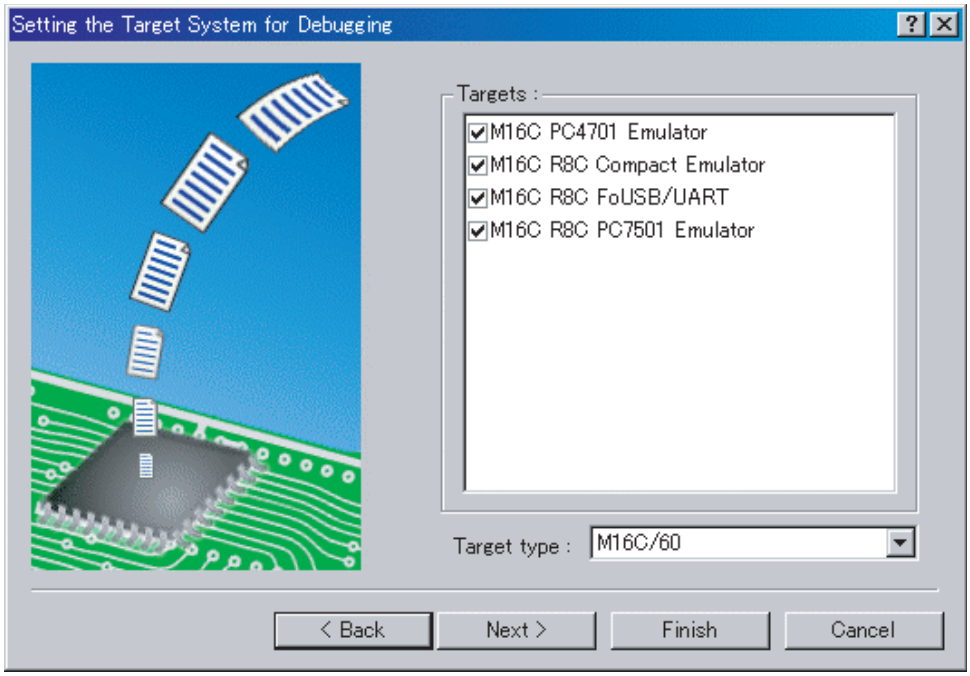


1. 选择CPU族
   请在[CPU family] （CPU族）下拉式列表框中选择要使用的CPU族。
2. 选择工具链
   因不使用工具链，所以请在[Tool chain] （工具链）下拉式列表框中指定 “None”（无）。
   （如果工具链没有被注册，就不能选择此下拉式列表 （不需要选择）。）
3. 选择工程种类
   在不使用工具链时，左侧的[Project type] （工程种类）列表框中显示 “Debugger only-目标名"，请选择此内容 （当显示多个目标时，请选择1个要使用的工程种类）。
4. 指定工作空间名和工程名
   - 请在[Workspace Name] （工作空间名）编辑框中输入要创建新的工作空间名。
   - 请在[Project Name] （工程名）编辑框中输入工程名。如果和工作空间同名，就不需要输入。
   - 请在[Directory] （目录）编辑框中输入创建工作空间的目录。也能单击[Browse...] （浏览...）按钮，选择创建工作空间的目录。

请在输入后单击 [OK] （确定）按钮。

4.2.2.2 Step2：目标平台的选择

设置要使用的目标 （仿真器、模拟器）。
启动工程创建向导，显示以下的画面。



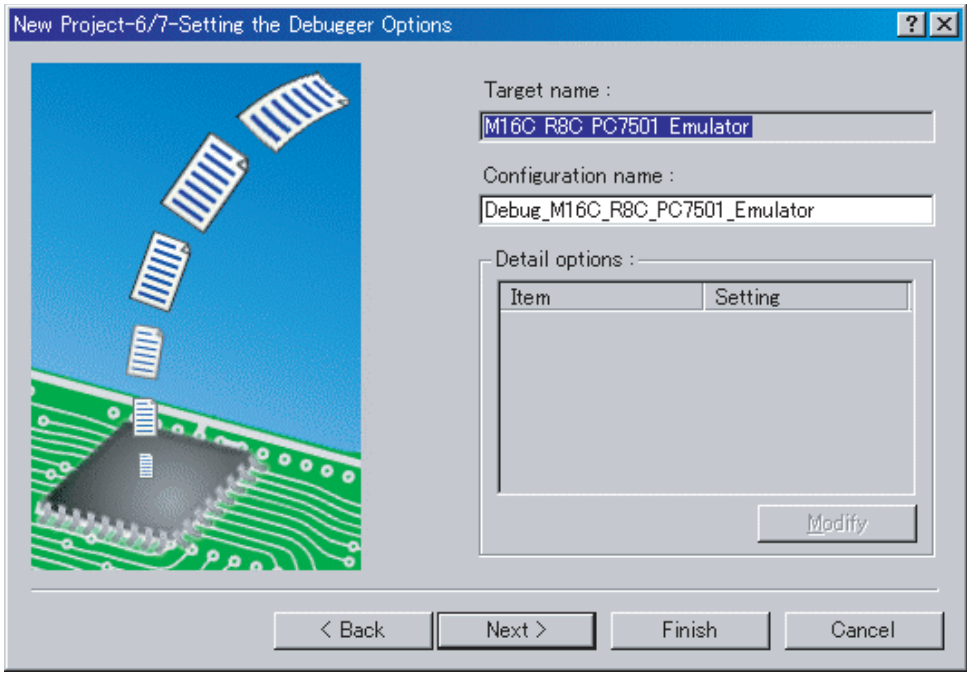
1. 目标种类的选择
   请在[Target type] （目标类型）下拉式列表框中选择所用目标的CPU种类。
2. 目标平台的选择
   在[Targets] （目标）栏显示能使用的目标。
   请选择要使用的目标 （可指定多个）。

请在输入后单击 [Next] （下一步）按钮。

4.2.2.3 Step3：配置文件名的设置

按所选的目标，设置配置文件名。

配置文件是指目标除外的保存 High-performance Embedded Workshop 状态的文件。



已设置了默认的文件名，如果不需要更改，请单击 [next]（下一步）按钮。

在此，结束有关仿真器的设置。

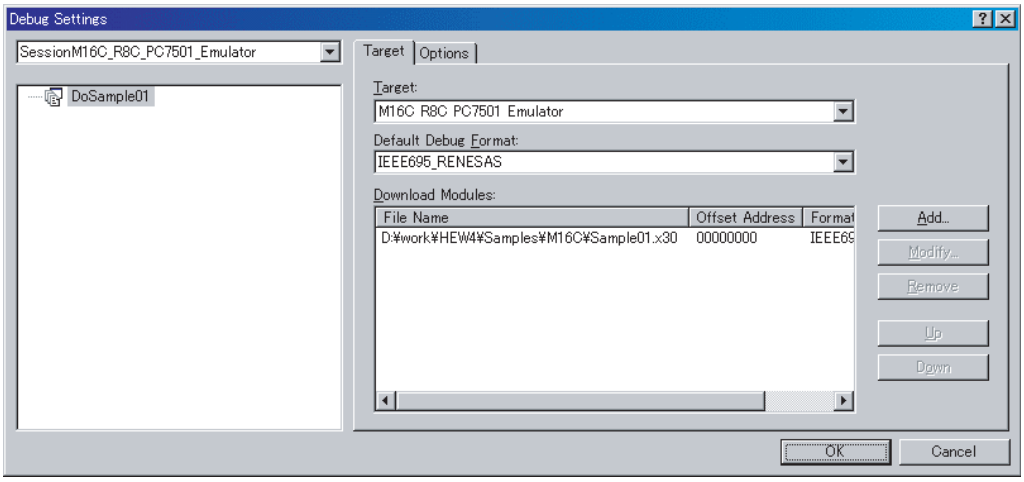
请按照画面的指示结束工程创建向导。

在启动 High-performance Embedded Workshop 的同时，显示开始设置调试器的对话框。如果仿真器的准备已经结束，请设置调试器并连接仿真器。

4.2.2.4 Step4：下载模块的注册

最后，注册要使用的加载模块文件。

请从 [Debug] （调试）菜单中选择 [Debug Settings...] （调试设置 ...），在打开的对话框中进行以下的设置：

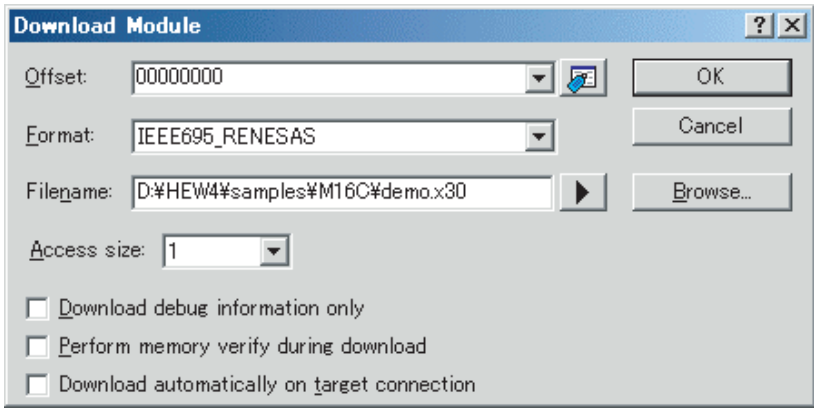


1. 请在[Target] （目标）下拉式列表框中选择要连接的产品名。
2. 请在[Default Debug Format] （默认对象格式）下拉式列表框中选择要下载的加载模块格式。

格式名	种类
IEEE695_RENESAS	IEEE-695 格式文件 （由本公司产的工具链生成）
IEEE695_IAR	IEEE-695 格式文件 （由 IAR 公司产的工具链生成）
IEEE695_TASKING	IEEE-695 格式文件 （由 TASKING 公司产的工具链生成）
ELF/DWARF2	ELF/DWARF2 格式文件 （由本公司产的工具链生成）
ELF/DWARF2_IAR	ELF/DWARF2 格式文件 （由 IAR 公司产的工具链生成）
ELF/DWARF2_TASKING	ELF/DWARF2 格式文件 （由 TASKING 公司产的工具链生成）
ELF/DWARF2_KPIT	ELF/DWARF2 格式文件 （由 KPIT 公司产的工具链生成）

不对应下拉式列表中没有显示的目标格式。

3. 请在[Download Modules]（下载模块）列表框中注册下载模块。
能在用[Add...]（添加...）按钮打开的以下对话框中指定下载模块。



- 请在[Format]（格式）编辑框中指定下载模块的格式，格式名请参照上述的一览表。
- 请在[Filename]（文件名）编辑框中输入下载模块的全路径名和文件名。
- 如果在连接目标后立即下载程序，请选定[Download automatically on target connection]（在连接目标时下载）。

请在设置结束后单击[OK]（确定）按钮。

注意事项

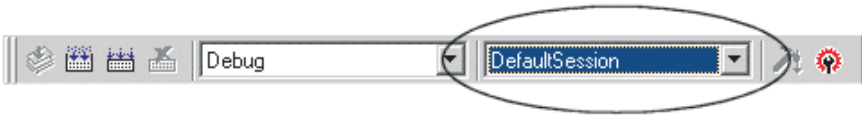
不支持[Offset]（偏移量）、[Access size]（存取长度）和[Perform memory verify during download]（下载时的存储器验证），所以[Offset]（偏移量）为“0”，[Access size]（存取长度）为“1”，[memory verify]（下载时的存储器验证）为“无”。

4.3 调试器的启动

能通过连接仿真器开始调试。

4.3.1 仿真器的连接

只要事先将要用的仿真器设置切换到已被注册的会话文件，就能简单地连接仿真器。
如果在创建工程时选择目标，就以默认值创建所选目标的会话文件。
请从以下工具栏的下拉式列表中选择与连接的目标对应的会话文件。



显示设置调试器的对话框。
如果没有显示此对话框，就从[Debug]（调试）菜单中选择[Connect]（连接）。

结束此设置，连接完成。

4.3.2 仿真器的结束

有以下的方法：

1. 选择 “Disconnect （断开连接）”。
   请从[Debug] （调试）菜单中选择[Disconnect] （断开连接）。

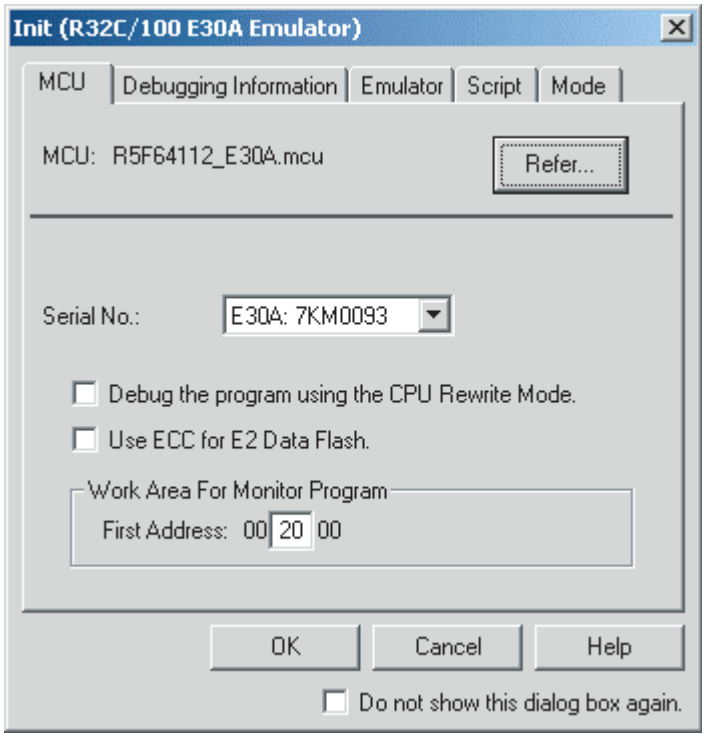
2. 将会话切换为 “DefaultSession”（默认会话）
   请在连接仿真器时使用的下拉式表中选择 “DefaultSession”（默认会话）。
3. 结束High-performance Embedded Workshop
   请从[File] （文件）菜单中选择[Exit] （退出），结束High-performance Embedded Workshop。

切换会话，并且在结束 High-performance Embedded Workshop 前显示确认保存会话的信息框。当需要保存会话时，请单击 [Yes] （是）按钮；当不需要时，请单击 [No] （否）按钮。

5. 调试器的设置

5.1 [Init] （初始化）对话框

在启动调试器时，通过 [Init] （初始化）对话框设置需要设置的项目。
在此对话框中设置的内容在下次启动时也有效。



支持的选项卡因产品而不同。

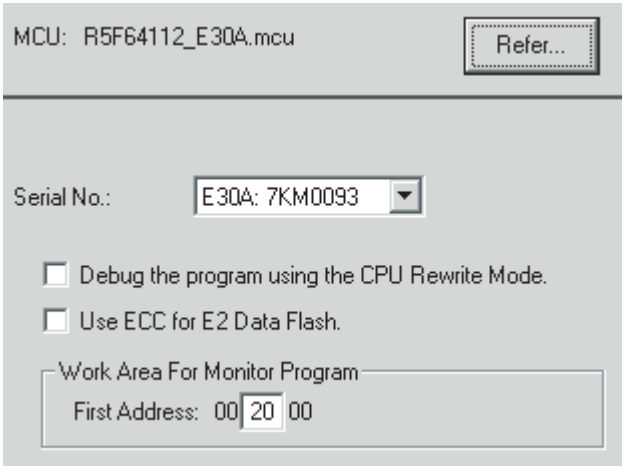
选项卡名	用于 R32C 的调试器
MCU	○
Debugging Information （调试信息）	○
Emulator （仿真器）	○
Script （脚本）	○
Debugging Mode （调试模式）	○

能通过以下的任意一个方法重新显示 Init 对话框：

- 在启动调试器后，选择菜单 [Setup] （设置） →[Emulator] （仿真器） →[System...] （系统...）。
- 边按Ctrl键边切换调试会话。

5.1.1 MCU 选项卡

指定的内容在下次启动时也有效。



5.1.1.1 MCU 文件的指定

请单击 [Refer] （参照）按钮。
打开文件选择对话框，请指定相应的 MCU 文件。

- MCU文件是保存目标MCU的固有信息的文件。
- 在MCU选项卡的MCU栏显示所指定的MCU文件。

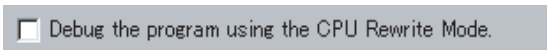
5.1.1.2 通信接口的指定

请选择要使用的接口。
能使用的通信接口因仿真器而不同。
各通信接口的设置如下所示：

- 请参照 “5.2.1 USB通信的设置”。

5.1.1.3 CPU 改写模式的使用或者不使用

指定是否调试 “CPU rewrite mode”（CPU 改写模式）。



当调试所用 “CPU rewrite mode”（CPU 改写模式）的目标系统时，请选定上述复选框。
此指定只能在启动调试器时设置或者更改。

补充事项

如果 CPU 改写模式调试有效，就能使用以下的功能：

- 给内部ROM区域设置S/W断点。

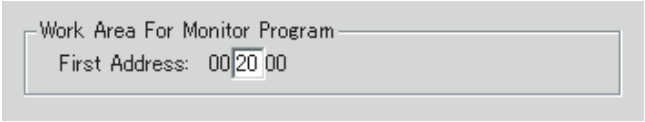
5.1.1.4 E2 数据闪存的 ECC 使用或者不使用

指定在 E2 数据闪存中是否使用 ECC。

在 E2 数据闪存中使用 ECC 时，请选定上述复选框。
此指定只能在启动调试器时设置或者更改。
对于不支持 E2 数据闪存的 MCU，此设置无效。

5.1.1.5 调试监视程序的起始地址的指定

指定调试监视程序使用的区域的起始地址 （MCU 的内部 RAM 区域）。

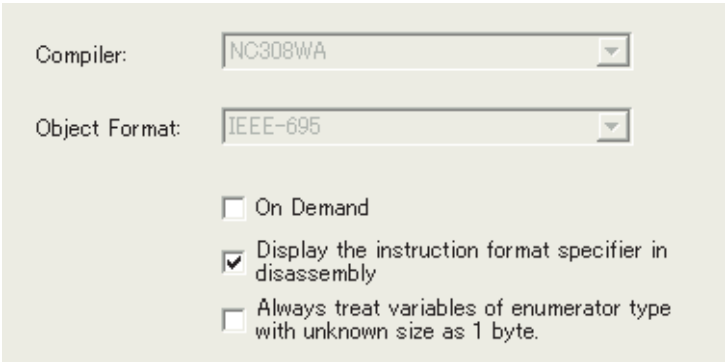


调试监视程序从指定的起始地址开始使用 1.2K 字节的区域。
因为事先保存存储器的内容，所以不需要特别在意，但是需要注意以下几点：

- 起始地址只能设置地址 00xx00 （xx 为任意数值），默认值为地址 002000。
- 指定的区域不能是与堆栈重复的区域以及 DMA 的对象区域。

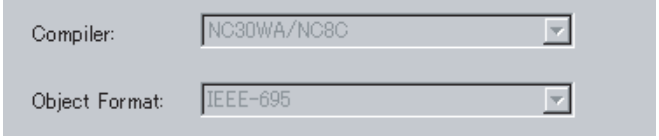
5.1.2 [Debugging Information] （调试信息）选项卡

指定的内容在下次启动时也有效。



5.1.2.1 使用的编译程序和目标格式的参照

显示所使用的编译程序和目标文件的格式。



能在此对话框中确认现在的设置内容。请在通过菜单 [Debug] （调试） →[Debug Settings...] （调试设置 ...）打开的对话框中进行设置。

5.1.2.2 调试信息的保存方式的指定

调试信息的保存方式有 [On Memory] （记忆方式）和按需方式。
请选择调试信息的保存方式 （默认值：记忆方式）。
当选择按需方式时，选定 [On Demand] （只在需要时读调试信息）复选框。
指定的内容在下次下载时有效。

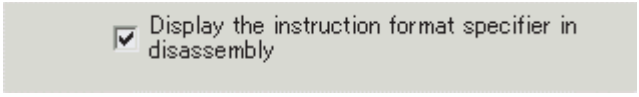
- 记忆方式
  将调试信息保存在个人计算机的存储器中。
  通常选择此方式。
- 按需方式
  将调试信息保存在能再利用的暂存文件中。
  在对同一个加载模块进行多次下载时，因为保存的调试信息被再利用，所以能高速进行下载。
  对于加载模块的规模，在因PC内置的存储器容量较小而花费调试信息的下载时间的情况下，适合使用此方式。

注意事项

- 当加载模块的规模较大时，记忆方式可能在下载处理时需要大量的时间。此时，请选择按需方式。
- 按需方式在存放被下载的加载模块的位置 （文件夹），创建能再利用的暂存文件的保存文件夹。文件夹名是“~INDEX_”＋加载模块名。例如，加载模块名为 “sample.abs”，文件夹名就为 “~INDEX_sample”。此文件夹即使在结束调试器后也不被删除。

5.1.2.3 指令格式指定符的显示或者隐藏

指定在显示反汇编中是否显示指令格式指定符。



要显示指令格式指定符时，请选定上述复选框。
此指定只能在启动调试器时进行设置和更改。

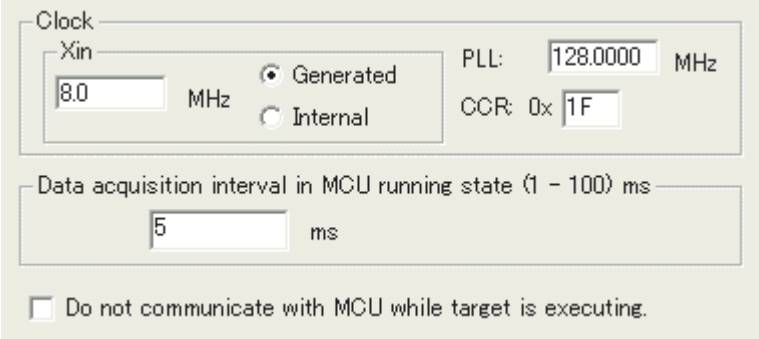
5.1.2.4 长度不明的枚举型的长度指定

当调试信息中存在没有长度信息的枚举型信息时，能指定是否总是以 1byte 处理此枚举型的长度。 NC30、NC308 和 NC100 有将枚举型的长度置为 1byte （而不是标准的 int 长度）的选项，以提高存储器的效率。另外，NC30、 NC308 和 NC100 不将枚举型的长度信息输出到调试信息中，所以调试器总是以 int 长度进行处理。因此，对于指定了枚举型为 1byte 选项的目标程序，有可能出现不能正确显示枚举型变量值的情况，而此指定能解决这种问题。

有关将枚举型的长度置为 1byte 的选项的详细内容，请参照各编译程序产品的手册。

如果总是要以 1byte 处理没有长度信息的枚举型的长度，请选定上述复选框。
为了使此设置有效，需要重新下载目标程序。

5.1.3 [Emulater] （仿真器）选项卡



5.1.3.1 目标时钟的指定

指定提供给 MCU （主时钟）的时钟。

必须根据目标单片机的使用时钟更改设置 （默认值：[Generated] （生成））。

当将用户设置的频率设置为仿真器内部生成的时钟 （用户定义时钟）时，选择 [Generated] （生成）；当将该频率设置为内部时钟时，选择 [Internal] （内部）。

无论是设置了 [user-defined clock] （用户定义时钟）还是设置了内部时钟，都必须在频率输入栏设置要使用的频率 （默认值：[no values are set] （什么也没有设置））。

- 调试器中用户定义时钟的可输入范围为1.0000MHz～99.9999MHz。能以0.0001MHz为单位进行设置。
- 频率输入栏的值只能在启动调试器时进行设置和更改。
- 如果在频率输入栏中没有设置值，就无法启动调试器。

5.1.3.2 PLL 电路的设置

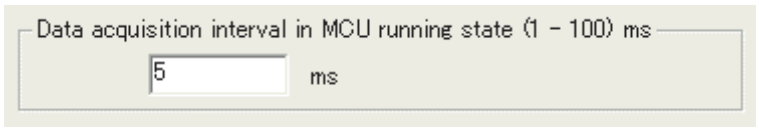
指定 PLL 电路的设置。

请输入 PLL 电路的频率和 CCR 寄存器的值。

必须用 0x00 ～ 0xFF 的 8 位值指定 CCR 寄存器的值。

5.1.3.3 目标程序执行过程中存储器数据获取间隔的指定

必须指定在目标程序执行过程中通过 RAM 监控功能获取存储器数据时的间隔 （各字节 / 字数据的存储器数据的获取间隔：单位 [ms]），默认值为 5ms。在此指定的间隔和在 RAM 监控对应的各窗口设置的显示更新间隔 （采样周期）不一样。



5.1.3.4 目标程序执行过程中是否与 MCU 通信

必须设置在目标程序执行过程中是否与 MCU 通信。

当不与 MCU 通信时，请选定上述复选框。

如果在执行 [WAIT/STOP] （等待 / 停止）指令的过程中与 MCU 通信，就会发生通信错误，所以必须设置为不与 MCU 通信。

5.1.4 启动 [Script] （脚本）选项卡

指定的内容只在启动时有效，启动后通过 Init 对话框重新设置的内容无效。

5.1.4.1 脚本命令的自动执行

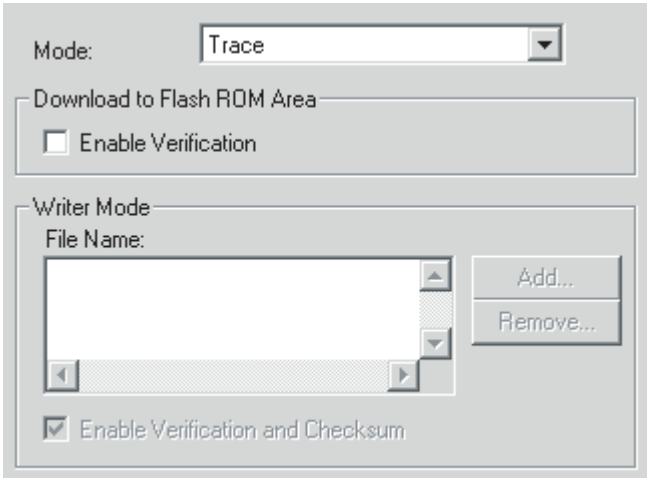
要在启动调试器时自动执行脚本命令时，请单击 [Refer...] （参照 ...）按钮，指定要执行的脚本文件。

通过单击 [Refer...] （参照 ...）按钮，打开文件选择对话框。

在 “Init File:”栏 （初始化文件：栏）中显示被指定的脚本文件。

当不要自动执行脚本命令时，请删除 “Init File:”栏 （初始化文件：栏）中显示的字符串。

5.1.5 运行模式选项卡



5.1.5.1 运行模式的选择

选择仿真器调试器的运行模式。

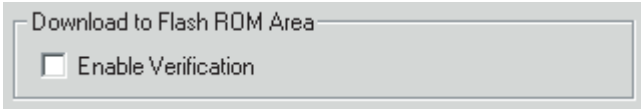
- 跟踪模式是用于使用跟踪功能的模式。
- 时间测量模式是用于使用时间测量功能的模式。
- RAM 监控模式是用于使用 RAM 监控功能的模式。
- 编程器模式是将仿真器用作闪存 ROM 编程器的模式。

不能使用所选模式以外的模式功能。
各模式能使用的调试功能如下所示：

模式	调试功能							
	暂停				跟踪	时间测量		RAM 监控
	执行 PC	数据存取	地址范围	数据比较		时间测量	间隔时间	
跟踪： 跟踪优先	○	○	×	×	○	×	×	×
跟踪： MCU 执行优先	○	○	×	○	○	×	×	×
时间测量	○	○	×	×	×	○	○	×
RAM 监控	○	○	○	×	×	×	×	○
编程器	×	×	×	×	×	×	×	×

5.1.5.2 下载到闪存 ROM 区域的指定

指定在下载目标程序时是否验证闪存 ROM 区域的内容。
要使用验证功能时，请选定 “Enable Verification （验证功能有效）” 复选框。



5.1.5.3 编程器模式的指定

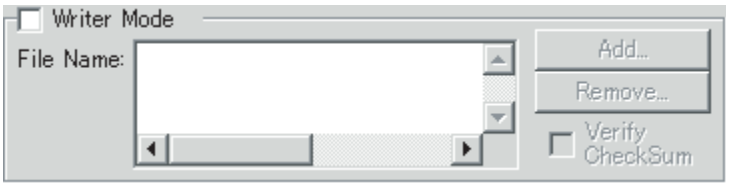
在要将仿真器用作闪存 ROM 编程器时指定编程器模式。在编程器模式中，不能调试目标程序。

要指定编程器模式时，请选择要下载的加载模块文件。

请在通过单击 [Add...]（添加 ...）按钮打开的文件选择对话框中选择文件。在 “file name:（文件名：）” 栏列表显示所选的文件，能选择多个文件。

要从列表显示删除文件时，请在选择要删除的文件后单击 [Remove...]（删除 ...）按钮。

当使用验证功能及校验和功能时，请选定 “Enable Verification and Checksum”（验证及校验和功能有效）复选框。



在编程结束后，请重新启动或者结束仿真调试器。

注意事项

- 编程器功能只能在选择了编程器模式中的运行模式时使用。

5.2 通信接口的设置

5.2.1 USB 通信的设置

USB 通信使用个人计算机的 USB 接口，符合 USB 2.0 规格。

要进行 USB 通信时，需要事先安装专用的设备驱动程序。有关 USB 设备驱动程序的安装，请参照“3.3.1.1 USB 设备驱动程序的安装”。

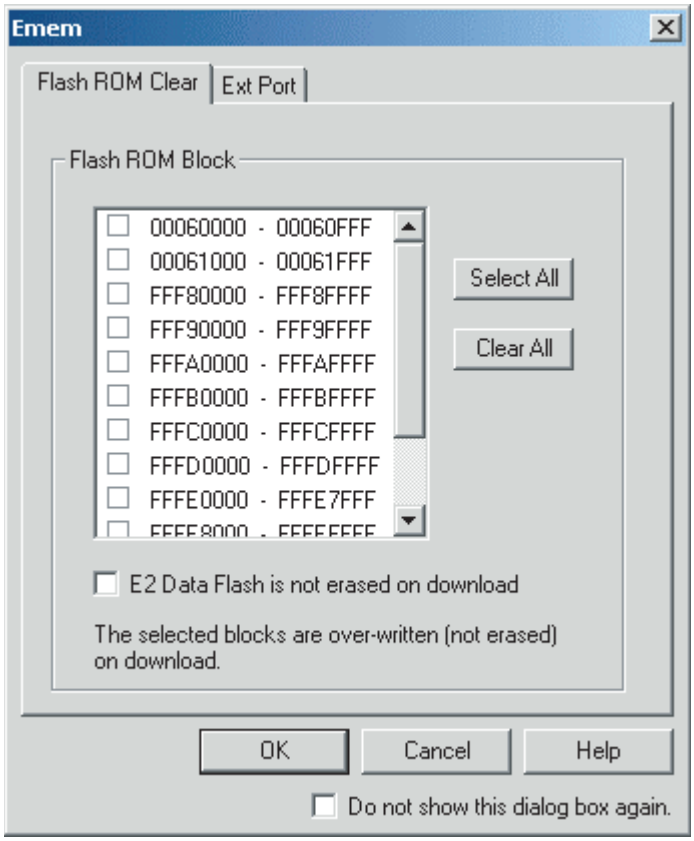
在 [Serial No.]（序列号）栏显示现在连接 USB 的仿真器一览表。

请选择被连接的仿真器的序列号。

5.3 调试器的设置 （用于 R32C）

5.3.1 Emem 对话框

Emem 对话框是用于设置用户目标信息的对话框，在关闭 [Init] （初始化）对话框后被打开。



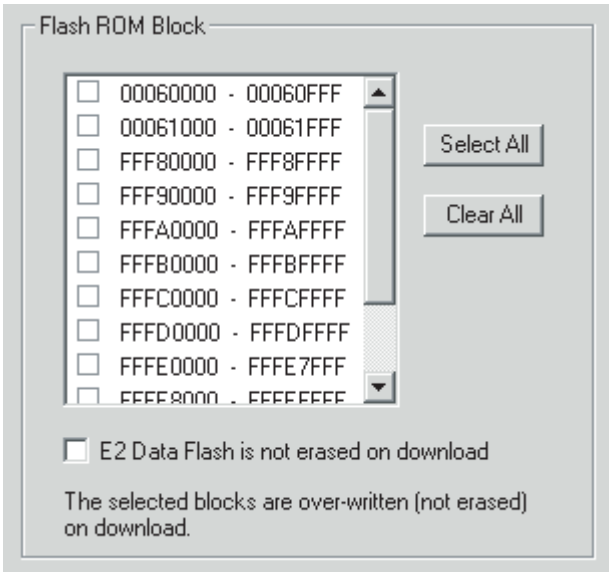
选项卡名	内容
Flash ROM Clear （闪存 ROM 的清除）	指定 MCU 内部闪存 ROM 的清除内容。
Ext Port （扩展端口）	指定是否将目标的复位置为无效。

能通过以下的方法重新显示 Emem 对话框：

- 在启动调试器后，选择菜单 [Setup] （设置） →[Emulator] （仿真器） →[Target...] （目标...）。

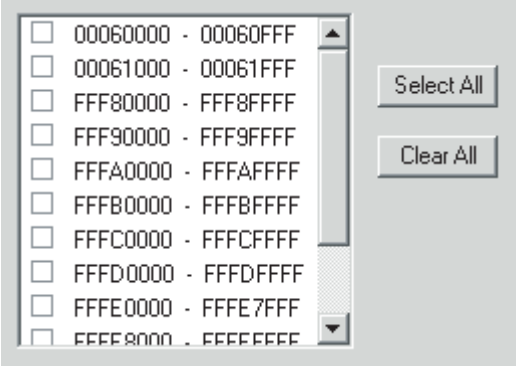
5.3.1.1 [Flash ROM Clear] （闪存 ROM 清除）选项卡

指定的内容在下次启动时也有效。



(1) MCU 内部闪存 ROM 的清除设置

指定在下载目标程序和数据时是否清除 （用 0xFF 填写） MCU 内部闪存 ROM 的内容。



在列表中以块为单位显示 MCU 的内部闪存 ROM。

- 带复选标记的块在下载时不清除闪存的内容。下载时不被盖写的存储器内容保持不变。
- 没有复选标记的块在下载时清除闪存的内容。
- 如果单击[Select All] （全选）按钮，全部的块就被设置复选标记 （下载时不清除全部的块）。
- 如果单击[Clear All] （全部清除）按钮，全部的块就被取消复选标记 （下载时清除全部的块）。

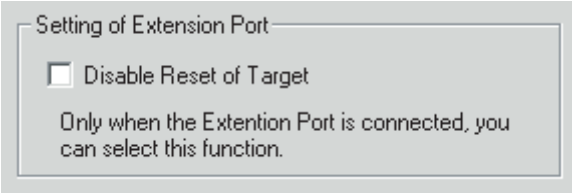
(2) E2 数据闪存的清除设置

指定在下载目标程序和数据时是否清除 （用 0xFF 填写） E2 数据闪存的内容。

如果在下载时不要清除 E2 数据闪存的内容，请选定上述复选框。
下载时不被盖写的存储器内容保持不变。
对于不支持 E2 数据闪存的 MCU，此设置无效。

5.3.1.2 [Ext Port]（扩展端口）选项卡

指定的内容在下次启动时也有效。



(1) 扩展端口的设置

指定是否将目标的复位置为无效。

- 要使目标的复位无效时，请选定上述复选框。在没有连接扩展端口时不能设置。
- 当已经连接扩展端口时，既使在启动调试器后也能更改设置。

参考篇

6. 教程

6.1 绪论

为了介绍本调试器的主要功能，我们提供了教程程序。在 High-performance Embedded Workshop 安装驱动器的 \WorkSpace\Tutorial 中按不同的目标和不同的 MCU 分别保存了教程程序。请从 [Open Workspace...]（打开工作空间 ...）打开与所用系统对应的教程工作空间文件 （*.hws）。

以下用此程序说明各种功能。

此教程程序是用 C 语言编写的，按照升降顺序将 10 个随机数据进行排序。

教程程序进行以下处理：

- 由 tutorial 函数生成要排序的随机数据。
- 由 sort 函数输入数组，用于保存 tutorial 函数生成的随机数据，并且按照升序进行排序。
- 由 change 函数输入 tutorial 函数生成的数组，并且按照降序进行排序。

注意事项

如果重新编译，就有可能和本章说明的地址不同。

6.2 使用方法

请按以下的步骤进行。

6.2.1 Step1：调试器的启动

6.2.1.1 调试的准备

启动 High-performance Embedded Workshop，并连接仿真器。

详细内容请参照 “4. 调试的准备”。

6.2.1.2 调试器的设置

一旦连接仿真器，就显示调试器的设置对话框。通过此对话框进行调试器的初始设置，详细内容请参照 “5. 调试器的设置”。

当结束调试器的设置时，就进入能调试的状态。

6.2.2 Step2：RAM 的运行检查

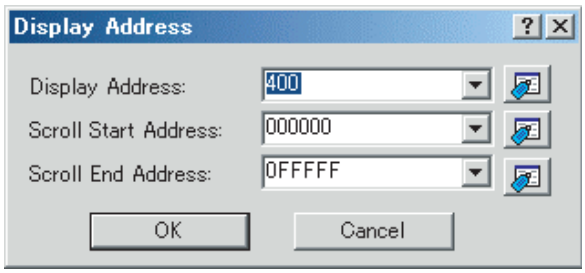
检查 RAM 是否正常运行。在 [Memory]（存储器）窗口中显示和编辑存储器的内容，确认存储器是否正常运行。

注意事项

有些单片机能在电路板上安装存储器。此时，只用上述的存储器运行检查可能不够完善，建议创建存储器的检查程序进行确认。

6.2.2.1 RAM 的运行检查

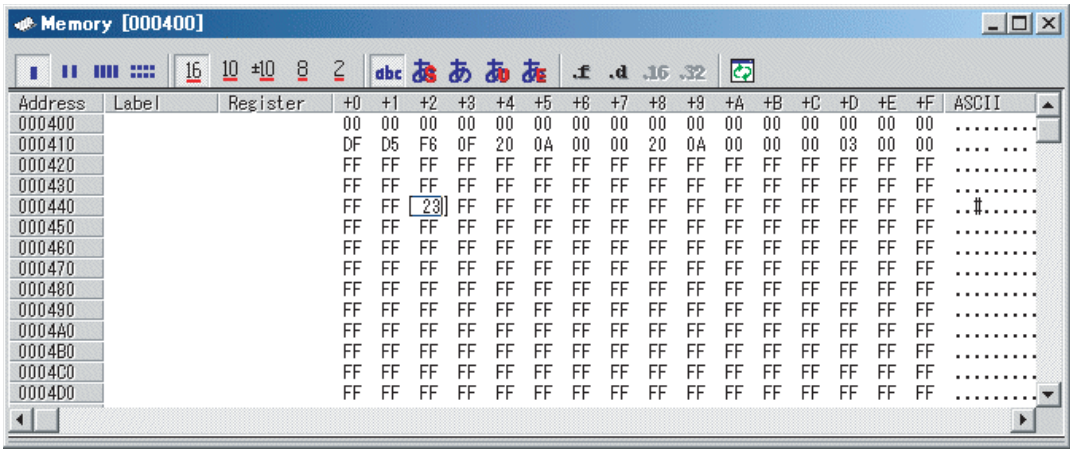
请从 [View] （视图）菜单的 [CPU] 子菜单中选择 [Memory] （存储器），在 [Display Address] （显示开始地址）编辑框中输入 RAM 地址 （在此，输入 “400”）。不改变 [Scroll Start Address] （滚动开始地址）和 [Scroll End Address] （滚动结束地址）编辑框的默认设置 （默认值：存储器的全部空间为滚动栏）。



注意事项

RAM 区域的设置因各产品而不同，请参照各产品的硬件手册。

请单击 [OK] （确定）按钮， [Memory] （存储器）窗口显示被指定的存储区域。



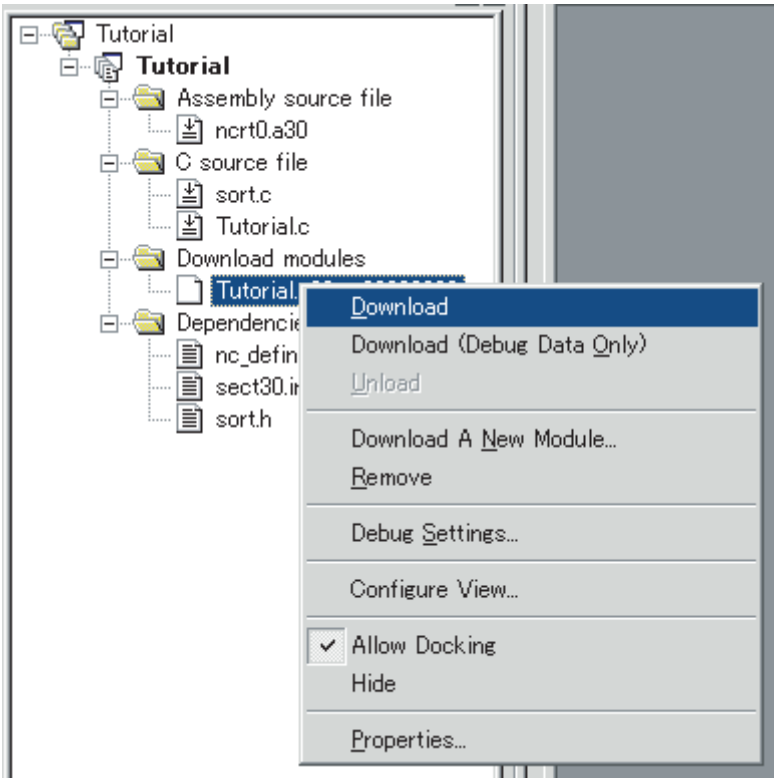
能通过双击 [Memory] （存储器）窗口中的数据部分更改其值。

6.2.3 Step3：教程程序的下载

6.2.3.1 教程程序的下载

下载要调试的目标程序。下载的程序和下载的地址因使用的单片机而不同，画面显示等方面因所用单片机而有一些不同。

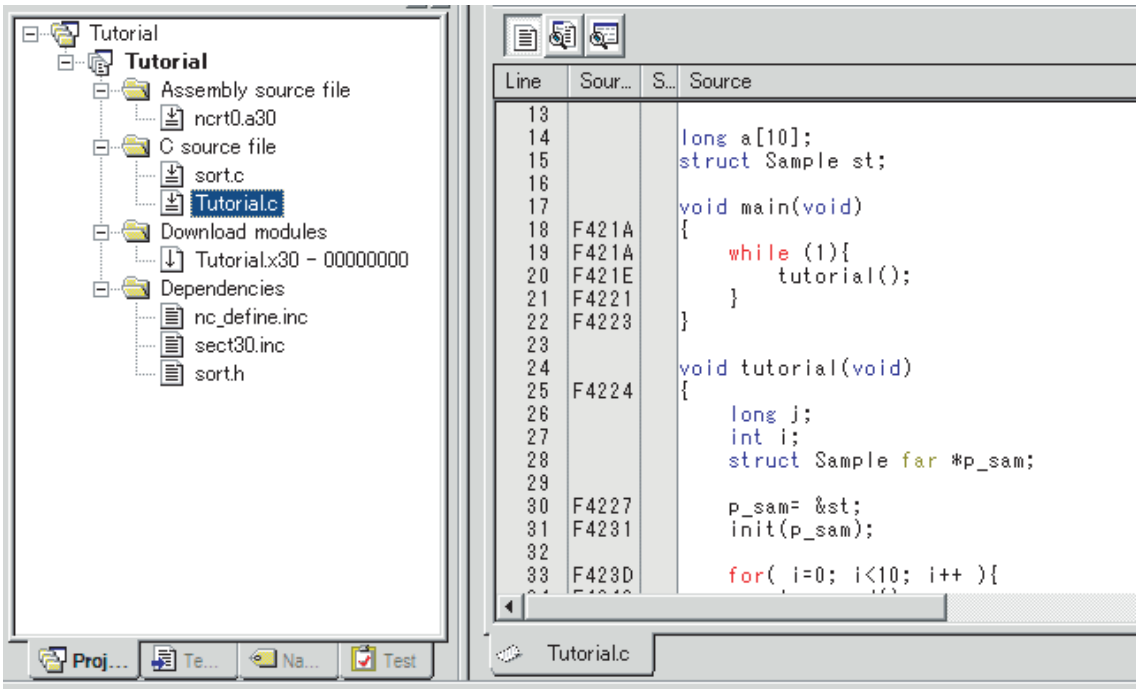
- 用于M16C/R8C、 M32C或者R32C的调试器的情况
  从[Download modules] （下载模块）的[Tutorial.x30]中选择[Download] （下载）。



6.2.3.2 源程序的显示

本调试器能以源级调试器。

请双击 [C source file] （C 源文件）的 [Tutorial.c]，打开 [Editor(Source)] （编辑器 （源））窗口，显示“Tutorial.c”文件的内容。



如果需要，请从 [Setup] （设置）菜单中选择 [Format Views...] （格式视图 ...）选项，并且选择易读的字体和大小。

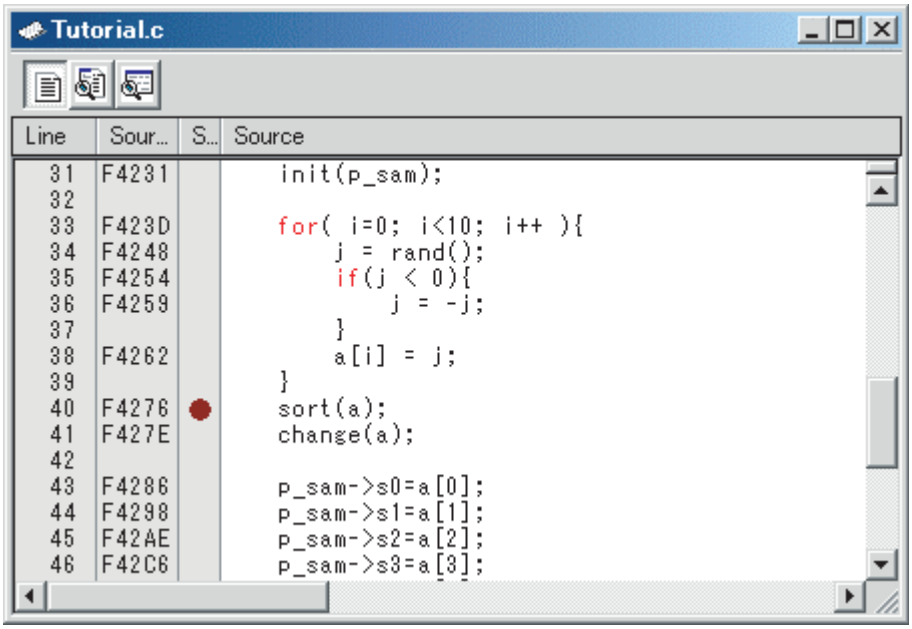
[Editor(Source)] （编辑器 （源））窗口最初显示程序的起始部分，但是能使用滚动栏查看其他部分。

6.2.4 Step4：断点的设置

基本的调试功能之一有软件断点。
能在 [Editor(Source)] （编辑器 （源））窗口中简单地设置软件断点。

6.2.4.1 软件断点的设置

例如，在调用 sort 函数的位置设置软件断点。
请双击含 sort 函数调用行的 [S/W breakpoints] （S/W 断点）列。



在 sort 函数行上显示红色圆点，表示 PC 断点已被设置。

6.2.5 Step5：程序的执行

以下说明程序的执行方法。

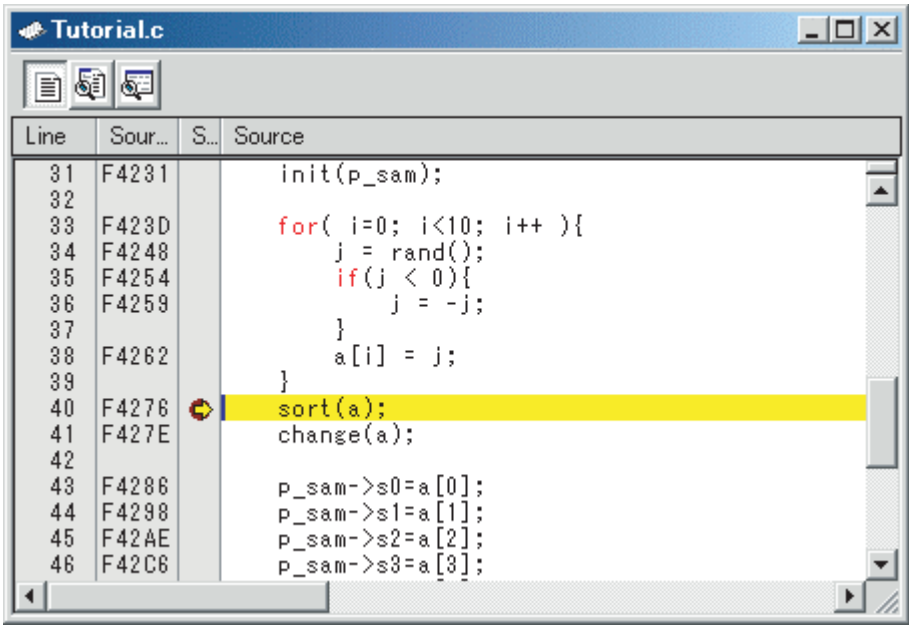
6.2.5.1 CPU 的复位

要对 CPU 进行复位时，请从 [Debug] （调试）菜单中选择 [Reset CPU] （CPU 的复位），或者选择工具栏上的 [Reset CPU] （CPU 的复位）按钮  。

6.2.5.2 程序的执行

要执行程序时，请从 [Debug] （调试）菜单中选择 [Go] （执行），或者选择工具栏上的 [Go] （执行）按钮  。

程序执行到设有断点的位置。在 [S/W breakpoints] （S/W 断点）列中显示箭头，表示程序停止的位置。

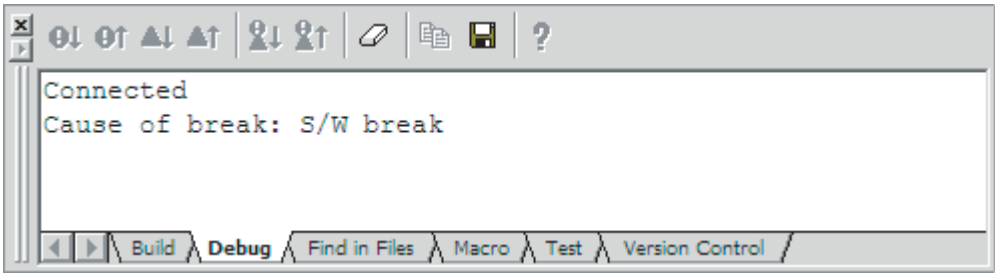


注意事项

在暂停后显示源文件时，有可能要询问源文件的路径。此时，请指定源文件的位置。

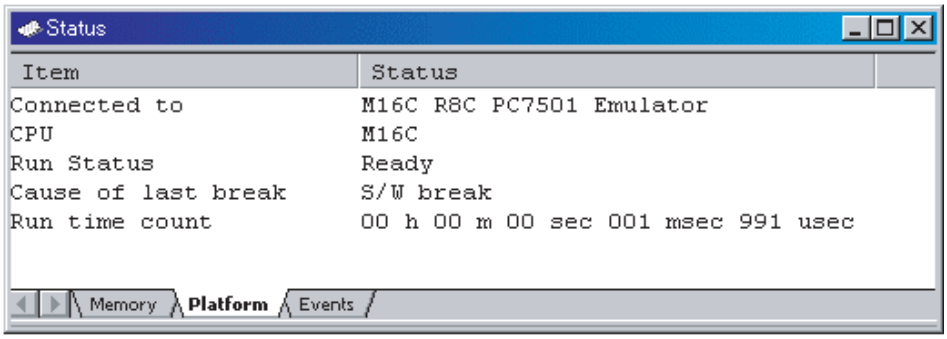
6.2.5.3 暂停原因的确认

在 [Output]（输出）窗口中显示暂停的原因。



另外，也能在 [Status]（状态）窗口中确认最后发生暂停的原因。

请从 [View]（视图）菜单的 [CPU] 子菜单中选择 [Status]（状态）。显示 [Status]（状态）窗口，请打开 [Platform]（平台）表，确认 [Cause of last break]（上次中断原因）的 [Status]（状态）。



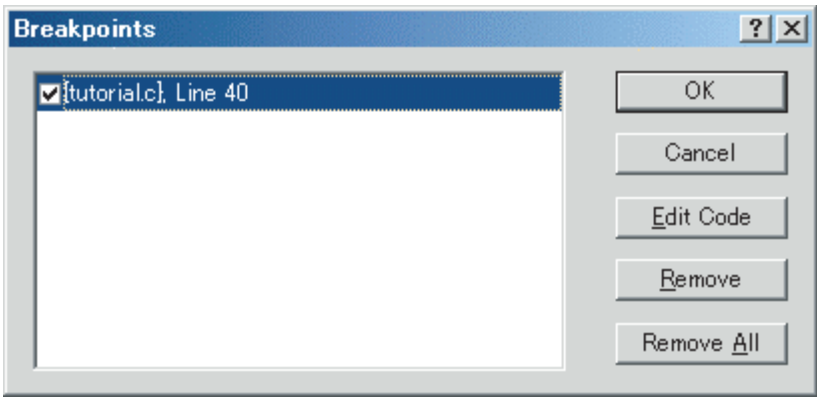
有关暂停原因的显示，请参照 “11. 程序停止原因的显示”。

6.2.6 Step6：断点的确认

能在 [Breakpoints] （断点）对话框中确认已设置的全部软件断点。

6.2.6.1 断点的确认

请按键盘的 Ctrl+B，显示 [Breakpoints] （断点）对话框。



通过此对话框能删除断点以及选择断点的有效或者无效。

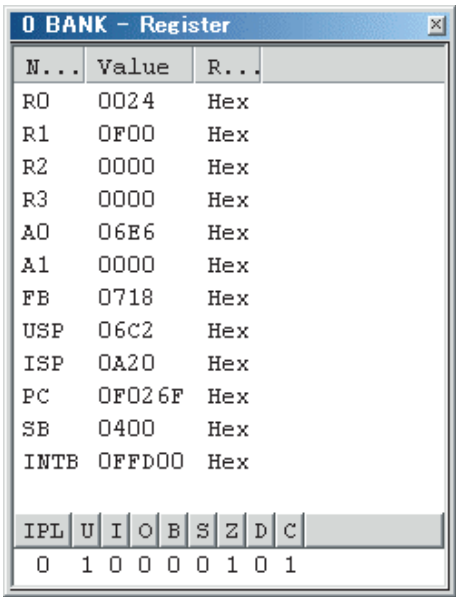
6.2.7 Step7：寄存器内容的确认

能在 [Register] （寄存器）窗口中确认寄存器的内容。

6.2.7.1 寄存器内容的确认

请从 [View] （视图）菜单的 [CPU] 子菜单中选择 [Registers] （寄存器），显示 [Register] （寄存器）窗口。

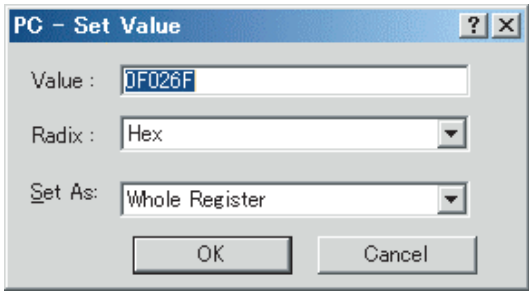
下图显示用于 M16C/R8C 的调试器的情况。



6.2.7.2 寄存器内容的更改

能更改任意寄存器的内容。

请双击要更改的寄存器行，显示对话框，输入要更改的值。



6.2.8 Step8：存储器内容的确认

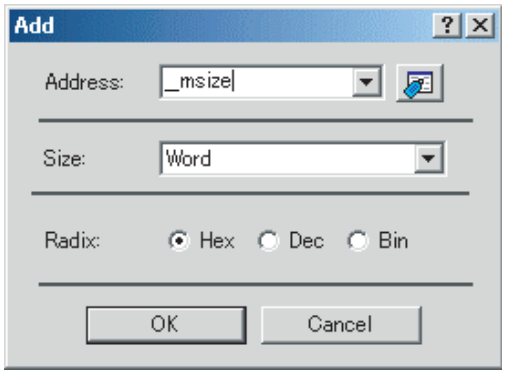
能通过指定标签名，在 [ASM Watch] （ASM 监视）窗口中确认已注册标签的存储器内容。

6.2.8.1 存储器内容的确认

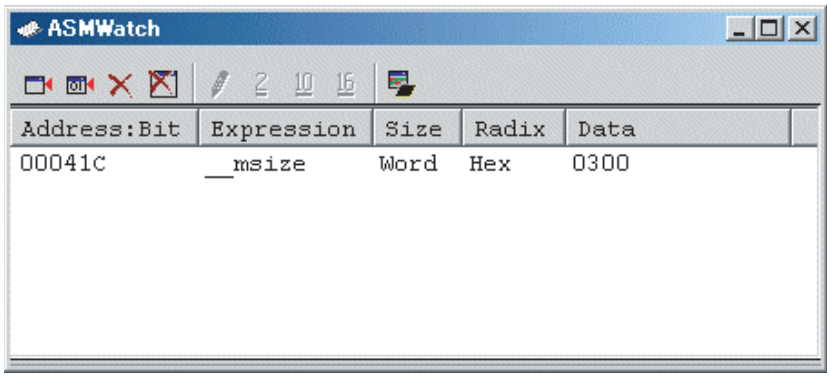
例如，按照以下的步骤，以字为单位确认对应 _msize 的存储器内容。

请从 [View] （视图）菜单的 [Symbol] （符号）子菜单中选择 [ASM Watch] （ASM 监视），显示 [ASM Watch] （ASM 监视）窗口。

请选择 [ASM Watch] （ASM 监视）窗口的弹出式菜单 [Add...] （添加 ...），在 [Address] （地址）编辑框输入 “_msize”，并且将 [Size] （长度）组合框设置为 “Word”（字）。



一旦单击 [OK] （确定）按钮，就在 [ASM Watch] （ASM 监视）窗口中显示指定的存储区域。



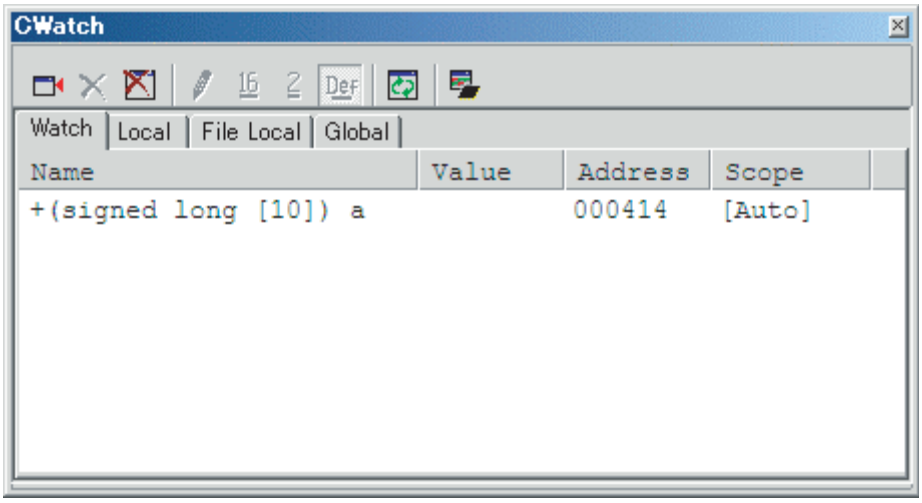
6.2.9 Step9：变量的参照

在单步处理程序时，能确认程序使用的变量值的变化。

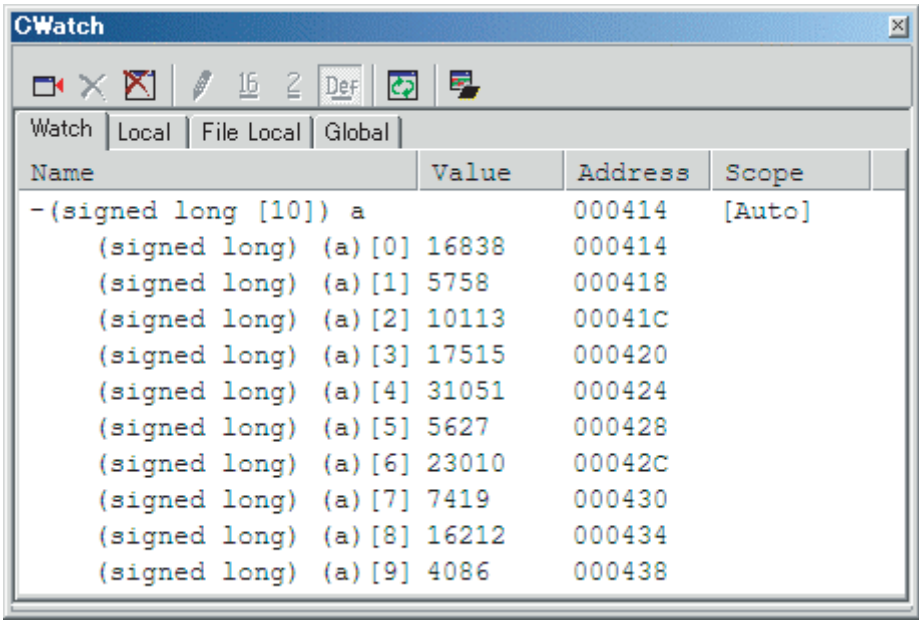
6.2.9.1 变量的参照

例如，能按照以下的步骤查看在程序开始部分声明的 long 型的数组 a。

请选择 [Editor(Source)] （编辑器 （源））窗口中显示的数组 a，用鼠标右键选择被显示的弹出式菜单的 [Add C Watch...] （添加到 C 监视 ...）。打开 [C watch] （C 监视）窗口的 [Watch] （监视）选项卡，显示数组 a 的内容。

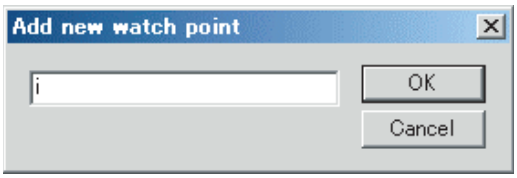


单击 [C watch] （C 监视）窗口的数组 a 左侧的 “+”标记，能参照数组 a 的各元素。

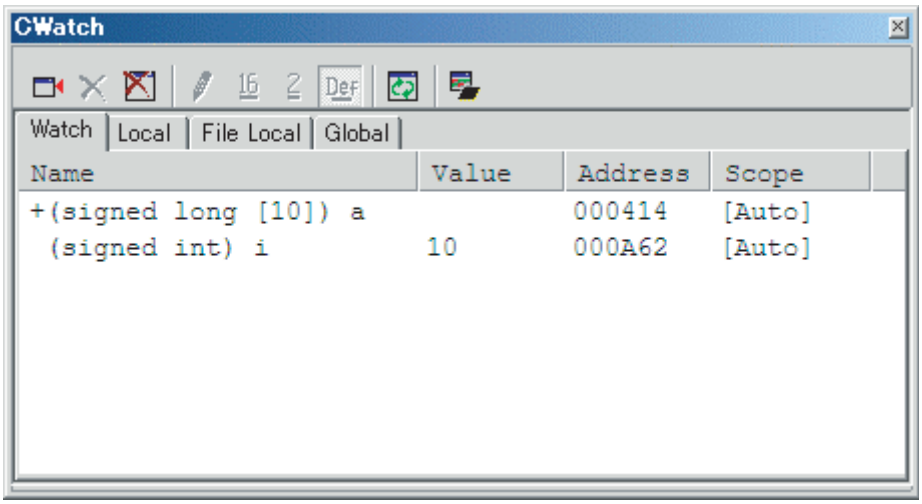


6.2.9.2 参照变量的注册

能指定变量名，给 [C Watch] （C 监视）窗口添加变量。
请从 [C Watch] （C 监视）窗口的弹出式菜单中选择 [Add...] （添加 ...）。
显示以下的对话框，请输入变量 i。



一旦单击 [OK] （确定）按钮，就在 [C Watch] （C 监视）窗口中显示 int 型的变量 i。



6.2.10 Step10：程序的单步执行

本调试器为程序的调试提供了各种有效的单步命令。

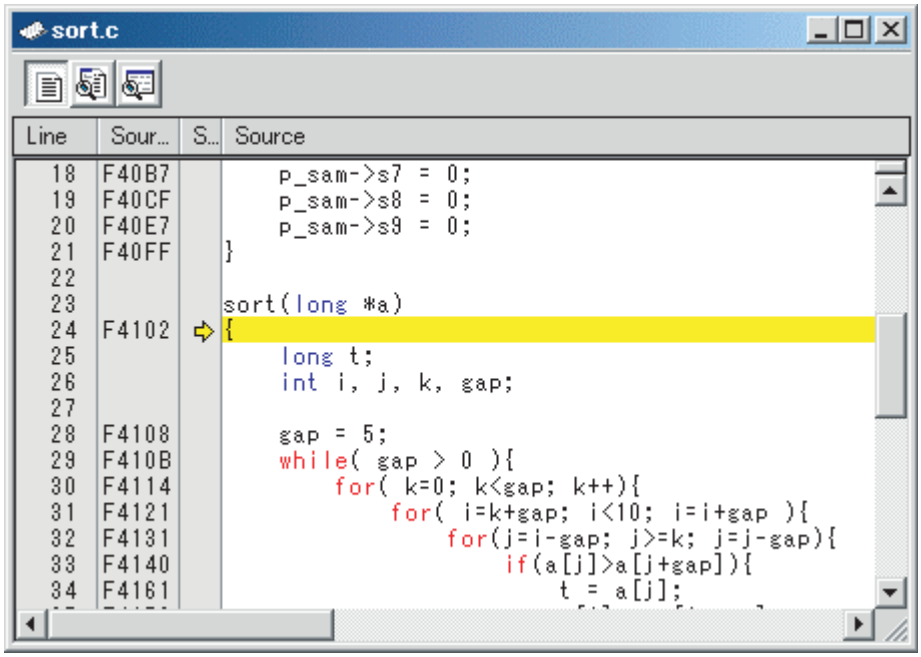
1. Step In （步进）
   执行各语句 （包括函数 （子程序）内的语句）。
2. Step Out （步出）
   跳出函数 （子程序）并且停止在调用该函数 （子程序）的程序的下一个语句。
3. Step Over （跨步）
   将函数 （子程序）调用作为1步进行单步执行。
4. Step... （单步...）
   以指定的速度进行指定次数的单步执行。

6.2.10.1 Step In （步进）的执行

Step In （步进）功能进入调用函数 （子程序），停止在调用函数 （子程序）的第一个语句。

要进入 sort 函数时，请从 [Debug] （调试）菜单中选择 [Step In] （步进），或者单击工具栏的 [Step In] （步进）按钮  。

[Editor(Source)] （编辑器 （源））窗口中指示 PC 位置的光标移到 sort 函数的第一个语句。

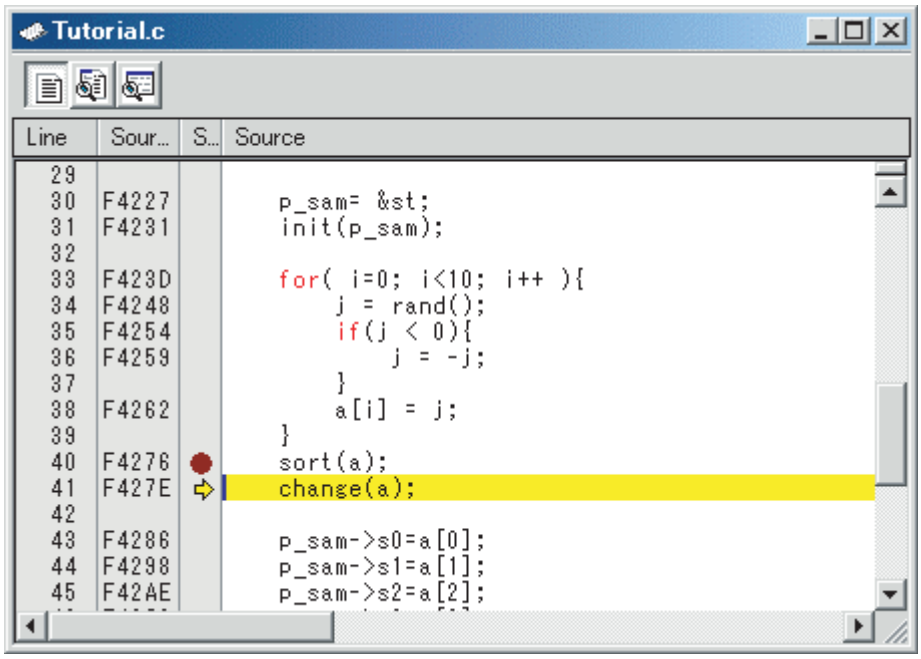


6.2.10.2 Step Out （步出）的执行

Step Out （步出）功能从调用函数 （子程序）退出，停止在调用源程序的下一个语句。

要从 sort 函数退出时，请从 [Debug] （调试）菜单中选择 [Step Out] （步出），或者单击工具栏的 [Step Out] （步出）按钮  。

[Editor(Source)] （编辑器 （源））窗口中指示 PC 位置的光标退出 sort 函数，移到 change 函数之前的位置。



注意事项

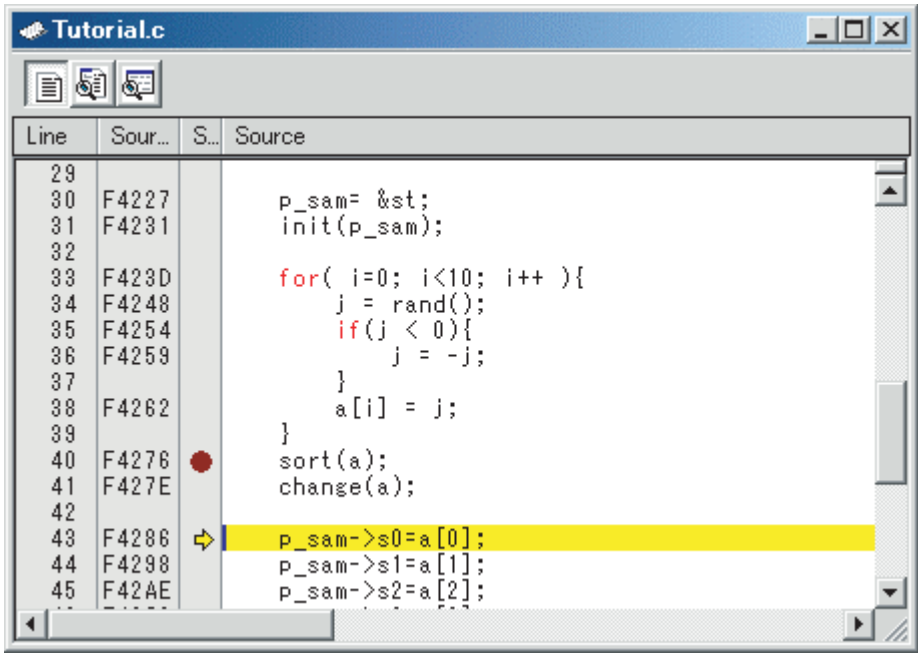
本功能需要处理时间。在知道调用源时，请使用 [Go To Cursor] （转至光标）。

6.2.10.3 Step Over （跨步）的执行

Step Over （跨步）功能将函数 （子程序）调用作为 1 步进行单步执行，停止在主程序的下一个语句。

要一次单步执行 change 函数中的全部语句时，请从 [Debug] （调试）菜单中选择 [Step Over] （跨步），或者单击工具栏的 [Step Over] （跨步）按钮  。

[Editor(Source)] （编辑器 （源））窗口中指示 PC 位置的光标移到 change 函数的下一个位置。



6.2.11 Step11：程序的强制暂停

本调试器能强制暂停程序。

6.2.11.1 程序的强制暂停

请解除全部的暂定。

要执行 main 函数的剩余部分时，请从 [Debug] （调试）菜单中选择 [Go] （执行），或者选择工具栏上的 [Go] （执行）按钮  。

因为程序正在执行无限循环处理，所以请从 [Debug] （调试）菜单中选择 [Halt Program] （停止程序），或者选择工具栏上的 [Halt] （停止）按钮  ，强制暂停程序的执行。

6.2.12 Step12：局部变量的显示

能使用 [C Watch] （C 监视）窗口显示函数内的局部变量。

6.2.12.1 局部变量的显示

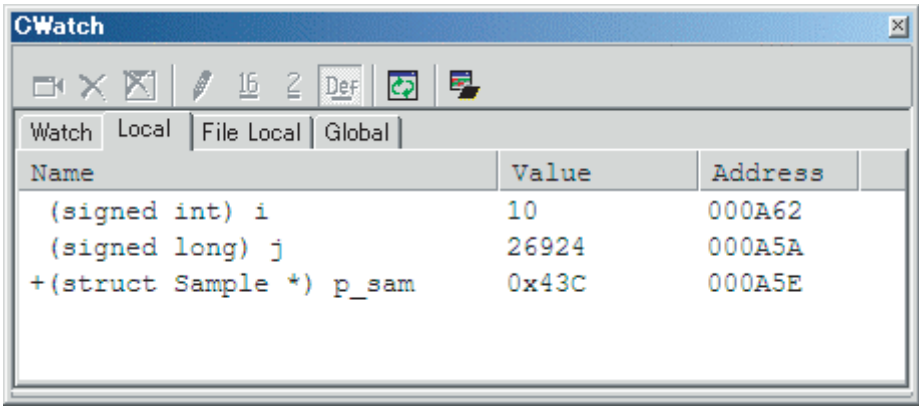
例如，检查 tutorial 函数的局部变量。

此函数声明 3 个局部变量 i、 j 和 p_sam。

请从 [View] （视图）菜单的 [Symbol] （符号）子菜单中选择 [C Watch] （C 监视），显示 [C Watch] （C 监视）窗口。在 [C Watch] （C 监视）窗口中，默认设置有以下 4 个选项卡。

- [Watch] （监视）选项卡
  只显示用户注册的变量。
- [Local] （局部）选项卡
  显示当前PC所在块中的全部能参照的局部变量。如果因执行程序而使作用域发生变化， [Local] （局部）选项卡的内容也会发生变化。
- [File Local] （文件局部变量）选项卡
  显示当前PC所在文件中的全部文件局部变量。如果因执行程序而使作用域发生变化， [File Local] （文件局部变量）选项卡的内容也会发生变化。
- [Global] （全局）选项卡
  显示被下载的程序所使用的全部全局变量。

要显示局部变量时，请选择 [Local] （局部）选项卡。



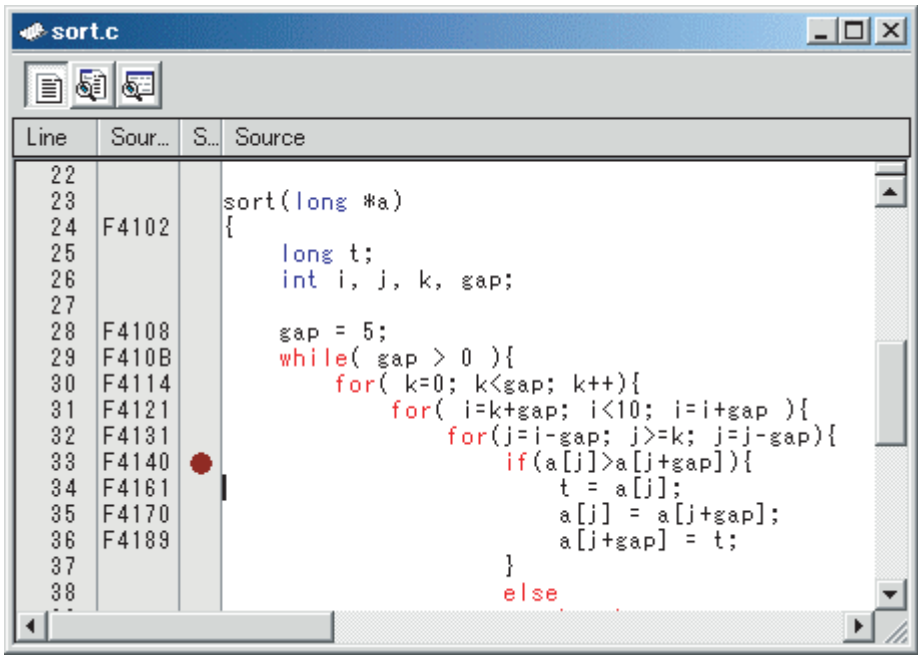
请双击指针变量 p_sam 左侧的 “+” 标记，显示结构体 *(p_sam)。
如果在 Tutorial 函数的最后参照 *(p_sam) 的结构体成员，就能看到随机数据已按照降序进行排序。

6.2.13 Step13 ： 堆栈跟踪

本调试器能使用堆栈信息，显示当前 PC 所在的函数被哪个函数调用。

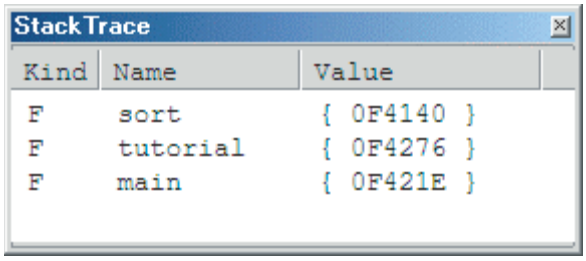
6.2.13.1 函数调用情况的参照

请双击 sort 函数内的行的 [S/W Breakpoints] （S/W 断点）列，设置软件断点。



将程序复位后重新执行。请从 [Debug] （调试）菜单中选择 [Reset Go] （复位执行），或者选择工具栏上的 [Reset Go] （复位执行）按钮  。

在程序暂停后，请从 [View] （视图）菜单的 [Code] （代码）子菜单中选择 [Stack Trace] （堆栈跟踪），打开 [Stack Trace] （堆栈跟踪）窗口。



能知道当前 PC 在 sort() 函数内并且 sort() 函数被 tutorial() 函数调用。

6.2.14 结尾

本教程介绍了本调试器的主要使用方法。

能通过所用仿真器提供的仿真功能进行更高级的调试。因此，如果正确地区分和识别硬件和软件问题的发生条件，就能有效地检查这些问题。

参考篇

7. 窗口一览表

本调试器使用的窗口如下所示：

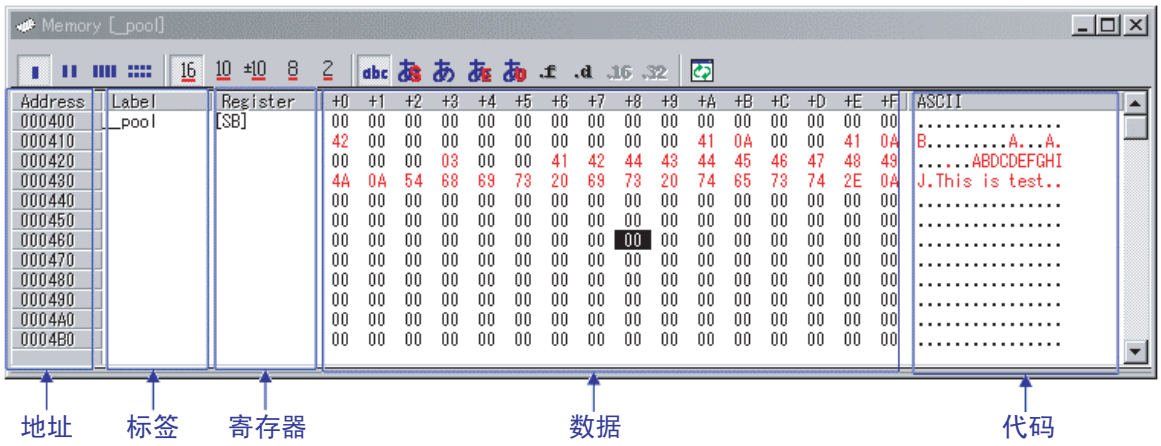
窗口名	显示的菜单
[RAM Monitor] （RAM 监控）窗口	[View] （视图） →[CPU]→[RAM Monitor] （RAM 监控）
[ASM Watch] （ASM 监视）窗口	[View] （视图） →[Symbol] （符号） →[ASM Watch] （ASM 监视）
[C Watch] （C 监视）窗口	[View] （视图） →[Symbol] （符号） →[C Watch] （C 监视）
[Script] （脚本）窗口	[View] （视图） →[Script] （脚本）
[S/W Breakpoint Setting] （S/W 断点设置）窗口	[View] （视图） →[Break] （暂停） →[S/W Break Points] （S/W 断点）
[Event Setting] （事件设置）窗口	[View] （视图） →[Trace] （跟踪） →[Trace Points] （跟踪点） [View] （视图） →[Break] （暂停） →[H/W Break Points] （H/W 断点）
[Time Measurement] （间隔时间测量）窗口	[View] （视图） →[Trace] （跟踪） →[Time Measure] （间隔时间测量）
[Trace] 跟踪窗口	[View] （视图） →[Trace] （跟踪） →[Trace]
[GUI I/O] （GUI 输入 / 输出）窗口	[View] （视图） →[Graphic] （图形） →[GUI I/O]
MR 窗口	[View] （视图） →[RTOS]→[MR]
[MR Trace] （MR 跟踪）窗口	[View] （视图） →[RTOS]→[MR Trace] （MR 跟踪）
[MR Analyze] （MR 分析）窗口	[View] （视图） →[RTOS]→[MR Analyze] （MR 分析）

以下的窗口请参照 High-performance Embedded Workshop 的帮助。

- [Differences] （差异）窗口
- [Map] （映像）窗口
- [Command Line] （命令行）窗口
- [Workspace] （工作空间）窗口
- [Output] （输出）窗口
- [Disassembly] （反汇编）窗口
- [Memory] （存储器）窗口
- [IO] 窗口
- [Status] （状态）窗口
- [Register] （寄存器）窗口
- [Image] （图像）窗口
- [Waveform] （波形）窗口
- [Stack Trace] （堆栈跟踪）窗口

7.1 [RAM Monitor] （RAM 监控）窗口

RAM 监控窗口是显示目标程序执行过程中的存储器变化的窗口。
在目标程序执行过程中，按一定的间隔更新显示内容。



- 有1K字节的RAM监控区域。此RAM监控区域能分配到任意的连续地址，或者能以16字节为单位拆分为64块区域。
- RAM监控区域能更改任意的地址范围。
  有关RAM监控区域的更改方法，请参照 “7.1.2 RAM监控区域的设置”。
- 能按各窗口设置显示内容的更新间隔。
  在[Address] （地址）显示栏的标题部分显示目标程序执行过程中的实际更新间隔。

注意事项

- 根据运行情况 （以下情况），显示的更新间隔可能长于指定的更新间隔。
  - 主机的性能/负载情况
  - 通信接口
  - 窗口的大小 （存储器的显示范围）和显示的窗口数

7.1.1 选项菜单

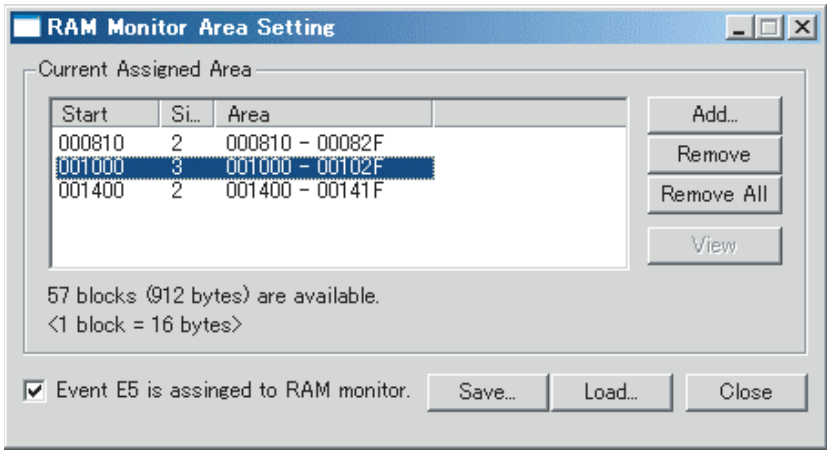
如果在窗口中单击鼠标的右键，就显示以下的弹出式菜单。在工具栏的按钮中也分配了这些菜单的主要功能。

菜单名		功能
[RAM Monitor Area...]（RAM 监控区域 ...）		设置 RAM 监控区域。
[Sampling Period...]（采样周期 ...）		设置采样周期。
[Clear]（清除）		清除存取履历。
[Up]（上移）		将显示位置移置紧邻的上一 RAM 监控区域（较小地址）。
[Down]（下移）		将显示位置移置紧邻的下一 RAM 监控区域（较大地址）。
[Address...]（地址 ...）		更改显示的起始地址。
[Scroll Area..]. （滚动范围 ...）		设置滚动范围。
[Data Length]（数据长度）	[1byte]（1 字节）	以 1Byte 为单位显示。
	[2bytes]（2 字节）	以 2Byte 为单位显示。
	[4bytes]（4 字节）	以 4Byte 为单位显示。
	[8bytes]（8 字节）	以 8Byte 为单位显示。
[Radix]（基数）	[Hex]（16 进制）	用 16 进制数显示。
	[Dec]（10 进制）	用 10 进制数显示。
	[Single Dec]（带符号的 10 进制）	用带符号的 10 进制数显示。
	[Oct]（8 进制）	用 8 进制数显示。
	[Bin]（2 进制）	用 2 进制数显示。
[Code]（代码）	[ASCII]	用 ASCII 码显示。
	[SJIS]	用 SJIS 码显示。
	[JIS]	用 JIS 码显示。
	[UNICODE]	用 UNICODE 码显示。
	[EUC]	用 EUC 码显示。
	[Float]	用 Float 型显示。
	[Double]	用 Double 型显示。
[Layout]（布局）	[Label]（标签）	切换标签显示栏的显示或者隐藏。
	[Register]（寄存器）	切换寄存器显示栏的显示或者隐藏。
	[Code]（代码）	切换代码显示栏的显示或者隐藏。
[Column...]（列 ...）		更改显示的列数。
[Split]（拆分）		窗口拆分显示。
[Toolbar display]（工具栏显示）		切换工具栏的显示或者隐藏。
[Customize toolbar...]（自定义工具栏 ...）		自定义工具栏。
[Allow Docking]（允许入坞）		允许窗口入坞。
[Hide]（隐藏）		隐藏窗口。

7.1.2 RAM 监控区域的设置

请选择 RAM 监控窗口的弹出式菜单 [RAM Monitor Area...]（RAM 监控区域 ...）。

打开 [RAM monitor area setting]（RAM 监控区域设置）窗口，在一览表中显示现在设置的 RAM 监控区域。



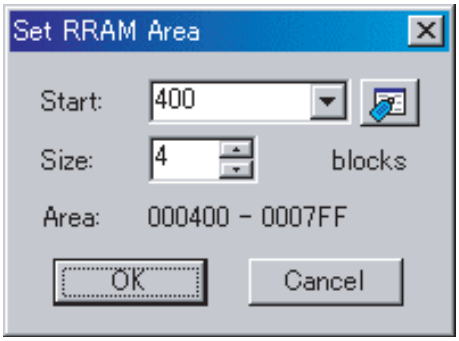
通过此窗口进行 RAM 监控区域的添加、删除和更改。

- 用起始地址和大小（用块数指定）指定RAM监控区域。
- 能以0x10字节为单位指定起始地址。
  如果指定尾数的地址值，就设置以0x10字节为单位舍入的值。
- 用块数指定大小。
  E30A的1块大小为16字节，最多能指定64块。
- 可以一直添加RAM监控区域，直到使用的总块数达到64块。
  （在列表的下部显示现在能使用的块数（和大小））

7.1.2.1 RAM 监控区域的更改

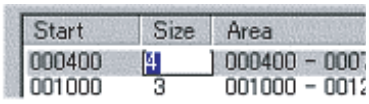
能更改 RAM 监控区域的起始地址和大小。

- 通过对话框进行更改。
  请从RAM监控区域的一览表中选择并双击要更改的RAM监控区域。
  显示以下的对话框，请给[Start] （开始地址）栏和[Size] （大小）栏分别指定起始地址和大小 （用块数指定）。



- 在窗口中直接更改。
  请将RAM监控区域的一览表中要更改的RAM监控区域置为选择状态，然后单击[Start] （开始）显示栏或者[Size] （大小）显示栏。
  显示编辑框，请分别指定要更改的内容。用Enter键确定输入，用Esc键取消操作。

更改地址



更改大小

7.1.2.2 RAM 监控区域的添加

请单击 [Add...] （添加 ...）按钮。
显示对话框，请给 [Start] （开始）栏和 [Size] （大小）栏分别指定起始地址和大小 （用块数指定）。

7.1.2.3 RAM 监控区域的删除

请将 RAM 监控区域的一览表中要删除的 RAM 监控区域置为选择状态，然后单击 [Remove] （移除）按钮。

要删除全部的 RAM 监控区域时，请单击 [Remove All] （全部移除）按钮。

7.1.2.4 RAM 监控事件的设置

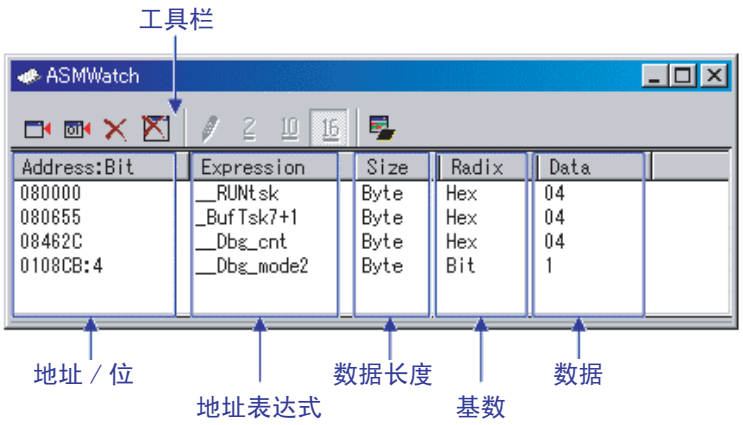
要设置 RAM 监控事件时，请选定以下的复选框。

将 RAM 监控事件设置为事件 E5。
请注意：在没有设置 RAM 监控事件的情况下， RAM 监控功能不运行。

7.2 [ASM Watch] （ASM 监视）窗口

[ASM Watch] （ASM 监视）窗口是能将特殊地址注册为监视点并参照存储器内容的窗口。

如果注册的地址在 RAM 监控区域内，就在目标程序执行过程中按一定的间隔 （默认值：100msec）更新存储器的内容。



- 注册的地址称为监视点。能注册以下的内容：
  - 地址 （可用符号指定）
  - 地址+位号
  - 位符号
- 注册的监视点在关闭ASM监视窗口时被保存，在重新打开ASM监视窗口时被自动注册。
- 如果给监视点指定了符号/位符号，就在下载目标程序时重新计算监视点的地址。
- 无效的监视点显示 “--<not active>--”。
- 能更改监视点的排列顺序 （通过拖放功能）。
- 能通过同址编辑更改监视点的地址表达式、长度、基数和数据。

注意事项

- 通过存储器转储实现 RAM 监控功能。如果通过 RAM 监控功能存取存储器，就会失去程序的实时性。
- 不显示存取属性。

7.2.1 选项菜单

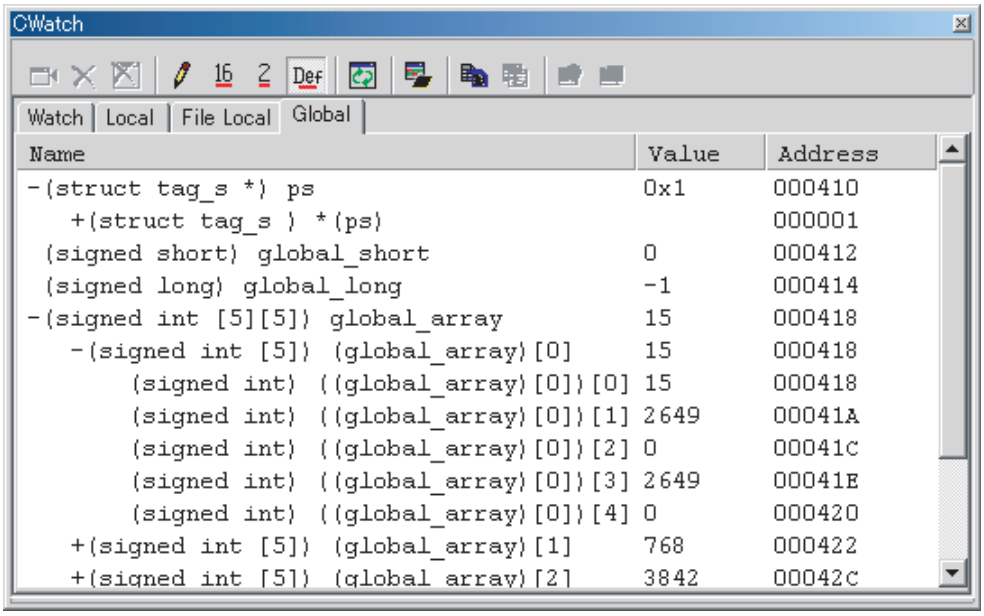
如果在窗口中单击鼠标的右键，就显示以下的弹出式菜单。在工具栏的按钮中也分配了这些菜单的主要功能。

菜单名		功能
[Add...]（添加）		添加监视点。
[Add Bit...]（添加位 ...）		添加位标签的监视点。
[Remove]（移除）		移除所选的监视点。
[Remove All]（全部移除）		移除全部监视点。
[Set...]（设置 ...）		设置所选的监视点的值。
[Radix]（基数）	[Bin]（2 进制）	以 2 进制数显示。
	[Dec]（10 进制）	以 10 进制数显示。
	[Hex]（16 进制）	以 16 进制数显示。
[Refresh]（刷新）		刷新存储器。
[Layout]（布局）	[Address Area]（地址区域）	切换地址的显示或者隐藏。
	[Size Area]（大小区域）	切换长度的显示或者隐藏。
[RAM Monitor]（RAM 监控）	[Enable RAM Monitor]（RAM 监控有效化）	将 RAM 监控功能置为有效。
	[Sampling Period...]（采样周期）	设置采样周期。
[Toolbar display]（工具栏显示）		切换工具栏的显示或者隐藏。
[Customize toolbar...]（自定义工具栏 ...）		自定义工具栏。
[Allow Docking]（允许入坞）		允许窗口入坞。
[Hide]（隐藏）		隐藏窗口。

7.3 [C Watch] （C 监视）窗口

[C Watch] （C 监视）窗口是参照用 C 语言或者 C++ 语言创建的程序所用变量的窗口。显示的变量称为 C 监视点。

如果注册的监视点在 RAM 监控区域内，就在目标程序执行过程中按一定的间隔 （默认值：100msec）更新存储器的内容。



- 能按作用域 （局部、文件局部或者全局）参照变量。
- 根据PC值的变化，显示自动更新。
- 能更改变量值。
- 能按变量更改显示基数。
  - 能更改默认的显示基数。
  - 当用16进制数显示时，能选择是否显示高位的0。
- 能将任意的变量注册到[Watch] （监视）选项卡，并且显示注册的变量。
  - 按工程保存被注册的内容。
  - 如果打开多个C监视窗口，就在全窗口中共有[Watch] （监视）选项卡的注册内容。
  - 能从停止时的作用域 （[Auto]）、全局 （[Global]）、各文件内的静态变量中选择参照目标。
- 能通过添加[Watch] （监视）选项卡来划分C监视点的注册处。
- 能通过拖放从其他窗口或者编辑器注册变量。
- 能按名字和地址的顺序排序。
- 能将RAM监控分配到指定变量的地址。

注意事项

- 不能更改以下的 C 监视点的值。
  - 寄存器变量
  - 没有指示存储器实体 （地址和大小）的C/C++语言表达式
- 当不能正确计算 C/C++ 语言表达式时 （未定义 C 符号等），被注册为无效的 C 监视点。
- 不保存 [Local] （局部）、 [File Local] （文件局部）、 [Global] （全局）选项卡的显示设置，而保存 [Watch] （监视）选项卡以及新添加的选项卡内容。
- 只有全局变量和文件局部变量才能通过 RAM 监控功能进行更新。
- 通过存储器转储实现 RAM 监控功能。如果通过 RAM 监控功能存取存储器，就会失去程序的实时性。
- 不显示存取属性。

有关其他 C 变量的内容，请参照 “12.1.3 C 变量的参照和设置”。

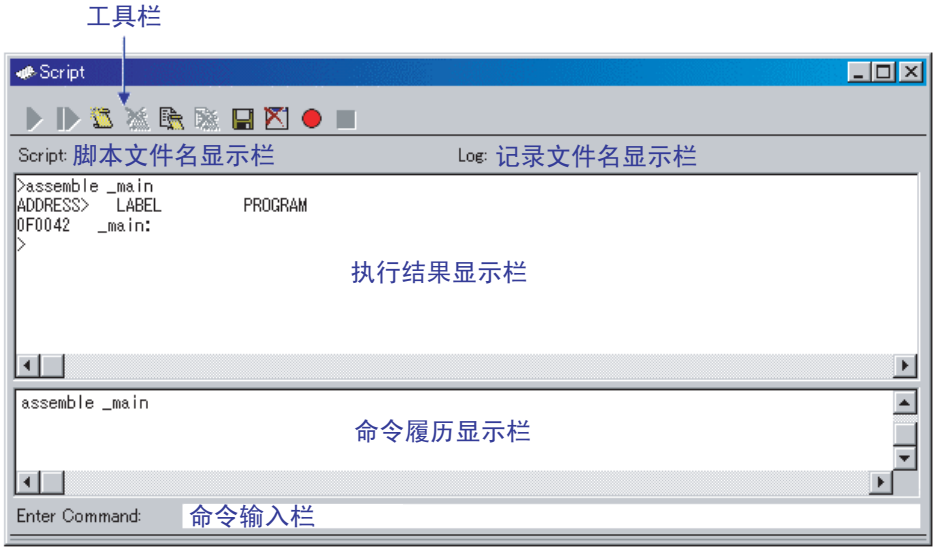
7.3.1 选项菜单

如果在窗口中单击鼠标的右键，就显示以下的弹出式菜单。在工具栏的按钮中也分配了这些菜单的主要功能。

菜单名		功能
[Add...]（添加）		添加符号。
[Remove]（移除）		移除所选的符号。
[Remove All]（全部移除）		移除全部符号。
[Initialize]（初始化）		重新评估所选的符号。
[Set New Value...]（设置新值 ...）		编辑所选的符号。
[Radix]（基数）	[Hex]（16 进制）	以 16 进制数显示。
	[Bin]（2 进制）	以 2 进制数显示。
	[Default]（默认值）	以默认的基数显示。
	[Toggle(All Variables)]（交替）（全部变量）	更改显示基数（交替）。
	[Set initial...]（设置初始基数 ...）	设置初始的显示基数。
[Refresh]（刷新）		刷新存储器。
[Hide type name]（隐藏类型名称）		隐藏型名。
[Show char*as string]（char* 的字符串显示）		显示 char* 的字符串。
[Zero suppress in Hex display（16 进制数显示中消零）		16 进制显示中消零。
[Sort]（排序）	[Sort by Name]（按名字顺序）	按名字顺序将符号重新排序。
	[Sort by Address]（按地址顺序）	按地址顺序将符号重新排序。
[RAM Monitor]（RAM 监控）	[Enable RAM Monitor]（RAM 监控有效化）	将 RAM 监控功能置为有效。
	[Sampling Period...]（采样周期）	设置采样周期。
	[Arrange a RAM monitor area around this variable]（将 RAM 监控区域设置给此变量）	将 RAM 监控区域设置给此变量。
	[Start Recording...]（开始记录 ...）	开始更新值的记录。
	[Stop Recording（停止记录）	停止更新值的记录。
[Add New Tab...]（添加新选项卡 ...）		添加选项卡。
[Remove Tab]（移除选项卡）		移除选项卡。
[Copy]（复制）		将选择的项目复制到剪贴板。
[Copy All]（全部复制）		将表内的全部项目复制到剪贴板。
[Toolbar display]（工具栏显示）		切换工具栏的显示或者隐藏。
[Customize toolbar...]（自定义工具栏 ...）		自定义工具栏。
[Allow Docking]（允许入坞）		允许窗口入坞。
[Hide]（隐藏）		隐藏状态。

7.4 [Script]（脚本）窗口

[Script]（脚本）窗口是执行脚本命令的窗口。
从窗口下部的命令输入栏输入脚本命令。命令的执行结果显示在执行结果显示栏。
工具栏的按钮分配了主要的操作。



- 能通过事先将要执行的脚本命令记述到文件（脚本文件）里进行批执行。
- 能将脚本命令的执行结果保存到事先指定的文件（记录文件）。
- 脚本窗口有保存最近1000行执行结果的缓冲器（视图缓冲器）。即使忘记指定记录文件，也能将脚本命令的执行结果保存到文件（视图文件）。
- 能将要执行的命令保存到事先指定的文件（能作为脚本文件再次使用）。

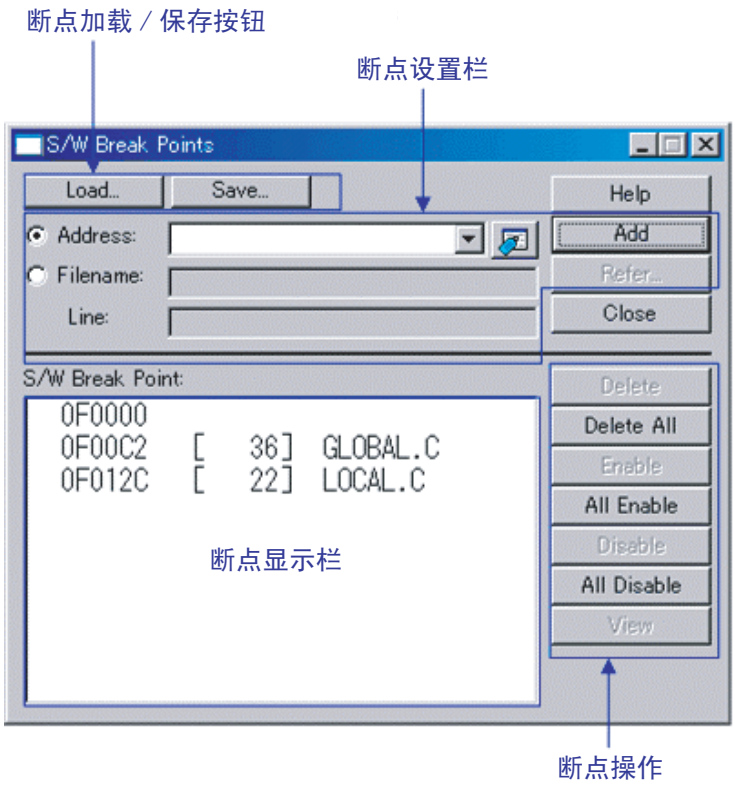
7.4.1 选项菜单

如果在窗口中单击鼠标的右键，就显示以下的弹出式菜单。在工具栏的按钮中也分配了这些菜单的主要功能。

菜单名		功能
[Script] （脚本）	[Open...] （打开）	打开脚本文件。
	[Run] （执行）	执行脚本文件。
	[Step] （单步）	单步执行脚本文件。
	[Close] （关闭）	关闭脚本文件。
[View] （视图）	[Save...] （保存）	将执行结果的显示保存到文件。
	[Clear] （清除）	清除执行结果的显示。
[Log] （日志）	[On...] （开）	打开记录文件，开始输出。
	[Off] （关）	结束输出，关闭记录文件。
[Record] （记录）	[On...] （开）	开始将命令记录到文件。
	[Off] （关）	停止将命令记录到文件。
[Copy] （复制）		复制选择范围，保存到剪贴板。
[Paste] （粘贴）		粘贴剪贴板的内容。
[Cut] （剪切）		剪切选择范围，保存到剪贴板。
[Delete] （删除）		删除选择范围。
[Undo] （撤销）		撤销刚才的操作。
[Toolbar display] （工具栏显示）		切换工具栏的显示或者隐藏。
[Customize toolbar...] （自定义工具栏 ...）		自定义工具栏。
[Allow Docking] （允许入坞）		允许窗口入坞。
[Hide] （隐藏）		隐藏窗口。

7.5 [S/W Breakpoint] （S/W 断点）设置窗口

S/W 断点设置窗口是设置软件断点的窗口。
软件暂停是指在执行指定地址的指令前停止执行。



- 能用 “地址” 或者 “文件名+行号” 指定断点。
- 如果设置多个断点，就在到达某个断点时 （OR条件）停止执行目标程序。
- 能对各断点进行删除以及无效和有效的切换。
- 能将断点信息保存到文件，也能读入被保存的断点信息。

7.5.1 命令按钮

窗口中的各按钮的含义如下：

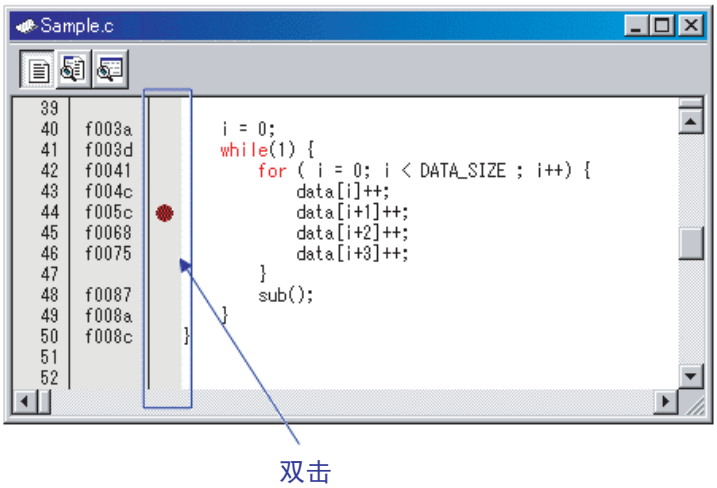
按钮名	功能
[Load...] （加载）	加载被保存在文件内的设置内容。
[Save...] （保存）	将窗口中设置的内容保存到文件。
[Help] （帮助）	显示帮助。
[Add] （添加）	设置软件断点。
[Refer...] （参照）	指定源文件。
[Close] （关闭）	关闭窗口。
[Delete] （删除）	删除所选软件断点。
[Delete All] （全部删除）	删除全部软件断点。
[Enable] （有效）	将选择的软件断点置为有效。
[All Enable] （全部有效）	将软件断点全部置为有效。
[Disable] （无效）	将选择的软件断点置为无效。
[All Disable] （全部无效）	将软件断点全部置为无效。
[View] （视图）	在编辑器 （源）窗口中显示所选软件断点的位置。

7.5.2 [Editor(Source)] （编辑器 （源））窗口中的断点设置和解除

可设置软件断点的区域因产品而不同。

详细内容请参照 “12.1.2 能设置软件断点的区域”。

请在编辑器 （源）窗口的 S/W 断点设置列上双击要设置断点的行 （在设置行显示红色圆点）。

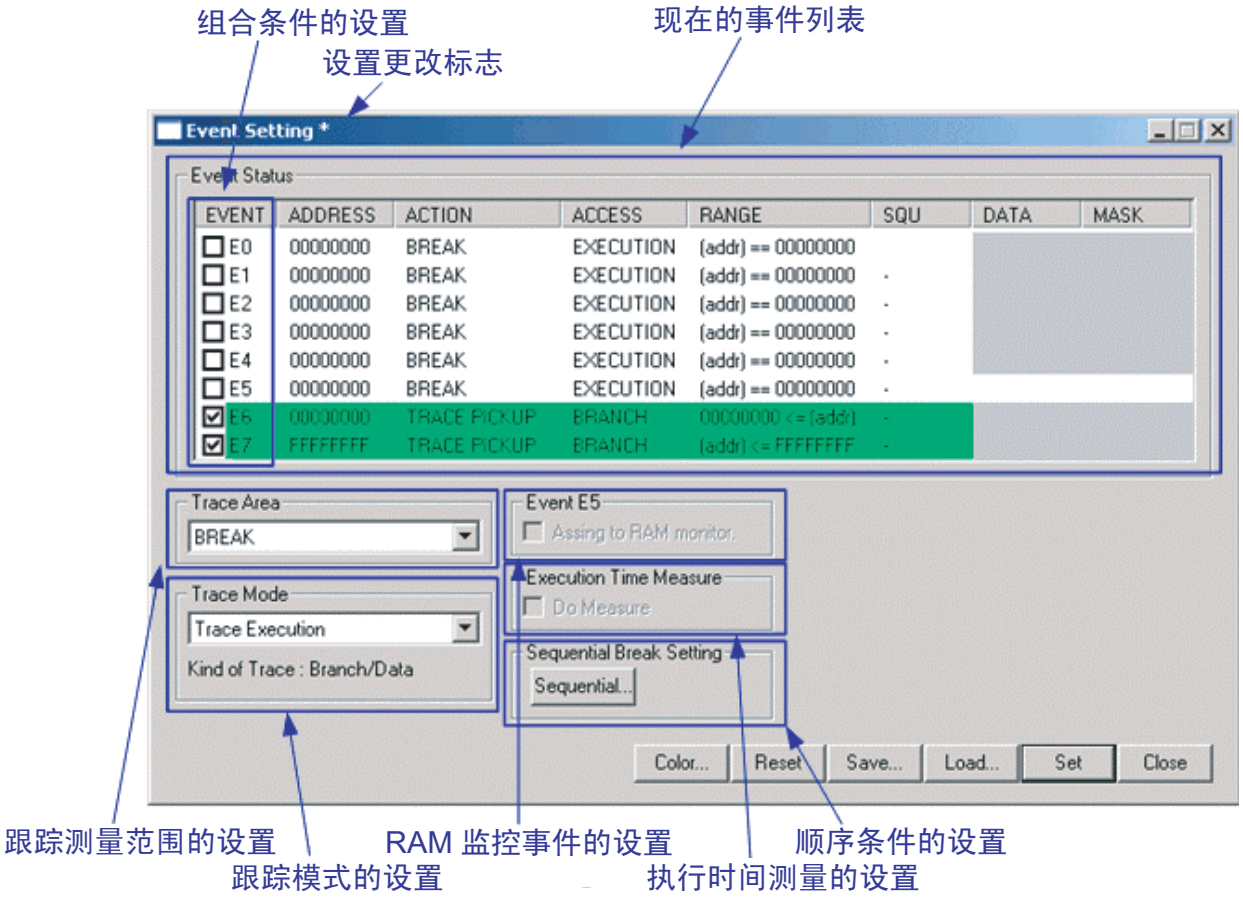


如果再次双击此行，就解除被设置的断点 （红色圆点消失）。

根据默认值，在编辑器 （源）窗口中显示用于设置 S/W 断点的列。要设置为 [Hide] （隐藏）时，请在通过菜单 [Edit] （编辑）→[Define Column Format] （定义列格式）打开的对话框中取消 [S/W breakpoints] （S/W 断点）复选框的选定，全部编辑器 （源）窗口的 S/W 断点设置列为隐藏状态。另外，通过选择编辑器 （源）窗口的弹出式菜单 [Columns] （列） →[S/W breakpoints] （S/W 断点），能按各编辑器 （源）窗口设置 S/W 断点的列。

7.6 事件设置窗口

事件设置窗口是设置暂停事件、跟踪事件、时间测量事件和 RAM 监控事件的窗口。



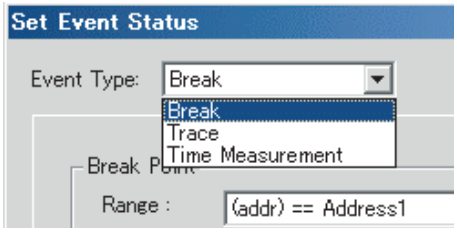
- 如果更改事件的内容，就在标题栏显示 “*”。对仿真器进行设置后不显示 “*”。
- 暂停事件、跟踪事件、时间测量事件和RAM监控事件能共用8点事件。事件的组合条件如下：
  - 有效事件中的某个事件成立 （OR条件）。
  - 按指定的顺序事件成立 （顺序条件）。

注意事项

- 要将已设置的事件更改为其他事件时，请先在 [EVENT] （事件）栏解除事件的设置。
- 只能在暂停事件的情况下指定事件组合条件的顺序条件。

7.6.1 事件的设置

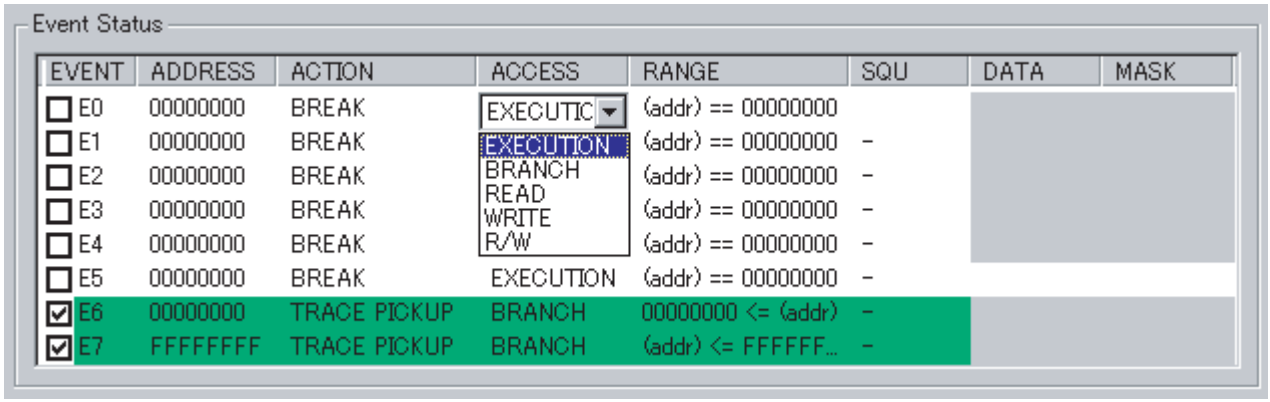
要设置事件时，请双击事件设置窗口的事件设置栏，打开事件的设置对话框。
通过更改事件设置对话框的事件种类，能更改所指定的事件的种类。



如果在事件的设置对话框中设置了事件，就设置为未使用事件。未使用事件的检索顺序如下：

E7->E6->E3->E2->E1->E5->E4->E0

E0 和 E4 事件是指定跟踪或者时间测量的开始和结束的特定事件，E5 是数据比较暂停事件或者 RAM 监控事件，所以降低了优先顺序。
另外，也能在事件设置窗口的事件设置栏中直接设置。请选择要设置的事件的行，然后单击条件。



能进行以下的设置作为各事件的条件。

7.6.1.1 存取条件的选择

选择事件成立的存取条件。在事件设置窗口的 [ACCESS] （存储）栏中进行选择。

存取条件	内容
[EXECUTION] （执行）	执行指令
[BRANCH] （转移）	转移
[READ] （读）	读数据
[WRITE] （写）	写数据
[R/W] （读 / 写）	读写数据

各事件能选择的 ACCESS 条件如下所示：

事件	ACCESS 条件				
	EXECUTION	BRANCH	READ	WRITE	R/W
暂停事件	○	×	○	○	○
跟踪事件 （跟踪抽取事件）	×	○	○	○	○
跟踪事件 （跟踪开始事件 / 跟踪结束事件）	○	×	○	○	○
间隔时间测量事件	×	×	○	○	○
RAM 监控事件	—	—	—	—	—

7.6.1.2 地址匹配或者范围指定的选择

选择地址匹配或者范围指定作为事件成立的地址范围条件。在事件设置窗口的 RANGE 栏中进行选择。

范围条件	内容
EQU	地址的匹配
IN(START)	地址的范围指定 （起始地址）
IN(END)	地址的范围指定 （结束地址）

对于由 ACCESS 条件选择 [EXECUTION] （执行）的事件，不能设置地址的范围指定。

只能选择以下的事件组合：

- E0 （起始地址）和E1 （结束地址）
- E2 （起始地址）和E3 （结束地址）
- E4 （起始地址）和E5 （结束地址）
- E6 （起始地址）和E7 （结束地址）

7.6.2 事件成立时的运行选择

选择事件成立时的运行条件。在事件设置窗口的 [ACTION] （运行）栏中进行选择。

EVENT 栏中选择的事件的显示颜色因 ACTION 栏中选择的事件运行而不同，能通过显示颜色按钮更改显示的颜色。

运行条件	内容	显示的颜色 （默认值）
[BREAK] （暂停）	暂停	红色
[TRACE PICKUP] （跟踪抽取）	跟踪抽取	绿色
[TIME START] （开始时间）	开始测量执行时间。	黄色
[TIME END] （结束时间）	结束测量执行时间。	黄色
[TIME] （时间）	测量间隔时间。	橙色
[TRACE START] （开始跟踪）	开始跟踪。	蓝色
[TRACE END] （结束跟踪）	结束跟踪。	蓝色
[RAM MONITOR] （RAM 监控）	使用 RAM 监控功能。	紫色

各事件能选择的 ACTION 条件如下所示：

事件	ACTION 条件							
	BREAK	TRACE PICKUP	TIME START	TIME END	TIME	TRACE START	TRACE END	RAM MONITOR
事件 E0	○	○	○	×	×	○	×	×
事件 E1	○	○	×	×	○	×	×	×
事件 E2	○	○	×	×	○	×	×	×
事件 E3	○	○	×	×	○	×	×	×
事件 E4	○	○	×	○	×	×	○	×
事件 E5	○	○	×	×	○	×	×	○
事件 E6	○	○	×	×	○	×	×	×
事件 E7	○	○	×	×	○	×	×	×

7.6.3 组合条件的选择

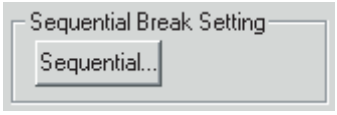
选择事件的组合条件。在事件设置窗口的 EVENT 栏和 SEQ 栏中进行选择。

- OR 条件的选择
  能在EVENT栏中选择要使用的事件。

- 顺序条件的选择
  能在SEQ栏中选择要设置顺序条件的事件，顺序条件只能在暂停事件的情况下选择。但是，如果给事件E5设置了数据比较暂停事件或者RAM监控事件，就不能给事件E5指定顺序条件。

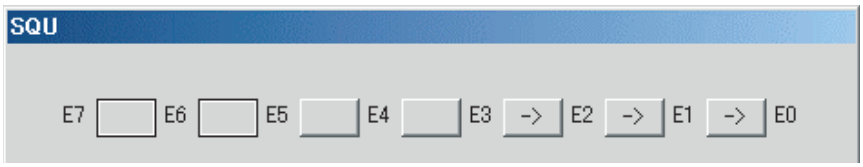
顺序条件	内容
—	指定单独条件。
Ex->Ex-1	指定顺序条件。

也能通过 [Sequential…] （顺序 ...）按钮选择顺序条件。



请单击要指定顺序条件的事件之间的按钮。用箭头显示事件之间的转移。只有对设置了暂停事件的事件才能指定顺序条件。

例）设置 E3→E2→E1→E0 连续发生的事件。



顺序条件能指定 2 ～ 8 点的事件，并且也能设置多对的顺序条件。如果给事件 En 指定了顺序条件，就指定事件 En 和事件 En-1 的顺序。能指定的顺序为 En->En-1->En-2...。例如，能指定 E2->E1->E0 的顺序，但是不能指定 E0->E1->E2 或者 E3->E0 等的顺序。

7.6.4 RAM 监控事件的设置

要设置 RAM 监控事件时，请选定以下的复选框。

将 RAM 监控事件设置为事件 E5。

请注意：在没有设置 RAM 监控事件的情况下， RAM 监控功能不运行。

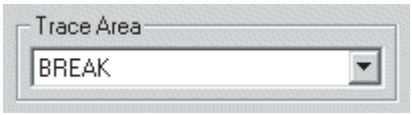
注意事项

- 只有在 Init 对话框的运行模式选项卡中选择了 RAM 监控模式时，才能使用 RAM 监控功能。

7.6.5 跟踪范围的选择

能对跟踪事件选择跟踪范围。

能记录 8M 周期的跟踪测量。



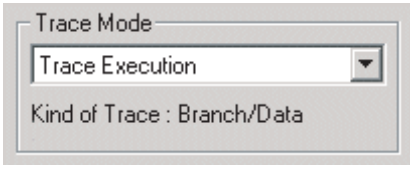
跟踪范围	内容
[Break] （暂停）	目标程序停止前的 8M 周期
[Before] （之前）	跟踪条件成立前的 8M 周期
[After] （之后）	跟踪条件成立后的 8M 周期
[Full] （完整）	跟踪开始后的 8M 周期

注意事项

- 只有在 Init 对话框的运行模式选项卡中选择了跟踪模式时，才能使用跟踪功能。

7.6.6 [Trace Mode]（跟踪模式）的选择

能选择跟踪模式。



模式	内容
[Trace priority]（存储器转储）	存储器转储数据的输出。可能延迟 MCU 的执行。
[MCU execution]（优先 MCU）执行	优先 MCU 的执行。跟踪范围为 512 个周期。

在跟踪种类栏中显示要取的跟踪数据的种类。

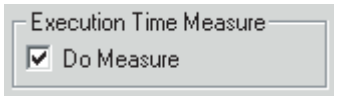
种类	内容
[Branch/Data]（转移 / 数据）	转移跟踪、数据跟踪
[Branch(Compression)]（转移 （压缩））	条件转移跟踪

注意事项

- 只有在 Init 对话框的运行模式选项卡中选择了跟踪模式时，才能使用跟踪功能。

7.6.7 执行时间的测量设置

要测量执行时间时，请选定以下的复选框。



将执行时间测量事件设置为事件 E0 和事件 E4。
请注意：在没有设置执行时间测量事件的情况下，执行时间测量功能不运行。

注意事项

- 只有在 Init 对话框的运行模式选项卡中选择了时间测量模式时，才能使用执行时间测量功能。

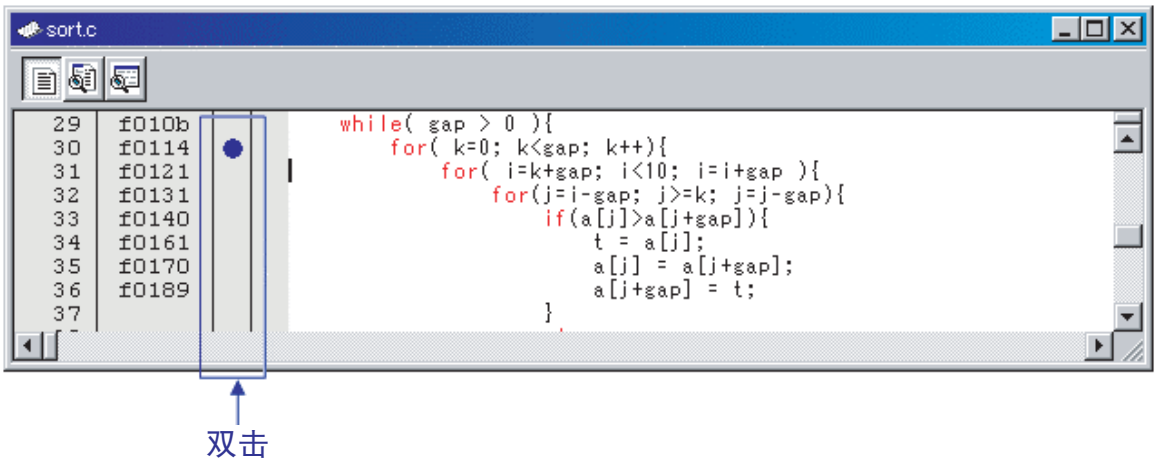
7.6.8 命令按钮

事件设置窗口下部的按钮的含义如下：

按钮名	内容
[Color...]（颜色 ...）	更改事件的显示颜色。
[Reset]（复位）	放弃窗口中显示的内容，加载被设置在仿真器中的内容。
[Save...]（保存 ...）	将窗口中设置的内容保存到文件。
[Load...]（加载 ...）	加载被保存在文件中的事件信息。
[Set]（设置）	将窗口中设置的内容发送给仿真器。
[Close]（关闭）	关闭窗口。

7.6.9 编辑器 （源）窗口的断点设置和解除

请在编辑器 （源）窗口的 H/W 断点设置列上双击要设置断点的行 （在设置行显示蓝色圆点）。



如果再次双击此行，就解除被设置的断点 （蓝色圆点消失）。

根据默认值，在编辑器 （源）窗口中显示用于设置 H/W 断点的列。要设置为 [Hide] （隐藏）时，请在通过菜单 [Edit] （编辑）→[Define Column Format] （定义列格式）打开的对话框中取消 [H/W breakpoints] （H/W 断点）复选框的选定，全部编辑器 （源）窗口的 H/W 断点设置列为隐藏状态。另外，通过选择编辑器 （源）窗口的弹出式菜单 [Columns] （列） →[H/W breakpoints] （H/W 断点），能按各编辑器 （源）窗口设置 H/W 断点的列。

7.6.10 暂停事件的设置

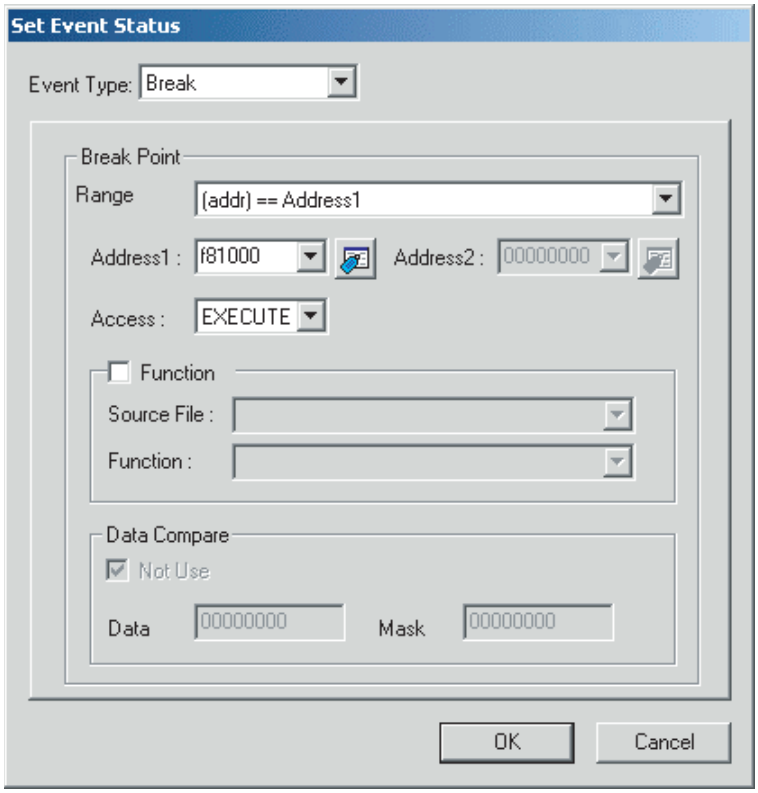
要指定暂停事件时，请在事件选择对话框的事件种类中选择 “Break”（暂停）。
能对暂停事件设置以下的事件：

- 指令的执行
  要指定指令执行的事件时，请在存取条件栏中选择 “EXECUTION”（执行）。在即将执行指定地址的指令前事件成立。
- 存储器的存取
  要指定存储器存取的事件时，请在所用事件的存取条件栏中选择 “READ / WRITE / R/W”（读 / 写 / 读/写）。在指定地址或者指定地址范围的设置条件下存取时事件成立。当指定存储器的存取暂停事件时，能指定比较数据，和指定地址的读写数据进行比较，而且也能屏蔽比较数据。

7.6.10.1 指定地址的指令执行

设置如下：

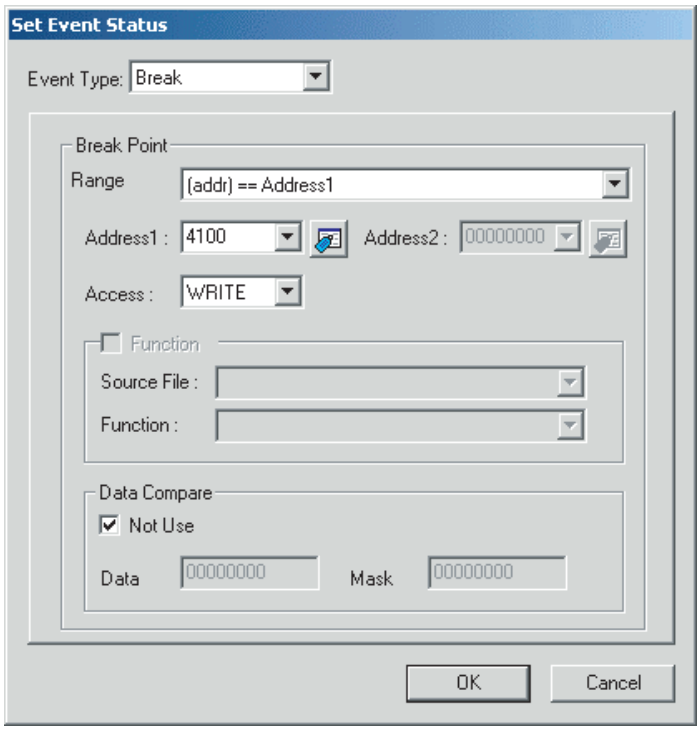
例）执行地址 F81000h 的指令。



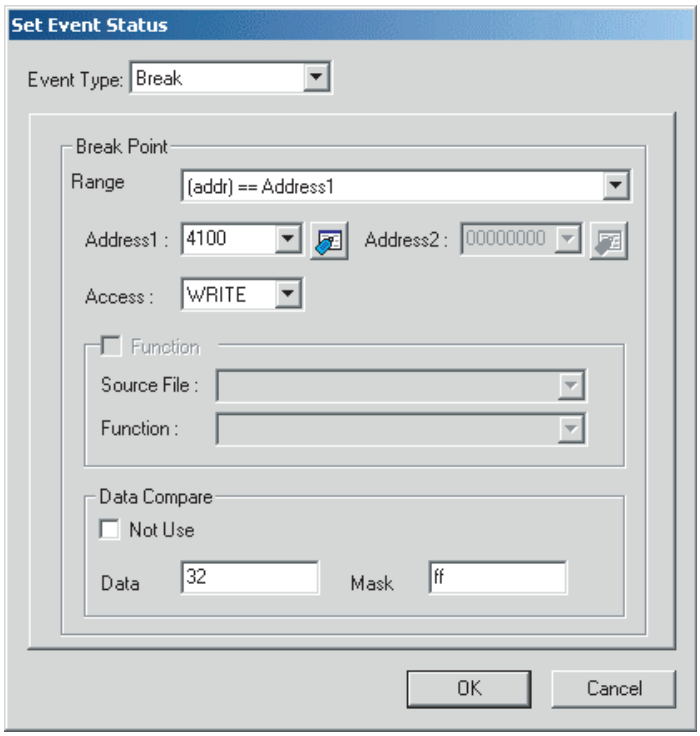
7.6.10.2 指定地址的读写数据

设置如下：

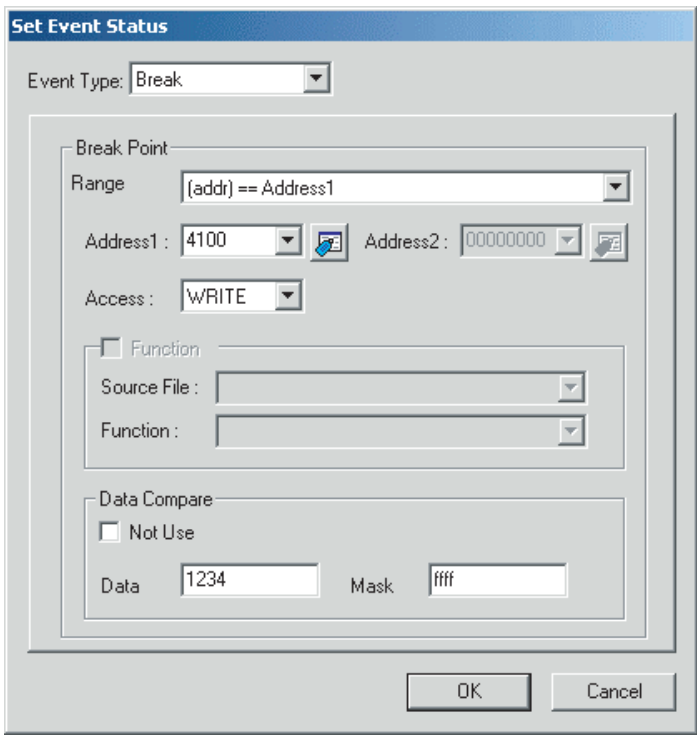
例）给地址 4100h 写数据。



例）给地址 4100h 写字节数据 32h。



例）给地址 4100h 写字数据 1234h。



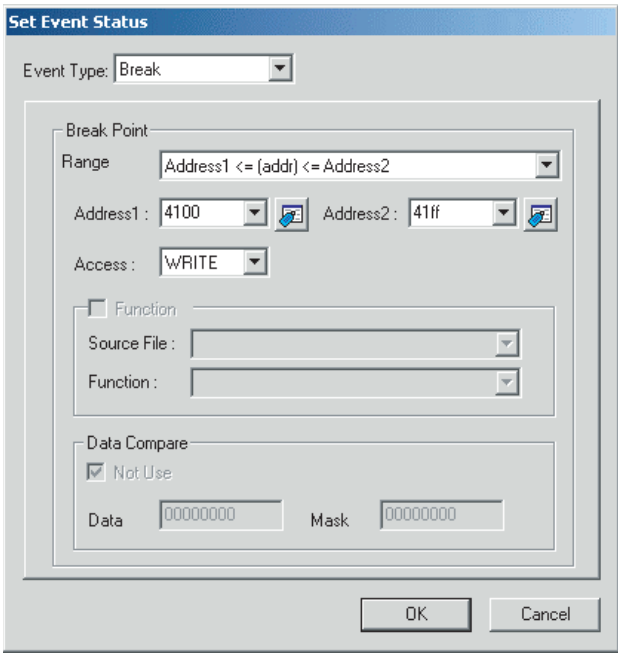
注意事项

- 只能在 Init 对话框的运行模式选项卡中选择了跟踪模式时，才能给事件 E5 指定数据比较。
- 在暂停事件中，当通过无数据比较的暂停来使用事件 E5 时，必须给屏蔽值指定 0000h；当通过字节数据的数据比较来使用时，必须给屏蔽值指定 00FFh。也能给数据比较指定从奇数地址开始的字数据。
- 如果设置数据比较，就会失去执行用户程序的实时性。

7.6.10.3 指定地址范围的读写数据

设置如下：

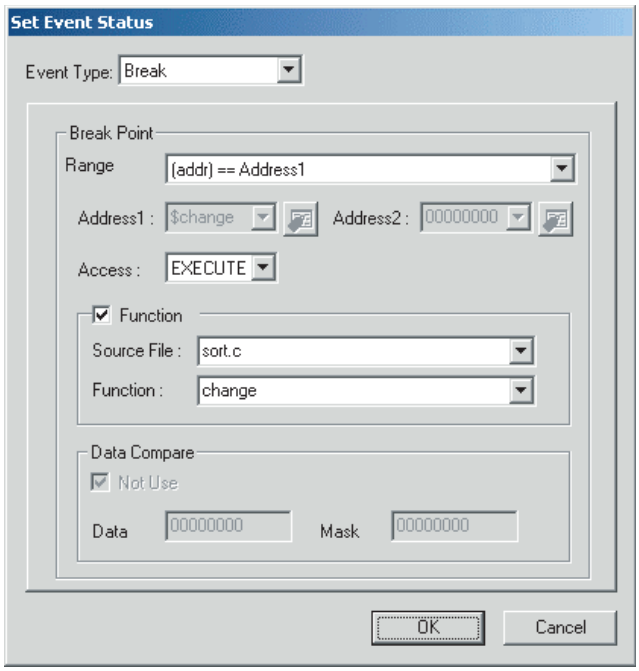
例）给地址 4100h ～ 41FFh 写数据。



7.6.10.4 指定函数的进入

设置如下：

例）进入函数名 change。



7.6.11 事件组合条件的设置

暂停事件能进行以下的组合 （能指定的暂停事件数为 8 点，跟踪事件和时间测量事件共用）。

- 某个事件成立 （OR 条件）。
- 按指定的顺序事件成立 （顺序条件）。

7.6.11.1 OR 的指定

请在事件指定栏中选定要使用的事件。

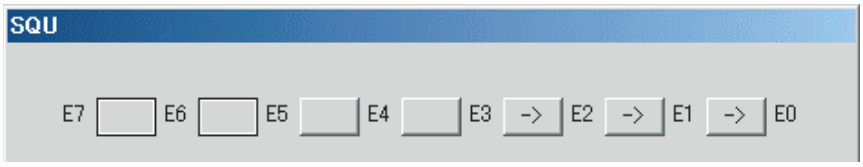
7.6.11.2 顺序的指定

使用 [Seaquential...] （顺序 ...）按钮。

请单击要指定顺序条件的事件之间的按钮。用箭头显示事件之间的转移。只有对设置了暂停事件的事件才能指定顺序条件。但是，如果给事件 E5 设置了数据比较暂停事件，就不能给事件 E5 指定顺序条件。

顺序条件能指定 2 ～ 8 点的事件，并且也能设置多对的顺序条件。
如果给事件 En 指定了顺序条件，就指定事件 En 和事件 En-1 的顺序。
能指定的顺序为 En->En-1->En-2...。例如，能指定 E2->E1->E0 的顺序，但是不能指定 E0->E1->E2 或者 E3->E0 等的顺序。

例）设置 E3→E2→E1→E0 连续发生的事件。



例）设置 E3→E2 或者 E7→E6→E5 连续发生的事件。

7.6.12 跟踪事件的设置

要指定跟踪事件时，请在事件选择对话框的事件种类中选择 [Trace]（跟踪）。

跟踪事件能指定记录转移地址信息的转移跟踪事件以及记录数据存取信息的数据跟踪事件。通过跟踪条件指定是设置哪个跟踪事件。

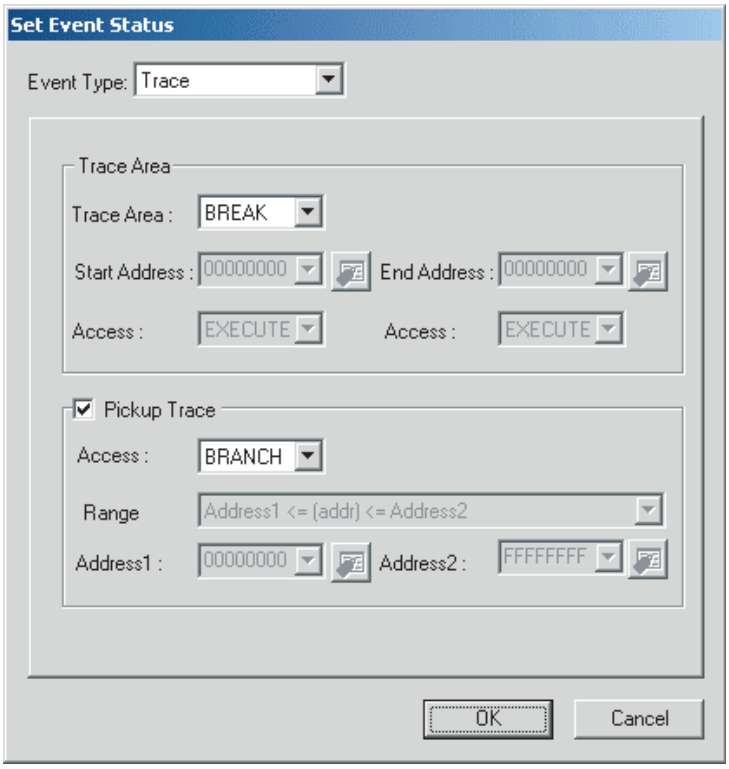
能设置转移跟踪事件和数据跟踪事件，记录混合信息。

在没有设置跟踪事件的情况下，跟踪功能为条件转移跟踪。条件转移跟踪压缩记录条件转移指令的信息。因此，条件转移跟踪能比转移跟踪多记录执行履历。

在跟踪条件中能设置的条件如下：

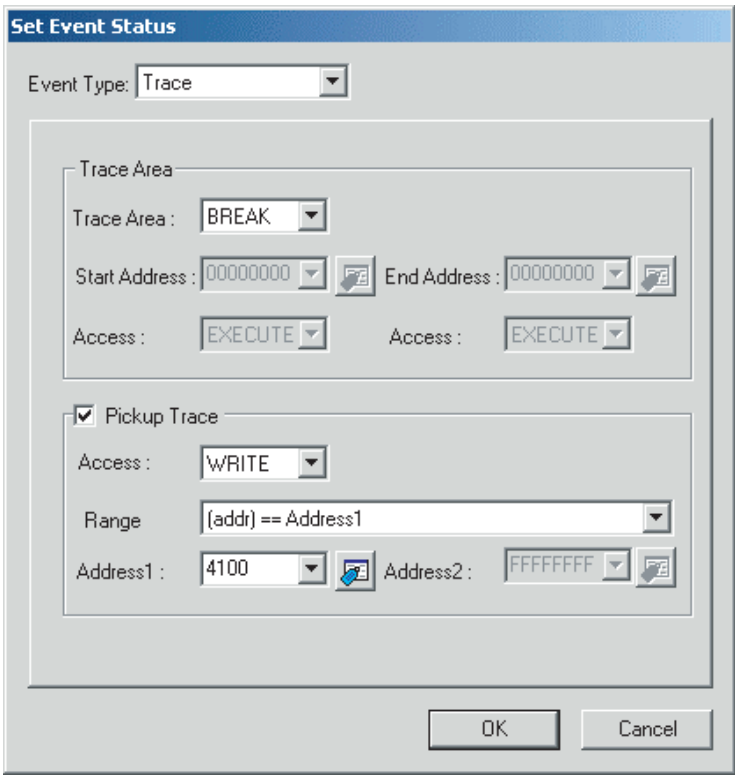
- 转移跟踪事件的设置
  记录转移目标和转移源的信息。要指定转移跟踪事件时，请在所用事件的存取条件栏中选择“BRANCH”（转移）。转移跟踪事件只能指定为记录全地址范围的转移地址信息。

例）记录全地址范围的转移地址信息。

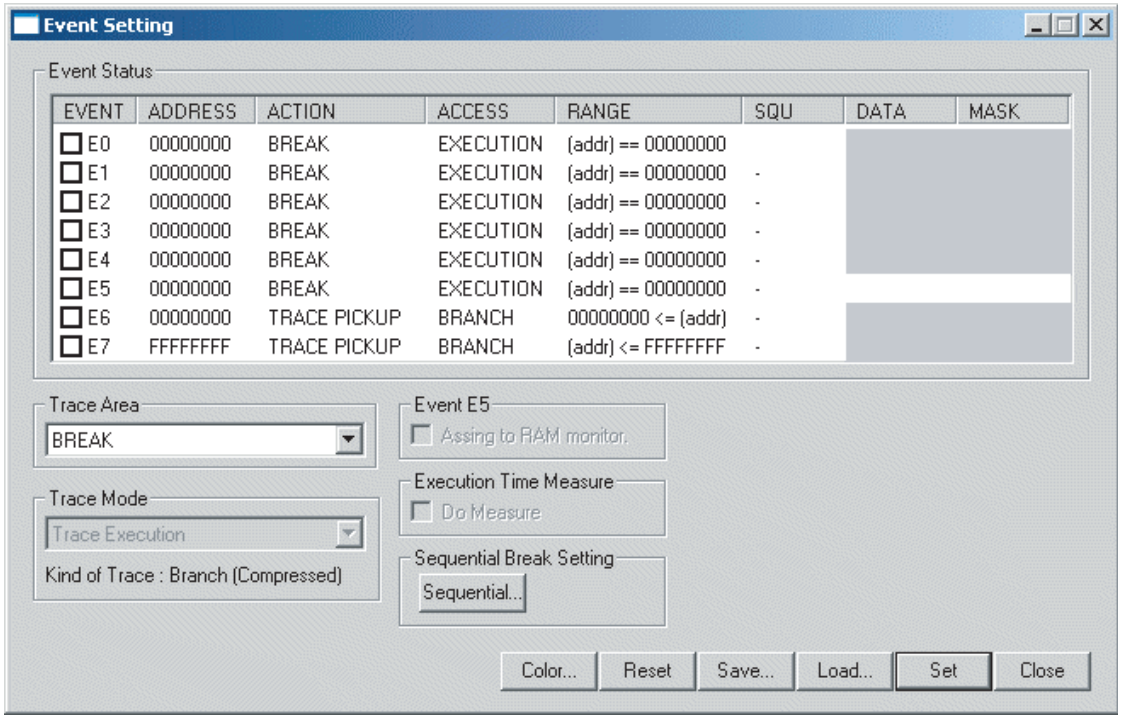


- 数据跟踪事件的设置
  记录指定地址的存取信息。要指定数据跟踪事件时，请在所用事件的存取条件栏中选择“READ / WRITE / RW”（读 / 写 / 读写）。在指定地址或者指定地址范围的设置条件下存取时事件成立。
  设置如下：

例）记录地址 4100h 的写数据。



- 条件转移跟踪的设置
  记录全地址范围的转移地址信息。
  必须解除全部跟踪事件。



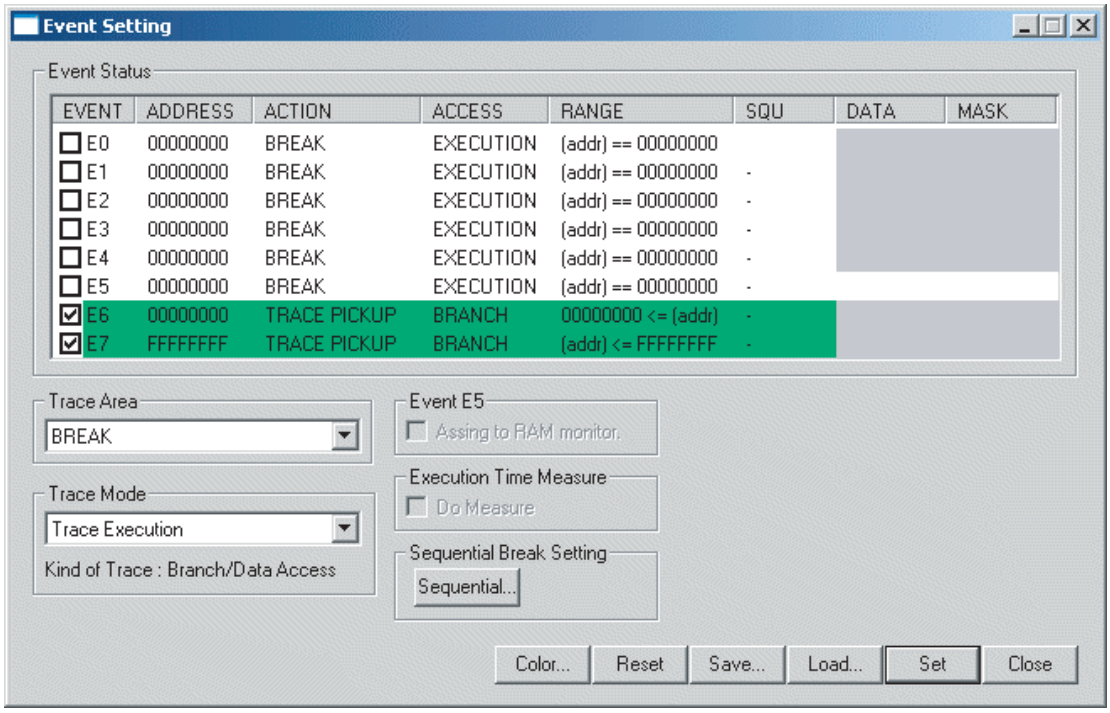
7.6.12.1 跟踪范围的指定以及跟踪事件的设置

要设置跟踪事件时，请指定跟踪范围。事件的设置条件因跟踪范围指定的模式而不同。
能指定的跟踪范围和事件的设置条件如下：

- 在跟踪范围中指定 “Break”（暂停）或者 “Full”（完整），设置跟踪事件。
  请指定任意地址或者地址范围，设置跟踪事件。

  作为默认值，给跟踪范围设置了 “Break”（暂停），并且作为跟踪事件，将记录全地址范围的转移地址信息的事件设置为E6和E7。

  默认值的设置：

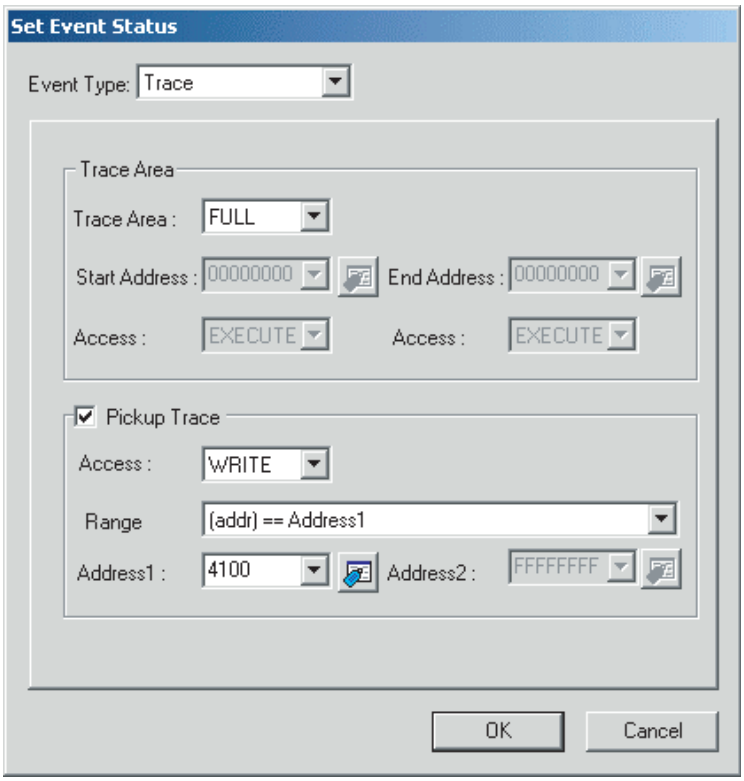


请在所用事件的ACCESS栏中指定跟踪条件 （转移跟踪/数据跟踪）。要指定地址范围时，使用事件的范围指定。能指定范围的事件组合如下：

- E0和E1的组合
- E2和E3的组合
- E4和E5的组合
- E6和E7的组合

要通过任意指定的条件进行跟踪时，请更改上述的设置。
设置如下：

例）记录Full模式中的地址4100h、 4110h和4120h的写数据。



请再次打开事件设置对话框，更改上述Address1栏的地址，以添加方式注册到4110h和4120h的写数据设置。

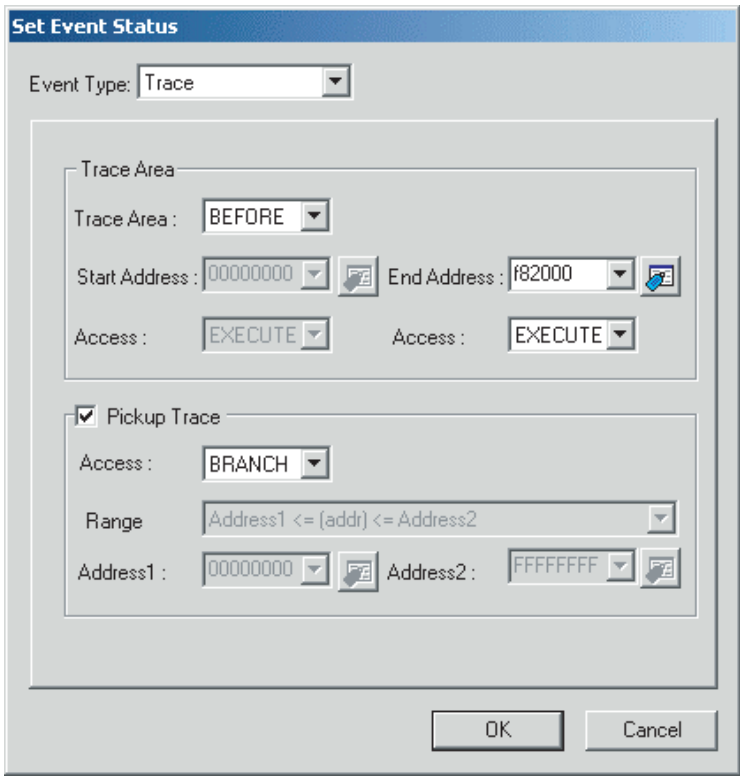
- 在跟踪范围中指定Before，设置跟踪事件。
  请设置跟踪开始事件 （只能指定事件E0）和跟踪结束事件 （只能指定事件E4），然后指定任意地址或者地址范围，设置跟踪事件。

  在Before模式中，记录从跟踪开始事件 （程序开始执行）到跟踪结束事件成立为止所发生的跟踪事件。对于跟踪开始事件，将用于Before模式的跟踪开始事件 （给地址指定当前的PC值，给存取条件指定EXECUTE）设置为事件E0。对于跟踪结束事件，给事件E4设置任意地址和存取条件作为跟踪结束条件。

  请在所用事件的存取条件栏中指定跟踪事件的跟踪条件 （转移跟踪/数据跟踪）。要指定地址范围时，使用事件的范围指定。因为跟踪开始事件和跟踪结束事件分别使用E0事件和E4事件，所以能指定范围的事件组合如下：
  - E2和E3的组合
  - E6和E7的组合

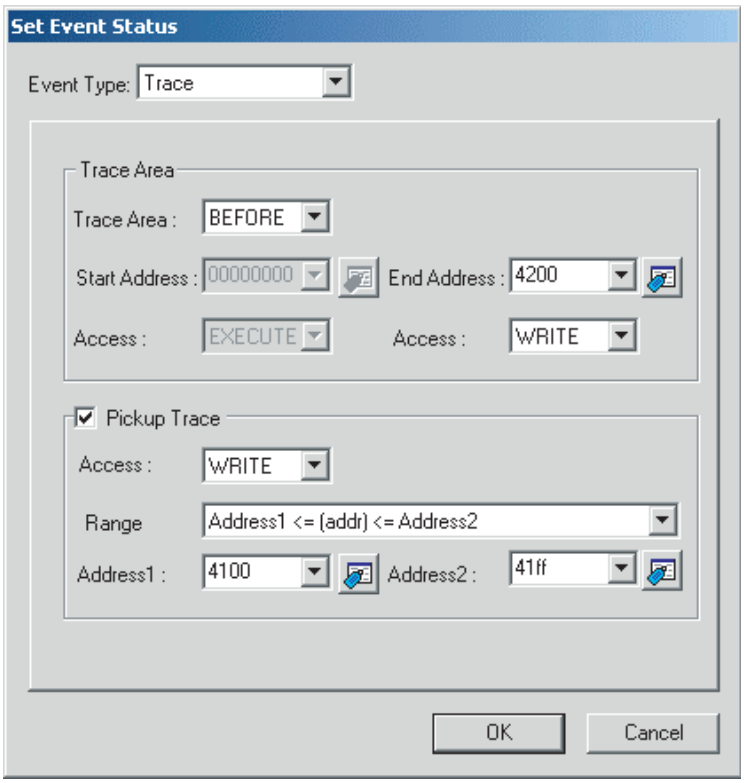
  根据默认值，将作为跟踪事件的测量全地址范围的转移地址信息的事件设置为E6和E7。
  设置如下：

例）记录Before模式中的全地址范围的转移地址信息，指定F82000h的执行指令作为跟踪结束条件。



要通过任意指定的地址范围进行跟踪时，请更改上述指定范围的事件设置。
设置如下：

例）记录Before模式的地址4100h～41FFh的写数据，指定4200h的写数据作为跟踪结束条件。



- 在跟踪范围中指定After，设置跟踪事件。
  请设置跟踪开始事件 （只能指定事件E0）和跟踪结束事件 （只能指定事件E4），然后指定任意地址或者地址范围，设置跟踪事件。

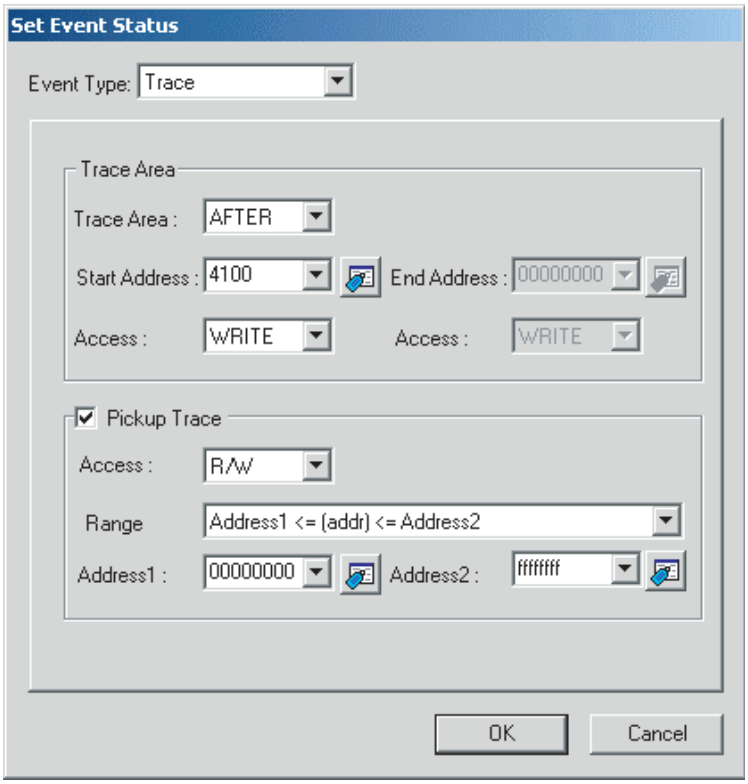
  在After模式中，记录从跟踪开始事件成立到程序停止执行为止所发生的跟踪事件。

  对于跟踪开始事件，请给事件E0设置任意地址和存取条件作为跟踪开始条件。对于跟踪结束事件，将用于After模式的跟踪结束事件 （作为不发生的条件，给地址指定0h，给存取条件指定WRITE）设置为事件E4。

  请在所用事件的存取条件栏中指定跟踪事件的跟踪条件 （转移跟踪/数据跟踪）。要指定地址范围时，使用事件的范围指定。因为跟踪开始事件和跟踪结束事件分别使用E0事件和E4事件，所以能指定范围的事件组合如下：
  - E2和E3的组合
  - E6和E7的组合

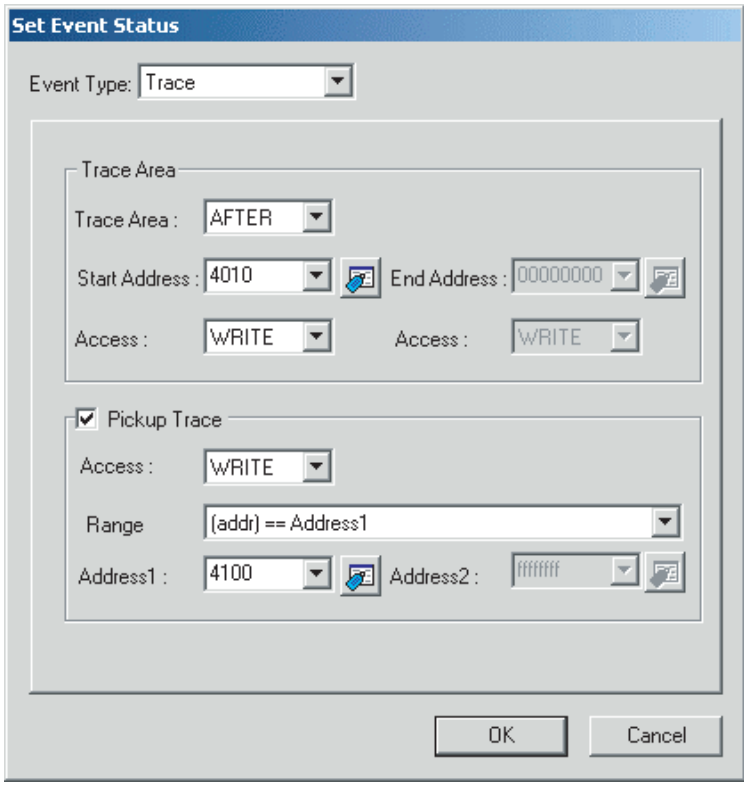
  根据默认值，将作为跟踪事件的测量全地址范围的转移地址信息的事件设置为E6和E7。
  设置如下：

  例）记录After模式中的全地址范围的存取数据 （读写数据），指定4100h的写数据作为跟踪开始条件。



  要通过任意指定的地址范围进行跟踪时，请更改上述指定范围的事件设置。
  设置如下：

例）记录After模式中的地址4100h、 4110h和4120h的写数据，指定4010h的写数据作为跟踪开始条件。



请再次打开事件设置对话框，更改上述Address1栏的地址，以添加方式注册到4110h和4120h的写数据设置。

注意事项

- 在 Before 模式和 After 模式中，记录跟踪范围内发生的跟踪事件以及此期间发生的跟踪开始事件，但是不记录跟踪结束事件。
- 对于条件转移跟踪，在显示跟踪窗口的反汇编模式和源模式时可能需要时间。
- 如果在跟踪模式中选择了 MCU 执行优先模式，跟踪范围就为 512 个周期。要将跟踪范围置为 8M 周期时，请在跟踪模式中选择跟踪优先模式。在跟踪优先模式中，优先输出跟踪数据并延迟 MCU 的执行。和 MCU 执行优先模式的情况相比，跟踪优先模式的程序处理时间较长。另外，在条件转移跟踪的情况下，跟踪模式变为跟踪优先模式。
- 只有在 Init 对话框的运行模式选项卡中选择了跟踪模式时，才能使用跟踪功能。

7.6.13 时间测量事件的设置

要设置时间测量事件时，请双击事件设置栏。打开事件选择对话框，请在事件选择对话框的事件种类中选择 “Time Measurement”（时间测量）。

时间测量有以下的测量功能：

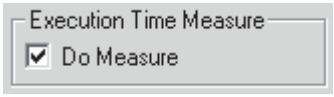
- 执行时间的测量
  测量程序开始执行到停止的执行时间。

- 间隔时间的测量
  测量开始事件成立到结束事件成立的间隔时间。
  能给间隔时间测量事件设置以下的事件：
  — 存储器的存取
  要指定存储器存取的事件时，请在存取条件栏中选择 “READ / WRITE / R/W”。在指定地址或者指定地址范围的设置条件下存取时事件成立。

  请在间隔时间测量窗口的测量条件中设置测量事件 （测量开始事件和测量结束事件）。

7.6.13.1 执行时间测量事件的设置

- 测量程序开始执行到停止的执行时间。
  请选定以下的复选框。

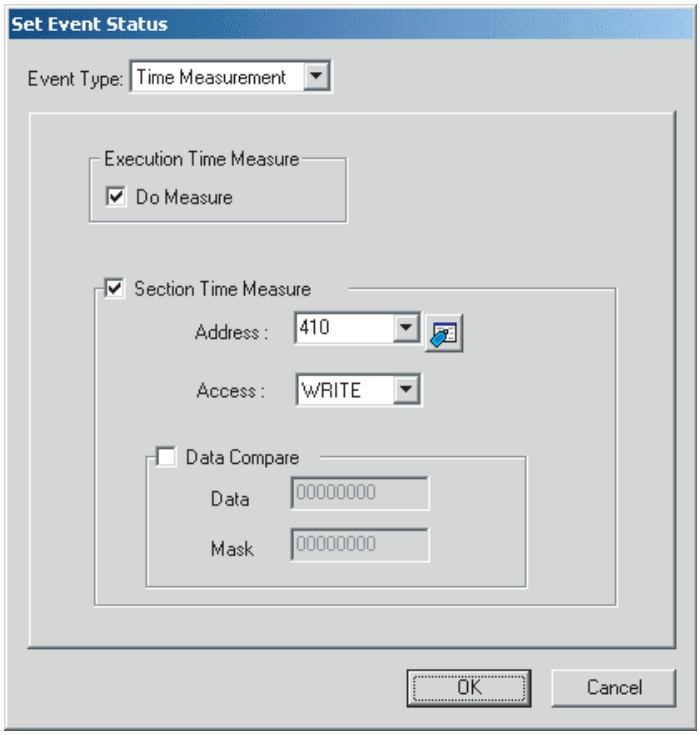


将执行时间测量事件设置为事件E0和事件E4。
请注意：在没有设置执行时间测量事件的情况下，执行时间测量功能不运行。

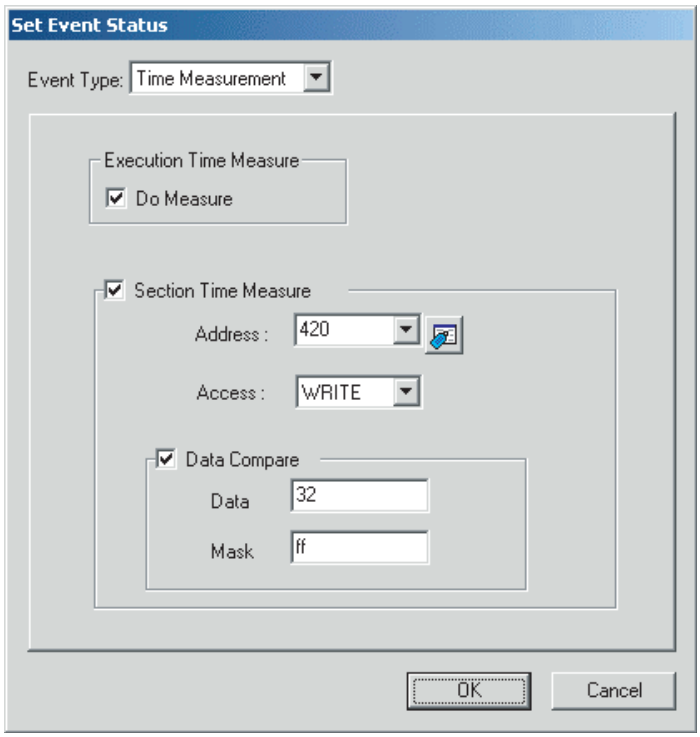
7.6.13.2 间隔时间测量事件的设置

- 给指定地址读写数据。
  设置如下：

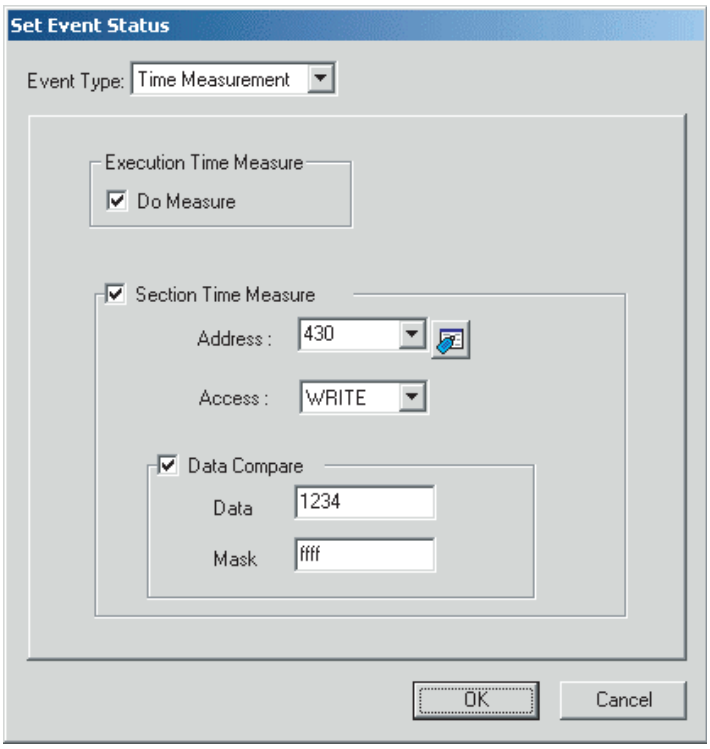
  例）给地址410h写数据。



  例）给地址420h写字节数据32h。



例）给地址430h写字数据1234h。



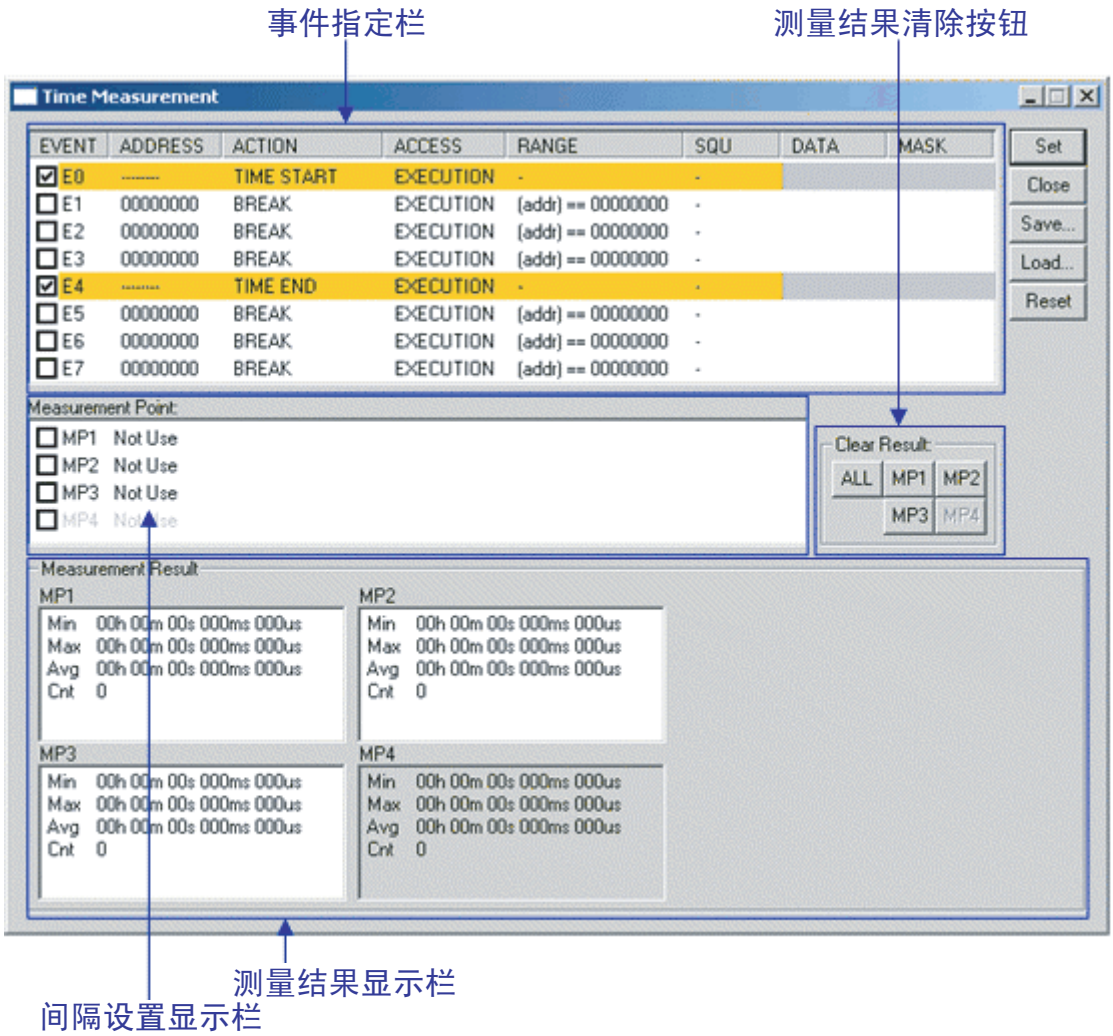
注意事项

- 只有在 Init 对话框的运行模式选项卡中选择了时间测量模式时，才能使用时间测量功能。

7.7 间隔时间测量窗口

间隔时间测量窗口是显示任意间隔的最小 / 最大 / 平均执行时间以及测量次数的窗口。最多能同时测量 3 点的间隔时间。

能通过在事件设置窗口中设置事件的相同方法，指定测量条件的事件。



- 如果更改事件的内容，就在标题栏显示 “*”。对仿真器进行设置后不显示 “*”。

注意事项

- 事件设置窗口和间隔时间测量窗口使用仿真器相同的资源。如果在间隔时间测量窗口中更改事件，事件设置窗口中的设置内容也被更改。

7.7.1 测量事件的指定

能指定以下的事件作为测量事件：

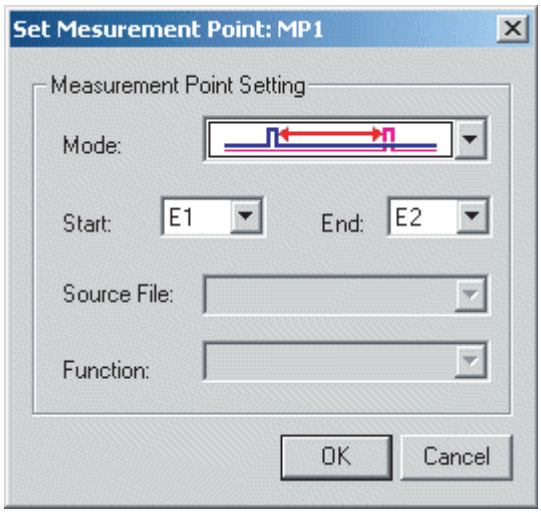
- 存储器的存取

要设置事件时，双击间隔时间测量窗口的事件指定栏，打开事件设置对话框。

有关间隔时间测量事件的设置方法，请参照 “7.6.13 时间测量事件的设置”。

7.7.2 间隔时间的测量条件

能按测量间隔指定以下的间隔时间的测量条件：



	测量间隔开始事件成立到间隔结束事件成立的时间。

7.7.3 命令按钮

窗口中的各按钮的含义如下：

按钮名	功能
[Reset] （复位）	放弃窗口中显示的内容，加载被设置在仿真器中的内容。
[Save...] （保存 ...）	将窗口中设置的内容保存到文件。
[Load...] （加载 ...）	加载被保存在文件中的事件信息。
[Set] （设置）	将窗口中设置的内容发送给仿真器。
[Close] （关闭）	关闭窗口。

7.7.4 测量条件的设置

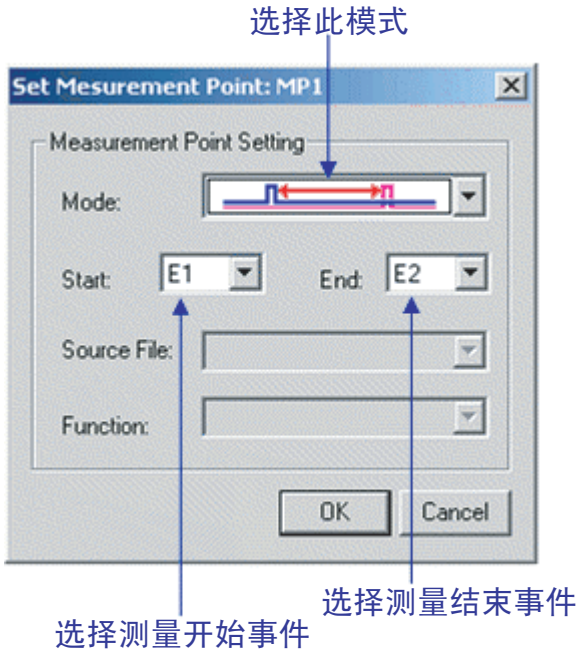
本调试器能指定以下的测量条件：

- 测量事件之间的执行时间。

能指定 3 个测量间隔。要指定测量间隔时，请单击间隔时间测量窗口的 [Measurement Point] （测量点）群的任意行 （MP1 ～ MP3），打开测量条件指定对话框。

7.7.4.1 事件之间的执行时间测量

1. 请设置测量事件 （测量开始事件和测量结束事件）。
2. 请在测量条件指定对话框中进行以下的指定。



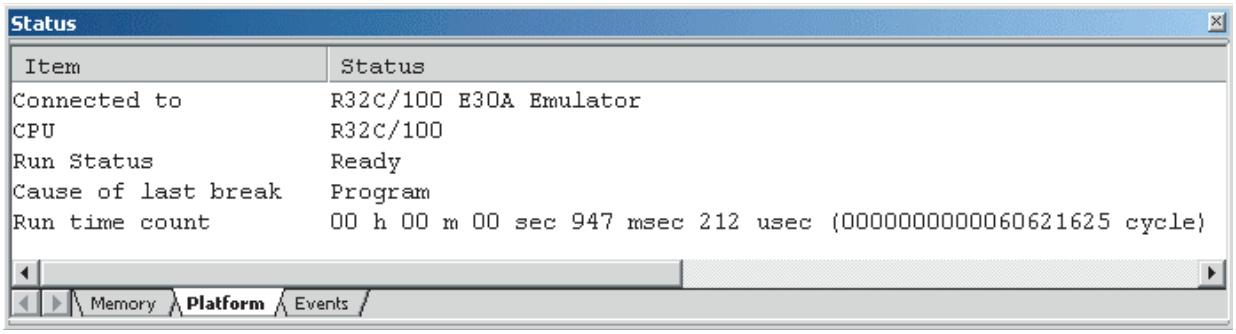
注意事项

- 时间测量间隔是指开始事件成立到结束事件成立的间隔，但是不包括结束事件的成立。

7.7.5 时间测量结果的参照

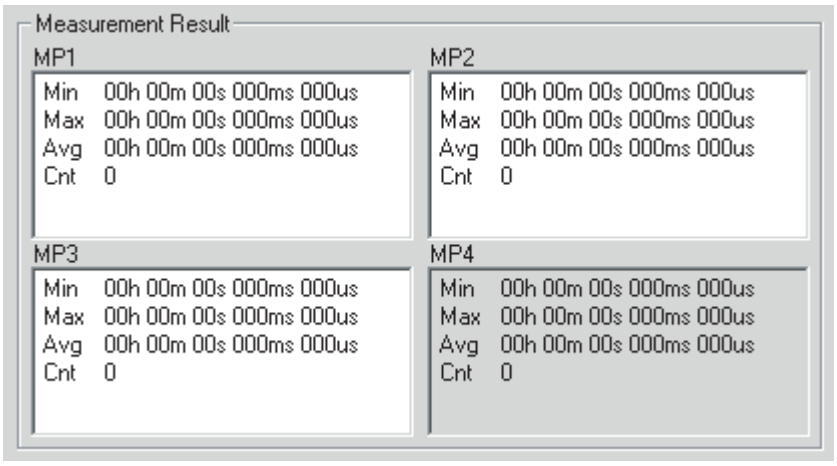
7.7.5.1 执行时间测量结果的参照

在状态窗口的 [Platform]（平台）选项卡中显示执行时间的测量结果。在多次测量的情况下，显示累积结果。



7.7.5.2 间隔时间测量结果的参照

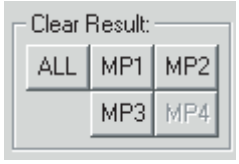
在间隔时间测量窗口的下部显示间隔时间的测量结果。



[Min]（最小）	最小时间
[Max]（最大）	最大时间
[Avg]（平均）	平均时间
[Cnt]（次数）	测量次数

7.7.5.3 间隔时间测量结果的清除

单击要清除的测量结果的按钮。



通过单击 [ALL]（全部）按钮，能清除全部测量结果。

7.8 跟踪窗口

跟踪窗口是显示实时跟踪测量结果的窗口。能通过以下的显示格式进行显示：

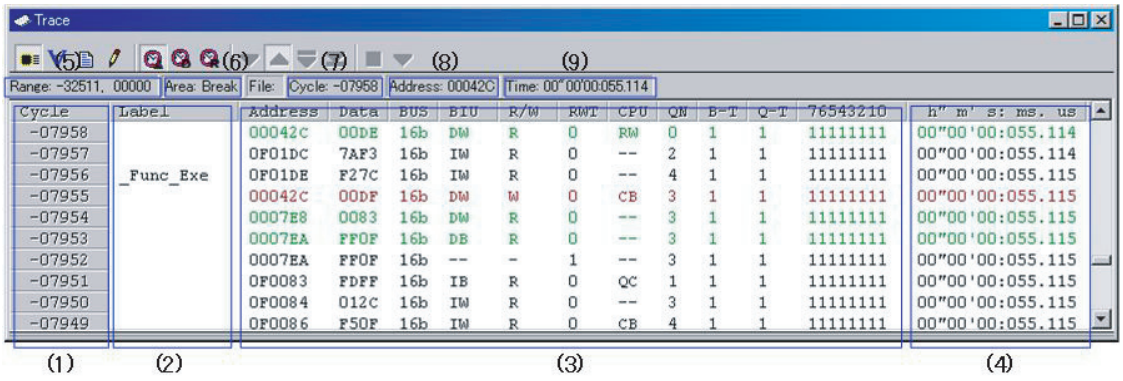
- “Bus mode”（总线模式）
  能参照各周期的总线信息。显示的内容取决于所使用的MCU和仿真器系统。除了总线信息以外，能混合显示反汇编信息、源行信息和数据存取信息。
- “Disassemble mode”（汇编模式）
  能参照被执行的指令。除了反汇编信息以外，能混合显示源行信息和数据存取信息。
- “Data access mode”（数据存取模式）
  能参照数据的R/W周期。除了数据存取信息以外，能混合显示源行信息。
- “Source mode”（源模式）
  能在源程序上参照程序的执行路径。

在结束跟踪测量时显示测量结果。如果重新开始跟踪测量，就清除窗口的显示内容。

能在跟踪点设置窗口中更改跟踪测量范围。有关跟踪点设置窗口的详细内容，请参照 “7.6 事件设置窗口”。作为初始状态，记录程序刚要停止执行前的信息。

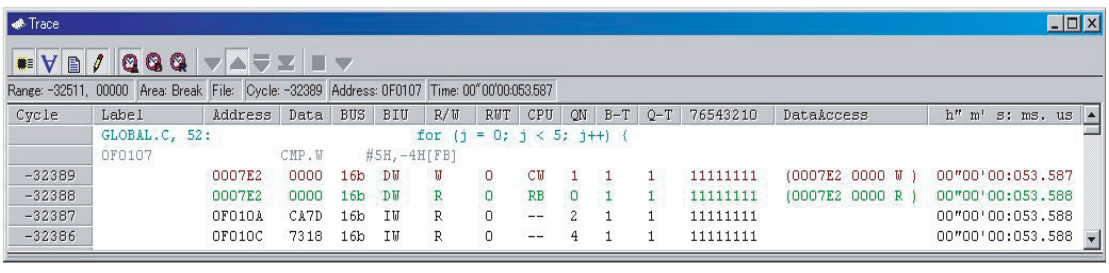
7.8.1 总线模式的构成

当选择了总线模式时，就以总线模式显示跟踪信息。总线模式的构成如下。
总线模式的显示内容因使用的 MCU 和仿真器系统而不同。



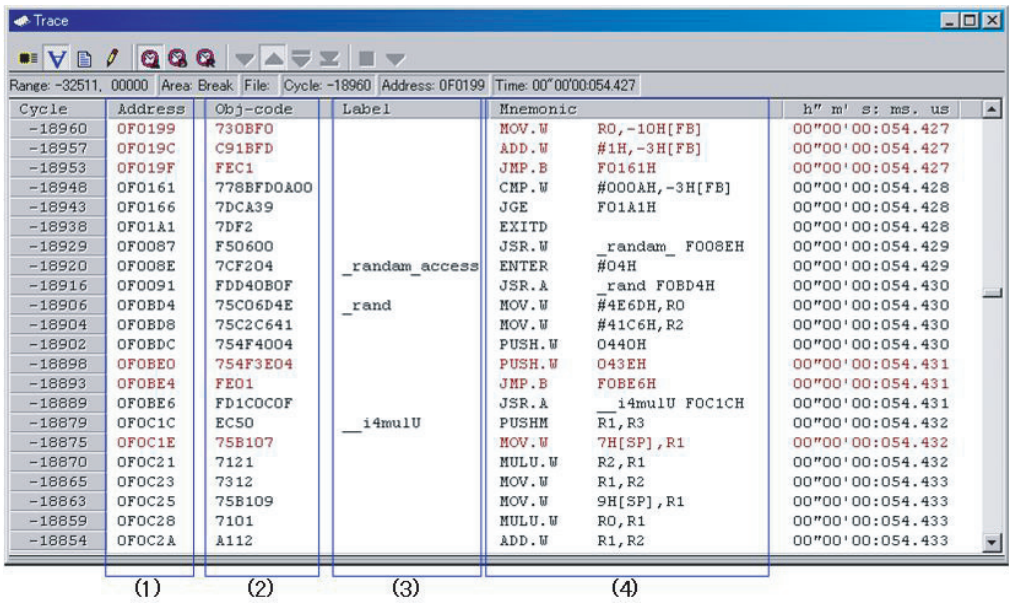
1. 周期显示栏：
   显示跟踪周期。如果双击，就显示用于更改显示周期的对话框。
2. 标签显示栏：
   显示对应地址总线信息的标签。如果双击，就显示用于检索地址的对话框。
3. 总线信息显示栏：
   显示内容因使用的MCU和仿真器系统而不同。
   - 请参照 “7.8.6 用于R32C的调试器的总线信息显示”。
4. 时间信息显示栏：（本产品不支持）
   显示跟踪测量结果的时间信息。能从菜单中选择以下3种方法：
   - “Absolute Time”（绝对时间）：用绝对时间显示程序开始执行时的经过时间 （默认值）。
   - “Differences”（差异）：显示前一个周期的差分时间。
   - “Relative Time”（相对时间）：显示所选周期的相对时间。如果跟踪测量结果有更新，就更改绝对时间的显示。
5. 取到的跟踪测量结果的范围：
   显示现在取到的跟踪测量结果的范围。
6. 跟踪测量的范围：
   显示现在已设置的跟踪测量的范围。
7. 起始行的周期：
   显示起始行的周期。
8. 起始行的地址：
   显示起始行的地址。
9. 起始行的时间：
   显示起始行的时间。
10. 窗口拆分框：
    如果双击，窗口就进行拆分显示。

除了总线信息以外，能混合显示反汇编信息、源行信息和数据存取信息。显示如下：



7.8.2 [Disassemble Mode]（反汇编模式）的构成

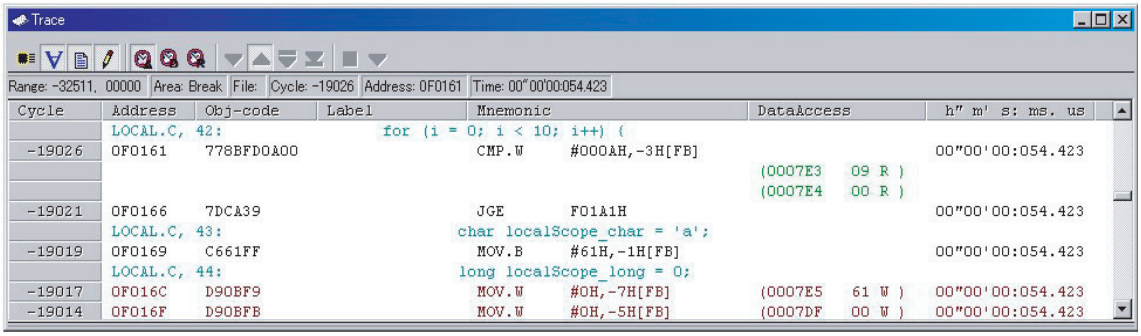
当选择的不是总线模式而是反汇编模式时，就以反汇编模式显示跟踪信息。反汇编模式的构成如下：



1. 地址显示栏：
   显示对应指令的地址。如果双击，就显示用于检索地址的对话框。
2. 目标码显示栏：
   显示指令的目标码。
3. 标签显示栏：
   显示对应指令地址的标签。如果双击，就显示用于检索地址的对话框。
4. 助记符显示栏：
   显示指令的助记符。

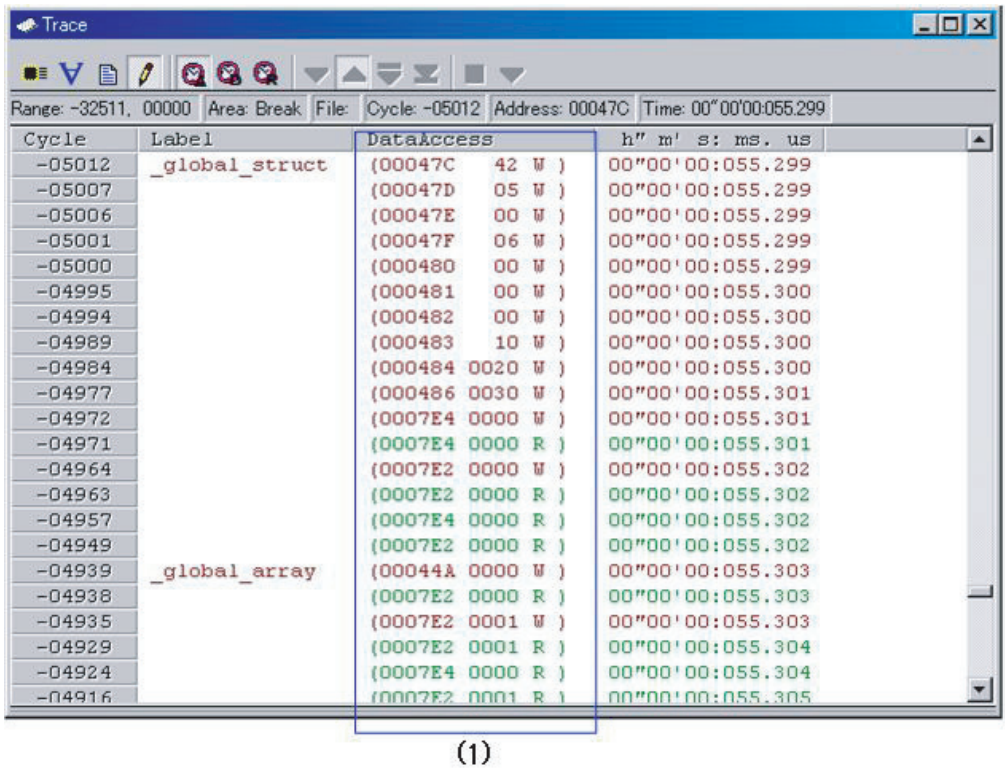
其他的显示内容和总线模式相同。

除了反汇编信息以外，能混合显示源行信息和数据存取信息。显示如下：



7.8.3 [Data Access Mode]（数据存取模式）的构成

当选择的不是总线模式和反汇编模式而是数据存取模式时，就以数据存取模式显示跟踪信息。数据存取模式的构成如下：

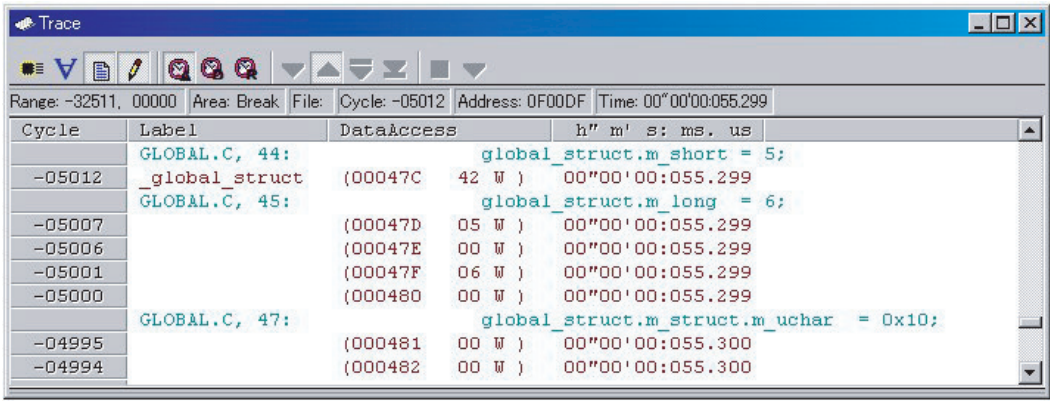


(1)

1. 数据存取显示栏：
   显示数据存取信息。如果显示 “(000400 1234 W)”，就表示以2字节长度给地址000400H写了数据1234H。

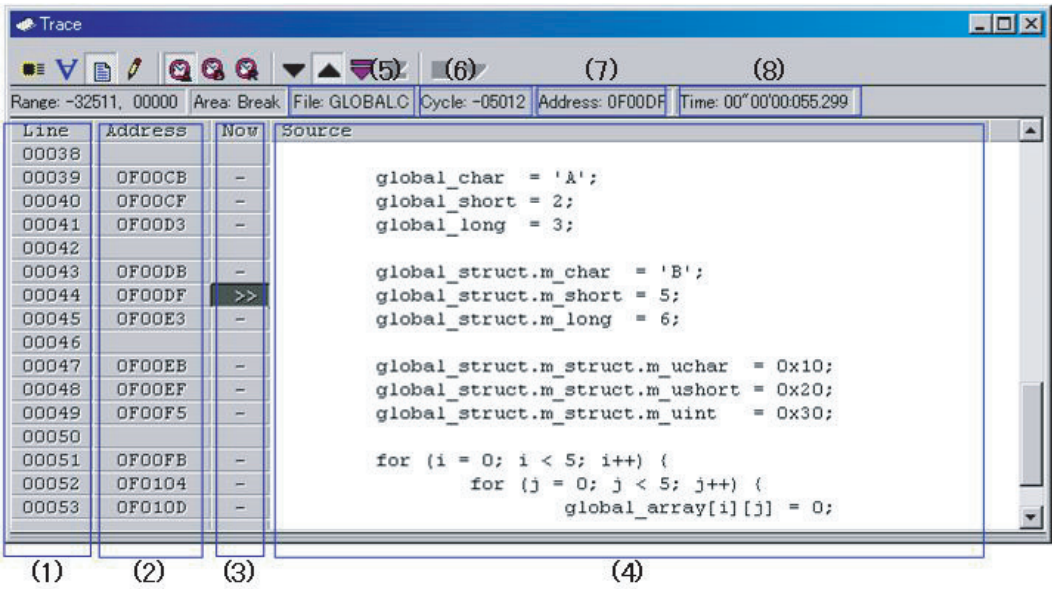
其他的显示内容和总线模式相同。

除了数据存取信息以外，能混合显示源行信息。显示如下：



7.8.4 [Source Mode]（源模式）的构成

当只选择了源模式时，就以源模式显示跟踪信息。源模式的构成如下：



1. 行号显示栏：
   显示文件的行号信息。如果双击，就显示用于更改显示文件的对话框。
2. 地址显示栏：
   显示对应源行的地址。如果双击，就显示用于检索地址的对话框。
3. 参照周期显示栏：
   对于现在正在参照的周期，显示 “>>”。当有对应源行的地址时，显示 “-”。
4. 源显示栏：
   显示源文件。
5. 文件名：
   显示现在正在显示的源文件名。
6. 参照周期：
   显示现在正在参照的周期。
7. 参照地址：
   显示与现在正在参照的周期对应的地址。
8. 参照时间：
   显示与现在正在参照的周期对应的时间。

其他的显示内容和总线模式相同。

7.8.5 选项菜单

如果在窗口中单击鼠标的右键，就显示以下的弹出式菜单。在工具栏的按钮中也分配了这些菜单的主要功能。

菜单名		功能
[BUS]		显示总线 （BUS）信息。
[DIS]		显示反汇编 （DIS）信息。
[SRC]		显示源 （SRC）信息。
[DATA]		显示数据存取 （DATA）信息。
[View] （视图）	[Cycle...] （周期 ...）	指定显示周期。
	[Address...] （地址 ...）	检索地址。
	[Source...] （源 ...）	选择要显示的源文件。
Time （时间显示）	[Absolute Time] （绝对时间）	用执行开始的绝对时间显示时戳。
	[Differences] （差异）	用前一个周期的差异时间显示时戳。
	[Relative Time] （相对时间）	用指定周期开始的相对时间显示时戳。
Trace （跟踪操作）	[Forward] （正向）	检索方向为正向。
	[Backward] （反向）	检索方向为反向。
	[Step] （单步）	向指定的方向单步执行。
	[Come] （指定位置）	以 “Come”（指定位置）检索指定行的执行周期。
	[Stop] （停止）	停止跟踪测量，显示结果。
	[Restart] （再启动）	重新测量跟踪数据。
[Layout...] （布局 ...）		选择显示列。
[Copy] （复制）		将选择的行复制到剪贴板。
[Save...] （保存 ...）		将跟踪数据保存到文件。
[Load...] （加载 ...）		从文件中读跟踪数据。
[Toolbar display] （工具栏显示）		切换工具栏的显示或者隐藏。
[Customize toolbar...] （自定义工具栏 ...）		自定义工具栏。
[Allow Docking] （允许入坞）		允许窗口入坞。
[Hide] （隐藏）		隐藏窗口。

7.8.6 用于 R32C 的调试器的总线信息显示

从左到右，含义如下：

- Src, Dest

  表示地址总线的状态。

- Data

  表示数据总线的状态。

- Size

  表示数据存取的长度。

显示格式	长度
B	8 位
W	16 位
L	32 位
Q	64 位

- Status

  表示存储器和I/O之间的状态。

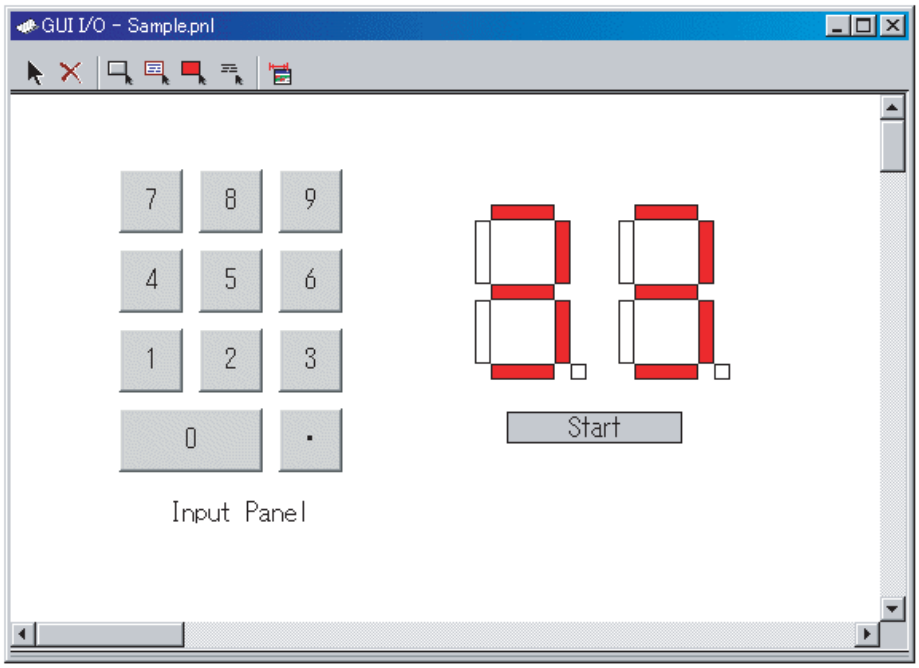
显示格式	状态	原因
JMP	调用 / 返回	记录为转移信息。 将转移源地址输出到 Src，将转移目标地址输出到 Dest。
RD	读	记录为数据存取信息。
WR	写	记录为数据存取信息。

- Jcnd

  表示条件转移指令的状态 （“0”成立或者 “1”不成立）。

7.9 [GUI I/O]（GUI 输入 / 输出）窗口

GUI 输入 / 输出窗口是能创建虚拟输入 / 输出面板的窗口。在窗口中配置了虚拟的按钮和 LED，能进行输入和输出操作。



- 能在窗口中配置以下的项目：
  - 标签
    在给指定地址（或者位）写指定值时，显示或者隐藏字符串。
  - LED
    在给指定地址（或者位）写指定值时，用指定的颜色显示（替代LED点灯）。
  - 按钮
    通过单击按钮，进行虚拟端口的输入。
  - 文本
    显示文本字符串。
- 能将创建的输入/输出面板保存到文件（输入/输出面板文件），并且能从被保存的文件中重新加载创建的输入/输出面板。
- 能给创建的项目设置的地址最多为200点。当各项目设置的地址都不相同时，能配置的项目数为200个。

7.9.1 选项菜单

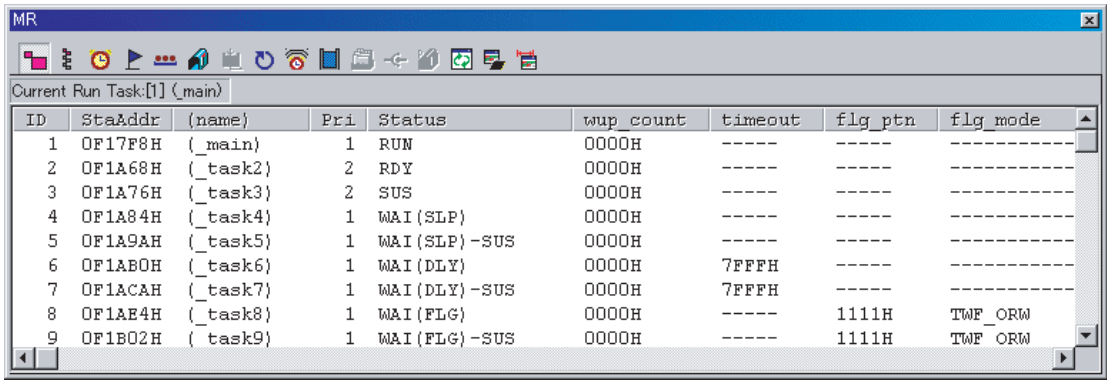
如果在窗口中单击鼠标的右键，就显示以下的弹出式菜单。在工具栏的按钮中也分配了这些菜单的主要功能。

菜单名	功能
[Select Item]（选择项目）	将被单击的项目置为选择状态。
[Delete]（删除）	删除被单击的项目。
[Copy]（复制）	复制被单击的项目。
[Paste]（粘贴）	粘贴被复制的项目。
[Create Button]（创建按钮）	创建按钮。
[Create Label]（创建标签）	创建标签。
[Create LED]（创建 LED）	创建 LED。
[Create Text]（创建文本）	创建文本。
[Display grid]（显示网格）	显示网格。
[Save...]（保存 ...）	保存输入 / 输出面板文件。
[Load...]（加载 ...）	加载输入 / 输出面板文件。
[Sampling Period...]（采样周期 ...）	设置显示更新间隔。
[Toolbar display]（工具栏显示）	切换工具栏的显示或者隐藏。
[Customize toolbar...]（自定义工具栏 ...）	自定义工具栏。
[Allow Docking]（允许入坞）	允许窗口入坞。
[Hide]（隐藏）	将窗口置为隐藏状态。

7.10 MR 窗口

MR 窗口是显示实时 OS 状态的窗口。

只有在下载了使用实时 OS 的程序后，才能使用此窗口。
如果下载的程序没有使用 MR，即使打开 MR 窗口也不显示任何内容。



- 能打开的MR窗口个数最多是显示模式的种类数。
- 通过单击各按钮，切换MR窗口的显示模式和显示内容。
- 能通过双击各任务的行，显示该任务的上下文内容。
- 能通过拖动操作，更改各模式的各显示栏的显示宽度。
- 如果下载的程序没有使用MR，就不能选择全部的选择显示模式的菜单。
- 能选择的显示模式因使用的MR而不同。

注意事项

在创建目标程序时，请使用与所用 MRxx 的版本对应的启动文件 （crt0mr.xxx），否则有可能无法调试依靠实时 OS 的部分。

7.10.1 选项菜单

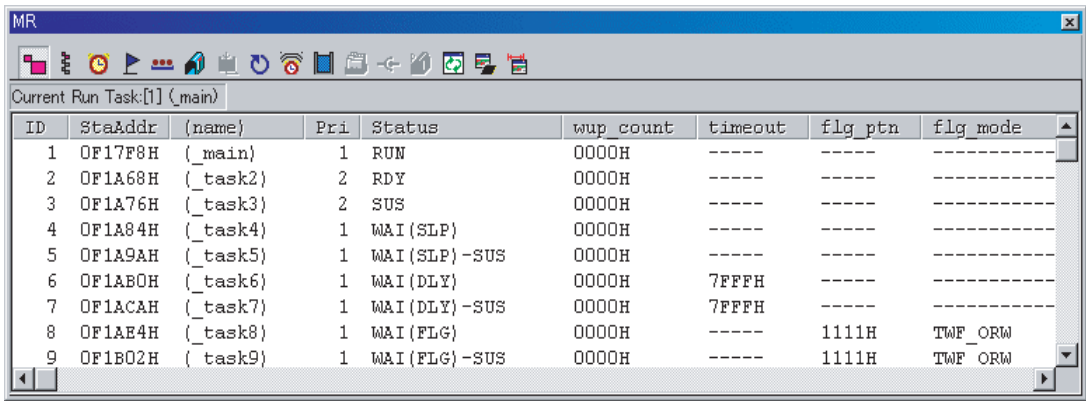
如果在窗口中单击鼠标的右键，就显示以下的弹出式菜单。在工具栏的按钮中也分配了这些菜单的主要功能。

菜单名		功能
[Mode] （模式）	[Task] （任务）	显示任务的状态。
	[Ready Queue] （就绪队列）	显示就绪队列的状态。
	[Timeout Queue] （超时队列）	显示超时队列的状态。
	[Event Flag] （事件标志）	显示事件标志的状态。
	[Semaphore] （信号量）	显示信号量的状态。
	[Mailbox] （邮箱）	显示邮箱的状态。
	[Data Queue] （数据队列）	显示数据队列的状态。
	[Cyclic Handler] （周期处理程序）	显示周期启动处理程序的状态。
	[Alarm Handler] （警报处理程序）	显示警报处理程序的状态。
	[Memory Pool] （存储池）	显示存储池的状态。
	[Message Buffer]* （信息缓冲器）	显示信息缓冲器的状态。
	[Port] （端口）	显示端口的状态。
	[Mailbox(with Priority)] （邮箱 （带优先级））	显示邮箱 （带优先级）的状态。
[Context...] （上下文 ...）		显示任务的上下文。
[Layout] （布局）	[Status Bar] （状态栏）	切换状态栏的显示或者隐藏。
[Refresh] （刷新）		刷新存储器。
[RAM Monitor] （RAM 监控）	[Enable RAM Monitor] （RAM 监控有效化）	将 RAM 监控功能置为有效。
	[Sampling Period...] （采样周期 ...）	设置采样周期。
[Toolbar display] （工具栏显示）		切换工具栏的显示或者隐藏。
[Customize toolbar...] （自定义工具栏 ...）		自定义工具栏。
[Allow Docking] （允许入坞）		允许窗口入坞。
[Hide] （隐藏）		隐藏窗口。

*： 本产品不支持。

7.10.2 任务的状态显示

请选择 MR 窗口的弹出式菜单 [Mode] （模式） →[Task] （任务）。



通过双击任意的行，在 [Context] （上下文）对话框中显示该任务的上下文信息。
有关 Context 对话框的详细内容，请参照 “7.10.12 任务上下文的参照和设置”。
在状态栏中显示当前正在执行的任务 ID 和任务名。

7.10.2.1 任务的状态显示 （使用符合 µITRON3 规格的 MRxx 的情况）

按 ID 号的顺序显示配置中定义的全部任务。各项目的内容如下 （使用符合 µITRON3 规格的 MRxx 的情况）：

项目	内容
[ID]	显示任务 ID 号。
[StaAddr]	显示任务的起始地址。
[(name)]	显示任务名。
[Pri]	显示优先级。
[Status]*1	显示任务的状态。
[wup_count]	显示唤醒计数值。
[timeout]	在任务处于时间等待状态时，显示该超时值。
[flg_ptn]	在任务处于事件标志等待状态时，显示该等待位图形。
[flg_mode]*2	在任务处于事件标志等待状态时，显示该等待解除条件。

- *1任务的状态显示

显示	状态
RUN	执行状态
RDY	能执行的状态
SUS	强制等待状态
DMT	休止状态
WAI(SLP)	睡眠状态
WAI(SLP)-SUS	睡眠状态 （二重等待）
WAI(SLP-TMO)	带超时的睡眠状态
WAI(SLP-TMO)-SUS	带超时的睡眠状态 （二重等待）
WAI(DLY)	dly_tsk 的时间经过等待状态
WAI(DLY)-SUS	dly_tsk 的时间经过等待状态 （二重等待）
WAI(FLG)	事件标志等待状态
WAI(FLG)-SUS	事件标志等待状态 （二重等待）
WAI(FLG-TMO)	带超时的事件标志等待状态
WAI(FLG-TMO)-SUS	带超时的事件标志等待状态 （二重等待）
WAI(SEM)	信号量资源的获得等待状态
WAI(SEM)-SUS	信号量资源的获得等待状态 （二重等待）
WAI(SEM-TMO)	带超时的信号量资源的获得等待状态
WAI(SEM-TMO)-SUS	带超时的信号量资源的获得等待状态 （二重等待）
WAI(MBX)	邮箱接收等待状态
WAI(MBX)-SUS	邮箱接收等待状态 （二重等待）
WAI(MBX-TMO)	带超时的邮箱接收等待状态
WAI(MBX-TMO)-SUS	带超时的邮箱接收等待状态 （二重等待）

- *2事件标志的等待解除条件的显示

flg_mode	状态
TWF_ANDW	等待由等待位图形设置的位全部被置为事件标志 （AND 等待）。
TWF_ANDW+TWF_CLR	在发生 AND 等待并解除了任务的等待状态时，将事件标志的值清 “0”。
TWF_ORW	等待由等待位图形设置的位中的某位被置为事件标志 （OR 等待）。
TWF_ORW+TWF_CLR	在发生 OR 等待并解除了任务的等待状态时，将事件标志的值清 “0”。

7.10.2.2 任务的状态显示 （使用符合 µITRON4 规格的 MRxx 的情况）

按 ID 号的顺序显示配置中定义的全部任务。各项目的内容如下 （使用符合 µITRON4 规格的 MRxx 的情况）：

项目	内容
[ID]	显示任务 ID 号。
[Name]	显示任务名。
[Pri]	显示优先级。
[Status]*1	显示任务的状态。
[Wupcnt]	显示唤醒计数值。
[Actcnt]	显示启动请求的队列数。
[Tmout]	在任务处于时间等待状态时，显示该超时值 （ms 单位）。
[Flgptn]	在任务处于事件标志等待状态时，显示该等待位图形。
[Wfmode]*2	在任务处于事件标志等待状态时，显示该等待解除条件。

- *1任务的状态显示

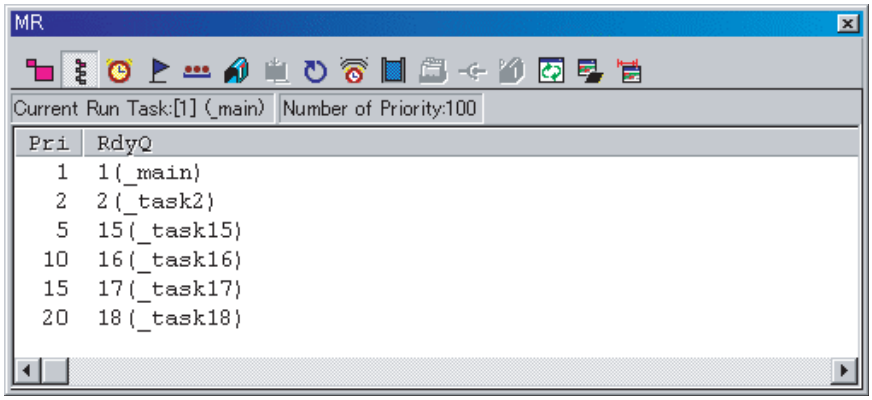
Status	状态
RUN	执行状态
RDY	能执行的状态
SUS	强制等待状态
DMT	休止状态
WAI(SLP)	睡眠状态
WAI(SLP)-SUS	睡眠状态 （二重等待）
WAI(SLP-TMO)	带超时的睡眠状态
WAI(SLP-TMO)-SUS	带超时的睡眠状态 （二重等待）
WAI(DLY)	dly_tsk 的时间经过等待状态
WAI(DLY)-SUS	dly_tsk 的时间经过等待状态 （二重等待）
WAI(FLG)	事件标志等待状态
WAI(FLG)-SUS	事件标志等待状态 （二重等待）
WAI(FLG-TMO)	带超时的事件标志等待状态
WAI(FLG-TMO)-SUS	带超时的事件标志等待状态 （二重等待）
WAI(SEM)	信号量资源的获得等待状态
WAI(SEM)-SUS	信号量资源的获得等待状态 （二重等待）
WAI(SEM-TMO)	带超时的信号量资源的获得等待状态
WAI(SEM-TMO)-SUS	带超时的信号量资源的获得等待状态 （二重等待）
WAI(MBX)	邮箱接收等待状态
WAI(MBX)-SUS	邮箱接收等待状态 （二重等待）
WAI(MBX-TMO)	带超时的邮箱接收等待状态
WAI(MBX-TMO)-SUS	带超时的邮箱接收等待状态 （二重等待）
WAI(SDTQ)	数据队列的发送等待状态
WAI(SDTQ)-SUS	数据队列的发送等待状态 （二重等待）
WAI(SDTQ-TMO)	带超时的数据队列的发送等待状态
WAI(SDTQ-TMO)-SUS	带超时的数据队列的发送等待状态 （二重等待）
WAI(RDTQ)	数据队列的接收等待状态
WAI(RDTQ)-SUS	数据队列的接收等待状态 （二重等待）
WAI(RDTQ-TMO)	带超时的数据队列的接收等待状态
WAI(RDTQ-TMO)-SUS	带超时的数据队列的接收等待状态 （二重等待）
WAI(VSDTQ)	扩展数据队列的发送等待状态
WAI(VSDTQ)-SUS	扩展数据队列的发送等待状态 （二重等待）
WAI(VSDTQ-TMO)	带超时的扩展数据队列的发送送信等待状态
WAI(VSDTQ-TMO)-SUS	带超时的扩展数据队列的发送送信等待状态 （二重等待）
WAI(VRDTQ)	扩展数据队列的接收等待状态
WAI(VRDTQ)-SUS	扩展数据队列的接收等待状态 （二重等待）
WAI(VRDTQ-TMO)	带超时的扩展数据队列的接收等待状态
WAI(VRDTQ-TMO)-SUS	带超时的扩展数据队列的接收等待状态 （二重等待）
WAI(MPF)	固定长存储块的获得等待状态
WAI(MPF)-SUS	固定长存储块的获得等待状态 （二重等待）
WAI(MPF-TMO)	带超时的固定长存储块的获得等待状态
WAI(MPF-TMO)-SUS	带超时的固定长存储块的获得等待状态 （二重等待）

- *2 事件标志的等待解除条件的显示

Wfmode	状态
TWF_ANDW	等待由等待位图形设置的位全部被置为事件标志 （AND 等待）。
TWF_ORW	等待由等待位图形设置的位中的某位被置为事件标志 （OR 等待）。

7.10.3 就绪队列的状态显示

请选择 MR 窗口的弹出式菜单 [Mode] （模式） →[Ready Queue] （就绪队列）。



在状态栏中显示当前正在执行的任务 ID、任务名和最大优先级。

7.10.3.1 就绪队列的状态显示 （使用符合 µITRON3 规格的 MRxx 的情况）

按优先级从高到低的顺序显示连接就绪队列的任务。各项目的内容如下 （使用符合 µITRON3 规格的 MRxx 的情况）：

项目	内容
[Pri]	显示优先级。
[RdyQ]	显示排列在就绪队列中的任务 ID 号。

- 在 RdyQ 栏中显示的任务名的显示字符数最多为 8 个。当任务名超过 8 个字符时，省略超过部分。

7.10.3.2 就绪队列的状态显示 （使用符合 µITRON4 规格的 MRxx 的情况）

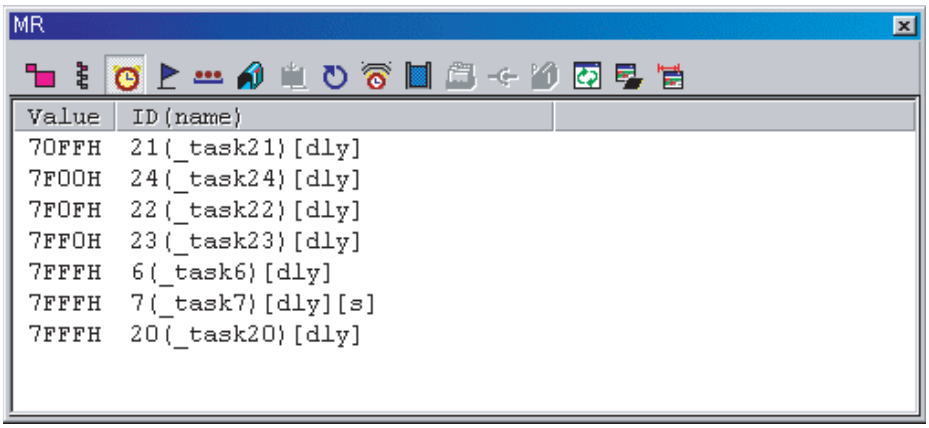
按优先级从高到低的顺序显示连接就绪队列的任务。各项目的内容如下 （使用符合 µITRON4 规格的 MRxx 的情况）：

项目	内容
[Pri]	显示优先级。
[Ready Queue] （就绪队列）	显示排列在就绪队列中的任务 ID 号。

- 在 RdyQ 栏中显示的任务名的显示字符数最多为 8 个。当任务名超过 8 个字符时，省略超过部分。

7.10.4 超时队列的状态显示

请选择 MR 窗口的弹出式菜单 [Mode] （模式） →[Timeout Queue] （超时队列）。



7.10.4.1 超时队列的状态显示 （使用符合 µITRON3 规格的 MRxx 的情况）

按超时值从小到大的顺序显示现在的时间等待状态的任务。各项目的内容如下 （使用符合 µITRON3 规格的 MRxx 的情况）

项目	内容
[Value] （值）	显示各任务从现在开始的超时值。
[ID(name)] （ID （名称））	显示排列在超时队列中的任务 ID 号、任务名和等待状态的种类。

- 表示等待状态种类的字符串如下：

字符串	等待状态
[slp]	tslp_tsk 的等待
[dly]	dly_tsk 的等待
[flg]	twai_flg 的等待
[sem]	twai_sem 的等待
[mbx]	trcv_msg 的等待

- 当连接超时队列的任务为强制等待状态 （二重等待状态）时，就在ID(name)栏中显示的字符串后面附加字符串 “[s]”，以表示二重等待状态。

普通的显示	26(_task26)
[WAIT-SUSPEND] （二重等待）状态的显示	26(_task26)[s]

7.10.4.2 超时队列的状态显示 （使用符合 µITRON4 规格的 MRxx 的情况）

按超时值从小到大的顺序显示现在的时间等待状态的任务。各项目的内容如下 （使用符合 µITRON4 规格的 MRxx 的情况）

项目	内容
[Tmout]	显示各任务从现在开始的超时值。
[ID(Name)] （ID （名称））	显示排列在超时队列中的任务 ID 号、任务名和等待状态的种类。

- 表示等待状态种类的字符串如下：

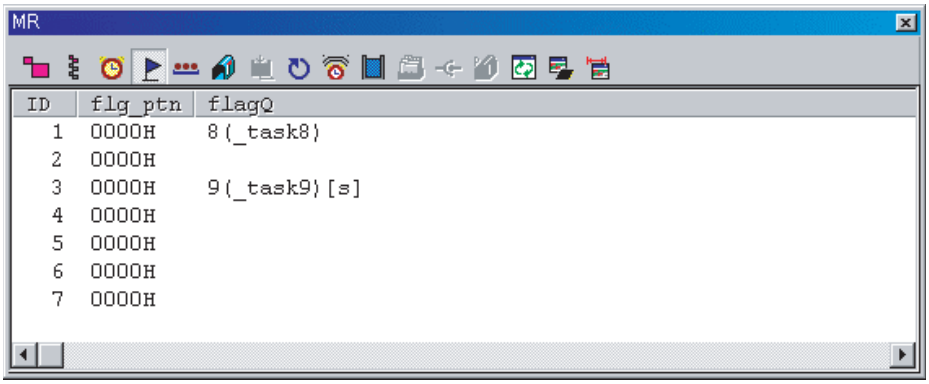
字符串	等待状态
[slp]	tslp_tsk 的等待
[dly]	dly_tsk 的等待
[flg]	twai_flg 的等待
[sem]	twai_sem 的等待
[mbx]	trcv_mbx 的等待
[mpf]	tget_mpf 的等待
[sdtq]	tsnd_dtq 的等待
[rdtq]	trcv_dtq 的等待
[vsdtq]	vtsnd_dtq 的等待
[vrdtq]	vtrcv_dtq 的等待

- 当连接超时队列的任务为强制等待状态 （二重等待状态）时，就在[ID(name)] （ID （名称））栏中显示的字符串后面附加字符串 “[s]”，以表示二重等待状态。

普通的显示	26(_task26)
[WAIT-SUSPEND] （二重等待）状态的显示	26(_task26)[s]

7.10.5 事件标志的状态显示

请选择 MR 窗口的弹出式菜单 [Mode] （模式） →[Event Flag] （事件标志）。



7.10.5.1 事件标志的状态显示 （使用符合 µITRON3 规格的 MRxx 的情况）

按 ID 号的顺序显示全部事件标志。各项目的内容如下 （使用符合 µITRON3 规格的 MRxx 的情况）：

项目	内容
[ID]	显示事件标志的 ID 号。
[flg_ptn]	显示事件标志的位图形。
[flagQ]	显示排列在事件标志队列中的任务 ID 号。

- 当连接事件标志队列的任务为带超时的等待状态 （twai_flg的等待状态）时，就在flagQ栏中显示的字符串后面附加字符串 “[tmo]”，以表示带超时的等待状态。
  当连接事件标志队列的任务为强制等待状态 （二重等待状态）时，就在flagQ栏中显示的字符串后面附加字符串 “[s]”，以表示二重等待状态。

普通	26(_task26)
[WAIT-SUSPEND] （二重等待）状态	26(_task26)[s]
带超时的等待状态 + 二重等待状态	26(_task26)[tmo][s]

- 在flagQ栏中显示的任务名的显示字符数最多为8个。当任务名超过8个字符时，省略超过部分。

7.10.5.2 事件标志的状态显示 （使用符合 µITRON4 规格的 MRxx 的情况）

按 ID 号的顺序显示全部事件标志。各项目的内容如下 （使用符合 µITRON4 规格的 MRxx 的情况）：

项目	内容
[ID]	显示事件标志的 ID 号。
[Flgatr]	显示事件标志的属性。
[Flgptn]	显示事件标志的位图形。
[Flag Queue] （标志队列）	显示排列在事件标志队列中的任务 ID 号。

- Flgatr栏的显示内容有以下几种：

TA_TFIFO	按 FIFO 进行等待任务的排队。
TA_TPRI	按优先级的顺序进行等待任务的排队。
TA_WSGL	禁止多个任务的等待。
TA_WMUL	允许多个任务的等待。
TA_CLR	清除指定。

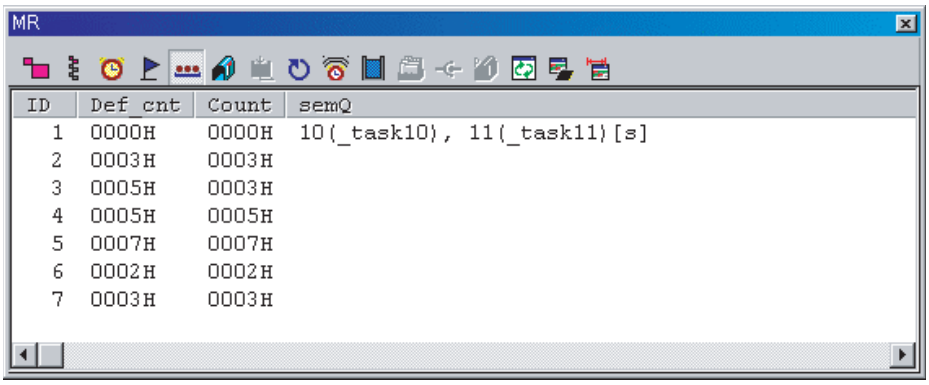
- 当连接事件标志队列的任务为带超时的等待状态 （twai_flg的等待状态）时，就在[Flag Queue] （标志队列）栏中显示的字符串后面附加字符串 “[tmo]”，以表示带超时的等待状态。
  当连接事件标志队列的任务为强制等待状态 （二重等待状态）时，就在[Flag Queue] （标志队列）栏中显示的字符串后面附加字符串 “[s]”，以表示二重等待状态。

普通	26(_task26)
[WAIT-SUSPEND] （二重等待）状态	26(_task26)[s]
带超时的等待状态 + 二重等待状态	26(_task26)[tmo][s]

- 在[Flag Queue] （标志队列）栏中显示的任务名的显示字符数最多为8个。当任务名超过8个字符时，省略超过部分。

7.10.6 信号量的状态显示

请选择 MR 窗口的弹出式菜单 [Mode] （模式） →[Semaphore] （信号量）。



7.10.6.1 信号量的状态显示 （使用符合 µITRON3 规格的 MRxx 的情况）

按 ID 号的顺序显示全部信号量。各项目的内容如下 （使用符合 µITRON3 规格的 MRxx 的情况）：

项目	内容
[ID]	显示信号量的 ID 号。
[Def_cnt]	显示信号量计数器的初始值。
[Count]	显示当前信号量的计数器。
[semQ]	显示排列在信号量队列中的任务 ID 号和任务名。

- 当连接信号量队列的任务为带超时的等待状态 （twai_sem 的等待状态）时，就在 semQ 栏中显示的字符串后面附加字符串 “[tmo]”，以表示带超时的等待状态。
  当连接信号量队列的任务为强制等待状态 （二重等待状态）时，就在 semQ 栏中显示的字符串后面附加字符串 “[s]”，以表示二重等待状态。

普通	26(_task26)
[WAIT-SUSPEND] （二重等待）状态	26(_task26)[s]
带超时的等待状态 + 二重等待状态	26(_task26)[tmo][s]

- 在 semQ 栏中显示的任务名的显示字符数最多为 8 个。当任务名超过 8 个字符时，省略超过部分。

7.10.6.2 信号量的状态显示 （使用符合 µITRON4 规格的 MRxx 的情况）

按 ID 号的顺序显示全部信号量。各项目的内容如下 （使用符合 µITRON4 规格的 MRxx 的情况）：

项目	内容
[ID]	显示信号量的 ID 号。
[Sematr]	显示信号量的属性。
[Semcnt]	显示当前信号量的计数器。
[Semaphore Queue]	显示排列在信号量队列中的任务 ID 号和任务名。

- Sematr栏的显示内容有以下几种：

TA_TFIFO	按 FIFO 进行等待任务的排队。
TA_TPRI	按优先级的顺序进行等待任务的排队。

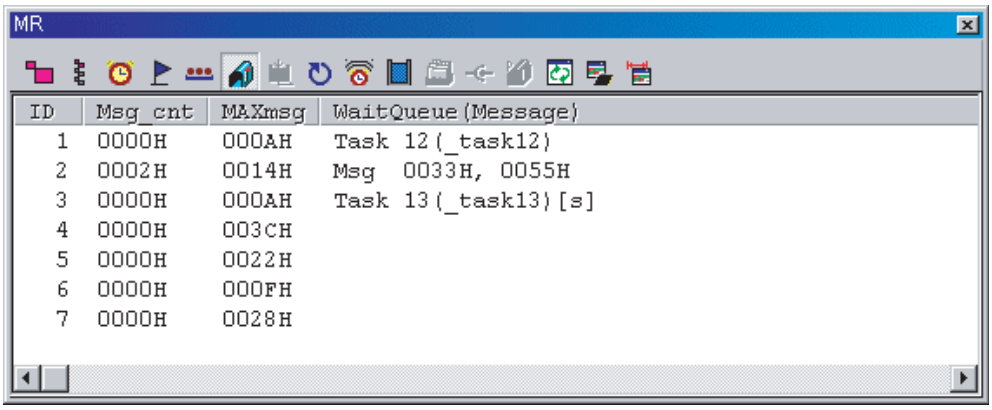
- 当连接信号量队列的任务为带超时的等待状态 （twai_sem的等待状态）时，就在[Semaphore Queue]（信号量队列）栏中显示的字符串后面附加字符串 “[tmo]”，以表示带超时的等待状态。
  当连接信号量队列的任务为强制等待状态 （二重等待状态）时，就在[Semaphore Queue] （信号量队列）栏中显示的字符串后面附加字符串 “[s]”，以表示二重等待状态。

普通	26(_task26)
[WAIT-SUSPEND] （二重等待）状态	26(_task26)[s]
带超时的等待状态 + 二重等待状态	26(_task26)[tmo][s]

- 在[Semaphore Queue] （信号量队列）栏中显示的任务名的显示字符数最多为8个。当任务名超过8个字符时，省略超过部分。

7.10.7 邮箱的状态显示

请选择 MR 窗口的弹出式菜单 [Mode] （模式） →[Mailbox] （邮箱）。



7.10.7.1 邮箱的状态显示 （使用符合 µITRON3 规格的 MRxx 的情况）

按 ID 号的顺序显示全部邮箱。各项目的内容如下 （使用符合 µITRON3 规格的 MRxx 的情况）：

项目	内容
[ID]	显示邮箱的 ID 号。
[Msg_cnt]	显示被保存在邮箱中的信息。
[MAXmsg]	显示邮箱能保存的信息数。
[Wait Queue(Message)] （等待队列 （信息））	显示被保存在邮箱中的信息，或者显示等待信息的任务 ID 号和任务名。

- WaitQueue (Message)栏的显示内容：当保存的内容是信息时 （上述的Msg_cnt不是 “0”的情况），就在显示字符串 “Msg”后继续显示被保存的信息。
  当保存的内容不是信息时 （上述的Msg_cnt是 “0”的情况），如果存在等待信息的任务，就在显示字符串 “Task”后继续显示等待信息的任务ID号和任务名。
- 当连接邮箱队列的任务为带超时的等待状态 （trcv_msg的等待状态）时，就在WaitQueue(Message)栏中显示的字符串后面附加字符串 “[tmo]”，以表示带超时的等待状态。
  当连接邮箱队列的任务为强制等待状态 （二重等待状态）时，就在WaitQueue(Message)栏中显示的字符串后面附加字符串 “[s]”，以表示二重等待状态。

普通	26(_task26)
[WAIT-SUSPEND] （二重等待）状态	26(_task26)[s]
带超时的等待状态 + 二重等待状态	26(_task26)[tmo][s]

- 在WaitQueue(Message)栏中显示的任务名的显示字符数最多为8个。当任务名超过8个字符时，省略超过部分。

7.10.7.2 邮箱的状态显示 （使用符合 µITRON4 规格的 MRxx 的情况）

按 ID 号的顺序显示全部邮箱。各项目的内容如下 （使用符合 µITRON4 规格的 MRxx 的情况）：

项目	内容
[ID]	显示邮箱的 ID 号。
[Mbxatr]	显示邮箱的属性。
[Mailbox Queue (Wait)] （邮箱队列 （等待））	显示等待信息的任务 ID 号和任务名。
[Mailbox Queue (Message)] （邮箱队列 （信息））	显示被保存在邮箱中的信息。

- Mbxatr栏的显示内容有以下几种：

TA_TFIFO	按 FIFO 进行等待任务的排队。
TA_TPRI	按优先级的顺序进行等待任务的排队。
TA_MFIFO	按 FIFO 进行信息的排队。
TA_MPRI	按优先级的顺序进行信息的排队。

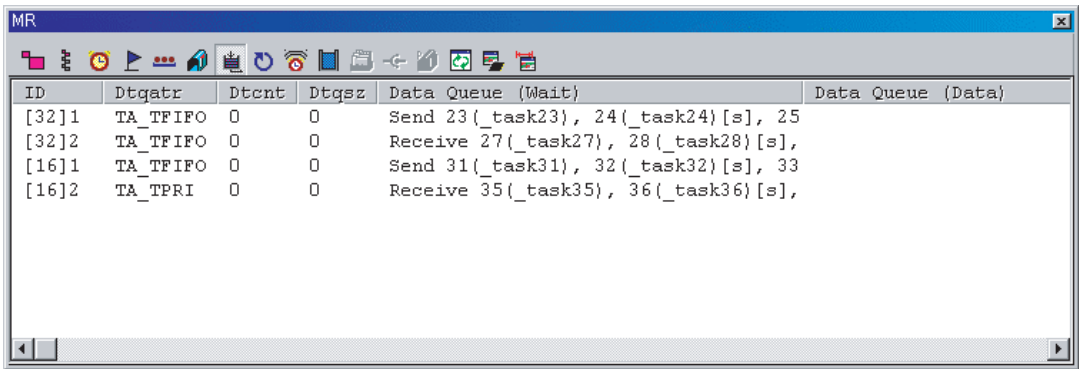
- 当连接邮箱队列的任务为带超时的等待状态 （trcv_mbx的等待状态）时，就在Mailbox Queue (Wait)栏中显示的字符串后面附加字符串 “[tmo]”，以表示带超时的等待状态。
  当连接邮箱队列的任务为强制等待状态 （二重等待状态）时，就在Mailbox Queue (Wait)栏中显示的字符串后面附加字符串 “[s]”，以表示二重等待状态。

普通	26(_task26)
[WAIT-SUSPEND] （二重等待）状态	26(_task26)[s]
带超时的等待状态 + 二重等待状态	26(_task26)[tmo][s]

- 在Mailbox Queue (Wait)栏中显示的任务名的显示字符数最多为8个。当任务名超过8个字符时，省略超过部分。

7.10.8 数据队列的状态显示

请选择 MR 窗口的弹出式菜单 [Mode] （模式） →[Data Queue] （数据队列）。



7.10.8.1 数据队列的状态显示 （使用符合 µITRON4 规格的 MRxx 的情况）

按 ID 号的顺序显示全部数据队列。各项目的内容如下 （使用符合 µITRON4 规格的 MRxx 的情况）：

项目	内容
[ID]	显示数据队列的 ID 号。
[Dtqatr]	显示数据队列的属性。
[Dtcnt]	显示排列在数据队列中的数据个数。
[Dtqsz]	显示能排列在数据队列中的数据个数。
[Data Queue (Wait)] （数据队列 （等待））	显示等待发送数据或者等待接收数据的任务 ID 号和任务名。
[Data Queue (Data)] （数据队列 （数据））	显示排列在数据队列中的数据。

- ID栏的显示内容因标准数据 （32bit）和扩展数据 （16bit）而不同。
  MR100/4的情况
  - 标准数据 （32bit）：显示字符串 “[32]”和数据队列的ID号。
  - 扩展数据 （16bit）：显示字符串 “[16]”和数据队列的ID号。
- Dtqatr栏的显示内容有以下几种：

TA_TFIFO	按 FIFO 进行等待任务的排队。
TA_TPRI	按优先级的顺序进行等待任务的排队。

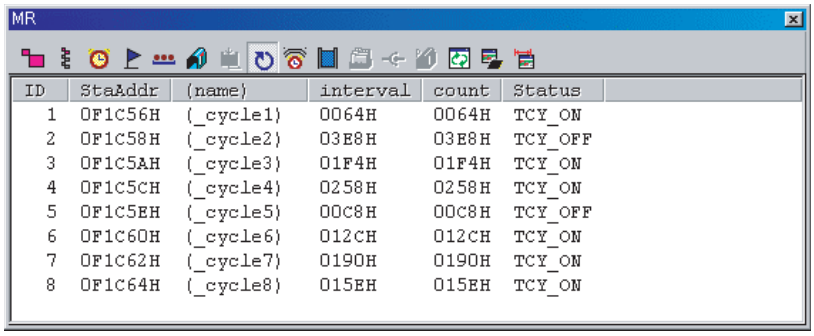
- Data Queue (Wait)栏的显示内容：当是等待发送的任务时，就在显示字符串 “Send”后继续显示任务ID和任务名。当是等待接收的任务时，就在显示字符串 “Receive”后继续显示任务ID和任务名。
- 当连接队列的任务为带超时的等待状态时，就在Data Queue (Wait)栏中显示的字符串后面附加字符串“[tmo]”，以表示带超时的等待状态。
  当连接队列的任务为强制等待状态 （二重等待状态）时，就在Data Queue (Wait)栏中显示的字符串后面附加字符串 “[tmo]”，以表示带超时的等待状态。

普通	26(_task26)
[WAIT-SUSPEND] （二重等待）状态	26(_task26)[s]
带超时的等待状态 + 二重等待状态	26(_task26)[tmo][s]

- 在Data Queue (Wait)栏中显示的任务名的显示字符数最多为8个。当任务名超过8个字符时，省略超过部分。

7.10.9 周期启动处理程序的状态显示

请选择 MR 窗口的弹出式菜单 [Mode] （模式） →[Cyclic Handler] （周期启动处理程序）。



7.10.9.1 周期处理程序的状态显示 （使用符合 µITRON3 规格的 MRxx 的情况）

按 ID 号的顺序显示全部周期启动处理程序。各项目的内容如下 （使用符合 µITRON3 规格的 MRxx 的情况）：

项目	内容
[ID]	显示周期启动处理程序的 ID 号。
[StaAddr]	显示周期启动处理程序的起始地址。
[(name)] （名称）	显示周期启动处理程序名。
[interval] （间隔）	显示周期启动处理程序的周期启动间隔。
[count] （次数）	显示到下次启动周期启动处理程序为止的中断次数 （剩余次数）。
[Status] （状态）	显示周期启动处理程序的激活状态。

- Status栏的显示内容有以下几种：

TCY_ON	周期启动处理程序有效。
TCY_OFF	周期启动处理程序无效。

7.10.9.2 周期处理程序的状态显示 （使用符合 µITRON4 规格的 MRxx 的情况）

按 ID 号的顺序显示全部周期启动处理程序。各项目的内容如下 （使用符合 µITRON4 规格的 MRxx 的情况）：

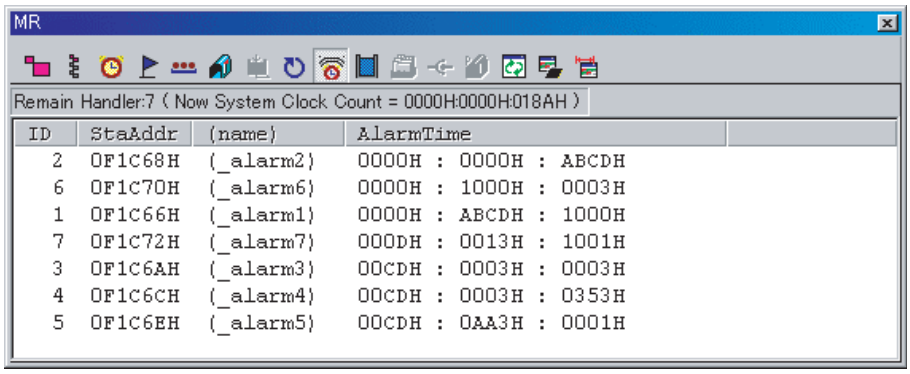
项目	内容
[ID]	显示周期启动处理程序的 ID 号。
[Name] （名称）	显示周期启动处理程序名。
[Cycphs]	以 ms 为单位显示启动相位。
[Cyctim]	以 ms 为单位显示周期启动间隔。
[Tmout]	以 ms 为单位显示到下次启动为止的时间。
[Status] （状态）	显示周期启动处理程序的激活状态。

- Status栏的显示内容有以下几种：

TCYC_STA	激活中
TCYC_STP	停止中

7.10.10 警报处理程序的状态显示

请选择 MR 窗口的弹出式菜单 [Mode] （模式） →[Alarm Handler] （警报处理程序）。



在状态栏中显示等待启动的警报处理程序数和现在的系统时钟计数 （只限于使用符合 µITRON3 规格的 MRxx 的情况）。

7.10.10.1 警报处理程序的状态显示 （使用符合 µITRON3 规格的 MRxx 的情况）

按启动时刻从早到晚的顺序显示现在还没有启动的警报处理程序。各项目的内容如下 （使用符合 µITRON3 规格的 MRxx 的情况）：

项目	内容
[ID]	显示警报处理程序的 ID 号。
[StaAddr]	显示警报处理程序的起始地址。
[(name)] （（名称））	显示警报处理程序名。
[AlarmTime] （警报时间）	显示警报处理程序的启动时刻。

7.10.10.2 警报处理程序的状态显示 （使用符合 µITRON4 规格的 MRxx 的情况）

按启动时刻从早到晚的顺序显示现在还没有启动的警报处理程序。各项目的内容如下 （使用符合 µITRON4 规格的 MRxx 的情况）：

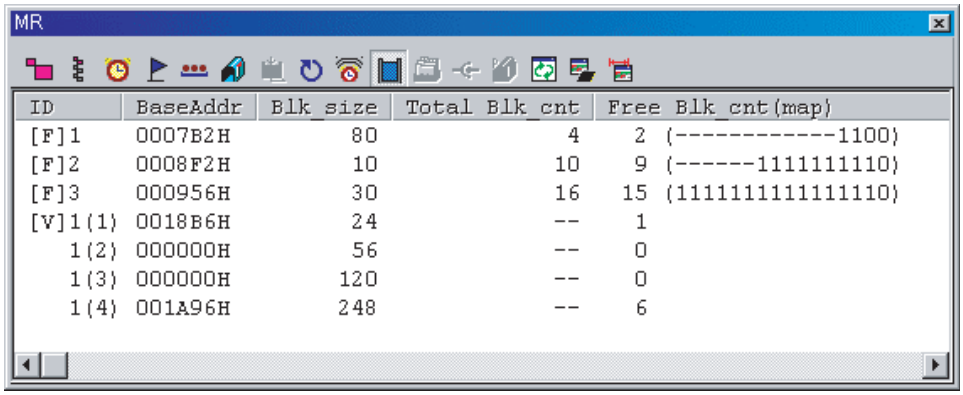
项目	内容
[ID]	显示警报处理程序的 ID 号。
[Name] （名称）	显示警报处理程序名。
[Almtim]	以 ms 为单位显示到下次启动警报处理程序为止的时间。
[Status] （状态）	显示警报处理程序的启动状态。

- Status栏的显示内容有以下几种：

TALM_STA	运行中
TALM_STP	停止中

7.10.11 存储池的状态显示

请选择 MR 窗口的弹出式菜单 [Mode] （模式） →[Memory Pool] （存储池）。



7.10.11.1 存储池的状态显示 （使用符合 µITRON3 规格的 MRxx 的情况）

按 ID 号的顺序 （固定长和任意长的顺序）显示存储池。各项目的内容如下 （使用符合 µITRON3 规格的 MRxx 的情况）：

项目	内容
[ID]	显示存储池的 ID 号。
[BaseAddr]	显示存储池的基址。
[Blk_size]	显示存储池的块大小。
[Total Blk_cnt] （全部块数）	显示存储池的全部块数。
[Free Blk_cnt(map)] （空闲块数 （映射））	显示未使用的块数和存储块信息 （位信息）。

- ID栏的显示内容因固定长和任意长而不同。
  - 固定长：显示字符串 “[F]”和存储池的ID号。
  - 任意长：在第1行显示字符串 “[V]”、存储池ID号和块ID号，在第2～4行显示存储池ID号和块ID号。块ID号加括号显示。
- 在任意长存储池的情况下， Total Blk_cnt栏显示 “--”。
  不显示Free Blk_cnt(map)栏的位信息。
- 在固定长存储池的情况下， Free Blk_cnt(map)栏的存储块信息的各位显示格式如下：

显示	内容
“0”	不可使用存储块 （使用中）。
“1”	可使用存储器块 （未使用）。
“—”	存储块不存在。

7.10.11.2 存储池的状态显示 （使用符合 µITRON4 规格的 MRxx 的情况）

按 ID 号的顺序 （固定长和任意长的顺序）显示存储池。各项目的内容如下 （使用符合 µITRON4 规格的 MRxx 的情况）：

项目	内容
[ID]	显示存储池的 ID 号。
[Mplatr]	显示存储池的属性。
[Mpladr]	显示存储池区域的起始地址。
[Mplsz]	显示存储池区域的大小。
[Blkcnt]	显示固定长存储池的全部块数。
[Fblkcnt]	显示未使用的块数。
[Memory Pool Queue] （存储池队列）	显示等待存储池的任务 ID 号和任务名。

- Mplatr栏的显示内容有以下几种：

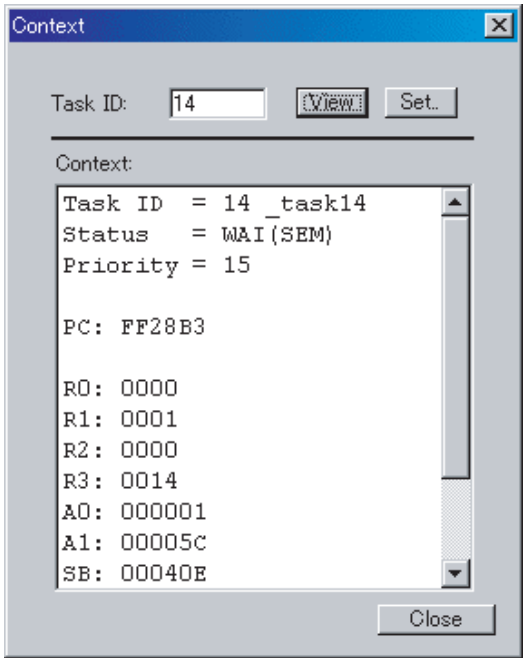
TA_TFIFO	按 FIFO 进行等待任务的排队。
TA_TPRI	按优先级的顺序进行等待任务的排队。

- ID栏的显示内容因固定长和任意长而不同。
  - 固定长：显示字符串 “[F]”和存储池的ID号。
  - 任意长：在第1行显示字符串 “[V]”、存储池ID号和块ID号，在第2～4行显示存储池ID号和块ID号。块ID号加括号显示。

7.10.12 任务上下文的参照和设置

7.10.12.1 任务上下文的参照

请在 MR 窗口中选择弹出式菜单 [Context...]（上下文 ...）。
打开以下的对话框，此 [Context]（上下文）对话框是参照和设置所选任务的上下文信息的对话框。
能通过双击任务状态显示模式的数据显示部分，打开此对话框。



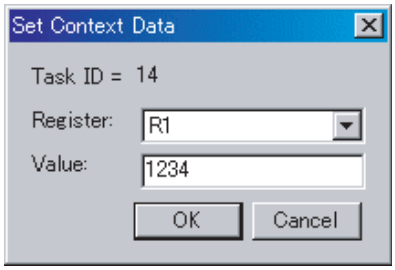
请在 [Task ID]（任务 ID）区中输入任务 ID 号，并且单击（或者输入 Enter 键）[View]（视图）按钮。
在 [Context]（上下文）栏中显示所选任务的上下文。

- 如果在单击 [View]（视图）按钮时给 [Task ID]（任务 ID）栏输入的任务是“RUN”（执行）或者“DMT”（休眠）状态，就不显示上下文（[Context]（上下文）栏只显示任务 ID 和任务的状态）。
- 如果在单击 [View]（视图）按钮时给 [Task ID]（任务 ID）栏输入了不存在的任务 ID 号，就会出错。

7.10.12.2 任务上下文的更改

请在 [Context]（上下文）对话框的 [Task ID]（任务 ID）栏中输入任务 ID 号，并且单击 [Set]（设置）按钮。

打开以下的对话框，此 [Set Context Data]（设置上下文数据）对话框是设置所选任务的指定上下文寄存器值的对话框。



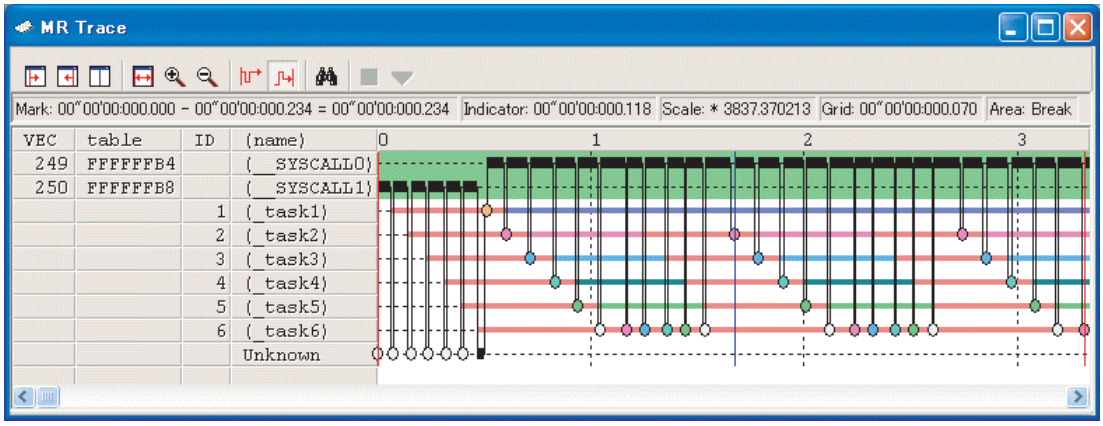
请在 [Register]（寄存器）栏的列表框中指定要更改的寄存器，并且在 [Value]（值）栏中输入要设置的值。
如果 [Value]（值）栏中设置的表达式记述有错，就认为超过了指定寄存器能设置的值的范围而会出错。

7.11 MR 跟踪窗口

MR 跟踪窗口是测量使用实时 OS 的程序的任务执行履历等并进行图形显示的窗口。
除了任务执行履历以外，还同时测量和显示中断处理、任务状态转移、系统调用发行的各种履历。
只有在下载了使用本公司产实时 OS （MRxx）的目标程序后，才能使用 MR 跟踪窗口。

MR100/4 的情况

- 对象是MR100/4 V.1.01以上的版本。如果下载了MR100/4 V.1.00创建的目标程序， MR跟踪窗口就不工作也不显示任何内容。
- 必须对系统时钟使用定时器A或者定时器B。如果对系统时钟使用其他定时器或者不使用系统时钟，就不能正确地测量和显示。
- 不能测量和显示高速中断处理的履历。
- 不能测量和显示使用固定向量表的中断处理的履历。
- 不能测量和显示非内核管理中断处理的履历，但是能测量和显示从非内核管理中断处理程序中调用的测量宏指令。详细内容请参照MR100/4的用户手册。



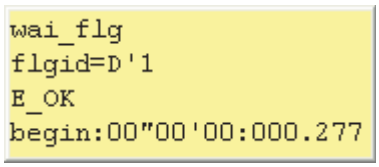
各项目的内容如下：

项目	内容
[VEC]*1	显示软件中断序号。
[table] （表）	显示中断向量表地址。
[ID]	显示任务的 ID 号。
[(name)] （（名称））	显示中断程序名、任务名、空闲处理 （显示为 “Idle”）和不明 （显示为 “Unknown”）。

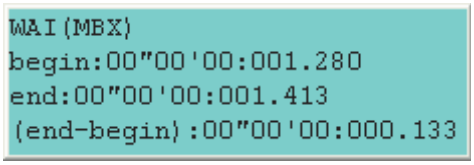
通过在窗口中显示的各信息上移动鼠标，打开以下的弹出式窗口，显示详细的内容信息。
中断处理和任务执行履历的详细内容显示信息：

```
ID=D'  3 (_task3)
begin:00"00'00:002.304
end:00"00'00:002.314
(end-begin):00"00'00:000.009
```

系统调用发行履历的详细内容显示信息：



任务状态转移履历的详细内容显示信息：



在状态栏中显示以下的信息：

- 始点标记位置的时刻值
- 终点标记位置的时刻值
- 始点标记和终点标记之间的时间宽度
- 指示器位置的时刻值
- 显示倍率
- 网格线间隔时间的宽度
- 测量 （跟踪）范围

以始点标记为基点，显示网格线。
以始点标记的位置时刻为 0，左侧 （前方时间）为负，右侧 （后方时间）为正，显示刻度。
能通过网格线大致掌握中断发生周期和处理时间等。
在状态栏的 “Grid”栏中显示网格线的间隔时间宽度。

MR 跟踪窗口的时刻值是以记录在跟踪存储器中的测量结果的起始数据为 0 的执行经过时间。
对此， MR 跟踪窗口的网格线 （刻度）上部的数字是以始点标记为 0 的相对值 （在设置对话框中指定网格间隔），与时刻值无关 （为了容易看清窗口）。

补充事项

VEC 列 *1 的软件中断序号因产品而不同。
哪个系统调用分配到哪个中断序号，请参照 MR100/4 的用户手册。

注意事项

- 只有在 Init 对话框的运行模式选项卡中选择了跟踪模式时，才能使用 MR 跟踪窗口。
- 在 MR 跟踪窗口中，将跟踪模式设置为跟踪优先模式进行测量，因此可能延迟 MCU 的执行。

7.11.1 选项菜单

如果在窗口中单击鼠标的右键，就显示以下的弹出式菜单。在工具栏的按钮中也分配了这些菜单的主要功能。

菜单名		功能
[Start Marker] （始点标记）		将始点标记移到显示画面内。
[End Marker] （终点标记）		将终点标记移到显示画面内。
[Indicator] （指示器）		将当前的位置标记 （指示器）移到显示画面内。
[Adjust] （调整）		以横向整个宽度显示始点 / 终点标记的范围。
[Expand] （扩大）		扩大显示倍率。
[Reduce] （缩小）		缩小显示倍率。
[Trace Stop]*2 （停止跟踪）		停止跟踪测量并显示结果。
[Trace Restart]*2 （重新跟踪）		重新测量跟踪数据。
[Search...] （检索 ...）		检索系统调用发行履历。
[Trace Range] （跟踪范围）	[After]	将测量范围条件设置为 After。
	[Break]	将测量范围条件设置为 Break。
[Value...] （值 ...）		设置网格间隔和显示倍率。
[Color...] （颜色 ...）		设置各种显示颜色。
[Init Order...] （初始化顺序 ...）		对显示顺序进行初始化。
[Toolbar display] （工具栏显示）		切换工具栏的显示或者隐藏。
[Customize toolbar...] （自定义工具栏 ...）		自定义工具栏。
[Allow Docking] （允许入坞）		允许窗口入坞。
[Hide] （隐藏）		隐藏窗口。

*2： 本产品不支持。

7.11.2 任务执行履历的参照 （MRxx 窗口）

在 MR 跟踪窗口中参照任务的执行履历，并且在 MR 分析窗口中参照执行履历的统计处理结果。
在使用本公司产实时 OS(MRxx) 的目标程序的情况下，能使用此窗口。

7.11.2.1 跟踪范围的选择以及程序的执行

请单击 MR 跟踪窗口的 [After] 按钮 （或者选择菜单 [Trace Range] （测量范围条件） →[After]）或者 [Break] 按钮 （或者选择菜单 [Trace Range] （跟踪范围） →[Break])。

After	记录跟踪存储器的记录数据记满前的任务执行履历。
Break	记录目标程序停止前的任务执行履历 （跟踪存储器的大小）。

请执行目标程序并记录任务的执行履历。

注意事项

事件设置窗口中设置的跟踪点无效。

7.11.2.2 任务执行履历测量的停止

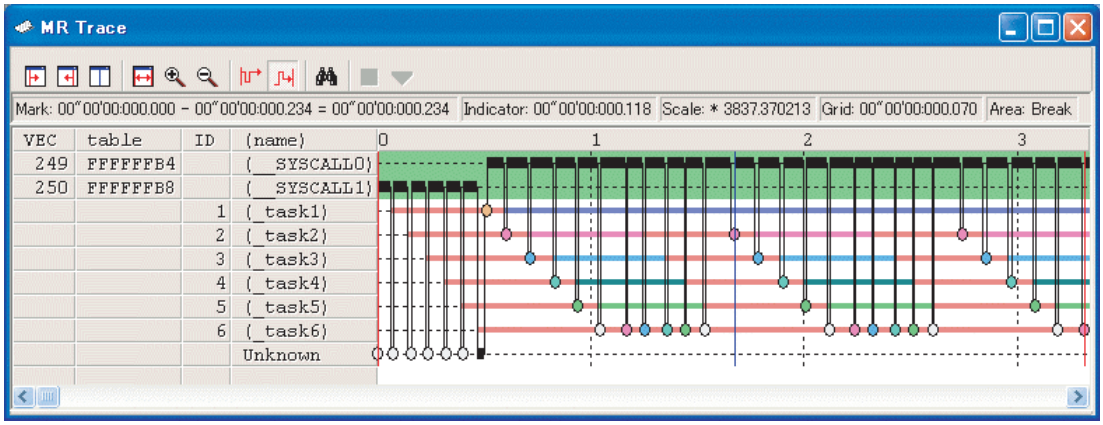
请单击 MR 跟踪窗口的工具栏中的 [Stop] （停止测量）按钮 （或者选择菜单 [Trace Stop] （停止测量））。在 MR 跟踪窗口中显示停止前的测量结果。（本产品不支持。）

7.11.2.3 任务执行履历测量的重新开始

请单击 MR 跟踪窗口的工具栏中的 [Restart] （重新测量）按钮 （或者选择菜单 [Trace Restart] （重新测量））。删除停止前的测量结果。（本产品不支持。）

7.11.2.4 任务执行转移的参照

在 MR 跟踪窗口中参照任务的执行转移。



通过在窗口中显示的各信息上移动鼠标，打开以下例子的窗口并显示详细的内容信息。

中断处理和任务执行履历的详细内容信息显示：

```
ID=D'  3 (_task3)
begin:00"00'00:002.304
end:00"00'00:002.314
(end-begin):00"00'00:000.009
```

系统调用发行履历的详细内容信息显示：

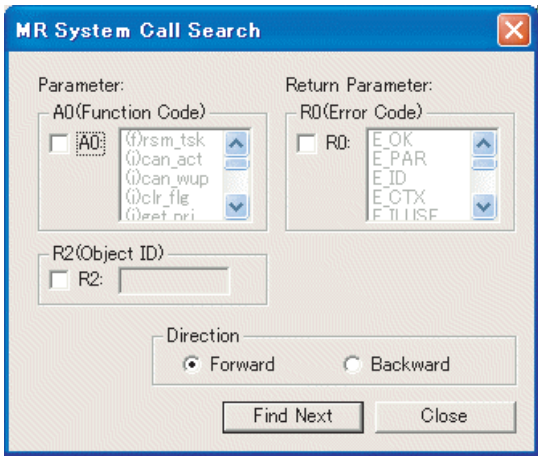
```
wai_flg
flgid=D'1
E_OK
begin:00"00'00:000.277
```

任务状态转移履历的详细内容信息显示：

```
WAI(MBX)
begin:00"00'00:001.280
end:00"00'00:001.413
(end-begin):00"00'00:000.133
```

(1) 系统调用发行履历的检索

请单击工具栏的 [Search] （检索）按钮，打开系统调用的检索对话框 （或者选择菜单 [Search...] （检索 ...））。



请指定检索条件。

能给功能码和错误码指定多个值 （OR 条件）。

用 AND 条件检索其他检索项目。

请指定下一个检索方向。在对话框中以指示器指示的位置为基点，按指定的方向进行检索。

当没有选定全部的检索项目时，检索结果就是检索方向的下一个系统调用发行履历。

请单击 [Find Next] （检索下一步）按钮，检索符合指定条件的系统调用发行履历。用 AND 条件检索指定的各项目。

在满足检索条件时，将指示器移到该位置。

(2) 显示倍率的更改

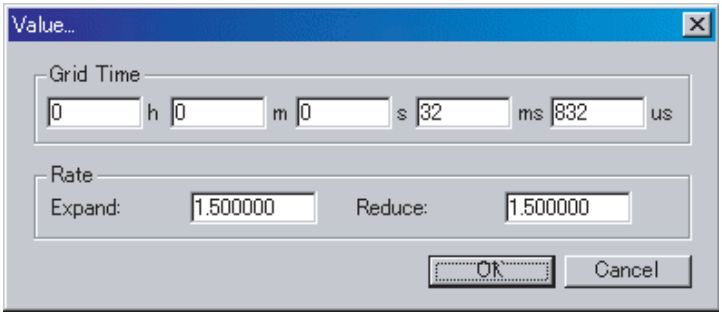
请单击工具栏的 [Expand] （显示倍率的扩大）按钮或者 [Reduce] （显示倍率的缩小）按钮 （或者选择菜单 [Expand] （显示倍率的扩大）或者 [Reduce] （显示倍率的缩小））。

以图形显示栏的左侧为基点，进行扩大 / 缩小的显示。默认值是每次以 1.5 倍进行扩大 / 缩小的显示。

在状态栏的 “Scale:*” 栏中显示倍率。

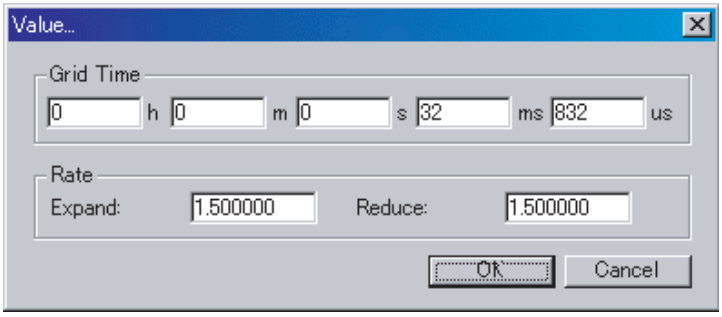
扩大 / 缩小率的默认值是 1.5 倍。要更改扩大 / 缩小率时，请选择菜单 [Value...] （值 ...）。

打开设置对话框，请指定显示的扩大 / 缩小率。



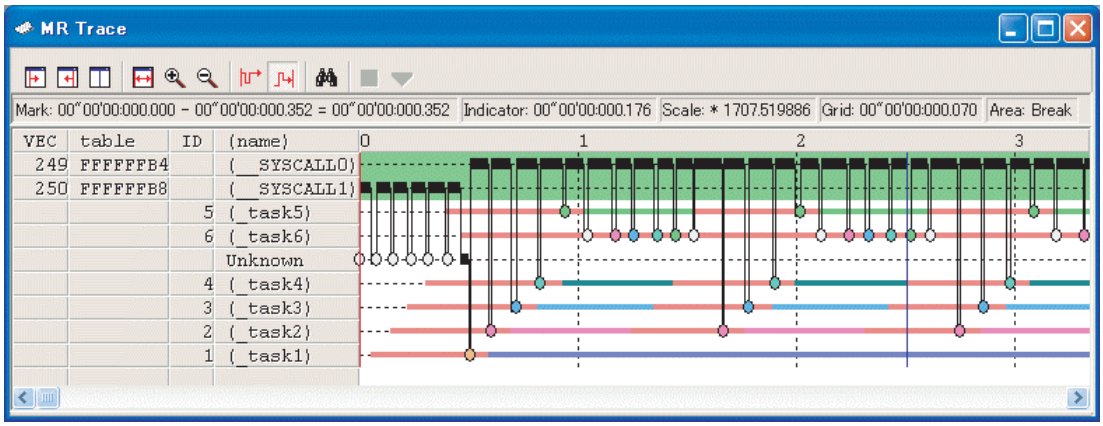
(3) 网格线显示间隔的更改

请选择菜单 [Value...] （值 ...），打开设置对话框。
请指定网格间隔的时间宽度。



(4) 任务显示顺序的更改

请将要移动的任务和中断程序 （图形显示的左侧部分）拖动到移动目标位置。



要对显示顺序进行初始化时，请选择菜单 [Init Order] （初始化顺序）。

(5) 特殊任务的显示

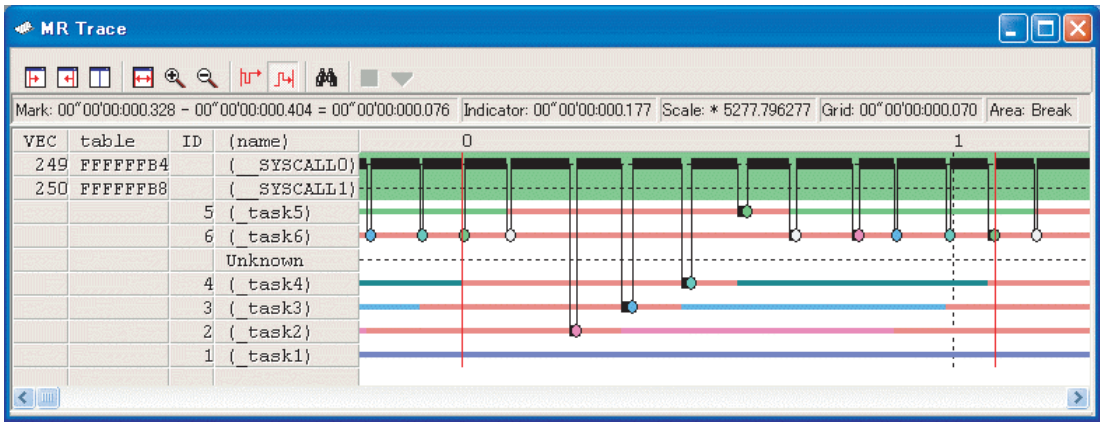
请单击要显示的任务和中断程序 （图形显示的左侧部分）。
每次单击能切换显示或者隐藏。

(6) 显示颜色的更改

请选择菜单 [Color...] （颜色 ...），打开显示颜色的设置对话框。
请单击对应各项目的按钮。打开颜色的设置对话框，请更改显示颜色。

7.11.2.5 任务执行时间的测量

能通过更改 MR 跟踪窗口的始点标记和终点标记的位置，测量标记之间的执行时间。



请拖动始点标记和终点标记的位置。
在状态栏中显示标记之间的时间宽度。

补充事项

[MR 跟踪窗口的时刻值的定义]

MR 跟踪窗口中的时刻值是指以跟踪存储器中记录的测量结果的起始数据为 0 的执行经过时间。

对此， MR 跟踪窗口的网格线 （刻度）上部的数字是以始点标记为 0 的相对值 （在设置对话框中指定网格间隔），与时刻值无关 （为了容易看清窗口）。

(1) 标记的移动

能通过拖动来移动各标记。一旦将移动鼠标到标记上，光标就发生变化，请在此状态下拖动标记。

通过单击工具栏的 [Start Marker] （始点标记）按钮，将始点标记移到窗口中 （左侧）（或者选择菜单 [Start Marker] （始点标记））。

通过单击 [End Marker] （终点标记）按钮，将终点标记移到窗口中 （右侧）（或者选择菜单 [End Marker] （终点标记））。

通过单击 [Indicator] （指示器）按钮，将指示器移到窗口中 （中央）（或者选择菜单 [Indicator] （指示器））。

但是，只能将各种标记移动到以下的位置：

- 中断处理和任务执行转移后的位置
- 任务状态转移后的位置
- 系统调用发行履历的显示位置

7.12 MR 分析窗口

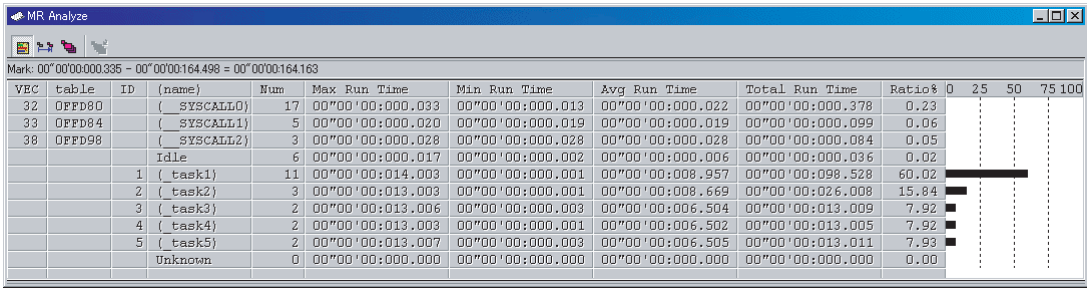
MR 分析窗口是显示由 MR 跟踪窗口的始点标记和终点标记指定范围内的测量数据的统计结果的窗口。
MR 分析窗口支持以下的 3 个显示模式：

- 各中断处理和任务的CPU 占有情况
- 各任务的就绪状态时间
- 系统调用发行履历的一览表显示 （能通过指定特殊条件进行抽取显示）

MR 分析窗口和 MR 跟踪窗口一起工作。
只有在下载了使用本公司产实时 OS(MRxx) 的目标程序后，才能使用 MR 分析窗口。

7.12.1 CPU 占有情况显示模式的构成

CPU 占有情况的显示模式是显示各中断处理和任务的 CPU 占有时间和占有率的模式。
显示由 MR 跟踪窗口的始点标记和终点标记指定的范围内的统计结果。

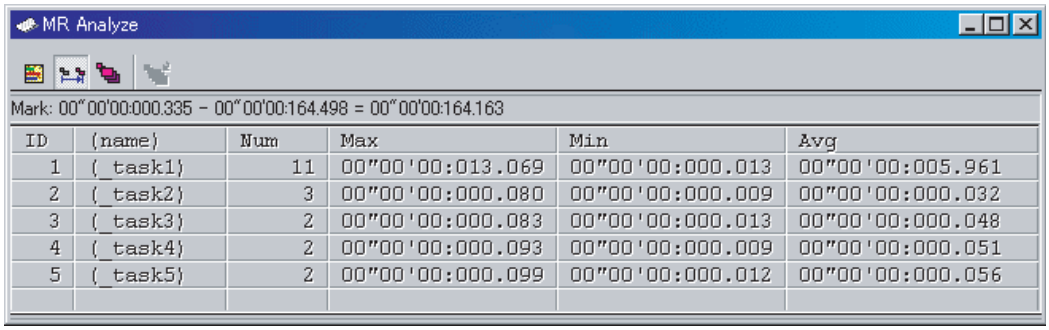


VEC	table	ID	(name)	Num	Max Run Time	Min Run Time	Avg Run Time	Total Run Time	Ratio%
32	OFFD80		(_SYSCALL0)	17	00"00'00:000.033	00"00'00:000.013	00"00'00:000.022	00"00'00:000.378	0.23
33	OFFD84		(_SYSCALL1)	5	00"00'00:000.020	00"00'00:000.019	00"00'00:000.019	00"00'00:000.099	0.06
38	OFFD98		(_SYSCALL2)	3	00"00'00:000.028	00"00'00:000.028	00"00'00:000.028	00"00'00:000.084	0.05
			Idle	6	00"00'00:000.017	00"00'00:000.002	00"00'00:000.006	00"00'00:000.036	0.02
		1	(_task1)	11	00"00'00:014.003	00"00'00:000.001	00"00'00:008.957	00"00'00:098.528	60.02
		2	(_task2)	3	00"00'00:013.003	00"00'00:000.001	00"00'00:008.669	00"00'00:026.008	15.84
		3	(_task3)	2	00"00'00:013.006	00"00'00:000.003	00"00'00:006.504	00"00'00:013.009	7.92
		4	(_task4)	2	00"00'00:013.003	00"00'00:000.001	00"00'00:006.502	00"00'00:013.005	7.92
		5	(_task5)	2	00"00'00:013.007	00"00'00:000.003	00"00'00:006.505	00"00'00:013.011	7.93
			Unknown	0	00"00'00:000.000	00"00'00:000.000	00"00'00:000.000	00"00'00:000.000	0.00

能通过单击各行的最大执行时间和最小执行时间显示栏，检索所击行对应的中断处理或者任务的最大执行时间和最小执行时间的处理履历。
MR 跟踪窗口的指示器指示检索结果的移动位置。

7.12.2 各任务的就绪状态时间显示模式的构成

各任务的就绪状态时间显示模式是显示各任务从可执行状态转移到执行状态的时间统计结果的模式。
显示由 MR 跟踪窗口的始点标记和终点标记指定的范围内的统计结果。

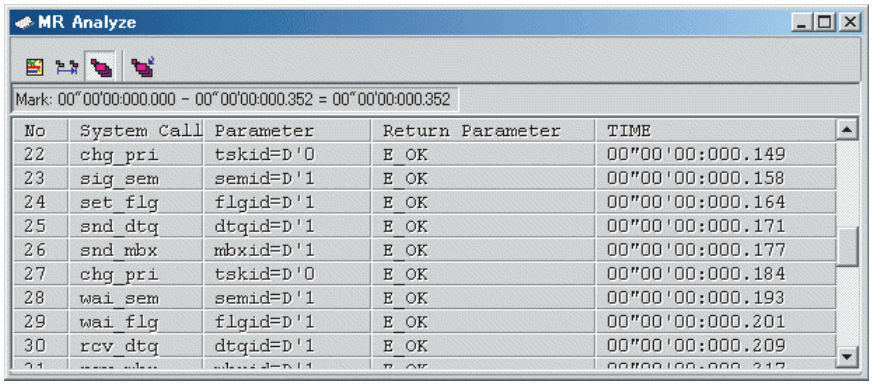


ID	(name)	Num	Max	Min	Avg
1	(_task1)	11	00"00'00:013.069	00"00'00:000.013	00"00'00:005.961
2	(_task2)	3	00"00'00:000.080	00"00'00:000.009	00"00'00:000.032
3	(_task3)	2	00"00'00:000.083	00"00'00:000.013	00"00'00:000.048
4	(_task4)	2	00"00'00:000.093	00"00'00:000.009	00"00'00:000.051
5	(_task5)	2	00"00'00:000.099	00"00'00:000.012	00"00'00:000.056

能通过单击各行的最大就绪时间和最小就绪时间显示栏，检索所击行对应的任务的最大就绪时间和最小就绪时间的处理履历。
MR 跟踪窗口的指示器指示检索结果的移动位置。

7.12.3 系统调用发行履历的一览表显示模式的构成

系统调用发行履历的一览表显示模式是显示被发行的系统调用列表的模式。
以列表形式显示由 MR 跟踪窗口的始点标记和终点标记指定的范围内的系统调用发行履历一览表。
序号表示在被测量的范围内从第一个系统调用开始计数的数值。



能通过单击各行，检索所击行对应的系统调用发行履历。
MR 跟踪窗口的指示器指示检索结果的移动位置。

7.12.4 选项菜单

如果在窗口中单击鼠标的右键，就显示以下的弹出式菜单。在工具栏的按钮中也分配了这些菜单的主要功能。

菜单名	功能
[Run Time] （执行时间）	显示 CPU 的占有情况。
[Rdy->Run] （就绪 -> 执行）	显示各任务的就绪状态时间。
[System Call] （系统调用）	显示系统调用发行履历一览表。
[Pick Up System Call...] （抽取系统调用 ...）	通过指定特殊条件，抽取系统调用发行履历一览表。
[Save...] （保存 ...）	将分析数据保存到文件。
[Toolbar display[（工具栏显示）	切换工具栏的显示或者隐藏。
[Customize toolbar...] （自定义工具栏 ...）	自定义工具栏。
[Allow Docking] （允许入坞）	允许窗口入坞。
[Hide] （隐藏）	隐藏窗口。

7.12.5 任务执行履历的统计处理

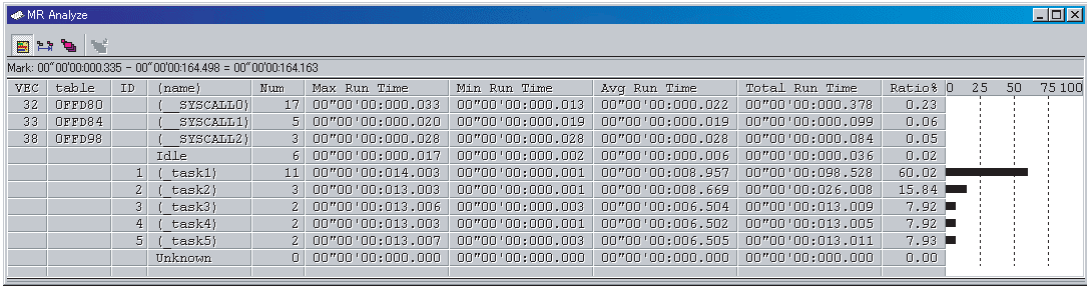
在 MR 分析窗口中参照执行履历的统计处理。

MR 分析窗口和 MR 跟踪窗口一起工作。当没有打开 MR 跟踪窗口或者 MR 跟踪窗口中没有显示任何内容时， MR 分析窗口不工作。

能通过执行履历的统计处理参照以下内容。

7.12.5.1 CPU 占有情况的参照

请单击工具栏的 [Run Time] （执行时间）按钮 （或者选择菜单 [Run Time]）， MR 分析窗口变为 CPU 占有情况的显示模式。



显示各中断处理和任务的 CPU 占有时间和占有率。

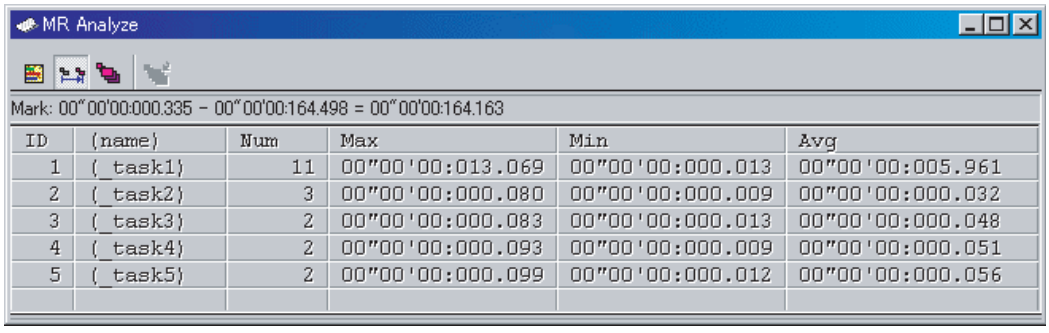
显示内容是由 MR 跟踪窗口的始点标记和终点标记指定的范围内的统计结果。

能通过单击各行的最大执行时间和最小执行时间显示栏，检索所击行对应的任务的最大执行时间和最小执行时间的处理履历。

MR 跟踪窗口的指示器指示检索结果的移动位置。

7.12.5.2 就绪状态时间的参照

请单击工具栏的 [Ready->Run] （就绪 -> 执行）按钮 （或者选择菜单 [Ready->Run]）。



统计并显示各任务从可执行状态转移到执行状态的时间。

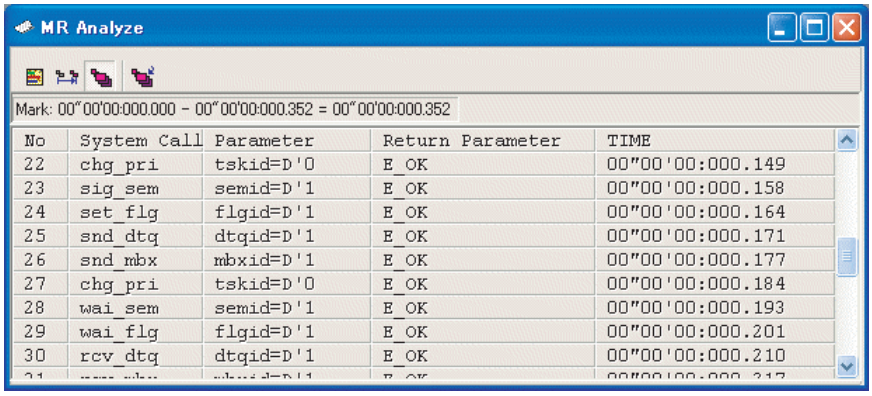
显示内容是由 MR 跟踪窗口的始点标记和终点标记指定的范围内的统计结果。

能通过单击各行的最大就绪时间和最小就绪时间显示栏，检索所击行对应的任务的最大就绪时间和最小就绪时间的处理履历。

MR 跟踪窗口的指示器指示检索结果的移动位置。

7.12.5.3 系统调用发行履历的参照

请单击工具栏的 [System Call] （系统调用）按钮 （或者选择菜单 [System Call]）。

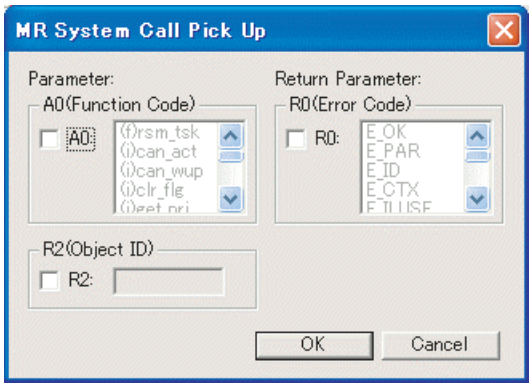


按系统调用的顺序显示被发行的系统调用的列表。
显示内容是由 MR 跟踪窗口的始点标记和终点标记指定的范围内的统计结果。

能通过单击各行，检索所击行对应的系统调用发行履历。
MR 跟踪窗口的指示器指示检索结果的移动位置。

(1) 发行履历的抽取

请从工具栏单击 [Pick Up] （抽取系统调用）按钮 （或者选择菜单 [Pick Up]），打开以下的对话框。请指定要抽取和显示的系统调用的检索条件。



抽取和显示符合指定条件的系统调用发行履历。

8. 脚本命令一览表

本调试器能使用以下的脚本命令。
能在运行时执行黄色显示的脚本命令。
根据产品，不支持后面附带 “*” 的命令。

8.1 脚本命令一览表 （功能顺序）

8.1.1 执行的相关命令

命令名	缩略名	内容
Go	G	执行目标程序。
GoFree	GF	以自由运行方式执行目标程序。
GoProgramBreak*	GPB	带断点执行目标程序 （指定地址）。
GoBreakAt*	GBA	带断点执行目标程序 （指定行号）。
Stop	—	停止执行目标程序。
Status	—	显示目标程序的执行状态。
Step	S	以源行为单位单步执行。
StepInstruction	SI	以机器语言为单位单步执行。
OverStep	O	以源行为单位跨步执行。
OverStepInstruaction	OI	以机器语言为单位跨步执行。
Return	RET	以源行为单位返回执行。
ReturnInstruction	RETI	以机器语言为单位返回执行。
Reset	—	对目标程序进行复位。
Time	—	设置执行时间的显示。

8.1.2 下载的相关命令

命令名	缩略名	内容
Load	L	一次性下载目标程序。
LoadHex	LH	下载机器语言的信息 （Intel HEX 格式文件）。
LoadMot*	LM	下载机器语言的信息 （Motorola S 格式文件）。
LoadSymbol	LS	下载源行 / 汇编符号的信息。
Reload	—	重新下载目标程序。
UploadHex	UH	上载到机器语言信息的 Intel HEX 格式文件。
UploadMot*	UM	上载到机器语言信息的 Motorola S 格式文件。

8.1.3 寄存器操作的相关命令

命令名	缩略名	内容
Register	R	参照和改变指定的寄存器的值。

8.1.4 存储器操作的相关命令

命令名	缩略名	内容
DumpByte	DB	以 1 字节为单位显示存储器的内容。
DumpWord*	DW	以 2 字节为单位显示存储器的内容。
DumpLword*	DL	以 4 字节为单位显示存储器的内容。
SetMemoryByte	MB	以 1 字节为单位更改存储器的内容。
SetMemoryWord*	MW	以 2 字节为单位更改存储器的内容。
SetMemoryLword*	ML	以 4 字节为单位更改存储器的内容。
FillByte	FB	以 1 字节为单位填充存储器的内容。
FillWord*	FW	以 2 字节为单位填充存储器的内容。
FillLword*	FL	以 4 字节为单位填充存储器的内容。
Move	—	以 1 字节为单位传送存储器的内容。
MoveLWord*	MOVEL	以 4 字节为单位传送存储器的内容。
MoveWord*	MOVEW	以 2 字节为单位传送存储器的内容。

8.1.5 汇编 / 反汇编的相关命令

命令名	缩略名	内容
Assemble	A	以 1 行为单位从指定的地址开始汇编。
DisAssemble	DA	显示指定范围的反汇编结果。
Module	MOD	显示全部模块 （目标名）。
Scope	—	显示和更改当前的作用域。
Section	SEC	显示会话信息。
Bit*	—	参照和设置位符号。
Symbol	SYM	显示符号。
Label	—	显示标签。
Express	EXP	显示指定的汇编表达式的值。

8.1.6 软件断点设置的相关命令

命令名	缩略名	内容
SoftwareBreak	SB	显示和设置软件断点。
SoftwareBreakClear	SBC	清除软件断点。
SoftwareBreakClearAll	SBCA	清除全部软件断点。
SoftwareBreakDisable	SBD	使软件断点无效。
SoftwareBreakDisableAll	SBDA	使全部软件断点无效。
SoftwareBreakEnable	SBE	使软件断点有效。
SoftwareBreakEnableAll	SBEA	使全部软件断点有效。
BreakAt	—	用行号指定软件断点。
BreakIn	—	在函数的起始位置指定软件断点。

8.1.7 实时跟踪的相关命令

命令名	缩略名	内容
TraceData*	TD	显示实时跟踪结果的总线信号。
TraceList*	TL	显示实时跟踪结果的反汇编。

8.1.8 事件设置的相关命令

命令名	缩略名	内容
EVent	EV	设置事件。

8.1.9 脚本 / 记录文件的相关命令

命令名	缩略名	内容
Script	—	打开脚本文件。
Exit	—	退出脚本文件。
Wait	—	等待输入命令。
Pause	—	显示指定的信息，等待按钮的输入。
Sleep	—	以指定的秒数等待输入命令。
Logon	—	打开记录文件。
Logoff	—	关闭记录文件。
Exec	—	启动外部的应用程序。

8.1.10 程序显示的相关命令

命令名	缩略名	内容
Func	—	参照函数名，显示函数的内容。
Up*	—	显示调用函数。
Down*	—	显示被调用函数。
Where*	—	显示函数的调用情况。
Path	—	指定源文件的路径。
AddPath	—	添加指定源文件的路径。
File	—	显示指定的源文件。

8.1.11 提供时钟的相关命令

命令名	缩略名	内容
Clock	CLK	参照 MCU 提供的时钟。

8.1.12 C 语言的相关命令

命令名	缩略名	内容
Print	—	参照 C 语言的变量表达式。
Set	—	给 C 语言的变量表达式指定数据。

8.1.13 实时 OS 的相关命令

命令名	缩略名	内容
MR*	—	显示实时 OS(MRxx) 的状态。

8.1.14 实用程序的相关命令

命令名	缩略名	内容
Radix	—	设置和参照常数的规定值。
Alias	—	定义命令的别名，参照命令的定义情况。
UnAlias	—	删除命令的别名定义。
UnAliasAll	—	删除全部命令的别名定义。
Version	VER	显示调试器的版本。
Date	—	显示现在的日期和时间。
Echo	—	显示信息。
CD	—	设置和参照当前的目录。

8.2 脚本命令一览表 （字母顺序）

命令名	缩略名	内容
AddPath	—	添加指定源文件的路径。
Alias	—	定义命令的别名，参照命令的定义情况。
Assemble	A	以 1 行为单位从指定的地址开始汇编。
Bit*	—	参照和设置位符号。
BreakAt	—	用行号指定软件断点。
BreakIn	—	在函数的起始位置指定软件断点。
CD	—	设置和参照当前的目录。
Clock	CLK	参照 MCU 提供的时钟。
Date	—	显示现在的日期和时间。
DisAssemble	DA	显示指定范围的反汇编结果。
Down*	—	显示被调用函数。
DumpByte	DB	以 1 字节为单位显示存储器的内容。
DumpLword*	DL	以 4 字节为单位显示存储器的内容。
DumpWord*	DW	以 2 字节为单位显示存储器的内容。
Echo	—	显示信息。
EVent	EV	设置事件。
Exec	—	启动外部的应用程序。
Exit	—	退出脚本文件。
Express	EXP	显示指定的汇编表达式的值。
File	—	显示指定的源文件。
FillByte	FB	以 1 字节为单位填充存储器的内容。
FillLword*	FL	以 4 字节为单位填充存储器的内容。
FillWord*	FW	以 2 字节为单位填充存储器的内容。
Func	—	参照函数名，显示函数的内容。
Go	G	执行目标程序。
GoBreakAt*	GBA	带断点执行目标程序 （指定行号）。
GoFree	GF	以自由运行方式执行目标程序。
GoProgramBreak*	GPB	带断点执行目标程序 （指定地址）。
Label	—	显示标签。
Load	L	一次性下载目标程序。
LoadHex	LH	下载机器语言的信息 （Intel HEX 格式文件）。
LoadMot*	LM	下载机器语言的信息 （Motorola S 格式文件）。
LoadSymbol	LS	下载源行 / 汇编符号的信息。
Logoff	—	关闭记录文件。
Logon	—	打开记录文件。
Module	MOD	显示全部模块 （目标名）。
Move	—	以 1 字节为单位传送存储器的内容。
MoveLord*	MOVEL	以 4 字节为单位传送存储器的内容。
MoveWord*	MOVEW	以 2 字节为单位传送存储器的内容。
MR*	—	显示实时 OS 的状态。
OverStep	O	以源行为单位跨步执行。

命令名	缩略名	内容
OverStepInstruaction	OI	以机器语言为单位跨步执行。
Path	—	指定源文件的路径。
Pause	—	显示指定的信息，等待按钮的输入。
Print	—	参照 C 语言的变量表达式。
Radix	—	设置和参照常数的规定值。
Register	R	参照和更改指定的寄存器的值。
Reload	—	重新下载目标程序。
Reset	—	对目标程序进行复位。
Return	RET	以源行为单位返回执行。
ReturnInstruction	RETI	以机器语言为单位返回执行。
Scope	—	显示和更改当前的作用域。
Script	—	打开脚本文件。
Section	SEC	显示会话信息。
Set	—	给 C 语言的变量表达式指定数据。
SetMemoryByte	MB	以 1 字节为单位更改存储器的内容。
SetMemoryLword*	ML	以 4 字节为单位更改存储器的内容。
SetMemoryWord*	MW	以 2 字节为单位更改存储器的内容。
Sleep	—	以指定的秒数等待输入命令。
SoftwareBreak	SB	显示和设置软件断点。
SoftwareBreakClear	SBC	删除软件断点。
SoftwareBreakClearAll	SBCA	删除全部软件断点。
SoftwareBreakDisable	SBD	使软件断点无效。
SoftwareBreakDisableAll	SBDA	使全部软件断点无效。
SoftwareBreakEnable	SBE	使软件断点有效。
SoftwareBreakEnableAll	SBEA	使全部软件断点有效。
Status	—	显示目标程序的执行状态。
Step	S	以源行为单位单步执行。
StepInstruction	SI	以机器语言为单位单步执行。
Stop	—	停止执行目标程序。
Symbol	SYM	显示符号。
Time	—	设置执行时间的显示
TraceData*	TD	显示实时跟踪结果的总线信号
TraceList*	TL	显示实时跟踪结果的反汇编。
UnAlias	—	删除命令的别名定义。
UnAliasAll	—	删除全部命令的别名定义。
Up*	—	显示调用函数。
UploadHex	UH	上载到机器语言信息的 Intel HEX 格式文件。
UploadMot*	UM	上载到机器语言信息的 Motorola S 格式文件。
Version	VER	显示调试器的版本。
Wait	—	等待输入命令。
Where*	—	显示函数的调用情况。

9. 脚本文件的记述

脚本文件是为自动执行脚本命令而记述控制语句等的文件。
在脚本窗口中执行脚本文件。

9.1 脚本文件的构成要素

能在脚本文件中记述以下语句：

- 脚本命令
- 赋值语句
- 判断语句 （if、 else、 endi）
  判断表达式的结果，转移到要执行的语句。
- 循环语句 （while、 endw）
  判断表达式的结果，循环执行语句。
- break 语句
  从最内侧的循环执行跳出。
- 注释语句
  脚本文件能记述注释。在执行脚本命令时，忽视注释语句。

在脚本文件中，必须在 1 行中记述 1 个语句，而不能在 1 行中记述多个语句，也不能跨行记述 1 个语句。

注意事项

- 不能在同一行中记述脚本命令的注释。
- 脚本文件的嵌套为 10 层。
- if 语句和 while 语句分别最多能嵌套 32 层。
- 在一个脚本文件中， if 语句和 endi 语句、 while 语句和 endw 语句必须成对使用。
- 用 unsigned 型计算脚本文件中记述的表达式。因此，如果在 if 语句和 while 语句的表达式中比较负值，运行就处于不确定状态。
- 1 行能记述的字符数最多为 4096 个字符。如果执行超过 4096 个字符的行，就会出错。
- 当自动执行有不适当记述的脚本文件时，除了无法加载脚本行本身以外，即使检测到错误也继续执行，直到脚本文件结束为止。此时，在检测到错误后运行处于不确定状态，因此检测到错误后的执行结果缺乏可靠性。

9.1.1 脚本命令

能直接记述在脚本窗口中输入的命令，并且能从脚本文件调用脚本文件 （最多能嵌套 10 层）。

9.1.2 赋值语句

赋值语句进行宏变量的定义、初始化和赋值，记述方法如下所示：

```
% 宏变量名 = 表达式
```

- 宏变量名能使用字母、数字和 （_），但是不能在宏变量名的开头记述数字。
- 给宏变量赋值的表达式能处理的值的范围为 0h ～ FFFFFFFFh 的整数。
  当指定负数时，作为 2 的补码进行处理。
- 能在表达式中使用宏变量。
- 在使用宏变量时，需要在宏变量名的开头加 （%）。

9.1.3 判断语句

判断语句判断表达式的结果，转移到要执行的语句，记述方法如下所示：

```
if （表达式）
  语句 1
else
  语句 2
endi
```

- 当表达式是真 （不为0）时，执行语句1；当表达式是伪 （为0）时，执行语句2。
- 能省略else语句。如果在省略else语句时表达式是伪，就执行endi语句的下一行。
- if语句最多能嵌套32层。

9.1.4 循环语句 （while、 endw）和 break 语句

循环语句判断表达式的结果，循环执行语句，记述方法如下所示：

```
while （ 表达式）
  语句
endw
```

- 当表达式是真时，循环执行语句；当表达式是伪时，从循环跳出 （执行endw的下一个语句）。
- while语句最多能嵌套32层。
- 要强制跳出while语句时，使用break语句。如果While语句有嵌套，就从最内侧的循环跳出。

9.1.5 注释语句

在脚本文件中要记述注释时，使用注释语句，记述方法如下所示：

```
；字符串
```

- 从分号 （;）开始记述，在分号前只能记述空格和制表符。
- 在执行脚本文件时忽视注释语句的行。

9.2 表达式的记述方法

能在指定地址、数据和通过次数时记述表达式。

使用表达式的命令例子如下所示：

```
>DumpByte TABLE1
>DumpByte TABLE1+20
```

能使用以下的表达式的构成要素：

- 常数
- 符号和标签
- 宏变量
- 寄存器变量
- 存储器变量
- 行号
- 字符常数
- 运算符

9.2.1 常数

能输入 2 进制数、 8 进制数、 10 进制数和 16 进制数的数值。在数值的开头或者末尾附加表示基数的记号，区分数值的基数。

用于 M32C、 M16C/R8C、 740 的调试器的情况：

	16 进制数	10 进制数	8 进制数	2 进制数 *2
开头	0x、 0X	@	无	%
末尾	h、 H	无	o、 O	b、 B
例	0xAB24 AB24h	@1234	1234o	%10010 10010b

*2 在基数的规定值为 16 进制数时，只能指定 （%）。

- 在用和规定值相同的基数输入数值时，能省略表示基数的记号 （2进制数除外）。
- 用RADIX命令设置基数的规定值。但是，在输入以下的数据时，与RADIX命令的设置无关，基数为固定值。

种类	基数
地址	16 进制数
行号 执行次数 通过次数	10 进制数

9.2.2 符号和标签

能使用目标程序或者 Assemble 命令定义的符号和标签。

- 符号名和标签名能使用字母、数字、下划线 （_）、点 （.）和问号 （?），但是不能在开头使用数字。
- 符号名和标签名最多能记述255个字符。
- 区分大小写字母。

产品名	注意事项
用于 M32R 的调试器、 用于 R32C 的调试器、 用于 M32C 的调试器、 用于 M16C/R8C 的调试器	• 不能使用汇编的结构体指令、伪指令、宏指令、操作码、保留字。 （.SECTION、 .BYTE、 switch、 if 等） • 以 “..” 开始的字符串不能用于符号名和标签名。

9.2.2.1 局部标签、符号和作用域

本调试器支持了能在程序的全区域内参照的全局标签和全局符号以及只能在声明文件内参照的局部标签和局部符号。

局部标签和符号的有效范围称为作用域，作用域的单位是目标文件。

根据下述情况切换作用域：

- 输入命令的情况
  包含程序计数器指向地址的目标文件为当前的作用域。在用SCOPE命令设置了作用域后，设置的作用域有效。
- 正在执行命令的情况
  根据命令处理的程序地址自动切换当前的作用域。

9.2.2.2 标签和符号的优先顺序

按下述优先顺序进行从值到标签或者符号的转换以及从标签或者符号到值的转换：

- 转换地址值的情况
  1. 局部标签
  2. 全局标签
  3. 局部符号
  4. 全局符号
  5. 作用域范围外的局部标签
  6. 作用域范围外的局部符号

- 转换数据值的情况
  1. 局部符号
  2. 全局符号
  3. 局部标签
  4. 全局标签
  5. 作用域范围外的局部符号
  6. 作用域范围外的局部标签

- 转换位值的情况
  1. 局部位符号
  2. 全局位符号
  3. 作用域范围外的局部位符号

9.2.3 宏变量

用脚本文件中的赋值语句定义宏变量。在使用宏变量时，需要在宏变量名的开头加 （%）。

详细内容请参照 “9.1.2 赋值语句”。

- （%）后面的变量名能使用字母、数字和 （_），但是不能在宏变量名的开头记述数字。
- 变量名不能使用寄存器名。
- 区分变量名的大小写字母。
- 最多能定义255个宏变量。定义的宏变量一直有效到结束调试器为止。

如果利用宏变量指定 while 语句的循环次数，就会很方便。

9.2.4 寄存器变量

在表达式中利用寄存器的值时，使用寄存器变量。在使用寄存器变量时，需要在寄存器名的前面加 （%）。能使用的寄存器名如下所示：

产品名	寄存器名
用于 R32C 的调试器	PC、 USP、 ISP、 INTB、 FLB、 SVF、 SVP、 VCT、 DMD0、 DMD1、 DMD2、 DMD3、 DCT0、 DCT1、 DCT2、 DCT3、 DCR0、 DCR1、 DCR2、 DCR3、 DSA0、 DSA1、 DSA2、 DSA3、 DSR0、 DSR1、 DSR2、 DSR3、 DDA0、 DDA1、 DDA2、 DDA3、 DDR0、 DDR1、 DDR2、 DDR3、 0R0、 0R1、 0R2、 0R3、 0R4、 0R5、 0R6、 0R7、 ← 寄存器组 0 0A0、 0A1、 0A2、 0A3、 0FB、 0SB← 寄存器组 0 1R0、 1R1、 1R2、 1R3、 1R4、 1R5、 1R6、 1R7、 ← 寄存器组 1 1A0、 1A1、 1A2、 1A3、 1FB、 1SB← 寄存器组 1

不区分寄存器名的大小写字母，用哪个指定结果都一样。

9.2.5 存储器变量

在表达式中利用存储器的值时，使用存储器变量，存储器变量的记述方法如下所示：

[地址]. 数据长度

- 能对地址记述表达式 （也能指定存储器变量）。
- 数据长度的指定如下：

数据长	对应调试器	指定
1 字节	全部	B 或者 b
2 字节	用于 M32R 的调试器	H 或者 h
	其他	W 或者 w
4 字节	用于 M32R 的调试器	W 或者 w
	用于 R32C、 M32C、 M16C/R8C 的调试器	L 或者 l

例：以 2 字节长参照地址 8000h 的存储器内容的情况。

```
[0x8000].w
```

- 如果省略数据长度的指定，就为字长。

9.2.6 行号

这是源文件的行号，行号的记述方法如下所示：

```
# 行号
# 行号 ." 源文件名 "
```

- 用10进制数指定行号。
- 只有能设置软件暂停的行，才能指定行号。不能指定如注释行和空行等不生成汇编指令的行。
- 如果省略源文件名，就为现在激活的编辑器 （源）窗口中显示的源文件的行号。
- 还需要给源文件名指定文件属性。
- 不能在行号和源文件名之间插入空格字符。

9.2.7 字符常数

将指定的字符或者字符串转换为 ASCII 码并且作为常数进行处理。

- 字符需用单引号括起来。
- 字符串需用双引号括起来。
- 字符串必须是2个字符以内 （16位长）。如果超过2个字符，处理对象就为所记字符串最后的2个字符。例如，“ABCD”情况的处理对象是字符串最后的2个字符 “CD”，值为4344h。

9.2.8 运算符

表达式能记述的运算符如下所示：

- 运算符的优先级：1为最高，8为最低。当优先顺序相同时，从表达式的左边开始顺序计算。

运算符	含义	优先级
( )	括号	1
+、 –、 ~	一元正、一元负、一元逻辑非	2
*、 /	二元乘法运算、二元除法运算	3
+、 –	二元加法运算、二元减法运算	4
>>、 <<	右移、左移	5
&	二元逻辑与	6
|、 ^	二元逻辑和、二元逻辑异或	7
<、 <=、 >、 >=、 ==、 !=	二元比较	8

10. C/C++ 语言表达式的记述

10.1 C/C++ 语言表达式的记述方法

在注册 C 监视点以及指定给 C 监视点赋的值时，能使用由以下记号构成的 C/C++ 语言表达式。

记号	例
立即值	10、 0x0a、 012、 1.12、 1.0E+3
作用域的解决	::name、 classname::member
四则运算符	+、 –、 *、 /
指针	*、 ** 等
参照	&
符号的取反	—
“.”运算符的成员参照	Object.Member
“->”运算符的成员参照	Pointer->Member、 this->Member
指向成员的指针参照	Object.*var、 Pointer->*var
括号	(、)
数组	Array[2]、 DArray[2][3] 等
向基本型的数据类型转换	(int)、 (char*)、 (unsigned long*) 等
向 typedef 的数据类型转换	(DWORD)、 (ENUM) 等
变量名和函数名	var、 i、 j、 func 等
字符常数	'A'、 'b' 等
字符串文字	"abcdef"、 "I am a boy." 等

10.1.1 立即值

立即值能使用 16 进制数、 10 进制数和 8 进制数的值。以 0x 开始的数值为 16 进制数，以 0 开始的数值为 8 进制数，以其他开始的数值为 10 进制数。要给变量赋值时，还能使用浮点数。

注意

- 不能将立即值注册为 C 监视点。
- 立即值只在指定 C 监视点的 C 语言表达式中使用时以及在指定赋的值时才有效。在使用浮点数时，不能进行 1.0+2.0 等的运算。

10.1.2 作用域的解决

能使用作用域解决运算符 （::），使用例子如下所示：

全局作用域：:: 变量名

```
::x, ::val
```

类指定：类名 :: 成员名、类名 :: 类名 :: 成员名等

```
T::member, A::B::member
```

10.1.3 四则运算符

四则运算符能使用加法运算 （+）、减法运算 （–）、乘法运算 （*）和除法运算 （/），计算的优先顺序如下所示：

(*)、 (/)、 (+)、 (–)

注意

- 现在不支持对浮点进行四则运算。

10.1.4 指针

指针用 “*|” 表示，指针的指针用 “**” 表示，指针的指针的指针用 “***” 表示，……。
使用 “* 变量名”、“** 变量名”等的记述。

注意

- 不能将立即值用作指针。即，不能使用 *0xE000 等。

10.1.5 参照

参照用 “&” 表示，只能使用 “& 变量名”，而不能使用 “&& 变量名”等。

10.1.6 符号的取反

符号的取反用 “-” 表示，只能用 “- 立即值”和 “- 变量名”。如果有 2 个或者 2 个以上的偶数个连续的 “-”，就不将符号取反。

注意

- 现在不支持对浮点变量的符号取反。

10.1.7 “.” 运算符的成员参照

在用 “.” 运算符参照类、结构体、联合体的成员时，只能使用 “变量名 . 成员名”。
例）

```
class T {
public:
int member1;
char member2;
};
class T t_cls;
class T *pt_cls = &t_cls;
```

此时， t_cls.member1 和 (*pt_cls).member2 能正确地参照成员。

10.1.8 “->”运算符的成员参照

在用“->”运算符参照类、结构体、联合体的成员时，只能使用“变量名 -> 成员名”。

例）

```
class T {
public:
int member1;
char member2;
};
class T t_cls;
class T *pt_cls = &t_cls;
```

此时，(&t_cls)->member1 和 pt_cls->member2 能正确地参照成员，并且在成员函数内能参照使用 this->member1 等 this 指针的变量。

10.1.9 指向成员的指针

在用“.*”运算符和“->*”运算符参照指向成员的指针时，只能使用“变量名 .* 成员名”和“变量名 ->* 成员名”。

例）

```
class T {
public:
int member;
};
class T t_cls;
class T *pt_cls = &t_cls;

int T::*mp = &T::member;
```

此时， t_cls.*mp 和 pt_cls->*mp 能正确地参照成员。

注意

- print *mp 的记述不能正确地参照指向成员的指针变量。

10.1.10 括号

在表达式中能使用括号“(”和“)”指定计算的优先顺序。

10.1.11 数组

能使用“[”和“]”指定数组的元素。数组使用“变量名 [元素号或者变量)]”和“变量名 [(元素号或者变量)][(元素号或者变量)]”等的记述。

10.1.12 向基本型的数据类型转换

能使用向 C 基本型的 char 型、short 型、int 型、long 型以及指向这些基本型的指针型的数据类型转换。向指针型的数据类型转换还能使用指针的指针、指针的指针的指针等。另外，没有指定 [signed] （有符号）和 [unsigned] （无符号）时的默认值如下：

基本型	默认值
char	unsigned
short	signed
int	signed
long	signed

注意

- 不能使用向 C++ 基本型的 bool 型、 wchar_t 型、浮点型 （float 型和 double 型）的数据类型转换。
- 不能使用对寄存器变量的数据类型转换。

10.1.13 向 typedef 的数据类型转换

能使用向 typedef 型 （C/C++ 基本型以外的型）以及指向这些型的指针型的数据类型转换。向指针型的数据类型转换还能使用指针的指针、指针的指针的指针等。

注意

- 不能使用向 class 型、 struct 型、 union 型以及指向这些型的指针型的数据类型转换。

10.1.14 变量名

变量名能使用 C/C++ 约定的字母开始的字符串，最大字符数为 255 个字符，还能使用 this 指针变量。

10.1.15 函数名

函数名能使用 C 约定的字母开始的字符串。

注意

- 在 C++ 的情况下，不能使用函数名。

10.1.16 字符常数

能将单引号 （'）括起来的字符用作字符常数，例如 'A'、 'b' 等。将它们转换为 ASCII 码，用作 1 字节的立即值。

注意

- 不能将字符常数注册为 C 监视点。
- 字符常数只在指定 C 监视点的 C/C++ 语言表达式中使用时以及在指定赋的值时才有效 （字符常数的处理和立即值相同）。

10.1.17 字符串文字

能将双引号 （"）括起来的字符串用作字符串文字，例如 "abcde"、 "I am a boy." 等。

注意

- 字符串文字只能记述在赋值运算符的右边，而且只有在赋值运算符的左边是 char 数组或者 char 指针型时才能使用。否则，就会出现语法错误。

10.2 C/C++ 语言表达式的显示格式

显示在 C 监视窗口的数据显示栏中的 C/C++ 语言表达式由该表达式的型名、 C/C++ 语言表达式 （变量名）和计算结果 （值）构成。以下说明各型的显示格式。

10.2.1 枚举型

- 当计算结果的值有定义时，用该名字显示。

  (DATE)date = Sunday(全部Radix)_
- 当计算结果的值没有定义时，显示如下：

  (DATE) date=16(Radix为初始状态的情况)

  (DATE)date=0x10(Radix为16进制数的情况)

  (DATE)date=0000000000010000B(Radix为2进制数的情况)

10.2.2 基本型

- 当计算结果是char型和浮点型以外的基本型时，显示如下：

  (unsigned int) i = 65280(Radix为初始状态的情况)

  (unsigned int) i = 0xFF00(Radix为16进制数的情况)

  (unsigned int) i = 1111111100000000B(Radix为2进制数的情况)
- 当计算结果是char型时，显示如下：

  (unsigned char) c = 'J'(Radix为初始状态的情况)

  (unsigned char) c = 0x4A(Radix为16进制数的情况)

  (unsigned char) c = 10100100B(Radix为2进制数的情况)
- 当计算结果是浮点型时，显示如下：

  (double) d = 8.207880399131839E-304(Radix为初始状态的情况)

  (double) d = 0x10203045060708(Radix为16进制数的情况)

  (double) d = 0000000010.....1000B(Radix为2进制数的情况)

  (.....表示省略)

10.2.3 指针型

- 当计算结果是char*型以外的指针型时，用16进制数显示的内容如下：

  (unsigned int *) p = 0x1234(全部Radix)

- 当计算结果是char*型时，能用C监视窗口的菜单[Display String]（char*的字符串显示）指定字符串/字符的显示。显示例子如下所示：
  - 字符串的显示

    (unsigned char *) str = 0x1234 "Japan"(全部Radix)
  - 字符的显示

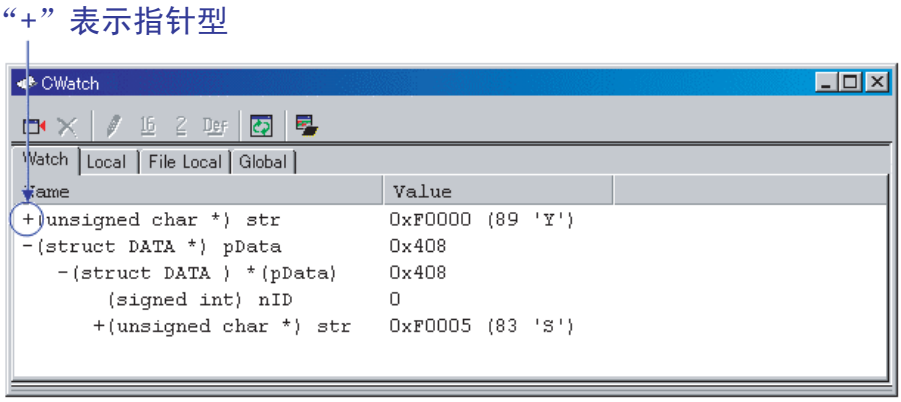
    (unsigned char *) str = 0x1234 (74 'J')(全部Radix)

在显示字符串时，如果在表示字符串结束的代码（0）前含有无法显示字符的代码，就不输出后面的双引号（"），如下所示：

(unsigned char *) str = 0x1234 "Jap(全部Radix)

在字符串长超过 80 个字符时，也同样不输出后面的双引号（"）。

当 C/C++ 语言表达式是指针型时，型名的左侧出现如下所示的“+”记号。



当双击显示“+”的行时，就显示该指针的目标。显示目标后，“+”变为“–”。当双击显示“–”的行时，就恢复原来的状态。同样，也能参照列表结构和树结构等的数据。

10.2.4 数组型

- 当计算结果是char[]型以外的数组型时，用16进制数显示的起始地址如下：

  (signed int [10]) z = 0x1234(全部Radix)

- 当计算结果是char[]型时，显示如下：

  (unsigned char [10]) str = 0x1234 "Japan"(全部Radix)

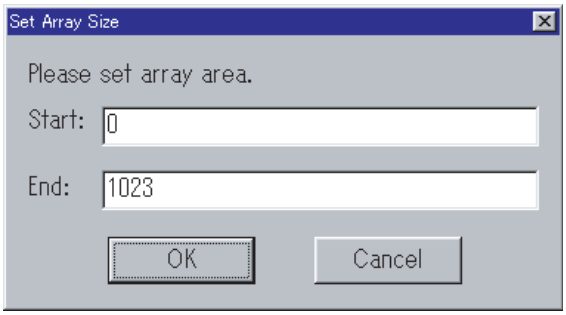
如果在表示字符串结束的代码（0）前含有无法显示字符的代码，就不输出后面的双引号（"），如下所示：

(unsigned char [10]) str = 0x1234 "Jap(全部Radix)

在字符串长超过 80 个字符时，也同样不输出后面的双引号（"）。

当 C/C++ 语言表达式是数组型时，就和指针型一样，型名的左侧出现“+”记号。展开方法和指针型相同，详细内容请参照 “10.2.3 指针型”。

当数组的大小超过 100 时，下述对话框就会打开，请指定要展开的元素个数。



显示用 Start 指定的元素到用 End 指定的元素。

如果指定值超过了数组元素个数的最大值，就为数组的最大值。

在单击 [Cancel]（取消）按钮后，数组不展开。

10.2.5 函数型

- 当计算结果是函数型时，用16进制数显示函数的起始地址如下：

  (void()) main = 0xF000(全部Radix)_

10.2.6 参照型

- 当计算结果是参照型时，用16进制数显示的参照地址如下：

  (signed int &) ref = 0xD038(全部Radix)

10.2.7 位域型

- 当计算结果是位域型时，显示如下：

  (unsigned int :13) s.f = 8191(Radix为初始状态的情况)

  (unsigned int :13) s.f = 0x1FFF(Radix为16进制数的情况)

  (unsigned int :13) s.f = 1111111111111B(Radix为2进制数的情况)

10.2.8 没有发现 C 符号的情况

- 当计算式中存在不能发现的C符号时，显示如下：

  () x = <not active>(全部Radix)

10.2.9 语法错误

- 当计算式出现语法错误时，显示如下：

```
() str*(p = <syntax error>(全部Radix)
(str*(p是语法错误)
```

10.2.10 结构体型和联合体型

- 当计算结果为结构体型或者联合体型时，用16进制数显示的地址如下：

```
(Data) v = 0x1234(全部Radix)
```

当 C/C++ 语言表达式有结构体型或者联合体型的成员时，型名 （标志名）的左侧出现 “+” 记号。

当双击显示 “+” 的行时，就显示该结构体 （或者联合体）的成员。

显示成员后，“+” 变为 “–”。当双击显示 “–” 的行时，就恢复原来的状态。同样，也能参照成员的数据。

注意

如果声明的变量名和用 typedef 声明的型定义名同名，就不能参照此变量。

- 寄存器变量

  当计算结果是寄存器变量时，就在型名的开头显示 “register”，如下所示：

```
(register signed int) j = 100
```

11. 程序停止原因的显示

在通过调试功能停止执行程序时，在输出窗口和状态窗口 （[Platform] （平台））中显示该停止原因。
停止原因的显示内容及其含义如下：

显示	停止原因
[Halt] （停止）	使用 [Halt Program] （停止程序）按钮或者菜单的停止
[Program] （程序）	程序暂停 （软件暂停）或者指令的暂停执行 （事件暂停）
[Event, Data Access] （时间、数据、存储）	存储器的暂停存取 （事件暂停）

12. 注意事项

12.1 产品的共同注意事项

12.1.1 Windows 上的文件操作

有关 Windows 上的文件操作，请注意以下几点：

1. 文件名和目录名
   - 不能使用汉字的文件名和目录名。
   - 不能使用有2个 “.”（点）以上的文件。
2. 文件和目录的指定
   - 不能使用 “...”（ 指定2个上层的目录）。
   - 不能使用网络路径名。要使用网络路径名时，请映射到网络驱动器后使用。

12.1.2 能设置软件断点的区域

能设置软件断点的区域因产品而不同。

12.1.2.1 用于 R32C 和 M32C 的调试器

能对 MCU 的内部 RAM 区域、 MCU 的内部闪存 ROM 区域和用户目标上的 RAM 区域设置软件断点。

注意事项

- 在对 MCU 的内部闪存 ROM 区域指定断点时，需要以块为单位改写或者回写指令，因此不能在目标程序执行过程中设置软件断点。
- 对于用户目标上的 ROM 区域等不能改写指令的区域，不能设置软件断点。

12.1.3 C 变量的参照和设置

- 如果声明的变量名和用typedef声明的型定义名同名，就不能参照此变量。
- 不能给寄存器变量赋值。
- 不能给64位长的变量 （long long型或者double型等）赋值。
- 不能给没有指示存储器的实体 （地址和大小）的变量赋值。
- 在多个局部变量因编译程序的最优化而被分配到相同的区域时，可能无法正确显示该变量的值。
- 不能将文字字符串赋值给char数组或者char指针型以外的变量。
- 不能对浮点数进行四则运算。
- 不能对浮点型变量进行符号取反。
- 不能进行向浮点型的数据类型转换。
- 不能对寄存器变量进行数据类型转换。
- 不能进行向结构体型、联合体型以及指向这些型的指针型的数据类型转换。
- 不能给字符常数和文字字符串记述转义序列。
- 能给位域成员赋以下的值：
  - 整数常数、字符常数、枚举符
  - bool型变量、字符型变量、整数型变量、枚举型变量
  - 位域成员

如果上述的值超过位域成员的位长，就舍去超出的值 （高位）后进行赋值。

- 如果读存储器，分配给SFR区域的位域成员的值就不能正确地被更改。
- 在程序执行过程中，不能更改局部变量和位域成员的值。

12.1.4 C++ 的函数名

- 如果在设置断点时用函数名设置地址，就不能使用类的成员函数、 operator 函数和过载 （多重定义）函数。
- 不能用函数名记述C/C++语言表达式。
- 不能使用给参数指定函数的脚本命令 （breakin、 func等）。
- 在地址值设置栏中不能指定使用函数名的地址。
- 在C监视窗口中不能正确地参照指向成员函数的指针。

12.1.5 目标程序的下载设置

本调试器不对应以下的注册下载模块时设置的选项：

- [Offset] （偏移量）：总是为 “0”。
- [Access size] （存取长度）：总是为 “1”。
- [Perform memory verify during download] （下载时的存储器验证）：无。

12.1.6 多模块的调试

不能给 1 个会话注册多个绝对模块文件，也不能同时下载多个文件，但是能同时下载 1 个绝对模块文件和多个机器语言文件。

12.1.7 同步调试

本调试器不对应同步调试。

12.1.8 固件的下载

需要给仿真器下载对应的固件。

调试器在启动时检查仿真器内部的固件版本。

如果发现仿真器内没有需要的固件文件，就在启动时进入仿真器强制下载固件的模式。

12.1.9 ID 码的检查

在启动调试器时，检查被写在闪存内的 ID 码。

在闪存内写有 ID 码时，显示输入 ID 码的对话框。请在显示的对话框中正确输入 ID 码。如果 ID 码不同，就不启动调试器。

12.1.10 调试器的启动步骤

按照以下的某个步骤启动调试器：

- 接通仿真器和目标的电源->启动调试器->Init对话框的设置->目标的硬件复位
- 接通仿真器电源->启动调试器->Init对话框的设置->接通目标电源

在需要对目标进行硬件复位或者接通目标电源时，会显示提示对话框。此时，请在对目标进行硬件复位或者接通目标电源后单击 [OK] （确定）按钮。

12.1.11 监视定时器的使用

本调试器能调试使用 MCU 的内部监视定时器的程序。只有在监视定时器运行时，仿真器的监视程序才能进行刷新。

12.1.12 MCU 内部的调试资源

MCU 内部的暂停事件与跟踪事件和时间测量事件共享，共计 8 点。

12.1.13 MCU 的内部 RAM 区域的断点设置

为了有效利用硬件断点资源，请给 MCU 的内部 RAM 区域设置软件断点。

12.2 用于 R32C 的调试器的注意事项

12.2.1 仿真器使用的堆栈区

仿真器将中断堆栈区用作工作区域 （最多 52 字节）。
在调试时，请确保通常使用的大小 +52 字节中断堆栈区。

12.2.2 目标程序复位时的中断堆栈指针

在目标程序复位时，仿真器将中断堆栈指针 （ISP）设置为 “0500h”。
请注意：对于正式的产品，中断堆栈指针 （ISP）为 “0000h”。

12.2.3 数据比较暂停功能使用的 RAM 区域

请注意：在通过事件 E5 使用数据比较暂停功能时，使用 MCU 内部 RAM 的开头 （0400h）的 8 字节。

12.2.4 编译程序 / 汇编程序 / 连接程序的选项

在调试时，需要考虑编译和连接时的选项设置。
设置的内容请参照 “12.3 编译程序 / 汇编程序 / 连接程序的选项”。

用于 R32C 的调试器能使用的编译程序：

- 本公司产C编译程序NCxx
- IAR公司产EC++编译程序
- IAR公司产C编译程序

12.3 编译程序 / 汇编程序 / 连接程序的选项

在调试时，需要考虑编译和连接时的选项设置。
设置的内容请参照以下的章节。
不评估也不推荐除此以外的设置。

12.3.1 本公司产 C 编译程序 NCxx 的使用

如果在编译时指定 “-O”、“-OR”、“-OS”选项，由于进行最优化处理，所以就可能不能正确地生成源行信息并且不能正确地进行单步执行等。

为了避免此问题，请在指定 “-O”、“-OR”、“-OS”选项的同时，也指定 “-ONBSD”（或者 Ono_Break_source_debug）选项。

12.3.2 IAR 公司产 EC++ 编译程序的 Workbench （EW）使用

请按以下的步骤设置工程：

1. 通过IAR Embedded Workbench设置工程。
   一旦选择菜单[Project]→[Options...]，就打开Options For Target"xxx"对话框。
   请在此对话框的Category中选择XLINK并进行以下的设置：
   - Output选项卡
     在Format栏中选定Other，在Output format中选择elf/dwarf。
   - Include选项卡
     请在XCL file name栏中指定要使用的XCL文件 （例：lnkm32cf.xcl）。
2. XCL文件的编辑
   请给所使用的XCL文件追记-y选项， -y选项的指定因产品而不同。

产品名	-y 选项
用于 R32C 的调试器	-yspc

3. 程序的创建
   请在进行上述设置后创建目标程序。

不运行评估也不推荐除此以外的设置。

12.3.3 IAR 公司产 C 编译程序的 Workbench(EW) 使用

请按以下的步骤设置工程：

1. 通过IAR Embedded Workbench设置工程。
   一旦选择菜单[Project]→[Options...]，就打开Options For Target"xxx"对话框。
   请在此对话框的Category中选择XLINK并进行以下的设置：
   - Output选项卡
     在Format栏中选定Other，在Output format中选择ieee-695。
   - Include选项卡
     请在XCL file name栏中指定要使用的XCL文件 （例：lnkm16c.xcl）。
2. XCL文件的编辑
   请给所使用的XCL文件追记 “-y”选项，“-y”选项的指定因产品而不同。

产品名	-y 选项
用于 R32C 的调试器	-ylmba

3. 程序的创建
   请在进行上述设置后创建目标程序。

不运行评估也不推荐除此以外的设置。

12.3.4 IAR 公司产 C 编译程序的命令行使用

12.3.4.1 选项指定

请按以下的步骤进行编译和连接：

- 编译时
  请指定 “-r” 选项。
- 连接前
  请打开连接时加载连接的程序选项定义文件 （扩展名.xcl），添加 “-FIEEE695” 和 “-y” 选项。
  “-y” 选项的指定因产品而不同。

产品名	-y 选项
用于 R32C 的调试器	-ylmba

- 连接时
  请用 “-f” 选项指定接程序选项定义文件。

不运行评估也不推荐除此以外的设置。

12.3.4.2 命令输入例子

命令输入例子如下所示：

- 用于R32C的调试器

```
>ICCR32C-r file1.c<Enter>
>ICCR32C-r file2.c<Enter>
>XLINK-o filename.695-f lnkr32c.xcl file1 file2<Enter>
```

XCL 文件名因产品和存储器模型而不同，详细内容请参照 ICCxxxx 的手册。

修订记录	R32C/100 E30A 仿真调试器 V.1.01 用户手册

Rev.	发行日	修订内容	
		页	修订处
1.00	2010.08.06	—	初版发行

R32C/100 E30A 仿真调试器 V.1.01
用户手册

Publication Date: Rev.1.00 Aug 06, 2010

Published by: Renesas Electronics Corporation

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

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

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

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

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

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

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

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

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

Renesas Electronics Taiwan Co., Ltd.
7F, No. 363 Fu Shing North Road Taipei, Taiwan, R.O.C.
Tel: +886-2-8175-9600, Fax: +886 2-8175-9670

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

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

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

Colophon 1.0

R32C/100 E30A仿真调试器V.1.01

RENESAS

瑞萨电子株式会社

RCJ10J0091-0100