To our customers,

Old Company Name in Catalogs and Other Documents

On April 1st, 2010, NEC Electronics Corporation merged with Renesas Technology Corporation, and Renesas Electronics Corporation took over all the business of both companies. Therefore, although the old company name remains in this document, it is a valid Renesas Electronics document. We appreciate your understanding.

Renesas Electronics website: http://www.renesas.com

April 1st, 2010
Renesas Electronics Corporation

Issued by: Renesas Electronics Corporation (http://www.renesas.com)

Send any inquiries to http://www.renesas.com/inquiry.

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.

用户手册

RX850 版本 3.20

实时操作系统

安装手册

目标工具

RX850 版本 3.20

文件编号 U17419CA1V0UM (第一版)

发布日期 2005 年 4 月 CP(k)

日本印刷

[备忘录]

Windows 和 Windows NT 是在美国及其他国家微软公司的注册商标或商标。

Green Hills Software 和 MULTI 是 Green Hills 软件公司的注册商标。

UNIX 是 X/Open 公司在美国及其他国家注册的商标。

PC/AT 是 IBM 公司的注册商标。

TRON 是 The Real-time Operating system Nucleus 的缩写。

ITRON 是 Industrial TRON 的缩写。

凹TRON 是 Micro Industrial TRON 的缩写。

- **本文档信息先于产品的生产周期发布。将来可能未经预先通知而更改。在实际进行生产设计时，请参阅各产品最新的数据表或数据手册等相关资料以获取本公司产品的最新规格。**
- 并非所有的产品和/或型号都向每个国家供应。请向本公司销售代表查询产品供应及其他信息。
- 未经本公司事先书面许可，禁止复制或转载本文件中的内容。本文件所登载内容的错误，本公司概不负责。
- 本公司对于因使用本文件中列明的本公司产品而引起的，对第三者的专利、版权以及其它知识产权的侵权行为概不负责。本文件登载的内容不应视为本公司对本公司或其他人所有的专利、版权以及其它知识产权作出任何明示或默示的许可及授权。
- 本文件中的电路、软件以及相关信息仅用以说明半导体产品的运作和应用实例。用户如在设备设计中应用本文件中的电路、软件以及相关信息，应自行负责。对于用户或其他人因使用了上述电路、软件以及相关信息而引起的任何损失，本公司概不负责。
- 虽然本公司致力于提高半导体产品的质量及可靠性，但用户应同意并知晓，我们仍然无法完全消除出现产品缺陷的可能。为了最大限度地减少因本公司半导体产品故障而引起的对人身、财产造成损害（包括死亡）的危险，用户务必在其设计中采用必要的安全措施，如冗余度、防火和防故障等安全设计。
- 本公司产品质量分为：

  “标准等级”、“专业等级”以及“特殊等级”三种质量等级。

  “特殊等级”仅适用于为特定用途而根据用户指定的质量保证程序所开发的日电电子产品。另外，各种日电电子产品的推荐用途取决于其质量等级，详见如下。用户在选用本公司的产品时，请事先确认产品的质量等级。

  “标准等级”： 计算机，办公自动化设备，通信设备，测试和测量设备，音频・视频设备，家电，加工机械以及产业用机器人。

  “专业等级”： 运输设备（汽车、火车、船舶等），交通用信号控制设备，防灾装置，防止犯罪装置，各种安全装置以及医疗设备（不包括专门为维持生命而设计的设备）。

  “特殊等级： 航空器械，宇航设备，海底中继设备，原子能控制系统，为了维持生命的医疗设备、用于维持生命的装置或系统等。

  除在本公司半导体产品的数据表或数据手册等资料中另有特别规定以外，本公司半导体产品的质量等级均为“标准等级”。如果用户希望在本公司设计意图以外使用本公司半导体产品，务必事先与本公司销售代表联系以确认本公司是否同意为该项应用提供支持。

（注）

（1） 本声明中的“本公司”是指日本电气电子株式会社（NEC Electronics Corporation）及其控股公司。

（2） 本声明中的“本公司产品”是指所有由日本电气电子株式会社或为日本电气电子株式会社（定义如上）开发或制造的产品。

M5 02.11-1

区域信息

本文档中的某些信息可能因国家不同而有所差异。用户在使用任何一种 NEC 产品之前，请与当地的 NEC 办事处联系，以获取权威的代理商和发行商信息。请验证以下内容：

- 设备的可用性
- 定货信息
- 产品发布进度表
- 相关技术资料的可用性
- 开发环境要求（例如：要求第三方工具和组件，主计算机，电源插头，AC 供电电源等）
- 网络要求

此外，对于商标、注册商标、出口限制条款和其他法律规定，不同的国家也有不同的要求。

详细信息请联系：

（中国区）

网址：
http://www.cn.necel.com/
http://www.necel.com/

[北京]

日电电子（中国）有限公司
中国北京市海淀区知春路 27 号
量子芯座 7，8，9，15 层
电话：(+86)10-8235-1155
传真：(+86)10-8235-7679

[上海]

日电电子（中国）有限公司上海分公司
中国上海市浦东新区银城中路 200 号
中银大厦 2409-2412 和 2509-2510 室
电话：(+86)21-5888-5400
传真：(+86)21-5888-5230

上海恩益禧电子国际贸易有限公司
中国上海市浦东新区银城中路 200 号
中银大厦 2511-2512 室
电话：(+86)21-5888-5400
传真：(+86)21-5888-5230

[深圳]

日电电子（中国）有限公司深圳分公司
深圳市福田区益田路卓越时代广场大厦 39 楼
3901，3902，3909 室
电话：(+86)755-8282-9800
传真：(+86)755-8282-9899

[香港]

香港日电电子有限公司
香港九龙旺角太子道西 193 号新世纪广场
第 2 座 16 楼 1601-1613 室
电话： (+852)2886-9318
传真： (+852)2886-9022
2886-9044

[备忘录]

引言

读者对象 本手册适用于那些设计和开发 V850 系列处理器应用系统的用户。

目的 本手册说明 RX850 的功能。

本文组织 本手册包含以下几个主要部分:

- 概述
- 安装
- 系统构建
- 内存及容量估算
- 系统配置文件
- 操作配置器

如何阅读本手册 本文读者应掌握具备电子工程、逻辑电路、微控制器、C 语言和汇编语言方面的一般知识。

如需了解 V850 系列微处理器的硬件功能

→参阅各个产品的**硬件用户手册**。

如需了解 V850 系列微处理器的指令功能

→参阅 **V850ES 架构用户手册(U15943E)或 V850E1 架构用户手册(U14559E)**。

约定

数据含义: 数据的高位部分在左边，低位部分在右边。

注: 本文中脚注用**注**标识

注意事项: 需要特别关注的信息

备注: 补充信息

数值表示法: 二进制...xxxx 或 xxxxB

十进制...xxxx

十六进制...0xXXXX

前缀表示 2 的指数 (地址空间和内存容量):

K(kilo) $2^{10} = 1024$

M(mega) $2^{20} = 1024^2$

相关文档　与阅读本手册相关的文档如下。

在本文档中涉及的相关文档可能包含其初始版本。

但初始版本并没有像这样标记。

和开发工具相关的文档(用户手册)

文档名称		文档编号
CA850 C 语言编译器包 3.00 版本	操作	U17293E
	C 语言	U17291E
	汇编语言	U17292E
	Link 指令	U17294E
ID850 版本 3.00 集成调试器	操作	U17358E
ID850NW3.00 版本，3.10 集成调试器	操作	U17369E
ID850NWC 2.51 版本 集成调试器	操作	U16525E
ID850QB 3.10 版本 集成调试器	操作	U17435E
SM+系统仿真器	操作	U17246E
	用户开放接口	U17247E
SM8502.50 版本系统仿真器	操作	U16218E
SM8502.00 版本或以上版本 系统仿真器	外部用户开放接口规范	U14873E
RX850 3.20 版本 实时操作系统	基本	U13430E
	安装	本手册
	技术	U13431E
	任务调试器	U17420E
AZ8503.30 版本系统性能分析器		U17423E
PG-FP4 闪存编程器		U15260E
TW8502.00 版本性能分析调试工具		U17421E
PM+6.00 版本项目管理器		U17178E

目录

图列表

表格列表

第1章　概述

1.1 概述

半导体技术的迅速发展使得微处理器应用取得了爆炸式的发展，其应用领略之广，超乎几年前的想象。伴随这种发展，微处理器所必须的处理程序也在不停的增长，这种增长规则使得针对某一特定硬件开发处理程序变得很困难。

因此，需要一种能充分利用最新一代高性能、多功能的微处理器能力的操作系统。

然而，控制操作系统是内嵌在控制器单元中的，也就是说，因为不同系统具有不同的硬件配置，且需要匹配应用的高效操作指令，因此通常标准的操作系统不能简单应用到控制操作系统的应用环境中。

针对这个市场背景，为了利用其高端 V850 系列微处理器的功能及性能，和支持今后软件的体系化组织，日电电子开发和发布了 RX850 操作系统。

RX850 是一个嵌入式实时多任务控制操作系统，它提供了一个高效率实时多任务环境以扩大处理控制单元的应用范围。

RX850 是一个高速，能够存储在目标系统的 ROM 中和从 ROM 运行的紧凑型操作系统。

1.2 特征

RX850 有以下特征:

1) 符合 µITRON 规范

   RX850 的设计符合 µITRON 3.0 规范，µITRON 3.0 规范定义了一个标准的嵌入式控制操作系统结构。RX850 实现了 µITRON 3.0 的 S 级功能。

   µITRON 3.0 规范适用于一个嵌入式实时控制操作系统。

2) 高通用性

   为了提供较好的应用系统通用性，RX850 支持 µITRON 3.0 规范规定的所有系统调用。

   由于应用系统所用的功能(系统调用)是可选的，可以利用 RX850 创建一个满足用户需求的紧凑的和最优化的实时多任务操作系统。

3) 实时处理技术和多任务的实现

   RX850 支持以下功能来彻底地实现实时处理和多任务:

   - 任务管理功能
   - 任务关联同步功能
   - 同步通信功能
   - 中断管理功能
   - 内存池管理功能
   - 时间管理功能
   - 系统管理功能
   - 调度功能

4) 调度锁定功能

   RX850 通过用户处理程序来支持禁用和恢复调度功能(任务调度)

5) 紧凑型设计

   RX850 是一个基于能嵌入目标系统的假设而设计的实时多任务的操作系统，因此为了能加载到目标系统的 ROM 中，它被设计的尽量的紧凑。

6) 独特指令的利用

V850 系列微处理器的高速执行速度和独特指令的配合，使系统能获得高速处理速度。

7) 实用工具支持

RX850 系统提供以下实用工具来辅助系统构建:

- 配置器 CF850
- 任务调试器 RD850
- 系统性能分析器 AZ850

[注意事项] RX850 的任务调试器叫 RD850。

1.3 执行环境

RX850 被开发为一个嵌入式控制操作系统，它能运行在配备以下硬件的目标系统中:

1) 目标 CPU

V851, V852, V853, V854

V850/SA1, V850/SBx, V850/SV1, V850 core, V850EI core, V850E2 core, V850ES core, V850E/MS1, V850E/MA1, NB85E core, V850E/IA1

2) 周边控制器

为了支持一系列的执行环境，RX850 从内核中去掉了依赖于硬件的部分，把这些部分是作为示例源文件提供。如果针对各种目标系统对这些示例源程序进行改写，那么就不在需要特定的周边控制器。

1.4 开发环境

这节主要介绍开发应用系统需要的软硬件环境。

1.4.1 硬件环境

1) 主机

- PC/AT 兼容机

2) 在线电路仿真器

- IE-703002-MC (V851, V852, V853, V854, V850/SA1, V850/SBx, V850/SV1)
- IE-703102-MC (V850E/MS1)
- IE-V850E-MC-A (V850E/MA1, NB85E core)
- IE-V850E-MC (V850E/IA1)

3) 在线电路仿真器 I/O 板

- IE-703003-MC-EM1 (V853)
- IE-703008-MC-EM1 (V854)
- IE-703017-MC-EM1 (V850/SA1)
- IE-703037-MC-EM1 (V850/SBx)
- IE-703040-MC-EM1 (V850/SV1)
- IE-703102-MC-EM1 (V850E/MS 1.5V)
- IE-703102-MC-EM1-A (V850E/MS1 3.3V)
- IE-703107-MC-EM1 (V850E/MA1)

- IE-703116-MC-EM1 (V850E/IA1)
- IE-V850E-MC-EM1-A (NB85E core 5V)
- IE-V850E-MC-EM1-B (NB85E core 3.3V)

[注意事项] 这些 I/O 板必须与在线电路仿真器配合使用。

4) PC 接口板

- IE-70000-PC-IF-C (PC/AT 兼容机 ISA 总线)
- IE-70000-CD-IF-A (PCMCIA 槽)
- IE-70000-PCI-IF (PCI 总线)

1.4.2 软件环境

1) 操作系统(():主机)

- Windows 98, Me, NT4.0, 2000, XP (PC/AT-兼容机)

2) 交叉工具

- CA850(日电电子公司)
- CCV850/CCV850E (Green Hills Software Inc)

3) 调试器

- ID850 (日电电子公司)
- SM850 (日电电子公司)
- MULTI (Green Hills Software Inc.)
- PARTNER (Kyoto Micro Computer Co.,Ltd.)

4) 任务调试器

- RD850 (日电电子公司)

5) 系统性能分析器

- AZ850(日电电子公司)

第 2 章　安装

本章主要介绍如何安装和卸载 RX850

2.1　安装

本节介绍如何安装 Windows 版本 RX850，在重新安装干净的 RX850 前，必须先卸载以前的安装程序。

不论是目标发布版本，还是源发布版本，RX850 都通过一张光盘提供。RX850 软件包中包含了 RD850、AZ850 及它们的在线帮助文件。也可以同时安装这些程序。

RX850 的安装步骤如下:

1) 启动 Windows 操作系统
2) 在光驱中插入 RX850 光盘后，安装程序将自动运行。如果安装程序没有自动运行，打开浏览器，双击光驱中的”INSTALL.EXE”。然后，根据显示的信息执行安装。
3) 通过 Windows 应用程序资源管理器等，检查存储在 RX850 发布媒质中的文件已经安装到主机上。

   关于每个文件夹的详细情况，参考”2.3 文件夹配置”。

2.2　卸载

RX850 卸载步骤如下:

1) 启动 Windows 操作系统
2) 打开控制面板中的”添加/删除程序”，选择要下载的项("NEC EL RX850 NEC EL(object release version)"等)并卸载。

2.3 文件夹配置

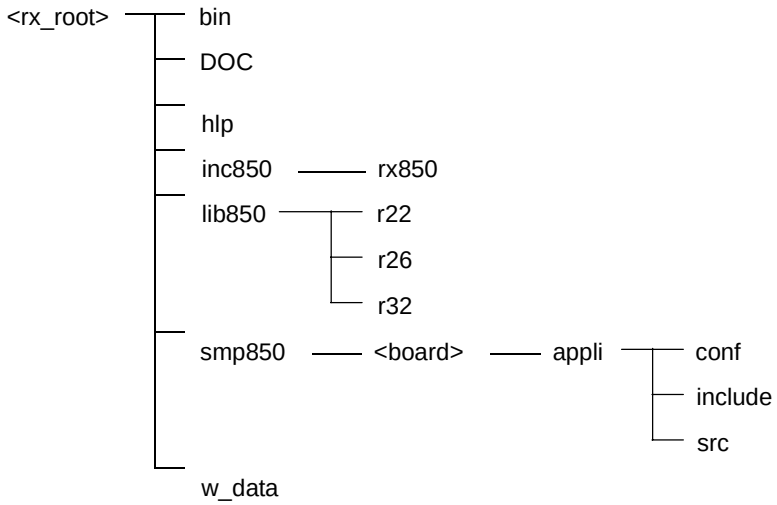
本节介绍 RX850 安装后，从发布媒质中读出的文件的文件夹配置情况。RX850 以目标发布版或源发布版形式提供，每个版本都有 CA850 版和 GHS 编译版。

- 目标发布版本/CA850
- 目标发布版本/GHS 编译版
- 源发布版本/CA850 版
- 源发布版本/GHS 编译版

2.3.1 目标发布版本/CA850 版

图 2-1 示出了当存放在 RX850 发布媒质中的文件(目标文件发布版本/CA850 版)安装后的文件夹配置。

图 2-1 文件夹配置 (目标发布版本/CA850 版本)



下面是每个文件夹的详细说明:

1) <rx_root>

这个文件夹是在安装时定义的 RX850 安装文件夹。

2) <rx_root>\bin

这个文件夹存储了 RX850 的实用工具。

cf850.exe : CF850 配置程序

rx703100p.dll :CF850 链接指令文件

3) <rx_root>\DOC

这个文件夹存储了 RX850 的文档文件。

4) <rx_root>\hlp

这个文件夹存储了 RX850 的在线帮助文件。

5) <rx_root>\inc850

这个文件夹存储了 RX850 的标准头文件。

stdrx850.h :标准头文件(C 语言)

stdrx850.inc :标准头文件(汇编语言)

6) <rx_root>\inc850\rx850

这个文件夹存储了 RX850 的头文件。

7) <rx_root>\lib850_ghs\r22

这个文件夹存储了 RX850 的库文件(22 位寄存器)。

librx.a :内核函数库

8) <rx_root>\lib850_ghs\r26

这个文件夹存储了 RX850 的库文件(26 位寄存器)。

librx.a :内核函数库

9) <rx_root>\lib850_ghs\r32

这个文件夹存储了 RX850 的库文件(32 位寄存器)。

librx.a :内核函数库

10) <rx_root>\smp850_ghs\<board>

这个文件夹存储了 RX850 的示例程序。

11) <rx_root>\smp850_ghs\<board>\appli\conf

这个文件夹存储了生成 RX850 加载模块的命令文件。

加载模块”sample.out”可以用命令文件生成到该文件夹。

sample.prj :生成加载模块的工程文件

sample.prw :生成加载模块的工作空间文件

12) <rx_root>\smp850_ghs\<board>\appli\include

这个文件夹存储了示例程序的头文件

13) <rx_root>\smp850_ghs\<board>\appli\src

这个文件夹存储了示例程序的源文件和链接指令文件

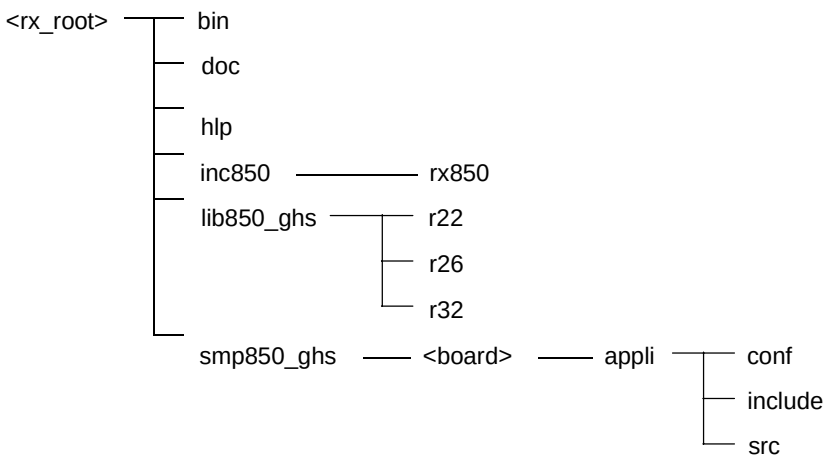
14) <rx_root>\w_data

当采用 PM＋ RX850 时，这个文件夹存储了示例的链接指令文件

2.3.2 目标发布版本/GHS 编译器版

图 2-2 示出了当存放在 RX850 发布媒质中的文件(目标发布版本/GHS 编译器版)安装后的文件夹配置。

图 2-2 文件夹配置 (目标发本版本/GHS 编译器版)



下面是各个文件夹的详细说明:

1) <rx_root>

这个文件夹是在安装时定义的 RX850 安装文件夹。

2) <rx_root>\bin

这个文件夹存储了 RX850 的实用工具。

cf850_ghs.exe ：配置器 CF850

3) <rx_root>\DOC

这个文件夹存储了 RX850 的文档文件

4) <rx_root>\hlp

这个文件夹存储了 RX850 的在线帮助文件

5) <rx_root>\inc850

这个文件夹存储了 RX850 的标准头文件。

stdrx850.h ： 标准头文件

6) <rx_root>\inc850\rx850

这个文件夹存储了 RX850 的头文件。

7) <rx_root>\lib850_ghs\r22

这个文件夹存储了 RX850 的函数库文件(22 位寄存器)。

librx.a ： 内核函数库

8) <rx_root>\lib850_ghs\r26

这个文件夹存储了 RX850 的函数库文件(26 位寄存器)。

librx.a ： 内核函数库

9) <rx_root>\lib850_ghs\r32

这个文件夹存储了 RX850 的库文件(32 位寄存器)。

librx.a ： 内核函数库

10) <rx_root>\smp850_ghs\<board>

这个文件夹存储了 RX850 的示例程序

11) <rx_root>\smp850_ghs\<board>\appli\conf

这个文件夹存储了生成 RX850 加载模块的命令文件。

加载模块”sample.out”可以用命令文件生成到这个文件夹。

sample.bld : 加载模块建立文件

12) <rx_root>\smp850_ghs\<board>\appli\include

这个文件夹存储了示例程序的主文件。

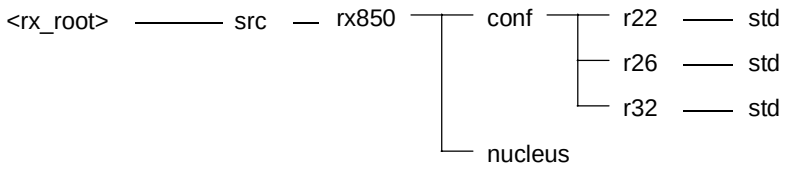
13) <rx_root>\smp850_ghs\<board>\appli\src

这个文件夹存储了示例程序的源文件和连接指令文件。

2.3.3 源发布版本/CA850 版

图 2-3 示出了当存放在 RX850 发布媒质中的文件(源发布版本/CA850 版)安装后的文件夹配置。

图 2-3 文件夹配置(源发布版本/CA850 版)



下面是各个文件夹的详细说明:

1) <rx_root>

   这个文件夹是在安装的时定义的 RX850 安装文件夹。

2) <rx_root>\src\rx850\conf\r22\std

   这个文件夹存储了产生内核函数库(22 位寄存器)的命令文件。

   利用在这个文件夹命令文件可以在<rx_root>\lib850\r22 中生成内核函数库(22 位寄存器)。

   Makefile :内核函数库"librx.a"生成文件

3) <rx_root>\src\rx850\conf\r26\std

   这个文件夹存储了产生内核函数库(26 位寄存器)的命令文件。

   利用在这个文件夹命令文件可以在<rx_root>\lib850\r26 中生成内核函数库(26 位寄存器)。

   Makefile :内核函数库"librx.a"生成文件

4) <rx_root>\src\rx850\conf\r32\std

   这个文件夹存储了产生内核函数库(32 位寄存器)的命令文件。

   利用在这个文件夹命令文件可以在<rx_root>\lib850\r26 中生成内核函数库(32 位寄存器)

   Makefile :内核函数库"librx.a"生成文件

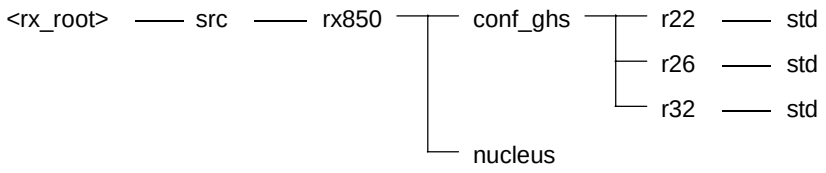
5) <rx_root>\src\rx850\nucleus

   这个文件夹储存了内核函数库"librx.a"的源文件。

2.3.4 源发布版本/GHS 编译版

图 2-4 示出了当存放在 RX850 发布媒质中的文件(源发布版本/GHS 编译版)安装后的文件夹配置。

图 2-4 文件夹配置(源发布版本/GHS 编译版)



下面是各个文件夹的详细说明:

1) <rx_root>

   这个文件夹是在安装时定义的 RX850 安装文件夹。

2) <rx_root>\src\rx850\conf_ghs\r22\std

   这个文件夹存储了产生内核函数库(22 位寄存器)的命令文件。

   利用在这个文件夹命令文件可以在<rx_root>\lib850_ghs\r22 中生成内核函数库(22 位寄存器)。

   nucleus.bld :为内核函数库"librx.a"生成文件

3) <rx_root>\src\rx850\conf_ghs\r26\std

   这个文件夹存储了产生内核函数库(26 位寄存器)的命令文件。

   利用在这个文件夹命令文件可以在<rx_root>\lib850_ghs\r26 中生成内核函数库(26 位寄存器)。

   nucleus.bld :为内核函数库"librx.a"生成文件

4) <rx_root>\src\rx850\conf_ghs\r32\std

   这个文件夹存储了产生内核函数库(32 位寄存器)的命令文件。

   利用在这个文件夹命令文件可以在<rx_root>\lib850_ghs\r32 中生成内核函数库(32 位寄存器)。

   nucleus.bld :为内核函数库"librx.a"生成文件

5) <rx_root>\src\rx850\nucleus

   这个文件夹储存了核心库"librx.a"的源文件。

第 3 章　系统构建

本章主要介绍如何构建一个系统。

3.1　概述

系统构建涉及到用从 RX850 发布媒介中拷贝到用户开发环境(主机)中的文件组将已经创建好的加载模块加载到目标系统中。

系统构建过程概述如下:

1) 创建一个系统配置文件
2) 创建信息文件
   - 系统信息表文件
   - 系统信息头文件

   [备注]:通过 CF850 创建这些信息文件。
3) 创建系统初始化模块
   - 引导过程
   - 初始化处理器
   - 中断入口
4) 创建空闲处理器
5) 创建处理程序
   - 任务
   - 中断处理器
   - 循环处理器

   [备注]　通过 C 语言或汇编语言创建这些处理程序。
6) 创建初始数据存储空间(仅当使用 CA850 版本时)
7) 生成链接指令文件
8) 生成加载模块
9) 嵌入系统

图 3-1 示出了采用 CA850 版本时，构建系统的过程。图 3-2 示出了采用 GHS 编译版本时，构建系统的过程。

图3-1系统构建(CA850版本)

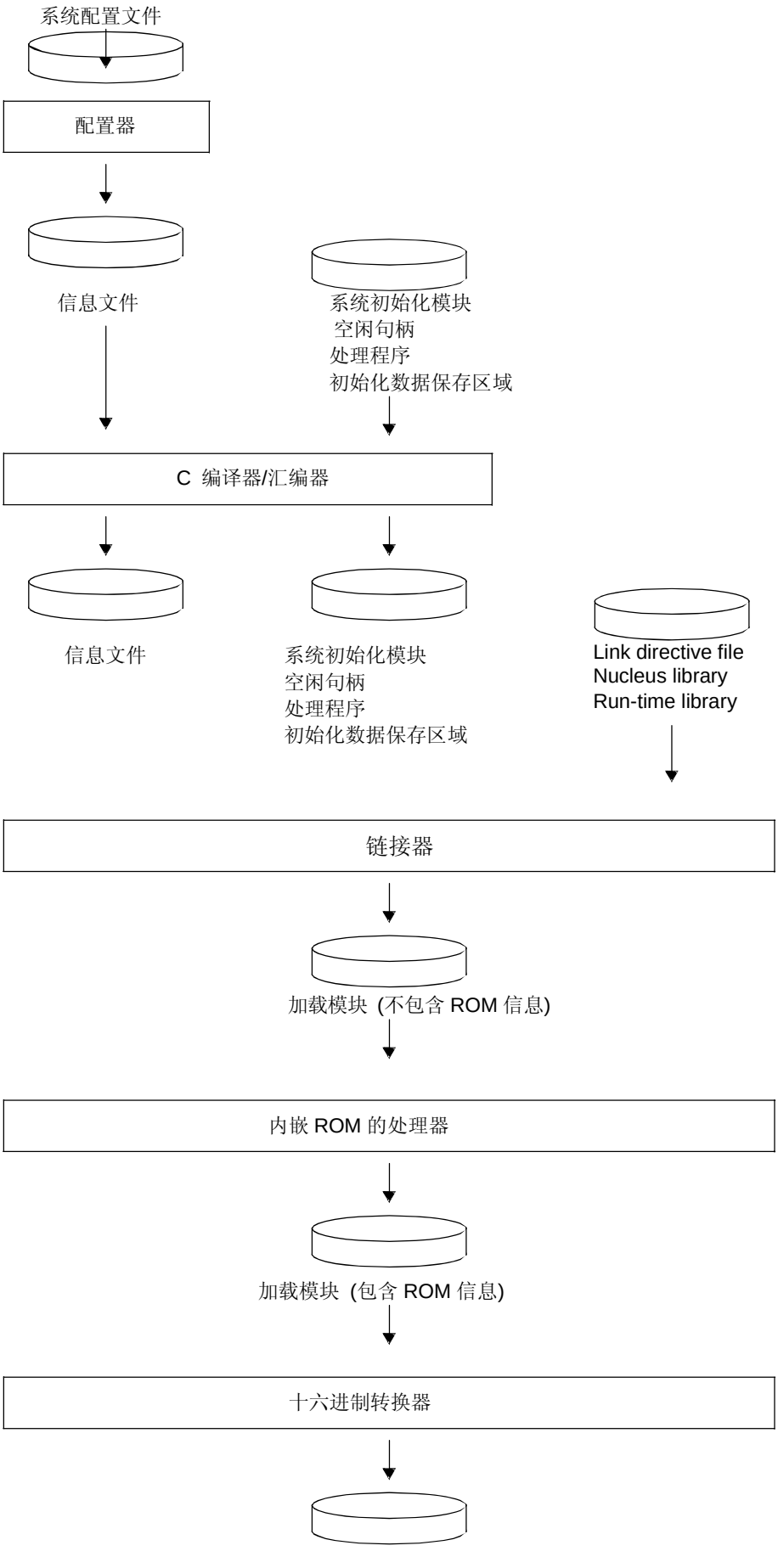
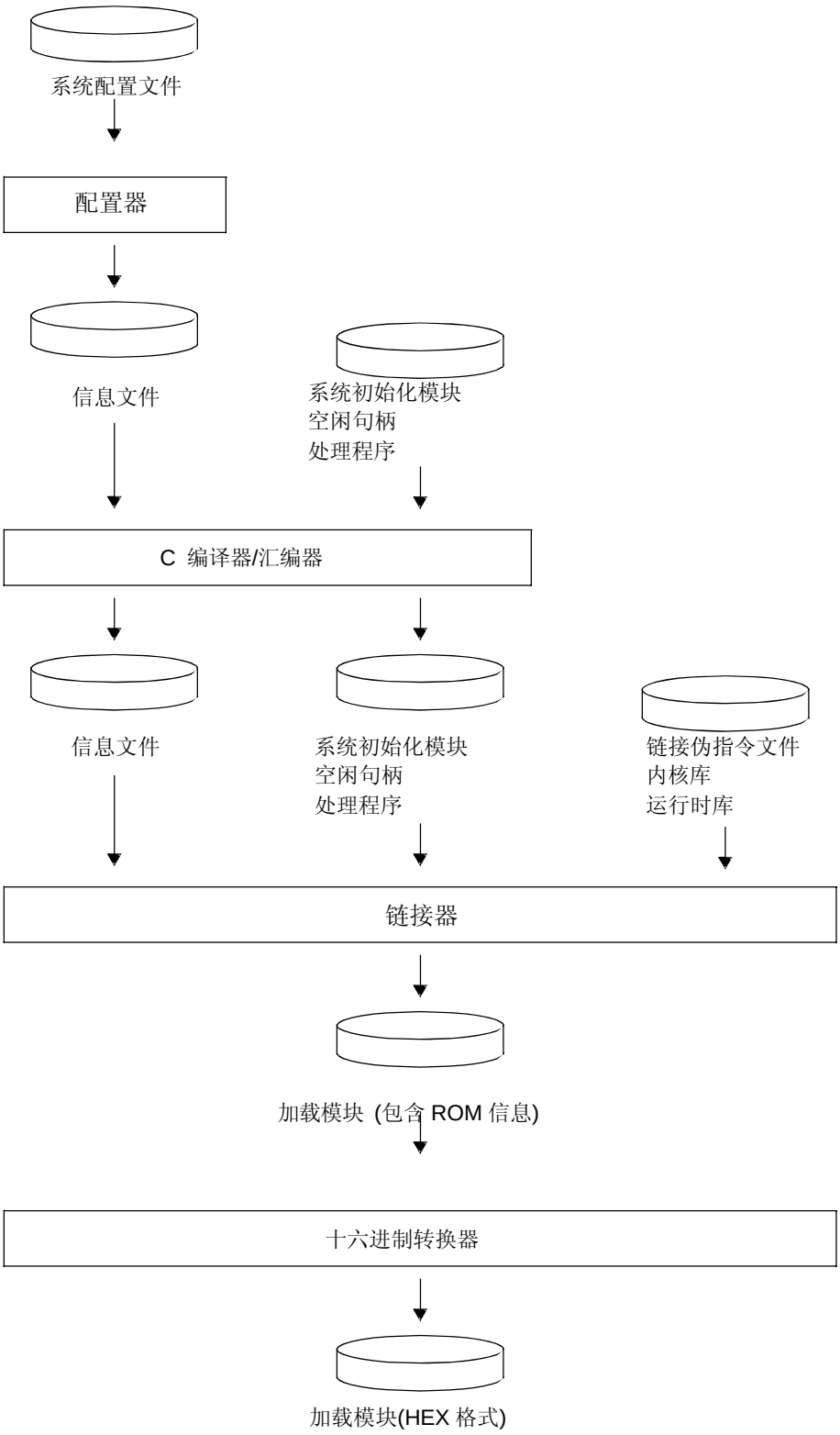


图 3-2 系统构建(GHS 编译器版本)



以上介绍的构建系统流程基于软件发行包中提供的示例程序。如果RX850安装在文件夹<rx_root>中，则示例程序位于如下位置:

表3-1 示例程序存储文件夹

编译器	存储文件夹
CA850版本	<rx_root>\smp850\rx850\src
GHS编译器版本	<rx_root>\smp850_ghs\rx850\src

根据使用的编译器分别参考上述文件夹之一。

3.2 创建一个系统配置文件

创建容纳 RX850 使用的各种数据的信息文件(系统配置文件)。

在使用 CF850 创建下列文件时必需该信息文件:

- 系统信息表文件
- 系统信息头文件

关于如何创建这些文件，参见"3.3 创建信息文件"。

“系统信息表文件”包含 RX850 的资源信息，如任务、semaphores、内存池。”系统信息头文件”利用#define 指令使作为资源 ID 定义的符号名，例如系统信息表文件创建的任务和信号量的符号名，与实际符号 ID 号对应起来。

示例系统信息文件是:

- sit.cf

关于系统信息文件的内容和语法，参见”第 5 章 系统配置文件”。

3.3 创建系统信息文件

使用 CF850，从”3.2 节创建系统配置文件”生成的系统配置文件中，创建 “系统信息表文件”和”系统信息头文件”。

建议采用如下文件名:

[系统信息表文件]

- CA850 版本:sit.s
- GHS 编译器版本:sit.850

[系统信息头文件]

- sys.h

为了用 RX850 构建一个应用，汇编 sit.s(sit.850)，并链接生成目标文件。C 源程序必须包含 sit.h。

关于如何利用CF850创建这些文件的细节，参见”第6章 操作配置器”。

3.4 创建系统初始化模块

系统初始化模块是一个函数，它由与用户目标系统相关若干程序段组成的。这个函数用来方便移植和定制。

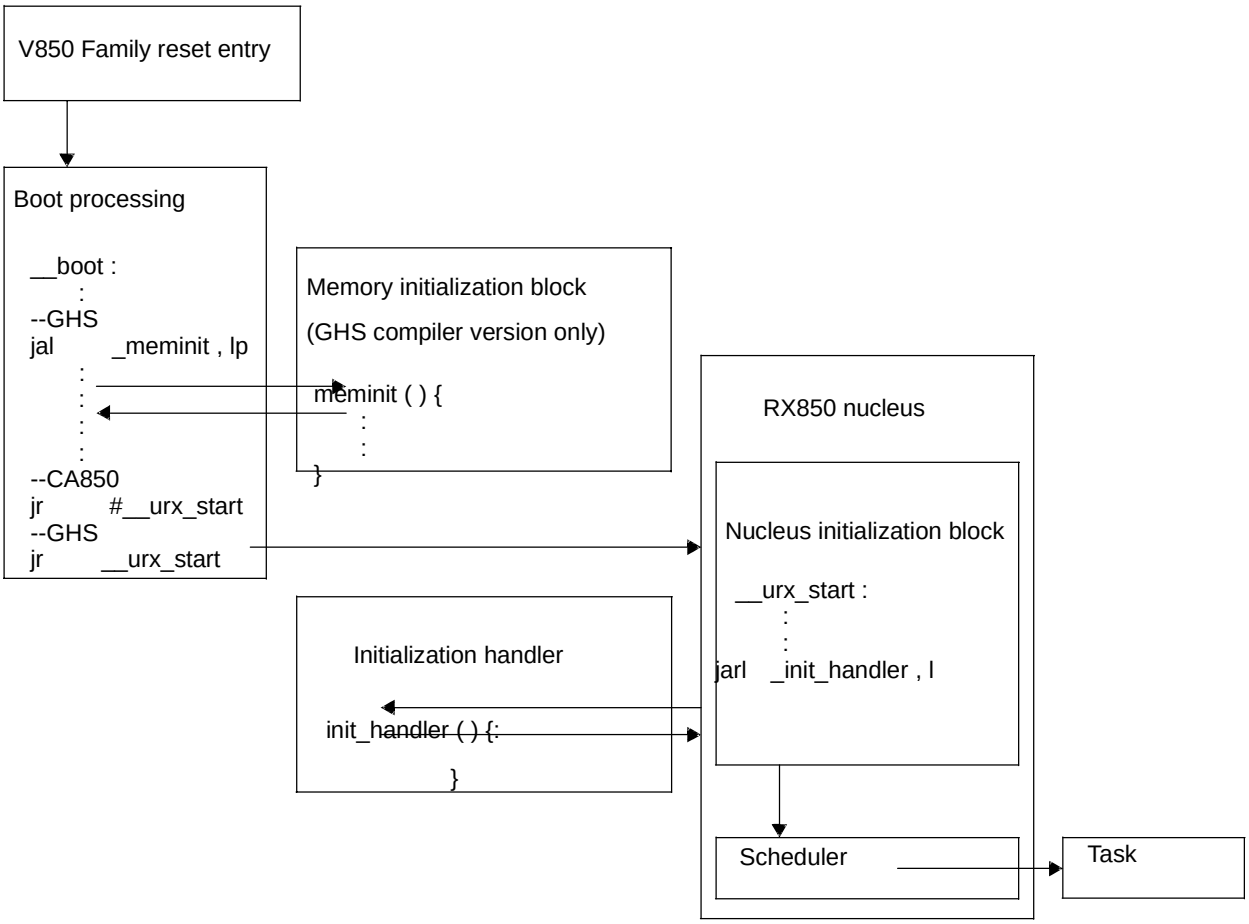
系统初始化模块的配置如下。

表 3-2 系统初始化模块配置

示例文件	类型	函数名	功能
boot.s (CA850 版本) boot.850 (GHS 编译器版本)	启动过程	Start	系统启动过程
inithdr.c	初始化处理器	init_handler	硬件初始化过程
entry.s (CA850 版本) entry.850 (GHS 编译器版本)	中断入口	None	导入中断过程的过程

系统初始化模块的粗略流程举例说明如下。

图 3-3 系统初始化模块流程图



3.4.1 引导过程

引导过程被指定到 V850 系列微处理器的启动入口(处理器地址:0X0)，是首先执行的系统初始化过程。

在示例文件 boot.c(boot.850)中符号 " _boot"之后定义的是引导过程的入口。使执行从复位入口跳转至这个标签的指令如下所示。GHS 版本 entry.850 文件中的这些指令与 CA850 版本 entry.s 文件中的指令是一样的。

[CA850 版本]

```
.section        "RESET"
.extern         __boot

mov             #__boot , lp
jmp             [ lp ]
```

[GHS 编译器版本]

```
.org            0x00000000
.extern         __boot

mov             __boot , lp
jmp             [ lp ]
```

指令的最低一行被指定到处理器地址[0X0]，因此，当执行复位操作时，将执行这些指令，程序执行跳转到_boot，然后执行引导过程。

作为引导过程的一部分，将执行如下过程:

1) Tp(文本指针)、gp(全局指针)和 ep"元素指针"的设定

2) 引导过程使用的 sp(堆栈指针)的设置

3) 利用 jr 指令跳转至_urx_start symbo，将控制转交给 RX850 内核初始化模块。

除了上述内容，在 GHS 编译器版本示例中使程序执行跳转至 meminit 函数(内存初始化模块)的过程(jarl-meimit,lp)在 2)和 3)之间执行。Meminit 函数初始化 bss 空间并拷贝默认值。

在 CA850 版本的示例中，RAM 上的 bss 空间在 boot.s 中被初始化(清零)。在 CA850 版本中，通过创建默认值存储空间(rompcrt.s)和利用函数_rcopy ( )来拷贝默认数值。关于其详细信息，参考 CA850 C 编译器包用户使用手册-操作。

虽然在 GHS 编译器版本示例不执行上面 2)，但 GHS 编译器版本示例中设置与上面 2)中类似的堆栈指针。这里设置的堆栈指针独立于任务和中断处理程序中的堆栈。当 RX850 启动后，任务和中断处理器使用的堆栈由 RX850 自身利用系统信息表文件维护和管理。利用任务切换或中断堆栈指针自动切换。因此，在引导过程中定义的堆栈指针在 RX850 启动之前使用。例如，当程序执行跳转至一个函数，而该函数有数据需存储在堆栈中时，将使用该堆栈指针。如果示例中 meminit 函数必需使用堆栈，也将使用该堆栈指针。

在启动过程的最后，过程3)是必需的。它执行如下过程:

[CA850 版本]

```
.extern         __urx_start
jr              __urx_start
```

[GHS 编译器版本]

```
.extern         __urx_start
jr              __urx_start
```

在符号 "__urx_start"之后定义的是 RX850 的内核初始化过程。当引导过程执行完成后，利用 jr 指令，控制将转交给内核初始化过程。根据”系统配置文件”创建的”系统信息表文件”，初始化过程创建资源并执行初始化。

日电电子建议修改示例中引导过程的定义来适合用户的具体环境。

3.4.2 内核初始化模块

核心初始化模块是引导过程完成后执行的一个 RX850 内部例行程序。根据系统配置文件建立的”系统信息表文件”，该模块创建 RX850 系统管理模块，并建立和初始化诸如任务、semaphores、内存池等的信息。

内核初始化模块中的初始化过程一经完成，便调用初始化处理器。RX850 初始化处理器的函数名确定为 ini_handler(采用汇编语言，则定义为标签”_init_handler”)。因此，建立一个以该函数名命名的函数是必需的。关于该函数的详细信息，参见”3.4.3 初始化处理器”

当控制从初始化处理器返回后，将启动调度管理器，然后启动 RX850。

3.4.3 初始化处理器

初始化处理器是从内核初始化模块调用的一个函数(处理器)。它定义在启动 RX850 前执行的过程，以及在这个处理器中的硬件过程。在初始化处理器中，它可能会发出可以被中断处理器或循环处理器处理的系统调用

初始化处理器是一个没有任何参数的 FP 类型函数。其函数名为 init_handler(在汇编语言中，为标签”_init_handler”)因此，建立一个以该函数名命名的函数是必需的。甚至该过程不需要的情况下，也要创建一个不执行任何处理的空函数。初始化处理器一旦完成，将通过返回指令将控制返回给内核初始化过程。

示例的初始化处理器初始化外设 I/O，设置中断控制寄存器和启动任务。在 CA850 版本中，默认数据值可以被拷贝到初始化处理器中。

关于如何拷贝默认数据值的详细信息，参考 CA850 C 编译器包用户手册－操作。

3.4.4 中断入口

中断入口是中断发生时执行的一条指令，它被指定到 V850 系列微处理器的”中断处理器地址”。用户使用的所有中断必须定义中断入口，并且必须使用汇编语言编写。CA850 版本示例的中断处理器在 entry.s 中定义，GHS 编译器版本示例的中断处理器在 entry.850 中定义。

RX850 中断由两种类型中断处理器处理:”直接激活中断处理器”和”间接激活中断处理器”。只有当使用直接激活中断处理器时需要定义中断入口。

在中断入口中，和普通中断入口定义一样的方式定义一个旁路指令。示例中，中断”INTP01(处理器地址:0x130)”是一个直接激活中断处理器的例子。在 CA850 版本中，使用.section 伪指令，在 GHS 编译器版本中，使用.org 指令。关于每个指令的详细情况，CA850 版本参考 CA850 用户手册-CA850 版本汇编语言。GHS 编译器版本参考与 GHS 编译器语言相关的手册。

直接激活中断处理器的入口如下:

[CA850 版本]

```
.section        "INTP01"
jr      entry_P01
```

[GHS 编译器版本]

```
.org 0x00000130
jr      entry_P01
```

如果跳转目的地标签”entry_P01”是在另外单独文件中定义的，则使用.extern 指令。

在 CA850 版本中，标签"entry_P01"在"intp01.c"中定义，在 GHS 编译器版本中，标签"entry_P01"在"intp01.850"中定义，在直接激活中断处理器的预处理和后处理指令被定义(在 macro 中)后，它使程序执行跳转到中断处理器程序(inthdrP01())。

关于如何描述一个直接响应中断处理器的详细情况，请参考 RX850 手册- 基础。

3.5 创建空闲处理器

空闲处理器是在没有任务被安排执行时启动的一个函数(处理器)。通过该函数，可以检测到系统的空闲状态，同时可以使用 CPU 的节电功能。例如:如果在空闲处理器中定义了将 CPU 设置为 HALT 模式的过程，则在系统空闲状态下，CPU 将被设置为 HALT 模式。空闲处理器是一个没有任何参数 FP 类型函数。其函数名确定为 idle_handler(在汇编语言中，为标签"_idle_handler")。因此，建立一个以该函数名命名的函数是必要的，甚至在空闲过程不需要的情况，也要创建一个不执行任何处理的空闲函数。然而，因为利用中断从空闲处理器退出，因此在空闲处理器中必须编写允许中断(EI 指令)的过程。

示例的空闲处理器如下文件所示:

- idlehdr.c

V850 系列支持"HALT 模式","IDLE 模式",和 "STOP 模式"三种节电功能。为了将 CPU 置于其中的某种模式，需要编写特定的过程。因此，对应于每种模式，本文都提供了一个示例。

每种模式、示例文件和函数名对应关系如下。

表 3-3 节电函数示例

模式	示例文件	函数名
HALT 模式	halt.s (CA850 版本) halt.850 (GHS 编译器版本)	__halt
IDLE 模式	idle.s (CA850 版本) idle.850 (GHS 编译器版本)	__idle
STOP 模式	stop.s (CA850 版本) stop.850 (GHS 编译器版本)	__stop

为了在 CA850 版本中编译和汇编这些文件，必须定义一种寄存器模式(在文件开始#if 和#elif 或 endif 之间的.option 伪指令为选择寄存器模式的指令)。默认寄存器模式是 32-寄存器模式。为了选择 22-或 27-寄存器模式，需明确明确指明如下汇编启动选项。

表 3-4 汇编启动选项

寄存器模式	启动选项(CA850 版本)
22-寄存器模式	-D__850_22__
26-寄存器模式	-D__850_26__

关于 V850 系列微处理器的节电功能详细情况，参考每个设备的用户手册－硬件。

3.6 创建处理程序

创建一个处理程序(应用程序)。

在一个应用中 RX850 必须的处理单元可以大致分为如下类别:

- 任务
- 直接激活中断处理器
- 间接激活中断处理器
- 循环处理器

示例的内容如下所示:

表 3-5 处理程序配置

示例文件名	类型	函数名	特点
task.c	任务	task1 task2	任务实体
handler.c	间接激活中断处理器	inthdrP00	处理器过程
	直接激活中断处理器	inthdrP01	处理器过程
	循环处理器	cychdr0 cychdr1	处理器过程

如果用 C 源语言编写的一个处理程序发起一个系统调用，则需要在处理程序中包含 RX850 提供的”stdrx850.h”头文件。该头文件包含了使用系统调用所必需的定义。示例的头文件”sample.h” (<rx_root>\smp850\rx850\include)包含 port.h，它定义了上述 stdrx850.h 和 V850 的外设 I/O 空间。

根据需要，在头文件中定义函数用到的常量，并且在程序中包含头文件。

3.7 创建初始数据存取空间

当采用 CA850 时，必须创建一个用于存储初始数据的空间。

这是因为需要在 ROM 中存储初始数据并且在程序执行前需要将这些默认数据值拷贝到 RAM 中。创建初始数据存储空间涉及在 ROM 中为初始数据预留存储空间。

关于如何创建该空间的详细信息，参见 CA850 C 编译器包–操作 用户手册中"可嵌入式 ROM 处理器”部分的说明。

当采用GHS编译器版本时，该过程在示例的meminit函数中执行。

3.8 创建链接指令文件

创建一个包含有链接器在进行模块链接时需要参考的"分段信息"和"地址信息"的链接指令文件(分段图文件)。

下列示例文件是链接指令文件

- sample.dir (CA850 版本)
- sample.lx (GHS 编译器版本)

下表列出的段是 RX850 必须的段

表 3-6 RX850 必须的段

段名称	空间类型
.sit	系统信息空间
.text	RX850 系统调用位置空间
.pool0	系统内存池 0
.pool1	系统内存池 1

.sit 段和.text 段是文本属性段。.pool0 和.pool1 段位于 RAM 空间。必须在链接指导文件(分段图文件)中定义这些信息。

必须创建.sit 和.pool0 段，因为 RX850 被设计为在地址 0 通过单条指令访问这两个段，这两个段必须位于地址 0 的+/- 32 Kbytes 的范围内(0xffff8000 到 0x7fff)。如果这两段没有在这个范围内，RX850 的管理模块将无法被正确访问。V850 系列微处理器架构建议将.pool0 设定在内部 RAM 中。关于.pool0 和.pool1 段的详细信息，参见"4.1 .pool0 and.pool1"。

和示例文件中编写到一样，在这些段的位置上定义信息。

示例中定义这些段的部分如下所示:

[CA850 版本]

```
TEXT :      !LOAD       ?RX {
            .sit        = $PROGBITS   ?AX       .sit ;
            .text       = $PROGBITS   ?AX       .text ;
} ;
EDATA :     !LOAD       ?RW V0x00100000 {
            .pool1      = $NOBITS     ?AW       .pool1 ;
} ;
IDATA :     !LOAD       ?RW {
            .pool0      = $NOBITS     ?AW       .pool0 ;
            .data       = $PROGBITS   ?AW       .data ;
            .sdata      = $PROGBITS   ?AWG      .sdata ;
            .sbss       = $NOBITS     ?AWG      .sbss ;
            .bss        = $NOBITS     ?AW       .bss ;
} ;
            :
            :
```

[GHS 编译器版本]

```
-sec {
            .sit        0x0000160     :
            .text                     :
            .pool1      0x0100000     :
            .pool0      0x0ffe000     :
            :
            :
```

此外，根据需要在文件中定义同 RAM 空间有关的段，如.data/.bss 段和同 ROM 空间有关的段，如.const 段。

日电电子建议根据用户使用环境修改链接指导文件。

关于如何编写链接指令文件的细节，参见 CA850 C 编译器包 – 操作 用户手册或者同 GHS 编译器语言有关的手册。

3.9 创建加载模块

下一步，创建加载模块(执行模块)。

根据"3.8 创建链接指令文件"创建的链接指令文件，链接由编译和汇编 C 源文件建立的目标文件(.o 文件)，并汇编源文件。

当用 RX850 链接应用程序时，需要参考如下函数库:

表 3-7 链接参考函数库

函数库	内容
librx.a	内核函数库

为 22-，26-和 36-寄存器模式都提供一个单独的函数库。不同寄存器模式应选用合适的函数库。

当链接成功后，将生成一个可执行模块(.out 文件)。在此阶段，可通过调试器读取并执行该可执行模块。由链接器正确生成的加载模块定位在 RAM 中的初始数据。如果在 VA850 版本中的应用中存在初始数据，必须创建一个能预留初始数据存储空间并含有拷贝例行程序的模块。这种情况下，必须通过可嵌入-ROM 处理器为链接器生成的加载模块创建一个加载模块。

关于如何使用可嵌入 ROM 处理器和关于拷贝例行程序的详细情况，参见参见 CA850 C 编译器包 – 操作用户手册中

"可嵌入式 ROM 处理器" 部分。

3.10 嵌入系统

在系统中嵌入完整的加载模块。

为了完成该工作，由"3.9 创建加载模块"生成的加载模块必须转换为 16 进制文件。

利用 CA850 和 GHS 编译器版本中提供的 16 进制转换器，生成必须格式的 16 进制文件，然后，通过 ROM 写入器将该文件嵌入系统。

第 4 章　内存和容量估算

这一章解释了如何管理 RX850 的内存(RAM)和被使用的内存容量。

4.1　.pool0 和.pool1

RX850 使用的内存空间包括如下两段:

-.pool0 段(系统内存池 0)

-. pool1 段(系统内存池 1)

这两段通过链接指令文件在 RAM 中定义。

分配到.pool0 段和.pool1 段的信息如下所示:

表 4-1　分配到.pool0 段和.pool1 段的信息

段名称	被分配信息
.pool0	系统基础表 备用队列 每一个管理模块 任务堆栈 中断处理器堆栈(系统堆栈) 可变长内存池 定长内存池
.pool1	任务堆栈 中断处理器堆栈(系统堆栈) 可变长内存池

因为 RX850 的系统信息分配在.pool0 段，所以必须创建该段。如果.pool0 段足够用，那么.pool1 段就不一定需要创建。

由系统的配置文件具体说明任务堆栈，中断处理器堆栈以及内存池在.pool0 段还是在.pool1 段中预留。关于如何定义的细节，参阅"第 5 章 系统配置文件"。

.pool0 段的位置是有限制的。因为 RX850 在地址 0 通过一条单指令访问.pool0 段，所该段必须位于从地址 0 开始的+/-32Kbytes(0xffff8000 到 0x7fff)的范围内。该段只有在这个范围内，RX850 管理模块才能被正确地访问。因为.pool0 部分必须位于 RAM 中，所以建议将该段分配在该地址范围内的内部 RAM 中。

.pool1 段不受此限制。

4.2 管理空间的内存容量

这一部分说明 RX850 系统基础表，备用队列和每一个管理模块的大小。这些都在.pool0 中预留。表 4-2 示出了每一个目标所使用的管理空间的大小以及如何估计这些空间的大小。

表 4-2 目标管理空间的大小

目标	所使用管理空间大小 (每一个目标)	计算大小
系统基础表	20 字节	固定大小(20 字节)同系统状态(任务的数目)无关。
备用队列	2 至 32 字节	系统配置文件中定义的优先级级数+ 1 (单位: 字节)
任务执行权	18 字节	在以下情况下可增加额外可用空间: 如果使用任务执行权组 [估计额外的大小] “有等待 task execution right 可能性”的 execution right groups 数 * 1 (单位: 字节) 如果可变长内存池和事件标记之一或都被使用 [估计额外的大小] 所有 task execution rights 的数目 * 8 (单位: 字节)
任务	9 字节	-
事件标记	5 字节	-
1-bit 事件标记	2 字节	-
信号量	2 字节	-
邮箱	8 字节	当具有使用 TA_MPRI 特征的邮箱时,每一个具有 TA_MPRI 特征的邮箱使用 1-字节空间
可变长内存池	16 字节	除了管理空间外，需要一个内存池空间
固定长内存池	4 字节	除了管理空间外，需要一个内存池空间
循环处理器	10 字节	-

4.3 堆栈的内存容量

4.3.1 任务堆栈空间

如果考虑任务是标准 C 函数(例如一个用于寄存器变量的存储空间)，那么为每一个任务分配的堆栈空间为如下 1-3 项使用的空间和已用空间之和。

1) 上下文空间

如果通过发起一个系统调用来进行任务分派，那么根据如下每一个寄存器模式占用堆栈的大小。

32-寄存模式 :136 字节

26-寄存模式 :112 字节

22-寄存模式 :96 字节

2) 可变长内存池工作空间

虽然假定在处理系统调用时 RX850 不使用堆栈，但当系统调用使用可变长内存池时，它将使用最多达 12 字节的任务堆栈。使用可变长内存池的系统调用如下所示。

get_blk,pget_blk,tget_blk,rel_blk,ter_tsk,rel_wai

3) 当中断产生时使用的临时寄存器存储空间

作为当中断产生时用来保存临时寄存器的空间，被占用的大小将根据下面每个寄存器模式而定。如果定义了通过 reti (RTOS_IntExit)指令返回的直接激活中断处理器，那么必须加上中断处理器引起的堆栈开销。

32-寄存模式 :68 字节

26-寄存模式 :56 字节

22-寄存模式 :48 字节

4.3.2 系统堆栈空间

当任一处理器(中断处理器，循环处理器，初始化处理器，空闲处理器)被激活，RX850 把堆栈空间从任务堆栈转换到处理器堆栈。此时，需要的堆栈大小计算如下:

1) 初始化处理器

考虑初始化处理器是一个标准 C 函数，由该函数带来的堆栈开销是必需的。因为初始化处理器执行过程中中断是被禁止的，还因为初始化处理器执行过程中处理不能转换到其他处理器或任务，因此如果由中断处理器占用的堆栈开销足够大，那么该堆栈开销大小可以不考虑。

2) 空闲处理器

考虑空闲处理器是一个标准 C 函数，由该函数带来的堆栈大小开销是必需的。因为假定空闲处理器执行过程中产生了中断，因此，必需为中断处理器或循环处理器在堆栈中单独为其分配占用的空间。

3) 中断处理器

当中断处理器被激活时，它占用一个包含 8 字节的处理器堆栈。如果中断处理器是嵌套的，则每嵌套一次，就占用另外一个 8 字节的堆栈。因此，除了 C 函数中断处理器占用的堆栈大小(在嵌套情况下的总大小)外，还需要占用的堆栈空间大小为”中断处理器嵌套最大次数*8”字节。

4) 计时器处理器

计时器处理器是一个间接激活中断处理器，当计时器处理器被激活时，将占用 20 字节的堆栈大小。然而，即使当计时器处理器正在执行过程中又被嵌套一次，也不用新增 20 字节的堆栈开销。

5) 循环处理器

循环处理器是一个由计时器处理器调用的 C 函数。因此，由这个处理器占用的堆栈大小已经在定时处理器中考虑了。

6) 可变长内存池

如果一个与可变长内存池相关的系统调用以与任务堆栈相同的方式被调用，将占用最多 12 字节的堆栈空间。如果系统调用由中断处理器调用，且这些系统调用也可能被嵌套的中断处理器再次调用。因此必需再加上最大为”嵌套中断处理器的最大次数*12”字节的堆栈空间消耗。

第 5 章　系统配置文件

本章说明一个系统配置文件并介绍如何编写该文件。

5.1　概述

为了用 RX850 构建一个系统，需要含有必要数据(系统信息和资源信息)的信息，保存这些信息的文件称为系统信息表文件。

系统信息表文件用汇编语言编写的，是一种特定格式数据枚举文件。虽然有用一个编辑器来编写系统信息表文件的可能，但是修改系统信息表文件或向系统信息表文件添加新数据非常困难，因此用编辑器编写该文件将花很多时间和精力。

因此，RX850 提供了一种叫"配置器 CF850"的应用程序。这个应用程序将包含原始格式系统信息和 RX850 资源信息的"系统配置文件"转换为系统信息表文件。用户可以通过创建一个系统配置文件和使用 CF850 来获得系统信息表文件。

CF850 从系统配置文件中输出两个文件:"系统信息表文件"和"系统信息头文件"。"系统信息头文件"利用#define 指令使作为资源 ID 定义的符号名，例如系统信息表文件创建的任务和信号量的符号名，与实际符号 ID 号对应起来。

关于如何启动 CF850，参考"第 6 章操作配置器"。

下面说明如何编写一个系统配置文件。

5.2　编写一个系统配置文件

本节主要介绍如何编写作为 CF850 输入的系统配置文件。

1) 字符编码

用 ASCII 码建立系统配置文件，系统要区分字母的大小写。用空格或 tab 来区分词语(如数值、符号名、关键词)，用换行符(LF)来区分语句。

[注意事项]日语编码，EUC 编码和 shift JIS 只能在注释中使用。

2) 数值

除非特别说明，可以用任意的 32 位数值(0x0 to 0xffffffff)表示数值。

3) 符号名

符号名可以用字母数字字符来表示(最多 31 个字符)，但是符号名必须要以下划线或字母开始。

4) 注释

在系统配置文件中，从"-"开始到该行结束之间的部分定义为注释。

5) 续行

在系统配置文件中，在一行的结尾处加上反斜杠表示下一行是该行的延续。注意反斜杠的前一个字符必须是空格或 tab 符。

6) 关键词

CF850 预留下面字符串作为关键词。这些关键词不能再用作其它目的。

Auto	az	Clkhdr	cyc	di	ei
Flg	flg1	Inthdr	intstk	maxpri	mbx
Mpf	mpl	Multi	no_use	no_wait	pool0
pool1	regmod	r22	r26	r32	RX850
Rxsers	sem	ser_def	sit_def	TA_MFIFO	TA_MPR
TCY_ULN	TCY_OFF	TCY_ON	trace	tsk	Tskgrp
TTS_DMT	TTS_RDY				

5.3 配置信息

在系统配置文件中编写的配置信息分为下面两个主要类型:

— 实时操作系统信息

与实时操作系统有关的数据。

— SIT(系统信息表)信息

RX850 操作必需的数据。

5.3.1 实时操作系统信息

在系统配置文件中说明的实时操作系统信息包含下面项目:

1) RX 系列信息

下列数据被当作是 RX 系列信息。

- 实时操作系统名称
- 版本号

5.3.2 SIT(系统信息表)信息

在系统配置文件中说明的 SIT 信息包含如下 12 项。

1) 系统信息

定义如下项作为系统信息:

- 寄存器模式
- 系统堆栈信息
- 时间中断源
- 跟踪信息

2) 系统最大值信息

定义如下项作为系统最大值信息:

- 任务优先级范围

3) 任务执行权组信息

为每个任务执行权组定义如下任务执行权组信息的项目:

- 任务执行权组名称
- 任务执行权组堆栈信息
- 任务执行权组等待状态信息

4) 任务信息

为每个任务定义如下任务信息项:

— 任务名

— 任务开始地址

— 任务堆栈信息

— 任务初始优先级

— 任务初始状态

— 任务激活程序

— 中断状态

5) 信号量信息

为每个信号量定义下列信号量信息项目:

— 信号量名

— 信号量的初始资源数

6) 事件标志信息

为每个事件标志定义如下事件标志信息项目:

— 事件标志名

7) 1-bit 事件标志信息

为 1-bit 事件标志定义下列项目:

- 1-bit 事件标志名

8) 邮箱信息

为每个邮箱定义的如下邮箱信息项目:

— 邮箱名

— 消息排队方法

9) 间接激活中断处理器信息

为每个间接激活中断处理器定义如下项目:

— 中断源

— 间接激活中断处理器的开始地址

10) 固定大小内存池信息

为每个固容大小内存池定义如下项目:

— 固定大小内存池信息名

— 内存块信息

— 内存块的总数目

11) 可变长内存池信息

为每个可变长内存池定义如下项目:

— 可变长内存池名

— 可变长内存池信息

12) 循环处理器信息

为每个循环处理器定义如下项目:

— 循环处理器名

— 循环处理器的开始地址

— 循环处理器的初始激活状态

— 循环处理器的激活间隔

5.4 实时操作系统信息的说明格式

下面示出了在系统配置文件中编写实时操作系统信息需要遵守的说明格式。

在下面的阐述中，黑体表示保留字，斜体表示用户提供的数值，符号名或关键词。

5.4.1 RX 系列信息

RX 系列信息定义了正在使用的实时操作系统的名称和版本，对一个系统配置文件，需要 RX 系列信息。

下面是 RX 系列信息的格式

rxsers	*rtos_nam*	*rtos_ver*

RX 系列信息的项目说明如下:

- *rtos_nam*

  明确说明实时操作系统的名字

  RX850 是唯一能为 *rtos_nam* 定义的关键词。

- *rtos_ver*

  定义了实时操作系统的版本

  能定义为 *rtos_ve* 关键词是 V31x(x 是任意数)。

5.5 SIT 信息的说明格式

下面示出了在系统配置文件中编写 SIT 信息需要遵守的说明格式。

在下面的说明中，黑体表示保留字，斜体表示用户提供的数值，符号名或关键词。

5.5.1 系统信息

系统信息定义了寄存器模式，系统堆栈信息，时钟中断源和跟踪信息，对于系统配置文件来说，系统堆栈信息的说明是必要的。

下面示出了系统信息的格式

```
regmod      reg_mode
intstk      intstk_siz : mem_nam
clkhdr      int_nam
trace       tracer
```

系统信息的项目说明如下:

- reg_mod

  定义当系统建立时被连接的"librx.a"内核函数库的寄存器模式。

  reg_mode 能使用关键词只有 r22、r26、r32:

  r22:　22 位寄存方式

  r26:　26 位寄存方式

  r32:　32 位寄存方式

  [注意事项 1] 当 regmod 没有定义时，默认为”regmod r32”模式。

  [注意事项 2] 当”-regxx”定义为 CF850 的激活选项时，该项中的定义将被忽略，CF850 激活选项变成有效信息。

- intstk_siz : mem_nam

  定义系统堆栈的大小(字节表示),和分配给系统堆栈的系统内存类型。intstk_siz 可以定义为 0x1 到 0xfffffffc 之间的某个值(与 4-字节边界对齐)。可以用关键词 pool0 或 pool1 定义 mem_nam,:

  pool0　: 分配系统栈到系统内存池 pool0(.pool0 段)

  pool1　: 分配系统栈到系统内存池 pool1(.pool1 段)

- *int_nam*

  定义作为系统时钟使用的时钟中断源，使用的 C 语言编译包不同，int_nam 能使用的数值和关键词也不同。

  [CA850 版本]

  中断源的名称在设备文件中定义，可以定义为从(异常编码-0x80)/0x10 计算出数值或 no_use 关键词。

  [GHS 版本]

  可以定义为从(异常编码-0x80)/0x10 中计算数值或 no_use 关键词。

  [注意事项 1] 当 clkhdr 没有定义时，默认为 clkhdr no_use。

  [注意事项 2] 定义 no_use 导致 RX850 提供的定时器操作功能(循环唤醒，延迟唤醒，超时等)被禁用。

- *tracer*

指定使用跟踪信息的工具类型。

如果省略能被定义成 *tracer* 关键词及 trace 的定义，将执行的操作将根据使用的交叉工具不同而不同。

[CA850 版]

只有关键词 az 能被定义。

Az : 使用 AZ850 提供的跟踪功能。

[注意事项] 如果省略了 trace 的定义，那么默认为是"trace az”。

[GHS 编译版]

可以定义的关键词是 az 或 multi

az : 使用 AZ850 提供的跟踪功能。

Multi : 使用 MULTI 提供的任务调试功能。

[注意事项] 如果省略了 trace 的定义，那么默认为是"trace multi”

5.5.2 系统最大值信息

系统最大值信息定义任务优先级范围值。

下面示出了系统最大值信息的格式。

```
maxpri      pri_lvl
```

系统最大值信息的项目说明如下:

-pri_lvl

指定任务优先级范围。

pri_lvl 可以指定为在 0x1～0x1f 之间的一个值或者关键词”auto”。

[注意事项 1]当省略 maxpri 定义时，默认为"maxpri auto".

[注意事项 2]当定义为 auto ，CF850 参考在任务信息中指定的初始优先级，并输出相关的优先级范围。

5.5.3 任务执行权组信息

为每个任务执行权组定义诸如"任务执行权组名"、"任务执行权组使用的堆栈信息"、"任务执行权组使用的等待状态信息"的项目作为任务执行权组信息。

但是，任务执行权组信息的项目数目限制在 0～127 之间。

下面是任务执行权组信息的格式:

tskgrp *tskgrp_id* *stk_siz* : *mem_nam* *wait*

任务信息权组信息的项目说明如下:

- *tskgrp_id*

  指定任务执行权组信息的名称

  只能为 *tskgrp_id* 指定符号名。

- *stk_siz* : *mem_nam*

  指定任务执行权组的系统堆栈的大小(字节表示),和分配给执行权组系统堆栈的系统内存类型。*stk_siz* 可以定义为在 0x1 和 0xfffffffc 之间的某个值(与 4-字节边界对齐)。可以用关键词 pool0 或 pool1 指定 *mem_nam*,:

  pool0:分配执行权组系统堆栈到系统内存池 pool0(.pool0 段)

  pool1:分配执行权组系统堆栈到系统内存池 pool1(.pool1 段)

- *wait*

  指定任务执行权等待状态是否允许发生

  可以为 *wait* 指定的关键词是 no_wait.

  no_wait :当任务开始时，任务执行权组等待状态不会发生。

  [注意事项] 当没有为*wait*定义关键词时，在任务开始时任务执行权组等待状态可能会发生。

5.5.4 任务信息

为每个任务定义诸如"任务名"、"任务开始地址"、"任务使用的堆栈信息"，"任务的初始优先级"，"任务初始状态"，"激活码"和中断状态"等项目作为任务信息。

对于一个系统配置文件，需要至少定义一个任务信息项目。

但是，任务信息的项目数目限制在 0～127 之间。

下面是任务信息的格式:

tsk	*tsk_id* *sts*	*sta_adr* *sta_code*	*tskgrp_id* | *stk_siz* : *mem_nam* *intr*	*pri*	\

任务信息信息的项目说明如下:

- *tsk_id*

  指定一个任务名

  只能为 tsk_id 指定一个符号名。

  [注意事项] CF850 用下列格式将 tsk_id 和任务 ID 号之间的关系输出到系统信息头文件:

  #define tsk_id 任务-ID-号

- *sta_adr*

  指定特定任务的开始地址。

  可以为 sta_adr 定义 0x0 和 0x3fffffe 之间的一个值(2-字节边界对准)。或者，可以定义一个符号名。

  [注意事项] 当为 sta_adr 定义一个符号名时，指定一个在汇编语言编程中使用的标签名。

- *tskgrp_id*, *stk_siz* : *mem_nam*

  定义任务使用的堆栈的大小(字节表示),和分配给该堆栈的系统内存类型。

  只能为 *tskgrp_id* 定义一个在"任务执行权信息"中定义的任务执行权组名。

  stk_siz 可以定义为在 0x1 和 0xfffffffc 之间的某个值(与 4-字节边界对齐)。可以为 *mem_nam* 指定的关键词是 pool0 或 pool1。

  pool0:分配执行堆栈到系统内存池 pool0(.pool0 段)

  pool1:分配执行堆栈到系统内存池 pool1(.pool1 段)

- *pri*

  定义任务的初始优先级。

  可以为 *Pri* 指定 0x1~ 0x1f 之间的一个数值。

- *sts*

  指定任务的初始状态

  可以为 *sts* 定义的关键词是 TTS_DMT 和 TTS_RDY

  TTS_DMT: 在系统被激活时，进入休眠状态。

  TTS_RDY: 在系统被激活时，进入准备状态。

- *sta_code*

  定义任务激活码

  可以为 *sta_code* 定义 0x1 和 0xfffffffc 之间的某一值或一个符号名。

  [注意事项] 只有为 *sts* 指定了 TTS_RDY，*sta_code* 才有效，当为 *sts*.指定了 TTS_DMT 时，*sta_code* 无效。

- *intr*

  定义中断状态

  可以为 *intr* 指定的关键词是 ei 和 di.

  ei : 任务激活时，允许所有的中断。

  di : 任务激活时，禁止所有的中断。

  [注意事项]省略 int 的定义导致在任务激活时，允许所有中断。

5.5.5 信号量信息

为每个信号量定义诸如"信号量名"、"信号量使用的初始资源计数"等项目作为信号量信息。

但是，信号量信息的项目数目限制在 0～127 之间。

下面是信号量信息的格式:

```
sem          sem_id   init_cnt
```

信号量信息的项目说明如下:

- *sem_id*

  定义信号量名

  只能为 *sem_id* 定义符号名。

  [注意事项]　CF850 用下列格式将 *sem_id* 和信号量 ID 号之间的关系输出到系统信息头文件:

  #define　*sem_id*　semaphore-ID-号

- *init_cnt*

  为 semaphore 指定初始资源计数。

  可以为 *int_cnt* 指定在 0x0 和 0x7f 之间的数值。

5.5.6 事件标志信息

为每个事件标志定义”事件标志名”等项目作为事件标志信息。

但是，事件标志信息的项目数目限制在 0～127 之间。

下面是事件标志信息的格式:

```
flg   flg_id
```

事件标志信息的项目说明如下:

-flg_id

定义事件标志名

只能为 *sem_id* 定义符号名。

[注意事项] CF850 用下列格式将 *flg_id* 和事件标志 ID 号之间的关系输出到系统信息头文件:

#define　　*sem_id*　事件-标志-ID-号

5.5.7 1-bit 事件标志信息

为每个 1-bit 事件标志定义”1-bit 事件标志名”等项目作为 1-bit 事件标志信息。

但是，1-bit 事件标志信息的项目数目限制在 0～127 之间。

下面是 1-bit 事件标志信息的格式:

flg1 *flg1_id*

1-bit 事件标志信息的项目说明如下:

- *flg1_id*

  flg1_id 定义 1-bit 事件标志名

  只能为 *flg1_id* 定义符号名。

  [注意事项] CF850 用下列格式将 *fl1g_id* 和 ID 号之间的关系输出到系统信息头文件:

  #define *flg1_id* *1-bit*-事件-标志-ID-号

5.5.8 邮箱信息

为每个邮箱定义邮箱信息项目，如”邮箱名”和”邮箱排队方法”。

但是，邮箱信息的项目数目限制在 0～127 之间。

下面是邮箱信息的格式:

mbx *mbx_id* *mwai_opt*

邮箱标志信息的项目说明如下:

- *mbx_id*

  定义一个邮箱名

  只能为 *mbx_id* 定义符号名。

  [注意事项] CF850 用下列格式将 *mbx_id* 和 ID 号之间的关系输出到系统信息头文件:

  #define mbx_id 邮箱-标志-ID-号

- *mwai_opt*

  定义消息排队方法

  可以为 *mwai_opt* 定义的关键词是 TA_MFIFO 和 TA_MPRI。

  TA_MFIFO:消息以与其发送相同的顺序排队。

  TA_MPRI.: 消息根据它们的优先级排队。

5.5.9 间接激活中断处理器信息

为每个间接激活中断处理器定义信息项目，如"中断源"，"间接激活中断处理器的开始地址"。

但是，对每个中断源只能定义一个间接激活中断处理器信息的项目。

下面是间接激活中断处理器信息的格式:

```
inthdr          int_nam  hdr_adr
```

间接激活中断处理器信息的项目说明如下:

- *int_nam*

  指定一个中断源.

  可以为 int_nam 定义的值或关键词根据采用 C 编译器包的不同而不同。

  [CA850 版本]

  可以定义为在设备文件中定义的中断源名或从(异常编码-0x80)/0x10 计算出的数值。

  [GHS 编译器版本]

  可以定义为从(异常编码-0x80)/0x10 中计算出的数值。

- *hdr_adr*

  定义间接激活中断处理器的开始地址。

  可以为 hdr_adr 定义 0x0 和 0x3fffffe 之间的一个值(2-字节边界对准)。或者，可以定义一个符号名。

  [注意事项] 当为hdr_adr定义一个符号名时，定义一个在用汇编语言编程中使用的标签名。

5.5.10 固定大小内存池信息

为每个固定大小内存池定义信息项目，如"内存池名"，"内存块信息"，"内存块信息的总数"。

但是，固定大小内存池信息的项目数目限制在 0～127 之间。

下面是固定大小内存池信息的格式:

mpf *mpf_id* *blk_siz* : *mem_nam* *blk_cnt*

固定大小内存池信息的项目说明如下:

- *mpf_id*

  定义一个固定大小内存池的名字

  只能为 mpf_id 定义一个符号名。

  [注意事项] CF850 用下列格式将 mpf_id 和 ID 号之间的关系输出到系统信息头文件:

  #define *mpf_id* *固定*-大小-内存-池-ID-号

- *blk_siz* : *mem_nam*

  定义内存块的大小(用字节表示基本块大小)和分配给固定大小内存池的系统内存类型。

  blk_siz 可以定义为在 0x4 和 0xfffffffc 之间的某个值(与 4-字节边界对齐)。可以为 mem_nam 指定的关键词是 pool0 或 pool1:

  pool0 : 分配固定大小内存池到系统内存池 pool0(.pool0 段)

  pool1 : 分配固定大小内存池到系统内存池 pool1(.pool1 段)

- *blk_cnt*

  定义内存块段总数

  *blk_cnt 可以定义为*0x1 到0xffffffff之间到某个值。

5.5.11 可变长内存池信息

为每个可变长内存池定义信息项目，如"内存池名"，"内存块信息"。

但是，可变长内存池信息的项目数目限制在 0～127 之间。

下面是可变长内存池信息的格式:

mpl *mpl_id* *mpl_siz* : *mem_nam*

可变长内存池信息的项目说明如下:

- *mpl_id*

  指定一个可变长内存池的名字

  只能为 mpl_id 指定一个符号名。

  [注意事项] CF850 用下列格式将 mpl_id 和 ID 号之间的关系输出到系统信息头文件:

  #define *mpl_id* *可变*-大小-内存-池-ID-号

- *mpl_siz* : *mem_nam*

  定义可变内存池的大小(单位:字节)和分配给可变长内存池的系统内存类型

  blk_siz 可以定义为在 0x4 和 0xfffffffc 之间的某个值(与 4-字节边界对齐)。可以为 mem_nam 指定的关键词是 pool0 或 pool1:

  pool0:分配可变长内存池到系统内存池 pool0(.pool0 段)

  pool1:分配可变长内存池到系统内存池 pool1(.pool1 段)

5.5.12 循环处理器信息

为每个循环处理器定义信息项目，如”循环处理器名”，”循环处理器开始地址”，”循环处理器初始激活状态”和”循环处理器激活间隔”。

但是，循环处理器信息的项目数目限制在 0～127 之间。

下面是循环处理器信息的格式:

cyc *cyc_no* *hdr_adr* *act* *intvl*

循环处理器信息的项目说明如下:

- *cyc_no*

  定义一个循环处理器的名字

  只能为 *cyc_no* 指定一个符号名。

  [注意事项] CF850 用下列格式将 cyc_no 和循环处理器 ID 号之间的关系输出到系统信息头文件:

  #define *cyc_no* *循环*-处理器-说明 -号

- *hdr_adr*

  定义特定的循环处理器到开始地址。

  hdr_adr 可以定义为在 0x0 和 0x3fffffe 之间的某个值(与 2-字节边界对齐)。或者指定一个符号名。

  [注意事项]当为 hdr_adr 定义一个符号名时，定义一个在汇编语言编程中使用的标签名。

- *act*

  定义循环处理器的初始激活状态，只能为 *act* 定义关键词 TCY_ON、TCY_OFF 和 TCY_ULNK

  TCY_ON: 当系统激活时，活动状态设置为 ON.

  TCY_OFF: 当系统激活时，活动状态设置为 OFF.

  TCY_ULNK:循环处理器从计时器队列中离队。

- *intvl*

  为循环处理器指定激活间隔(单元:基本时钟周期)

  intvl 可以定义为 0x1 到 0xffffffff 之间到一个值。

5.6 注意事项

在一个系统配置文件中，按照下列顺序说明配置信息(实时操作系统信息和系统信息表信息):

1)实时操作系统信息说明开始的声明

2)实时操作系统信息说明

3)系统信息表信息说明开始的声明

4)系统信息表信息说明

图 5-1 示出了如何编写一个系统配置文件

图 5-1 编写一个系统文件

```
------------------------------------------------------------
--Declaration of start of "Real-time OS information" description
------------------------------------------------------------

ser_def


-----------------------------
--"Real-time OS information" description
-----------------------------

.....
.....
.....


------------------------------------------------------------
-- Declaration of start of "SIT (system information table) information" description
------------------------------------------------------------

sit_def


-----------------------------
--"SIT (system information table) information" description
-----------------------------

.....
.....
.....
```

5.7 举例说明

图 5-2 和图 5-3 示出了当使用 V851 时的系统配置文件说明的例子。
图 5-2 是当使用 CA850 版时的系统配置文件的例子，图 5-3 使用 GHS 编译版时的系统配置文件的例子。
在图 5-2 和图 5-3 中的例子中，编写了如下信息:

```
- RX 系列信息
  实时操作系统名    :RX850
  版本号            :V320
- 系统信息
  寄存器模式        :32-寄存器模式
  系统堆栈信息      :分配系统内存池 1 的 0x100 bytes
  时钟中断源        :异常码
  CA850 版          : INTCM4
                      GHS 编译版: 0x5

  跟踪信息          :工具类型
                      CA850 版: az
                      GHS 编译版: multi
- 系统最大值信息
  任务优先级范围    : 0x1f
- 任务执行权组信息
  任务执行权组名    : txcb0
  堆栈信息          :分配系统内存池 1 的 0x100 bytes
  等待信息          :任务执行权组等待状态可能发生
  任务执行权组名    :txcb1
  堆栈信息          :分配系统内存池 1 的 0x200 bytes
  等待信息          :任务执行权组等待状态不会发生
-- 任务信息
  任务名            :task1
  初始地址          :abel-name _task1
  栈信息            :分配系统内存池 1 的 0x100 bytes
  任务初始优先级    :0x1
  初始状态          :就绪
  激活码            :0xa
  中断状态          :禁止所有中断

  任务名             :task2
  起始地址           :abel-name _task2
  堆栈信息           :使用任务执行权组 txcb0
  任务初始优先级     0x2
  初始状态           :就绪
  激活码             :0x14
  中断状态           :允许所有中断
  任务名             :task3
  起始地址           :abel-name _task3
  堆栈栈信息         :使用任务执行权组 txcb1
  任务初始优先级     :0x3
  初始状态           :休眠
  激活码             :0x1e
  中断状态           :允许所有中断
-- 信号量信息
  信号量名           :sem1
  初始资源计数       0x0

  信号量名           :sem2
  初始资源计数       :0x0
```

信号量名 :sem3
初始资源计数 :0x0
信号量名 :sem4
初始资源计数 :0x7f

-- 事件标志信息
事件标志名 :evf32_1

-- 1-bit 事件标志信息
1-bit 事件标志名 :evf1_1

-- 邮箱区信息
邮箱区名 :mbx1
排队方法 :FIFO
邮箱区名 :mbx2
排队方法 :FIFO
邮箱区名 mbx3
排队方法 :FIFO
邮箱区名 :mbx4
排队方法 :FIFO
邮箱区名 :pmbx1
排队方法 :根据优先级
邮箱区名 :pmbx2
排队方法 :根据优先级
邮箱区名 :pmbx3
排队方法 :根据优先级
邮箱区名 :pmbx4
排队方法 :根据优先级

-- 间接激活中断处理信息
中断源 :异常码
CA850 版--- INTP00
GHS 编译版-- 0xa
初始地址 :Label name _inthdr

-- 固定大小内存池信息
内存池名 :mpf1
内存块信息 :系统内存池 1 的(0x8bytes)基本内存块被分配
内存块总数 :0xa

-- 可变长内存池信息
内存池名 :mpl1
内存池大小信息 :系统内存池 1 的 0x20 bytes 被分配

-- 循环处理器信息
循环处理器名 :cychdr1
初始地址 :label-name _cychdr1
初始活动状态 :Off
循环激活间隔 :0x1 (单位: 时钟中断循环)
循环处理器名 :cychdr2
初始地址 :label-name _cychdr2
初始活动状态 :On
循环激活间隔 :0x2 (单位: 时钟中断循环)

循环处理器名 :cychdr3

初始地址 :label-name _cychdr3

初始活动状态 :循环处理器是从定时序列中离队出来的

循环激活间隔 :0x3(单位: 时钟中断循环)

图 5-2 系统配置文件举例(CA850 版)

```
--“实时操作系统信息”说明开始声明
------------------------------
 ser_def

----“实时操作系统信息说明”
------------------------------
-- RX 系列信息
rxsers          RX850   V320
------------------------------------------------------------
--“系统信息表信息”说明开始声明
------------------------------------------------------------
sit_def

------------------------------
-- "系统信息表信息"说明
------------------------------
--系统信息
Regmod      r32
Intstk          0x100:pool1
Clkhdr          INTCM4
Trace           az
--系统最大值信息
Maxpri          0x1f
--任务执行权组信息
Tskgrp      txcb0       0x100:pool0
Tskgrp      txcb1       0x200:pool1     no_wait
--任务信息
Tsk         task1       _task1          0x100:pool1     0x1         TTS_RDY         \
                0xa         di
Tsk         task2       _task2          txcb0           0x2         TTS_RDY         \
                0x14            ei
Tsk         task3       _task3          txcb1           0x3         TTS_RDY         \
                0x1             ei
-- -Semaphore 信息
Sem         sem1        0x0
Sem         sem2        0x0
Sem         sem3        0x1
Sem         sem4        0x7f
--事件标志信息
Flg             exv32_1
```

```
--1-bit 事件标志信息
Flg1        exv1_1
--邮箱区信息
Mbx         mbx1        TA_MFIFO
Mbx         mbx2        TA_MFIFO
Mbx         mbx3        TA_MFIFO
Mbx         mbx4        TA_MFIFO
Mbx         pmbx1       TA_MPRI
Mbx         pmbx2       TA_MPRI
Mbx         pmbx3       TA_MPRI
Mbx         pmbx4       TA_MPRI
--间接激活中断处理信息
Inthdr          INTP00          _inthdr
--固定大小内存池信息
Mpf             mpf1            0x8:pool1    0xa
--可变大小内存池信息
Mpl             mpl1             0x20:pool1
--循环处理器信息
Cyc             cychdr1         _cychdr1     TCY_OFF      0x1
Cyc             cychdr2         _cychdr2     TCY_ON       0x2
Cyc             cychdr3         _cychdr3     TCY_ULNK     0x3
```

图 5-3 系统配置文件举例(GHS 编译版)

```
--“实时操作系统信息“说明开始声明
------------------------------
ser_def
------------------------------
----“实时操作系统信息“说明
------------------------------
-- RX 系列信息
rxsers          RX850   V320
------------------------------------------------------------
--“系统信息表信息“说明开始声明
------------------------------------------------------------
 sit_def
------------------------------
-- "系统信息表信息“说明
------------------------------
--系统信息
Regmod     r32
Intstk          0x100:pool1
Clkhdr          0x5
Trace           multi
--系统最大值信息
Maxpri          0x1f
--任务执行权组信息
Tskgrp      txcb0       0x100:pool0
Tskgrp      txcb1       0x200:pool1     no_wait
--任务信息
Tsk         task1       _task1          0x100:pool1     0x1         TTS_RDY         \
                0xa         di
Tsk         task2       _task2          txcb0           0x2         TTS_RDY         \
                0x14        ei
Tsk         task3       _task3          txcb1           0x3         TTS_RDY         \
                0x1         ei
-- -Semaphore 信息
Sem         sem1        0x0
Sem         sem2        0x0
Sem         sem3        0x1
Sem         sem4        0x7f
--事件标志信息
Flg         exv32_1
```

```
--1-bit 事件标志信息
Flg1        exv1_1
--邮箱区信息
Mbx         mbx1        TA_MFIFO
Mbx         mbx2        TA_MFIFO
Mbx         mbx3        TA_MFIFO
Mbx         mbx4        TA_MFIFO
Mbx         pmbx1       TA_MPRI
Mbx         pmbx2       TA_MPRI
Mbx         pmbx3       TA_MPRI
Mbx         pmbx4       TA_MPRI


--间接激活中断处理信息
Inthdr          0xa             _inthdr
--固定大小内存池信息
Mpf             mpf1            0x8:pool1    0xa
--可变大小内存池信息
Mpl             mpl1             0x20:pool1
--循环处理器信息
Cyc             cychdr1         _cychdr1     TCY_OFF      0x1
Cyc             cychdr2         _cychdr2     TCY_ON       0x2
Cyc             cychdr3         _cychdr3     TCY_ULNK     0x3
```

第 6 章　操作配置器

这一章解释了如何使用配置器 CF850 从系统配置文件创建一个信息文件(系统信息表文件或者系统信息头文件)

6.1　概述

为构建使用 RX850 提供的函数的系统(加载模块)，需要储存 RX850 数据的信息。

因为信息文件基本上是数据的枚举，这样就可能用各种各样的编辑器来编写它们。然而，信息文件在可读性和可编写方面较差，因此编写它们需要花费相当的时间和努力。

为了解决这个问题，RX850 提供了一个实用工具(配置器 CF850)，它可以把易于可读和编写的系统配置文件转换成信息文件。

CF850 将系统配置文件作为输入文件，输出信息文件。

从 CF850 输出的信息文件的说明如下。

- 系统信息表文件

  一个存储了 RX850 操作所需数据(RX850 的资源信息，例如任务，旗语和事件标记)的信息文件。

- 系统信息头文件

  一个存储了在系统配置文件中编写的 ID 号和目标名(例如，任务，旗语和事件标记名)之间对应信息的信息文件，

表 6-1 示出了 CF850 的操作环境。

表 6-1 用于 CF850 的操作环境

主机	操作系统
基于 Windows - PC/AT-兼容机	Windows 98, Me, NT4.0, 2000, XP

6.2 激活方法

6.2.1 从命令行激活

下面的内容是介绍如何从命令行激活 CF850。

注，在下面的例子中"C>"表示命令提示符，" "表示空格键。

包括在"[ ]"里面的激活选项可以被省略。

[CA850 版本]

C> cf850 [@*cmd_file*] [-cpu *name*] [-devpath=*path*] [-reg*xx*] [-i *sit_file*] [-ni] [-d *h_file*] [-nd] [-V] [-help] *cf_file*

[GHS 编译版本]

C> cf850 [@*cmd_file*] [-reg*xx*] [-i *sit_file*] [-ni] [-d *h_file*] [-nd] [-V] [-help] *cf_file*

每一个激活选项详细解释如下:

- *@cmd_file*

定义任务文件名。

如果省略:在命令行定义的激活选项有效.。

备注:关于命令文件的细节，参阅 "6.2.3 命令文件".

- cpu *name*

定义目标设备类型说明名。

如果省略:CF850 不读设备文件。因此，在系统配置文件中使用了"在设备文件中定义的中断源名"的定义将不能执行。

备注:只能为 CA850 版本定义该激活选项。

- *devpath=path*

从 path 文件中获取得到由-cpu name 指定的目标设备对应的设备文件。

如果省略:设备文件按当前文件夹，..\..\..\dev.的顺序获取。

备注 :只能为 CA850 版本定义该激活选项

- reg*xx*

定义输出文件格式(寄存器模式)

能为 xx 指定的关键词是 22，26 或 32。

22 : 22-寄存器模式

26 : 26-寄存器模式

32 : 32-寄存器模式

如果省略该项:由系统信息指定的寄存器模式有效。

如果该激活选项和系统信息中寄存器模式都没有被定义，CF850 默认寄存器模式为"-reg32"。

- i *sit_file*

定义将要输出的系统信息表文件名

如果省略该项:CF850 默认定义了以下的激活选项，并执行过程。

对于 CA850 版本: -i sit.s

对于 GHS 编译器版本: -i sit.850

备注 1:被定义的输出文件名 sit_file 包括路径名在内最多为 255 个字母。

备注 2: 当这个激活选项和-ni 选项被同时定义时，只有最后一个输入的才有效。

- ni

禁止系统信息表文件的输出。

如果省略:系统在 cf_file 定义的系统配置文件名后面添加了一个”i，修改扩展名为.tbl，然后将该文件作为系统信息表文件输出。

备注:当这个系统激活选项和-i sit 文件选项在同一个时间被定义时，只有最后一个输入的才有效。

- d *h_file*

定义输出的系统信息头文件名

如果省略该项:系统修改 cf_file 定义的系统配置文件名的扩展名为.h，然后将该文件作为系统信息表文件输出。

备注 1:定义的输出文件名 h_file.包括路径名在内最多为 255 个字母。

备注 2: 当这个激活选项和-nd 选项被同时定义时，只有最后一个输入的才有效。

- nd

禁止系统信息头文件的输出。

如果省略该项:系统修改 cf_file 定义的系统配置文件名的扩展名为.h，然后将该文件作为系统信息表文件输出。

备注:当这个系统激活选项和-d h_file 选项在同时被定义时，只有最后一个输入的才有效。

- V

把 CF850 的版本信息输出到标准输出设备上。

如果省略该项:不输出 CF850 的版本信息。

备注:定义这个激活选项将使所有其他激活选项无效。

- help

将 CF850 的激活选项使用方法输出到标准输出设备上。

如果省略:不输出 CF850 的激活选项使用方法。

备注:定义这个激活选项将使所有其他激活选项无效。

- *cf_file*

指定输入到 CF850 的系统配置文件名。

如果省略该项:这个激活选项不能被忽略。

备注:输入文件名sit_file包括路径名在内最多为255个字母。

6.2.2 从 PM+激活

下面的内容是介绍如何从 CA850 提供的集成开发环境平台 PM+设置 CA850 的激活选项。

另外，下面这个例子是已存在工程文件的情况下的设置方法。

1) 启动 PM+。

通过点击快捷图标开启 PM+(默认:Windows start menu -> [Program] -> [NEC Electronics Tools] -> [PM+] -> [Vx.xx] ->[PM+ Vx.xx])或者双击可执行文件(默认:C:\Program Files\NEC Electronics Tools\PM+\V*x.xx*\bin\PMplus.exe)。

2) 打开 [Open Workspace] 对话框。

通过选择[File] menu -> [Open Workspace...]打开[Open Workspace]对话框。

[注意]关于[Open Workspace]对话框的详细资料，参考 PM+用户手册。

3) 指定一个工作空间文件。

设置[Look in:]区域，[File name:] a 区域和[Files of type:]区域，然后点击 <OK>键指定工作空间文件(或者工程文件)，该文件设置了 CF850 的激活选项。

4) 打开 [Setting OS]对话框。

通过选择[Tool] menu -> [Select OS...]打开 [Setting OS] 对话框。

[警告] 关于[Setting OS] 对话框详细资料，参阅 “附录 A WINDOW 参考资料”。

5) 打开 [RX850 Settings]对话框。

从[Select OS] 区中列表框选择”RX850 V3.*xx*”之后，通过点击<OK>键打开[RX850 Settings]对话框

[警告] 关于[RX850 Settings]对话框详细资料，参阅”附录 A WINDOW 参考资料”。

6) 指定一个系统配置文件。

在[System configuration file:]区中指定输入文件名(系统配置文件名)。

7) 定义一个系统信息表文件。

在选取 [Generate a system information table file [-i/-ni]]复选框之后，在[File name]区定义输出文件名(系统信息表文件名)。

8) 定义一个系统信息头文件。

在选取 [Generate system information header file [-d/-nd]]复选框之后，在[File name]区定义输出文件名(系统信息头文件名)

9) 检查激活选项。

在[Command line options] 区显示了由过程 6)到 8)定义的 CF850 激活选项格式，它允许显式的检查是否上面过程定义的结果与预期的结果是否一致。

10) 在工程文件中反映操作结果。

点击<OK>键使上面操作的结果反映到工程文件中。

6.2.3 命令文件

出于消除命令行中可定义激活选项字母数的限制的目的，CF850 执行命令文件。

命令文件的编写格式说明如下。

1） 注释行

由#开始的行被认为是注释行。

2）激活选项

为了定义不同的激活选项，在它们之间插入一个空格或者换行符。

为了定义由一个参数和一个-xxx 部分例如-cpu, -i, and -d 所组成的激活选项，在参数和-xxx 部分之间插入一个空格或者换行符。

[注意事项] 当一个-devpath 参数(它的路径名为 *path*)有一个包含一个空格的文件名，这个参数必须括在双引号内。

- devpath="Program Files\DEV"

3）字母的最大数目

在一个命令文件中，单行最大可写 4096 个字母

图 6-1 示出了一个适用于 CA850 版本的命令文件的例子。

在下面图 6-1 的例子中，假设编写以下的激活选项。

目标设备	:uPD703000
设备文件的参考文件夹	:C:\Program Files\NEC Electronics Tools\dev
寄存器模式	:r26
系统信息表文件	:sit.s
系统信息头文件	:sit.h
系统配置文件	:sit.cf

图 6-1 命令文件说明(用于 CA850 版本)

```
# Command File
-cpu
3000
-devpath=C:\Program Files\NEC Electronics Tools\dev
-reg26
-i sit.s
-d
sit.h sit.cf
```

6.3 命令输入举例

6.3.1 适用于 CA850 版本的命令输入

适用于 CA850 版本 CF850 命令输入的例子如下。在这个例子中，uPD703000 作为目标设备使用。

- cf850 -cpu 3000 -devpath=C:\Program Files\NEC Electronics Tools\dev -reg26 -i sit.s -d sit.h sit.cf

  这个命令从当前文件夹读取作为输入文件的系统配置文件 sit.cf。它同时也从 C:\Program Files\NEC Electronics Tools\dev 文件夹读取作为输入文件的设备文件支持类型定义名 3000。然后，它输出 26-寄存器模式文件格式的系统信息表文件 sit.s 和系统信息头文件 sit.h。

- cf850 -cpu 3000 -devpath=C:\Program Files\NEC Electronics Tools\dev -i sit.s sit.cf

  这个命令从当前文件夹读取作为输入文件的系统配置文件 sit.cf。它同时也从 C:\Program Files\NEC Electronics Tools\dev 文件夹读取作为输入文件的设备文件支持类型定义名 3000。然后，它输出 32-寄存器模式文件格式的系统信息表文件 sit.s 和系统信息头文件 sit.h。

- cf850 -cpu 3000 -devpath=C:\Program Files\NEC Electronics Tools\dev -d sit.h sit.cf

  这个命令从当前文件夹读取作为输入文件的系统配置文件 sit.cf。它同时也从 C:\Program Files\NEC Electronics Tools\dev 文件夹读取作为输入文件的设备文件支持类型定义名 3000。然后，它输出 32-寄存器模式文件格式的系统信息表文件 sit.s 和系统信息头文件 sit.h。

- cf850 -cpu 3000 -devpath=C:\Program Files\NEC Electronics Tools\dev sit.cf

  这个命令从当前文件夹读取作为输入文件的系统配置文件 sit.cf。它同时也从 C:\Program Files\NEC Electronics Tools\dev 文件夹读取作为输入文件的设备文件支持类型定义名 3000。然后，它输出 32-寄存器模式文件格式的系统信息表文件 sit.s 和系统信息头文件 cffile.h。

- cf850 -cpu 3000 -nd sit.cf

  这个命令从当前文件夹或..\..\..\dev 文件夹读取作为输入文件的系统配置文件 sit.cf 和支持类型定义名 3000 的文件。然后，它输出 32-寄存器模式文件格式的系统信息表文件 sit.s。

- cf850 -cpu 3000 -ni sit.cf

  这个命令从当前文件夹或..\..\..\dev 文件夹读取作为输入文件的系统配置文件 sit.cf 和支持类型定义名 3000 的文件。然后，它输出 32-寄存器模式文件格式的系统信息头文件 cffile.h。

- cf850 -V

  这个命令向标准控制台输出 CF850 的版本信息。

- cf850 -help

  这个命令向标准控制台输出 CF850 的激活选项的使用方法。

6.3.2 适用于 GHS 编译器版本的命令输入

适用于 GHS 编译器版本 CF850 的命令输入例子如下。

- cf850 -reg26 -i sit.s -d sit.h sit.cf

  这个命令从当前文件夹读取作为输入文件的系统配置文件 sit.cf。然后，它输出适用于 26-寄存器模式的文件格式的系统信息表文件 sit.s 和系统信息头文件 sit.h。

- cf850 -i sit.s sit.cf

  这个命令从当前文件夹读取作为输入文件的系统配置文件 sit.cf。然后，它输出适用于 26-寄存器模式的文件格式的系统信息表文件 sit.s 和系统信息头文件 cffile.h。

- cf850 -d sit.h sit.cf

  这个命令从当前文件夹读取作为输入文件的系统配置文件 sit.cf。然后，它输出适用于 32-寄存器模式的文件格式的系统信息表文件 sit.s 和系统信息头文件 sit.h。

- cf850 sit.cf

  这个命令从当前文件夹读取作为输入文件的系统配置文件 sit.cf。然后，它以适用于 32-寄存器模式的文件格式输出系统信息表文件 sit.850。

- cf850 -nd sit.cf

  这个命令从当前文件夹读取作为输入文件的系统配置文件 sit.cf。然后，它以适用于 32-寄存器模式的文件格式输出系统信息表文件 sit.850。

- cf850 -ni sit.cf

  这个命令从当前文件夹读取作为输入文件的系统配置文件 sit.cf。然后，它以适用于 32-寄存器模式的文件格式输出系统信息头文件 cffile.h。

- cf850 -V

  这个命令向标准控制台输出 CF850 的版本信息。

- cf850 -help

  这个命令向标准控制台输出 CF850 的激活选项的使用方法

6.4 消息

在处理过程中一旦检测到诸如”系统配置文件中错误定义”等编写错误，CF850 立即生成消息并输出到标准控制台上。

消息分为 3 个级别:致命错误，非致命错误，警告错误。CF850 在输出的消息开头添加一个表示错误等级的字母。

F :致命错误

如果发生致命错误，CF850 输出一个消息，然后中止配置进程。

例子: Insufficient memory area.

E :非致命错误

如果非致命错误发生，CF850 输出一个消息阻止配置进程。

例子: Duplicated definition.

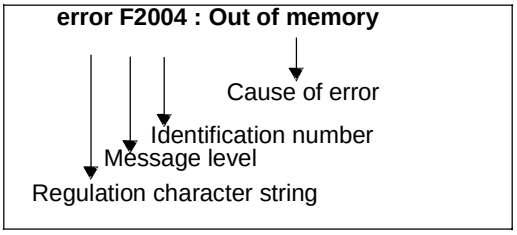
W 警告

如果警告错误发生，CF850 输出一个消息并且继续配置进程。

例子: The description of a parameter has been omitted.

图 6-2 示出了消息格式:

图 6-2 消息格式



CF850 在进程完成之后时输出下面的终止消息

注意"%d" 表示检测出的错误和警号数目

total error(s) : %d total warning(s) : %d

6.4.1 致命错误

下面列出了致命错误发生时输出的消息。

在下面的消息中，如果相关的致命错误发生，那些用斜体字写的项目(e.g., *file_name*)是确定的:

F2001 : 用法:cf850 [@command_file] [-cpu <name>] [devpath=<path>] [-reg22] [-reg26] [-reg32] [-i <SIT file>] [-d <include file>] [-ni] [-nd] [-V] [-help] <file>

对于 CF850 和 CA850 版本，激活选项指示无效。

F2001 : 用法: cf850 [@command_file] [-reg22] [-reg26] [-reg32] [-i <SIT file>] [-d <include file>] [-ni] [-nd] [-V] [-help] <file>

对于 CF850 和 GHS 编译器版本，激活选项指示无效。

F2002 : Can't allocate memory.内存不足。

F2003 : Can't open file *fime_name*. 文件 *file_name* 不能打开。

F2004 : Out of memory. 内存不足。

F2005 : Can't Open device file.设备文件不能打开。

F2006 : Can't ready device file.设备文件不能读取。

F2007 : Unknown device file format

指定不支持的设备文件。

F2008 : Output file "*file_name*" names are the same.

为输出文件中指定了相同的名字(系统信息表文件和系统信息头文件)。

6.4.2 非致命错误

下面列出了非致命错误发生时输出的消息。

在下面的消息中，如果相关的非致命错误发生，那些用斜体字写的项目(e.g., *file_name*)是确定的:

E2001 : ser_def not defined.

Real-time OS information 的开始声明，ser_def，在第一行没有被找到

E2002 : Illegal ser_def.

Real-time OS information 的开始声明，ser_def，遭遇与正确位置不同的位置。

E2003 : ser_def already defined.

Real-time OS information 的开始声明，ser_def，被定义了 2 次。

E2004 : sit_def not defined.

SIT (system information table) information 的开始声明，ser_def，没有找到。

E2005 : Illegal sit_def.

SIT (system information table) information 的开始声明，ser_def，遭遇与正确位置不同的位置。

E2006 : sit_def already defined.

SIT (system information table) information 的开始声明，ser_def，被定义了 2 次。

E2007 : Out of sit_def division.

属于 SIT (system information table) information 的一部分的数据在 SIT (system information table) information，sit_def 开始声明之前被定义。

E2012 : rxsers not defined.

RX series information rxsers i 没找到。

E2013 : Illegal rxsers.

RX series information rxsers 没有在正确的位置被定义。

E2014 : rxsers already defiend.

RX series information rxsers 被定义了 2 次。

E2101 : Integer overflow.

数字值超过了 32-bit data 范围。

E2102 : Syntax error.

系统配置编码格式不合法。

E2103 : Word too long.

符号名超过了最大特性数目。

E2104 : Address out of range.

一个指定地址超出了可允许的范围。

E2105 : Address must be aligned by 2.

在一个 2-byte 分界指定一个地址。

E2106 : Stack size out of range.

一个超过可允许范围的堆栈大小被指定。

E2107 : Symbol *name* already defined. 符号名 "*name*" 被指定 2 次。

E2201 : System stack size not defined.

系统堆栈信息 intstk 没有被定义。

E2202 : System stack size already defined. 系统堆栈信息intstk被定义2次。

E2203 : Interrupt level *name* already defined.

2 个或以上间接激活中断处理器以相同的中断源"*name*"注册。

E2204 : Interrupt level of system clock already defiend.

时钟中断源 clkhdr 被定义 2 次。

E2206 : Priority level already defined.

System maximum value information maxpri 被定义 2 次。

E2207 : Priority level out of range.

System maximum value information maxpri 超出可允许范围。

E2208 : Task group *name* already defined.

任务执行右组名"*name*"定义了 2 次。

E2209 : Task not defiend.

Task information tsk is 没有被定义。

E2210 : Too many tasks.

有 128 或者更多 Task information tsk 定义。

E2211 : Task *name* already defined.

任务名"*name*"被定义了 2 次。

E2212 : Task group *name* in task not defined.

任务执行右组名"*name*"没有找到。

E2213 : Task priority out of range.

初始优先 *pri* 没有在范围 0x1 到 0x1f 之间。

E2214 : Too many eventflags.

有 128 或者更多 Event flag information 标志定义。

E2215 : Eventflag *name* already defined. The

事件标志名"*name*"被定义 2 次。

E2216 : Too many 1bit eventflags.

有 128 或者更多 1-bit event flag information 标志 1 定义。

E2217 : 1bit eventflag *name* already defined. The 1-bit 事件标志"*name*"被定义 2 次。

E2218 : Too many Semaphores.

有 128 或者更多信号量信息 sem 定义。

E2219 : Semaphore *name* already defined. The 旗语名 "*name*" 被定义 2 次。

E2220 : Semaphore resource count out of range.

初始资源计数 *nit_cnt* 没有在范围 0x0 t 到 0x7f 之间。

E2221 : Too many mailboxes.

有 128 或者更多 Mailbox information mbx 定义。

E2222 : Mailbox *name* already defined. The 邮箱名 "*name*" 被定义 2 次。

E2223 : Too many fixed-size memorypools.

有 128 或者更多 Fixed-size memory pool information mpf 定义。

E2224 : Fixed-size memorypool *name* already defined. The 固定大小内存池名 "*name*"被定义了2次。

E2225 : Memory block size or block count is 0.
基本块大小 *blk_siz* 或者内存块 *blk_cnt* 总 *t* 数目在 0x0 定义。

E2226 : Memory block size must be aligned by 4.
基本块大小 *blk_siz* 必须在一个 4-byte 分界。

E2227 : Memory total size exceeds 4Gbytes.
系统内存总大小超过了 4Gbytes。

E2228 : Too many cyclic handlers.
有 128 或者更多 Cyclic handler information cyc 定义。

E2229 : Cyclic handler name already defined. The cyclic
处理器名 "*name*" 被定义 2 次。

E2230 : Interval time out of range.
用于 cyclic 处理器激活间隔超出了 0x1 到 0xffff ffff 的范围。

E2231 : Too many variable-size memorypools.
有 128 或者更多 Variable-size memory pool information mpl 定义。

E2232 : Variable-size memorypool *name* already defined.
可变长内存池名"*name*"定义了 2 次。

E2233 : Too small variable-size memorypool size.
有 128 或者更多 Variable-size memory pool information mpl 定义。

E2234 : Variable-size memorypool size must be aligned by 4.
可变长内存池大小必须在一个 4-byte 分界。

E2235 : Illegal trace.
追踪信息没有在正确的位置被定义。

E2236 : Trace already defined。

跟踪信息被定义了 2 次。

E2237 : regmod already defined.
寄存器模式 regmod 被定义了 2 次。

E2238 : The character cannot specify for register_mode.
一个不合法的关键词与寄存器模式*reg_mode* 一起使用。

6.4.3 警告

下面列出了当警告错误发生时输出的消息。

在下面的消息中，如果相关的警告错误发生，那些用斜体字写的项目(e.g., *file_name*)是确定的:

W2201 : Use *pri1* priority level, but priority *pri2* task is defined, define *pri2* priority level are used.

尽管 *pri1* 定义为任务优先范围，但初始任务优先级定义为 *pri2*。CF850 认为 *pri2* 被定义成任务优先范围，然后继续处理。

W2202 : Task group *name* is not used in task, ignored.

尽管 *name* 定义为任务执行权组，在任务信息中没有定义。CF850 认为任务执行权组"*name*"没有定义，并继续处理。

W2203 : Stack size must be aligned by 4, round up stack size.

定义的堆栈大小不在 4-byte 边界。

CF850 上舍入堆栈大小，以便堆栈与最近的 4-byte 边界对齐，然后继续处理。

W2204 : Register mode specified in CF file is different -reg option. (-reg option assumed)

System information 中定义的寄存器模式与激活选项定义的寄存器模式不一致。CF850 认为 System information 中定义的寄存器模式无效的，激活选项定义的寄存器模式有效的，然后继续处理。

W2205 : Nested command file "*file_name*".

命令文件"*file_name*"中定义了一个错误的激活选项"*@cmd_file*"。CF850认为"*@cmd_file*"没有定义，并继续处理。

附录 A　窗口参考

这个附录介绍当从 CA850 提供的开发环境平台 PM+定义适用于 CF850 的激活选项时，使用的对话框。

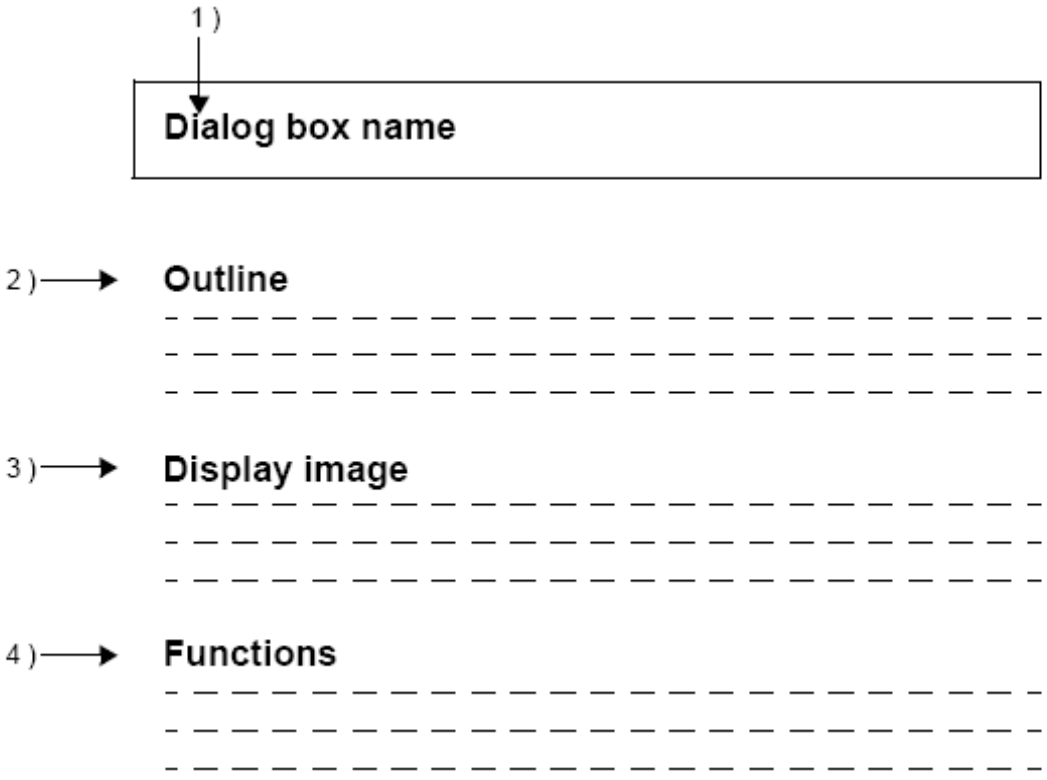
A.1　概况

表 A-1 对话框列表

对话框名	功能
[OS 设置] 对话框	这个对话框用来定义下面信息: 实时操作系统的名称
[RX850 设置] 对话框	这个对话框用于定义下列作为 CF850 激活选项的信息: . - 系统配置文件名 - 系统信息表文件名 - 系统信息头文件名
[选择系统配置文件] 对话框	这个对话框用于载入一个已存在的系统配置文件
[RX850 错误] 对话框	这个对话框主要用来显示错误信息.

A.2 Windows 解释

本节根据下面的说明格式来解释每个对话框



1) 对话框名

   显示每个对话框名

2) 概要

   介绍功能概要和如何打开对话框

3) 显示图象

   显示每个对话框的图象

4) 功能

   根据结构元素来详细介绍功能

[OS 设置] 对话框

概要

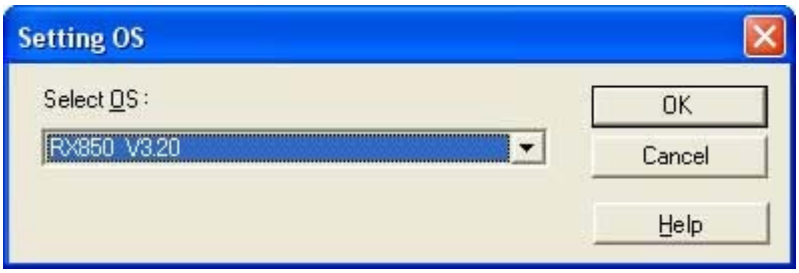
这个对话框主要用来定义下面信息:

- 实时操作系统名称

这个对话框可以按照下面操作打开:

- 在 PM+ 主窗口上选择[工具]菜单->[选择 OS…]

显示图象



功能

1) [选择 OS:]区

- 列表框

选择使用的实时操作系统名。

注意，在这个对话框中，只能定义”RX850 V3.xx”的对象。

2) 功能按钮

- <OK >按钮

打开[RX850 设置]对话框

- <Cancel>按钮

关闭这个对话框

- <Help>按钮

打开该对话框的在线帮助

[RX850 设置] 对话框

概要

这个对话框用于定义下列作为 CF850 激活选项的信息: .

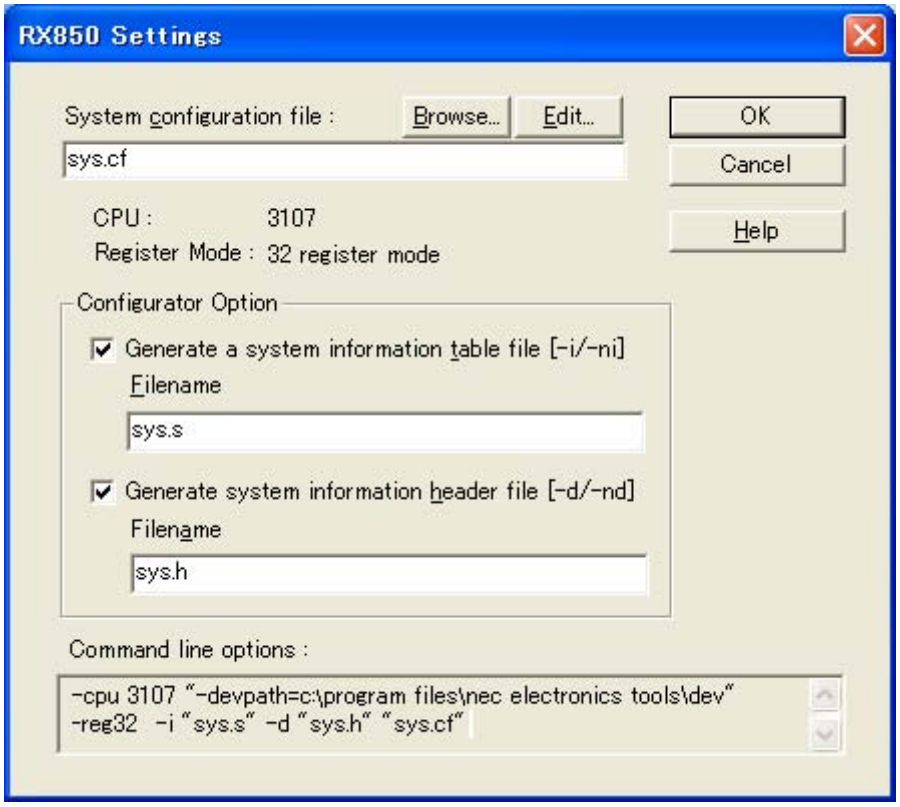
- 系统配置文件名
- 系统信息表文件名
- 系统信息头文件名

这个对话框可以按照下面操作打开:

- 选择[OS 设置]对话框上的"RX850 V3.xx"，然后，选择<OK>按钮。

[备注] 没有必要设置将在设备文件中搜索的类型定义名称、寄存器模式、文件夹名，因为在 CA850 所提供的 PM+开发环境中的[Project 设置]对话框中已经反映了这些信息项。如果要详细了解[Project 设置]对话框，参考用户手册 PM+ 。

显示图像



功能

1) [系统配置文件:]区

-文本框

定义输入到 CF850 的输入文件名(系统配置文件名)

[注意事项] 定义的输入文件名包括路径名应该在255字符内

- [浏览…]按钮

打开[选择系统配置文件]对话框

-[编辑…]按钮

打开编辑窗口

[注意事项 1] 如果要了解编辑窗口，参考用户手册 PM+

[注意事项 2] 当在编辑窗口显示的系统配置文件的内容用 PM+提供的编辑器"ideal-L"编辑时，本对话框和[OS 设置]对话框必须关闭。

2) [配置器选项]区

- [Generate a system information table file [-i/-ni]] 复选框当 CF850 激活时，定义系统信息表文件是否输出

选择:用在[文件名]区指定的文件名输出系统信息表文件

不选择: 禁止输出系统信息表文件

- [Filename] 区

当 CF850 激活时，定义输出文件名(系统信息表文件名)

[注意事项] 定义的输出文件名包括路径名应该在 255 字符内

- [Generate system information header file [-d/-nd]] 复选框

当 CF850 激活时，定义系统信息头文件是否输出

选择:用在[文件名]区定义的文件名输出系统信息头文件

不选择: 禁止输出系统信息表文件

- [Filename] 区

当 CF850 激活时，定义输出文件名(系统信息头文件名)

[注意事项] 定义的输出文件名包括路径名应该在 255 字符内

3) [命令行选项:]区

以适合 CF850 的命令输入格式显示用[系统配置文件:]区和[配置器选项]区(包含用 PM+开发环境平台 Project 设置对话框定义的信息)定义的信息。

4) 功能按钮

- <OK>按钮

将用[系统配置文件:]区和[配置器选项]区(包含 PM+开发环境平台用 Project 设置对话框中所定义的信息)所定义的信息作为 CF850 激活选项在 PM＋中反映，关闭对话框。

- <Cancel>按钮

关闭这个对话框

- <Help>按钮

打开这个对话框的在线帮助

[选择系统配置文件] 对话框

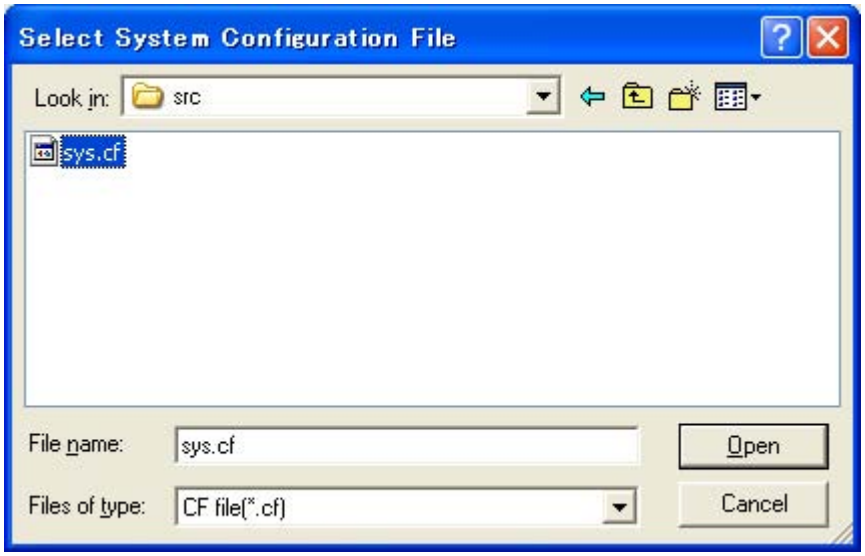
概要

这个对话框用来载入一个存在的系统配置文件

这个对话框可以按照下面操作打开

在[RX850 设置]对话框上选择<浏览…>

显示图象



功能

1) [搜索:]区

- 组合框

选择存放系统配置文件的文件夹

2) 文件名显示区

列出基于[搜索:]区和[文件类型:]区的文件名

3) [文件名:]区

- 文本框

定义要载入的系统配置文件名

4) [文件类型:]区

- 组合框

选择要在文件名显示区显示的文件类型

5) 功能按钮

- <打开>按钮

加载用[搜索:]区和[文件名:]区定义的系统配置文件

- <取消>按钮

关闭这个对话框

[RX850 错误] 对话框

概要

这个对话框用来显示错误信息

当在[RX850 设置]对话框中设置了错误信息，这个对话框自动打开。等等。

显示图象

功能

1) 消息显示区

显示与已探测到的错误关联的消息(消息级别 / 识别号.,错误原因)

消息级别 识别号	错误原因
E1006:	文件名太长
E1022:	系统配置文件名或路径名错误
E1023:	系统信息表文件名或路径名错误
E1024:	系统信息头文件名或路径名错误
E2011:	系统配置文件名不存在
E2012:	系统信息表文件名不存在
E2014:	系统信息头文件名不存在
E2040:	在线帮助文件名不存在

2) 功能按钮

<OK>按钮

关闭此对话框

索　引