前　言

MSP430 系列是美国德州仪器公司生产一种特低功耗的 Flash 微控制器，有业界最佳“绿色微控制器（Green MCU）”称号，其技术特征代表了单片机的发展方向。其存储器模块是目前业界所有内部集成 Flash 存储器产品中能耗最低的一种，消耗功率仅为其它闪烁微控制器（Flash MCU）的五分之一。

MSP430 微控制器具有高效的 16 位 RISC 内核，27 条指令，125ns 指令周期，绝大多数指令在一个时钟周期内完成，供电电压 1.8V~3.6V，程序代码加密后无法解密，只需外接一个电阻、一个电容即可实现高精度斜率 A/D 转换，待机激活时间仅需 6μs，串行在线编程的开发方式可以迅速地开发新品。另外，MSP430 系列具体有强大的中断功能、10 万次擦写、ESD 保护和超强抗干扰能力等。

MSP430 微控制器可广泛地用于煤气表、电子电度表、火警智能接头、通信产品、家庭自动化产品、便携式监视器及其它低能耗产品。由于 MSP430 微控制器的功耗极低，可设计出只需一节电池就可以使用长达 10 年的应用仪表产品。同时内部集成闪速存储器，也使 MSP430 微控制器系列产品的应用范围更加完整。

本书根据德州仪器公司 2001 年的光盘中《MSP430 Windows Workbench Interface Guide》一书译出，介绍了一种功能强大的集成开发环境 Windows Workbench，用于为 MSP430 系列微处理器开发 C 或汇编应用程序，是开发编程 MSP430 器件必读的资料。

利尔达电子（中国）有限公司

第 1 章　安装及引言

1.1　安装

将 MSP430 光盘放入光驱自启动，找到：　Click here to install MSP430 Tools

安装：MSP-FET430P140 Flash Emulation Tool

或者，直接查找：FET_R202.EXE 文件安装。

对 C 语言用户

安装：MSP-FET430 Upgrade Utility

用 MSP-PRG430S320 烧写程序时

安装：MSP-PRGS430 Programming Adapter

第2章　　引　　言

IAR 系统嵌入式 Workbench 是一种用于开发应用各种不同的目标处理器的灵活的集成环境。它提供一个方便的窗口界面用于迅速的开发和调试。

嵌入式 Workbench 支持多种不同的目标处理器，用户用不同的目标处理器开发的工程（Projects）可以在工程的基础上逐个规定目标工程。关于支持目标处理器的更多的信息请与当地的 IAR 销售商联系。

工具包括：快速编译器、高效的连接器、库、语法高亮度文本编辑器、自动的 Make 工具以及一个可选的的 C-SPY 调试器。

2.1　嵌入式 Workbench

IAR 系统嵌入式 Workbench 提供以下特点：

2.1.1　通用性

- 可在 Windows95、Windows NT 或 Windows3.11 下运行。
- 分层的项目（Project）表示。
- 直观的用户界面，利用了 Windows95 的优点。
- 只在必须时使用 Make 实用重新编译程序、重新汇编程序和连接文件。
- 嵌入式 Workbench 工具和编辑器的全集成。
- 支持阻滞和跌荡。
- 全面的超文本帮助。

2.1.2　嵌入式 Workbench 编辑器

- C 程序的句法用文本格式和颜色显示。
- 有力的搜套和置换命令，包括多个文件搜索。
- 从出错列表直接跳转到相关文件。
- 圆括号匹配。
- 自动缩进（indentation）。
- 每个窗口的多级取消和恢复（undo and redo）。

2.1.3　C 编译器和汇编器

- 项目在 Windows95 或 Windows NT 的背景下建立，允许同时编辑。
- 可以全局地设置选项，对多源文件或对单独的源文件。

2.2　C 编译器

运用于 MSP430 微处理器的 IAR 系统 C 编译器提供 C 语言的标准特性，再加上许多为利用 MSP430 专用工具而设计的扩展功能。编译器与 MSP430 IAR 系统汇编器一起提供，与它集成在一起，共享连接器和库管理工具。

它提供以下特性:

2.2.1 语言工具

- 与 ANSI 规格一致。
- 可应用于嵌入式系统的标准函数库，具有可选用的源（代码）。
- IEEE 兼容的浮点算法。
- 对 MSP430 特殊性能的有力扩展，包括高效的 I/O。
- 程序源的 LINT-like 检查。
- 用户代码与汇编子程序连接。
- 长识别符——多达 255 个有效字符。
- 多达 32000 个外部符号。
- 与其他 IAR 系统的 C 编译器有最大的兼容性。

2.2.2 性能

- 快速成编译。
- 避免暂时文件或覆盖的基于存储器的设计。
- 编译时严格的类型检查。
- 连接时严格的模块接口类型检查。

2.2.3 代码产生

- 可选择的代码速度或大小的最佳化。
- 综合输出选项，包括可重定位二进制、ASM、ASM+C、XREF 等等。
- 易于理解的出错和警告消息。
- 与 C-SPY 高级调试器兼容。

2.2.4 目标支持

- 灵活的变量分配。
- 不需要汇编语言的中断函数。
- 使用权用处理器专用扩展时保持可移植性的#Pragma 伪指令。

2.2.5 文档

- MSP430 C 编译器的文档是《MSP430 C Compile Programming Guide》。

2.3 汇编器

IAR 系统 MSP430 汇编器是一种功能强大具有通用伪指令组的重新定位宏汇编器。

该汇编器与微处理器制造商指令用的汇编器有高度的兼容性，以保证工厂原始开发的软件只需很少或者不需修改就可转换到 IAR 系统。

它有以下特点:

2.3.1 通用性

- 一旦通过汇编，即可快速执行。
- 与 XLINK 连接器和 XLIB 库集成在一起。
- 与其它 IAR 系统软件集成在一起。
- 自己说明的出错信息。

2.3.2 汇编器特性

- 支持 MSP430 系列微处理器。
- 每个模块有高达 256 个可重新定位的段。
- 32 位算术和 IEEE 浮点常数。
- 255 个有效的字符符号。
- 高效递归宏工具。
- 符号的数目和程序的大小只受可用存储器的限制。

- 支持带有外部参考的复杂的表达式。
- 前向基准允许有任何深度。
- 支持 C 语言预处理器伪指令和 sfr 关键词。
- Intel/Motorola 型的宏。

2.3.3 文档

MSP430 汇编器的文档是《MSP430 Assembler, Linker, and Librarian Programming Guide》。

2.3.4 XLINK 连接器

IAR 系统 XLINK 连接器把 IAR 系统汇编器或 C 编译器产生的一个或多个可重定位目标文件转换为特定目标处理器的机器代码。除了支持 C-SPY 高级调试器所使用的 IAR 系统调试格式外，它还支持许多业界标准（industry-standard）装载器格式。

XLINK 支持用户库，而且只装载用户正在连接中程序所实际需要的那些模块。

XLINK 产生的最终输出是绝对的、目标可执行（target-executable）的目标文件，它可以被编程入 EPROM，下载到硬件仿真器，或者直接在使用 IAR 系统 C-SPY 调试器的主机上运行。

XLINK 提供下列重要特性：

2.4.1 XLINK 的特点

- 输入文件数目不受限制。
- 搜索用户定义的库文件并且只装载应用程序所需的那些模块。
- 符号可长达 255 个字符，所有的字符均有效。大写和小写均可使用。
- 连接时可以定义全局符号。
- 灵活的段（segment）命令可完全控制可重定位代码和数据在存储器中的地址。
- 支持超过 30 种的仿真器格式。

2.4.2 文档

有关 XLINK 连接器的文档包括在《MSP430 Assembler, Linker, and Librarian Programming Guide (MSP430 汇编器、连接器和库管理器编程指南)》中。

2.5 XLIB 库管理器

IAR 系统 XLIB 库管理器使用户能处理由 IAR 系统汇编器和 C 编译器产生的可重定位目标文件。

XLIB 提供下列特性：

2.5.1 XLIB 的特性

- 支持模块化编程。
- 模块可以被列表、添加、插入、替代、删除或重新命名。
- 段可以被列表和重命名。
- 符号可以被列表和重命名。
- 模块可以在程序和库类型之间改变。
- 交互（Interactive）或批（batch）模式操作。
- 整组库（A full set of library）列表操作。

2.5.2 文档

有关 XLIB 库管理器的文档包括在《MSP430 Assembler, Linker, and Librarian Programming Guide (MSP430 汇编器、连接器和库管理器编程指南)》中。

2.6 C-SPY 调试器

可选的 C-SPY 调试器可以加到嵌入式工作平台 Embedded Workbench 中，以便运行和调试 MSP430 目标代码程序，如果添加了 C-SPY 调试器，那么可以从菜单和工具栏进行访问。

第3章 概 述

IAR 嵌入式工作平台 Embedded Workbench 为开发不同的目标处理器的项目提供强有为的开发环境，并为每一种目标处理器提供工具的选择。

本章给出嵌入式工作平台 Embedded Workbench 使用的项目模式（Project model）的简要讨论，并说明用户怎样用它来开发典型的应用程序。

3.1 怎样组织项目

嵌入式工作平台 Embedded Workbench 被专门设计成能适合通常的软件开发项目的组织方式。例如，用户可能需要开发适合于不同版本目标硬件的应用程序的相应版本，也可能想要的调试子程序包含到早期版本内，但不包含在最终代码中。

适用于不同目标硬件的用户应用程序版本常常具有通用的源文件，用户想要维护这此文件的唯一副本，以便对应用程序的每一个版本自动地进行改进。也存在在应用程序的不同版本之间有差异的源文件，例如与应用程序依赖于硬件的方面有关的那些文件，因此，这些文件将需要分别维护以适应每一个目标版本。

嵌入式工作平台 Embedded Workbench 符合这些需求，提供功能强大的开发环境，它适合于维护用于建造应用程序所有版本的源文件。它允许用户以树状体系结构组织项目，这种树状结构能一目了然地显示文件之间的依赖关系。

3.1.1 目标（TARGETS）

在结构的最高层，用户规定了他想要建立的应用程序的不同目标版本。对于简单的应用程序，用户可能只需要两个目标，称之为 Debug(调试)和 Release(发布)。较复杂的项目可能包含另外的目标，它们适用于每一种应用程序将在其上运行的不同的处理器的类别（variants）。

3.1.2 源文件（SOURCE FILES）

每一个组用于把一个或多个相关的源文件组合在一起，每一个组可以被包含在一个或多个目标中以达到最大的灵活性。此外，每一个源文件可以包含在一个或多个组中，虽然由于达连接时可能产生问题，这种做法并不被推荐。

当用户使用项目(Project)工作时，他总是有一个选定的当前目标(current target)，在Project(项目)窗口中，只有作为该目标成员（member）的组以及它们所包括的文件才是可见的。只有这些文件将真正被建立并连接到输出代码中。

3.2 设置选项

对于每一个目标，用户在目标层（target level）设置全局的汇编器和编译器选项，以规定怎样建立目标。在这一层上，用户通常定义他将使用的存储模式（memory model）以及处理器类型（processor variant）。

用户也可以在各个组和源文件上设置局部编译器和汇编器选项。这此局部选项将压倒（override）在目标层设置的任何相应的全局选项，并且是该目标所特有的。一个组可以含在两个不同的目标中且在每一个目标内可以具有不同的选取项设置。例如，对于包含已调试的源文件的组，用户可以把最佳化（optimization）设置为高（high）；但是，对于另一包含仍在开发之中的源文件的组，用户可以从中去掉最佳化（optimization）。

3.3 建立项目

嵌入式工作平台 Embedded Workbench Project（项目）菜单上的 Compile（编译）命令允许用户单独编译或汇编项目的文件，并调度任何产生的错误。嵌入式工作平台 Embedded Workbench 根据文件的扩展名，自动决定源文件应当被编译还是被汇编。

另一方面，用户可以建立整个项目，使用 Make（生成）命令自动编译和汇编所有的组成文件。这等同于在文件发生改变时，根据文件是否变化以及它们对于其他文件的依赖关系，在重新连接项目之前，仅仅重新编译或汇编必需的文件。

Build All （建立全部）选项也被提供，此选项将重新产生所有的文件，而不管它们是否已被编辑。

当在 Windows NT 或 Windows95 上运行嵌入式工作平台 Embedded Workbench 时，Compile（编译）、Make （生成）、Link（连接）以及 Build（建立）命令全都在后台运行，进行编辑或工作。当在 Windows3.1 下运行时，将显示一个对话框以便允许用户在需要时注销它。

3.4 测试代码

编译器和汇编器完全和开发环境集成在一起，所以如果在用户源代码中存在错误的话，那么用户可以错误列表直接跳到合适的源文件中需纠正的位置，使用户能定位并纠正错误。

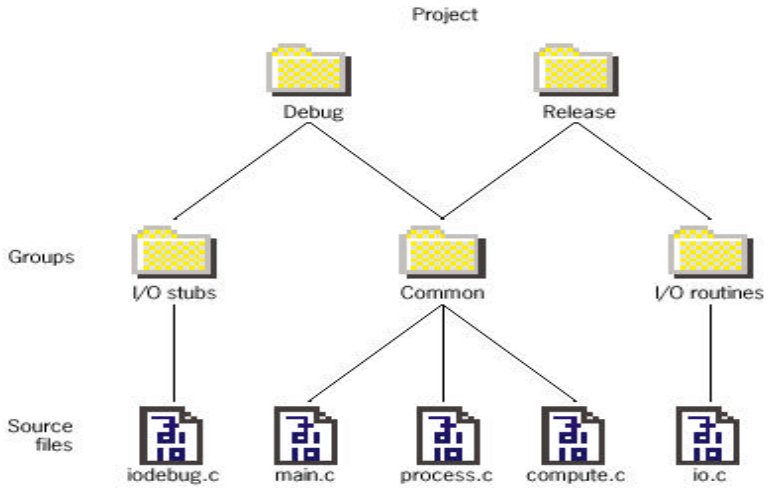
当用户解决了任何编译时（Compile-time）错误之后，他可以直接转到 C-SPY 调试器，以便在源文件层（source level）测试产生的代码。C-SPY 调试器在分开的窗口中运行，以便当用户在 C-SPY 中识别出问题时，可以对原先的源文件作出修改从而纠正这些问题。

3.5 样本应用程序

下面的例子叙述了两个样本应用程序以说明在典型的开发项目中怎样使用嵌入式工作平台 Embedded Workbench。

3.5.1 简单应用程序

如下图所示，在用户正在开发的简单应用程序中，对于目标硬件的一种类别，用户可能创建 Release（发行）和 Debug（调试）目标:



两个目标共用包含项目核心源文件的公共组（common group）。每一个目标还包含一个组，它包含了专用于该目标的源文件:

I/O routines 组——它包含有关被用于最终发行代码的输入/输出子程序的源文件；

I/O stubs——它包含输入/输出短程序（stubs），以便用 C-SPY 这样的调试器调试 I/O。

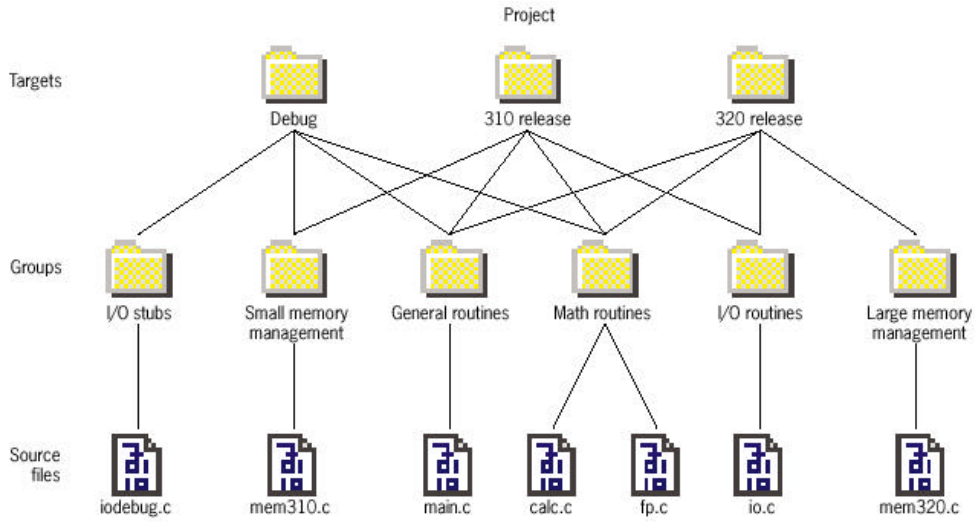
发行（release）和调试（Debug）目标通常具有适用于它们的不同的编译器选项；例如，用户可以用 trace（跟踪），assertions（确定）等编译 debug（调试）版本，编译 release（发行）版本时则没有这些选项。

3.5.2 较复杂的项目

在下面较复杂的项目中，正在为几种包含不同类型的 MSP430 处理器以及不同的 I/O 端口和存储器配置的不同的目标硬件开发应用程序。因此，项目（project）包含调试目标（debug target），以及适用于不同的目标硬件组中每一种的发行目标（release target）。

为了方便起见，把所有目标公用的源文件收集在一起，放在被包含在每一个目标之中的组内。这些组的名字反映了源代码与之有关的应用程序内的区域（area）；例如 I/O routines（I/O 子程序），Small memory management（小存储器管理），等等。

取决于目标硬件的应用程序区域，例如存储器管理，被包含在许多单独的组之中，每个目标一个。最后，如前所述，为 Debug（调试）目标提供调试程序。



当用像此例这样的大项目进行工作时，嵌入式工作平台 Embedded Workbench 通过帮助用户记住项目的结构使用户开发时间为最短，通过汇编和编译最小的源文件组（它们是文件被修改之后完全更新目标代码所必需的）优化开发周期。

第 4 章　指　导

本章说明用户可以怎样使用嵌入式工作平台 Embedded Workbench 来开发简单的 C 程序、编译此程序并使用 C-SPY 调试器运行该程序。

在阅读本章之前，读者应当：

- 如“Quick start card（快速启动图）”或“Installation and documentation route map（安装和文档关系图）”一章所述，已安装了嵌入式工作平台 Embedded Workbench 软件。
- 熟悉 MSP430 处理器的体系结构和指令集。
- 熟悉 MSP430 C 编译器。详细资料可参见《MSP430 C Compiler Programming Guide（MSP430 C 编译器编程指南）》。

使用 C-SPY

本章假设读者正与嵌入式工作平台 Embedded Workbench 一起使用 C-SPY 调试器，并说明怎样用 C-SPY 运行用户开发的程序。如果用户的安装不包括 C-SPY，那么可以用嵌入式工作平台 Embedded Workbench 编辑器检查列表文件。

4.1　开始

在嵌入式工作平台中用户正在其上工作的文件被组织在项目之中。因此，使用嵌入式工作平台 Embedded Workbench 的第一步是创建新项目以规定用户正在其上工作的目标处理器，并包括项目所包含的文件列表。

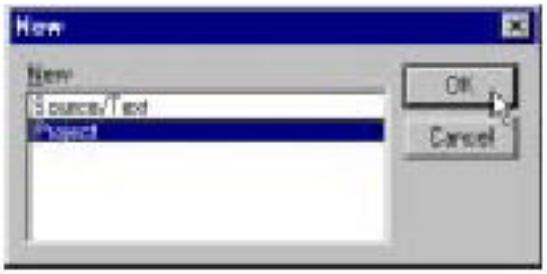
4.1.1 运行嵌入式工作平台 Embedded Workbench

要运行嵌入式工作平台 Embedded Workbench，双击 iarew.exe 图标。接着显示嵌入式工作平台 Embedded Workbench 窗口。

4.1.2 创建新项目

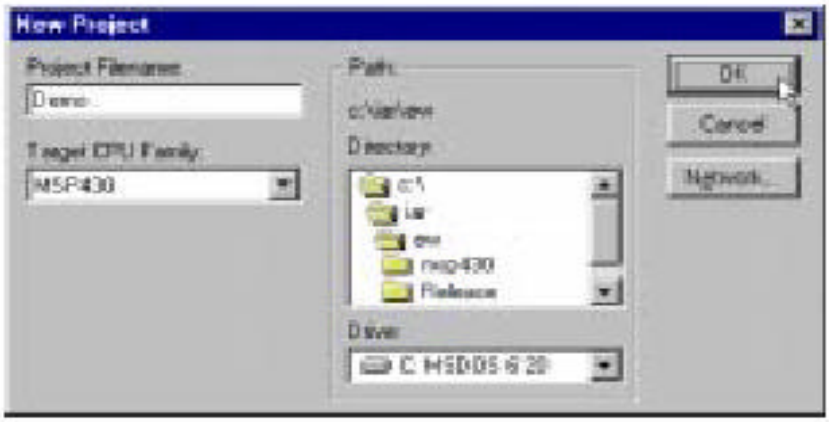
如下所述，创建本指导的项目。

从 File（文件）菜单中选定 New…（新）以显示下列对话框：



选择 Project（项目）并选定 OK 以显示 New Project（新项目）对话框。

在 Project Filename（项目文件名）框中键入 Demo，并把 Target CPU Family（目标 CPU 系列）设置为 MSP430：



然后选定 OK 以创建新项目。

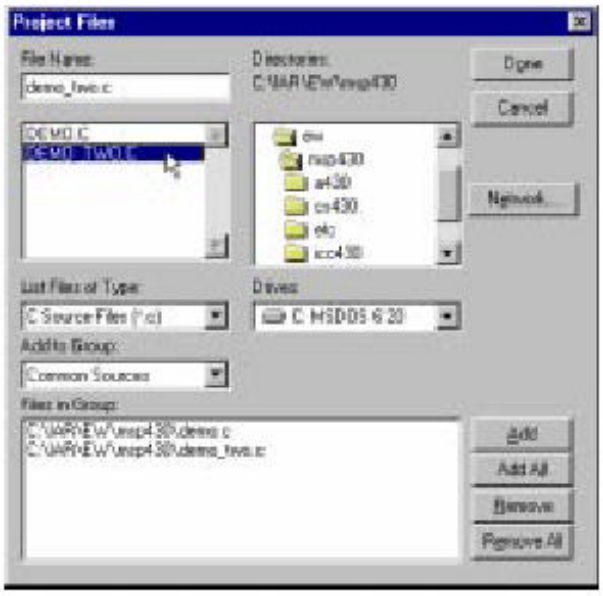
Project（项目）窗口将被显示。如果有必要，从 Targets（目标）下拉列表框中选择 Release（发行）以显示 Release（发行）目标：

下一步，如下所述，创建包含指导源文件的组。

从 Project（项目）菜单中选定 New Group…（新组…）并键入名字 Common Sources。缺省情况下两个目标均被选择，所以给将被加入到两个目标中：

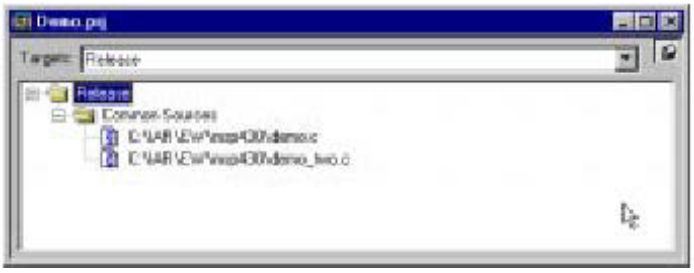
选定 OK 以创建组。它将显示在 Project（项目）窗口中。

从 Project（项目）菜单中选定 Files…（文件…）以显示 Project Files（项目文件）对话框。在对话框上半部的文件选样表中定位到文件 demo.c 和 demo-two.c，选定 Add（添加）以便把它们添加到 Common Sources（公共源文件）组中：



然后单击 Done（完成）以关闭 Project Files（项目文件）对话框。

单击+符号以便在 Project（项目）窗口的树状图中显示文件：



用户以后可以使用 project（项目）菜单中的 Files…（文件…）命令把文件加到项目中或删除文件。

4.1.3 编辑文件

要编辑项目中的一个文件，可以在 Project（项目）窗口中简单地双击其名字。例如，双击文件 demo.c。文件将显示在编辑器的窗口中：

注意：嵌入式工作平台 Embedded Workbench 编辑器提供了许多有用的特性以帮助用户正确地输入程序，并且在用户键入时提供立即的语法检查。

例如，程序的下列部分被识别：

条目	实出显示
关键字（keywords ）	黑色粗体
文本字符串（Text strings）	蓝色
预处理器伪指令（preprocessor directives）	绿色
数字常数（numeric constants）	红色/品红色/蓝色
条目	实出显示
注释（comments）	深蓝色斜体
其他程序结构	黑色

我们将使用编辑器把错误引入程序以便能看到工作平台 Workbench 所提供的错误处理特性。

把第 11 行末尾的 I++改为 j++，并通过在 File（文件）菜单中选定 save（保存）命令保存文件。

4.2 编译项目

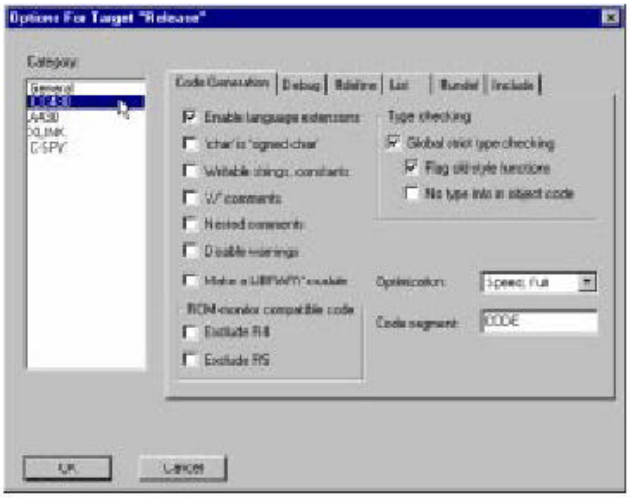
4.2.1 设置编译器选项

嵌入式工作平台 Embedded Workbench 允许用户为整个目标，文件组或单个源文件设置选项。

对于这个指导举例，因为不需要适用于目标中的组成文件的单独选项，所以我们为整个 release（发行）目标设置选项。

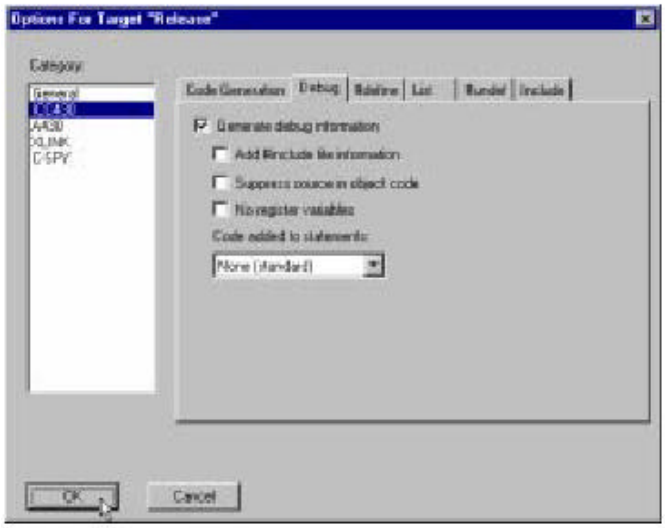
在 project（项目）窗口中选择 release（发行）文件夹图标，规定哪些选项将被设置，从 project（项目）菜单选定 option …（选项…）。

option （选项）对话框将被显示。然后在 category（类型）列表中选择 ICC430 以显示 C 编译器项页：



通过单击页顶部的标签（tab）用户可以显示任何页。

单击 debug（调试）以显示 C 编译器调试选项并选定复选框 Generate Debug Information （产生调试信息），以便为用 C-SPY 进行调试创建输出文件：



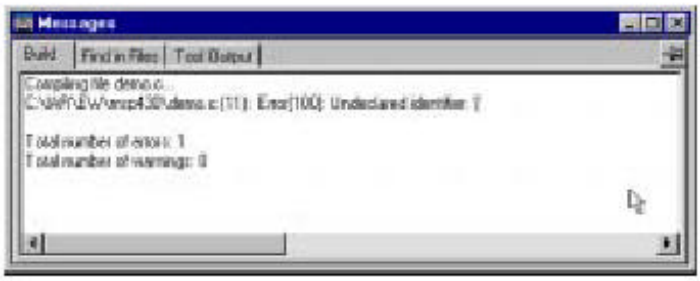
选定 OK 以保存用户规定的选项。

4.2.2 编译文件

为了编译用户正在编辑的源文件，从 project（项目）菜单选定 compile（编译），或者在工具栏上单击 Compile（编译）按钮。

用户也可以通过在 Project（项目）窗口中选择它并选定 Compile（编译）来编译源文件。

过程以及任何出错信息将显示在 Message（消息）窗口内：



这里有一个错误，它对应于我们插入的缺陷(bug)。

Messages（消息）窗口内双击错信息。光标将直接移至程序中适当的行，用户能简单地纠正错误。

例如，在这种情况下反 j++改为正确的版本 I++。

然后，如前所述重新编译。这次应当没有错误地进行编译，接着可通过从 file（文件）菜单中选定 close（关闭）来关闭 demo.c 源文件。

用同样的方式编译文件 demo-two.c。

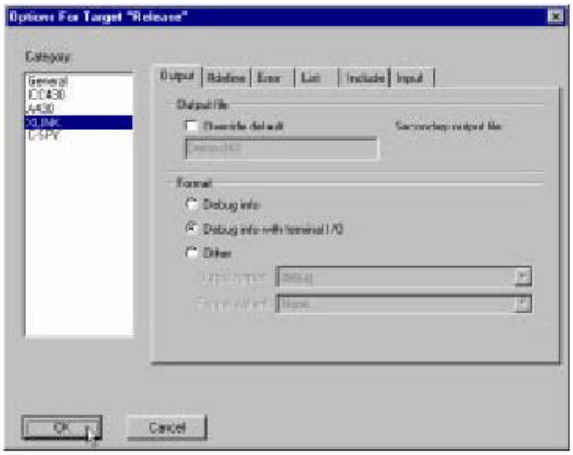
4.3 连接项目

在连接程序之前，用户需要为项目设置连接器选项。

4.3.1 设置连接器选项

在 Project（项目）窗口选择 Release（发行）文件夹。接着从 Project（项目）菜单选定 Options…（选项），并在 Category（类别）列表中选择 XLINK 以显示连接器选项页。

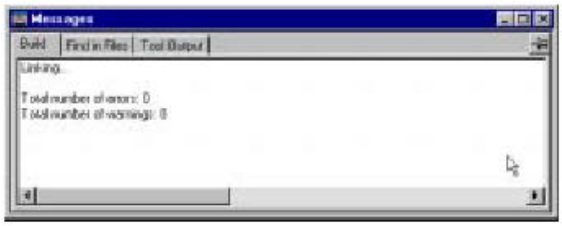
然后单击 Output（输出）以显示输出选项。核对 Debug Options（调试选项）选项被设置到 Debug info with terminal I/O（用终端 I/O 调试信息），以便为用 C-SPY 进行调试产生文件。



接着选定 OK 以关闭对话框并保存用户的设置。

4.3.2 连接文件

为了连接项目，从 Project（项目）菜单中选定 Link（连接）。接着文件将被连接，Message（消息）窗口将显示连接的过程:



假如没有错误，将产生输出 aout.d43，它可以和 C-SPY 模拟器（C-SPY Simulator）一起使用。

4.4 调试项目

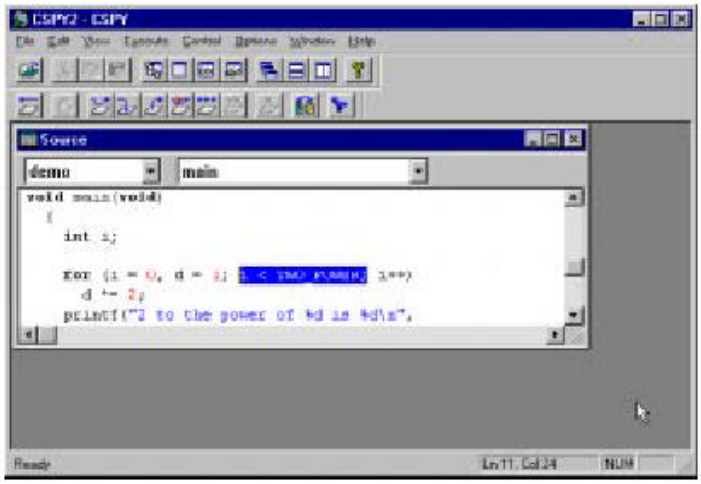
如果用户有 C-SPY 调试器，那么可用 C-SPY 来运行目标代码。

从 Project（菜单）中选定 Debugger（调试器），或者在工具栏上单击 C-SPY 按钮。

如果有必要，项目将和模拟器所使用时调试信息一起被重新连接，然后 C-SPY 自动被运行。

从 Action（动作）菜单中选定 step（单步）以启动源代码的执行。

源代码将显示在屏幕上，第一条可执行语句被突出显示。



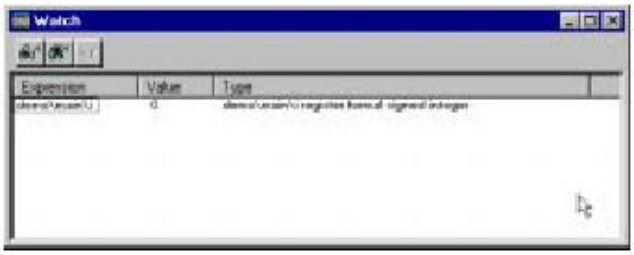
4.4.1 查看变量

跟踪用户可在其上设置察看点（watch point）的变量。

例如，为了在单步执行程序时察看变量 I 和 d 的值，首先通过从 Window（窗口）菜单中选定 Watch（查看）。

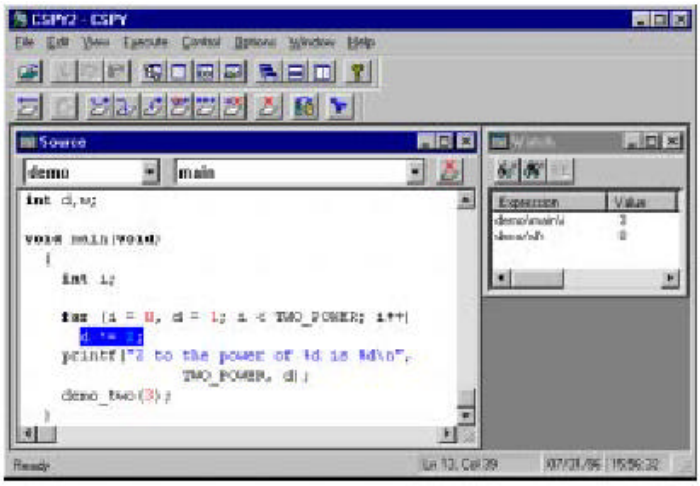
在 Watch（查看）窗口栏中单击 Watch（查看）按钮。

然后键入 I←（回车）以便把变量 I 添加到 Watch（查看）窗口中：



用同样的方法添加变量 d，并把 Watch（查看）窗口定位在屏幕上方便的位置上。

现在从 Execute（执行）菜单上选定 step（单步），或在工具栏上单击 step（单步）按钮。单步执行程序并在 Watch（查看）窗口中观察变量 I 和 d 的变化。



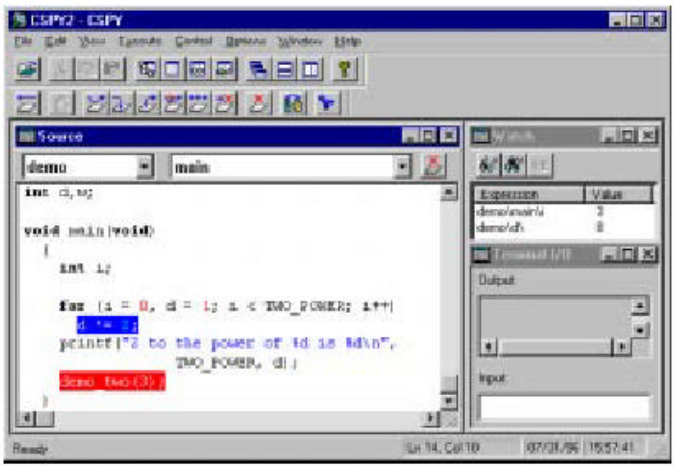
4.4.2 设置断点

用户可通过在规定的语句处设置断点而使程序执行到该语句。

首先，通过从 Window（窗口）菜单中选定 Terminal I/O（终端 I/O）打开 Terminals I/O（终端 I/O）窗口显示来自程序的输出。

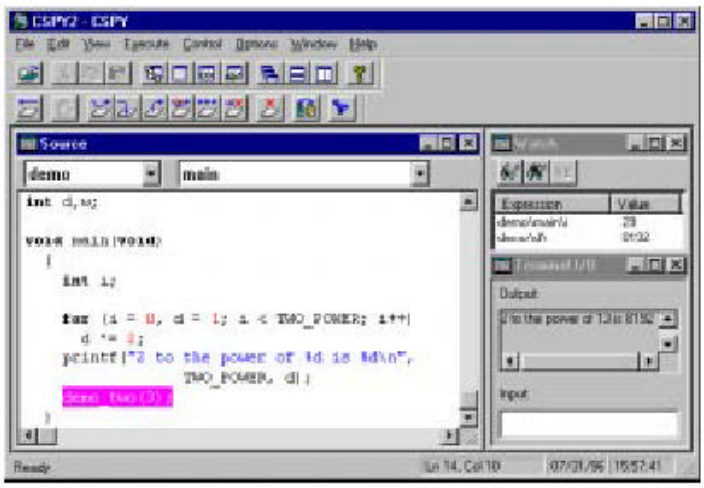
在 Source（源程序）窗口中，通过单击鼠标把光标定位到 demo –two（3）；并从 Control（控制）菜单中选定 Toggle Breakpoint（切换断点）。

语句将用红色突出显示以表示断点：



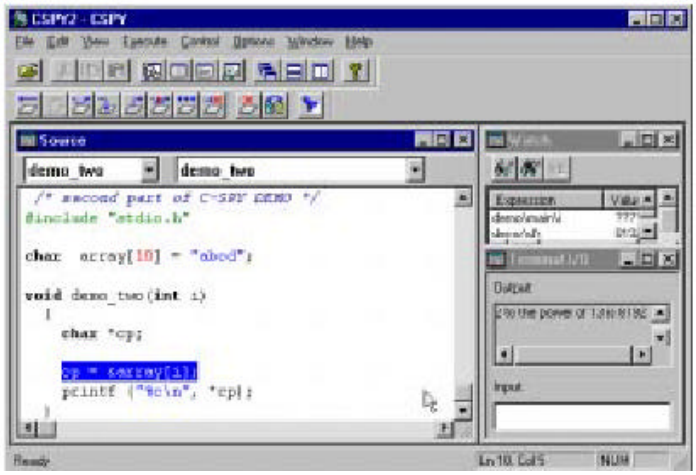
然后从 Execute（执行）菜单上选定 Go（到…去）或在工具栏中单击 Go（到…去）按钮，以便一直执行到断点。

程序的输出将显示在 Terminal I/O（终端 I/O）窗口中：



现在从 Execute（执行）菜单中选定 Step Into（步进到），或者在工具栏上单击 Step Into（步进到）按钮，以便执行到子程序 demo-two 之中。

Source（源程序）窗口接着将自动显示包含子程序 demo-two 的第二个源文件：



如果继续执行，将返回到文件 demo.c，然后从程序退出。

要从 C-SPY 退出并返回到嵌入式工作平台 Embedded Workbench，从 File（文件）菜单中选定 Exit（退出）。

4.5 使用 Make（生成）命令

代替单独编译和汇编项目中的文件并接着连接它们，用户可以使用 Make（生成）命令自动使项目得到更新。

嵌入式工作平台 Embedded Workbench 保存项目中所有文件以及它们相关的包含文件的列表。当用户运行 Make（生成）命令时，嵌入式工作平台 Embedded Workbench 只检查依赖性文件（dependent files），并在必需时重新编译或重新汇编以实现项目的更新。

注意：不需要把包含文件添加到项目中；当用户在#include 语句中引用它们时，它们将自动被添加到依赖性文件（dependent files）列表中。

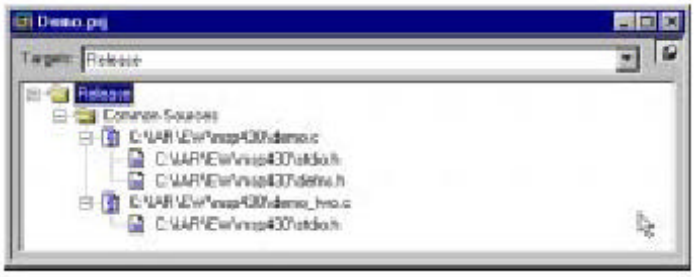
4.5.1 编辑包含文件

下面的例子说明 Make 命令怎样自动检测依赖性文件（dependent files）是否已被改变。

文件 demo.c 包含下列#include 语句：

#include “defns.h”

一旦用户编译了文件，Project（项目）窗口将显示它所引用的任何包含文件。单击靠近源文件的+符号以扩展树形显示并显示包含文件：



通过在 Project（项目）窗口中双击其名字打开文件 defns.h:

把 TWO-POWER 的定义改为 14 并保存文件。

4.5.2 生成项目

为了更新项目，从 Project（项目）菜单中选定 Make（生成），或在工具栏上单击 Make（生成）按钮。

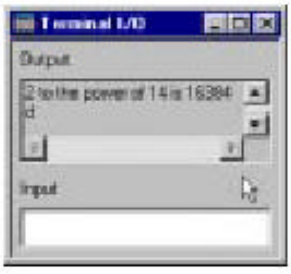
由于源文件 demo.c 所引用的包含文件已被修改，所以该源文件将被重新编译，接着整个项目被重新连接。

4.5.3 调试项目

通过从 Project（项目）菜单中选定 Debugger（调试器）以便使用 C-SPY 调试修改过的程序。

然后，通过从 Execute（执行）菜单中选定 Run（运行），或在工具栏中单击 Run（运行）按钮来运行程序。

Terminal I/O（终端 I/O）窗口将显示新的程序输出：



如前所述，从 File（文件）菜单中选定 Exit（退出）以便从 C-SPY 退出。

4.6 下一步做什么

现在已完成了这个简短的有关嵌入式工作平台 Embedded Workbench 的指导性例子。

有关使用嵌入式工作平台 Embedded Workbench 和嵌入式工作平台编辑器的更为详细的资料，请参见指南下面两章：“参考”和“嵌入式工作平台编辑器”。

有关使用嵌入式工作平台 Embedded Workbench 工具的更多的资料，请参见《MSP430 C

Compiler Programming Guide(MSP430 C 编译器编程指南)》和《MSP430 Assembler, Linker and Librarian Programming Guide(MSP430 汇编器、边接器以及库管理器编程指南)》。

第5章 参 考

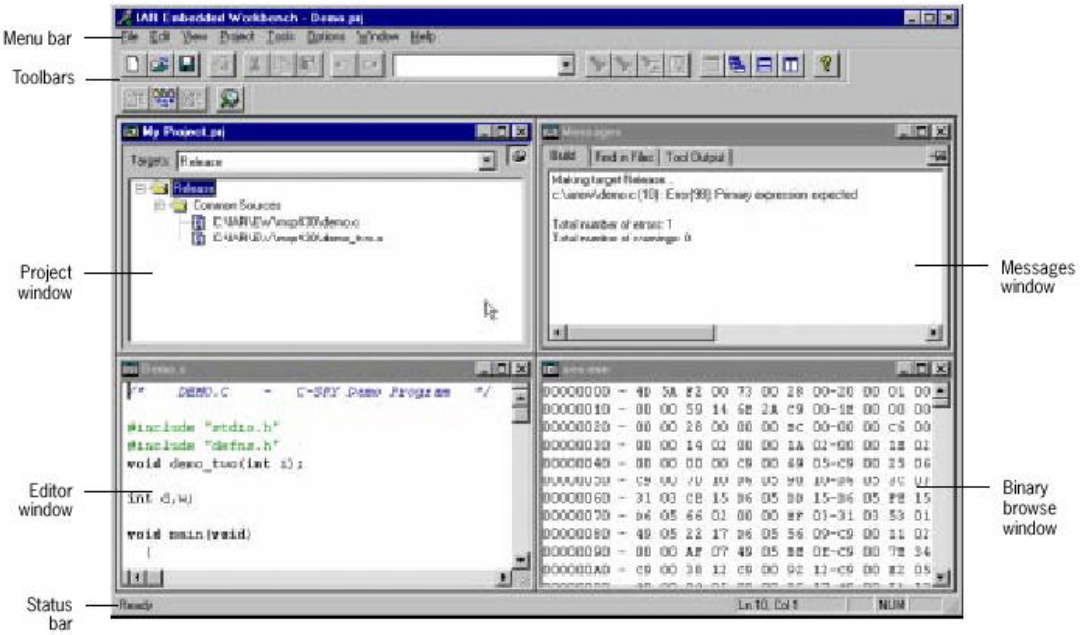
本章提供有关 MSP430 嵌入式工作平台 Embedded Workbench 的完整参考资料。

首先给出有关工作平台 Workbench 窗口组成部分，以及它所包含的每一种不同类型窗口的资料。

然后详细叙述菜单以及每一个菜单上的命令。

5.1 嵌入式工作平台 Embedded Workbench 窗口

下图显示了嵌入式工作平台 Embedded Workbench 窗口的不同组成部分。



这些组成部分将在以下各节中详细说明。

5.1.1 菜单栏（Menu Bar）

从菜单栏可访问嵌入式工作平台 Embedded Workbench 的菜单。

菜单	说明
File（文件）	File（文件）菜单提供打开项目和源文件、保存和打印、以及从嵌入式工作平台 Embedded Workbench 退出的命令。
Edit（编辑）	Edit（编辑）菜单提供在编辑窗口中编辑和搜索的命令。
View（视图）	View（视图）菜单上的命令允许用户改变显示在工作平台 Embedded Workbench 窗口中的信息。
Project（项目）	Project（项目）菜单提供把文件添加到项目，创建组，以及在当前项目上运行 IAR 工具的命令。
Tools（工具）	Tools（工具）菜单是用户可配置的菜单，用户可以把与工作平台 Workbench 一起使用的工具加到此菜单中。
Options（选项）	Options（选项）菜单允许用户定制嵌入式工作平台 Embedded Workbench 以符合用户自己的需求。
Window（窗口）	Window（窗口）菜单上的命令允许用户管理嵌入式工作平台 Embedded Workbench 的窗口并改变它们在屏幕上的排列。

Help（帮助）	Help（帮助）菜单上的命令提供有关嵌入式工作平台 Embedded Workbench 的帮助。

在下面几页中将详细叙述菜单。

5.1.2 工具栏（Toolbars）

嵌入式工作平台 Embedded Workbench 窗口包括两种工具栏:

- 编辑栏（edit bar）
- 项目栏（project bar）

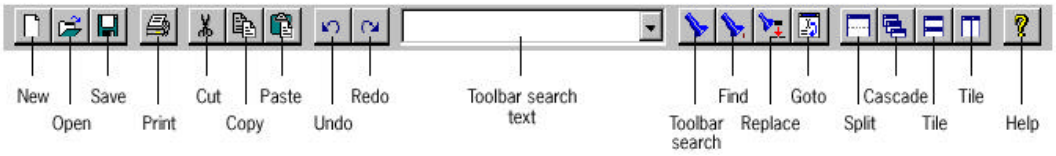
编辑栏提供嵌入式工作平台 Embedded Workbench 菜单上最有用命令的按钮，以及键入字符串以进行工具栏搜索的文本框。

项目栏提供在 Project（项目）菜单上建立和调试选项的按钮。

用户可以通过用鼠标按钮（mouse button）指向任何工具栏按钮来显示该按钮的描述。当命令不能使用时，相应的工具栏按钮变为灰色，用户将不能选择它。

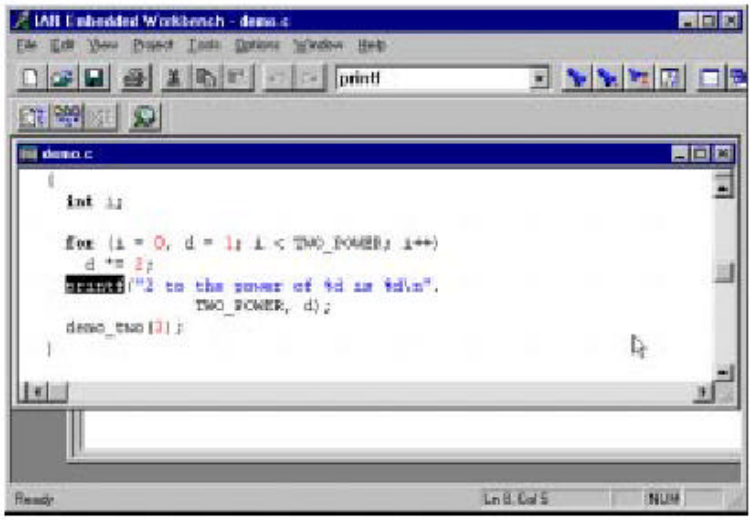
5.1.2.1编辑栏

下图表示与菜单命令相对应的每一个编辑栏按钮。



工具栏搜索（Toolbar search）

要搜索最近的编辑器窗口中的文本，可以在 Toolbar search（工具栏搜索）文本框中键入文本，并按←（回车键）或单击 Toolbar search（工具栏搜索）按钮。

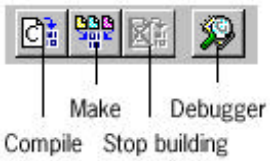


另一方面，用户也可以下拉列表框中选择原先已搜索过的字符串。

用户可以用 View（视图）菜单上的 Edit Bar（编辑栏）命令来选取定是否显示编辑栏。

5.1.2.2项目栏（Project bar）

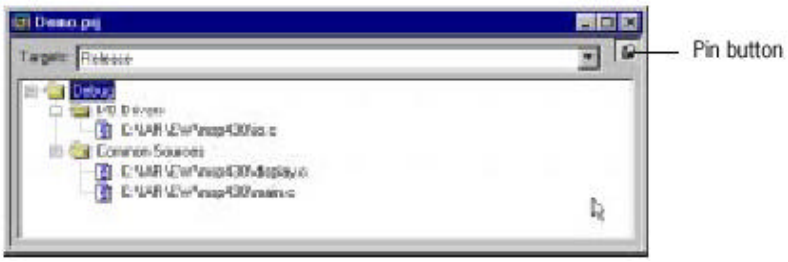
下图表示与菜单命令相对应的每一个项目按钮。



用户可以使用 View（视图）菜单上的 Project bar（项目栏）命令来选定是否显示项目栏。

5.1.3 项目窗口（Project Window ）

Project（项目）窗口显示当前项目的名字，并显示包含项目中的组和文件的树状关系。



在 Project（项目）窗口中按鼠标的右键将会显示弹出菜单，它使用户能方便地访问几个有用的命令。

5.1.3.1 Pin（钉住）按钮

Project（项目）窗口右上角的 Pin（钉住）按钮允许用户把窗口“钉”在桌面上，以便它能够不受 Window （窗口）菜单中 Tile（平铺）或 Cascade（层叠）命令的影响。

5.1.3.2 目标（Targets）

树状结构中顶部节点表示当前目标。用户可通过从 Project（项目）窗口顶部 Targets（目标）下拉列表框中选定不同的目标来改变目标。每一个目标对应于用户想要编译或汇编的用户项目的不同版本。

例如，用户可能具有名为 Debug（调试）的目标，它包含调试代码，以及名为 Release（发行）的目标，它省略了调试代码。

用户可通过双击目标图标来扩展树状结构，或通过单击+符号来显示包含在此目标中的组。

5.1.3.3 组（Groups）

组用于把相关的源文件收集在一起，每一个组可以被包含在一个或多个目标中，源文件可以出现在一个或多个组内。

5.1.3.4 源文件

用户可以通过双击组图标来扩展每一个组，或通过单击+符号来显示它所包含的源文件列表。

一旦成功地建立了项目，将在包含它们的源文件下面的结构中显示任何包含文件。注意：因为预处理器或目录选项可能影响哪一个包含文件与特定的源文件有关，所以与特定的源文件相关的包含文件可能取决于源文件出现在哪一个目标之中。

5.1.3.5 编辑文件

要编辑源或包含文件，可双击 Project（项目）窗口树状显示中该文件的图标。

5.1.3.6 在组之间移动源文件

通过在 Project（项目）树状显示中的组图标之间拖动源文件图标，可以在两个组之间移动源文件。

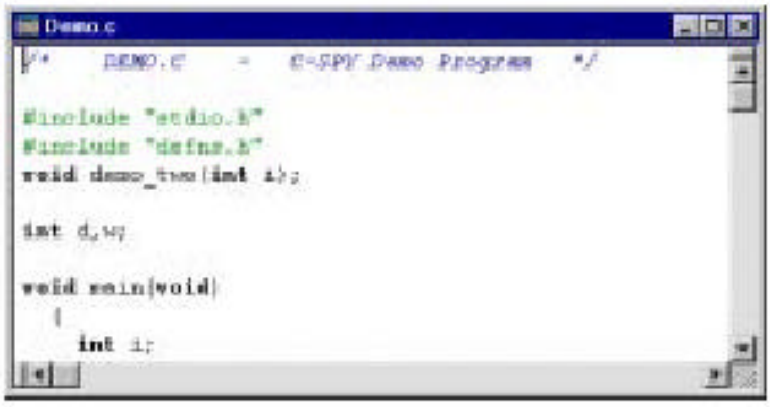
5.1.3.7 从项目中删除项

要从项目中删除一项（item），可在其上单击以选择它，然后接[Delete]。

为了从项目中删除文件，用户也可以使用 Project Files（项目文件）对话框，从 Project（项目）菜单中选定 Files…（文件…）可显示该对话框。

5.1.4 编辑器窗口

源文件显示在编辑器窗口中。编辑器自动识别 C 程序的语法，并用不同的文本样式显示 C 程序的不同组成部分。



下表显示了用于 C 程序每一个组成部分的缺省样式：

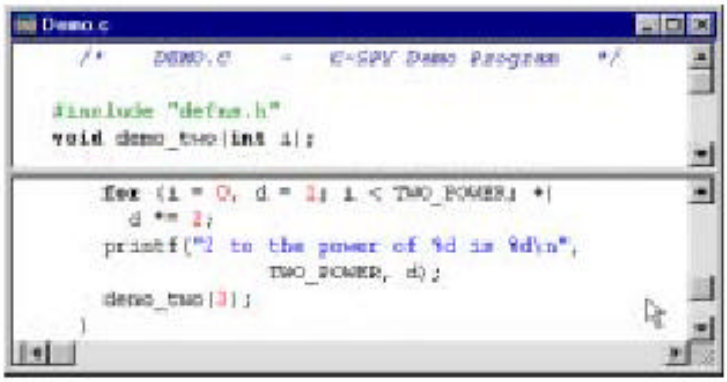
条目	样式
缺省（Default）	黑色普通体
C 关键字（C Keyword）	黑色粗体
字符串（Strings）	蓝色
预处理器（Preprocessor）	绿色
整型数（十进制）（Integer(dec)）	红色
整型数（八进制）（Integer(oct)）	品红
整型数（十六进制）（Integer(hex)）	品红
实型数（Real）	蓝色
C++注释（C++ comment）	深蓝色斜体
注释（Comment）	深蓝色斜体

要改变这此样式，可从 Options（选项）菜单中选定 Settings…（设置），然后在 Settings（设置）对话框中选择 Colors and Fonts（颜色和字体）页。

把编辑器窗口分为长方格

用户可以把编辑器窗口水平地或垂直地分为多个长方形格子，以便能同时看到同一源文件的两个不同的部分，或者在两个不同的部分之间剪切（Cut）和粘贴（Paste）。

要分割窗口，可把合适的分割线控件（Splitter Control）拖到窗口的中央：



要恢复到单个长方形窗口，双击合适的分割线控件，或把它拖回到滚动条的末端。

用户也可以使用 Window（窗口）菜单上的 Split（分割）命令把窗口分为长方形的格子。

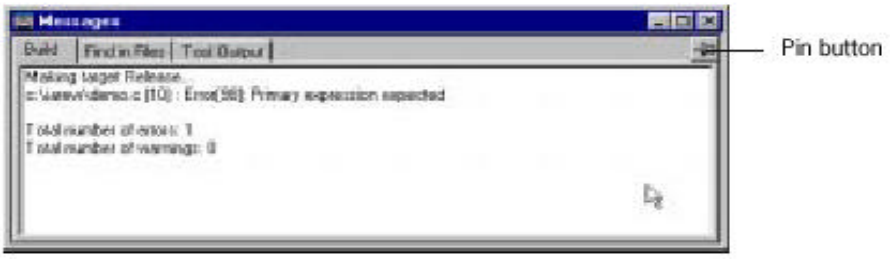
5.1.5 状态栏（Status Bar）

显示嵌入式工作平台 Embedded Workbench 的状态，以及修改键（Modifier keys）的状态。

用户可以使用 View（视图）菜单上的 Status Bar（状态栏）命令来选定是否显示状态栏。

5.1.5 消息窗口（Messages Window）

Messages（消息）窗口显示来自不同的嵌入式工作平台 Embedded Workbench 命令的输出。窗口被分为多个页，用户可以通过单击相应的标签（tab）选择合适的页。



在 Messages（消息）窗口中按鼠标右键将显示弹出菜单，它允许用户把窗口的内容保存为文本文件。

5.1.6.1 Pin（钉住）按钮

Project（项目）窗口右上角的 Pin（钉住）按钮允许用户反窗口“钉”在桌面上，以使它能不受 Window（窗口）菜单中 Tile（平铺）或 Cascade（层叠）命令的影响。

5.1.6.2 Build（建立）

显示建立项目是产生的消息。在 Build（建立）长方格中双击消息将打开合适的文件以供编辑，同时光标将位于正确的位置上。

5.1.6.3 Find Files（在文件中寻找）

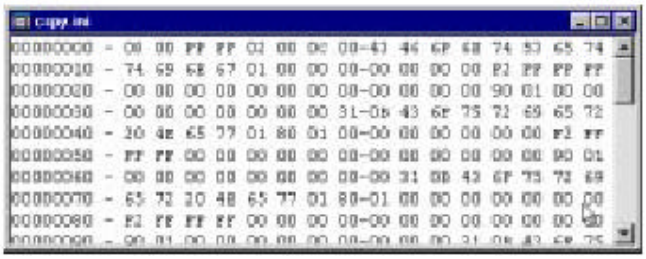
显示 Edit（编辑）菜单中 Find Files（在文件中寻找）命令的输出。双击长方格中的条目将打开合适的文件，同时光标定位在正确的位置上。

5.1.6.4 Tool output（工具输出）

显示 Tools（工具）菜单中用户定义的工具产生的任何消息输出。在 Windows3.1X 下不能使用此特性。

5.1.7 二进制浏览窗口（Binary Browse Window）

Binary Browse（二进制浏览）窗口以十六进制数据的形式显示二进制文件的内容，其 ASCII 等效位于每一行的右边。要显示二进制数据，可以从 Tools（工具）菜单中选定 Browse Binary Data…（浏览二进制数据…）。



5.2 文件菜单（File Menu）

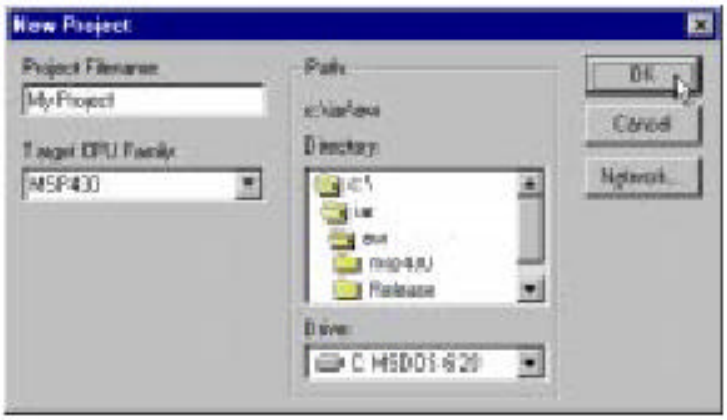
File（文件）菜单提供打开项目和源文件、保存和打印以及从嵌入式工作平台 Embedded Workbench 退出的命令。

菜单也包括最近被打开文件的顺序列表，通过从菜单中选择其名字可打开一个文件。

5.2.1 New…（新…）

显示下列对话框，以便用户规定他想要创建一个新项目还是创建一个新文本文件。

选定 Source/Text（源/文本）将打开一个新编辑器窗口以便用户键入文本文件。
选定 Project（项目）将显示下列对话框以便用户规定项目的名字以及目标 CPU 系列：



然后项目将显示在新 Project（项目）窗口中。
缺省情况下，创建的新项目有两个目标，Release（发行）和 Debug（调试）。

5.2.2 Open…（打开…）

显示标准的 Open（打开）对话框，允许用户选择打开文本或项目文件。打开新项目文本将自动关闭并保存任何当前打开的项目。

5.2.3 Close（关闭）

关闭活动窗口

如果从上一次被保存以来文本文档已发生了变化，那么将提醒用户，以便给出一个在关闭之前保存文件的机会。项目自动被保存。

5.2.4 Save（保存）

保存当前文本或项目文档。

5.2.5 Save as…（另存为…）

显示标准的 Save as（另存为）对话框，允许用户用不同的名字保存活动的文档。

5.2.6 Print…（打印…）

显示标准的 Print（打印）对话框，以便用户打印文本文档。

5.2.7 Print Setup…（打印设置…）

显示标准的 Print Setup（打印设置）对话框，允许用户在打印之前设置打印机。

5.2.8 Exit （退出）

从嵌入式工作平台 Embedded Workbench 退出。在关闭之前将询问用户是否保存文本窗口的任何改变。项目的变化将自动被保存。

5.3 编辑菜单（Edit Menu）

Edit（编辑）菜单提供在编辑器窗口中编辑和搜索的命令。

5.3.1 Undo（撤消）

撤消最近一次对当前编辑器窗口所作的 Undo（撤消）操作。

5.3.2 Redo（恢复）

恢复最近一次在当前编辑器窗口所作的 Undo（撤消）操作。

用户可以独立地在每一个编辑器窗口中撤消和恢复数目不受限制的编辑操作。

5.3.3 Cut, Copy , Paste（剪切，复制，粘贴）

在编辑器窗口和对话框中提供标准的编辑文本的 Windows 功能。

5.3.4 Find…（寻找…）

显示下列对话框，允许用户在当前编辑器窗口中搜索文本：

在 Find What（寻找什么）文本框中键入要搜索的文本。

选择 Match Whole Only（仅整个字匹配）将寻找规定的文本，只能当它作为独立的字出现时才算找到。否则规定 int 还将找到 Print, Sprintf 等等。

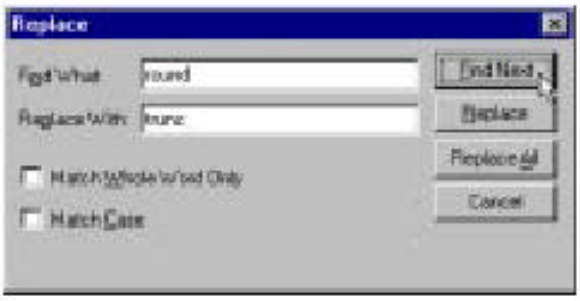
选择 Match Case（大小写匹配）将寻找规定的文本的大小写严格匹配的文本。否则规定 int 还将找到 INT 和 Int。

选择 Up（上）或 Down（下）将规定搜索的方向。

选择 Find Next（寻找下一个）将寻找用户规定文本的下一次出现。

5.3.5 Replace…（替换…）

允许用户搜索规定的字符串并在它每一次出现时用另一个字符串替换之。



在 Replace With（用…替换）框中键入每次发现时要替换的文本。其他选项与 Find…（寻找…）相同。

选定 Find Nest（寻找下一个）以寻找规定文本的下一次出现，Replace（替换）以便用规定的文本替换它。另外，Replace All（替换全部）可替换当前编辑器窗口中所有找到的文本。

5.3.6 Fine in Files…（在文件中寻找…）

允许用户在多个文本文件中搜索规定的字符串。下面的对话框使用户能规定搜索的标准。

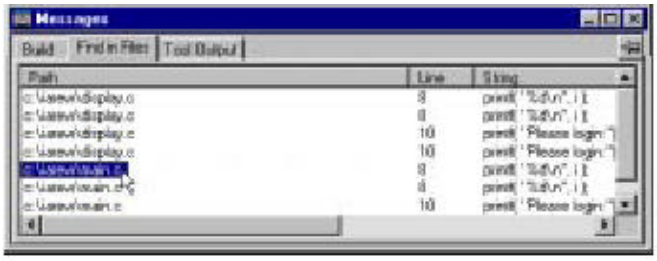
在 Search String（搜索字符串）文本框中规定想要搜索的字符串，或者从下拉列表框中选择以前已搜索过的字符串。

选择 Match Whole Word（整个字匹配）或 Match Case（大小写匹配）把搜索分别限制为整个字匹配或大小写严格匹配。

在 File Name（文件名）列表中选择每一个想要搜索的文件，并选定 Add（添加）以便把它们添加到 Selected Files（已选择的文件）列表中。

通过选定 Add All（添加全部），可以把全部文件添加到 File Name（文件名）列表中，或者使用[shift]和 [ctrl]键选择多个文件并选定 Add（添加）添加所选择的文件。与此相同，用户可以使用 Remove（删除）和 Remove All（删除全部）按钮从 Selected Files（已选择的文件）列表中删除文件。

当用户已选择了他想要搜索的文件时，选定 Find（寻找）以进行搜索。所有匹配的情况将在 Messages（消息）窗口中列出：



接着，用户可以非常简单地通过在 Messages（消息）窗口中双击每一个找到的文本来编辑它。这将在编辑窗口中打开相应的文件，同时光标将定位在包含规定文本的哪一行的开始处：

5.3.7 Match Brackets（匹配括号）

如果光标定位在紧靠括号处，此命令将把光标移到匹配的括号，如果没有匹配的括号，那么将发出鸣叫声。

5.4 视图菜单（View Menu）

View（视图）菜单上的命令允许用户改变显示在嵌入式工作平台 Embedded Workbenc 窗口中显示的信息。

5.4.1 Edit Bar（编辑栏）

使编辑栏打开或关闭。

5.4.2 Project Bar（项目栏）

使项目栏打开或关闭。

5.4.3 Status Bar（状态栏）

使状态栏打开或关闭。

5.4.4 Goto Line…（到行…）

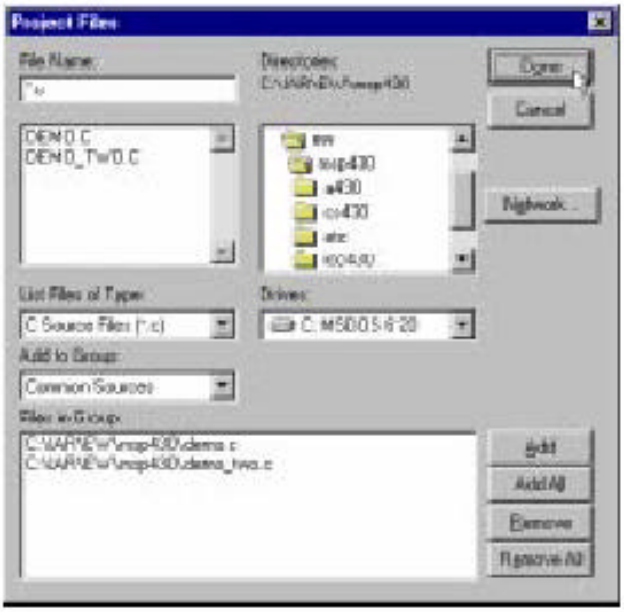
显示下列对话框，允许用户把光标移到当前编辑器窗口的规定行和列：

5.5 项目菜单（Project Menu）

Project（项目）菜单提供把文件添加到项目、创建组以及在当前项目上运行 IAR 工具的命令。

5.5.1 Files…（文件…）

显示下列对话框，允许用户编辑当前项目的内容:



Add to Group（添加到组）下拉列表框显示包含在当前目标中所有的组。选择想要编辑的组，当前在该组中的文件将显示在对话框底部 Files in Group（组中的文件）列表中。

Project Files（项目文件）对话框的上半部是标准的文件对话框，允许用户定位并选择想要加到每一个特定的组中的文件。

5.5.1.1把文件添加到组中

为了把文件添加到当前显示的组中，在对话框的上半部使用标准的文件控制选择文件并选定 Add（添加）按钮，或者选定 Add All（添加全部）以便添加 File Name（文件名）列表上所有的文件。

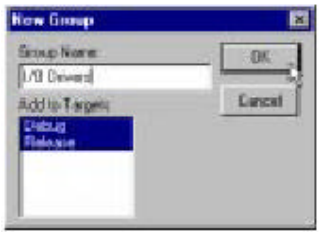
5.5.1.2 从组中删除文件

要从当前显示的组中删除文件，在 Files in group（组中文件）列表中选择它们并选定 Remove（删除），或选定 Remove All（删除全部）以便从组中删除所有的文件。

用户可以使用 Project Files（项目文件）对话框以便对几个组进行修改。然后选定 Done（完成）将把所有的修改应用于项目。另一方面，若选定 Cancel（取消）将言放弃所有的修改并保持项目不受影响。

5.5.2 New Group…（新组…）

显示下列对话框以允许用户创建新的组:

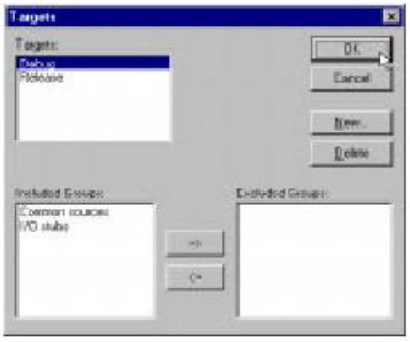


在 Group Name（组名）文本框中规定想要创建的组的名字。

在 Add to Targets（添加到目标）列表中选择想要把新组添加到其中的目标。缺省情况下，组将添加到所有的目标。

5.5.3 Targets…（目标…）

显示下列对话框，以便用户创建新的项目，并显示和修改包含在每一个目标中的组：



单击 New…（新…）以创建新的目标，并键入新目标的名字。

在 Targets（目标）列表中选择要删除的目标，并单击 Delete（删除），便可删除目标。

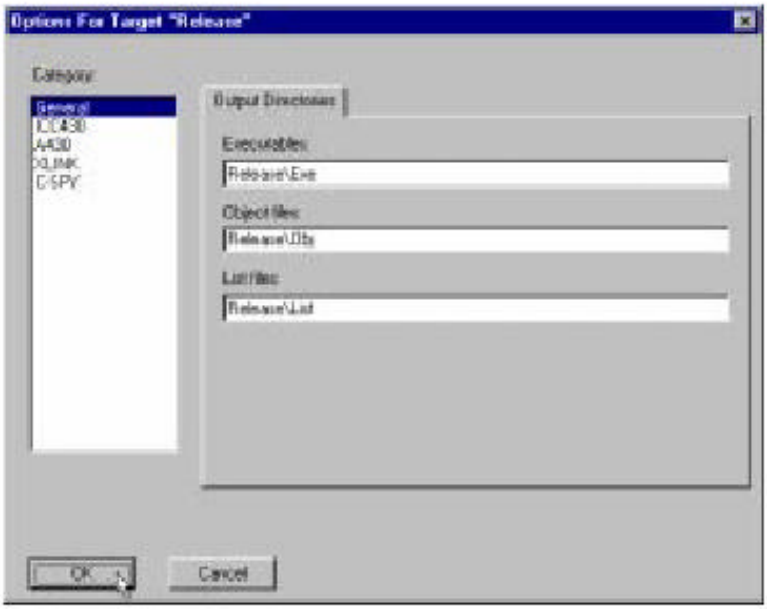
要观察包含在目标中的组，可以在 Targets（目标）列表中选取择它。

组被显示在 Included Groups（包含的组）列表中，用户可以用箭头按钮添加或删除组→和←。

5.5.4 Options…（选项…）

显示 Options(选项)对话框，允许用户设置目录和有关 Project（项目）窗口中当前选择条目（Currentlg-selected item）的编译器选项。

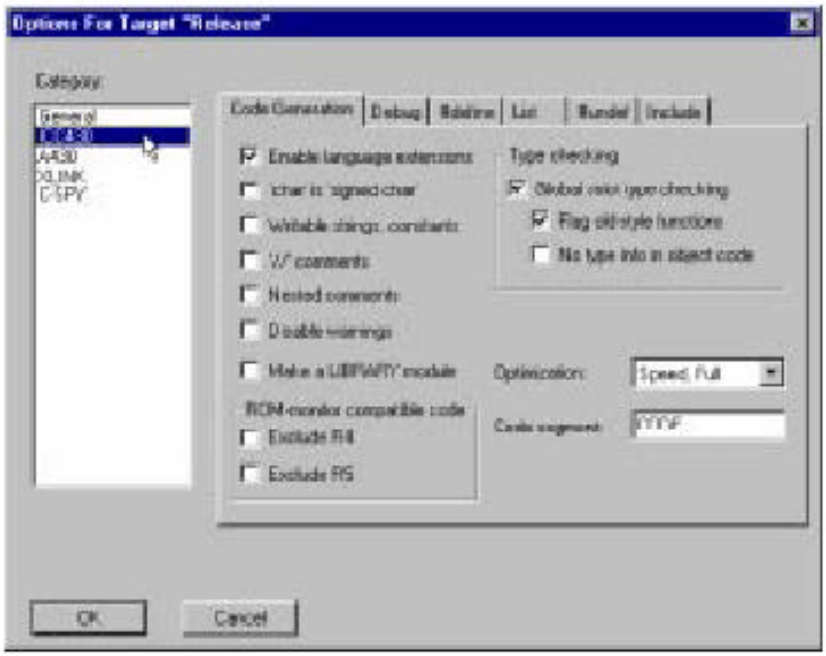
用户可以设置关于整个目标、关于文件组或关于单个文件的选项。



Category（类别）列表允许用户选择他想要编辑的选项组。Category（类别）表中可供使用的选项将取决于安装在嵌入式工作平台 Embedded Workbench 中的工具，通常将包括下列选项：

类别	说明
General	一般选项
ICC430	MSP430 C 编译器选项
A430	MSP430 汇编器选项
XLINL	XLINL 连接器选项
C-SPY（可选）	C-SPY 选项

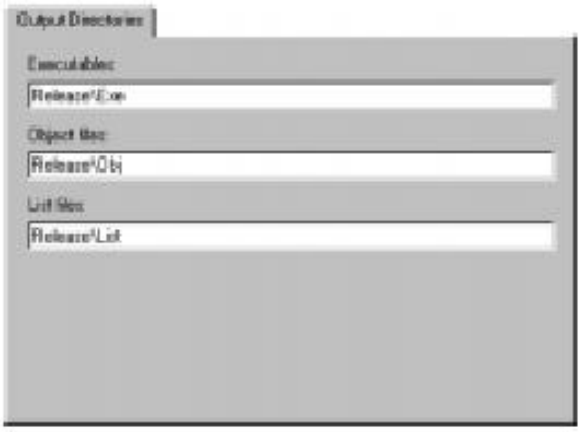
然后，选择类别（category）显示嵌入式工作平台 Embedded Workbench 的该组件的一个或多个选项页：



单击合适的标签以显示选项的相应页。

5.5.4.1 一般选项（General options）

General（一般）类别提供 Output Directories（输出目录）页，允许用户指定可执行、目标以及列表文件的路径：



5.5.4.2 其它选项

在其他类别中可供使用的选项的详细资料请参见《MSP430 C Compiler Programming Guide（MSP430 C 编译器编程程指南）》，《MSP430 Assembler, Linker, and Librarian Programming Guide（MSP430 汇编器、连接器、以及库管理器编程指南）》或《MSP430 C-SPY User Guide, Windows Workbench Version（MSP430 C-SPY 用户指南，视窗工作平台版本）》。

5.5.5 Compile（编译）

编译或汇编合适的当前活动的文件。

用户可通过在 Project（项目）窗口中选择文件的图标并选定 Compile（编译）来编译一个文件。另一方面，如果文件是当前目标的成员，那么用户也可以在当前活动的编辑器窗口中编译该文件。

5.5.6 Make（生成）

通过仅仅编译、汇编和连接那些必要的文件来更新当前目标。

5.5.7 Link（连接）

明确地重新连接当前目录。

5.5.8 Build All（建立全部）

把当前项目中的所有组成部分标记为已改变的，然后运行 Make（生成）命令来重建和重新连接当前目标中所有的文件。

5.5.9 Stop Build（停止建立）

停止当前建立的操作。

5.5.10 Librarian（库管理器）

运行 XLIB 库管理器，以便用户对库文件中的库模块进行操作。

5.5.11 Debugger（调试器）

运行可选的 Windows 版本的 C-SPY 调试器，以便用户可以调试项目目标文件。

用户可以在目标的 Debug（调试）选项中规定要运行的 C-SPY 的版本。如果有必要可以在运行 C-SPY 之前执行 Make （生成）命令以确保项目被更新。

5.6 工具菜单（Tools Menu）

Tools（工具）菜单是用户定义的菜单，用户可以把与嵌入式工作平台 Embedded Workbench 一起使用的工具添加到菜单中。

5.6.1 Add Tool…（添加工具…）

A dd Tool…（添加工具…）显示下列对话框，以便用户规定添加到菜单中的用户定义工具：

在 Menu Text（菜单文本）框中规定用于菜单项的文本，在 Command（命令）文本框中规定选择该项时运行的命令。另外，选定 Browse（浏览）可显示标准的文件对话框，以便用户定位于磁盘上的可执行文件并把其路径添加到 Command（命令）文本框中。

在 Argument （参数）文本框中规定命令的参数，或选择 Prompt for Command Line（命令行提示）以便当从 Tools（工具）菜单中选择命令时显示命令行的提示。

Initial Directory（初始目标）文本框允许用户规定工具的初始工作目录。

选择 Redirect to Output Window（重新导入输出窗口）将在工具窗口中显示任何来自工具的控制台输出（Console output）。注意，在 Windows3.1 之下不能使用此选项。

当用户规定了他想要添加的命令时，选定 Add（添加）以便把它添加到 Menu Content（菜单内容）列表中。通过在该列表中选择命令并选定 Remove，可以从 Tools（工具）菜单中删除该命令。

要确认对 Tools（工具）菜单所作的修改并关闭对话框，选定 OK。

用户所规定的菜单项将显示在 Tools （工具）菜单中：

规定 MS-DOS 命令或批文件

MS-DOS 命令或批文件需要从 Command Shell（命令外壳）运行，所以若要把它们添加到 Tools（工具）菜单中，用户需要在 Command（命令）文本框中规定合适的 Command Shell（命令外壳），在 Argument（参数）文本框中规定 MS-DOS 命令或批文件名。

Command Shell（命令外壳）规定如下：

系　统	Command Shell
Windows95 或 Windows3.11	Command.com
WindowsNT	Cmd.exe（推荐）或 Command.com

Argument（参数）文本应当规定为：

/C name

其中 name 是用户想要运行的 MS-DOS 命令或批文件的名字。

/C 选项在执行之后终止 shell（外壳），允许嵌入式工作平台 Embedded Workbench 检测工具在何时完成。

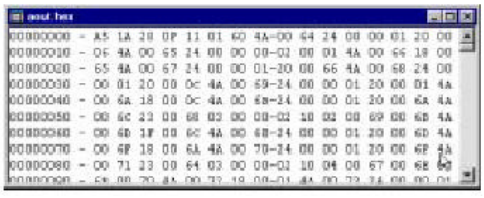
注意：运行需要用户输入的 MS-DOS 命令或批文件将导致嵌入式工作平台 Embedded Workbench 挂起。

例如：要把 Backup（备份）命令添加到 Tools（工具）菜单以便把整个项目目录的副本制作到网络驱动器中，用户可以在 Command （命令）框中规定 Command，在 Argument（参数）框中规定：

/c copy c:\project*.*F:

5.6.2 Browse Binary Data…（浏览二进制数据…）

以二进制和 ASCII 形式列出文件以供浏览和调试。此命令将显示标准的文件对话框以便用户选择文件，然后在 Browse（浏览）窗口中显示它：



注意：用户不能编辑 Browse（浏览）窗口的内容。

5.6.3 Record Macro（记录宏）

允许用户把键入的序列当作 Windows 宏进行记录。

5.6.4 Stop Record Macro（停止记录宏）

停止宏的记录。

5.6.5 Play Macro（播放宏）

重新播放已记录的宏。

5.7 选项菜单（Options Menu）

Options（选项）菜单的 Settings…（设置）命令允许用户定制嵌入式工作平台 Embedded Workbench 以符合自己的需求。

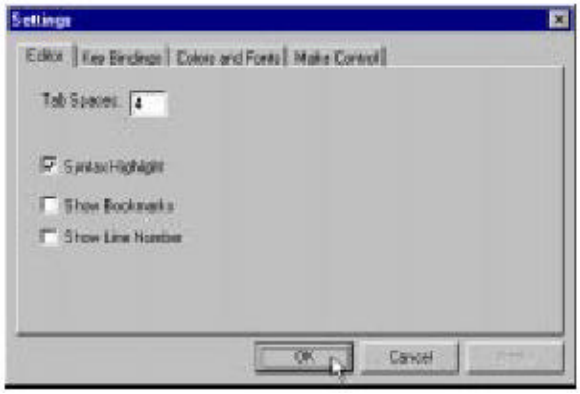
5.7.1 Settings…（设置…）

显示 Settings（设置）对话框以便用户定制嵌入式工作平台 Embedded Workbench。

用户可通过单击 Editor（编辑器）、Key Bindings（快捷键组合）、Colors and Fonts（颜色和字体）或 Make Control（生成控制）标签来选择想要定制的特性。

5.7.1.1 Editor（编辑器）

允许用户改变编辑器选项:

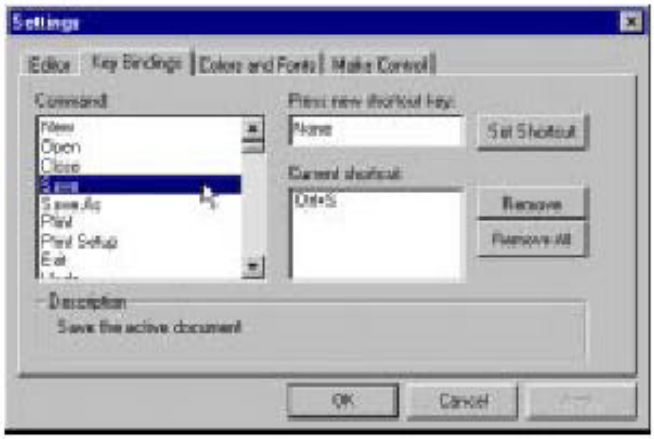


Editor Settings（编辑器设置）面板提供下列选项:

选项	说明
Tab Spaces（制表符空格数）	对应于每一制表符的空格字符数
Syntax Highlight（语法实出显示）	用不同的文本样式显示 C 程序的语法
Show Bookmarks（显示书签）	在编辑器窗口的左边沿显示书签
Show Line Number（显示行号）	在编辑器窗口中显示行号

5.7.1.2 Key Bindings（快捷键组合）

显示用于每一个菜单选项的快捷键，并允许用户改变它们:



在 Command（命令）列表中选择用户想要编辑的命令，任何当前定义的快捷键显示在 Current Shortcut（当前快捷键）列表中。

要把快捷键添加到命令中，在 Press new shortcut key（按新的快捷键）框中单击并键入想要使用的键组合。然后单击 Set Shortcut（设置快捷键）以便把它添加到 Current Shortcut（当前快捷键）列表中。如果某快捷键已被另一个命令所使用，那么将不允许用户添加快捷键。

如果用户把多于一个的快捷键用于特定的命令，那么它们之中只有一个显示在菜单上，但所有的快捷键均有效。

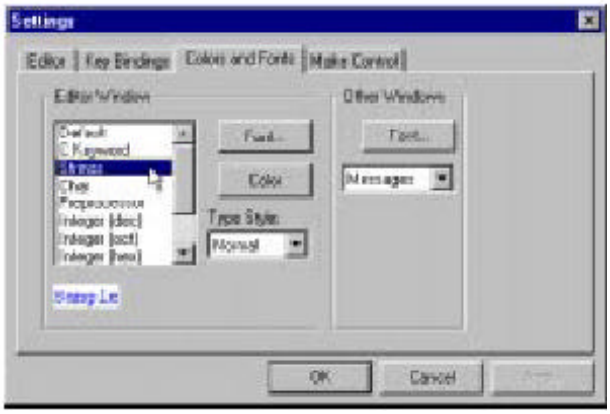
要删除快捷键，在 Current Shortcut（当前快捷键）列表中选择它并单击 Remove（删除），或单击 Remove All（删除全部）以删除所有命令的快捷键。

然后选定 OK，以便使用所定义的新的快捷键组合。

5.7.1.3 Colors and Fonts（颜色和字体）

允许用户规定在编辑器窗口中文本的颜色和字体，以及在其他窗口文本的字体。

面板显示用户可以在 Editor（编辑器）窗口中定制的 C 语法成分的列表：



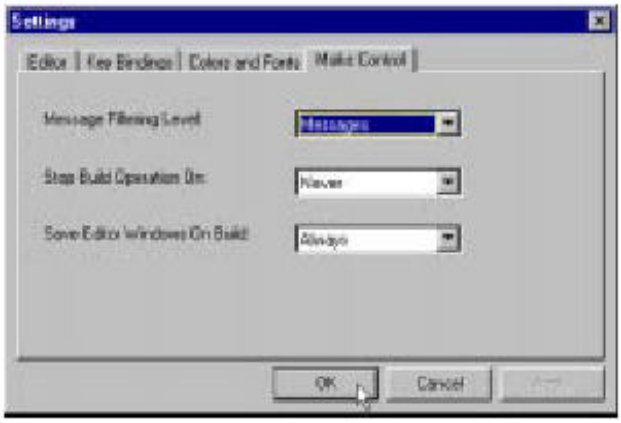
要规定在编辑器窗口中用于每一个 C 语法成分的样式，从 Editor Window（编辑器窗口）列表中选择想要定义的条目。当前设置由列表框下面的 Sample（样本）显示。

用户可通过单击 Color（颜色）选定文本颜色，单击 Font（字体）选定字体。用户也可从 Type Style（打印样式）下拉菜单中选定打印样式。

然后，选定 OK 以使用已定义的新样式或 Cancel（取消）以恢复原来的样式。

5.7.1.4 Make Control（生成控制）

用户可以设置适用于 Make（生成）和 Build（建立）的选项：



下表给出选择以及每一个选项的可替换的设置:

选项	设置
Message Filtering Level（消息过滤层）	All:显示所有消息
	Messages:显示消息、警告以及错误
	Warnings:显示警告和错误
	Errors:只显示错误
Stop Build Operation On（停止建立操作）	Never:不停止
	Warnings：在警告和出错时停止
	Errors:在出错时停止
Save Editor Windows On Build（建立时保存编辑器窗口）	Always:在 Make（生成）或 Build（建立）前总是保存
	Ask:在保存前提示
	Never:不保存

5.8 窗口菜单（Window Menu）

Window（窗口）菜单上的命令允许用户管理嵌入式工作平台 Embedded Workbench 的窗口并改变它们在屏幕上的排列。

Window（窗口）菜单的最后一部分列出了当前屏幕上打开的窗口，并允许用户通过选择一个窗口激活它。

5.8.1 New Window（新窗口）

为当前文件打开一个新窗口。

5.8.2 Cascade, Tile Horizontal, Tile Vertical（层叠，水平平铺，垂直平铺）

提供用于在屏幕上排列嵌入式工作平台 Embedded Workbench 窗口的标准 Windows 功能。

5.8.3 Arrange Icons（排列图标）

在嵌入式工作平台 Embedded Workbench 窗口的底部整齐地排列最小化窗口图标。

5.8.4 Split（拆分）

允许用户把编辑器窗口水平拆分为两部分以便能同时看到文件的两部分。

5.8.5 Message Window（消息窗口）

打开 Messages（消息）窗口，该窗口显示来自嵌入式工作平台 Embedded Workbench 命令的消息和文本输出。

5.9 帮助菜单（Help Menu）

Help（帮助）菜单上的命令提供有关嵌入式工作平台 Embedded Workbench 的帮助。

5.9.1 Contents（目录）

显示关于嵌入式工作平台 Embedded Workbench 帮助的 Contents（目录）页。

5.9.2 Search For Help On…（搜索关于…的帮助）

允许用户搜索有关关键字的帮助。

5.9.3 How To Use Help（怎样使用帮助）

显示有关使用帮助的帮助

5.9.4 About…（关于…）

显示嵌入式工作平台 Embedded Workbench 的版本号，并允许用户显示其他信息。

第6章 嵌入式工作平台编辑器

本章叙述嵌入式工作平台 Embedded Workbench 编辑器，并详细说明除了通常的鼠标操作之外可用于编辑 C 和汇编器源文件的键盘命令。

它也可以用于编辑普通的文本文件，例如 Read me 文件。

6.1 使用编辑器

6.1.1 打开一个源文件

要在嵌入式工作平台 Embedded Workbench 编辑器中编辑源文件，可在 Project（项目）窗口的树状显示中双击其图标，或者从 File（文件）菜单中选定 Open…（打开…），并选择想要编辑的文件。

当用户在编辑器窗口中编辑时，状态栏显示光标所在的当前行号和列号，以及[Caps lock]、[Numlock]以及[Scroiilock]的状态：

6.1.2 创建新文件

要创建新文件，可从 File（文件）菜单中选定 New…（新…）。下列对话框允许用户规定要创建的文件类型：

选择 Source/Text（源/文件）并选定 OK

新编辑器窗口将被打开。

6.1.3 保存文件

要保存当前编辑器窗口，从 File（文件）菜单中选定 Save（保存），或者选定 Save As…（另存为…）以便把它保存在不同的名字之下。

6.1.4 自动缩进

编辑器自动按照与前一行相同的缩进量缩进一行，这将使以结构化的方式设计程序变得容易。

6.1.5 匹配括号

当光标紧靠括号时，用户可通过选定 Edit（编辑）菜单上的 Match Brackets（匹配括号）来寻找匹配的括号。

6.1.6 语法突出显示

编辑器自动识别 C 程序的语法，并使用不同的文本样式和颜色显示 C 程序的不同组成成分。

用户可以使用 Settings（设置）对话框的 Colors and Fonts（颜色和字体）面板定制样式和颜色。

6.2 编辑器选项

嵌入式工作平台 Embedded Workbench 编辑器提供许多特殊的性能，在 Settings（设置）对话框的 Editor（编辑器）页中可以独立地使能和禁止这些性能中的每一个。详细资料请参见 5.7.1Setting…（设置…）。

6.3 编辑器键摘要

下表概括了编辑器的键盘命令:

6.3.1 移动插入点

移动插入点	按…
向左一个字符	←
向右一个字符	→
向左一个字	[Ctrl] ←
向右一个字	[Ctrl] →
向上一行	↑
向下一行	↓
至行的开始	[Home]
至行的末尾	[End]
至文件的第一行	[Ctrl] [Home]
至文件的最后一行	[Ctrl] [End]

6.3.2 文本滚动

滚动	按…
向上一行	[Ctrl] ↑
向下一行	[Ctrl] ↓
向上一页	[PgUp]
向下一页	[PgDn]

6.3.3 选择文本

选择…	按…
向左一个字符	[Shift] ←
向右一个字符	[Shift] →
向左一个字	[Shift] [Ctrl] ←
向右一个字	[Shift] [Ctrl] →
至上一行的相同位置	[Shift] ↑
至下一行的相同位置	[Shift] ↓
至行的开始	[Shift] [Home]
至行的末尾	[Shift] [End]
向上一屏	[Shift] [PgUp]
向下一屏	[Shift] [PgDn]
至文件的开始	[Shift] [Ctrl] [Home]
至文件的末尾	[Shift] [Ctrl] [End]

6.3.4 编辑

要…	按…
在插入（Insert）/覆盖键入（Overtype）方式之间切换	[Ins]
复制所选择的文件	[Ctrl]C 或[Ctrl] [Ins]
剪切所选择的文本	[Ctrl]X 或[Shift] [Del]
粘贴文本	[Ctrl]V 或[Shift] [Ins]

撤消上一次编辑	[Ctrl] Z
插入空格直至下一个制表位置为止	[Tab]

6.3.5 删除

删除…	**按…**
向左一个字符	←
向右一字符	[Delete]

6.3.6 搜索

要…	**按…**
用编辑栏搜索文本	[Ctrl]F