

# 目 录

## 引言

- 介绍
- 重点
- 关于本指南
- 质量保证注册
- 推荐阅读
- MICROCHIP 互联网站
- 开发系统用户信息更新服务
- 用户支持

## 第1章 MPLAB 概述

- 1.1 介绍
- 1.2 重点
- 1.3 什么是 MPLAB-IDE?
- 1.4 MPASM 能帮你做什么?
- 1.5 MPLAB-IDE: 集成开发环境(IDE)
- 1.6 MPLAB-IDE 开发工具

## 第2章 MPASM-安装与入门

- 2.1 介绍
- 2.2 重点
- 2.3 对于计算机主机的配置要求
- 2.4 从哪里得到软件包
- 2.5 安装 MPLAB-IDE
- 2.6 卸除安装 MPLAB IDE 集成开发环境

## 第3章 MPLAB IDE 使用入门 – 指导

- 3.1 介绍
- 3.2 重点
- 3.3 设置开发模式
- 3.4 建立一个简单的项目
- 3.5 建立一个简单的源文件
- 3.6 输入源文件代码
- 3.7 对源文件进行汇编
- 3.8 运行你的程序
- 3.9 打开其他窗口帮助调试
- 3.10 使用观察窗口
- 3.11 设置断点
- 3.12 总结

**第4章 MPLAB IDE “项目” 指南**

- 4.1 介绍
- 4.2 重点
- 4.3 MPLAB IDE“项目”概述
- 4.4 建立一个只有一个 MPASM 源文件的“项目”
- 4.5 不用建立项目编译单个 MPASM 源文件
- 4.6 使用 MPLINK 创建一个有多个源文件的项目
- 4.7 使用 HI-TECH PIC C 编译器创建项目
- 4.8 用 MPLAB-C17 或者 MPLAB-C18 创建一个项目

**第5章 MPLAB 编辑器**

- 5.1 介绍
- 5.2 提示
- 5.3 什么是编辑器
- 5.4 对你有何帮助
- 5.5 MPLAB 编辑器的特点

**第6章 调试和 MPLAB-SIM 模拟器**

- 6.1 概述
- 6.2 重点
- 6.3 MPLAB IDE 调试功能
- 6.4 “实时”执行程序
- 6.5 MPLAB-SIM 模拟器环境
- 6.6 模拟器需要考虑的问题
- 6.7 断点和跟踪点
- 6.8 条件断点会话窗口
- 6.9 “激励”功能
- 6.10 12 位核芯片的模拟
- 6.11 14 位核芯片的模拟
- 6.12 16 位核处理器的模拟
- 6.13 扩展型 16 位核处理器的模拟

**第7章 MPLAB 编辑器工具栏和菜单的使用**

- 7.1 介绍
- 7.2 重点
- 7.3 MPLAB IDE 桌面
- 7.4 文件菜单
- 7.5 “项目”菜单
- 7.6 编辑菜单
- 7.7 汇编菜单
- 7.8 编程器菜单
- 7.9 工具菜单 (F11)
- 7.10 窗口菜单
- 7.11 帮助菜单

**附录 A MPLAB IDE 里使用的快捷键及其相应的功能**

- A.1 概述
- A.2 MPLAB IDE 快捷键及其对应功能

**附录 B: MPLAB IDE 里使用的文件扩展名****附录 C: MPLAB IDE 工具栏和状态栏定义**

- C.1 MPLAB IDE 工具栏
- C.2 MPLAB IDE 状态栏

**附录D MPLAB编辑器默认命令键**

- D. 1 概述
- D. 2 重点
- D. 3 功能键
- D. 4 移动键
- D. 5 控制键 (Control Keys)
- D. 6 格式和编辑键 (Formatting and Editing Keys)

**缩略语**

- 概述
- 术语 – MPLAB IDE, MPLAB-SIM, MPLAB Editor
- 术语 – MPASM, MPLINK, MPLIB
- 术语 – MPLAB-CXX
- 术语 – MPLAB-ICD
- 术语 – PICSTART PLUS, PRO MATE

## 引言

### 介绍

在开始使用 MPLAB-IDE 之前，看一下本章介绍的通用信息将很有好处。

### 重点

本章你会得到如下信息：

- 关于本指南
- 质量保证注册
- MICROCHIP 的互联网站
- 开发工具用户信息更新服务
- 客户支持

### 关于本指南

文档资料安排如下：

本手册介绍如何使用 MPLAB-IDE ( INTEGRATED DEVELOPMENT ENVIRONMENT 集成开发环境。本指南的内容安排如下：

- **第 1 章 MPLAB-IDE 概述** – 介绍什么是 MPLAB-IDE 以及它的工作原理。
- **第 2 章 MPLAB-IDE 的安装与入门** - 描述如何安装 MPLAB-IDE 到计算机里。
- **第 3 章 开始使用 MPLAB-IDE – 示范** - 描述如何使用 MPLAB-IDE 进行开发工作。
- **第 4 章 MPLAB-IDE “项目” 指南** - 如何使用 MPLAB-IDE 的“项目”。
- **第 5 章 MPLAB 编辑器** – 介绍基本的 MPLAB 编辑器的功能和特性。
- **第 6 章 调试和 MPLAB-SIM 模拟器** – 讨论 MPLAB-IDE 的调试功能以及使用 MPLAB-SIM 模拟器需要考虑的问题。
- **第 7 章 MPLAB-IDE 菜单和工具栏的选项** – 介绍 MPLAB-IDE 菜单和工具栏的选择项。本章包括所有与 MPLAB 编辑器相关的菜单选项。

### 附录

- **附录 A MPLAB-IDE 功能键** – 列出所有 MPLAB-IDE 环境下使用的功能键。
- **附录 B MPLAB-IDE 安装后的用户配置** – 通过修改 MPLAB-IDE 配置文件 (MPLAB.ini) 来进行用户自定义设置。
- **附录 C MPLAB-IDE 使用的文件扩展名** – 介绍 MPLAB-IDE 使用的文件扩展名，辨别讲解了每种默认的文件扩展名。
- **附录 D MPLAB-IDE 工具栏和状态栏的定义** – 辨别讲解了每种 MPLAB-IDE 工具栏按钮和它们的功能，而且讲解了如何识别 MPLAB-IDE 状态栏里显示的信息。
- **附录 E MPLAB-IDE 默认命令键** - 详细说明由 MPLAB 编辑器指定使用的默认命令键，同时列出了等价的菜单命令（假如存在的话）。
- **术语** - 在本指南中用到的术语。
- **索引** - 本手册中涉及到的术语 (TERMS)，特性 (FEATURES) 和章节 (SECTIONS)。
- **全球销售服务网** - 列出 MICROCHIP 全球的销售服务点的电话，传真以及地址。

本指南中用到的一些表示符号：

本指南中用到了下列约定符号：

描述	说明	例子
<b>代码 (Courier 字体)</b>		
Courier 字体	用户输入代码或例子	#define ENIGMA
尖括号 <>	变量，需要你填入参数	<标号>,<表达式>
方括号 [ ]	可选项	MPASMWIN [main.asm]
大括号{} 竖线	互斥选项；“或”选项	Errorlevel { 0   1 }
引号中的小写体	数据类型	“filename”
省略号 ...	表示一下而不是一列出与本例无关的文本项	List [“列表选项”,...,”列表选项”]
0xnnnnn	表示 16 进制数	0xFFFF, 0x007A
<b>界面</b>		
带下划线和右箭头的斜体字	表示菜单选项	<i>File&gt;Save</i>
文本中的粗体字	表示一个按钮	<b>OK,Cancel</b>
尖括号中的大写字	表示一些特殊键	<TAB>,<ESC>
<b>文档资料</b>		
斜体字	参考书	<i>MPLAB User's Guide</i>

## 资料更新

所有的文档资料都有过时的时候，本用户指南也不例外。因为 MPASM, MPLINK, MPLIB 及其他 MICROCHIP 的开发工具需要经常更新，以满足用户的要求。某些实际看到的对话框和/或工具的使用说明可能与本指南有所不同。请到 MICROCHIP 的网站获取最新的文档资料。

## 质量保证注册

请填写登记卡并立即邮寄出来，这将保证您得到及时的软件升级，MICROCHIP 的网站上可得到这些更新软件。

## 推荐阅读

本指南讲述如何使用 MPASM, MPLINK 和 MPLAB。用户也可以在开发软件包里找到某一特定型号单片机的详细资料。

README.ASM, README.LKR

阅读 README 文档 (ASCII 文档资料) 了解最新的 MPASM 和 MPLINK 信息。该文档资料中包含了本指南没有提及的软件的升级信息。

### MPLAB 用户指南 (DS51025)

详细地介绍了 MPLAB-IDE 集成开发环境软件包的特性及安装以及编辑器和模拟器的使用。

### 技术资料 CD-ROM (DS00161)

这张 CD-ROM 中包含了 MICROCHIP 最新的 PICmicro 器件详细数据手册。请联系您最近的 MICROCHIP 办事处索取光碟。也可以从 WWW.MICROCHIP.COM 网站下载。

### 嵌入控制技术手册 (DS00092 和 DS00167) 第一卷和第二卷。

这两本手册包含丰富的单片机应用信息。请联系您最近的 MICROCHIP 办事处索取。(见手册后面附录)

这些手册中讲述的应用例子也可以从 MICROCHIP 的销售和办事处得到，还可以从 MICROCHIP 的网站上下载。

### **MICROSOFT WINDOWS 手册**

本指南认为用户已熟悉 WINDOWS 的使用，市面有许多关于 WINDOWS 的书可以作为参考。

### **MICROCHIP 互联网站**

MICROCHIP 在 INTERNET 上提供在线帮助，

MICROCHIP 把 INTERNET 作为一个方便地给用户提文件和信息的手段。用户需要有 INTERNET 浏览器，比如 NETSCAPE NAVIGATOR 或者 MICROSOFT INTERNET EXPLORER。用户也可以从我们的 FTP 服务器上下载。

连接到 MICROCHIP 的 INTERNET 站点

你可以在你熟悉的浏览器中输入 MICROCHIP 的网站地址：

**HTTP: //WWW.MICROCHIP.COM**

(中文网站为: **HTTP: //WWW.MICROCHIP.COM.CN – 译者**)

若想连接到 FTP 服务器，请输入：

**FTP: //FTP.MICROCHIP.COM**

WEB 网和 FTP 站提供很多服务。用户可以下载最新的开发软件，数据手册，应用笔记，用户指南，技术文章和例程。还有许多 MICROCHIP 的商业信息，包括办事处，分销商。其他用户关心的信息还包括：

- MICROCHIP 最新发布消息
- 常见问题解答
- 设计指导与提示
- 器件资料勘误
- 工作机会
- MICROCHIP 咨询顾问成员名单
- 与其他和 MICROCHIP 产品相关网站的链接。
- 更多关于产品，开发系统，技术研讨会的信息。

### **开发系统用户最新信息提供服务**

MICROCHIP 通过用户最新信息提供服务尽力使用户随时得知 MICROCHIP 的最新产品信息。一旦你购买以下产品，我们会随时 EMAIL 通知你关于软件升级，修改或关于产品系列及开发工具的勘误。其他服务请参见 MICROCHIP 网站。

开发系统产品包括：

- 。编译器
- 。仿真器
- 。编程器
- 。MPLAB 软件包
- 。其他工具

假如你对以上产品之一感兴趣的话，可以联系下列 EMAIL 地址订购：

[listserv@mail.microchip.com](mailto:listserv@mail.microchip.com)

请使用如下格式：

subscribe <listname> yourname

给你一个范例：

```
subscribe mplab John Doe
```

假如你想撤消定单的话，可以 EMAIL 给以下地址：

[listserv@mail.microchip.com](mailto:listserv@mail.microchip.com)

请使用如下格式：

```
unsubscribe <listname> yourname
```

给你一个范例：

```
unsubscribe mplab Jhon Doe
```

以下将一一介绍现有的开发系统。

### 仿真器：

MICROCHIP 最新的在线仿真器。包括 MPLAB-ICE，PICMASTER。

假如你对以上产品感兴趣的话，可以联系下列 EMAIL 地址订购：

[listserv@mail.microchip.com](mailto:listserv@mail.microchip.com)

请使用如下格式：

```
subscribe emulators yourname
```

### 编程器：

MICROCHIP 最新的编程器，包括 PROMATE，PICSTART PLUS。

假如你对以上产品感兴趣的话，可以联系下列 EMAIL 地址订购：

[listserv@mail.microchip.com](mailto:listserv@mail.microchip.com)

请使用如下格式：

```
subscribe programmers yourname
```

### 编译器：

MICROCHIP 最新的 C 编译器，链接器，汇编器的信息包括 MPLAB-C17，MPLAB-C18，MPLINK，MPASM 和库，MPLINK 的库 MPLIB。

假如你对以上产品感兴趣的话，可以联系下列 EMAIL 地址订购：

[listserv@mail.microchip.com](mailto:listserv@mail.microchip.com)

请使用如下格式：

```
subscribe compilers yourname
```

### MPLAB 集成开发软件包：

WINDOWS 集成开发环境 MPLAB 的最新信息，本产品集中于 MPLAB，MPLAB-SIM 模拟器，MPLAB 项目管理器，欲知通用编辑器，调试器。欲知编译器，链接器，汇编器的详细情况，请订阅编译器产品清单（COMPILER LIST）。欲知 MPLAB 模拟器的详细情况，可订阅模拟器产品清单。欲知 MPLAB 编程器的详细情况，可订阅编程器产品清单。

假如你对以上产品清单感兴趣的话，可以联系下列 EMAIL 地址订阅：

[listserv@mail.microchip.com](mailto:listserv@mail.microchip.com)

请使用如下格式：

```
subscribe mplab yourname
```

### 其他开发工具：

MICROCHIP 提供的最新关于其他开发工具的信息。欲知 MPLAB 及其集成开发工具

的详细情况。可联系其他 EMAIL 地址。

假如你对以上产品清单感兴趣的话，可以联系下列 EMAIL 地址订阅：

[listserv@mail.microchip.com](mailto:listserv@mail.microchip.com)

请使用如下格式：

`subscribe otools yourname`

## 用户支持

使用 MICROCHIP 的产品可以通过以下渠道得到帮助：

- 分销商及代表处
- 当地销售处
- 现场应用工程师（FAE）
- 厂家应用工程师（CAE）

热线电话

用户可以打电话给分销商，代表处，或现场应用工程师寻求帮助。当地销售处也可接受用户的求助。见手册后面的销售处电话和地址。联系厂家应用工程师（CAE）可打电话：

**(602) 786-7627**

另外，一条专线电话专门为用户提供系统信息和升级消息。用户可以及时得到开发系统软件的最新版本清单。用户还可以得到如何得到最新升级软件的信息。

这些热线电话是：

**1-800-755-2345** 美国及加拿大的大部分地区

**1-602-786-73-2** 世界其他地区



## 第1章 MPLAB 概述

### 1. 1 介绍

本章就 MPLAB 及其功能做一个概述。

### 1. 2 重点

本章内容包括：

- 。什么是 MPLAB-IDE？
- 。MPLAB-IDE 能帮你干什么？
- 。MPLAB-IDE：集成开发环境（Integrated Development Environment）
- 。MPLAB-IDE 开发工具

### 1. 3 什么是 MPLAB-IDE？

MPLAB-IDE是基于Windows®的集成开发环境应用软件包，是为PICmicro®系列微控制器（MCU）专门设计开发的。MPLAB-IDE允许用户编写、调试和优化作PICmicro®系列MCU的应用程序代码。它包含文本编辑器、模拟器、项目管理器。该软件包环境还支持MPLAB-ICE 和 PICMASTER® 仿真器以及PICSTART® Plus、PRO MATE® II烧写器。还支持其他的MICROCHIP和第三方开发系统工具。

### 1. 4 MPASM 能帮你做什么？

MPLAB-IDE 集成开发环境里的工具按照相应的功能来安排归类，这有助于让下拉菜单和用户自定义项很容易被找到和使用。MPLAB-IDE 允许你：

- 汇编、编译、链接源代码
  - 通过监视模拟器的程序流程或 MPLAB-ICE 仿真器的适时操作，可以对可执行逻辑代码进行调试
  - 进行定时测量（timing measurements）
  - 察看监视窗口里的变量值
  - 使用 PICSTART Plus 或 PRO MATE II 芯片烧写器把支持程序写入相应的芯片里。
  - 通过 MPLAB-IDE 在线帮助可以快捷地查询到所遇到问题的解答。
- 还有好多功能.....

### 1. 5 MPLAB-IDE：集成开发环境 Integrated Development Environment (IDE)

MPLAB-IDE 是一种易学的软件包，而且拥有集成开发环境。该集成软件包为支持软件（firmware）开发工程师提供灵活的开发和调试 Microchip 的 PICmicro 系列 MCU 支持软件的工具。MPLAB-IDE 可以运行于 Windows 3.1x、Windows 95/98、WindowsNT 或 Windows 2000。

注意：不是所有的硬件都能运行于 MPLAB-IDE 集成开发环境之下（比如仿真器、烧写器等），运行于一切操作系统。详细情况请参考相应硬件的用户指南。

MPLAB-IDE 集成开发环境提供以下功能：

- 建立和编辑源文件
- 将多个文件组合到项目里
- 调试源代码
- 使用模拟器或仿真器对可执行逻辑代码进行调试

MPLAB-IDE 提供一个全功能的文本编辑器，用它来进行源文件的建立和编辑工作。

更进一步，用户可以在创建结果（**Build Results**）窗口里显示的出错信息帮助下，很轻松地调试源代码。出错信息是由编译器、汇编器、链接器在生成可执行代码文件时输出的。

项目管理器可以把多个源文件、预先编译目标文件、库文件、链接器描述文件组合成为一个“项目”格式。

MPLAB-IDE 集成开发环境还提供强大的模拟和仿真特性，用来调试可执行代码。主要特性有：

- 大量的观察窗口可以让用户可以观察所有的数据和程序存储器里的内容。
- 源代码、程序存储器、绝对列表窗口允许用户既可以分别也可以同时观察源代码和汇编级指令（绝对列表）。
- 可以跟踪进入（trace through）、应用断点（apply break）、跟踪（trace）、标准或复杂的触发点（trigger point）。

## 1. 6 MPLAB-IDE 开发工具

MPLAB-IDE 集成了几种工具，这些工具提供了完全的开发环境。

- **MPLAB 项目管理器**

使用项目管理器来建立项目，使得与项目有关的文件一起工作。当使用项目的时候，只需点击鼠标用户就可以重新创建（rebuild）源代码并将其下载到仿真器里。

- **MPLAB 编辑器**

使用 MPLAB 编辑器可以建立和编辑文本文件，比如源文件、代码、链接器描述文件等。

- **MPLAB-ICD 在线调试器**

MPLAB-ICD 在线调试器是一个功能强，价格低的开发与评估套件。针对 PIC16F87X 系列带有快闪存储器（FLASH）的微控制器。

- **MPLAB-SIM 软件模拟器**

软件模拟器对 PICmicro 系列微控制器的指令和 I/O 进行建模。

- **MPLAB-ICE 仿真器**

MPLAB-ICE 仿真器使用硬件手段实时地对 PICmicro 系列微控制器进行仿真。有或没有目标系统都可以照样进行仿真。

- **MPASM 汇编器、MPLINK 链接器、MPLIB 库管理器**

MPASM 汇编器允许用户不用离开 MPLAB-IDE 集成开发环境就可以对源代码进行汇编。MPLINK 将从 MPASM 宏汇编、MPLAB-C17 和 MPLAB-C18 生成的可重定位模块进行链接，产生最后的可执行应用代码文件。

- **MPLAB-CXX 系列 C 编译器**

MPLAB-C17 和 MPLAB-C18C 编译器提供基于 ANSI 标准的高级语言解决方案。

- **PROMATE II 和 PICSTART Plus 芯片烧写器**

用模拟器和仿真器开发应用代码，对源程序进行汇编或编译，然后使用这些工具进行烧写。这些工作都可以通过 MPLAB-IDE 集成开发软件包来完成。尽管 PRO MATE II 不用通过 MPLAB-IDE，但如果使用 MPLAB-IDE 开发环境的话，会更加容易和方便。

- **PICMASTER 和 PICMASTER-CE 仿真器**

这些仿真器可以在没有目标板的情况下使用硬件方式来实时地仿真 PICmicro 系列微控制器。MPLAB-ICE 是 MICROCHIP 最新的仿真器系列。

- **第三方开发工具**

很多其他的公司有针对 MICROCHIP 产品的工作于 MPLAB-IDE 集成开发环境的开发工具。详细情况请参考《MICROCHIP 第三方开发工具指南》文档资料号：DS00104。

## 第2章 MPASM-安装与入门

### 2. 1 介绍

本章指导你安装 MPLAB-IDE 集成开发环境到你的 PC 上。

### 2. 2 重点

本章内容包括：

- 软件包对于计算机主机的配置要求
- 从哪里得到软件包？
- 安装 MPLAB-IDE 集成开发环境软件包
- 卸除安装 MPLAB-IDE 集成开发环境软件包

### 2. 3 对于计算机主机的配置要求

以下是 MPLAB-IDE 集成开发环境对于计算机的最低配置要求：

- 奔腾（PENTIUM）级兼容计算机
- WINDOWS3.X、WINDOWS95 或 WINDOWS98、WINDOWS NT 或 WINDOWS2000 版本。
- 16MB 的系统内存（推荐使用 32MB 内存）。
- 45MB 的空余硬盘空间。

注意：不是所有的硬件都能运行于 MPLAB-IDE 集成开发环境之下（比如仿真器、烧写器等），运行于一切操作系统。详细情况请参考相应硬件的用户指南。

### 2. 4 从哪里得到软件包

在 MICROCHIP 的每一个开发工具里都配套有 MPLAB-IDE 集成开发软件包。而且 MPLAB-IDE 软件包还可以通过 MICROCHIP 的销售办公室索取软件包的 CD-ROM 光碟。也可以访问 MICROCHIP 的国际互联网站点：

[www.microchip.com](http://www.microchip.com) 或 [www.microchip.com.cn](http://www.microchip.com.cn)

软件包里的文件名称和数量可能因版本的不同而有所不同。比如：

MPLAB-IDE 集成开发软件包的版本 4.00 有如下文件：

MP4000.EXE

MP4000.W02

MP4000.W03

MP4000.W04

MP4000.W05

MP4000.W06

### 2. 5 安装 MPLAB-IDE

可执行文件 MPXXXXX.EXE 用来安装 MICROCHIP 的 MPLAB-IDE 集成开发环境。其中 XXXX 是软件包的版本代号。

按如下步骤安装 MPLAB-IDE 集成开发环境软件包：

1. 进入 MICROSOFT 的 WINDOWS 操作系统。
2. 假如从 CD-ROM 安装 MPLAB-IDE 集成开发环境软件包，请将 CD-ROM 放置到光盘驱动器里。

## 3. 执行安装程序:

对于 WINDOWS31 操作系统: 从文件管理器里或选择命令:  
**Program Manager>Run**, 运行 **X:\MPvvvvv.exe**, 其中: **X** 是 MPLAB-IDE 集成开发环境软件包所在的驱动器盘符, **vvvvv** 是当前安装软件包的版本号。例如: 输入 **d:\MP41219.exe** 意为: 在光盘驱动器 **d** 盘里包含有 MPLAB-IDE 软件包的 CD-ROM 软件包的版本号为: **4.12.19**。

对于 WINDOWS95/98, WINDOWS NT 或 WINDOWS 2000 操作系统: 点击 Start 按钮, 选择 Run, 输入 X: \MPvvvvv.exe, 其中: **X** 是 MPLAB-IDE 集成开发环境软件包所在的驱动器盘符, **vvvvv** 是当前安装软件包的版本号。例如: 输入 **d:\MP41219.exe** 意为: 在光盘驱动器 **d** 盘里包含有 MPLAB-IDE 软件包的 CD-ROM 软件包的版本号为: **4.12.19**。然后点击 **OK** 按钮即可。

**注意:** WINDOWS NT 用户必须有管理权限以便安装 MPLAB-ICE

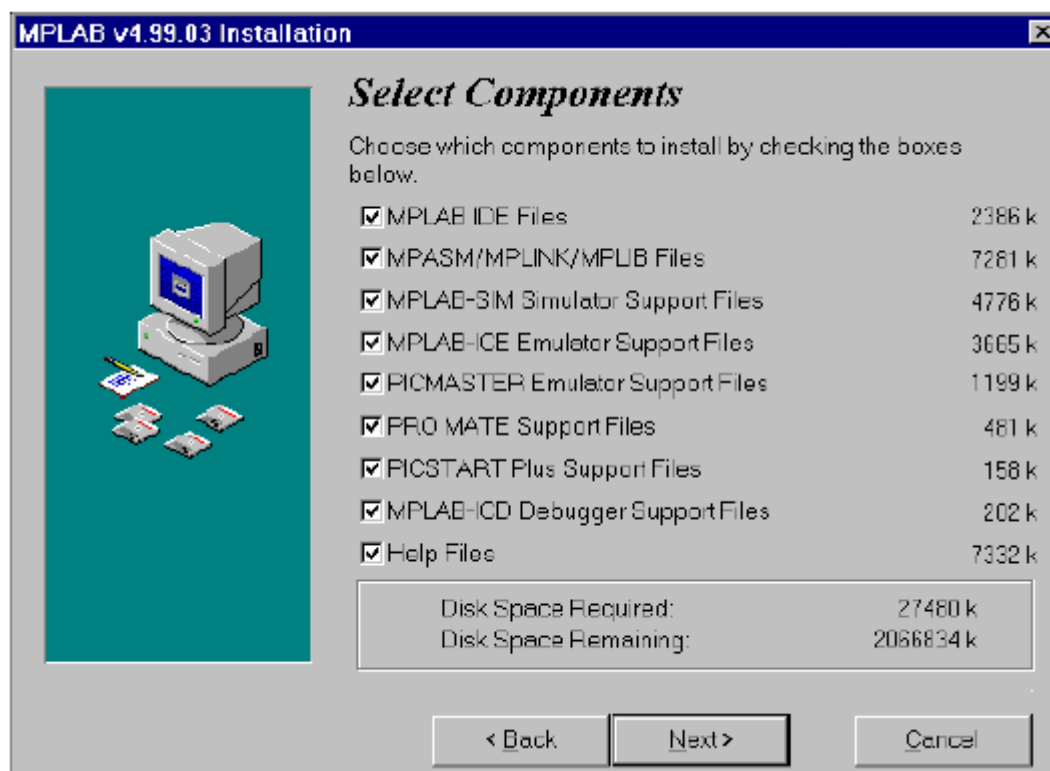
4. 安装程序将安装 MPLAB-IDE 集成开发环境软件包到你的系统里。  
安装程序将会显示出软件包里的所有的部件, 以便你选择安装所需要的部件。

图 2-1: 选择软件包部件会话窗口

如果你还没有购买芯片烧写器或仿真器, 则安装以下部件就可以了:

- MPLAB IDE 集成开发软件包文件
- MPASM/MPLINK/MPLIB 文件
- MPLAB-SIM 模拟器支持文件

- 帮助文件

你也可以随后重新安装 MPLAB-IDE，添加其他的功能元件模块。

5. 下面的会话窗口将会允许你选择想要安装的 MICROCHIP 开发语言工具模块。通常需要全部选中（默认配置）。

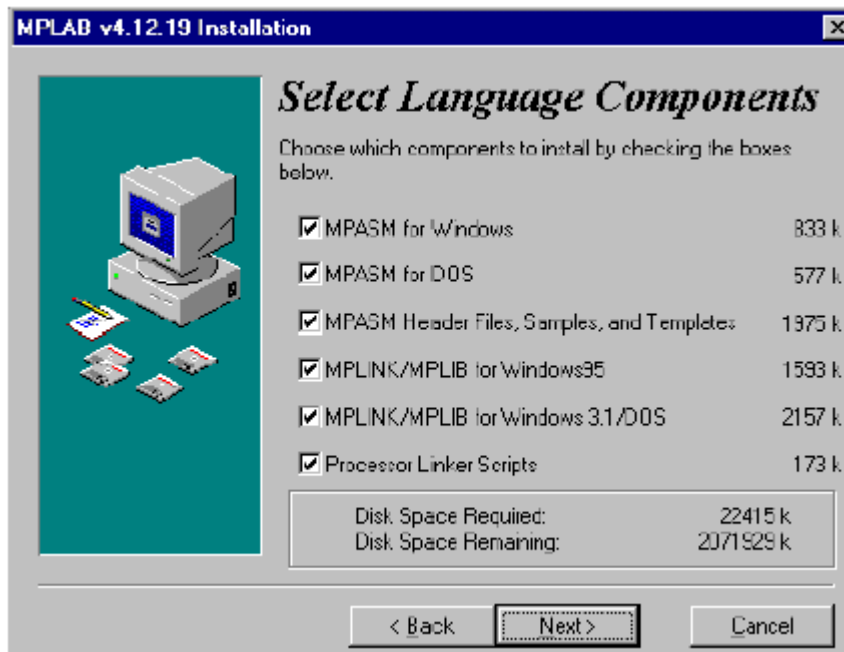


图 2-2: 选择开发语言工具会话窗口

6. 接下来将要选择 MPLAB-IDE 软件包要安装到的目录。通常要选择默认目录路径 C:\Program Files\MPLAB。推荐将 MPLAB-IDE 软件包安装到本地硬盘，而不是安装到其他工作驱动器。

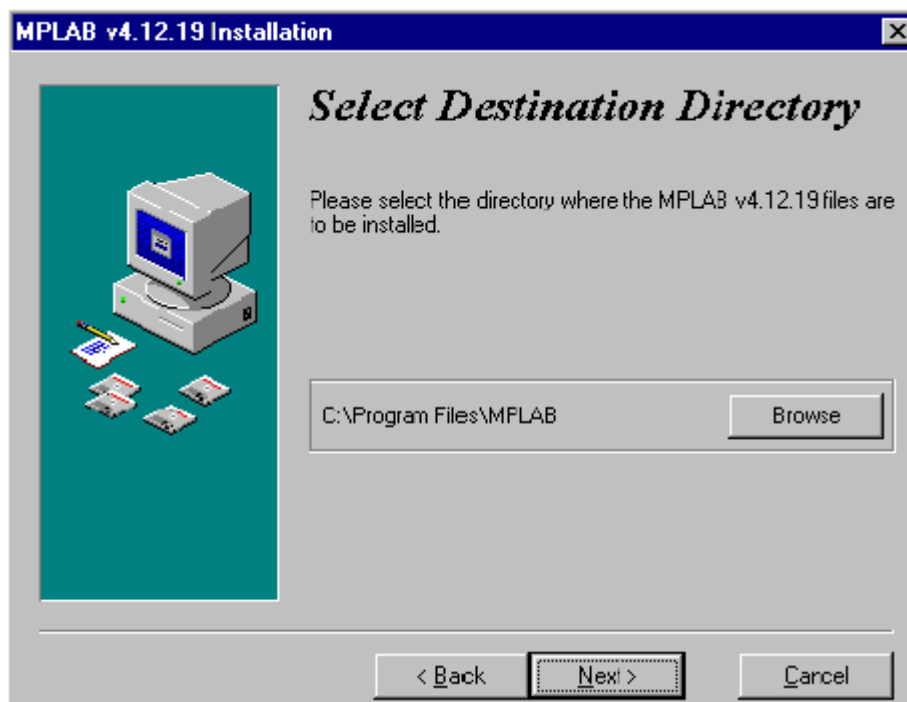


图 2-3：选择安装目录

7. 下一个会话窗口将会选择是否对安装过程中被安装程序替换的文件做一个备份。

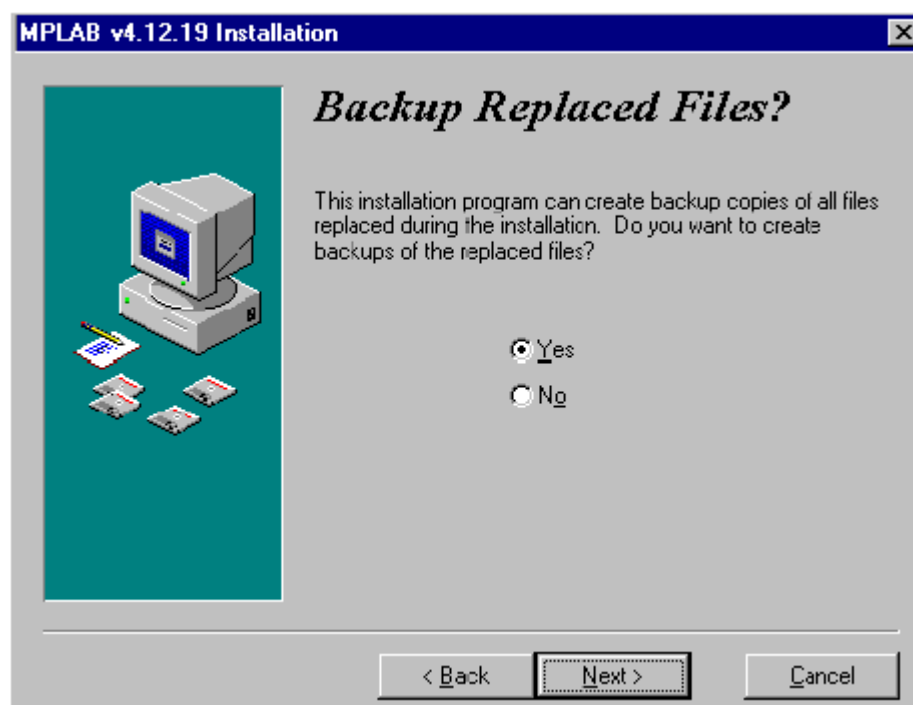


图 2-4：备份被替换文件会话窗口

8. 假如上一步里选择了 YES，则选择备份目录窗口将会弹出来，所有的备份文件都必须放在该目录里。选择默认的备份目录 BACKUP，该目录是 MPLAB 目录的子目录。

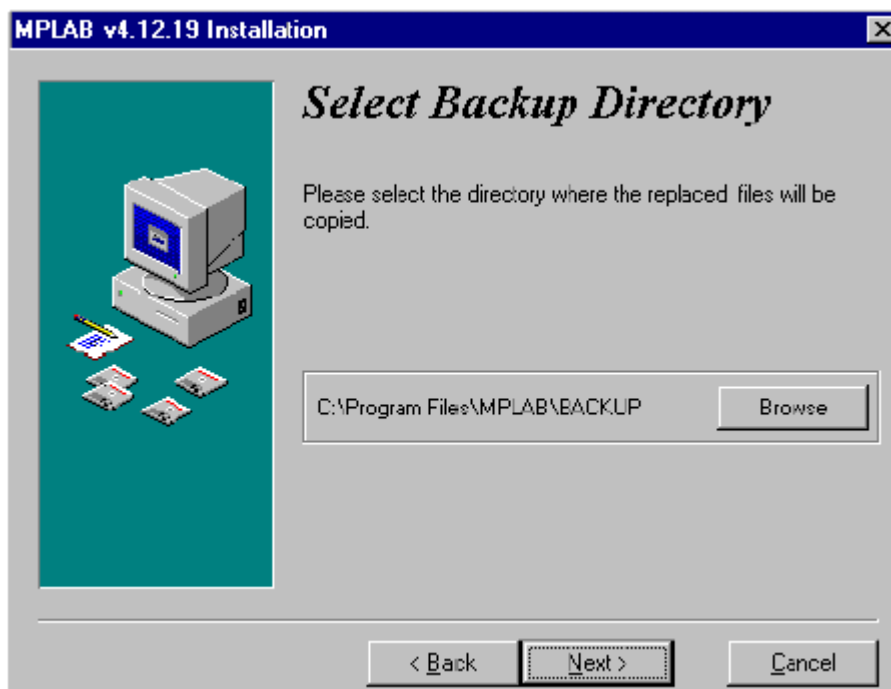


图 2-5：选择备份目录会话窗口

9. 接下来的窗口将允许你选择是否将 MPLAB-IDE 程序（可执行）放置到 Windows 95/98、Windows NT 或 Windows 2000 环境下的启动菜单里。（对于 Windows 3.1 不具有这个选择项）

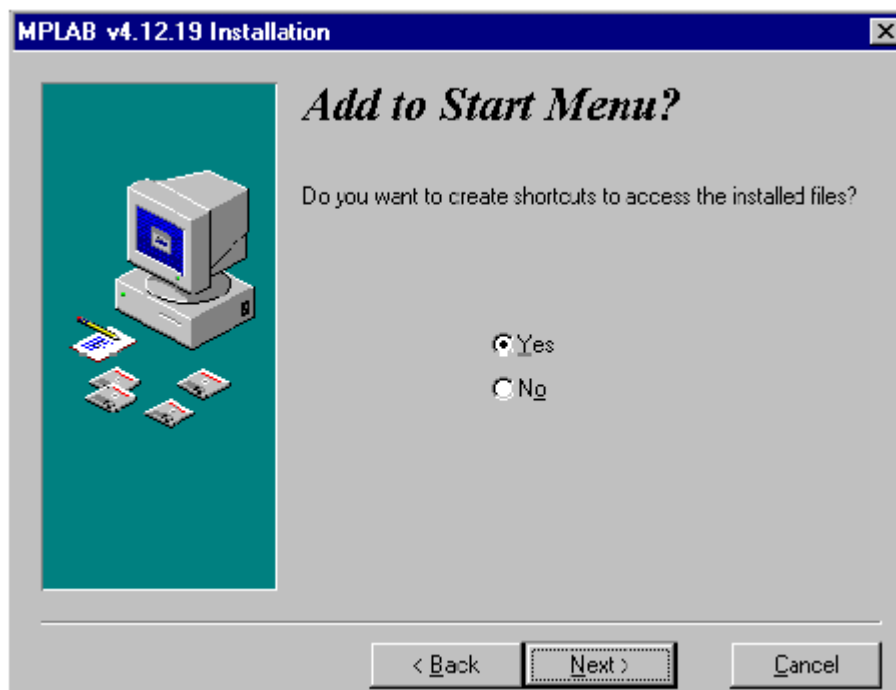


图 2-6：启动菜单会话窗口



10. 接下来的菜单允许你选择是否将 MPLAB-IDE 链接器描述内容安装到 MPLAB 主目录里还是安装到“\Lkr”子目录里。



图 2-7: 链接器描述内容安装目录选择

11. 下一个窗口允许你选择是否将 MPLAB-IDE 使用的系统文件安装到 MPLAB 子目录里还是安装到计算机上的 System 目录里。把一些诸如数据链接库文件（.DLL）安装到 System 目录里可以更好地管理并可以防止将来安装另外的程序时把 MPLAB IDE 文件覆盖。



图 2-8: 选择系统文件安装目录会话窗口

12. MPLAB-IDE 安装程序将安装 MPLAB-IDE 到你的计算机上。在安装过程中，会显示一些关于 MPLAB-IDE 的新特性。
13. 安装过程最后的会话窗口（图 2-9）将会询问你是否阅读 README 文件。该文件包含了很多有用的信息，比如：版本的新特性、软件的局限、已知存在的一些问题。

注意：假如选择 NO，你可以以后再到 MPLAB 安装目录里阅读这些内容。建议你在寻求技术支持之前先阅读 readme 文件。

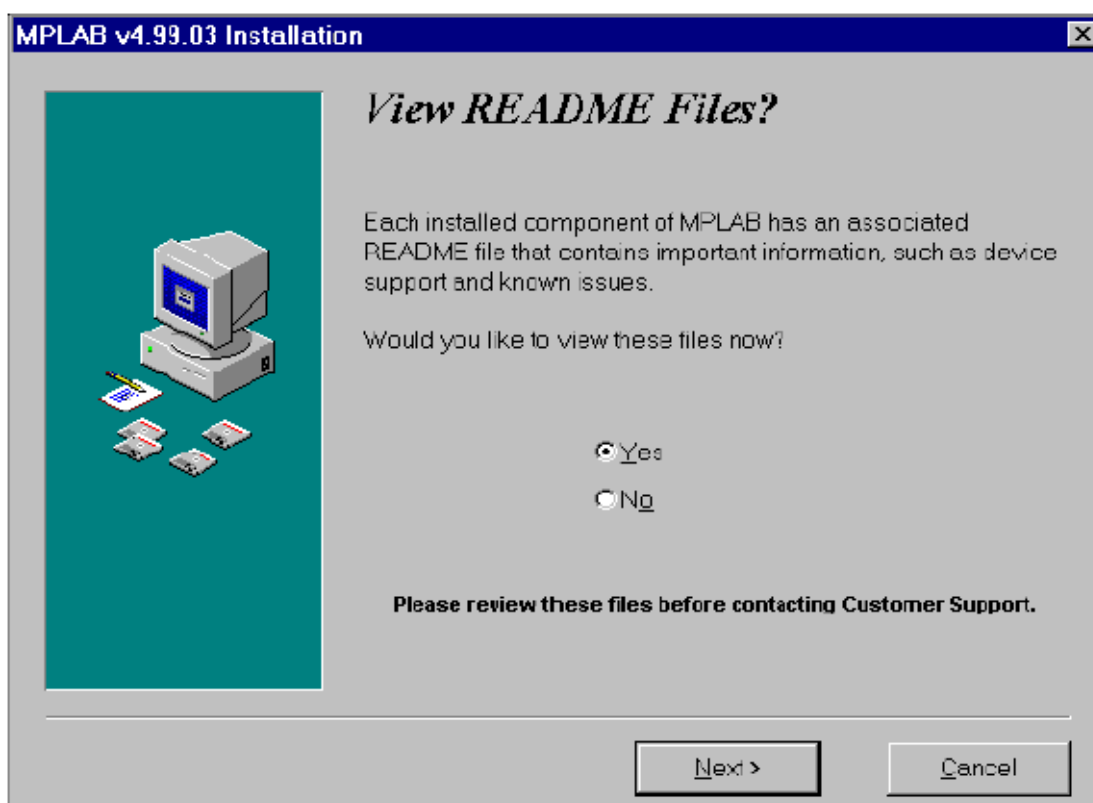


图 2-9: Readme.tif 文件

14. 安装完毕后，执行 MPLAB.EXE 或点击 MPLAB IDE 图标来启动 MPLAB IDE 集成开发环境。你将看到 MPLAB IDE 的桌面，见图 2-10 所示：

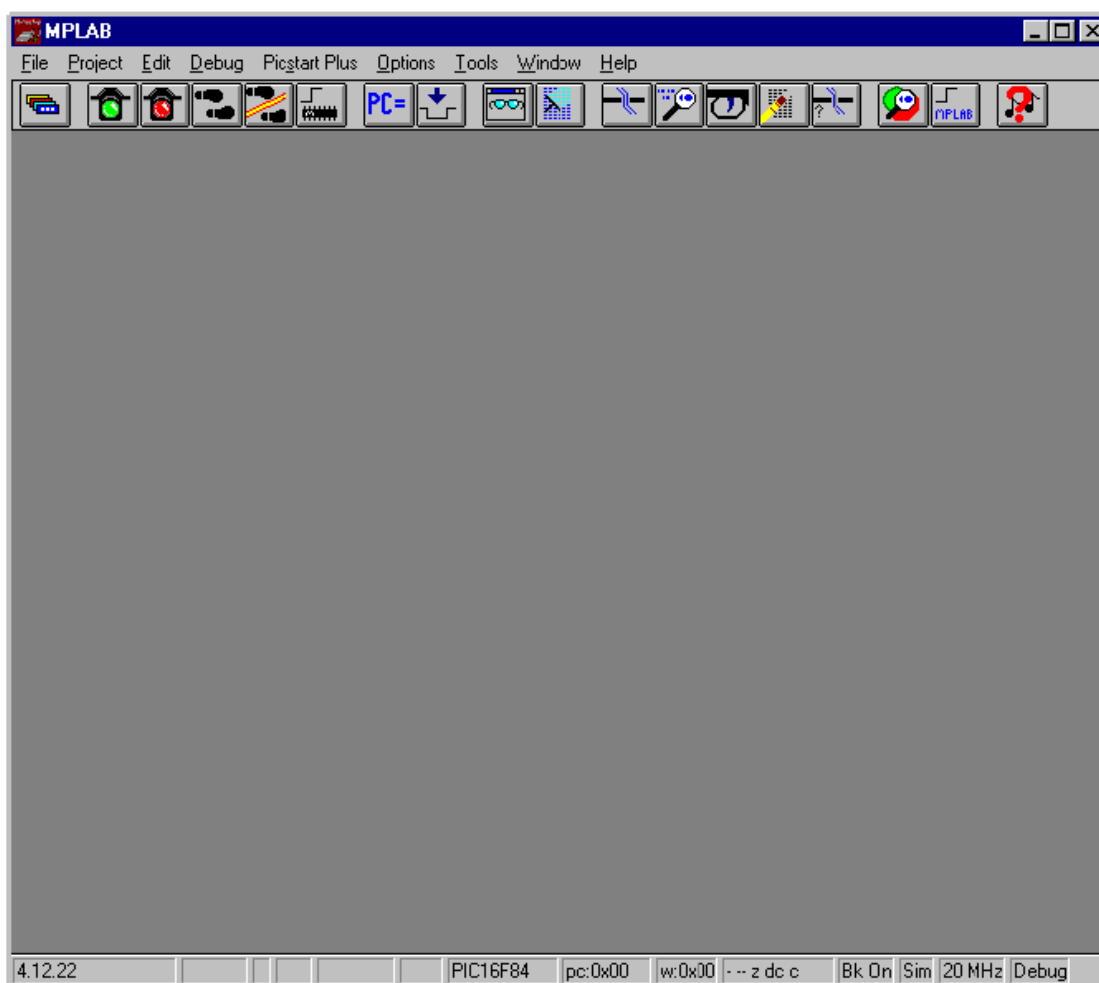


图 2-10: MPLAB-IDE 桌面

### 1 2 . 1 卸除安装 MPLAB IDE 集成开发环境

如果你想卸除安装 MPLAB IDE，打开 Windows 浏览器（或文件管理器 WINDOWS31）并双击 unwise.exe 文件，这时候反安装文件将会运行，并确定删除位于 MPLAB IDE 目录、Windows 目录、系统目录里相应的文件。

## 第3章 MPLAB IDE 使用入门 – 指导

### 3.1 介绍

本指导对 MPLAB IDE 用户界面做一个简要快速的介绍。花一到两个小时可以初步了解 MPLAB IDE 的使用入门。

本章并不准备详细地介绍 MPLAB IDE 的功能，仅仅做一个入门介绍，让你可以立即开始使用 MPLAB IDE 集成开发软件包。

### 3.2 重点

本章的指导内容包括：

- 设置开发模式
- 建立一个简单的新“项目”
- 建立一个简单的新源文件
- 输入源文件代码
- 汇编源文件
- 运行你的程序
- 打开其他的 WINDOWS 窗口用于调试
- 建立监视窗口
- 保存监视窗口
- 设置断点

另外，在以后的章节中将会介绍对这些内容详细介绍，这里只是一个简要的介绍。

在本指导对 MPLAB IDE 集成开发环境的操作介绍之后，你应当：

- 熟悉 MPLAB IDE 集成开发环境桌面
- 建立一个新源文件的方法和进入一个针对 PIC16F84 的新项目里。
- 在模拟器里运行“创建 (built)”。
- 设置断点。
- 建立监视窗口
- 熟悉各种调试窗口

### 3.3 设置开发模式

前一章介绍了如何安装 MPLAB IDE 集成开发环境。现在要开始来设置应用功能。

MPLAB IDE 桌面（见图 3-1）包含以下主要部分：

1. 横跨桌面上部的菜单条。
2. 紧接着菜单条下是工具栏。
3. 然后是工作区域，这里有各种文件、窗口、会话窗口等。
4. 在底部有一个状态栏。

值得一提的是：状态栏里包含了当前系统设置情况。随后我们会详细介绍这些设置项的具体情况。现在让我们来看如何设置开发模式。

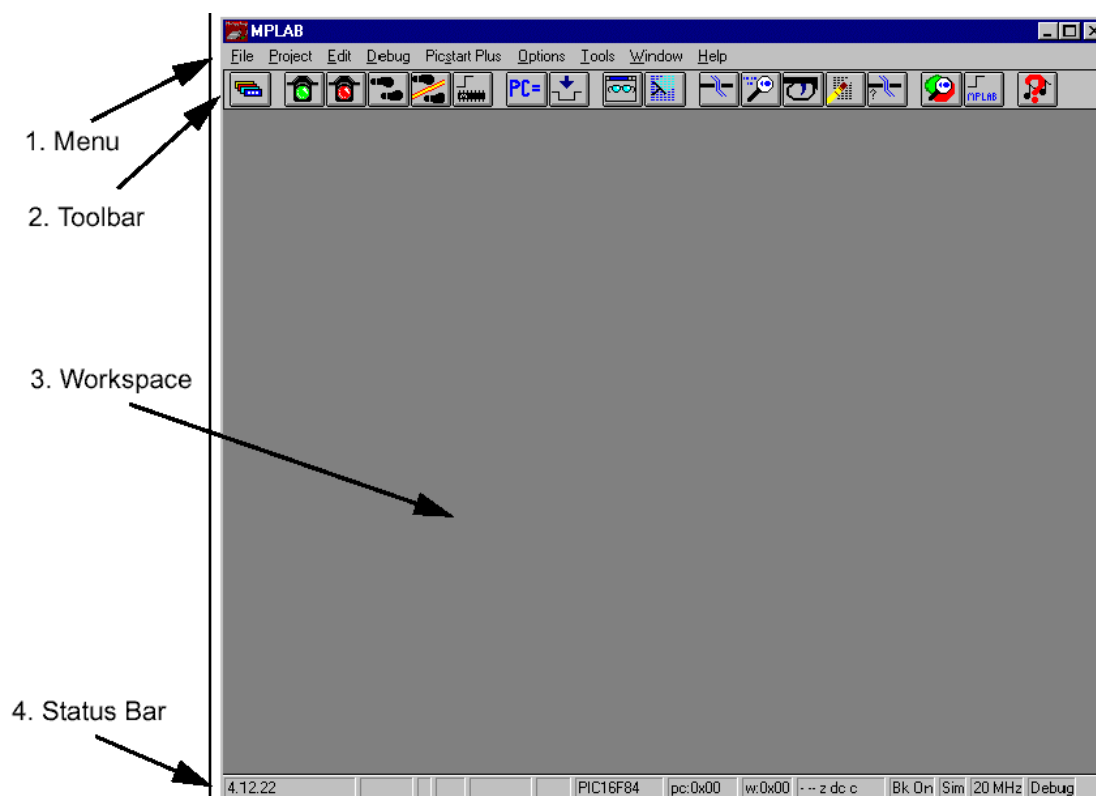


图 3-1: MPLAB IDE 开发环境桌面

开发模式工具设置的是一些可执行文件。本指导将选择使用 **MPLAB-SIM** 模拟器程序。假如你手头有仿真器，则可以随后切换到仿真操作模式。操作过程是相似的。“**Editor Only**（纯编辑模式）”不允许代码执行，假如你还没有安装模拟器，没有仿真器，刚要开始建立 **PICmicro** 微控制器的代码的情况下，这个方式是很有用的。

选择命令：***Options > Development Mode*** 菜单选项，点击 **Tools** 标签为“项目”选择相应的开发工具和处理器类型。

**MPLAB IDE** 集成开发环境是经常变化的，所以用户看到的环境可能和这里描述的稍微有所不同。选择 **MPLAB-SIM** 模拟器并从下拉菜单中选择模拟器支持的处理器类型：**PIC16F84**。再点击“**OK**”按钮。之后模拟器将初试化，你应当能在 **MPLAB IDE** 集成开发环境窗口下部的状态栏中看到“**PIC16F84**”和“**Sim**”的选择项。现在系统处于模拟状态下，被模拟的芯片是 **PIC16F84**。

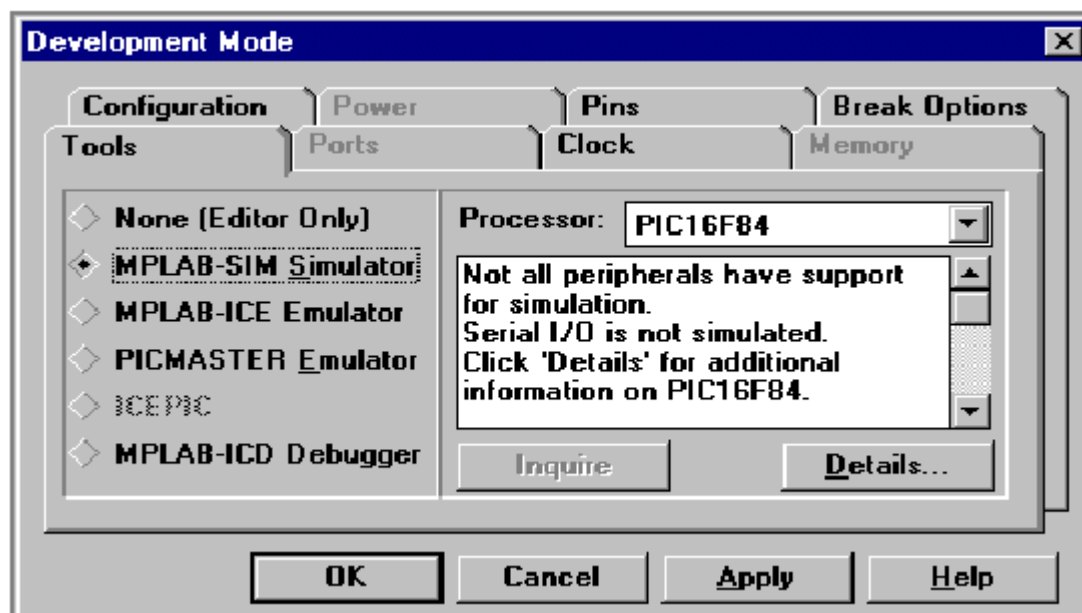


图 3-2: 开发模式工具会话窗口

### 3.4 建立一个简单的项目

模拟器使用的代码和写入到 PICmicro 微控制器中的可执行代码是一个相同的十六进制文件。为了使模拟器可以工作，用户必须首先建立一个源文件，然后对源文件成功地进行汇编。

汇编器产生的文件当中有一个十六进制文件。该文件含有扩展名 .HEX。在本指导中，该文件将被命名为 “tutor84.hex”。不用汇编器或 MPLAB IDE “项目” 的参与，该文件可以被直接载入到芯片烧写器里。该文件也可以直接载入到大多数的第三方烧写器中。

从菜单中选择命令 “**File > New**”，你将会看形如图 3-3 的会话窗口：

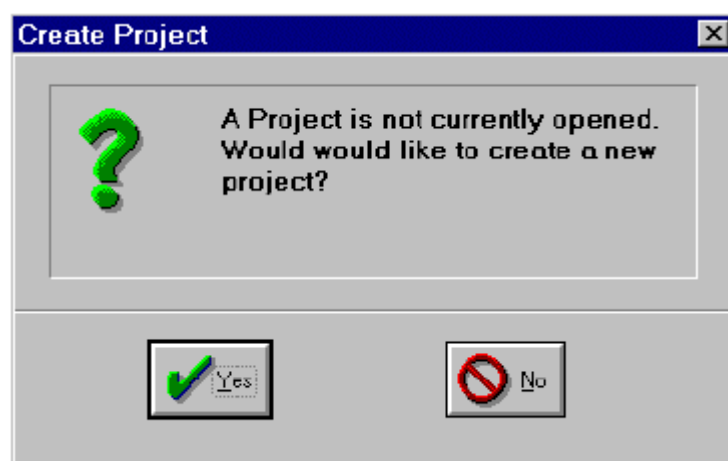


图 3-3: 建立“项目”会话窗口

点击“**Yes**”按钮，形如图 3-4 的一个标准 WINDOWS 浏览会话窗口将会出现。在这个会话窗口里，给出你想将“项目”存到哪个目录。

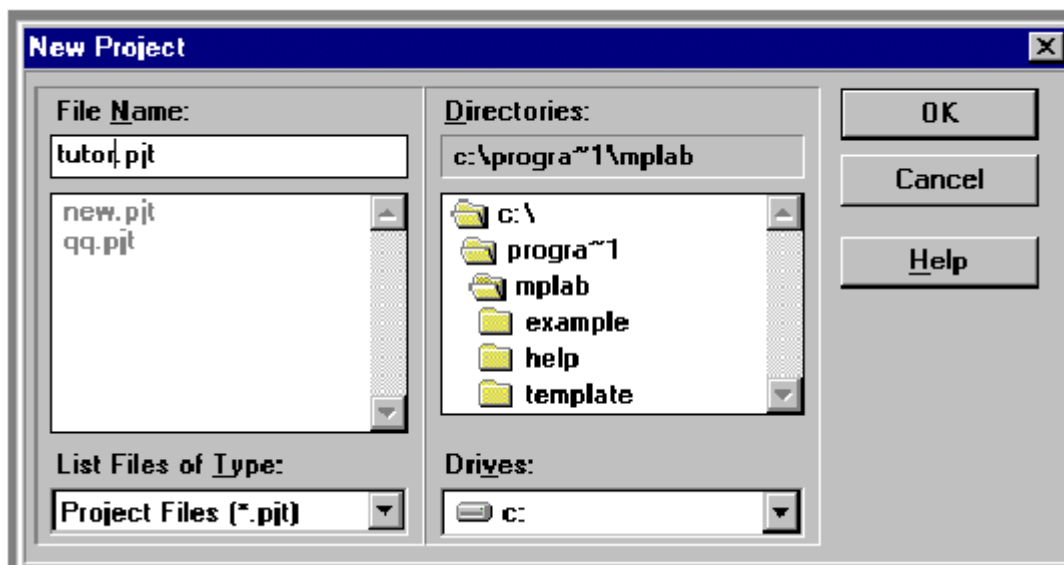


图 3-4: 建立新“项目”会话窗口

使用目录 `c:\Program Files\MPLAB` 并且建立“项目”文件，文件名为: `tutor84.pjt`。

“PJT”是一个标准的 MPLAB IDE “项目”文件扩展名，而本例中“项目”的文件名为 `tutor84`，对于本例中与 MPLAB IDE 相关的文件名都将使用这个文件名。

用鼠标点击“**OK**”按钮，这时候将进入到编辑“项目”会话窗口 (`Edit Project Dialog`)，见图 3-5 所示。

工作在 MPLAB IDE 集成开发环境下的模拟器 (`simulator`)、烧写器 (`programmers`)、仿真器 (`emulator`) 使用十六进制文件，该文件经由汇编、编译、链接源代码后生成。有若干开发工具可以生成十六进制文件，这些工具是每个“项目”的一部分。项目可以给你相当的灵活性，让你知道应用对象是如何建立、使用那些开发工具来生成 `.HEX` 文件。本指南并不详细介绍具体设置的细节，如果你需要的话可以使用**节点属性** (`Node Properties`) 进行设置。更复杂的“项目”设置见第 4 章介绍。

需要注意的是：编辑项目会话窗口里的目标文件已经为你填好了。它使用的开发模式是用户先前设置好的，默认使用 MICROCHIP 的开发语言工具。

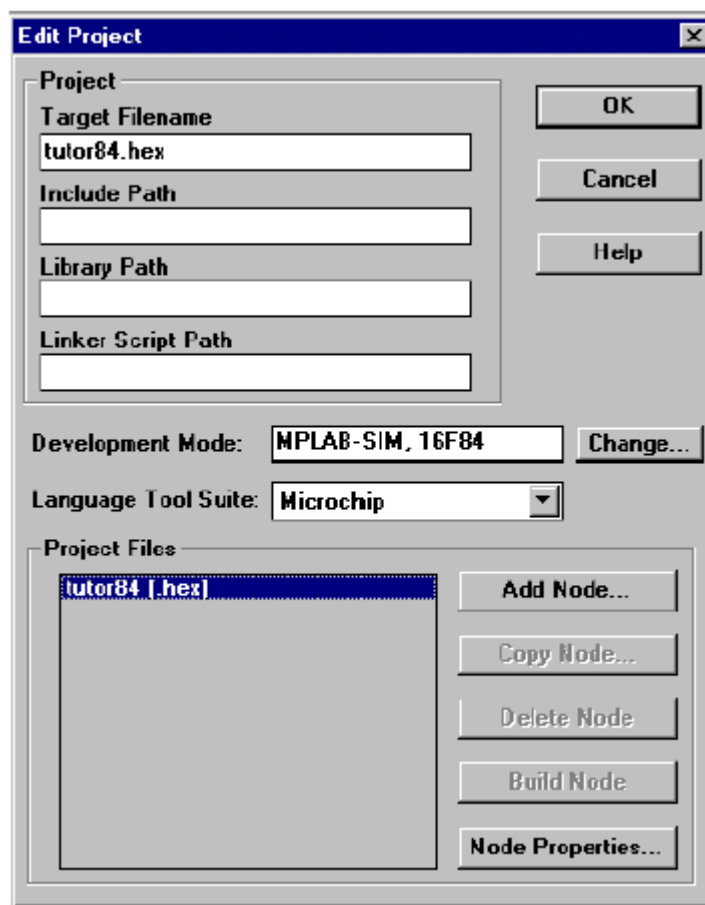


图 3-5: 编辑项目会话窗口-允许修改节点属性

另外，默认开发语言工具、路径、所有“项目”的节点都可以通过这种方法设置：选择命令 **Options > Environment Setup**，然后点击 **Projects** 属性标签。对于所有新“项目”的编辑“项目”会话窗口里都将出现这些默认设置。

在项目文件窗口（Project Files）里，你将会发现文件 tutor84.[hex]。如果我们将这个文件高亮的话，这时候节点属性（Node Properties）按钮将变为可使用（右虚变实）。

在我们做其他事情之前，需要讲一下 MPLAB IDE 是如何建立十六进制文件的。点击节点属性（Node Properties）按钮。节点属性会话窗口将会出现。见图 3-6 所示。



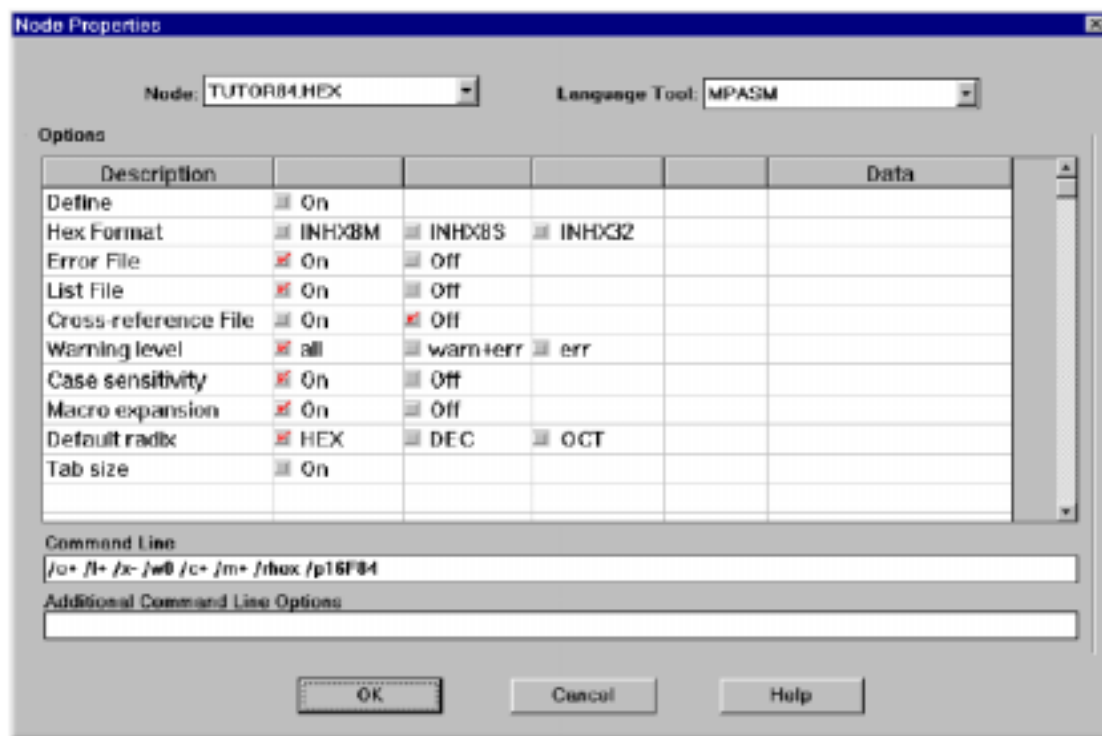


图 3-6：节点属性会话窗口

在这个会话窗口里显示了所有与开发语言工具（本例中是 MPASM）相关的默认设置。最简单的情形是项目中包含了一个由汇编而来的十六进制文件。这是节点属性会话窗口出现后看到的默认设置。

在这个会话窗口里，你可以看到很多的“√”符号和选择栏。每个“√”符号通常对应一个“开关”，如果某一个工具被选用，则命令行里就会有相应的设置命令。事实上这些工具的设置情况在窗口下方的命令行窗口里反映出来。当从 MPLAB IDE 集成开发环境下调用 MPASM 时，这些设置将会传递给 MPASM。

现在，任何其他地方都会用到这一默认设置。但随着你对软件应用越来越熟悉，就会发现需要修改其中一些设置。

点击“OK”按钮将会确认使用这些默认设置，你将会来到编辑项目会话窗口（Edit Project），激活增加节点（Add Node）按钮（见图 3-7）。

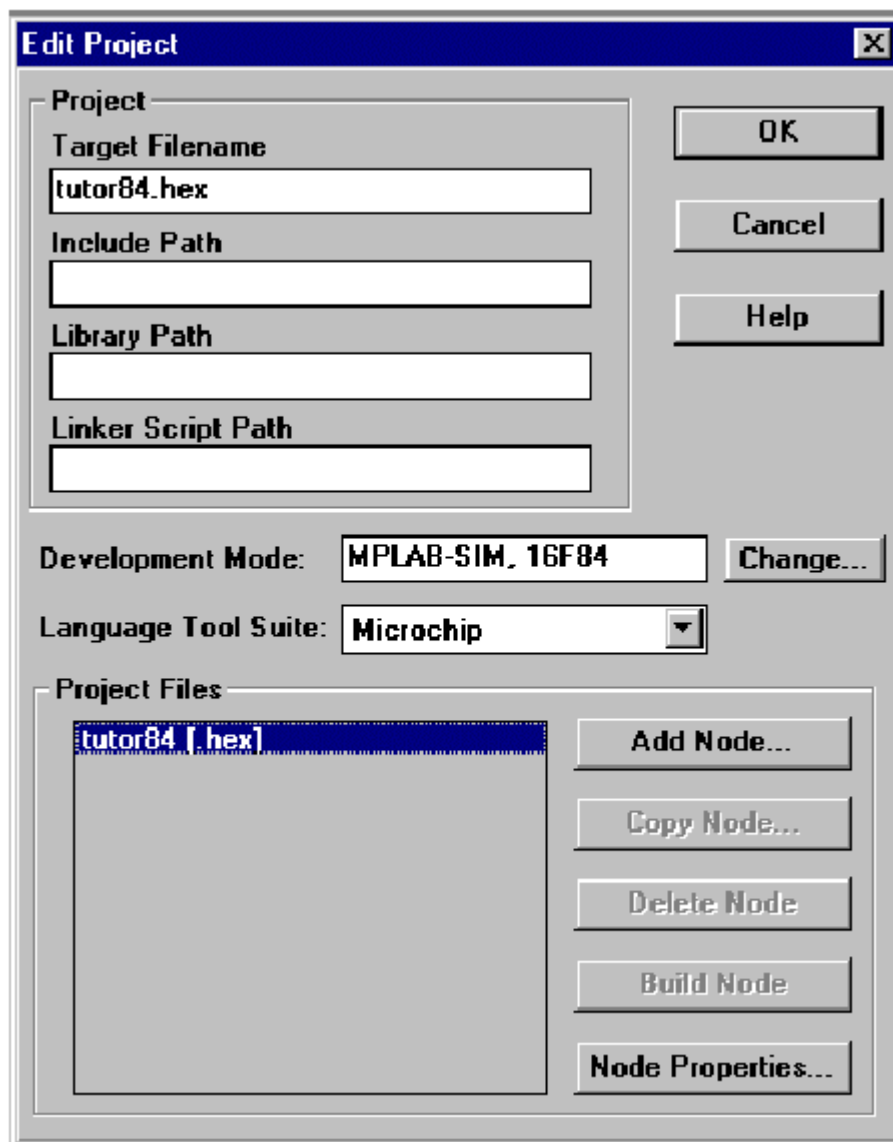


图 3-7: 编辑项目会话窗口-激活增加节点功能

点击“**Add Node**”你将会看到一个标准的 WINDOWS 浏览会话窗口（见图 3-8 所示），其工作目录将和“项目”的目录一样。键入文件名 `tutor84.asm`，再点击“**OK**”按钮。

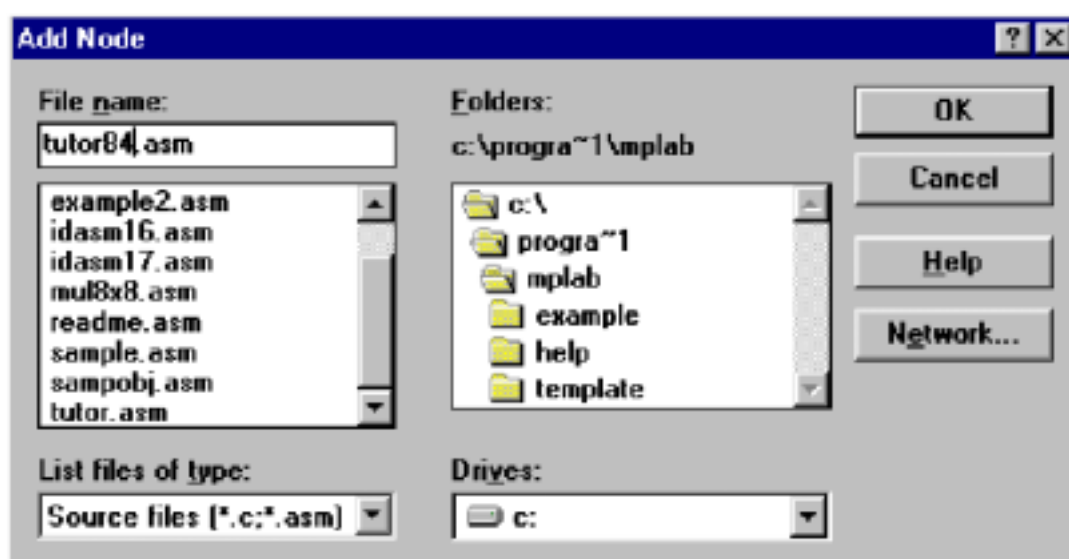


图 3-8: 增加节点会话窗口

这时候你将回到编辑项目（Edit Project）会话窗口，并且会看到文件名：tutor84.asm。紧接着 HEX 文件的下面，列出了“项目”的节点（见图 3-9 所示）。

点击“OK”后将回到 MPLAB IDE 集成开发环境的桌面，这时候将有一个已经打开而且还没有命名的源文名。

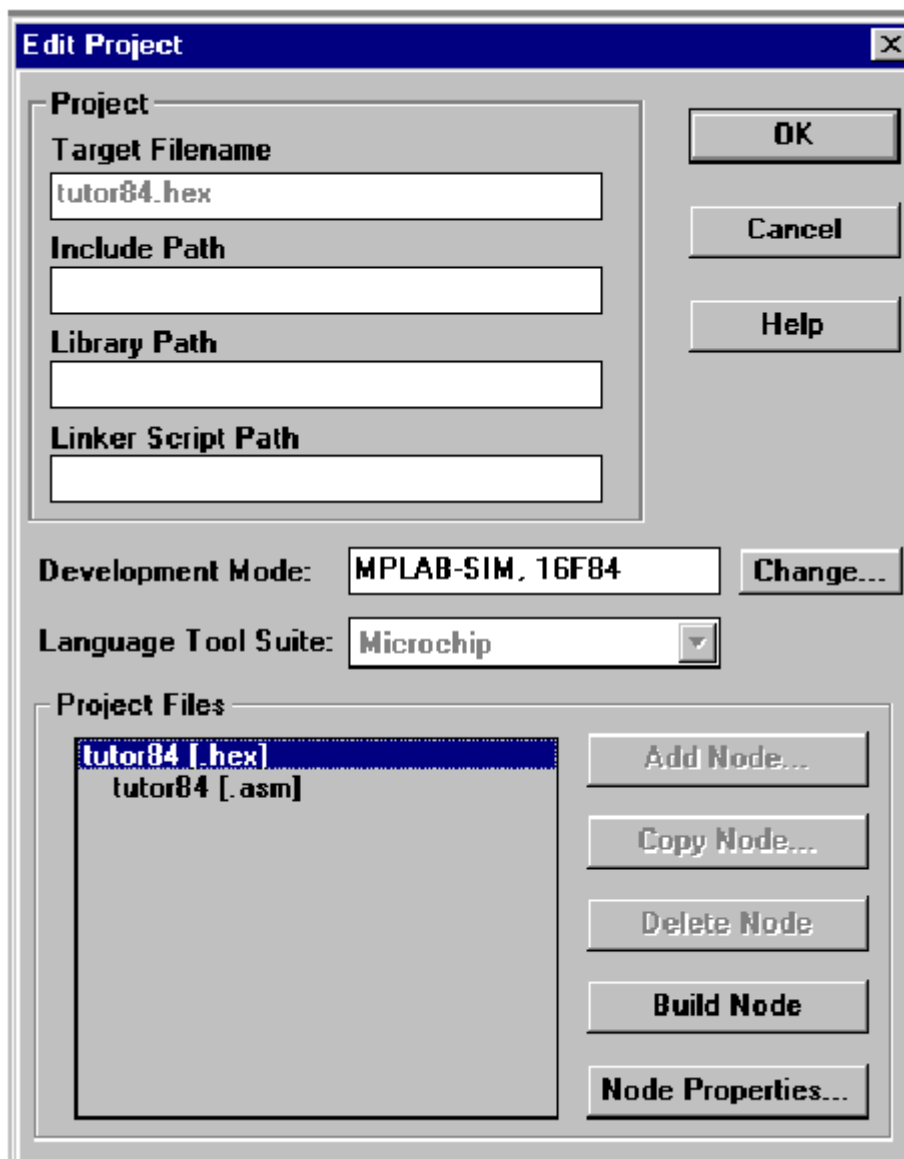


图 3-9: 编辑项目会话窗口 – “节点”已经增加完毕

### 3. 5 建立一个简单的源文件

在为你建立的文件窗口里的空白地方点击一下。可能这里有文件名为“Untitled”为的文件。这指示出该窗口的“中心内容”是什么。使用命令：**File > Save As...**菜单选项，将空文件存为 tutor84.asm。当标准浏览窗口打开时，你将会发现文件 tutor84.asm 位于当前“项目”的工作目录。这时候请输入文件名然后点击“OK”按钮。

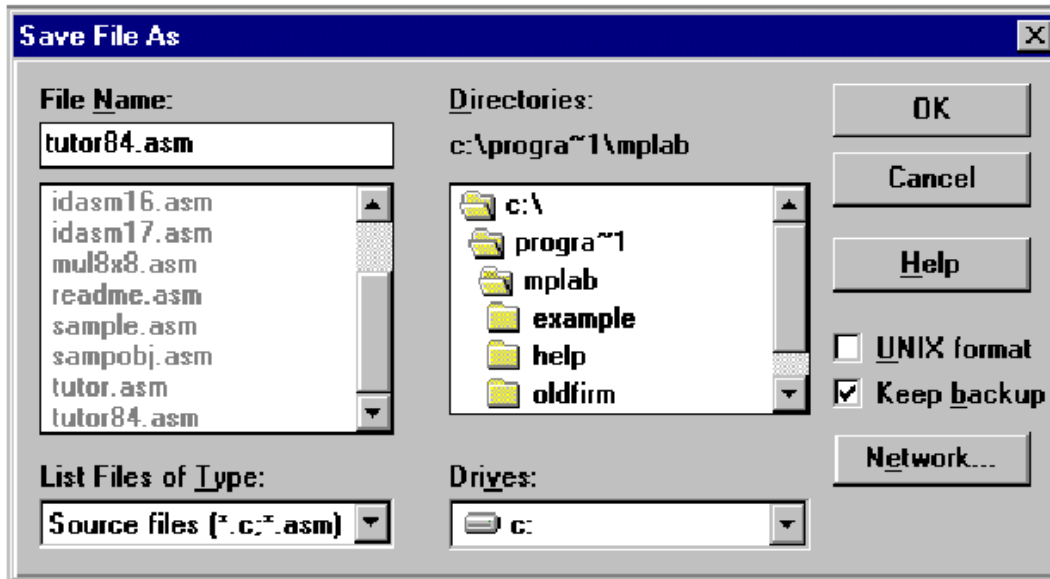


图 3-10：保存源文件

这时候你将会看到 MPLAB IDE 桌面和空的文件窗口，但文件窗口里的文件名已经有了一个新的名字。

在这种项目里，源文件名称和“项目”名称（在本指南中是 tutor84）必须相同。如果你改变了源文件名，你也必须改变“项目”名称，以便匹配。其他使用链接器的“项目”允许其输出文件和输入文件名不同。（见 4.6 节介绍了使用链接器建立“项目”的指导）

**注意：**对于只有一个源文件的“项目”，MPASM 将建立一个与源文件的文件名一样的 HEX 文件。项目文件名，HEX 文件名和源文件名必须相同。

### 3. 6 输入源文件代码

用鼠标将光定位于文件 tutor84.asm 窗口的开始部分，输入以下文字，一定要准确地书写在每一行。你可以不必输入分号后面注释内容。

```
list p=l6f84
include <p16F84.inc>

cl equ 0x0c ; 在地址 0x0c 处设置临时变量计数器 CL

    org 0x00 ; 在复位向量 0x00 处设置程序存储器基址
reset
    goto start ; 跳转到主程序开始处

    org 0x04 ; Set program memory base to beginning of user code
start
    movlw 0x09 ; 初始化计数器为大于0的任意值
    movwf cl ; Store value in temp variable a defined above
loop
    incfsz cl,F ; 计数器增加，结果放到文件寄存器里
    goto loop ; 循环，知道计数器溢出

    goto bug ; 当计数器溢出时，跳转到开始位置，重新初始化
end
```

这是一个非常简单的程序，程序里不断将计数起做加 1 的操作，当计数器恢复到 0 的时候就将计数值恢复为预先设置的数值。

所有标号开始于第一列，最后一行用“END”指示符结束。更多关于指示符的信息请参考《MPASM 及 MPLINK 和 MPLAB 用户指南》。在 PICMICRO 系列 MCU 的数据手册里包含了所有的指令集和相应的应用范例。

使用菜单指令 “**File > Save**” 保存文件。

### 3.7 对源文件进行汇编

对源文件进行汇编有几种方式都可以完成。这里要介绍的方法是使用菜单命令 “**Project > Build All**” 来完成。执行过程将是在后台运行 MPASM 汇编器，使用的参数是“项目”里的默认值。一旦汇编过程结束，则“创建结果 (Build Results)” 窗口将会出现（见图 3.11）。

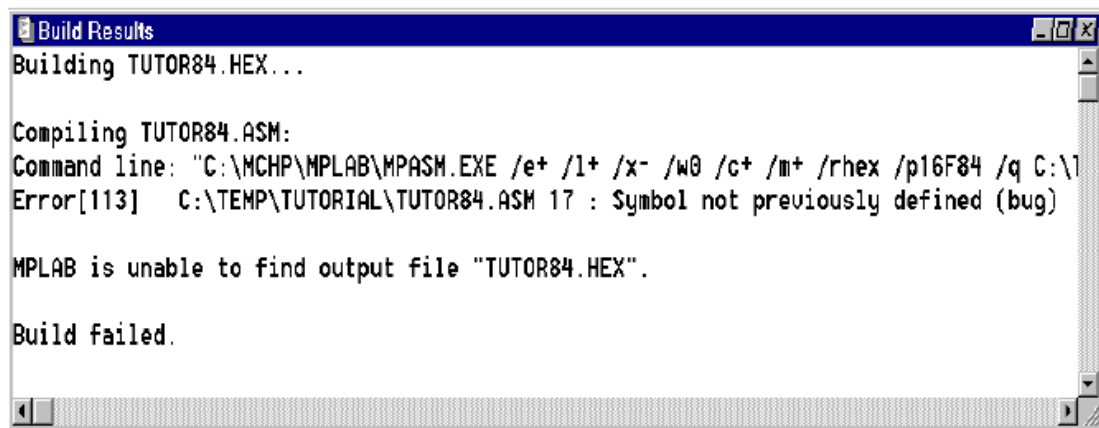


图 3-11：创建结果窗口 – 创建失败

假如你按照 3.6 节介绍的程序输入的话，已经输入了至少一个事先安排的错误。程序里最后一个“goto”语句使程序指向一个并不存在的标号“bug”。因为该标号没有事先定义，汇编器就会报告一个错误。你也可能会看到其他的错误报告。

使用鼠标，在一条错误信息上双击。这时候光标会自动跳到源文件里有错误的那条语句上。把“bug”改为“start”。可以使用创建结果 (Build Results) 窗口来帮助定位错误位置，并修改你程序里的其他错误。修改结束后，使用命令 “**Project > Build All**” 来重新汇编源代码。此过程将重复好几次才会编译成功。

注意：无论何时，重新创建了“项目”后，所有的源文件都要存盘。

当你修改了源文件里的所有的错误以后，创建结果窗口里将会显示“创建成功 (Build completed successfully)”（图 3-12 所示）。现在你便有了一个完整的“项目”，可以用模拟器进行调试执行。

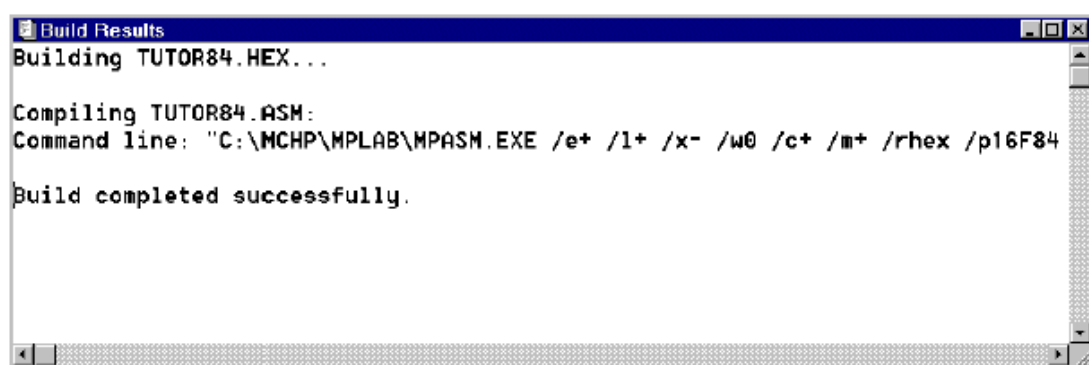


图 3-12: 创建结果窗口 – 创建成功

### 3. 8 运行你的程序

使用命令 “**Debug > Run > Reset**” 来初始化系统。程序指针将会指向零，也就是 PIC16F84 的复位向量。在源代码里面这一地址上的语句会用一暗背景条高亮。而且你会发现 MPLAB IDE 桌面下面的状态栏里的 PC 指向了 “0x00”。

使用命令 “**Debug > Run > Step**”，将会导致程序指针加一，指向下一条指令的位置。高亮暗条将会跟随源代码，状态栏 (status bar) 里的程序指针将会指向 “pc:0x04”

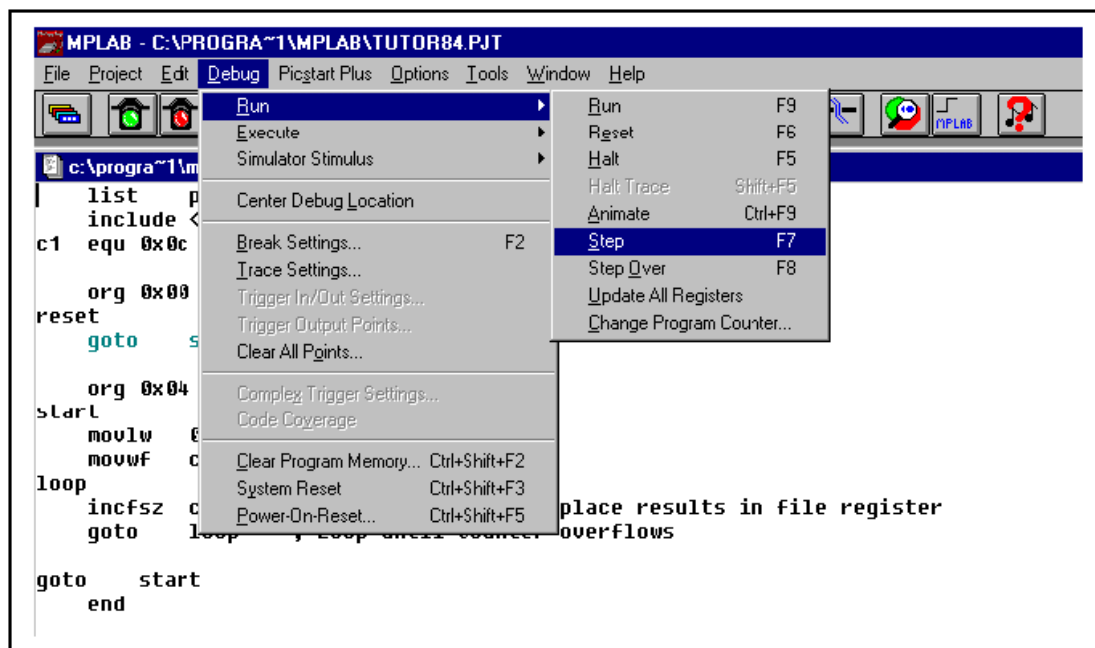


图 3-13: Debug &gt; Run &gt; Step 菜单项

当你执行 “**Debug > Run > Step**” 命令的时候，你会发现在菜单的右侧有文本 “<F7>”。这表示你键盘上的 “第 7 功能键”。许多 MPLAB-IDE 集成开发环境的功能都有 “快捷键” 定义。这些快捷键和执行菜单命令的效果是一样的。按<F7>键几次，观察程序指针的变化和高亮暗条在程序上的移动。

执行命令 “**Debug > Run > Run**” 或敲功能键<F9>，则程序从当前程序指正所指的位置开始执行。状态栏将会改变颜色，指示出程序正在执行。其他位置的状态栏里的信息将不会改变，一直到程序执行结束。

执行命令 “***Debug > Run > Halt***” 或敲功能键<F5>。状态栏的颜色将会改变为原来的颜色，当前程序指针和其他状态信息也将会更新。

另外一种执行程序功能的方法是使用桌面上部的工具按钮。你可以在窗口下部的状态栏里看到工具按钮的功能说明。左边的按钮是一个标准的“改变工具栏 (**Change Tool bar**)”按钮，该按钮允许你滚动现有的工具按钮，以便选择。工具按钮可以用户自定义（见 7.8.5.1.4 节所示）。在调试工具按钮上，绿色表示<F9>（运行），红色表示<F5>（暂停）。

### 3. 9 打开其他窗口帮助调试

在 MPLAB-IDE 集成开发环境里有很多方法来观察你的程序及其执行过程。例如：有一个程序用来执行将一个临时计数器加一的操作，可是你如何才知道这个操作真的发生了呢？一个办法是打开一个检查文件寄存器的窗口。具体办法是：通过菜单命令执行 ***Window > File Registers*** 操作，一个包含了 PIC16F84 的所有文件寄存器或 RAM 的小窗口将会出现。

敲功能键<F7>（执行单步操作）几次，同时观察窗口里的文件寄存器的变化情况。我们把计数器变量放置在地址 0x0C。随着临时计数器加一，其过程将会反映到文件寄存器窗口里数值的变化。当文件寄存器的数值变化的时候其颜色将会改变，以便可以方便地观察到该寄存器已经发生了变化。然而，在有些非常复杂的程序里，可能同时有很多的寄存器数值发生变化，这样一来，可能就不那么容易地观察到一两个寄存器数值的变化了。为解决这一问题，可以使用观察窗口（Watch window）。

### 3. 10 使用观察窗口

MPLAB IDE 允许开设一个观察窗口，来监视一些文件寄存器的变化情况。

#### 3. 10. 1 建立一个观察窗口

建立一个观察窗口可以用以下方法来完成：通过菜单命令执行 ***Window > Watch Window > New Watch Window***。如果你已经建立了观察窗口，而且已经存盘，选择菜单命令：***Window > Watch Window > Load Watch Window***，选择一个观察窗口文件并点击 OK 按钮，或者双击想要打开的观察窗口文件。

这时候增加观察符号（Add Watch Symbol）会话窗口将会出现。（见图 3.14）



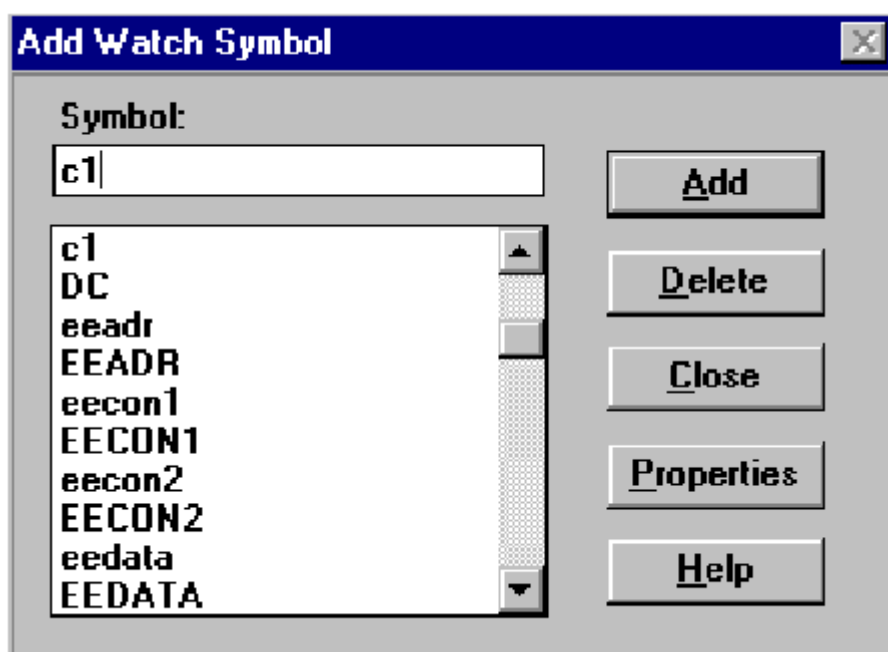


图 3-14: 增加观察符号会话窗口

在符号名称窗口里输入“C1”，列表将会滚动到符号“C1”上。使该符号高亮后点击“ADD”按钮，然后点击“CLOSE”按钮。这样一来监视窗口将会在你的 MPLAB-IDE 开发环境的桌面上出现。（见图画 3-15 所示）。这里显示了临时变量“C1”的当前数值。

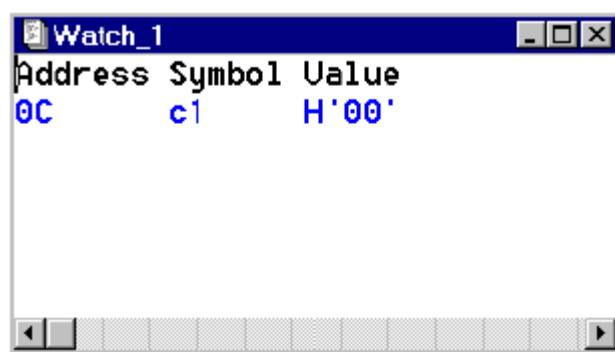


图 3-15: 监视窗口

你可以带行号，也可以不带行号显示监视窗口的内容。要想改变设置，可以从监视窗口的系统菜单里选择命令：“Toggle Line Numbers”。

按功能键“<F7>”来使程序单步执行几次。观察随着计数器值的增加的时候，监视窗口里数值的变化。假如你让文件寄存器窗口打开，其内容也会随着变化。

### 3. 10. 2 保持监视窗口参数的设置

你可以保持监视窗口和它的设置参数。方法是：从 MPLAB IDE 菜单里选择命令：“Window > Watch Window > Save Active Watch”，或者从

监视窗口的系统菜单里选择“Save Watch”。（注意：系统菜单按钮位于监视窗口的右上方，点击该按钮一次将会弹出一个下拉菜单）选择一个名称，然后点击“OK”。

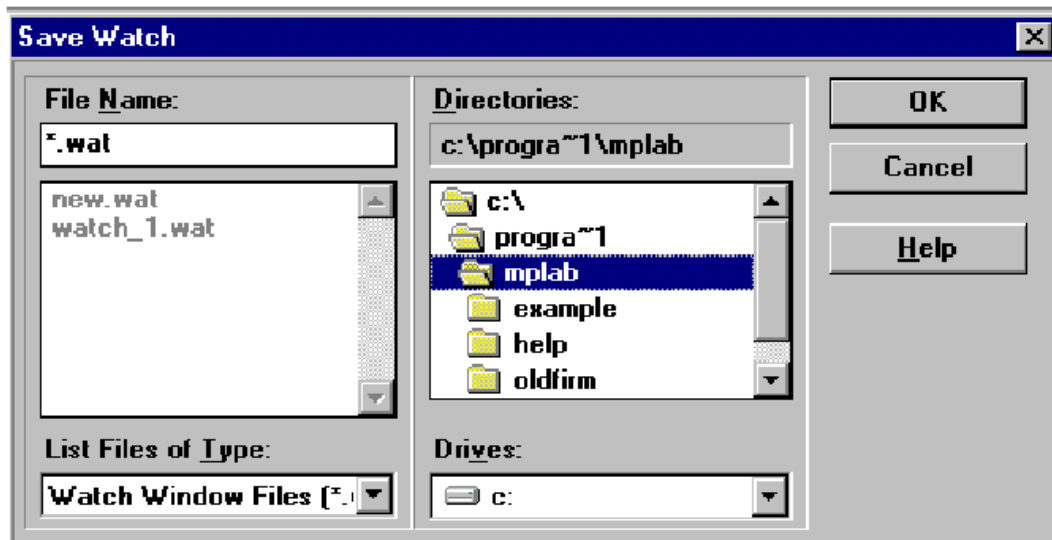


图 3-16: 保存监视窗口信息会话窗口

这样一来，窗口的打开状态和位于窗口的位置将会和“项目”一起保存起来，下次打开你的“项目”的时候，你的监视窗口将会出现。

### 3. 10. 3 编辑监视窗口

你也可以在建立监视窗口以后去编辑监视窗口。

使用监视窗口里的“**Window > Watch Window**”子菜单或者系统菜单都可以编辑监视窗口的信息。

给监视窗口里添加一个符号：从 MPLAB IDE 的菜单里选择命令：“**Window > Watch Window > Add to Active Watch**”或者使用监视窗口里的命令“**Add Watch**”。

从监视窗口里删除一个符号：点击监视窗口里的一个符号，然后从系统菜单里选择命令：“**Delete Watch**”。即可删除一个符号。

改变符号的显示模式：从 MPLAB IDE 的菜单里选择命令：“**Window > Watch Window > Edit Active Watch**”，或者使用监视窗口里的命令“**Edit Watch**”，然后点击“**Properties**”属性按钮。会话属性允许你选择监视窗口里的字符的模式，大小，字节顺序和显示位。

### 3. 1 1 设置断点

按功能键“<F5> (**Debug > Run > Halt**)”确认被仿真的处理器停止下来。点击源代码窗口里的以标号“start”开始的语句：movlw 0x09，点击鼠标右键，一个快捷菜单将会出现（见图 3-17 所示）。



图 3-17: 点击鼠标右键弹出菜单

选择命令 “***Break Point(s)***” 菜单项，这时候菜单将会随之消失，同时原来光标所在的地方的指令将会改变颜色。这意味着这条指令所在的地方被设置为断点。

按功能键 “<F6>” 或执行菜单命令 “***Debug > Run > Reset***” 可以将系统复位。然后按功能键 “<F9>”，可以运行系统。这时候程序将运行，当执行完用户所设置断点处的指令后，程序将会暂停下来。监视窗口里的 “C1” 的值或文件寄存器窗口（假如已经打开的话）将会显示复位状态 “0”；单步执行一步，将会执行一条指令，同时 “C1” 的数值将会变为 “”。按功能键 “<F9>” 几次，同时观察状态栏在运行的同时改变颜色，当处理器停止后颜色又将再次改变。

注意：假如程序在断点处没有停止，选择命令 “***Options > Development Mode***” 然后点击 “***Break Options***”。确信选择了 “Global Break Enable（允许全局断点）”（选中标记）

### 3. 1 2 总结

被指南已经给你示范了如何：

- 设置一个新“项目”
- 为“项目”建立和输入源文件
- 汇编源代码
- 用仿真器运行代码
- 设置断点并用单步执行用户的代码
- 观察你的代码中变量的变化情况

假如本提纲中介绍的内容你已经完全没有问题，你就可以开始下一节中关于 MPLAB IDE 的更多内容。

一些提醒和建议：

**断点：**在源文件（这里源文件是 `tutor84.asm`）窗口里使用菜单命令 “***Window > Program Memory***” 或者用窗口菜单命令 “***Window > Absolute Listing***” 来设置断点。

**源文件：**使用命令 “***Window > Project Window***” 显示出你的源文件列表。在某一个源文件上双击鼠标就会进入对该文件的编辑状态。

**MPASM 错误：**假如 MPASM 给出一个错误，可以双击这个错误，进入到出现这个错误的源程序语句上。假如有多个错误，记住先点击第一个错误。因

为通常一个错误可能引起后面多个错误。更正一个错误就可能更正后面的好几个错误。

**配置位和处理器模式：**源文件里的配置位并不能为仿真器（或者模拟器）提供处理器模式。比如，看门狗允许配置位（Watch Dog Timer Enable）必须处于允许状态时候才能使一个芯片被编程后看门狗(WDT)处于打开状态。你必须选择命令“**Options > Development Mode**”，然后点击“**Configuration（配置）**”选择盒，为仿真器（或者模拟器）打开看门狗定时器。这种方法可以使你不用修改源文件就可以使门狗定时器打开或者关闭。也可以使用命令“**Options > Development Mode**”来配置处理器模式。尽管你可在 MPASM 或者 MPLAB-CXX 环境里面设置这些配置位，MPLAB IDE 并不自动改变模式。

**选择项：**选择命令“**Options > Environment Setup**”并点击“**General**”来完成以下操作：

- 改变显示字体和字体的大小
- 定位工具栏的位置，可以处于屏幕的一边，也可以位于屏幕的下方
- 修改工具栏
- 改变标号的字符长度

在你关闭会话窗口之前，点击“**Key Mappings（功能键映射）**”，将欧洲键盘与 MPLAB IDE 功能键和 ASCII 代码对应起来。

**映射文件：**选择命令“**Project > Edit Project**”改变 MPLAB 的节点属性，从而产生一个名为“tutor84.map”的映射文件。在你创建完“项目”后，请查看文件“tutor84.map.”以观察创建信息。

**灰暗菜单项：**假如你发现菜单变灰暗了的话，检查一下你是否处于“纯编辑”状态下。假如你确认一切处于正确状态的话，可以实验推出 MPLAB IDE，然后重新启动软件。

## 第4章 MPLAB IDE “项目” 指南

### 4.1 介绍

本章将详细介绍如何使用 MPLAB IDE “项目”。如果你学习完第 3 章的指南，你可以跳过本章回到第 3 章学习更多关于“项目”的知识。

MPLAB IDE v3.40 以及以后的版本，其“项目”都支持多个文件。MPLAB IDE v3.31 及更早版本所建立的“项目”在新版本的 MPLAB IDE 开发环境中打开的时候将会自动地被转换成新版本格式。一旦一个“项目”被转换它将再也不能用旧版本的 MPLAB IDE 打开了。

### 4.2 重点

本章里你将会学会 MPLAB IDE “项目”的下列功能：

- MPLAB IDE “项目”概述
- 建立一个只有一个 MPASM 源文件的“项目”
- 不用建立“项目”，编译一个 MPASM 源文件
- 使用 MPLINK 链接器创建一个有多个 MPASM 源文件的“项目”
- 使用 Hi-Tech PIC C 来创建一个“项目”

为了实现这些功能特性，你将会用到 MPLAB IDE 的如下特性：

- 安装开发语言工具
- 新“项目”
- 设置“项目”节点属性
- 创建“项目”
- “项目”窗口

### 4.3 MPLAB IDE“项目”概述

MPLAB IDE 的“项目”是由一系列的文件组成，这些文件用来完成一个特定的功能。他们同时需要相应的创建工具。一个“项目”由一个“项目”节点和一个或多个源文件节点组成。典型的源文件节点是汇编文件、C 源文件、预编译的目标文件、库文件和链接器描述文件组成。作为主要的源文件，“项目”通常被放在相同的目录里面。

在本 MPLAB IDE “项目”里，C 语言源文件 `main.c` 和 MPLAB CXX 编译器相关联。MPLAB IDE 将利用该信息来生成一个目标文件 (`MAIN.O`)，用来输入到链接器 (MPLINK)。更多关于编译器的使用可以参见《MPLAB-CXX 用户指南》(文档资料号 DSXXXXX)。

汇编语言源文件 (`prog.asm`) 也和汇编器 MPASM 相关联。MPLAB IDE 将利用该信息来生成一个目标文件 (`PROG.O`)，用来输入到链接器 (MPLINK)。更多关于汇编器的使用可以参见《MPASM 及 MPLAB 和 MPLIB 用户指南》(文档资料号 DS33014)。

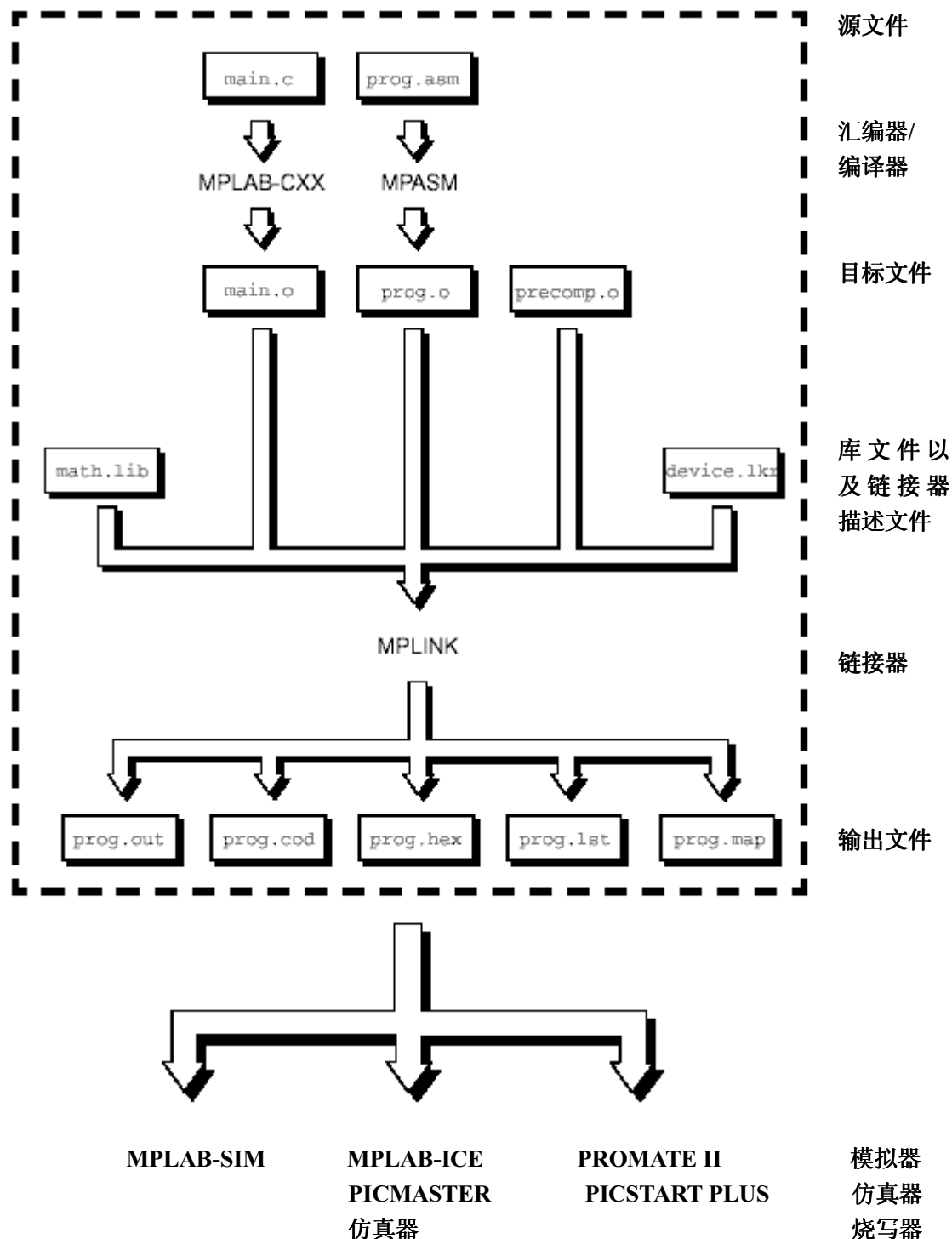


图 4-1: “项目”内部元件之间的关系

另外, 预先编译的目标文件 (`precomp.o`) 也可以包含在一个“项目”中, 他不需要关联的工具。一个“项目”通常要求预先编译目标文件的形式有:

- 启动代码
- 初始化代码
- 中断服务例程
- 寄存器定义

预编译目标文件通常依赖于芯片类型 和/或存储器结构。更多关于 MICROCHIP 预编译目标文件的信息，参考《MPLAB-CXX 用户指南》（文档资料号 DS51224）。

一些库文件 (`math.lib`) 随着编译器一起提供，其他的库文件可以用库管理工具 MPLIB 在“项目”外独立生成。独立关于库文件管理工具的使用请参考《MPASM 及 MPLAB 和 MPLIB 用户指南》（文档资料号 DS33014）。关于 MICROCHIP 现有库文件信息，参考《MPLAB-CXX 用户指南》（文档资料号 DS51224）。

目标文件和库文件、链接器描述文件 (`device.lkr`) 通过链接器来生成“项目”输出文件。关于链接器描述文件和链接器的更多信息请参考《MPASM 及 MPLAB 和 MPLIB 用户指南》（文档资料号 DS33014）。

- **COFF 文件(.out):** MPLINK 使用的中间文件，生成代码文件、十六进制文件、列表文件。
- **代码文件 (.cod):** MPLAB IDE 使用的调试文件。
- **列表文件 (.lst):** 初始源文件代码，与最终的机器代码一一对应。
- **映射文件(.map):** 显示链接以后的存储器布局。指示出已经被使用的和还没有使用的存储器。

这里描述的工具是 MICROCHIP 所提供的所有开发工具。但是，还有很多的第三方开发工具可以和 MPLAB IDE 一起工作。关于这些第三方的开发工具，可以参考《第三方工具指南》（DS00104）。

#### 4. 4 建立一个只有一个 MPASM 源文件的“项目”

按照以下步骤建立一个只含有一个源文件的“项目”，或者早期的“项目” (MPLAB IDE v3.31 或更早版本)，这种“项目”里只有一个源文件，用 `#include` 命令来包含其他文件。

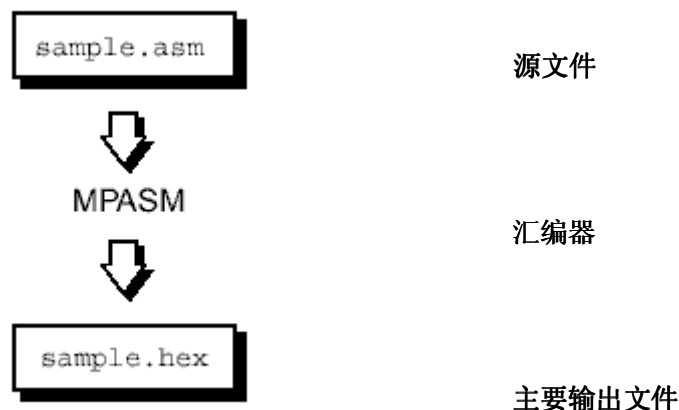


图 4-2: 只有一个 MPASM 源文件的项目内部关系

#### 4. 4. 1 设置开发模式

为一个应用建立相应的开发模式。选择命令“***Options > Development Mode***”并且点击“**Tools**”条。在本指南里，选择MPLAB-SIM模拟器，并且选择PIC16F84控制器（MCU），然后点击“**OK**”。

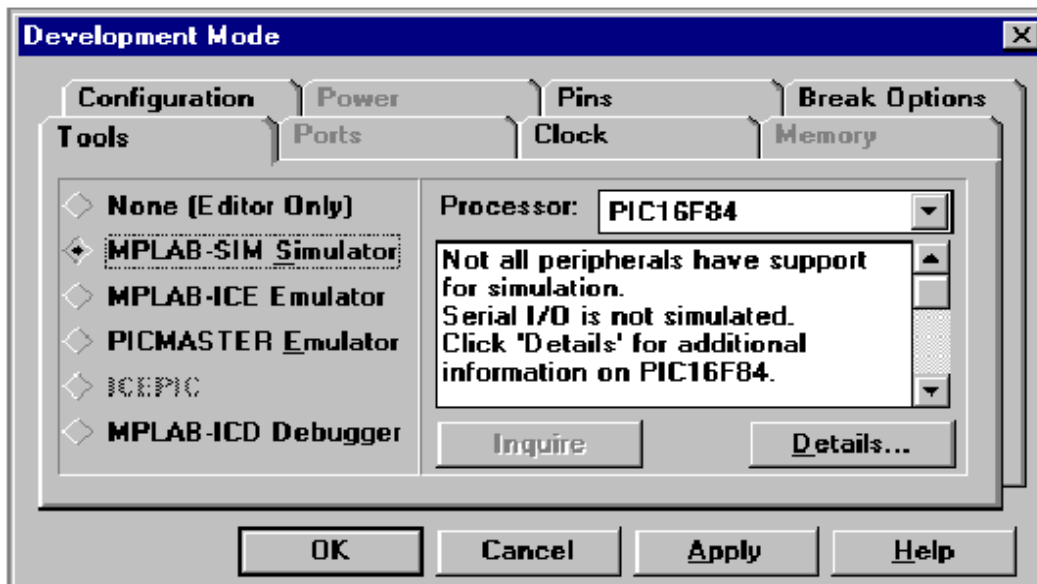


图4-3：开发模式会话窗口

#### 4. 4. 2 新“项目”

选择命令“***Project > New Project***”，为新“项目”选择一个目录，然后键入“项目”名称。在被指导里，使用/Program Files/MPLAB 目录，并且使用 SAMPLE.PJT 作为项目名称。

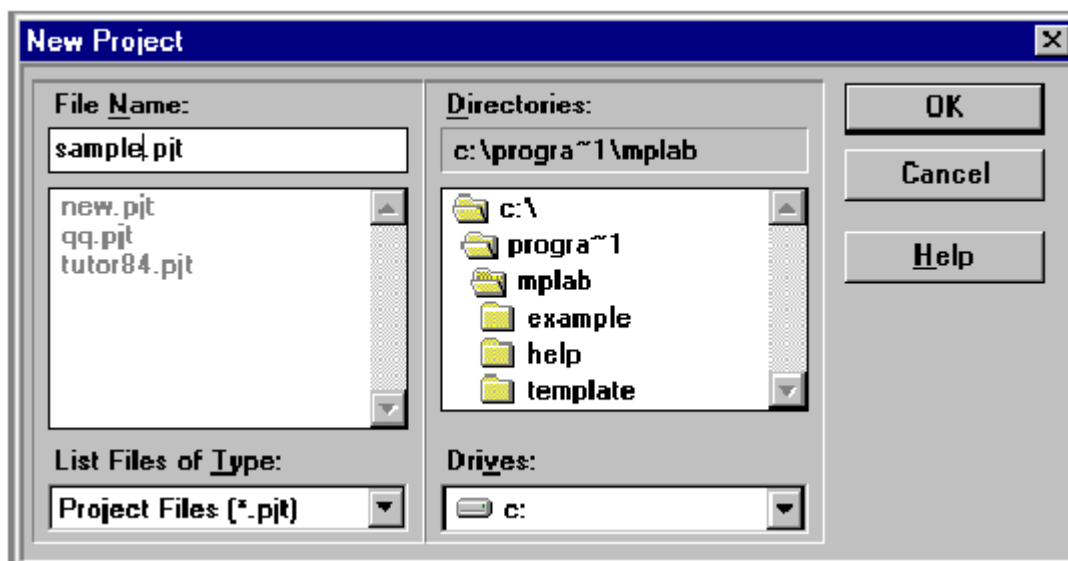


图 4-4：建立新“项目”会话窗口-sample.pjt



#### 4. 4. 3 “项目”会话窗口

点击完“OK”后，你可以看到编辑“项目”会话窗口：

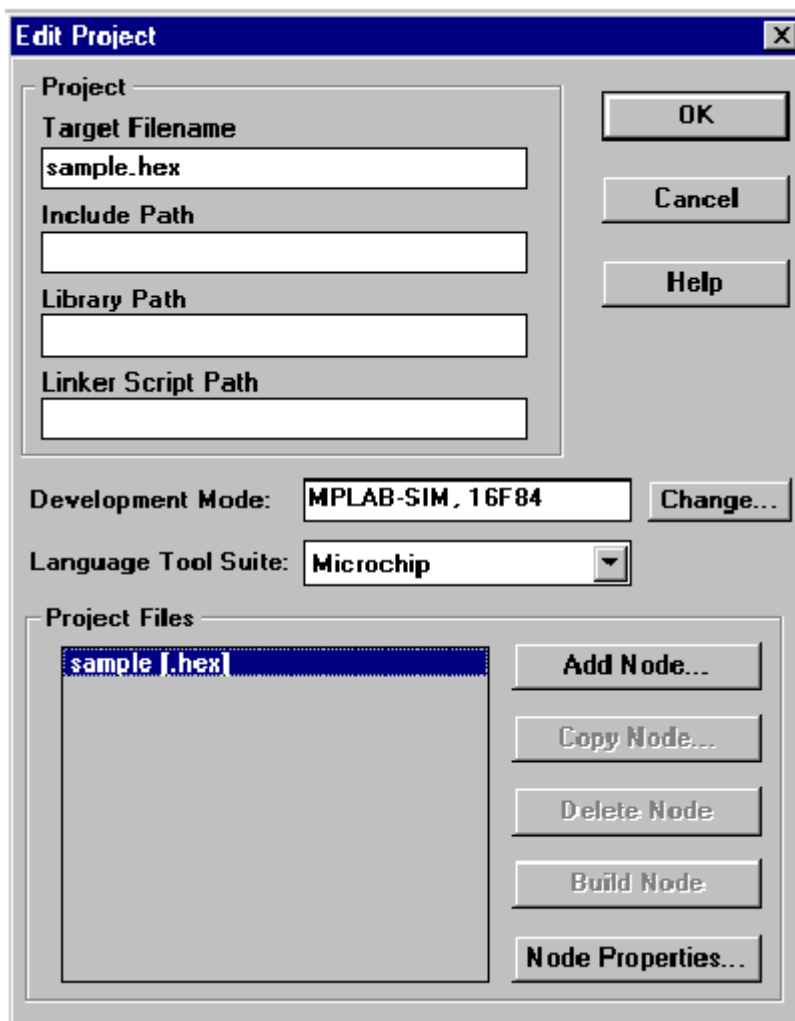


图 4-5: 编辑“项目”会话窗口

#### 4. 4. 4 设置“节点”属性

在“项目”文件窗口里选择文件名“sample.hex”然后点击“**Node Properties**（节点属性）”按钮。

节点属性会话窗口显示了开发工具的各种命令行开关。“选择盒”里有“V”符号的表示工具的默认设置。在本指导里，这些设置不必改变。关于这些命令行开关的使用，请参考《MPASM 及 MPLAB 和 MPLIB 用户指南》（文档资料号 DS33014）。

点击“**OK**”返回到编辑“项目”会话窗口。

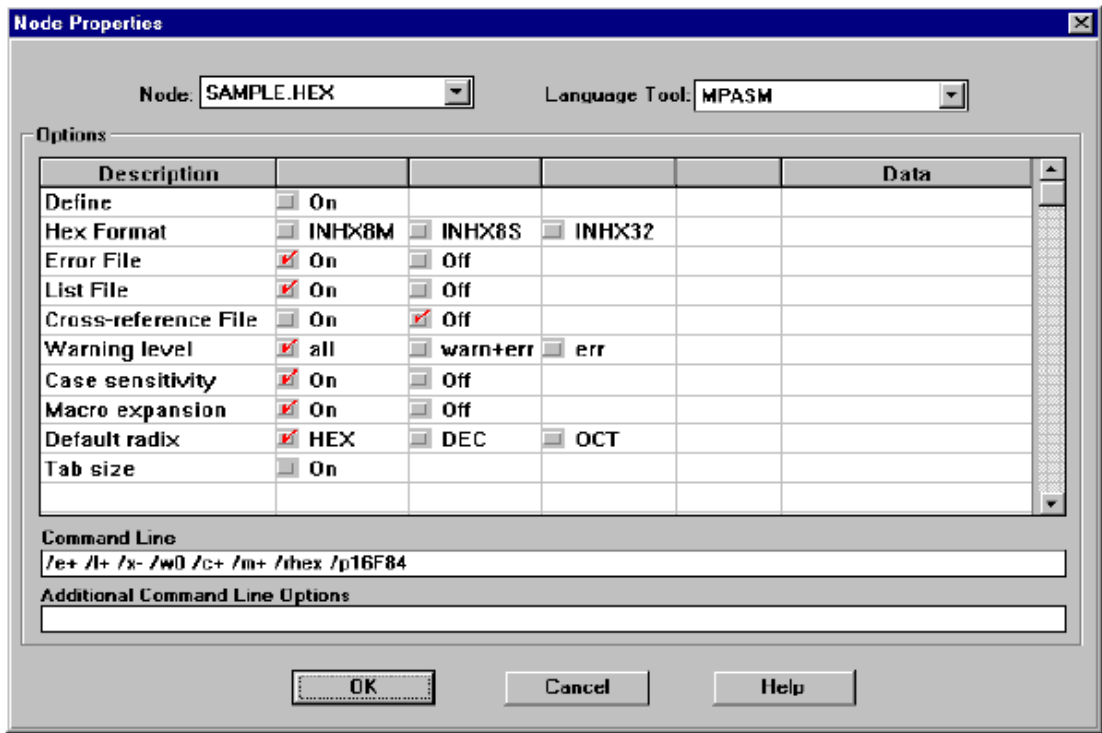


图 4-6：节点属性会话窗口

4. 4. 5 增加“节点”

在编辑项目会话窗口里点击“Add Node（增加接点）”本指导里使用文件 sample.asm。这是你点击“Add Node（增加接点）”后弹出的浏览窗口。

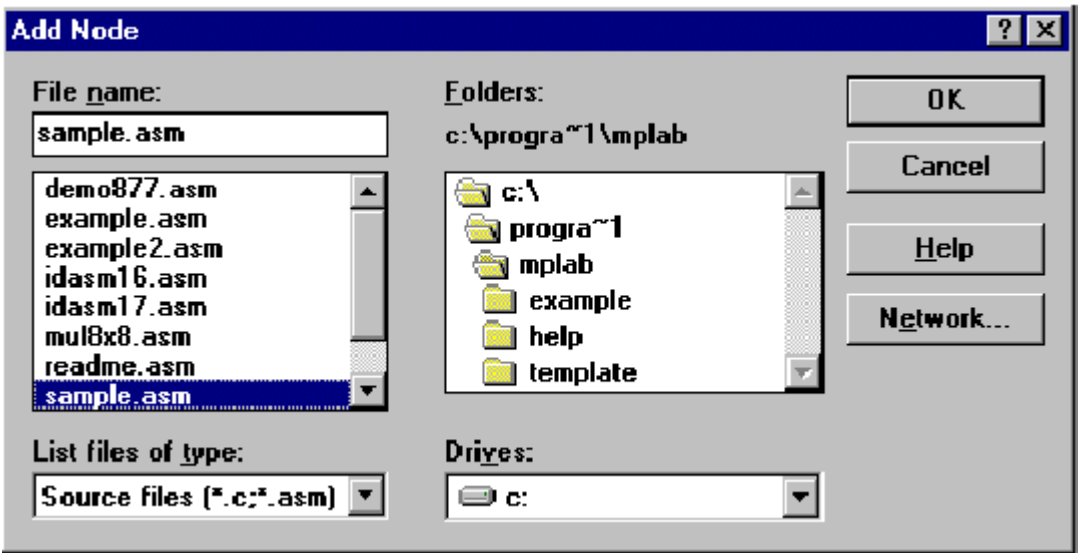


图 4-7：增加节点会话窗口

MPASM 通常产生一个 “.HEX” 十六进制文件。其文件名和源文件（.ASM）的文件名是一样的。当“项目”被创建（built）的时候，“项目”管理器将产生一个 “sample.hex” 文件。

项目会话窗口形如下图所示：

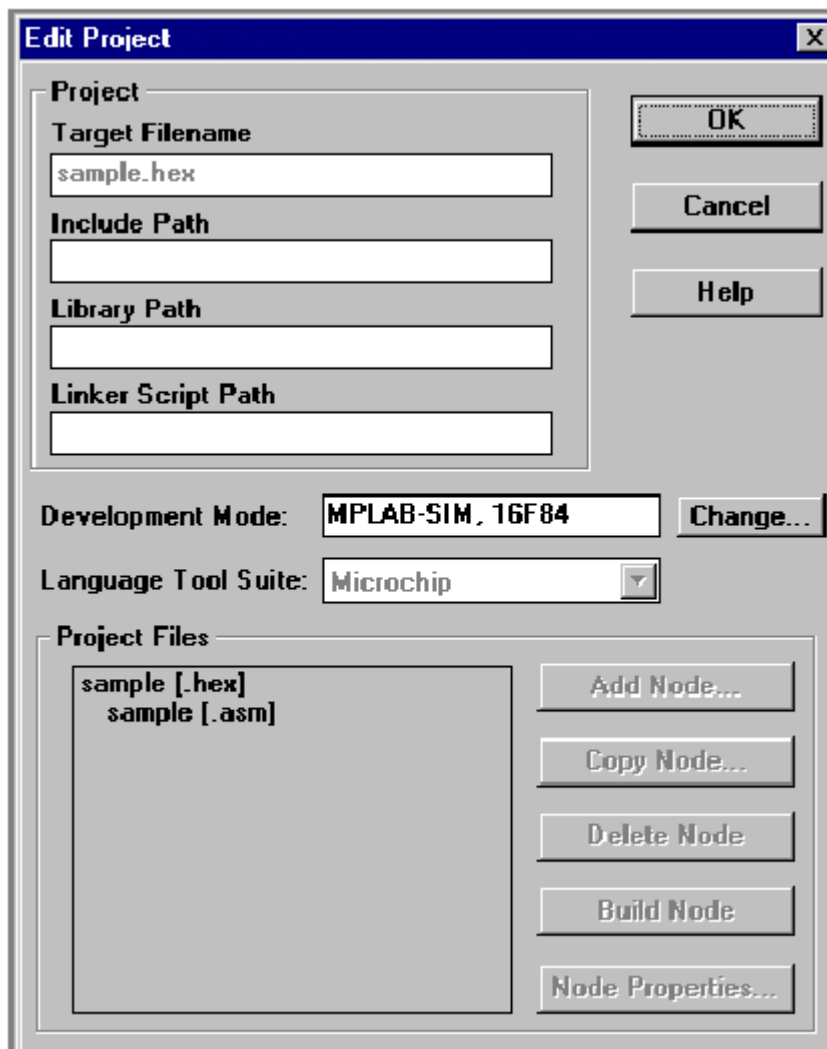


图 4-8：带节点的编辑“项目”会话窗口

这个例子里面，在路径窗口里面并没有生成路径。当你的应用越来越复杂，你可能需要在相应的窗口里面输入输入包含文件，库文件和链接器描述文件的路径。选择命令：***Options > Environment Setup*** 并按 **Projects** 图标，可以设置默认的开发语言，路径和所有项目的节点。

#### 4. 4. 6 创建项目

从菜单里面选择选择命令：***Project > Make Project*** 来使用 MPASM 完成创建工作。一个创建结果窗口将会出现，显示汇编器的命令行。该窗口如下图所示：

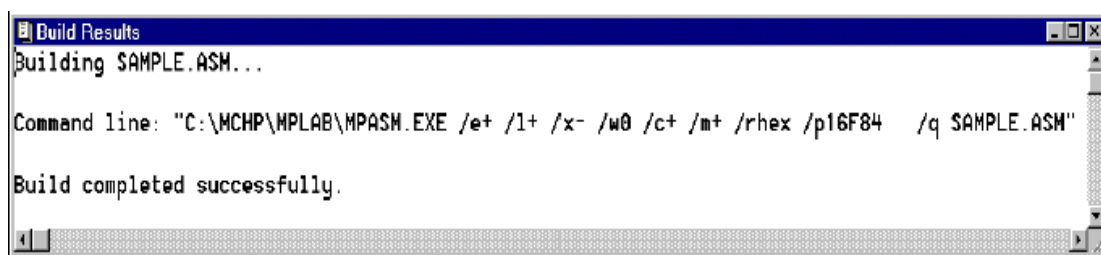


图 4-9： 创建结果窗口

#### 4. 4. 7 疑难问题解答

假如创建没有成功，请检查如下项目：

1. 根据创建结果，在源文件里面检查是否存在语法错误。假如发现了语法错误，就双击创建结果窗口里该条错误所在的地方，于是你会到达源文件里该条错误语句所在的地方。将错误语句改正过来，重新创建一次。
2. 选择命令：**Project > Edit Project.**，选择 HEX 文件的节点然后点击：**Node Properties**。检查是否在节点属性会话窗口里面是否有正确的创建工具（MPASM）。
3. 选择命令：**Project > Edit Project.**，检查在项目文件列表里面显示的文件名。假如你错误地添加了一个文件，点击它，然后点 **Delete Node**，再按照 4-4-5 节介绍的方法加入正确的节点。
4. 选择命令：**Project > Install Language Tool...**然后检查 MPASM 是否指向了 MPLAB-IDE 安装目录里的 MPASM.EXE。相应地，MPASM 也可以指向 MPASMWIN.EXE。但是应该相应地选择“WINDOWED”选择项。

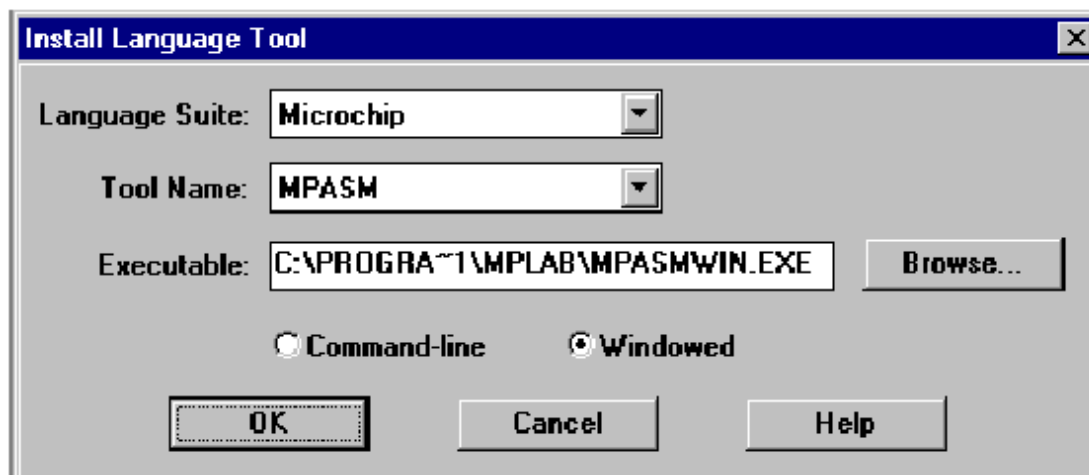


图 4-10： 安装语言工具会话窗口

5. 假如你从 DOS 得到信息说你已经用完了系统资源，你可以使用 WINDOWS EXPLORER 来在 MPLAB IDE 的安装目录里面选中 MPASM.EXE，然后点击右键打开节点属性会话窗口：

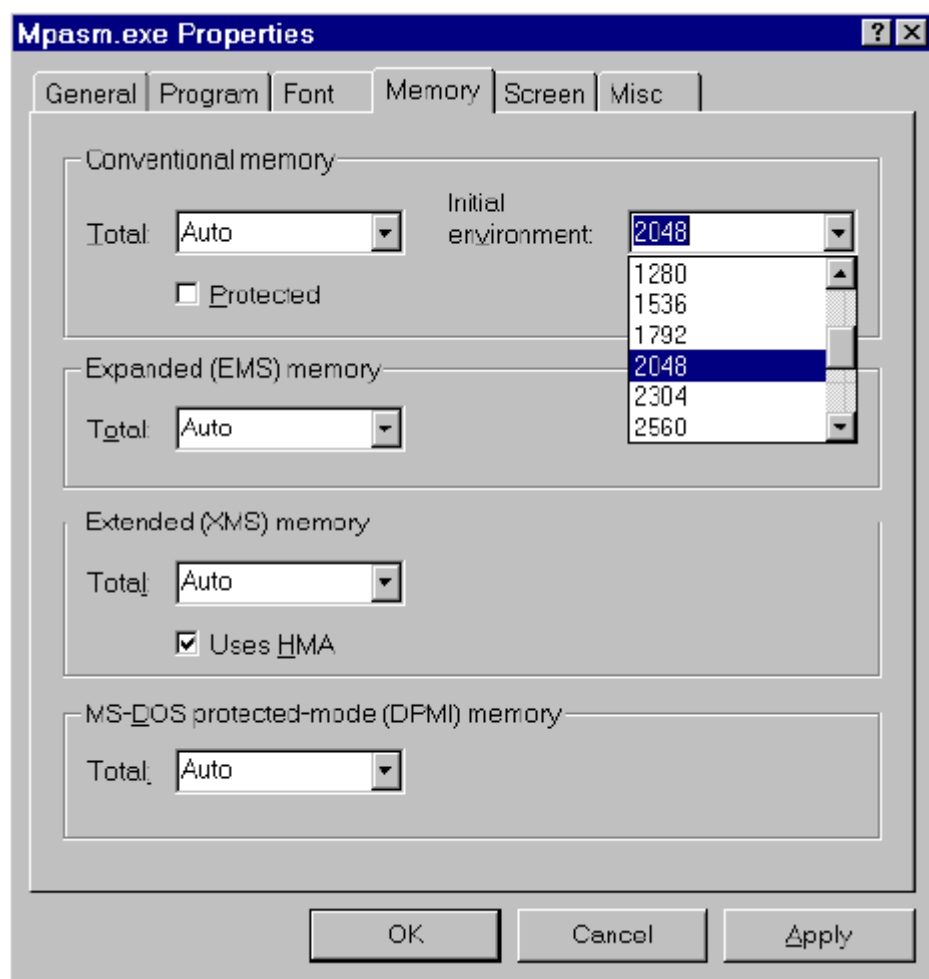


图 4-11：属性会话窗口

增加初试环境的大小。通常 2048 已经足够了。但假如在你的 AUTOEXEC.BAT 文件中有许多设置变量的应用程序的路径申明，你可能需要把该参数设得大一些。

#### 4. 4. 8 项目窗口

使用命令：**Window > Project** 打开项目，看看目标名称是否正确的设置，并与节点的源名称匹配。他们有不同的文件名，.ASM 和.HEX，但本范例里面都被命名为：SAMPLE。

这种项目窗口如下所示：

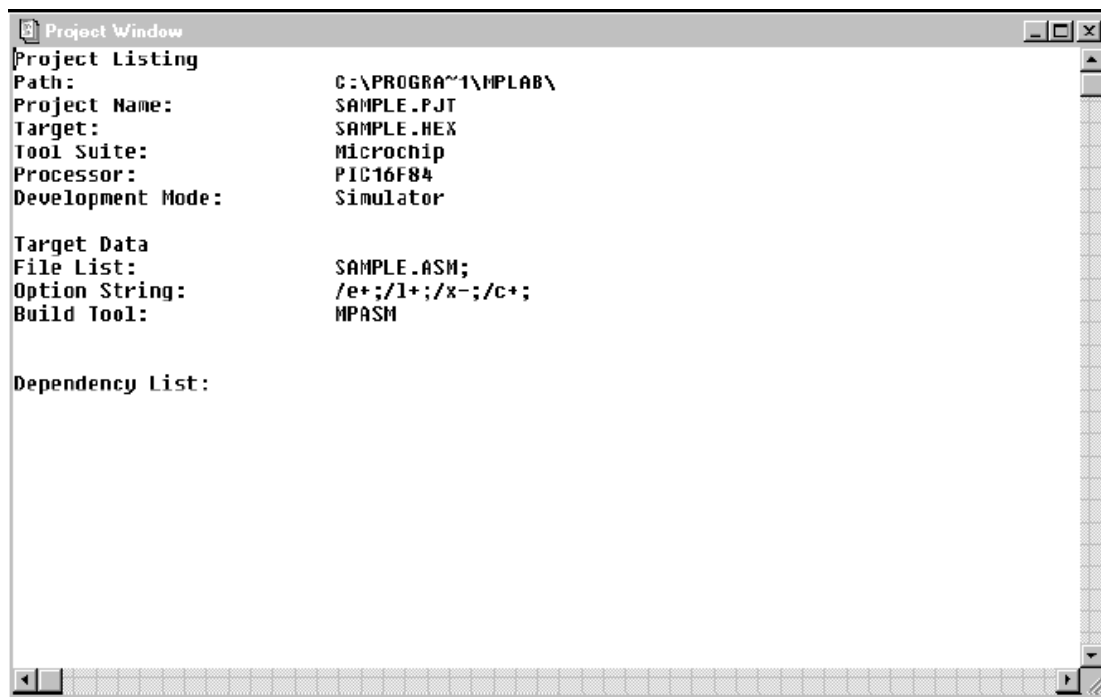


图 4-12: 项目窗口

#### 4. 4. 9 总结

这里针对以上的介绍，给出了一个简洁的设置新项目的步骤：

- 使用命令： **Project > New Project** 建立一个新项目。
- 为 MPASM 设置项目节点属性，选择合适的创建选择参数。
- 添加源文件节点。

#### 4. 5 不用建立项目编译单个 MPASM 源文件

不用打开一个项目而编译一个文件是可能的。这种方法的缺点是：尽管不用建立初始的项目，你必须在每次编译文件的时候指明选择项。这个范例将使用上一个范例相同的汇编文件。

你首先要关闭所有打开的项目，选择命令： **Project > Close Project** 。

##### 4. 5. 1 设置开发模式

为应用选择相应的开发模式。本范例里面，选择命令： **Options > Development Mode** 点击 **Tools** 图标，选择 MPLAB-SIM 模拟器，然后选择 PIC16F84 单片机，点击： **OK** 。

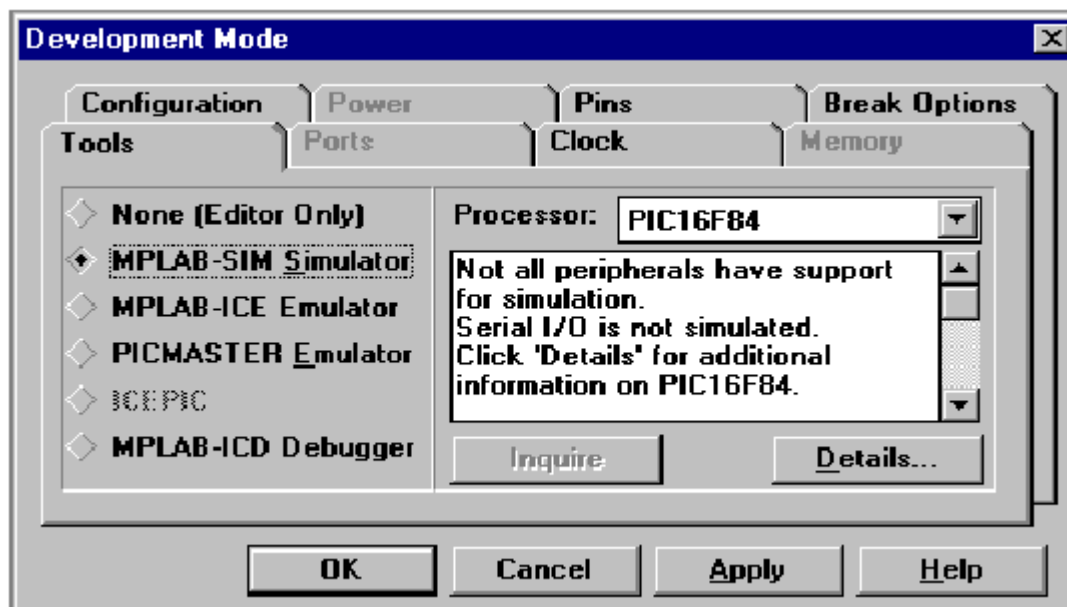


图 4-13: 开发模式会话窗口

#### 4. 5. 2 打开源文件

打开你想汇编的源文件。本范例中，使用了文件 `sample.asm`，该文件位于 MPLAB IDE 安装目录里面。

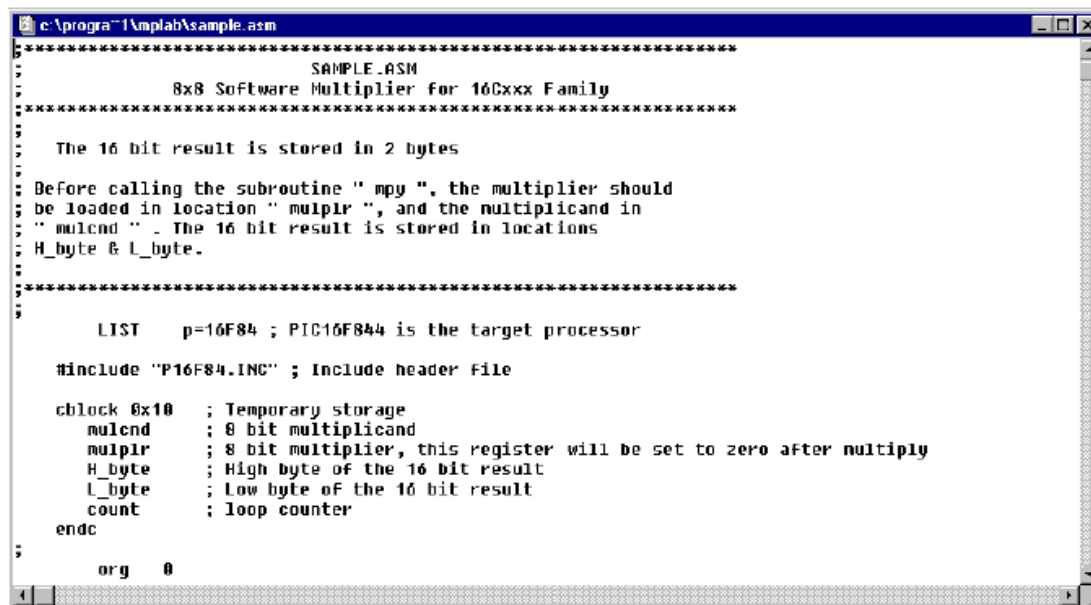


图 4-14: 源文件窗口

从菜单项里面选择命令: **Project > Build Node**，用 MPASM 来编译源文件 `sample.asm`。这时候 MPLAB IDE 会打开一个“创建工具会话窗口”(Build Tool Dialog)，形如下图:

### 4. 5. 3 编译源文件

选择命令：“**Project > Build Node**”，使用 MPASM 对源文件 sample.asm 进行编译。MPLAB IDE 会打开一个“创建工具会话窗口”（Build Tool Dialog），如下图所示：

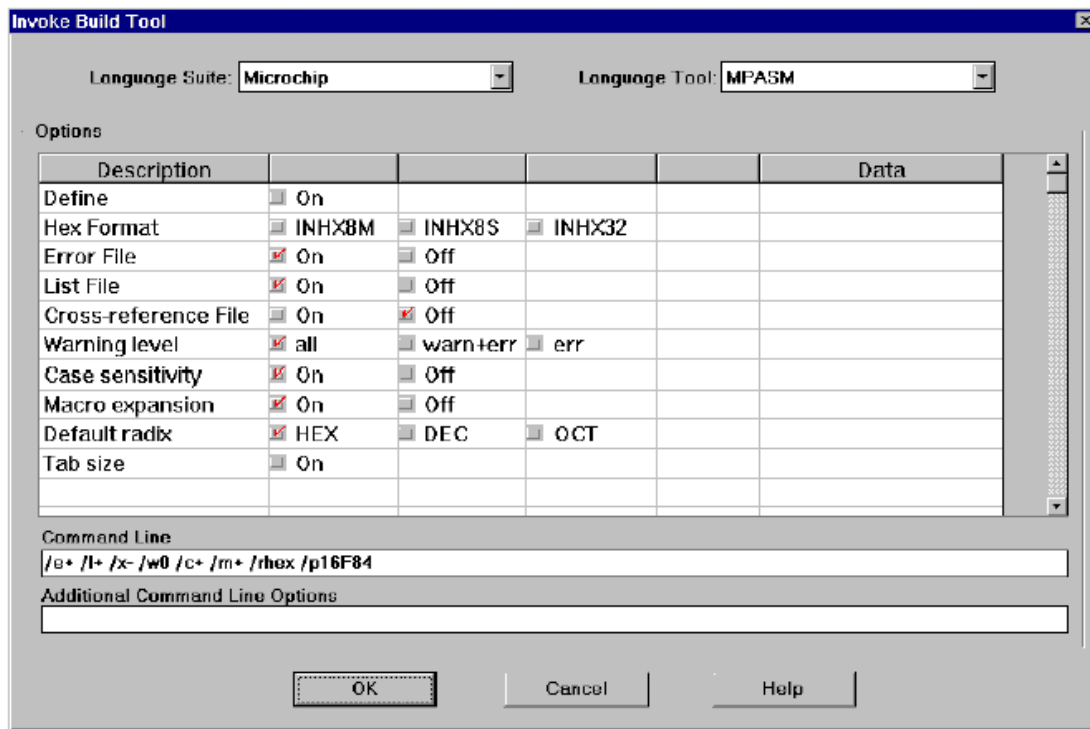


图 4-15：创建工具会话窗口

检查 MPASM 是否选中，并且设置工具选项，来与上面的设置相匹配。在“创建工具会话窗口”（Build Tool Dialog）里面点击“OK”按钮，开始创建工作。一个创建结果窗口将会生成，这里可以看见发送给汇编器的命令行和创建输出结果。如下图所示：

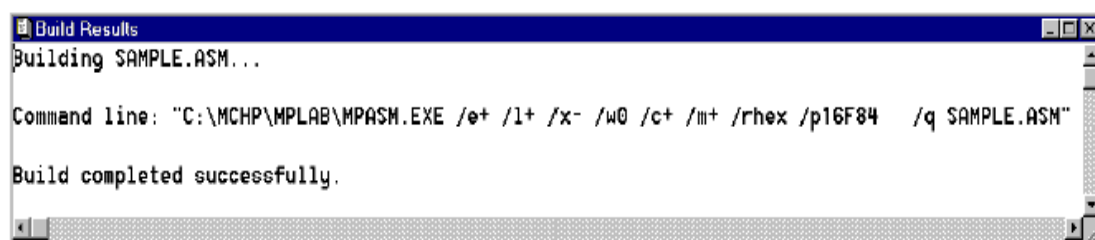


图 4-16：创建结果窗口



#### 4. 5. 4 疑难问题解答

假如创建没有成功，请检查如下项目：

1. 假如你修改过源文件，根据创建结果窗口，在源文件里面检查是否存在语法错误。假如发现了语法错误，就双击创建结果窗口里该条错误所在的地方，于是你会到达源文件里该条错误语句所在的地方。将错误语句改正过来，重新创建一次。
2. 选择命令：***Project > Install Language Tool...***然后检查 MPASM 是否指向了 MPLAB-IDE 安装目录里的 MPASM.EXE。相应地，MPASM 也可以指向 MPASMWIN.EXE，但是应该相应地选择“**WINDOWED**”选择项。



图 4-17：安装语言工具会话窗口

3. 假如你从 DOS 得到信息说你已经用完了系统资源，你可以使用 WINDOWS EXPLORER 来在 MPLAB IDE 的安装目录里面选中 MPASM.EXE，然后点击右键打开节点属性会话窗口：

增加初试环境的大小。通常 2048 已经足够了。但假如在你的 AUTOEXEC.BAT 文件中有许多设置变量的应用程序的路径申明，你可能需要把该参数设得大一些。

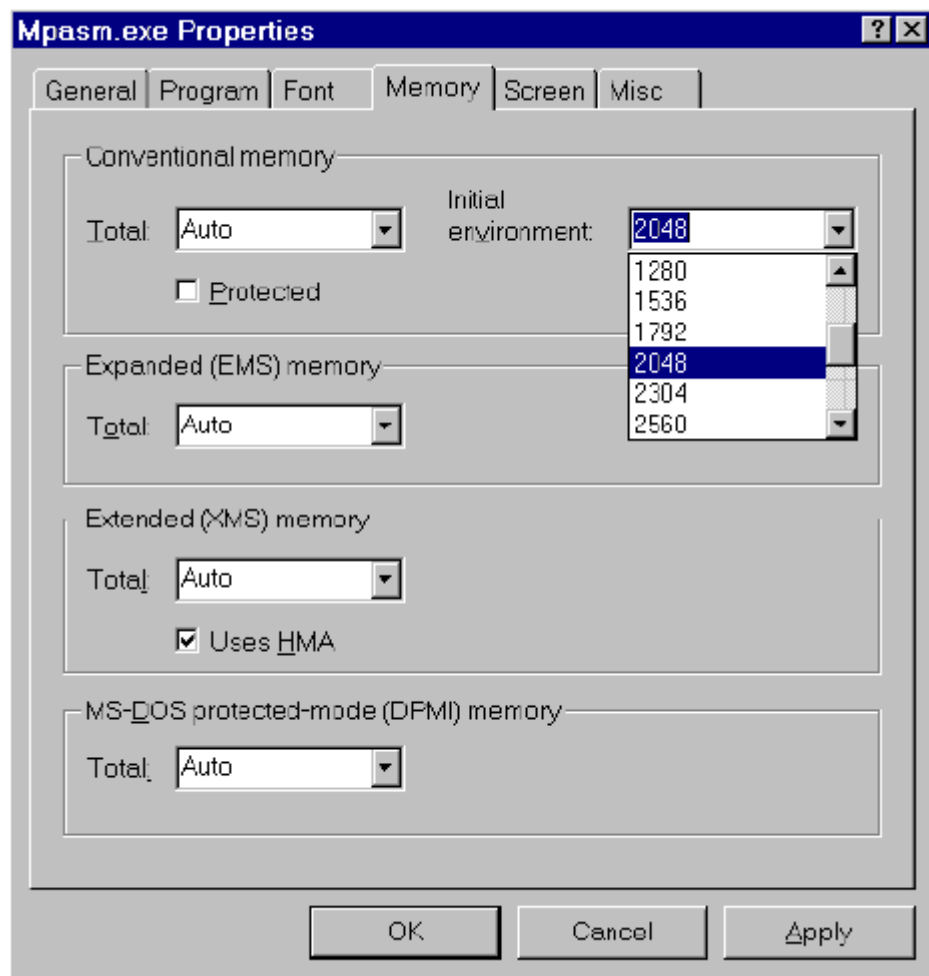


图 4-18: MPASM.EXE 属性会话窗口

#### 4. 5. 5 总结

这里针对以上的介绍，给出了一个简洁的设置新项目的步骤：

- 假如项目打开了，可以使用命令：***Project > Close Project***来关闭一个项目。
- 打开你想编译的源文件。
- 选择命令：***“Project > Build Node”***。
- 选择合适的开发语言工具，在“创建工具会话窗口”里面选择创建工具和创建选择项。

#### 4. 6 使用 MPLINK 创建一个有多个源文件的项目

按照以下步骤可以使用 MPLINK 来链接或两个以上的 MPASM 目标文件。假如你一直在进行前面几节的实验，请选择菜单命令：***“Project > Close Project”***来关闭项目。

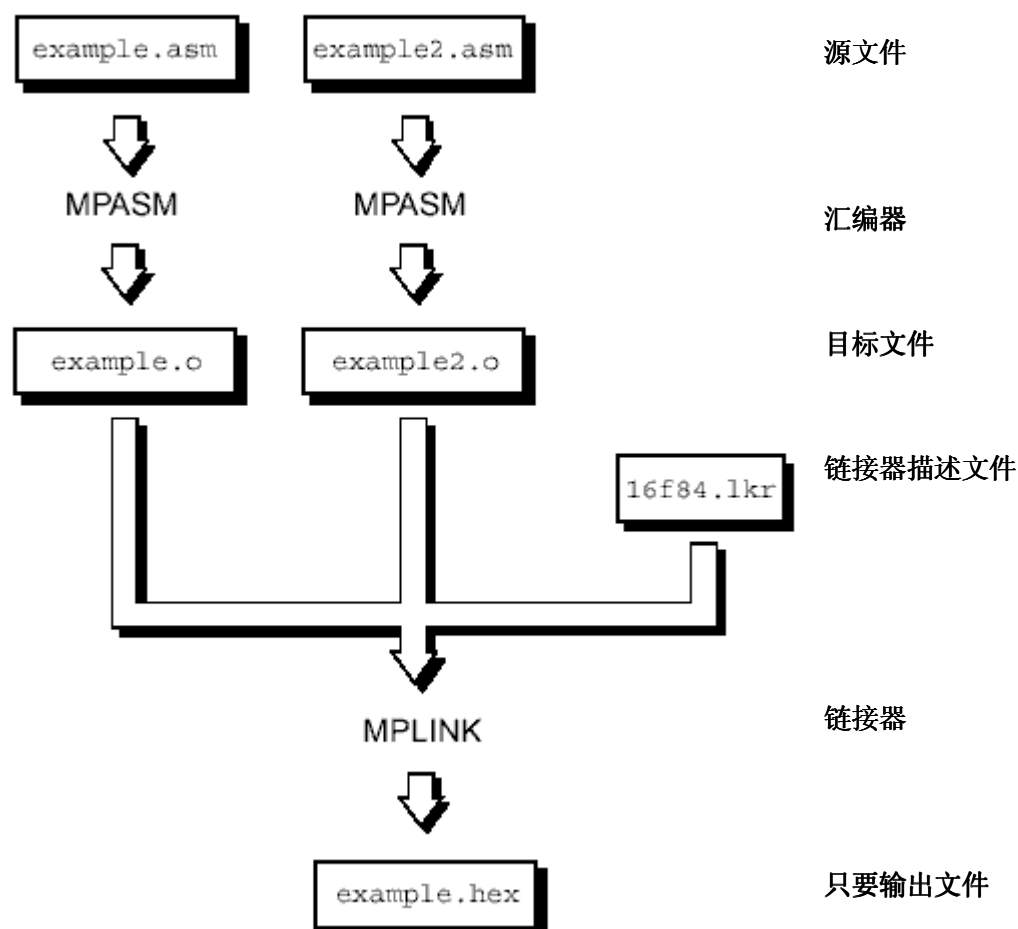


图 4-19: 有多个 MPASM 源文件的项目内部部件的关系

#### 4. 6. 1 设置开发模式

选择命令: “***Options > Development Mode***”, 点击 “**Tools**” 按钮。在本范例中, 选择 MPLAB-SIM 模拟器和 PIC16F84 单片机, 然后点击 “**OK**” 按钮。

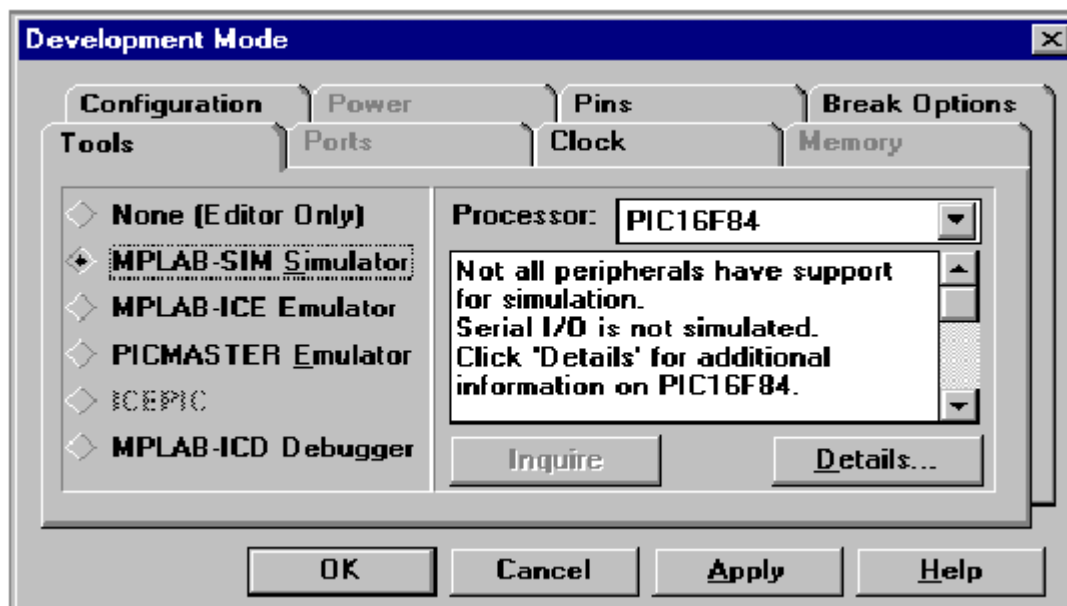


图 4-20: 开发模式会话窗口

#### 4. 6. 2 新项目

选择命令：“**Project > New Project**”，为新项目选择一个工作目录，然后将项目的名称键入文件名栏目里面。本范例中使用目录“\PROGRAM FILES\MPLAB\EXAMPLE”，并将项目命名为：“**EXAMPLE.PJT**”。

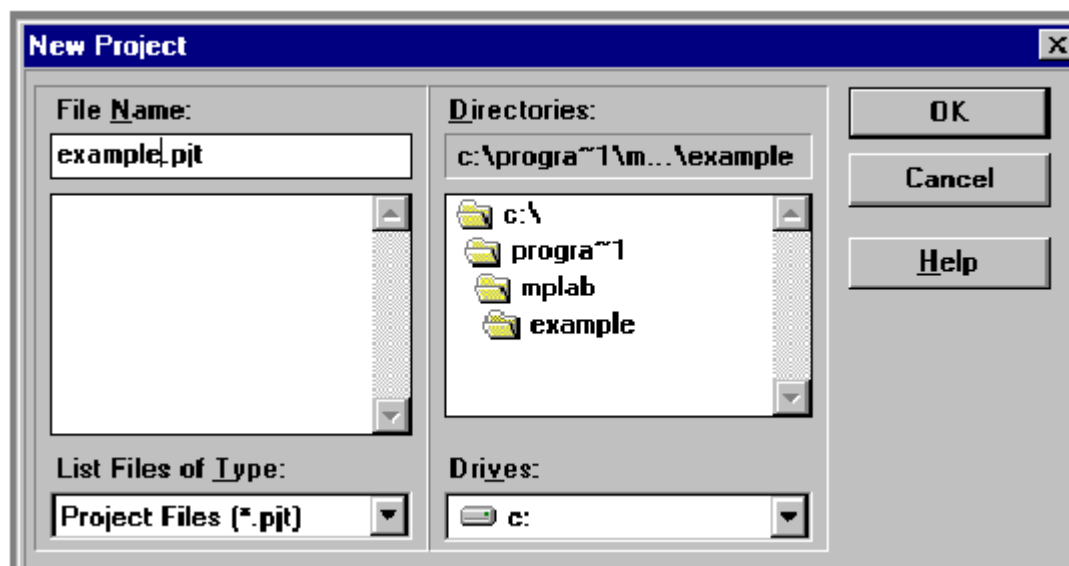


图 4-21: 新项目会话窗口 - example.pjt

#### 4. 6. 3 设置节点属性

在“编辑项目”（Edit Project）会话窗口里面的“项目文件”（Project Files）会话栏里选择新项目的名称，点击“**Node Properties**”（节点属性）打开这个会话窗口。为 MPLINK 设置语言工具。

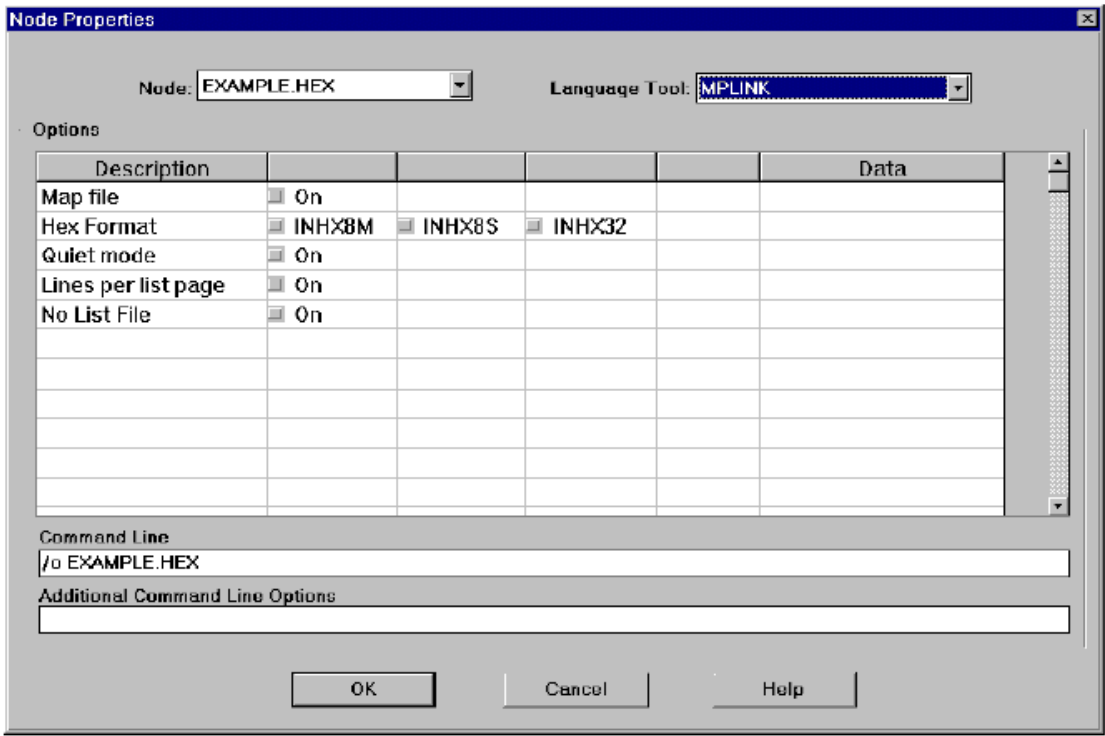


图 4-22：设置节点属性会话窗口

节点属性会话窗口显示了为语言工具（这里是 MPLINK）设置的命令行开关参数。当你第一次打开这个窗口的时候，选择盒（check box）所显示的是默认的选择项。在我们这里的范例里面，没有必要修改这些默认的设置。参考《MPASM 和 MPLINK, MPLIB 用户指南》，对这些命令行开关做进一步的了解。点击“OK”按钮返回到编辑项目（Edit Project）会话窗口。

4. 6. 4 添加第一个源文件节点

从编辑项目（Edit Project）会话窗口里面选择：“Add Node”（增加节点）。本范例中使用了在目录：“\PROGRAM FILES\MPLAB\EXAMPLE”下的源文件：“example.asm”。

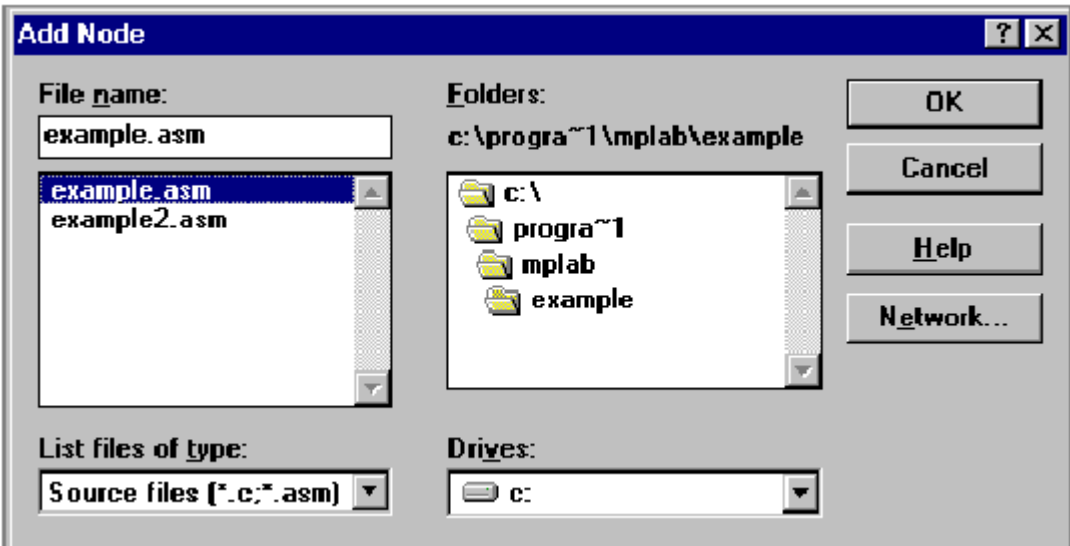


图 4-23: 增加节点会话窗口

你可以一次在这个会话窗口里面选择多个文件。假如需要在很多文件里面选择多个文件，可以按住 **<Ctrl>** 按键，再用鼠标点击逐个选择。要想选择某一范围内的文件。可以按住 **<Shift>** 按键，然后点击第一个文件和最后一个文件。

在编辑项目（Edit Project）会话窗口里面的项目文件列表里面选择文件：“example.asm”。然后点击：“Node Properties”（节点属性）。

检查开发语言工具应该选择为 MPASM。

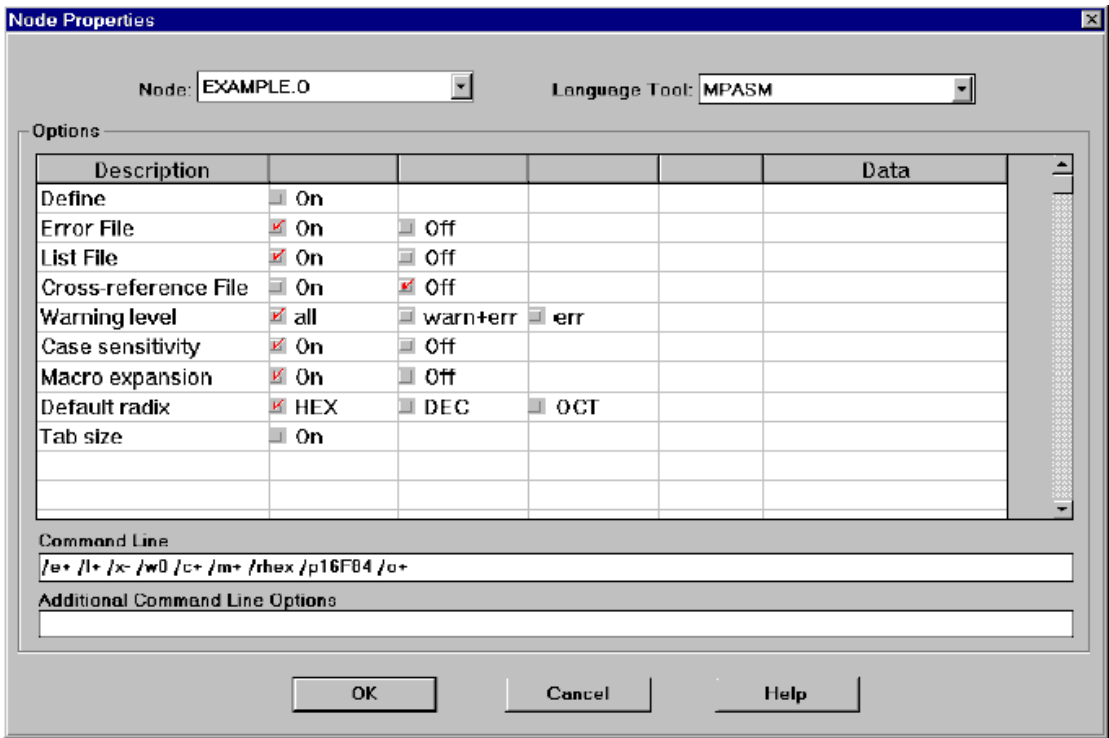


图 4-24: 节点属性会话窗口 - Example.o

节点属性会话窗口显示了为语言工具设置的命令行开关，在本范例里面，语言工具是 MPASM。请参照《MPASM 和 MPLINK, MPLIB 用户指南》进一步了解命令行参数的详细讲解。

点击 “OK” 返回到编辑项目会话窗口。

#### 4. 6. 5 添加新的源文件

按照前面两步的操作，可以添加本项目里的其他源文件节点。在这里的例子中，选择了目录为：“\PROGRAM FILES\MPLAB\EXAMPLE” 里的源文件：“example2.asm” 作为新的节点。当然你也可使用：“Copy Node”（拷贝节点）命令来输入和第一个源文件有相同节点设置的源文件。在 “Add Node”（添加节点）会话窗口里面，选择一个或一个以上的源文件。一旦文件选择完毕，点击：“OK” 按钮。这一步将为所选择的文件设置节点属性，设置方法与所使用的参考节点类似。这种方法在设置多个具有相同节点属性的源文件的时候非常有用。

#### 4. 6. 6 选择链接器描述（Linker Script）

使用 “Add Node” 按钮和前面描述的方法来选择链接器描述。链接器描述是一个 MPLINK 用来定义不同的 PICmicro MCU 存储器结构的文件。标准的链接器描述和 MPLINK 一起配套，并且都位于 MPLAB IDE 的安装目录里面。在本范例里面，从目录：\PROGRAM FILES\MPLAB\EXAMPLE 里面选择链接器描述文件：PIC16F84.LKR。对于一个链接器描述文件，无法设置节点选择项。

在编辑项目会话窗口里面点击 “OK”。

在这个简单的范例里面，在三个 “路径” 选择窗口里面，并没有选择目录入口。随着你的应用越来越复杂，假如项目的节点位于不同的目录里面，你就可能需要在相应的窗口里面输入包含文件，库文件和链接器描述文件所在的目录。通过选择命令：“Options > Environment Setup” 然后点击 “Projects” 按钮（见第 7-8-5-2 节）可以为所有的项目选择默认的语言工具，目录和节点。

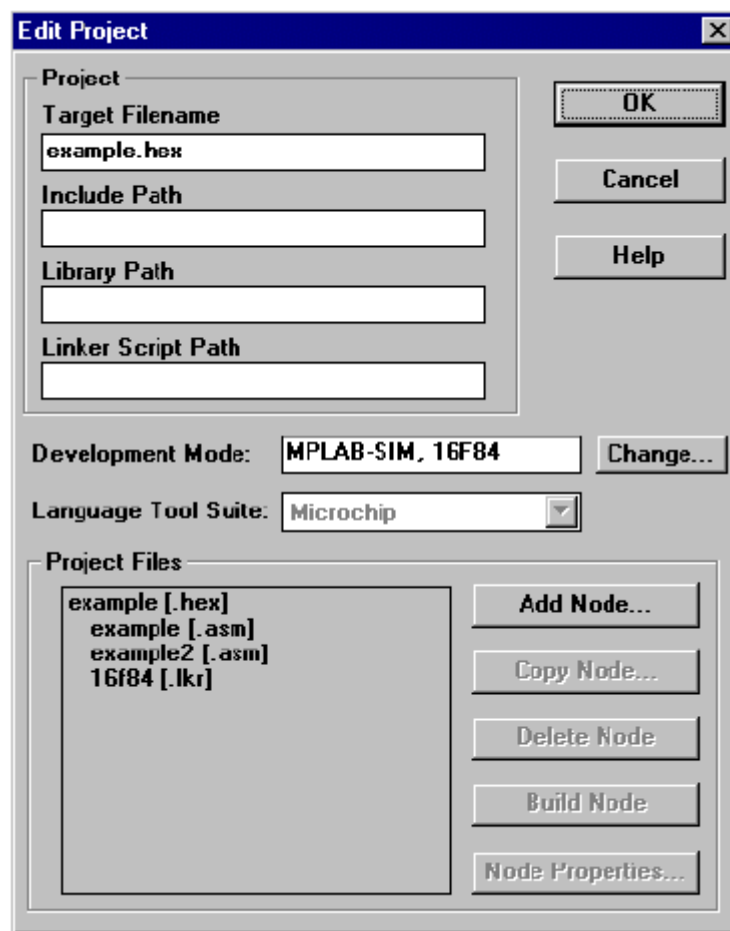
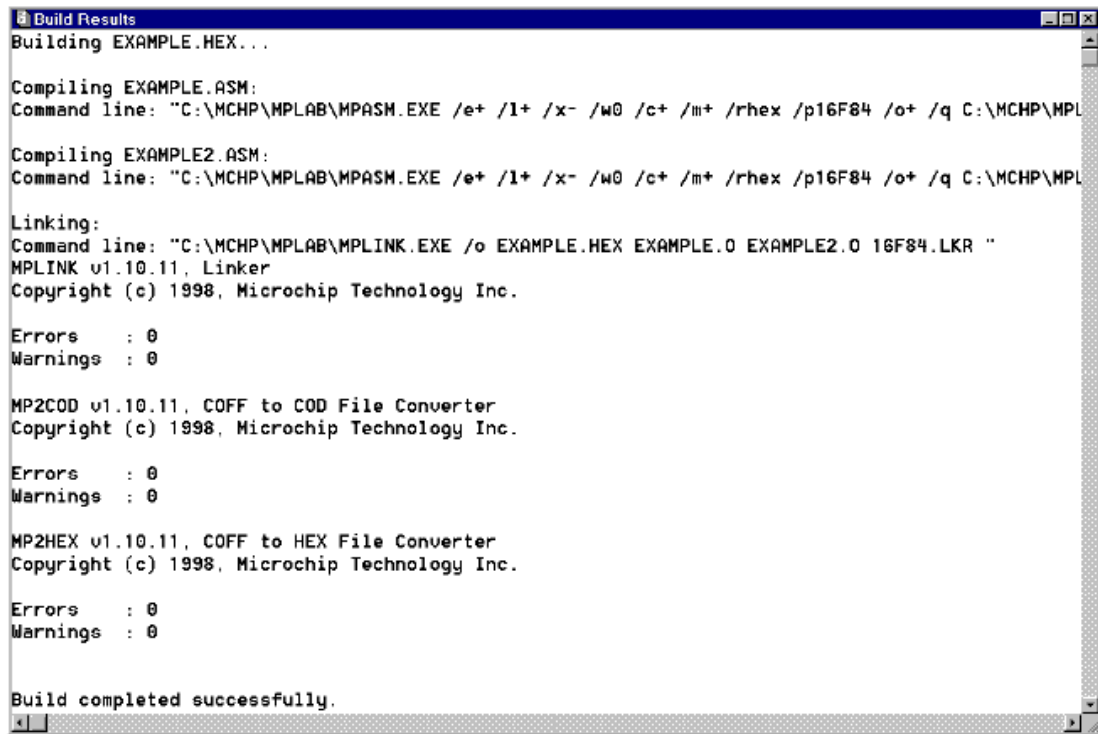


图 4-25: 编辑项目会话窗口 – 链接器描述 (Linker Script)

#### 4. 6. 7 创建项目 (Make Project)

选择命令: “***Project > Make Project***” 可以应用 MASM 和 MPLNK 来对应用进行编译。编译完成后, 会出现一个创建结果窗口, 该窗口里面显示了发送给每个工具的命令行指令, 如下图所示:





```
Build Results
Building EXAMPLE.HEX...

Compiling EXAMPLE.ASM:
Command line: "C:\MCHP\MPLAB\MPASM.EXE /e+ /l+ /x- /w0 /c+ /m+ /rhex /p16F84 /o+ /q C:\MCHP\MPLAB\EXAMPLE.ASM"

Compiling EXAMPLE2.ASM:
Command line: "C:\MCHP\MPLAB\MPASM.EXE /e+ /l+ /x- /w0 /c+ /m+ /rhex /p16F84 /o+ /q C:\MCHP\MPLAB\EXAMPLE2.ASM"

Linking:
Command line: "C:\MCHP\MPLAB\MPLINK.EXE /o EXAMPLE.HEX EXAMPLE.O EXAMPLE2.O 16F84.LKR "
MPLINK v1.10.11, Linker
Copyright (c) 1998, Microchip Technology Inc.

Errors      : 0
Warnings    : 0

MP2C0D v1.10.11, COFF to C0D File Converter
Copyright (c) 1998, Microchip Technology Inc.

Errors      : 0
Warnings    : 0

MP2HEX v1.10.11, COFF to HEX File Converter
Copyright (c) 1998, Microchip Technology Inc.

Errors      : 0
Warnings    : 0

Build completed successfully.
```

图 4-26: 创建结果窗口

#### 4. 6. 8 疑难问题解答

假如创建工作无法成功的进行，可以参照如下条款检查：

1. 假如你修改了源文件，检查窗口里的创建结果，看看是否有语法错误，假如有，则改正过来，重新编译一次。
2. 选择命令：**Project > Edit Project**，选择 HEX 文件的节点然后点击：**Node Properties**。检查在节点属性会话窗口里面是否有正确的创建工具（MPLINK）。本项目的源文件创建工具应该是 MPASM
3. 选择命令：**Project > Edit Project**，检查在项目文件列表里面显示的文件名。假如你错误地添加了一个文件，点击它，然后点 **Delete Node**，再按照 4-4-5 节介绍的方法加入正确的节点。
4. 假如 MPLAB IDE 报告信息：“Time-out”，点击 **OK** 继续。根据你的 PC 机速度的不同以及你的项目大小，你可以定义这个时间的长短，MPLAB IDE 在报告“Time-out”之前将会等待。通过选择命令：**Options > Environment Setup**，点击“**Project**”按钮，然后在会话窗口里面调整“**Build Timeout Length**”（创建时限）。假如你不想看到时间到的错误报告，就可以将创建时限设置为 0 即可。

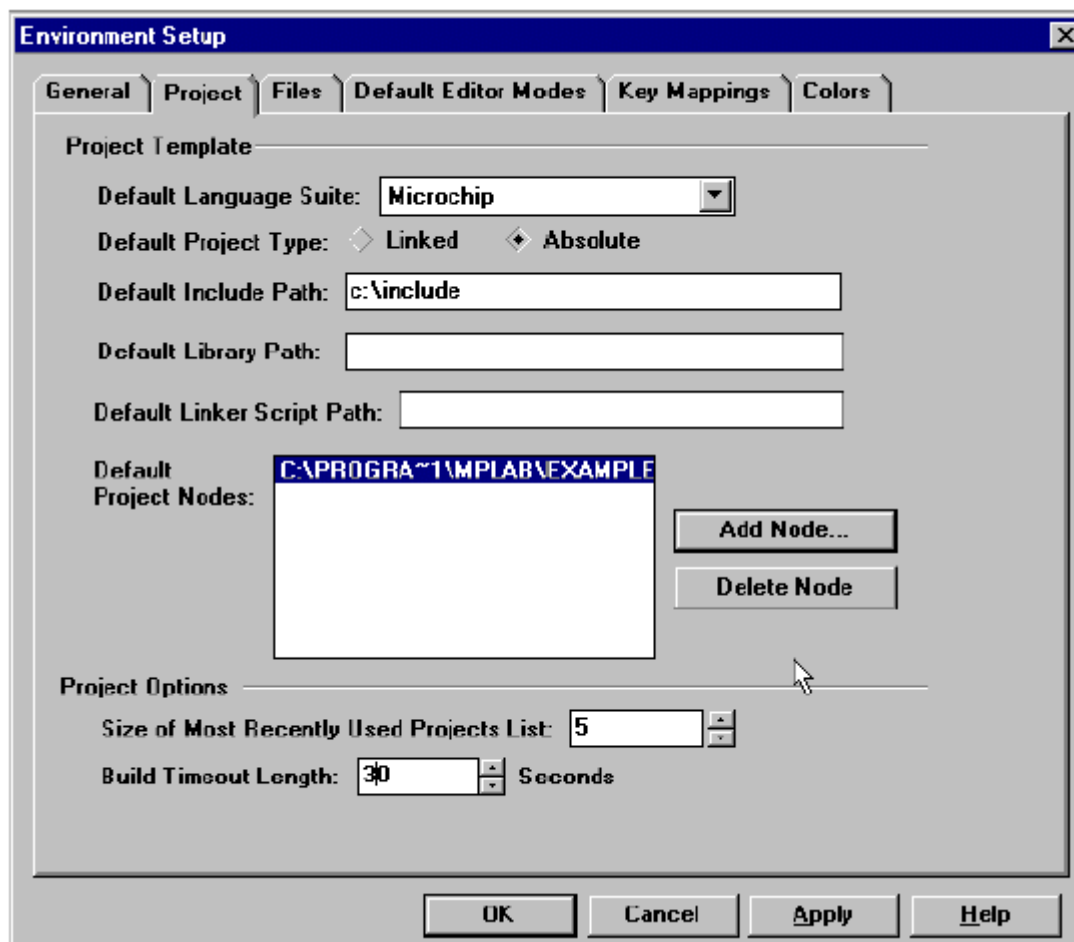


图 4-27: 环境设置会话窗口

5. 选择命令: **Project > Install Language Tool...** 然后检查 MPASM 是否指向了 MPLAB-IDE 安装目录里的 MPASM.EXE。相应地, MPASM 也可以指向 MPASMWIN.EXE。但是应该相应地选择 “WINDOWED” 选择项。

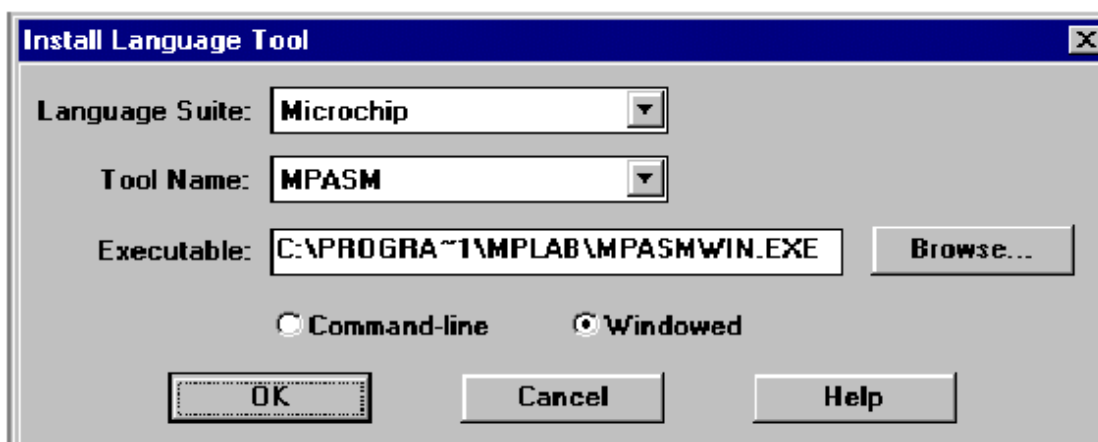


图 4-28: 安装语言会话窗口 – MPASM

相应的, MPASM 也可以指向 MPASM.EXE, 并且“**Command Line**”选择项应该被选中。

6. 假如你从 DOS 得到信息说你已经用完了系统资源, 你可以使用 WINDOWS EXPLORER 来在 MPLAB IDE 的安装目录里面选中 **MPASM.EXE**, 然后点击右键打开节点属性会话窗口:

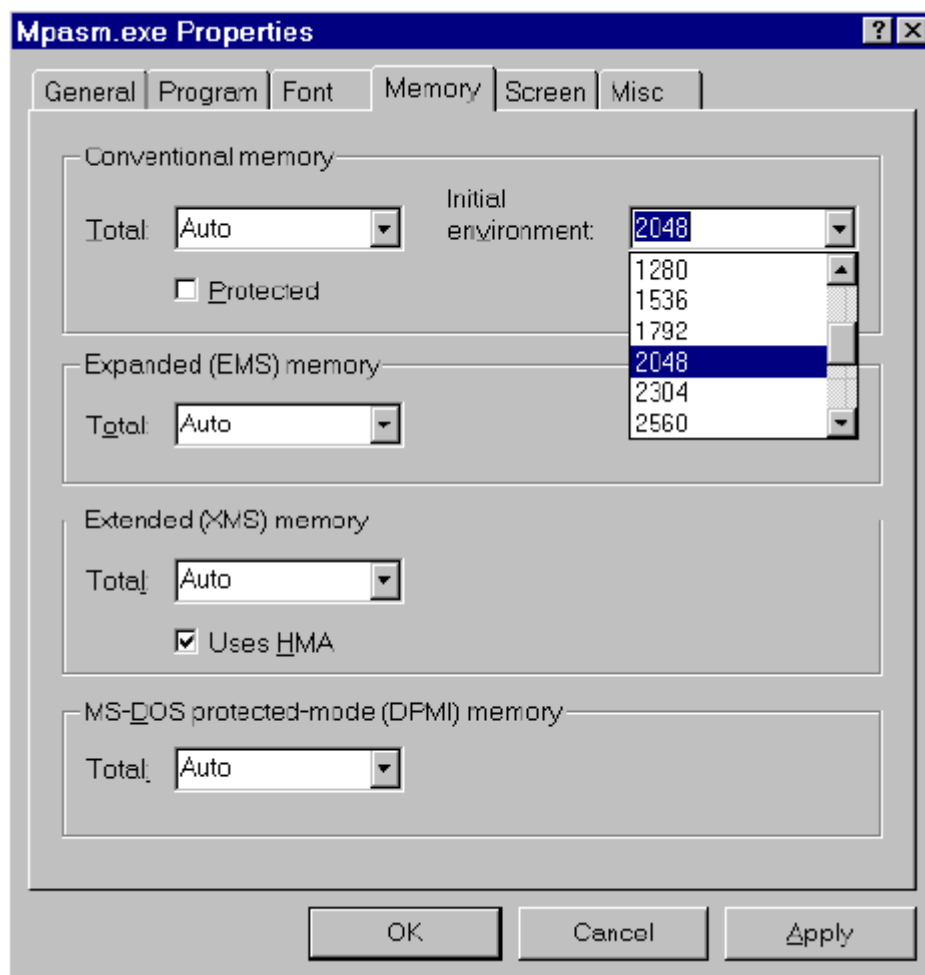


图 4-29: MPASM 属性会话窗口

增加初试环境的大小。通常 2048 已经足够了。但假如在你的 AUTOEXEC.BAT 文件中有许多设置变量的应用程序的路径申明, 你可能需要把该参数设得大一些。

#### 4. 6. 9 项目窗口

选择命令: “**Window > Project**”, 将会看到下面的窗口:

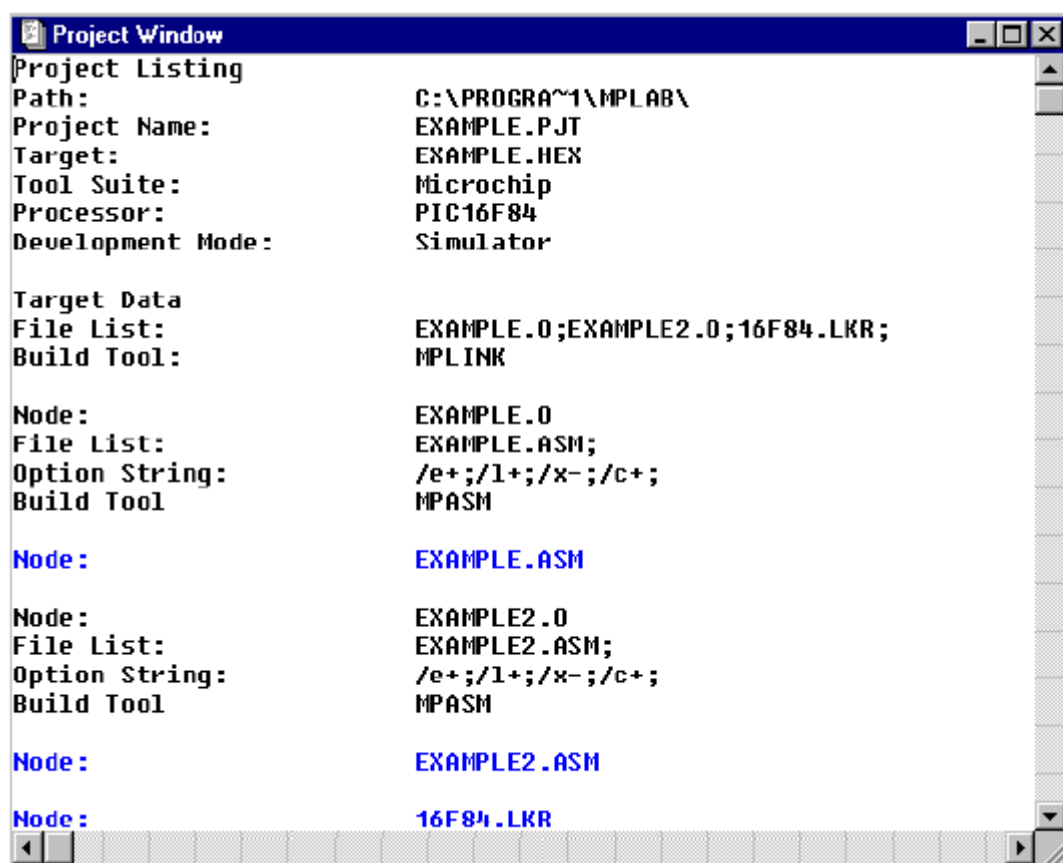


图 4-30: 项目窗口

#### 4. 6. 10 总结

这里针对以上的介绍，给出了一个简洁的设置新项目的步骤：

- 可以使用命令：***Project > New Project***来打开一个新项目。
- 为 MPLINK 设置项目节点属性。
- 增加源文件节点，并且按照需要设置节点属性。
- 添加链接器描述文件节点。

#### 4. 7 使用 HI-TECH PIC C 编译器创建项目

以下的指南将会介绍如何使用 Hi-Tech 公司的 PIC C compiler 对 MPLAB IDE 里面的项目进行创建工作。假如你按照前面指导进行了一系列的工作，可以使用命令：“***Project > Close Project***”来关闭项目。

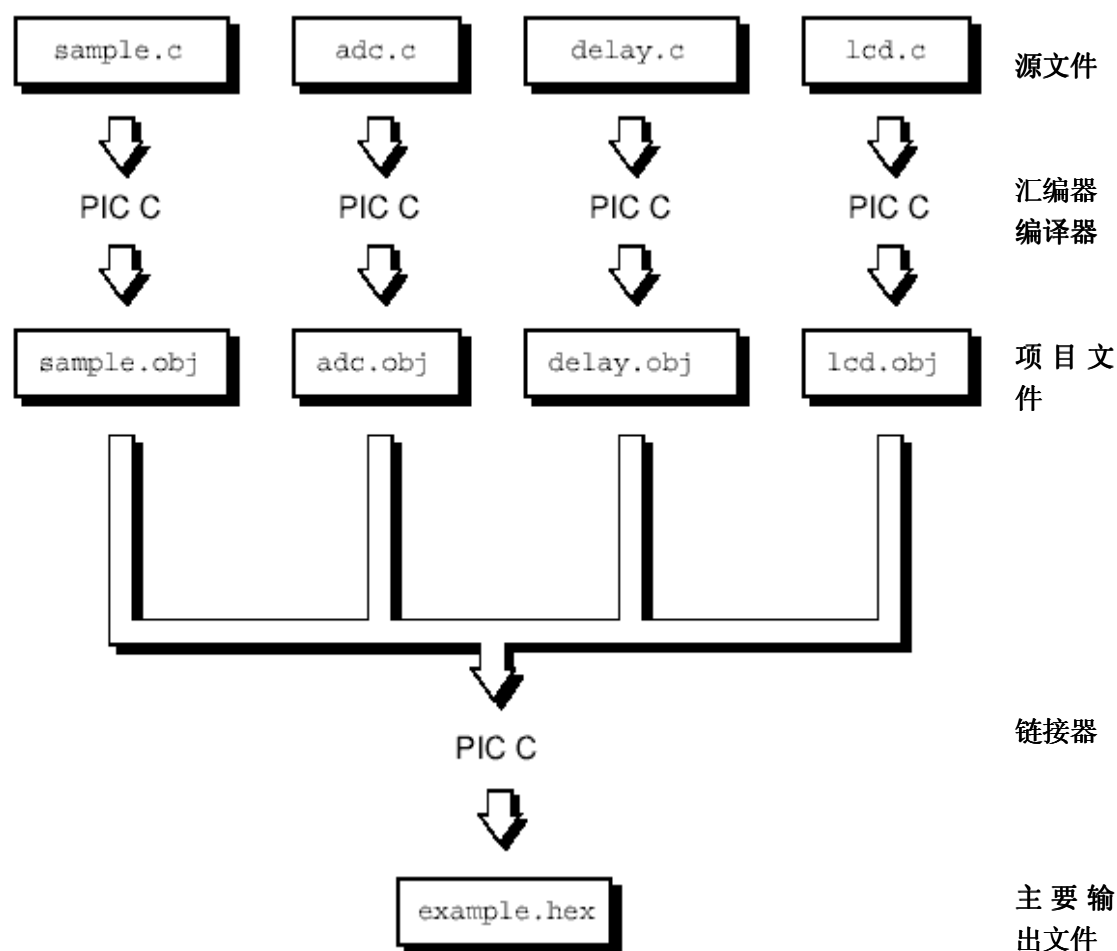


图 4-31: PIC C 项目源文件之间的关系

#### 4. 7. 1 设置开发模式

选择命令：“***Options > Development Mode***”并且点击：“**Tools**”按钮。本范例里面选择 MPLAB-SIM 模拟器和 PIC16C77 PICmicro 单片机。然后点击：“**OK**”按钮即可。

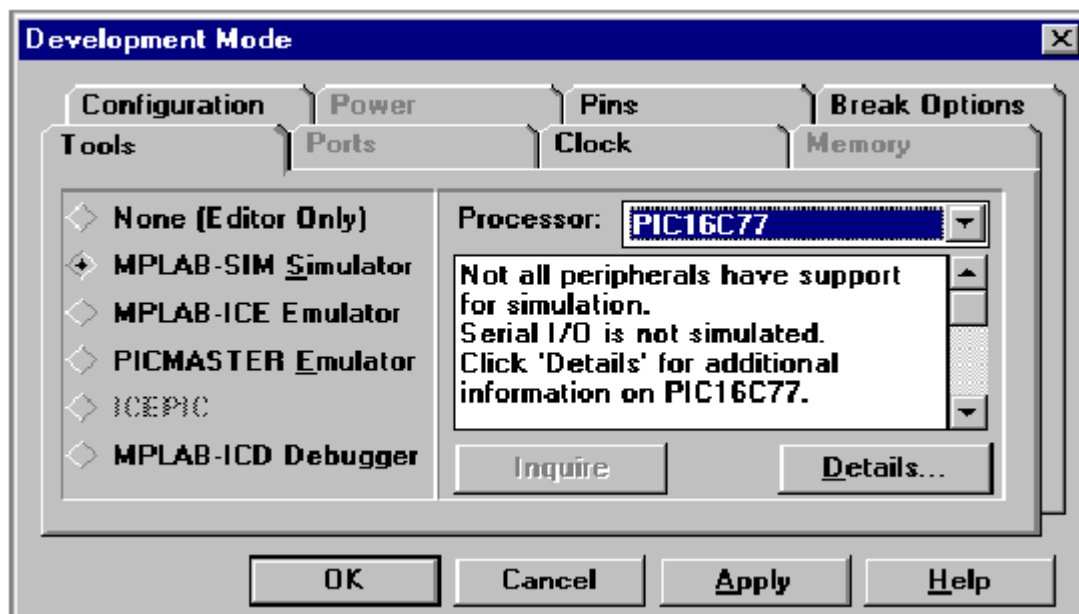


图 4-32: 开发模式会话窗口 – PIC16C77

#### 4. 7. 2 安装 PIC C 语言工具

确认PIC C 已经正确地安装在MPLAB IDE 之下。进入命令：“***Project >Install Language Tool***”，设置HI-TECH 开发工具（你自己的可执行文件的目录也许不同）。

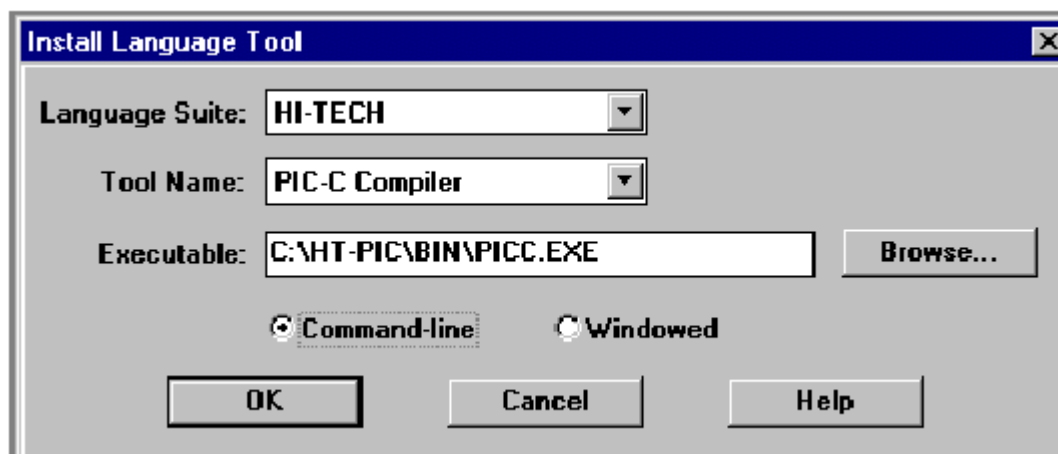


图4-33: 安装语言工具会话窗口 – PIC C 编译器

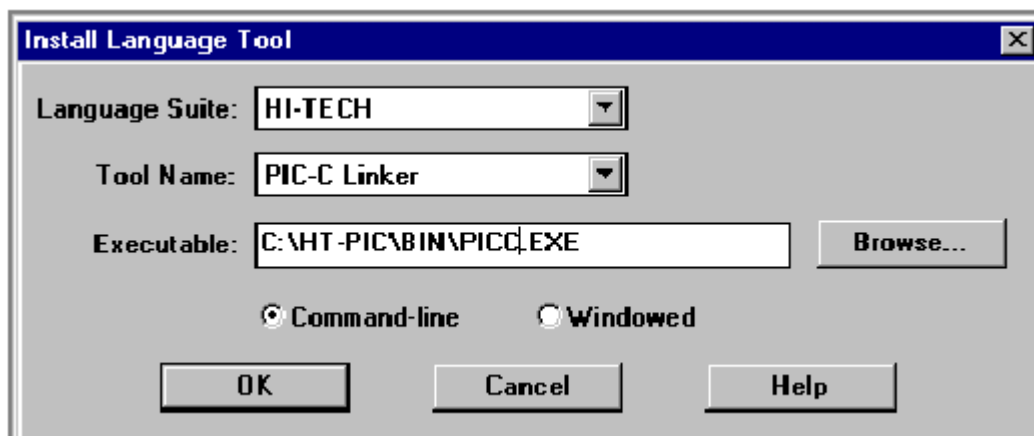


图4-34: 安装语言工具会话窗口 – PIC C 链接器

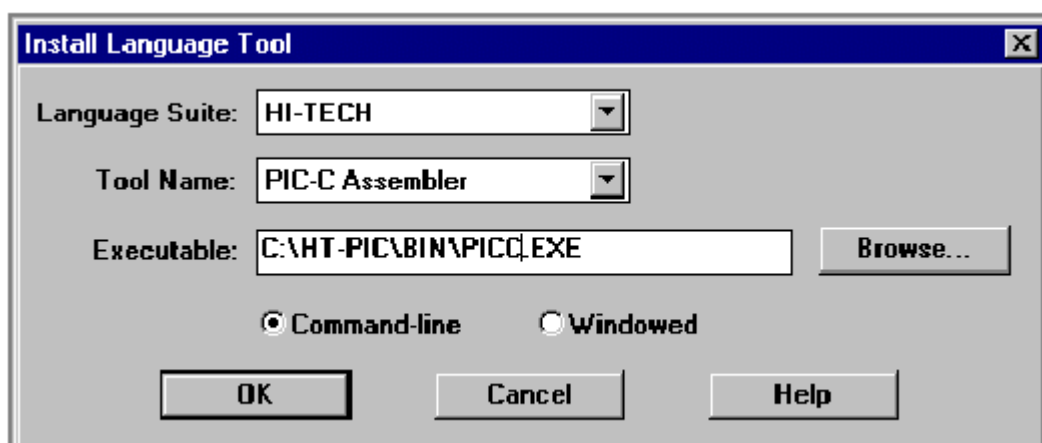


图4-35: 安装语言工具会话窗口 – PIC C 汇编器

#### 4. 7. 3 新项目

选择命令: “***Project > New Project***” 然后为新项目选择一个目录, 接着输入项目名称。本范例里面将项目取名为: “**SAMPLE.PJT**”, 位于路径: “\HT-PIC\SAMPLES”。

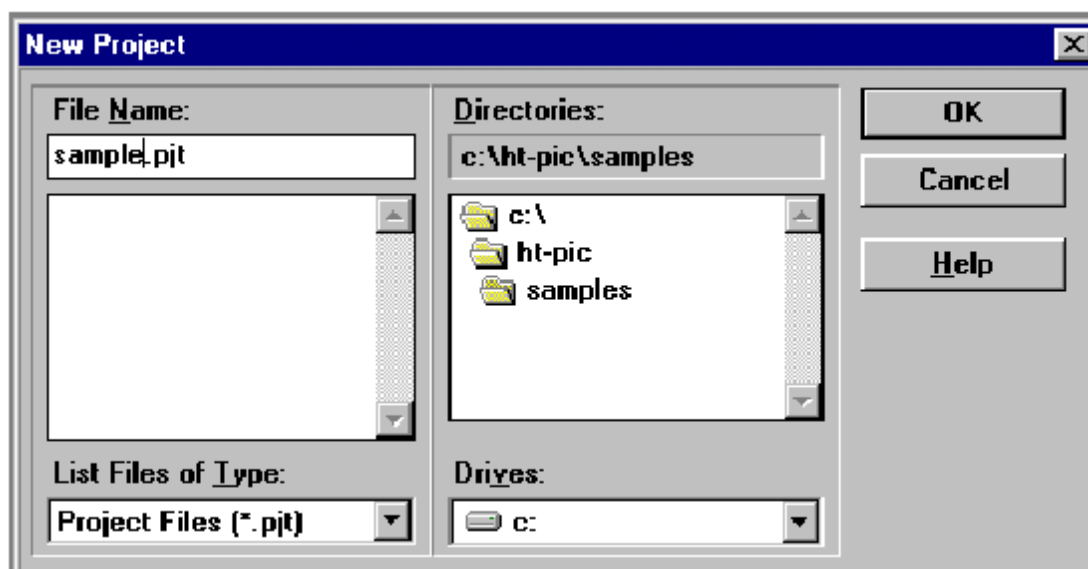


图 4-36: 新项目会话窗口 - sample.pjt

在设置好了项目名称后，点击：“OK”，接着编辑项目会话窗口将会出现：

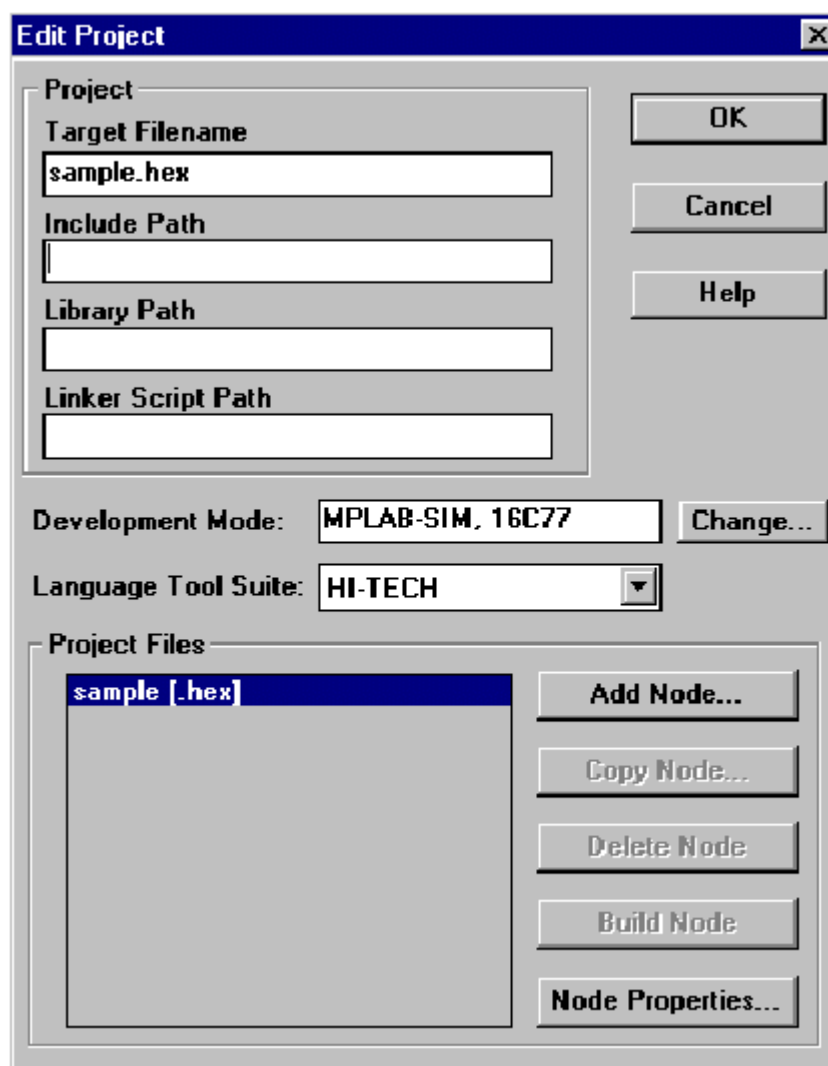


图 4-37: 编辑项目会话窗口 - sample.hex



确认将 “**Language Tool Suite**”（语言工具）设置为 HI-TECH。

注意：确认在 “**Edit Project**”（编辑项目）窗口里面设置路径，以便 MPLAB IDE 知道编译器所在的正确位置。假如你的编译器是安装在另外一个目录里面，你将会看见一条错误信息，报告无法打开源文件。

4. 7. 4    设置节点属性

在 “**Edit Project Dialog**”（编辑项目）会话窗口里的 “**Project Files**”（项目文件）会话窗口里面选择项目名称。然后点击 “**Node Properties**”（节点属性）。设置语言工具为 “**PIC-C Linker**” 然后检查 “**Generate Debug Info**”（生成调试信息）选择盒。在数据栏里面键入 “**Microchip**”。见下面图示：

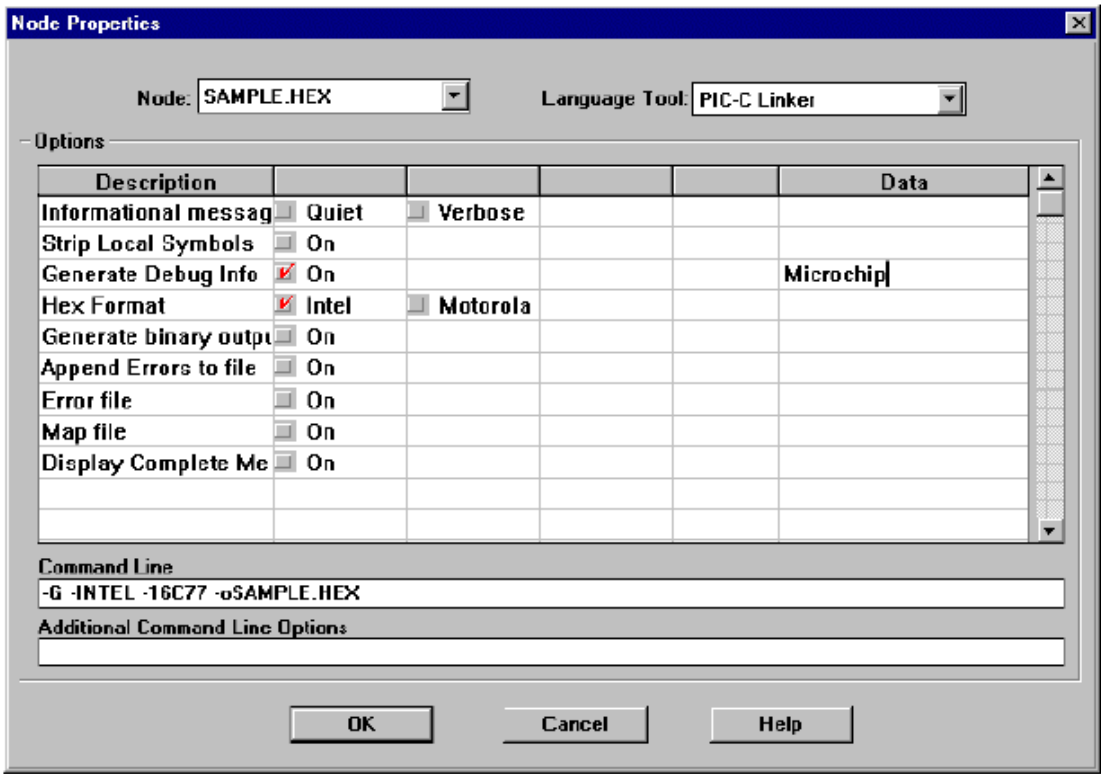


图 4-38：节点属性会话窗口 - sample.hex

节点属性会话窗口显示了开发工具的命令行开关，这里开发工具是 PIC C。当你第一次打开这个会话窗口的时候，选择盒里的选择项是开发工具的默认设置。本范例里面，只是 “**Debug Info Generate**” 设置需要改变。关于这一命令行开关的使用，可以参考 HI-TECH 有关资料。

在 “**Node Properties**”（节点属性）会话窗口里面点击 “**OK**” 按钮返回到 “**Edit Project**”（编辑项目）会话窗口。

4. 7. 5    添加源文件

假如在编辑项目（**Edit Project**）会话窗口里点击： “**Add Node**”。可以从目录： **\HT-PIC\SAMPLES** 添加源文件： **SAMPLE.C**。

当添加节点会话窗口里面显示的文件被选中后，点击 “**Node Properties**”。

按以下的方法设置该会话窗口：

- 为 **SAMPLE.OBJ** 设置的开发语言工具设置为：**PIC C Compiler**。
- 检查通用调试信息窗口（Generate Debug Info box）。
- 在通用调试信息栏里面填入：“Microchip”。

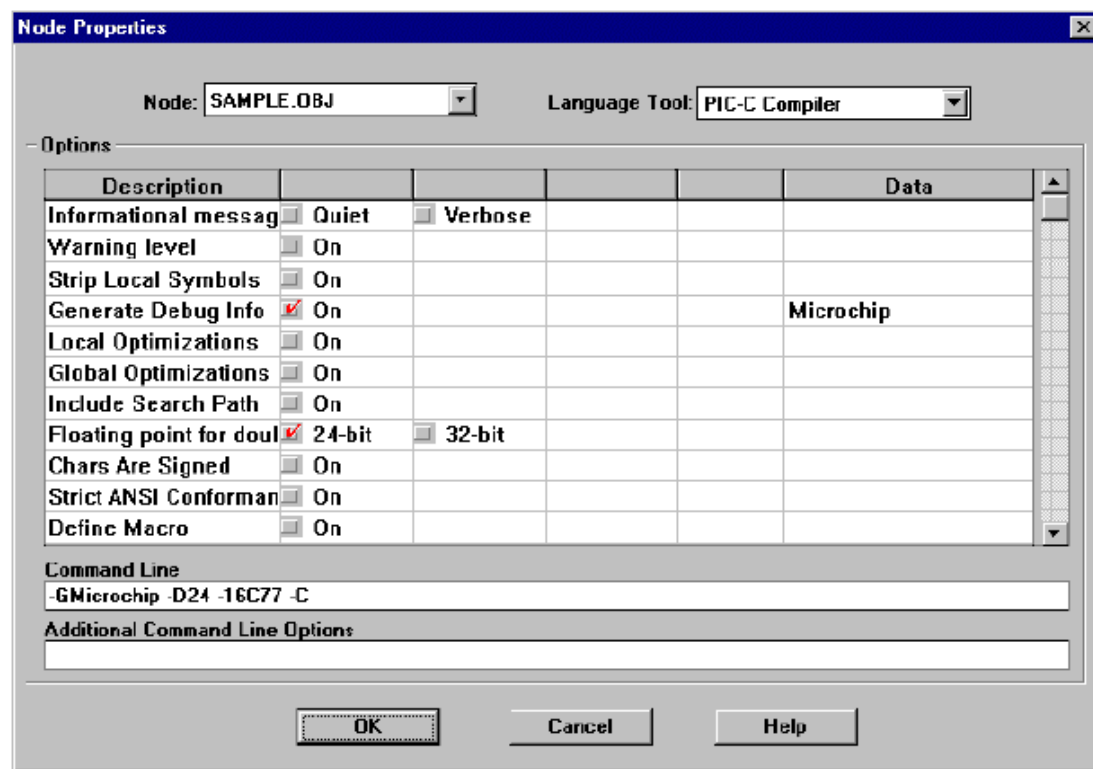


图 4-39：节点属性会话窗口 - sample.obj

这时候会看见目标文件名被自动设置为：“**SAMPLE.OBJ**”。

节点属性会话窗口显示了为语言工具（这里是 PIC C）设置的命令行开关参数。当你第一次打开这个窗口的时候，选择盒（**check box**）所显示的是默认的选择项。在我们这里的范例里面，仅仅调试信息设置需要改变。请参考有关 HI-TECH PIC C 编译器的文档资料对这些命令行开关做进一步的了解。点击“**OK**”按钮，选择节点“**SAMPLE.C**”并且使用 **Copy Node** 按钮来添加与 **SAMPLE.C** 有相同的节点属性的 **ADC.C**，**DELAY.C** 和 **LCD.C** 节点。最后建立的项目会向下图所示：

对于所有项目的默认开发工具，路径和节点等参数的设置，可以使用命令：“**Options > Environment Setup**”，然后点击项目（Projects）命令条。

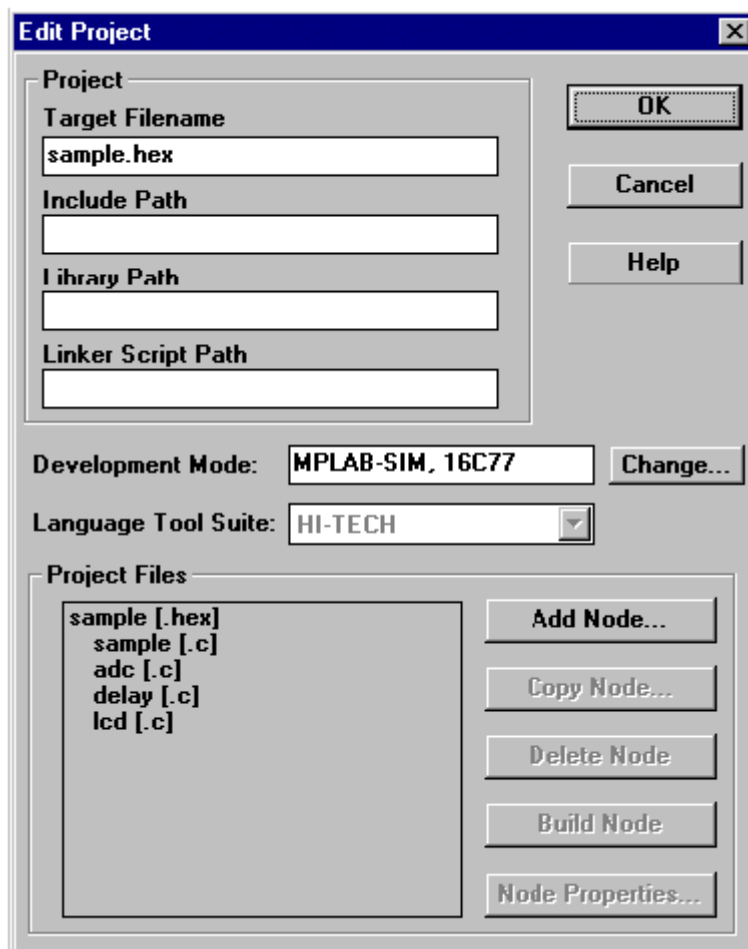


图 4-40：编辑项目会话窗口 - sample.hex

#### 4. 7. 6 创建项目

从菜单里选择命令：“***Project > Make Project***”可以应用 **HI-TECH C** 编译器和链接器来对应用进行编译。编译完成后，会出现一个创建结果窗口，该窗口里面显示了发送给每个工具的命令行指令，如下图所示：

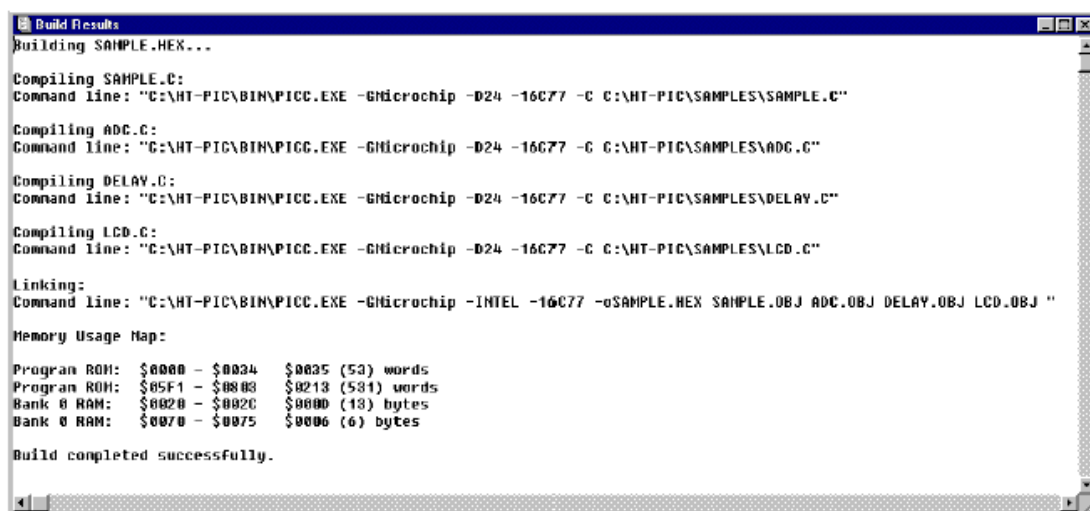


图 4-41：创建结果窗口 - sample.hex

#### 4. 7. 7 疑难问题解答

假如创建工作无法成功的进行，可以参照如下条款检查：

1. 假如你修改了源文件，检查窗口里的创建结果，看看是否有语法错误，假如有，则可以双击某一错误，这时候会自动到达源文件里面的错误语句上，改正过来，重新编译一次。
2. 选择命令：***Project > Edit Project***，选择 HEX 文件的节点然后点击：**Node Properties**。检查在节点属性会话窗口里面是否有正确的创建工具。本项目的源文件创建工具应该是 **PIC C 链接器**。
3. 选择命令：***Project > Edit Project***，检查在项目文件列表里面显示的文件名。假如你错误地添加了一个文件，点击它，然后点 **Delete Node**，再按照 4-7-5 节介绍的方法加入正确的节点。
4. 选择命令：***Project > Install Language Tool...***然后检查 PIC C 编译器和 PIC C 链接器是否都指向了 PICC.EXE。

#### 4. 7. 8 项目窗口

打开命令：“***Window > Project***”窗口，会看到下面图 4-42 的显示：

#### 4. 7. 9 总结

这里针对以上的介绍，给出了一个简洁的设置新项目的步骤：

- 为 PIC C 编译器，链接器和汇编器设置语言工具。
- 你需要设置 INCLUDE 文件的目录为：\HT-PIC\H（或者 PIC C 在你的机器上的安装目录）。
- 选择命令：“***Project > New Project***”来建立新项目。
- 为项目节点打开通用调试信息（Generate Debug Info）。
- 为每个源节点设置通用调试信息为：“Microchip”。

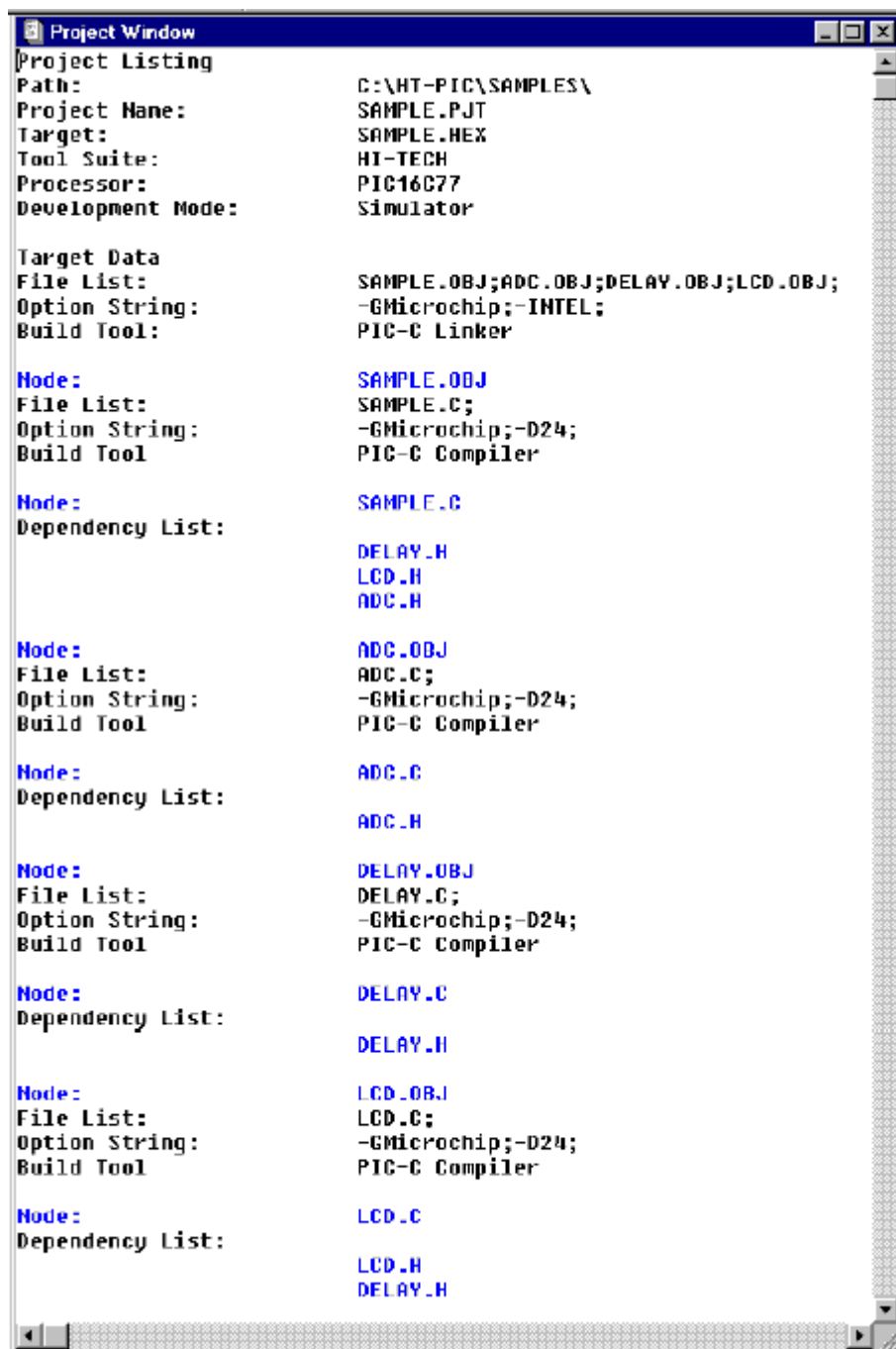


图 4-42: 项目窗口 - sample.pjt

#### 4. 8 用 MPLAB-C17 或者 MPLAB-C18 创建一个项目

关于如何使用 MPLAB-C17 或者 MPLAB-C18 来创建一个项目，参考《MPLAB-CXX 用户指南》（文档资料号：DS51217）。

## 第 5 章 MPLAB 编辑器

### 5.1 介绍

本章主要介绍了什么是 MPLAB 编辑器，对你有什么帮助，它的构造以及功能。由于 MPLAB 是组成 MPLAB IDE 的一部分，它的详细构造与功能将在第七章一起介绍。

### 5.2 提示

本章内容：

- 什么是 MPLAB Editor(编辑器)
- 对你有何帮助
- 构造
- 功能

### 5.3 什么是编辑器

MPLAB 编辑器是 MPLAB IDE 的一部分。当 MPLAB IDE 运行是，编辑器总是可用的。它不是一个独立的可执行性文件，而是集成在 MPLAB IDE 中的一组特性。

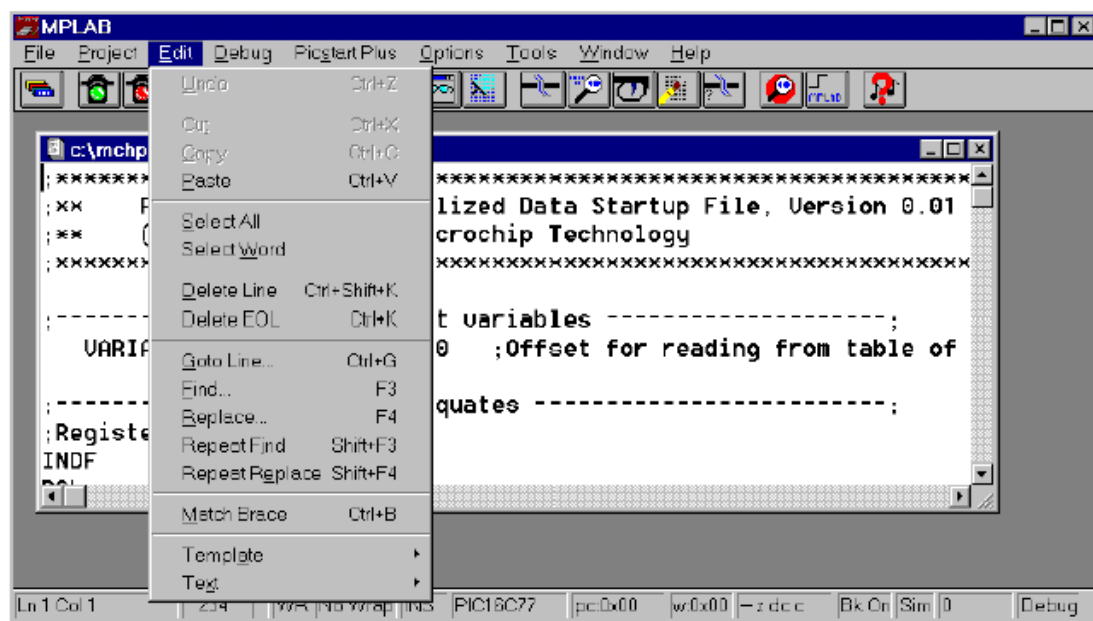


图 5-1: MPLAB Editor 的使用

### 5.4 对你有何帮助

MPLAB IDE 和编辑器是为微控制器开发者设计的，用来发展和调试微晶公司的 PIC 微控制器系列产品的硬件的一个简单，快捷的方式。

### 5.5 MPLAB 编辑器的特点

#### 5.5.1 文件大小

唯一的局限性是当前系统下全部可以利用的内存，它可编辑的文件的数目和打开编辑

窗口的数量是没有限制的。可打开文件的大小和包含的行数也都是没有限制的。

### 5.5.2 WINDOWS 下 MDI 的风格

- MPLAB 编辑器基于 WINDOWS MDI 之上。
- 菜单中包含大多数的命令和组件。
- 移动鼠标或使用键盘的快捷键。
- 提供“剪切板”和“粘贴板”。

### 5.5.3 重新配制键盘

你可以重新配制键盘来满足你的需要。可以通过两项组合键盘来表示，一种是<Esc+G>，<Ctrl+K>，和<Ctrl+B>。同样也可以使用<Alt+F>和<Alt+S>这样的键。甚至可以把所有你想用的用组合键来表示。

### 5.5.4 轻松“创建”文件

通过 MPLAB 编辑器你可以：

- 定制一个复制的模板---标准的文本---只需点击几下鼠标，就可以插入到一个当前的文件中。
- 把工作中所需要的模板分成几个不同的组。
- 调用模板来自动化使用。

## 5.6 MPLAB 编辑器的功能

MPLAB 编辑器提供了下列操作功能：

- 文件操作
- 模板操作
- 文件处理
- 编辑窗口模式
- C 语言认知

### 5.6.1 文件操作

创建一个文件，选择文件进行编辑，保存文件，运用文件菜单（见 7.4 章）。MPLAB 编辑器允许通过修改现有文件来进行保存或将文件保存到新的文件名中。

### 5.6.2 模板操作

如何避免与新建的源代码文件重复？你可以从原已完成的源文件中粘贴，然后，简单地向新的源文件中粘贴必要的模式及文本。但是，那是一种不恰当的处理方法。MPLAB IDE 为你提供了插入源文件的模板，它是先建立文本文件或部分文件，通过插入这些“固定”的文本，可以为你节省最初的代码编写时间，而不必向新建的源文件中重复输入。你可以使用 MPLAB IDE 中 TEMPLATES 子目录中的代码模板，或者自己创建。

一旦你安装了 MPLAB IDE，你可以创建属于自己的新的源文件（或是打开现有的文件），也可以向新的源文件中插入模板文本，在开发程序阶段，可以搜索到专门的制造者，帮你很快的确定你所需定位的区域。

你可以重复上述步骤建立不同的.tpl 文件。例如，任何一个典型的应用程序或芯片，都需要一个独立的.tpl 文件。

### 5.6.3 文本处理

尽管 MPLAB 编辑器被计划像文本编辑程序一样使用,但它还有一些特点,在一般的文本编辑中非常有用。详情请见 7.6 章。

#### 5.6.3.1 插入, 选中和删除文本

MPLAB 编辑器插入文本要在其他插入或重打模式下进行。MPLAB 编辑器在状态栏中显示的模式像“INS”和“OVR”其中的一种。

文本选中的功能是,可以选择两项,单词或整个一行。可以清除一项,行或者是光标所在的这一整行。甚至可以通过 MPLAB IDE 中的查找和替换,去搜索和替换文本或者是特殊选项。

#### 5.6.3.2 缩进与放大文本

编辑源程序,是非常普通的。MPLAB 编辑器提供了一个可以改变一行或多行文本的缩进和放大。

#### 5.6.3.3 更改文件

MPLAB 编辑器可以改变已选中的文本文件中的大小写字母。

#### 5.6.3.4 括号

用户可以在 MPLAB 编辑器中使用括号这项字符,包括中方括号和圆括号,它们常用来定义部分文本和源程序。

#### 5.6.3.5 撤消

MPLAB 编辑器能记录编辑过程,也可以撤消前面的命令。

#### 5.6.3.6 自动文本约束模式

输入原始文本时,在可以使用的行宽下,很容易获得适用于文本的程序。在编辑源程序的代码文件时,这是个典型的不能执行文件。

在状态栏范围区的底部,双击鼠标左键来改变自动文本的约束模式。在没有击活时,此区域显示“无范围”。双击鼠标,击活它。列如,约束模式在卷 72 中被击活时,状态栏会显示“Wr 72”。

MPLAB 编辑器在文件中约束行与用语言类定义窗口是不同的。

##### 5.6.3.6.1 语言类“ ”(空)或“C”

MPLAB 编辑器在定义范围栏中,在一行中若遇到空格或连字符会自动换行。

##### 5.6.3.6.2 语言类“TeXt ”

MPLAB 编辑器在定义范围栏中,在一行中空格处自动换行。

**注解:** 当光标指在行尾时, MPLAB 编辑器只能约束此行。如果将光标在此行移至任何地方然后输入文件, MPLAB 编辑器将不能约束此行,甚至是已约束的卷。

### 5.6.4 编辑窗口模板

MPLAB 编辑器是结合在一套窗口模板下的每一个编辑窗口。适合的窗口会影响屏幕格式化。展现文本,输入,打印,归档。(详细资料请见 7.8.3 章和 7.8.4 的编辑模板)。



### 5.6.5 C 语言认知

当编辑文件设置在语言类“C”下，MPLAB 将提供以下功能：

- MPLAB 经常在其他空卷中移动“#”字符。
- 在空卷中，如果在空的一行中只有开括号，MPLAB 编辑器将自动在另一行添加闭括号。

列如：

```
//
*****
// EXAMPLE.C
//
*****
#include <PIC16C84.H>
void delay(void);
void main(void)
{
    unsigned int i,j;
    TRISB = 0xff;
    PORTB = 0;
    i = 0x1;
    while(1)
    {
        PORTB = i;
        if (i == 0x80)
            i = 0x1;
        else
            i <= 1;
        TRISB = 0;
        delay();
        TRISB = 0xff;
        delay();
    }
}
void delay(void)
{
    int x, y;
    x = 0x3f;
    y = 0xff;
    while(x--)
    {
        while(y--)
            NOP();
    }
}
```

## 第6章 调试和 MPLAB-SIM 模拟器

### 6.1 概述

本章介绍了 MPLAB 调试器的功能以及相应的 MPLAB-SIM 模拟器需要考虑的问题。你可以在模拟器（MPLAB-SIM）方式或者仿真器（MPLAB-ICE，PICMASTER<sup>®</sup> 仿真器，ICEPIC，MPLAB-ICD）方式下进行调试功能，参考《MPLAB-ICE 用户指南》以获取更多的关于使用仿真器进行调试的信息。

### 6.2 重点

本章将介绍以下内容：

- MPLAB IDE 调试功能
- 程序的实时运行
- MPLAB-SIM 模拟器环境
- 使用模拟器需要考虑的问题
- 断点与跟踪点
- 条件断点会话窗口
- “激励”功能
- 激励输出
  - 12-BIT 核心的芯片
  - 14-BIT 核心的芯片
  - 16-BIT 核心的芯片
  - 扩展型 16-BIT 核心的芯片

### 6.3 MPLAB IDE 调试功能

在 MPLAB-IDE 环境里面设置和编译好“项目”以后，你会想知道你的代码运行起来的效果。假如你有一个芯片烧写器，你可以用一块空白的芯片，将代码烧写到芯片里面，然后插入到用户目标板上，看看程序运行的效果是否和预料的一样。通常，一个应用程序不会第一次运行就一点问题都没有。你必须去调试代码。你可以使用 MPLAB-SIM 模拟器程序来模拟你的代码运行情况；也可以使用 MPLAB-ICE 仿真器来调试你的代码。

无论哪个方法调试，你都需要设置断点和跟踪点来调试程序代码。在寄存器窗口里面观察寄存器数值的变化情况；可以在特殊功能寄存器（FSR）窗口里面观察处理器的状态以及单步运行你的代码。

MPLAB-ICE 仿真器可以实时速度（real-time）调试你目标板上的程序代码，直到遇见指定的断点。MPLAB-SIM 模拟器程序来模拟任何 PICmicro 芯片的代码运行情况和 I/O 的变化状况。模拟的速度取决于你的 PC 机的运行速度。

以下的调试功能与模拟器或仿真器一样。主要的功能有：

- 仿真内存（程序存储器窗口）
- 断点和跟踪点
- 单步运行
- 寄存器监视（特殊功能寄存器 FSR 或文件寄存器窗口）

所有这些功能都使用 MPLAB IDE “项目”的配置信息。源文件里的行号，

内存里的符号位置，代码里的函数名都可以用来设置断点和跟踪点并用来检查和修改寄存器。

## 6. 4 “实时”执行程序

在文章里术语“实时”(real-time)通常只实用于仿真器(ICE)或在线调试器(ICD)。

### 6. 4. 1 在 MPLAB-SIM 模拟器模式下执行程序

当系统在实时模拟模式下运行的时候，指令的执行速度和 PC 机的速度有关。这通常要比处理器工作在全速方式下时要慢。

模拟器的运行速度取决于 PC 机 CPU 的运行速度以及 PC 机后台运行的程序数量。软件模拟必须随时更新所有被模拟的寄存器和 RAM，监控 I/O，置位和复位 FLAG，检查软件里的断点和跟踪点，并用 PC 机的指令来模拟 PICmicro MCU 系列处理器的指令。

注意：通常循环代码被用来进行延时功能。当使用模拟器的时候，你可能希望减少这些延时的长度或有条件地使用“IFDEF”来屏蔽某些程序段，从而提高模拟速度。

总之，当本手册里面提到“real-time”(实时)的时候，而且你是在用模拟器调试软件，这意味着软件模拟是在执行模拟的 PICmicro MCU 系列处理器的码，PC 机越快，模拟速度越快。

### 6. 4. 2 慢速模式 (Animate Mode)

慢速模式是一种自动的单步运行方式。当模拟器处于“运行”(Run)模式的时候实际上是在单步运行。但只有在处于暂停状态的时候才更新寄存器的状态。使用慢速模式可以动态地观察特殊功能寄存器(SFR)或观察窗口(WATCH)里寄存器的变化情况。慢速模式要比运行(RUN)模式慢。但却可以允许你观察变化着的寄存器值。

## 6. 5 MPLAB-SIM 模拟器环境

MPLAB-SIM 模拟器集成于 MPLAB-IDE 集成开发环境中，是 PICmicro MCU 系列处理器的软件模拟工具。MPLAB-SIM 模拟器为下列器件而设计：

- Microchip 出产的 PICmicro MCU 系列处理器比如：PIC12CXX，PIC14000，PIC16C5X，PIC16CXX，PIC17CXXX，PIC18CXXX。
- 帮助用户调试 Microchip 的 PICmicro MCU 系列处理器的应用软件。

离散事件模拟器与在线仿真器(比如MPLAB-ICE)相比，它是用来调试软件的。MPLAB-SIM允许你修改“项目”的代码并且立即执行。模拟器和仿真相比有如下三方面的不同：

- I/O时序
- 执行速度
- 开销成本

### 6. 5. 1 I/O时序

MPLAB-SIM模拟器的外部时序在每个指令周期里只进行一次处理。瞬变信号，比如MCLR上小于一个指令周期的尖峰将无法被模拟但可能被在线仿真器发现。

注意：在下一个指令周期到来之前必须先将“激励”注入到MPLAB-SIM里。

### 6. 5. 2 执行速度

非实时软件模拟器的执行速度比基于硬件的解决方案要慢的多。用户可以以较慢的速度观察程序的执行。MPLAB-SIM给用户尽可能快的模拟速度。根据工作模式的不同，可以工作于数毫秒一条指令。

### 6. 5. 3 成本开销

Microchip Technology提供的MPLAB-SIM模拟器是非常低成本的软件开发工具。用户不要任何除了PC机以外的硬件，而且很多的操作和MPLAB-ICE硬件仿真器相类似。除非你需要针对你的硬件进行实时的调试，MPLAB-SIM可以用来找出并修改大部分的代码错误。

### 6. 5. 4 调试工具

MPLAB-SIM模拟器特别适合于用来优化代码。和仿真器不同的是，模拟器可以观察很多的内部寄存器并且可以提供很多的软件调试工具，对于硬件仿真器来说很难做到，或者非常昂贵。很多时候MPLAB-SIM都可以用来调试应用软件，除非你的系统要求实时运行或外设要求必须实时运行。

## 6. 6 模拟器需要考虑的问题

MPLAB-SIM 在指令周期边沿模拟，并且当分辨率低于一个指令周期(TCY)的时候就无法模拟了。MPLAB-SIM 模拟器是一个“离散事件”(discrete-event)的模拟器，它可以根据输入激励的情况得出相应的输出响应。

MPLAB-SIM 模拟器是一种“离散事件”(discrete-event)模拟软件，在软件里考虑了所有的激励，并且在指令边沿(instruction boundaries)也就是  $TCY = 4 TOSC$  (TOSC 是输入时钟周期)的时候输出所有相应的响应。因而一些物理事件就无法精确地模拟。由此可能导致以下一些情形的事件(events)发生：

- 纯粹的异步事件。
- 比一个指令周期要短的事件。

总之，指令边沿模拟的结果是所有的事件在指令边沿得到同步，而小于一个指令周期的事件将会被忽略。

以下各项列出了所有的 PICmicro 系列微控制器的功能和外设由于指令周期边沿模拟所带来的影响。

- 输入时钟脉冲的宽度小于一个周期的时候将无法模拟—即使通过预分频器可以接受小于一个周期的输入脉冲。
- 不支持 PWM 输出脉冲的分辨率小于一个周期的情形。
- 不支持大于 8 位的比较
- 在非同步计数模式下，不支持小于一个周期的输入时钟。

- 振荡器两端 RC0/RC1 上的波形无法看见。
- MPLAB-SIM 模拟器无法模拟串行外设。

#### 6.7 断点和跟踪点

以下一些因数会在调试代码的时候影响程序指令的执行：

- 断点（Break Points）
- 跟踪点（Trace Points）
- “通过计数器”地址（Pass Counter Addresses）

MPLAB IDE 将可命名的地址范围最大定为每个会话窗口 16 个。

跟踪点和断点的功能是相互独立的，而且用户可以在程序存储器的任何位置来设置。

下图是给地址范围分配名字的会话窗口。通过使用命令：“**Debug > Break Settings**”可以进入断点设置窗口。通过使用命令：“**Debug > Trace Settings**”可以进入跟踪点设置窗口。

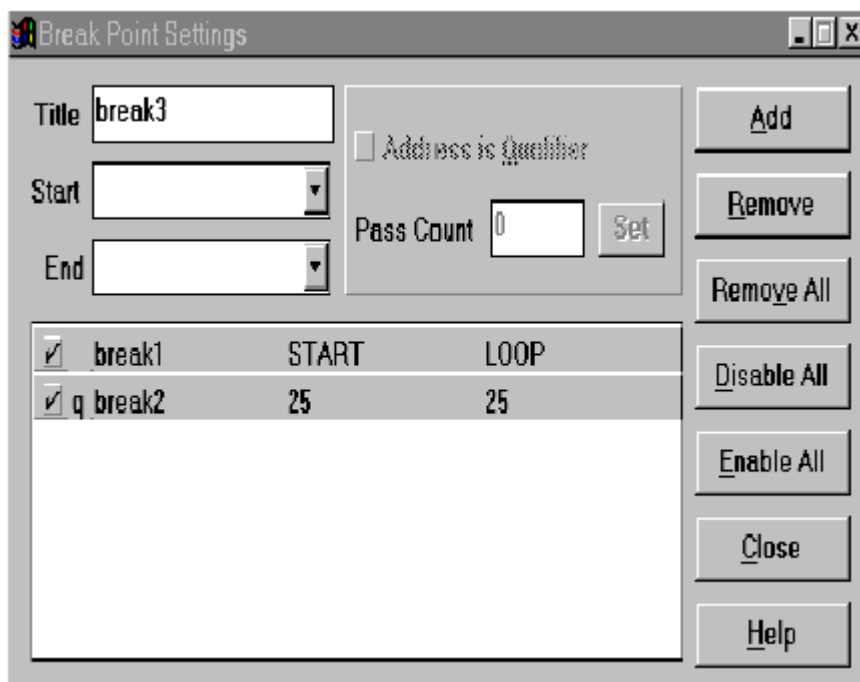


图 6-1：断点设置会话窗口

注意：MPLAB-ICD 允许只可以设置一个断点地址。

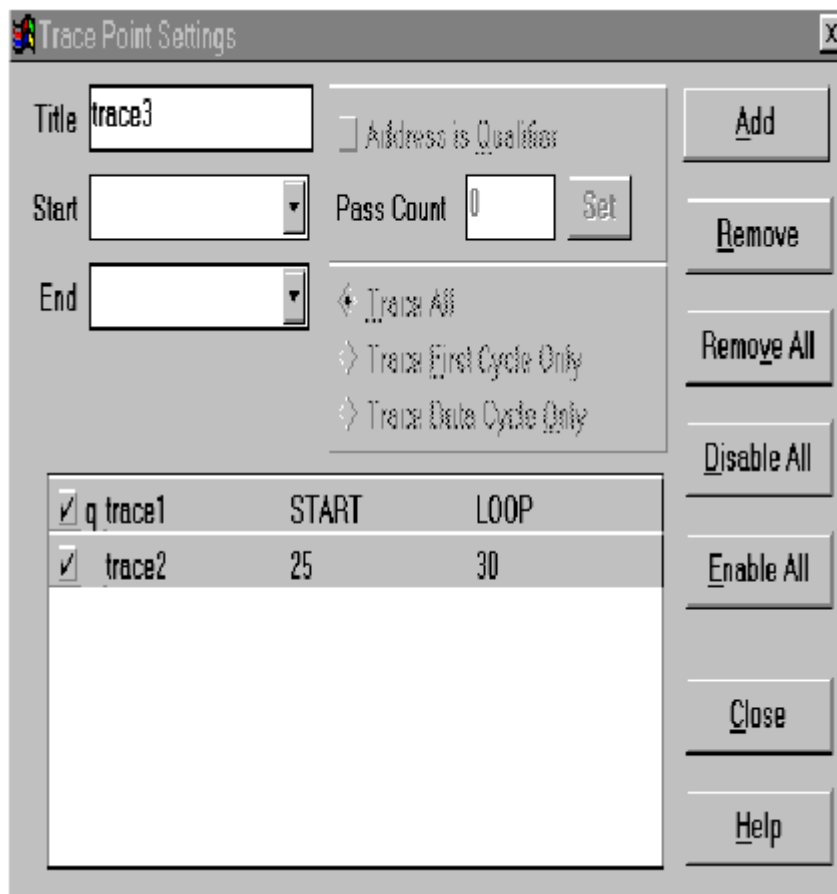


图 6-2: 跟踪点设置会话窗口

**注意:** 在 MPLAB ICE 或 MPLAB-ICD 里面没有跟踪点会话窗口

### 6. 7. 1 实时断点 (Real-Time Break Points)

当处理器执行程序遇到某一预先设置好的位置的时候就会暂停下来, 这一位置称为一个断点。

**注意:** 假如执行程序在断点处没有停下来, 可以使用命令: **Options > Development Mode** 然后点击 “**Break Options**” (断点选项) 即可。确信选中了 “**Global Break Enable**” (全局断点允许)。

MPLAB IDE 可以用下列方式来设置断点:

- 地址匹配断点 (Break on Address Match)
- 跟踪缓冲器满断点 (Break on Trace Buffer Full)
- 通过计数器满断点 (Break on Pass Count Reached)
- 堆栈满断点 (Break on Stack Overflow)
- 看门狗计数器溢出断点 (Break on Watch Dog Timer Time Out)
- 用户暂停 (User Halt)

图 6-3 显示的程序存储器窗口里面包含下列信息:

B - 断点 (Break Points)

T - 跟踪点 (Trace Points)

Q - “通过计数器” 地址 (Pass Counter Addresses)

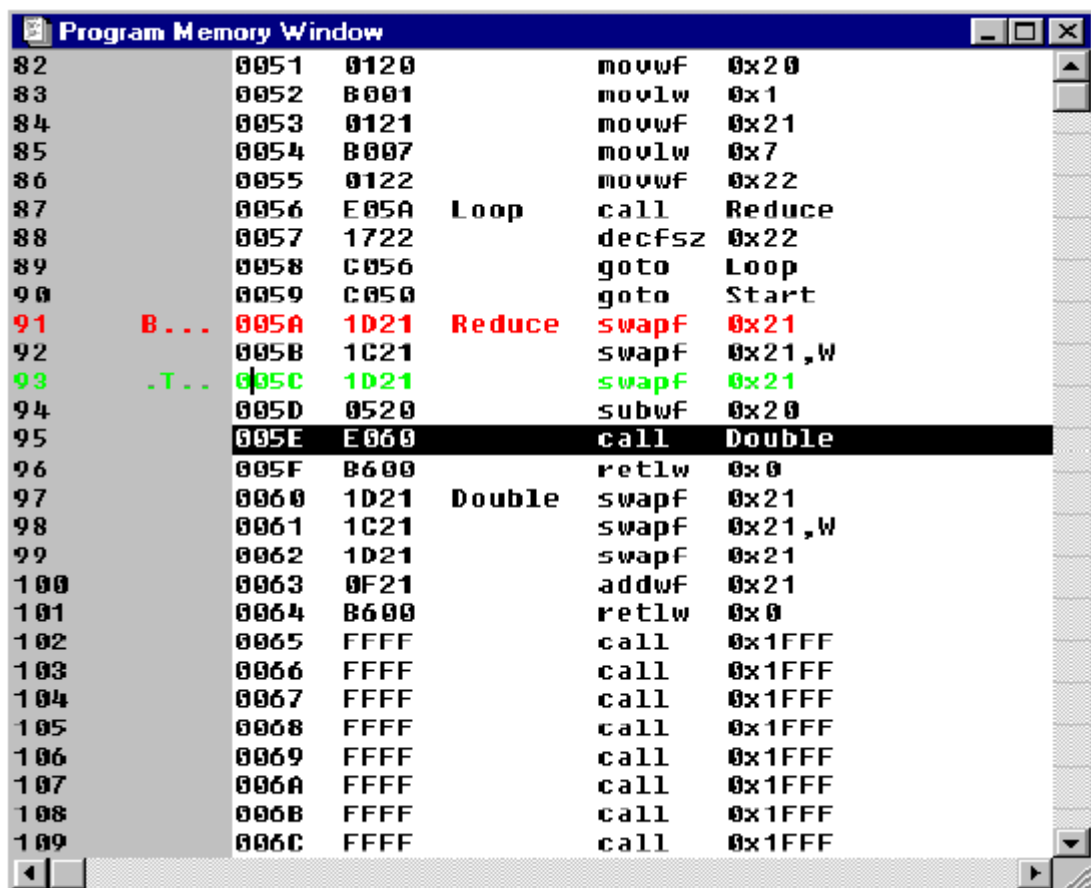


图 6-3: 程序存储器窗口

#### 6. 7. 1. 1 地址匹配断点

地址匹配断点允许用户在程序指针（PC）在和某一预先设定的数值匹配的时候将程序暂停下来。处理器在一条合法的指令被执行以后停止下来。例如：假如一个断点设置在 5AH，那末处理器会在执行完 5AH 处的指令后停止下来。

#### 6. 7. 1. 2 缓冲器满断点

MPLAB IDE 可以在捕捉了 8K 个预选周期（跟踪器缓冲满）后将处理器暂停下来。

#### 6. 7. 1. 3 当“通过计数器”和预设值相等断点

MPLAB IDE 里面设置了一个“通过计数器”（Pass Counter）开关，你可以分配给断点逻辑，也可以分配给跟踪逻辑。在一个预先设置好的时刻到达后对处理器中断或跟踪程序执行。

例如：假如计数器被分配给断点逻辑，那末当“通过计数器”的数值减到零的时候，“通过计数器”会作为一个断点，暂停处理器。

#### 6. 7. 1. 4 堆栈溢出断点

MPLAB 出 IDE 在堆栈溢出以后产生一个断点。

#### 6. 7. 1. 5 看门狗溢出端点

假如该选项处于允许状态，堆栈溢出断点在门狗计数器溢出

以后复位处理器的时候产生一个断点。

#### 6. 7. 1. 6 用户暂停

MPLAB IDE 在以下三种情形下任何时候会对处理器产生一个断点：

- 点击：“**Debug > Run > Halt**”
- 点击：“F5”
- 点击图标条里的“HALT”（暂停）按钮（红色灯）

#### 6. 7. 2 实时跟踪断点

跟踪器的功能是记录程序的执行信息。MPLAB-SIM 模拟器有 8K 深度的实时跟踪缓冲器，这些空间可以记录程序执行时候的操作码和相应的地址。这一环形跟踪缓冲器在缓冲器满以后还会不断地记录数据，将以前的数据覆盖掉（除非在“Break Options”菜单里面里选择了：“Break on Trace Buffer Full”选择项）。

##### 6. 7. 2. 1 环形跟踪缓冲器

MPLAB IDE 不断地捕捉预先设定好的总线周期，存储在跟踪缓冲器里面。这些状态信息在跟踪缓冲器里面按照以下方式排列：

- 16 位地址
- 16 位操作代码/数据
- 时标（Time Stamp）和发生过变化的寄存器。



图 6-4：跟踪存储器窗口



### 6. 7. 2. 2 从工具栏里暂停跟踪

暂停跟踪允许用户不用停止处理器的运行而查看跟踪缓冲器里面的内容。在工具栏里面，点击：“Halt Trace”就可以不用停止处理器的运行而观察当时的跟踪缓冲器里面的内容。一旦跟踪缓冲器被停止，再次点击：“Halt Trace”就可以开始另外一次的跟踪采样。

### 6. 7. 2. 3 MPLAB-SIM 模拟器跟踪显示

跟踪窗口可以用在 MPLAB-SIM 模拟器里收集被执行的指令。除了由 MPLAB ICE 仿真器显示的数据外，模拟器将还会在每一行显示时标（time stamp）并且回馈变化了的寄存器数据。时标和 MPLAB IDE 里的““跑表””（Stop Watch）有着相同的数据。用户可以通过重新设置“Stop Watch”来复位时标。

### 6. 7. 3 给断点和跟踪点分配“通过计数器”

MPLAB IDE 的 16 位“通过计数器”（Pass Counter）可以在当程序存储器里面遇到任何与设定地址匹配的地址的时候做减一操作。

当处理器处于“HALT”状态的时候，用户可以在“Break Point Settings”（断点设置）里面或者“Trace Point Settings”（跟踪点设置）修改“通过计数器”（pass counter）的数值。当设置的时候，首先设置所希望的地址范围，然后设置相应的计数值（可以多达 16 位）。当计数器归零的时候，仿真器将会暂停运行。

#### 6. 7. 3. 1 将“通过计数器”PASS COUNTER 分配给断点

假如“通过计数器”（Pass Counter）被分配给断点逻辑的时候，当遇到一个断点（无论内部还是外部断点）或者“通过计数器”Pass Counter 减到零的时候，处理器就会暂停下来。

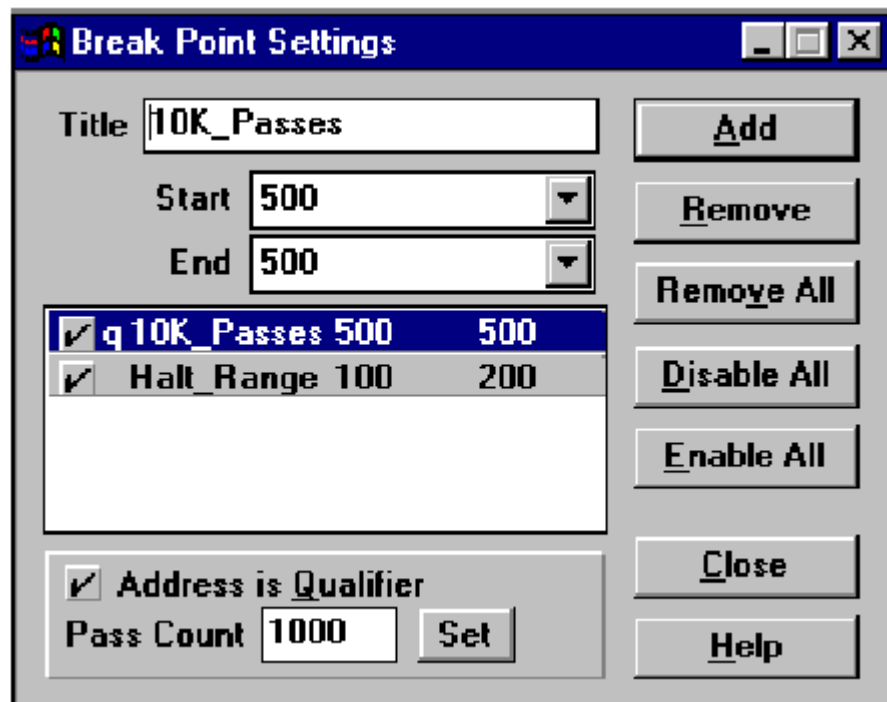


图 6-5: 断点设置会话窗口 - Pass Counter

**例 6-1:** 本范例显示了在相同的代码（图 6-5）里设置的断点和“通过计数器”pass counter 地址。记住：断点和“通过计数器”pass counter 地址之间是相互独立的。

1. 设置一个已经命名的从地址 100 到 200 的断点。
  - 在标题栏（Title box）里写上：Halt\_Range;
  - 在开始栏（Start box）里写入数值：100，结束栏（End box）里写入数值：200;
  - 点击：“”按钮，输入断点数据。
2. 将“通过计数器”Pass Counter 的地址设置为：500
  - 在标题栏里写入：10K\_Passes;
  - 在开始栏和结束栏里写入：500;
  - 点击：“Add”按钮，输入断点数据。
3. 给“通过计数器”Pass Counter 输入一个数值：1000
  - 选择（点击）断点：“10K\_Passes”
  - 在现在还没有变灰的“Address is Qualifier”左边的选择盒里点击选中。
  - 在“通过计数器”Pass Count 窗口里面输入数值：1000，然后点击：“Set”

当处理器执行了任何位于 100 到 200 之间地址里的程序代码，或者执行了地址 500 处的代码 1000 次以后，处理器会暂停下来。

#### 6. 7. 3. 2 分配给跟踪器的“通过计数器”Pass Counter

当 Pass Counter 被分配给跟踪器以后，一直到 Pass Counter 的数值减到零为止，实时跟踪缓冲器将不会捕捉数据。当 Pass Counter 的数值减到零以后，实时跟踪缓冲器将开始在合法的周期里捕捉数据。

#### 6. 7. 3. 3 使用“通过计数器”Pass Counter 来记录事件

当事件发生的时候“通过计数器”Pass Counter 将会做减一操作。用户可以使用这一特性来记录某一事件发生的次数。

### 6. 8 条件断点会话窗口

当设置了条件断点以后，MPLAB IDE 可以在内部寄存器的数值到达某一预先设置的数值或情形的时候暂停工作。

使用菜单命令：“***Debug > Execute > Conditional Break***”可以操作条件断点会话窗口。

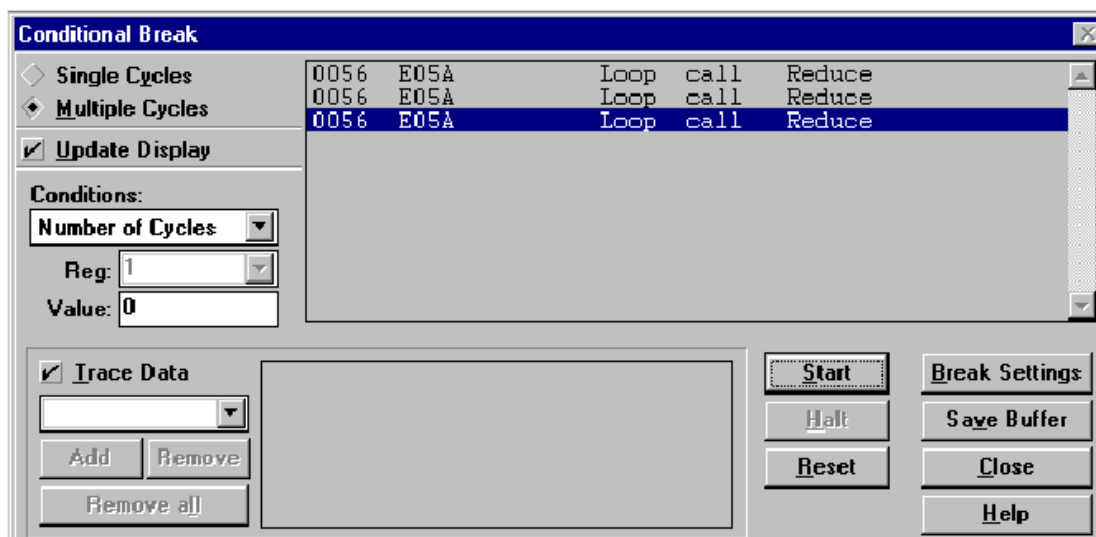


图 6-6: 条件断点会话窗口

注意：在 MPLAB-ICE 或 MPLAB-ICD 开发模式下没有条件断点会话窗口。

#### 6. 8. 1 条件

在以下这些情形时，MPLAB IDE 集成开发环境将会终止程序的运行，停在一个断点处。

- **User Halt** – 在条件断点会话窗口里，MPLAB IDE 会一直执行程序，直到用户按：“Halt”按钮才会停止。
- **Number of Cycles** – 当目标处理器执行了预先指定数目的周期之后暂停程序的执行。
- 满足逻辑条件。

#### 6. 8. 2 跟踪数据

跟踪数据允许用户在条件断点会话窗口里追踪某一寄存器的数值。

#### 6. 8. 3 单周期

在单周期模式下，MPLAB IDE 单不执行指令知道遇见设置的条件。

#### 6. 8. 4 多周期

在多周期模式下：

- 条件断点以实时（仿真器）模式执行指令，在用户指定的断点处暂停，检查指定的条件是否满足，然后继续以实时模式执行程序。只有当遇见预先设置的条件满足的时候仿真器或模拟器才会终止程序的运行。
- 只有用户在断点设置会话窗口（Break Settings）里设置的断点才会被检查

## 6.9 “激励”功能

“激励”为模拟器提供信号。用户可以设置管脚电平的高和低，给寄存器直接注入数值。有以下四种激励模式：

- 异步激励 – 一个交互会话模式，用来控制输入管脚上的信号。
- 管脚激励文件 – 一个文本文件，包含了对输入管脚上信号的描述。
- 寄存器激励文件 – 一个文本文件，包含了直接赋予寄存器的 8 位数值。
- 时钟激励文件 – 一个常用的，可编程的，周期性的脉冲激励源。

### 6.9.1 异步激励会话窗口

这一激励可以通过一个按钮来模拟加在管脚上的电平是+5V 还是 0V。当你的程序按照模拟器运行的时候，你可以点击这一按钮来改变管脚上电平的类型。

作为一个范例，我们假设有一个处理器是 PIC16F84，现在可以来设置一个信号，改变 I/O 口某一个管脚上的电平。

选择菜单命令：“**Debug > Simulator Stimulus > Asynchronous Stimulus**”，这时候将会看见下面显示的会话窗口：

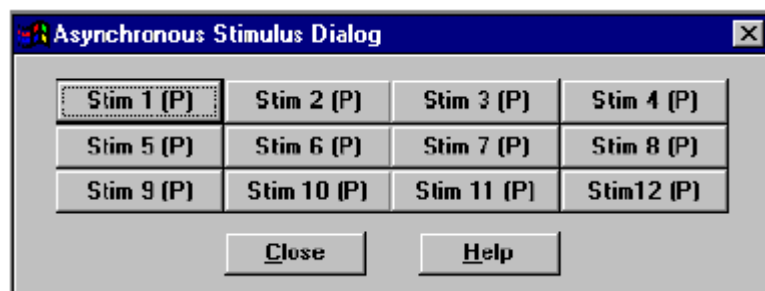
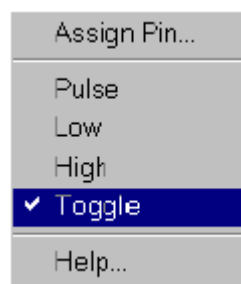


图 6-7：异步激励会话窗口

将光标放在标记有：“**Stim1 (P)**”的地方并且点击鼠标的右键，将会看见一个快捷菜单。往下卷动并选择：“***Toggle***”。



6-8：切换选择（Toggle Option）

然后将光标放置在标记有：“**Stim1 (T)**”的地方（这时候“P”被“T”所代替，意思是“**Toggle**” - 切换）。点击鼠标的右键并在快捷菜单里选择：“**Assign Pin**”

这时候将会出现一个会话窗口，列出了所有 PIC16F84 处理器的管脚：

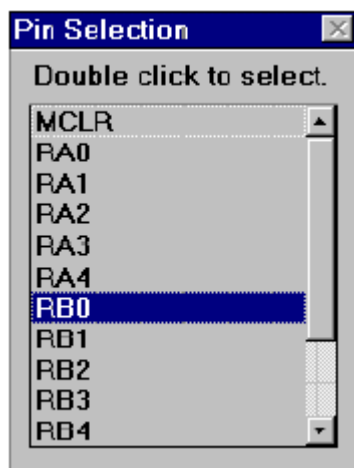


图 6-9: 管脚选择

将光标放置在：“RB0”上并双击鼠标。异步激励窗口将会象图 6-10 显示的那样。需要指出的是，现在按钮显示的是：“RB0 (T)”。

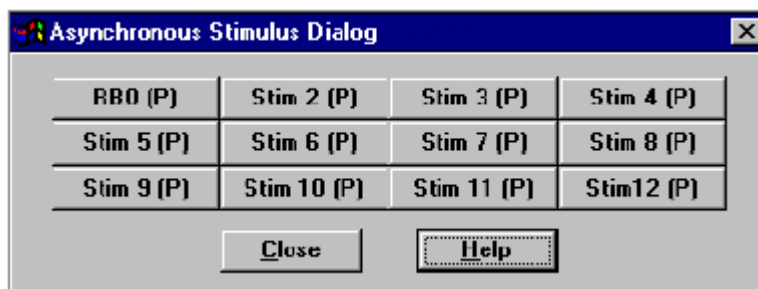


图 6-10: 异步激励会话窗口 – RB0 (T)

选择菜单命令：“***Debug > Run > Animate***”使处理器工作于一种“快速单步”模式下。状态栏将会很快的在“RUN”和“STOP”之间切换。

在异步激励会话窗口里点击：“RB0 (T)”按钮。用户可以看见当不断地对 PORTB 的管脚 0 进行高和低激励的切换模拟时，特殊功能寄存器 (SFR) 窗口里 PORTB 的数值会相应地变化。

### 6. 9. 2 管脚激励文件

管脚激励文件包括了输入的“1”或“0”信号行，这些信号在当“跑表” (STOPWATCH) 里的“周期”数值和 CYCLE 行的数值匹配的时候被加到管脚上。

#### 6. 9. 2. 1 建立一个管脚激励文件

1. 选择菜单命令：“***File > New File***”。这时候用户桌面上将会显示一个未命名的窗口。你可以在这个窗口里面建立自己的管脚激励文件。
2. 在未命名文件窗口里面的第一行里键入：“CYCLE”字样。

**注意：**为了和以前版本的模拟器相兼容，第一行必须以“CYCLE”或“STEP”开始。第一栏指定为“CYCLE”开始（由 MPLAB IDE 里的“Stopwatch”窗口设置），在其他的栏里将可以输入其相应的数值。

3. 在字符：“CYCLE”的右面，键入需要施加高电平激励的 PICmicro 系列微处理器的管脚名称。第一行的第三项也是最后一项是将要接受低电平激励的管脚名称。这些名称应该符合将要激励的 Microchip PICmicro 系列微控制器的管脚定义规范。

**注意：**要想看软件支持的管脚定义，使用命令：“**Debug > Simulator Stimulus > Asynchronous**”并且在一个激励按钮上点击右键。

4. 在文件余下的行里，键入管脚接受激励的周期数，紧跟着输入相应的激励电平。可以在行里添加注解，注解必须用“;”或“!”符号开始并至少跟一个空格。

5. 选择命令：“**File > Save As...**”来保存文件。选择好文件需要存储的目录和驱动器号，并且输入一个文件名。文件的扩展名用：“.sti”。现在你的文件可以使用了。

#### 6-9-2-2 使用管脚激励文件

1. 选择命令：“**Debug > Simulator Stimulus > Pin Stimulus > Enable**”允许使用管脚激励文件。
2. 使用命令：“**Window > Stopwatch**”打开“跑表”（STOPWATCH）窗口。使用命令：“**Window > Special Function Registers**”打开 SFR 窗口，观察管脚 打开的端口。或者简单的将打开管脚的端口添加到观察窗口里面。”跑表”（STOPWATCH）窗口将会显示出执行每个命令耗费的时间，时间由 CYCLE 值以及时钟频率决定。假如“跑表”（STOPWATCH）被复位为零，管脚激励文件也会有效的复位。
3. 复位并单步运行。端口将会根据用户设置的激励而相应得改变。

#### 6-9-2-3 管脚激励文件范例

**注意：**本范例假设用户已经完成了第三章介绍的建立简单“项目”的范例

1. 选择命令：“**File > New File**”并输入相应得文本。用户不必在“;”和“!”后面输入文字，但在文件里面保留这些符号是个号习惯。

```

CYCLE RB1 RB0
20    0    0
41    1    0    ;在port b bit 1 上加高电平
52    0    1    ;在port b bit 0上加高电平，在bit 1 上加低电平
55    1    1
60    0    0
65    1    0    ; toggle bit 1, 然后...
76    0    1    ! ...toggle bit 0.

```

在文件第一行的 CYCLE 符号后面是将要接受高或低电平激励的

PICmicro 系列微处理器的管脚名称。在这个范例里面，端口 B 上的两个管脚 RB1 和 RB0 将接受输入激励。

**注意：** 为了和以前的老版本模拟器相兼容，

在这个文件里，第二栏里面包涵了要加在 RB1 (PortB bit 1)上的数值，第三栏里面包涵了要加在 RB0 (PortB bit 0)上的数值。这些名称必须和将要模拟的 PICmicro 系列微处理器的定义一样。想看软件支持的定义管脚定义，在一个激励按钮上点击右键，察看管脚定义下拉菜单里的异步激励。

2. 选择命令：“**File > Save As**”将文件存为：tutor84.sti
3. 选择命令：“**Debug > Simulator Stimulus > Pin Stimulus > Enable**”允许管脚激励文件。

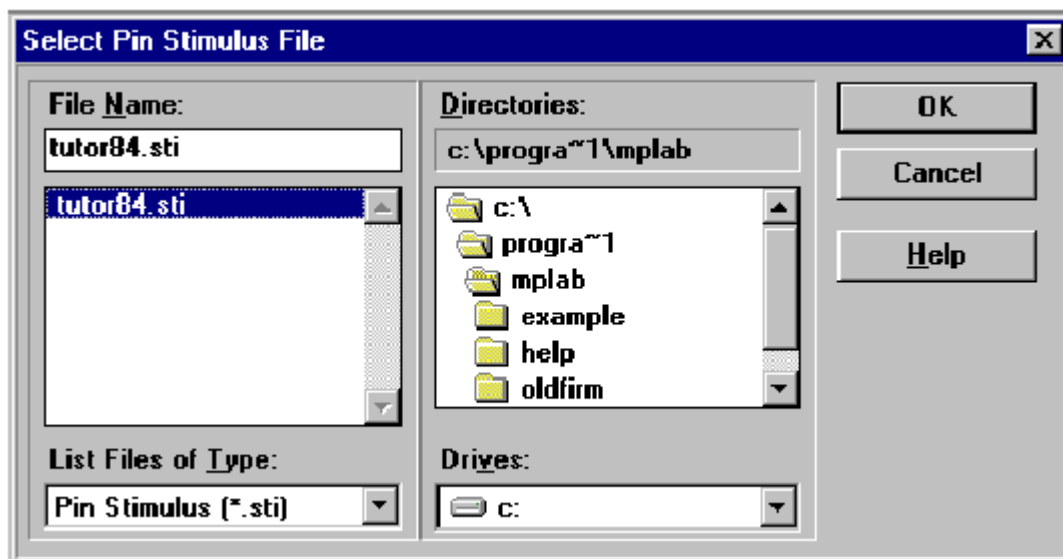


图 6-11：允许管脚激励

4. 使用命令：“**Window > Stopwatch**”打开“跑表”（STOPWATCH）窗口。然后选择命令：“**Window > Special Function Registers**”打开 SFR 窗口，然后观察 Portb 的数值。或者简单的将 Portb 加入到观察窗口里面。  
“跑表”（STOPWATCH）窗口将还会显示每个指令所耗费的时间，这个时间是由激励文件里的 CYCLE 数值和时钟频率所决定的。假如“跑表”（STOPWATCH）被复位到 0，那么激励文件也会被有效的复位。
5. 复位处理器并单步运行程序直到执行了 41 个周期。PORTB 将根据激励文件里第二行定义的方式改其变数值。



图 6-12: “跑表” (STOPWATCH) 窗口(41 个周期)

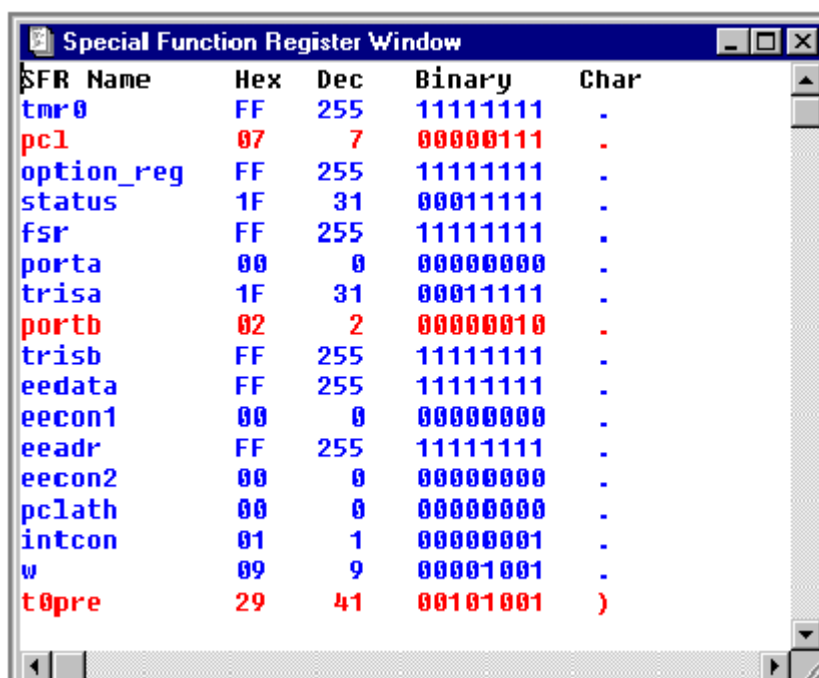


图 6-13: 特殊功能寄存器窗口

### 6-9-3 寄存器激励文件

寄存器激励文件由一个单独的栏组成，当寄存器激励会话窗口里设置的数值与程序存储器指针(PC)匹配的时候，其数值将会赋予到一个寄存器里面。这种方式可以很好的模拟 AD 转换操作。

#### 6-9-3-1 建立一个寄存器激励文件

- 1 选择命令: “**File > New File**” 来建立一个未命名的文件。运行可以在这个窗口里面建立自己的寄存器激励文件。
- 2 在这个未命名窗口里面，输入用户想注入一个寄存器的数值。确认按照你希望注入的顺序在激励文件里排列寄存器的数值。
- 3 选择命令: “**File > Save As**” 保存文件。选择文件保存的目录和驱



动器盘符然后输入文件名。文件的扩展名应该取为：“.reg”

4

#### 6-9-3-2 使用寄存器激励文件

- 1 选择命令：“**Debug > Simulator Stimulus > Register Stimulus > Enable**” 打开寄存器激励会话窗口。
- 2 在程序存储器地址窗口里面，输入需要注入激励的程序地址。
- 3 在寄存器地址窗口里面，输入需要注入激励的文件寄存器地址。
- 4 选择命令：“**Window > File Registers**” 打开文件寄存器窗口并且察看激励效果。
- 5 复位并单步运行程序，每次当到达程序存储器地址的时候，用户指定地址的文件寄存器数值将会改变。在寄存器激励文件里列出的数值将会顺序地注入用户所选择的寄存器里。  
当寄存器激励文件里的最后一个数值被注入以后，激励文件里的第一个数值将会被再次使用，只要 MPLAB SIM 在运行就会不断循环执行列表里的激励。

#### 6-9-3-3 寄存器激励范例

**注意：**本范例假设用户已经完成了第三章里“项目”的建立范例。

- 1 使用命令：“**File > New File**” 建立一个新文件并输入以下数值：  
10  
2E  
38  
41  
50  
7A  
99  
A0  
FD
- 2 选择命令：“**File > Save As**” 将文件保存起来，并取名为“tutor84.reg”。这个文件将会被模拟器用来将相应的数值顺序的注入寄存器里面。
- 3 选择命令：“**Debug > Simulator Stimulus > Register Stimulus > Enable**” 然后在程序里“LOOP”所在的地方设置为需要入数据的地方，为了方便示范，我们在地址 0x0d 的地方将这些数值注入到文件寄存器里面。当你在相应的窗口里面设置了“loop”和“0d”以后，点击“浏览”，可以看见文件会话窗口，然后选择 tutor84.reg 作为寄存器激励文件。

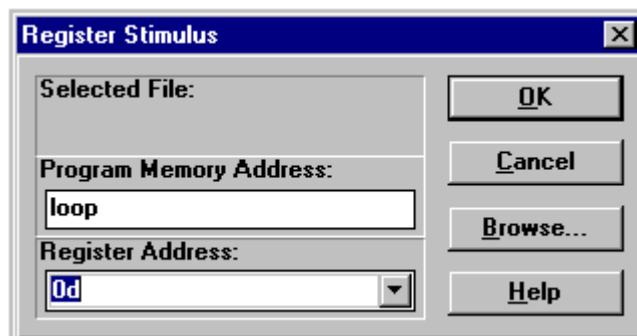


图 6-14: 寄存器激励会话窗口

- 4 选择命令: “**Window > File Registers**” 打开文件寄存器窗口并察看激励的效果。

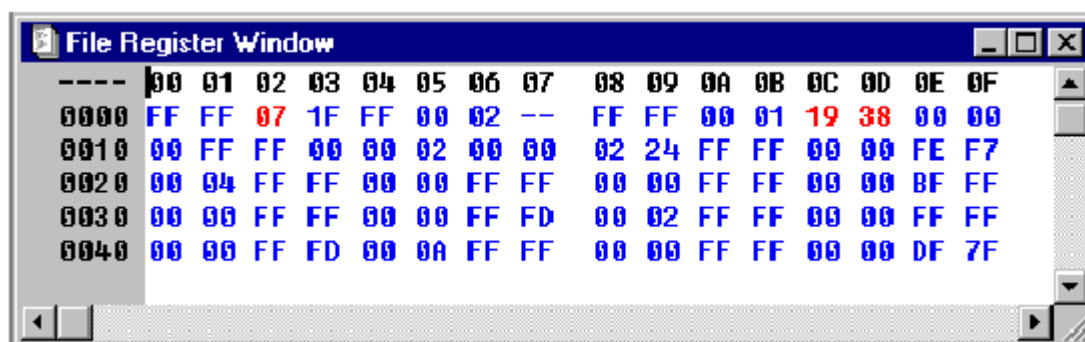


图 6-15: 文件寄存器窗口

- 5 复位并单步运行处理器。每次程序到达 LOOP 的时候, 在地址 0X0D 处的文件寄存器的数值将会改变。在文件 tutor84.reg 里列出的数值将会顺序的注入到被选择的文件寄存器里面。每次执行到 LOOP 的时候, 0x10, 0x2E 等的数值将会注入到使用命令 **Debug > Simulator Stimulus > Register Stimulus** 设置的寄存器里。

当最后一个数值被注入以后 (在文件 tutor84.reg 里的 0xFD), 表里的第一个数值 (0x10) 将会被重复使用。只要 MPLAB SIM 在运行, 这一列表将会循环使用。

#### 6-9-3-4 时钟模拟

时钟激励将会在一个管脚上产生一个规则的有一定占空比得周期性的波形, 这一波形模拟处理器的时钟周期波形。

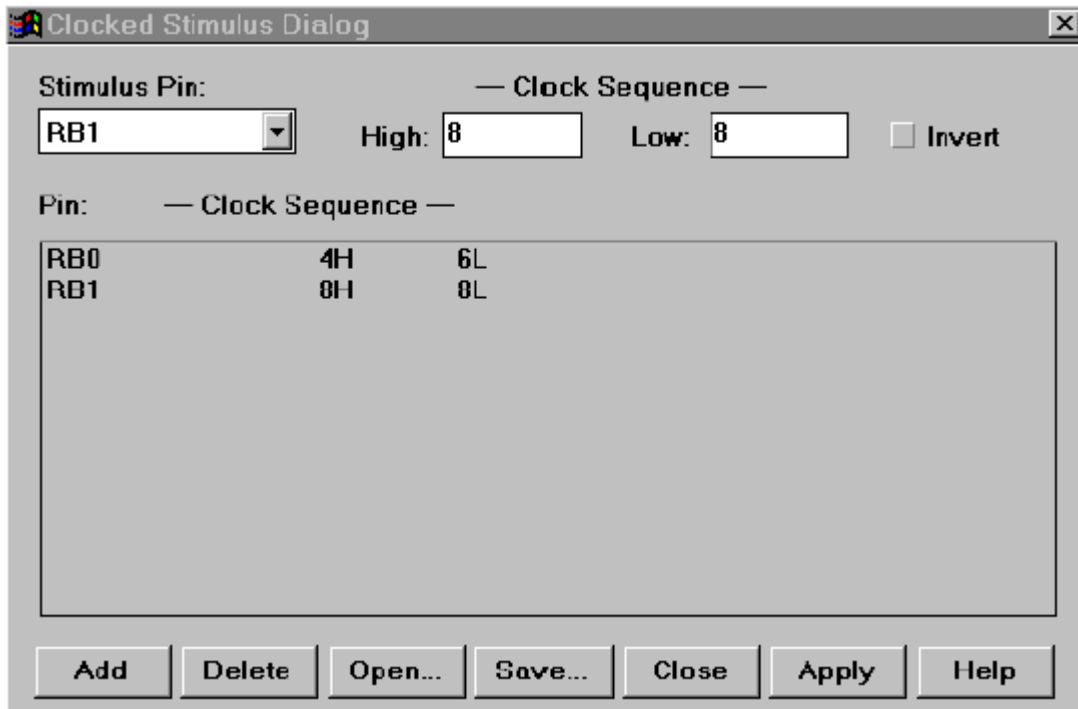


图 6-16: 时钟模拟

选择命令: “***Debug > Simulator Stimulus > Clock Stimulus***” 在时钟激励(Clocked Stimulus)会话窗口里 为各种激励时钟输入时钟序列。在用户退出 MPLAB IDE 或在时钟激励会话窗口里删除掉这些设置之前, 这些设置将不断重复。

当用户单步运行或用图 6-16 里的设置运行的时候, RB0 将维持 4 个时钟周期的高电平, 然后保持 6 个时钟周期的低电平。RB1 将维持 8 个时钟周期的高电平, 然后维持 8 个时钟周期的低电平。

假如想增加一个激励, 可以在激励管脚(Stimulus Pin)下拉菜单里选择一个管脚, 设置相应得高和低时钟序列, 然后点 “Add”。

想删除一个激励, 可以点击并高亮某一个激励然后点 “Delete”。

## 6-10 12 位核芯片的模拟

本节介绍了基于 12 位核处理器的 IO 管脚, 中断系统, 寄存器, 外设, 工作模式, 条件等。

### 6-10-1 基于 12 位核的处理器系列

在本文写作的时候, 已经有以下 PIC16C5X 系列处理器可供使用。

- PIC12C508/509
- PIC12CE518/519
- PIC16C52/54/55/56/57/58
- PIC16HV540
- PIC16C505

以上列表里隐含了一些系列芯片, 比如 ROM 版本(PIC16CR5X), 以及版本更新(PIC16C5XA)。

### 6-10-2 IO 管脚

不管是以手动方式还是通过激励文件的方式修改管脚，只能使用以下的管脚名称。只有这些是 MPLAB SIM 模拟器可以识别的 IO 管脚。因为管脚分配是因型号不同而不同的，一些管脚(比如 PIC16C54 芯片上的 RC0 )在这一系列里是没有的。

- MCLR
- T0CKI
- RA0-RA3
- RB0-RB7
- RC0-RC7

这些管脚名称可以在修改窗口( **Window > Modify**)和激励文件里使用。

### 6-10-3 CPU 模型

#### 6-10-3-1 复位和睡眠状态

MPLAB SIM 模拟器支持所有的复位状态。

处于正常操作状态下或处于 SLEEP 状态下的复位 MCLR 操作都可以简单地模拟：通过激励文件或使用菜单命令：“**Debug > Run > Reset**”将 MCLR 驱动到低电平(然后拉到高电平)。

当看门狗(WDT)被允许并设置了合适的预分频(初始化相应得 OPTION 寄存器)而且 WDT 确实溢出了就可以模拟 WDT 溢出复位。WDT 的基本溢出时间为 18 毫秒(预分频=1)。

状态寄存器里的 Time-out (TO) 和 Power-down (PD)反应了相应的复位条件。这个特性有利于模拟用户代码里的各种上电(power-up)和时间到(time out)的情形。

#### 6-10-3-2 看门狗定时器

MPLAB SIM 模拟器可以完全模拟看门狗定时器。因为在芯片上这是用熔丝编程的，必须在开发模式会话窗口里的配置选择盒里选择“允许”，操作过程是：在 MPLAB SIM 模拟器状态下选择命令“**Options > Development Mode**”。WDT 的溢出周期由 OPTION 寄存器里设置的预分频值决定。WDT 的基本溢出时间为 18 毫秒(预分频=1)，(最接近指令周期的倍数)。

### 6-10-4 外设

除了提供核心支持，定时计数器 0(TIMER0)模块可以工作于内部或外部时钟模式。预分频器可以被读和写，识别符号为“T0PRE”。

**注意：**因为 MPLAB-SIM 模拟器在指令周期边际执行，分辨率低于一个时钟周期将无法模拟。

## 6-11 14 位核芯片的模拟

本节介绍了基于 14 位核处理器的 IO 管脚，中断系统，寄存器，外设，工作模式，条件等。

### 6-11-1 14 位核芯片

在本文写作的时候，已经有以下 14 位核芯片可供使用：

- PIC12C671/672
- PIC12CE673/674
- PIC140000
- PIC16C62/62A/62B/63/63A/64A/65A/65B/66/67
- PIC16C71/72/72A/73A/73B/74A/74B/76/77
- PIC16C554/558
- PIC16C620/621/622
- PIC16C642/662
- PIC16C710/711/715
- PIC16C712/716
- PIC16C717
- PIC16C770/771
- PIC16C773/774
- PIC16C923/924
- PIC16CE623/624/625
- PIC16F83/84/84A
- PIC16F872/873/874/876/877

以上列表里隐含了一些系列芯片，比如 ROM 版本(PIC16CRXX)，以及版本更新(PIC16CXXA)。

### 6-11-2 IO 管脚

14 位处理器的 IO 管脚有和其他外设复用的(因此会有不同的名称定义)。不管是以手动方式还是通过激励文件的方式修改管脚，只能使用以下的管脚名称。只有这些是 MPLAB SIM 模拟器可以识别的合法 IO 管脚(不同的芯片的管脚可能不同，只能依照数据手册里的定义为标准)。

- MCLR
- RA0-RA5
- RB0-RB7
- RC0-RC7
- RD0-RD7
- RE0-RE7

这些管脚名称可以在修改窗口( **Window > Modify**)和激励文件里使用。

### 6-11-3 中断

MPLAB SIM 模拟器支持所有的 14 位核处理器的中断。(不同的芯片的管脚可能不同，只能依照数据手册里的定义为标准)

- Timer0 溢出
- Timer1 溢出

- Timer2
- CCP1中断
- CCP2中断
- SSP (只能SPI 模式)中断
- 端口B Port RB <7:4 >上的电平变化中断
- 从 RB0/INT 管脚上来的中断
- 从模式并行口中断
- 比较器中断
- A/D 转换结束中断
- 写 EEPROM 结束中断

#### 6-11-4 CPU 模式

##### 6-11-4-1 复位的情形

MPLAM SIM 模拟器支持所有的复位状态。

处于正常操作状态下或处于 SLEEP 状态下的复位 MCLR 操作都可以简单地模拟：通过激励文件或使用菜单命令：“***Debug > Run > Reset***” 将 MCLR 驱动到低电平(然后拉到高电平)。

当看门狗(WDT)被允许并设置了合适的预分频(初始化相应得 OPTION 寄存器)而且 WDT 确实溢出了就可以模拟 WDT 溢出复位。WDT 的基本溢出时间为 18 毫秒(预分频=1)。

状态寄存器里的 Time-out (TO) 和 Power-down (PD)反应了相应的复位条件。这个特性有利于模拟用户代码里的各种上电(power-up)和时间到(time out)的情形。

##### 6-11-4-2 睡眠

MPLAB-SIM 模拟器可以模拟睡眠(SLEEP)指令，直到将处理器从睡眠模式唤醒的事件发生，将一直保持睡眠“asleep”状态。例如：假如看门狗定时器被允许，到它溢出的时候将会将处理器从睡眠状态唤醒(溢出时间取决于 OPTION 寄存器里的预分频系数)。

另外一个从睡眠状态唤醒处理器的例子是，定时器 Timer1 从睡眠状态的唤醒。假如中断被允许，当定时器溢出的时候将唤醒处理器并转向对应的中断向量。

##### 6-11-4-3 看门狗定时器(WDT)

MPLAB SIM 模拟器可以完全模拟看门狗定时器。因为在芯片上这是用熔丝编程的，必须在开发模式会话窗口里的配置选择盒里选择“允许”，操作过程是：在 MPLAB SIM 模拟器状态下选择命令“***Options > Development Mode***”。WDT 的溢出周期由 OPTION 寄存器里设置的预分频值决定。WDT 的基本溢出时间为 18 毫秒(预分频=1)，(最接近指令周期的倍数)。

##### 6-11-5 特殊功能寄存器

为了帮助调试芯片，通常芯片里无法观察到的一些寄存器被定义为特殊功

能寄存器。预分频器和后分频器是不能在代码里定义为寄存器的，所以这些标号会出现在特殊功能寄存器(SFR)窗口里面。

以下是基于 14 位核的处理器里特殊功能寄存器的特殊符号。(查阅相应的数据手册确定哪些符号是可以使用的)

- **T0PRE** – timer0的预分频器
- **T1PRE** – timer1的预分频器
- **T2PRE** – timer2的预分频器
- **T2POS** – timer2的预分频器
- **CCP1PRE** – CCP1的预分频器
- **SPIPRE** – SPI的预分频器
- **SSPSR** – SSP 移位寄存器

## 6-11-6 外设

### 6-11-6-1 支持的外设

除了提供核心支持，同时也支持以下的一些外设模块(IN addition to general-purpose I/O)。(查阅相应的数据手册确定哪些符号是可以使用的)

- **Timer0**
- **Timer1**
- **Timer2**
- **CCP1**
- **CCP2**
- **从模式并行口**
- **SSP (只能SPI 模式)**
- **比较器**
- **A/D (有限)**

### 6-11-6-2 定时器 0(TIMER0)

MPLAB-SIM 模拟器可以完全支持 Timer0(可以产生溢出中断)，它可以在内部或外部脉冲的推动下作加一操作。鉴于模拟器的要求，输入脉冲的高电平宽度必须不低于一个 TCY，低电平也不能低于一个 TCY。可以通过寄存器 T0PRE 操作 Timer0 的预分频参数。

### 6-11-6-3 定时器 1(TIMER1)

除了工作于外部晶体的计数器模式，MPLAB-SIM 模拟器可以支持定时器 Timer1 的各种工作模式。MPLAB-SIM 模拟器支持 TIMER 物溢出产生的中断和从睡眠状态唤醒所产生的中断。用户可以通过特殊功能寄存器(SFR)窗口里的 T1PRE 观察 Timer1 的预分频参数。模拟器无法模拟外部振荡器 RC0/RC1，但用户可以加时钟激励在这些管脚上。

### 6-11-6-4 定时器 2(TIMER2)

MPLAB-SIM 模拟器完全支持 Timer2 和它的溢出中断，用户可以通过特殊功能寄存器 T2PRE 和 T2POS 观察 Timer2 的预分频器和后分频器的数值。

**6-11-6-5 CCP1 和 CCP2****捕捉(CAPTURE):**

MPLAB-SIM 完全支持捕捉功能及其产生的中断。用户可以通过特殊功能寄存器 CCP1PRE 观察 CCP 模块的预分频参数。

**比较器(COMPARE)**

这一版本的 MPLAB-SIM 完全支持比较器模式及其中断和特殊事件触发器(通过 CCP1 来复位 Timer1)。

**脉宽调制(PWM)**

这一版本的 MPLAB-SIM 模拟器还无法模拟 PWM 输出(只适应于分辨率大于一个 TCY )。

**6-11-6-6 SSP**

同步串行口只支持 SPI 模式。移位寄存器(SSPSR)可以被加入到观察窗口里面, 可以被观察和修改。MPLAB-SIM 目前尚不支持 I2C 串行工作模式。

**6-11-6-7 AD 转换器**

所有的寄存器, 时序功能和中断都可以完成模拟。然而, AD 转换结束后将没有任何有意义的数值存入 AD 转换结果寄存器(ADRES)里。

**6-11-6-8 EEPROM 数据存储器**

模拟器可以完全模拟 EEPROM 数据存储器(PIC16F8X 系列)。完全可以模拟寄存器和读写周期。写周期时间大约为 10 毫秒(最接近指令周期的倍数)。

模拟器可以模拟 EECON1 寄存器里的 WRERR 和 WREN 控制位。

**6-12 16 位核处理器的模拟**

本节介绍了基于 16 位核处理器的 IO 管脚, 中断系统, 寄存器, 外设, 工作模式, 条件等.

**6-12-1 基于 16 位核的处理器系列**

在本文写作的时候, 已经有以下系列的基于 16 位核的处理器可供使用:

- PIC17C42/43/44
- PIC17C752/756
- PIC17C762/766

以上列表包含了一些衍生系列型号, 比如基于 ROM 版本的芯片(PIC17CRXX)和更新版本的芯片(PIC17CXXA)。

**6-12-2 IO 管脚**

16 位处理器的 IO 管脚有和其他外设复用的(因此会有不同的名称定义)。不管是以手动方式还是通过激励文件的方式修改管脚, 只能使用以下的管脚名称。只有这些是 MPLAB SIM 模拟器可以识别的合法 IO 管脚(不同的芯片的管脚可能不同, 只能依照数据手册里的定义作为标准)。

- MCLR
- RA0-RA5
- RB0-RB7



- RC0-RC7
- RD0-RD7
- RE0-RE2

这些管脚名称可以在修改窗口( **Window > Modify**)和激励文件里使用

### 6-12-3 中断

MPLAB-SIM 模拟器支持所有 16 位核处理器的中断系统:

- INT 管脚上的外部中断
- TMR0 溢出中断
- RA0 管脚上的外部中断
- 端口Port B 变化中断
- Timer/Counter1 中断
- Timer/Counter2 中断
- Timer/Counter3 中断
- Capture1 中断
- Capture2 中断

### 6-12-4 CPU 模型

#### 6-12-4-1 复位的情形

MPLAB SIM 模拟器可以模拟所有的复位情形。

处于正常操作状态下或处于 SLEEP 状态下的复位 MCLR 操作都可以简单地模拟: 通过激励文件, 点击工作条上的“Reset”按钮或使用菜单命令: “**Debug > Run > Reset**”将 MCLR 驱动到低电平(然后拉到高电平)。

当看门狗(WDT)被允许(**Options > Development Mode**)并设置了合适的预分频(初始化相应得 OPTION 寄存器)而且 WDT 确实溢出了就可以模拟 WDT 溢出复位。WDT 的基本溢出时间为 12 毫秒(最接近指令周期的倍数), 但可以通过会话窗口来改变这个数值。

寄存器“ALUSTA”里的 Time-out (TO) 和 Power-down (PD)反应了相应的复位情形。这个特性有利于模拟用户代码里的各种上电(power-up)和时间到(time out)的情形。

#### 6-12-4-2 睡眠(SLEEP)

MPLAB-SIM 模拟器可以模拟睡眠(SLEEP)指令, 直到将处理器从睡眠模式唤醒的事件发生, 将一直保持睡眠“asleep”状态。例如: 假如看门狗定时器被允许, 到它溢出的时候将会将处理器从睡眠状态唤醒。另外一个从睡眠状态唤醒处理器的例子是, 当端口 B 上有输入状态的变化时。假如中断被允许, 而且 GLINTD 被置位, 处理器会被唤醒执行 SLEEP 指令后面的程序代码。假如 GLINTD = 0, 将会产生正常的中断响应。

#### 6-12-4-3 看门狗定时器

MPLAB SIM 模拟器可以完全模拟看门狗定时器。因为在芯片上这是用熔丝编程的, 必须在开发模式会话窗口里的配置选择盒里选择“允许”, 操作过程是: 在 MPLAB SIM 模拟器状态下选择命令“**Options > Development**

**Mode”**。WDT 的基本溢出时间为 12 毫秒(预分频=1), (最接近指令周期倍数)。

#### 6-12-5 特殊寄存器

为了帮助调试芯片, 通常芯片里无法观察到的一些寄存器被定义为特殊功能寄存器。预分频器和后分频器是不能在代码里定义为寄存器的, 所以这些标号会出现在特殊功能寄存器(SFR)窗口里面。

以下是基于 16 位核的处理器里特殊功能寄存器的特殊符号。(查阅相应的数据手册确定哪些符号是可以使用的)

- **T0PRE (Timer0 的预分频器)**
- **WDTPRE (WDT 的预分频器)**

#### 6-12-6 外设

##### 6-12-6-1 支持的外设

除了提供核心支持, 同时也支持以下的一些外设模块(IN addition to general-purpose I/O)。 (查阅相应的数据手册确定哪些符号是可以使用的)

- **Timer0 支持内部和外部时钟**
- **Timer1 和 Timer2 (以及他们的周期寄存器)**
- **Timer3**
- **两个 Capture 模块**
- **两个 PWM 模块**

##### 6-12-6-2 定时器 0(TIMER0)

MPLAB-SIM 模拟器可以完全支持 Timer0(可以产生溢出中断), 它可以在内部或外部脉冲的推动下作加一操作。还可以模拟外部时钟沿和定时器加一操作之间的延时以及中断响应时间延迟。鉴于模拟器的要求, 输入脉冲的高电平宽度必须不低于一个 TCY, 低电平也不能低于一个 TCY。可以通过寄存器 T0PRE 操作 Timer0 的预分频参数, 该寄存器可以被观察和修改。

##### 6-12-6-3 TIMER1 和 TIMER2

MPLAB-SIM 模拟器可以完全支持 Timer1 和 Timer2 及其各种工作模式。还可以模拟外部时钟沿和定时器加一操作(假如设置为在外部脉冲的上升或下降沿时作加一操作)之间的延时以及中断响应时间延迟。鉴于模拟器激励文件的要求, 输入脉冲的高电平宽度必须不低于一个 TCY, 低电平也不能低于一个 TCY。

##### 6-12-6-4 TIMER3 和捕捉器(CAPTURE)

MPLAB-SIM 模拟器可以完全支持 Timer3 和捕捉模块及其各种工作模式。还可以模拟外部时钟沿和定时器加一操作(假如设置为在外部脉冲的上升或下降沿时作加一操作)之间的延时以及中断响应时间延迟。鉴于模拟器激励文件的要求, 输入脉冲的高电平宽度必须不低于一个 TCY, 低电平也不能低于一个 TCY。

##### 6-12-6-5 脉宽调制 PWM

这一版本的 MPLAB-SIM 模拟器可以完全模拟两个 PWM 通道的输出(只适应于分辨率大于一个 TCY)。

#### 6-12-7 存储器模式

MPLAB-SIM 模拟器完全支持以下的存储器模式：

- 微控制器模式
- 扩展微控制器模式
- 微处理器模式

默认的方式是微控制器模式。假如用户想使用任何其他的模式，则必须使用开发模式会话窗口里的配置盒(Configuration tab)，操作方法是在 MPLAB-SIM 模拟器里使用命令：“***Options > Development Mode***”。

### 6-13 扩展型 16 位核处理器的模拟

本节介绍了基于扩展型 16 位核处理器(PIC18CXXX)的 IO 管脚，中断系统，寄存器，外设，工作模式，条件等。

#### 6-13-1 16 位核的处理器

在本文写作的时候，已经有以下系列的基于扩展型 16 位核的处理器可供使用：

- PIC18C242
- PIC18C252
- PIC18C442
- PIC18C452

以上列表包含了一些衍生系列型号，比如基于 ROM 版本的芯片(PIC18CRXX)和更新版本的芯片(PIC18CXXA)。需要指出的是，这一系列的 PICmicro 处理器的程序存储器是基于字节寻址的，而不象其他系列的基于字寻址的处理器。

#### 6-13-2 IO 管脚

扩展型 16 位核处理器的 IO 管脚有和其他外设复用的(因此会有不同的名称定义)。不管是以手动方式还是通过激励文件的方式修改管脚，只能使用以下的管脚名称。只有这些是 MPLAB SIM 模拟器可以识别的合法 IO 管脚(不同的芯片的管脚可能不同，只能依照数据手册里的定义为标准)。

- MCLR
- RA0-RA5
- RB0-RB7
- RC0-RC7
- RD0-RD7
- RE0-RE2

这些管脚名称可以在修改窗口( ***Window > Modify***)和激励文件里使用。

#### 6-13-3 中断

MPLAB-SIM 模拟器支持所有 16 位核处理器的中断系统：

- INT 管脚上的外部中断

- TMR0 溢出中断
- RA0 管脚上的外部中断
- 端口Port B 变化中断
- Timer/Counter1 中断
- Timer/Counter2 中断
- Timer/Counter3 中断
- Capture1 中断
- Capture2 中断

#### 6-13-4 CPU 模型

##### 6-13-4-1 复位情形

MPLAB SIM 模拟器可以模拟所有的复位情形。

处于正常操作状态下或处于 SLEEP 状态下的复位 MCLR 操作都可以简单地模拟：通过激励文件，点击工作条上的“Reset”按钮或使用菜单命令：“***Debug > Run > Reset***”将 MCLR 驱动到低电平(然后拉到高电平)。

当看门狗(WDT)在硬件上被允许(***Options > Development Mode***, 设置盒)或者即使硬件上没有允许但软件上在 WDTCON 寄存器里被允许了，就可以模拟看门狗溢出复位。在正常工作状态下，当 WDT 溢出的时候芯片要么复位并继续执行程序，要么断点停止运行(***Options > Development Mode***, 设置盒)。看门狗溢出时间与预分频器的设置有关。当没有设置预分频器的时候，WDT 的溢出时间为 18 毫秒(最接近指令周期倍数)。

##### 6-13-4-2 睡眠(SLEEP)

MPLAB-SIM 模拟器可以模拟睡眠(SLEEP)指令，直到将处理器从睡眠模式唤醒的事件发生，将一直保持睡眠“asleep”状态。例如：假如看门狗定时器被允许，到它溢出的时候将会将处理器从睡眠状态唤醒。另外一个从睡眠状态唤醒处理器的例子是，当端口 B 上有输入状态的变化时。假如中断被允许，而且 GLINTD 被置位，处理器会被唤醒执行 SLEEP 指令后面的程序代码。假如 GLINTD = 0，将会产生正常的中断响应。

##### 6-13-4-3 看门狗定时器

MPLAB SIM 模拟器可以完全模拟看门狗定时器。因为在芯片上这是用熔丝编程的，必须在开发模式会话窗口里的配置选择盒里选择“允许”，操作过程是：在 MPLAB SIM 模拟器状态下选择命令“***Options > Development Mode***”。WDT 的基本溢出时间为 12 毫秒(预分频=1)，(最接近指令周期倍数)。

#### 6-13-5 特殊功能寄存器

为了帮助调试芯片，通常芯片里无法观察到的一些寄存器被定义为特殊功能寄存器。预分频器和后分频器是不能在代码里定义为寄存器的，所以这些标号会出现在特殊功能寄存器(SFR)窗口里面。

- T0PRE (Timer0 的预分频器)
- WDTPRE (WDT 的预分频器)

## 6-13-6 外设

### 6-13-6-1 支持的外设

除了提供核心支持，同时也支持以下的一些外设模块(IN addition to general-purpose I/O)。(查阅相应的数据手册确定哪些符号是可以使用的)

- Timer0 支持内部和外部时钟
- Timer1 和 Timer2 (以及他们的周期寄存器)
- Timer3
- 两个 Capture 模块
- 两个 PWM 模块

所有外设的延迟都可以模拟，但是中断延迟却无法模拟。

### 6-13-6-2 定时器 0(TIMER0)

MPLAB-SIM 模拟器可以完全支持 Timer0(可以产生溢出中断)，它可以在内部或外部脉冲的推动下作加一操作。还可以模拟外部时钟沿和定时器加一操作之间的延时以及中断响应时间延迟。鉴于模拟器的要求，输入脉冲的高电平宽度必须不低于一个 TCY，低电平也不能低于一个 TCY。可以通过寄存器 T0PRE 操作 Timer0 的预分频参数，该寄存器可以被观察和修改。

### 6-13-6-3 定时器 1 和定时器 2

MPLAB-SIM 模拟器可以完全支持 Timer1 和 Timer2 及其各种工作模式。还可以模拟外部时钟沿和定时器加一操作(假如设置为在外部脉冲的上升或下降沿时作加一操作)之间的延时以及中断响应时间延迟。鉴于模拟器激励文件的要求，输入脉冲的高电平宽度必须不低于一个 TCY，低电平也不能低于一个 TCY。

### 6-13-6-4 定时器 3(TIMER3)和捕捉器(CAPTURE)

MPLAB-SIM 模拟器可以完全支持 Timer3 和捕捉模块及其各种工作模式。还可以模拟外部时钟沿和定时器加一操作(假如设置为在外部脉冲的上升或下降沿时作加一操作)之间的延时以及中断响应时间延迟。鉴于模拟器激励文件的要求，输入脉冲的高电平宽度必须不低于一个 TCY，低电平也不能低于一个 TCY。

### 6-13-6-5 脉宽调制(PWM)

这一版本的 MPLAB-SIM 模拟器可以完全模拟两个 PWM 通道的输出(只适应于分辨率大于一个 TCY)。

## 第7章 MPLAB 编辑器工具栏和菜单的使用

### 7.1 介绍

本章详细介绍了 MPLAB 编辑器桌面的工具栏和菜单选择项。在本章里将按照所有下拉菜单的安排来编制。

### 7.2 重点

本章将介绍以下几点：

- MPLAB 编辑器的桌面 (MPLAB IDE Desktop)
- 文件菜单 (File Menu)
- 项目菜单 (Project Menu)
- 编辑菜单 (Edit Menu)
- 调试菜单 (Debug Menu)
- 烧写器菜单 (Programmer Menu)
- 设置项菜单 (Options Menu)
- 工具栏菜单 (Tools Menu)
- 窗口菜单 (Window Menu)
- 帮助菜单 (Help Menu)

### 7.3 MPLAB IDE 桌面

MPLAB IDE 是一个可以调整大小的窗口，它能独立执行所有菜单栏中的条目。

调整 MPLAB IDE 在桌面上的大小，在桌面的右上角点击最大化。再一次点击最大化，使 MPLAB IDE 窗口最大化，然后启动 MPLAB IDE，MPLAB IDE 窗口将会自动打开最后设置的大小。使用完时，你可以点击最小化，使之变为图标。MPLAB IDE 将依然驻留在计算机的存储器中，你仍就可以自由使用 MPLAB IDE。

图 7.1 显示最大化桌面

通过使用在左上方的“**Microsoft Windows**”按钮，MPLAB IDE 对话框将和一般的窗口程序一样，你可以像使用标准窗口的程序一样使用 MPLAB IDE（除 MPLAB IDE 的特殊功能外）。其他窗口包括窗口的大小，图标，垂直和水平滚动条，升降条。

MPLAB IDE 所有的功能都可以通过点击桌面上的菜单来实现，可以通过下拉菜单来执行仿真功能。下拉菜单下的下划线是键盘快捷键。如何使用快捷键，按下<Alt>键和键盘快捷键。例如，按下<Alt>键，同时按下<F>，显示文件菜单。在文件菜单展开的状态下，仍就按住<Alt>，再打<O>，显示出“打开文件”对话框。

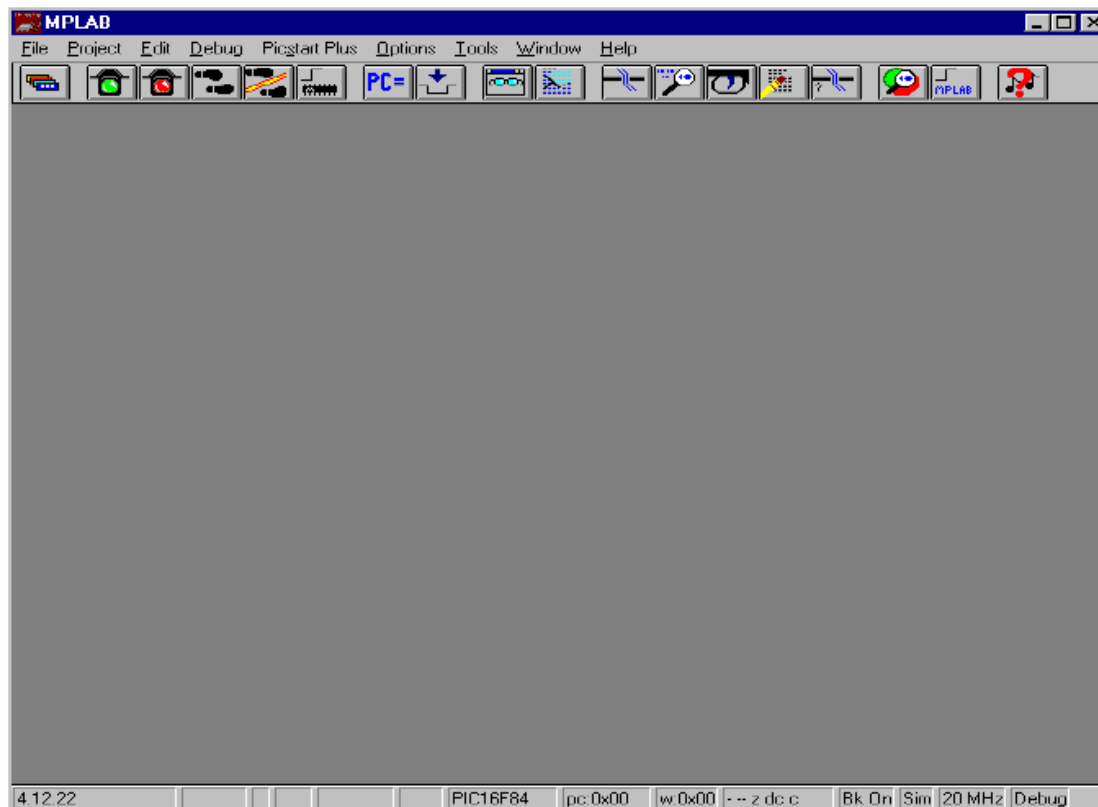


图 7.1: MPLAB IDE 桌面

### 7.3.1 工具栏

为了方便，MPLAB IDE 提供了四个快捷键方式的工具栏，它们是：

- 编辑工具栏(Edit Toolbar)
- 调试工具栏(Debug Toolbar)
- 项目工具栏(Project Toolbar)
- 用户定义工具栏 (User Defined Toolbar)

点击最左边的工具栏，显示你所需要的工具。参阅附录 C：“MPLAB IDE 工具栏和状态栏定义”获取关于所有四个工具栏的描述。

每一个工具栏上的按钮都可以根据你的特殊需要重新设置。详情请参阅 7.8.5.1.2 一节的内容获取更多信息。



图 7.2: MPLAB IDE 的测试工具栏

### 7.3.2 状态栏

下表描述了状态栏所提供的信息（图 7-3）。有关状态栏上符号的定义和描述，请参阅附录 C：MPLAB IDE 工具栏和状态栏。

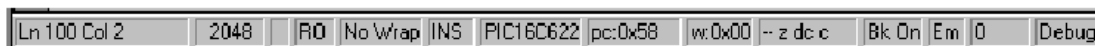


图 7.3: 状态栏

### 7.3.3 系统菜单

MPLAB IDE 为用户提供了许多可以看见许多不同的可视信息窗口。

**注解:** 使用系统窗口控制来改变如何显示窗口方式。

## 7.4 文件菜单

操作下列文件菜单:

- 新建 (New)
- 打开 (Open...)
- 察看 (View...)
- 保存 (Save)
- 保存为 (Save as...)
- 全部保存 (Save All)
- 关闭 (Close)
- 全部关闭 (Close All)
- 导入 (Import)
- 导出打印 (Export Print)
- 打印 (Print)
- 打印设置 (Print Setup)
- 退出 (Exit)
- 最近使用过的文件 (Most-Recently-Used File List)

### 7.4.1 打开新建文件

点击 **File>New**, 打开一个新的, 空的窗口。此窗口没有最初的文件名。保存这个新文件, 选择 **File>Save As**。

MPLAB IDE 建立这个窗口的时候, 采用了一套定义“新文件”的模式。

选择菜单命令: “**Options>Environment Setup**”以后, 在 Editor Modes (编辑模式) 标签下可以设置“新文件”涉及的 TAB 长度以及其他设置。

### 7.4.2 打开已有文件

编辑一个或多个已现有的文件, 选择菜单命令: “**File>Open**”。它从你选择的文件中打开一个标准的对话框来编辑。如果选择的文件已被打开, MPLAB 编辑器将激活包含有已被打开的文件的窗口。



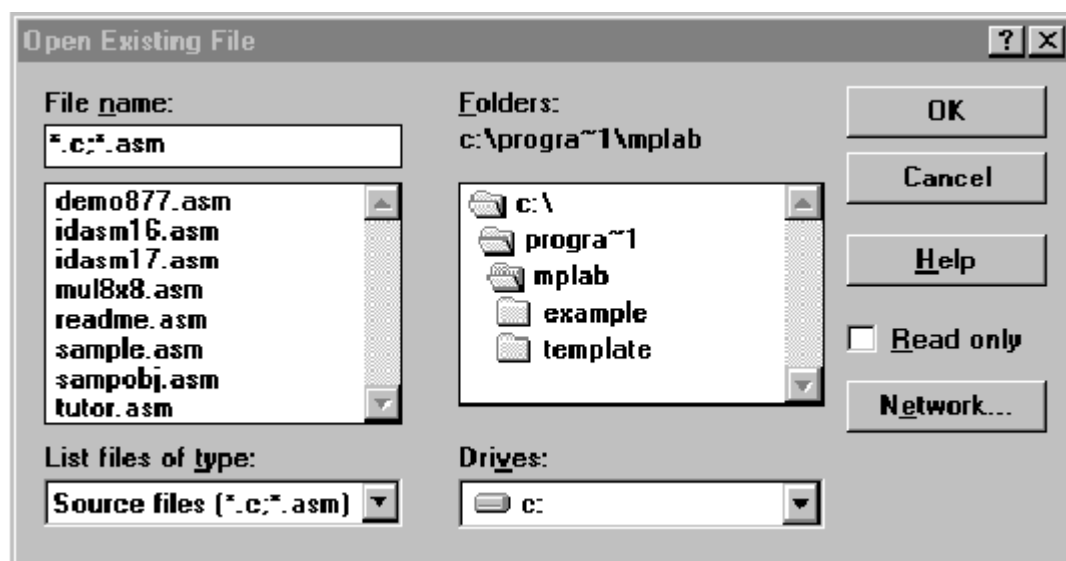


图 7.4: 打开现有文件

1. 用驱动器和目录列表来选择磁盘驱动器和目录。
2. 在“File Name”列表中选择你所需要的文件。
  - 从列表中增加一个文件的方法是：按住〈Ctrl〉键，点击你想要的文件。
  - 从列表中增加一系列文件的方法：在希望选择的范围内点击第一个文件然后要按住〈Shift〉键，再点击最后一个文件。或是选中第一个文件，然后拖至最后一个文件就可以了。

或者

在“File Name”（文件名）栏里面，输入文件名。

3. 如果你想以只读文件模式打开一个文件，点击“Read Only”（只读）。这样所有的文件将以这种模式打开。
4. 点击“OK”，打开文件

在左下方的“List Files of Type”栏中，可以查找你所需要的文件。列如，在此栏中输入“\*.MAS”，列栏中将显示所有扩展名为“.MAS”的文件。

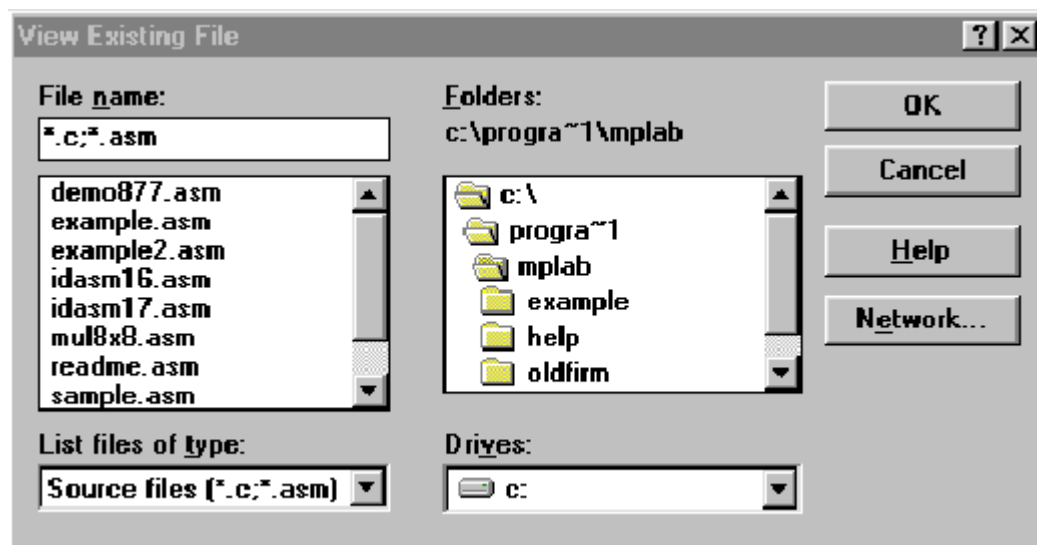
通过点击“OK”或“Cancel”按钮来关闭对话框，MPLAB 编辑器就会按照会话栏里给定的名称创建当前的工作目录。

如果被选中的文件正在被编辑，MPLAB 编辑器就会激活显示该文件的窗口。

### 7.4.3 察看文件

在只读模式下，打开一个或多个已有的文件。你可以查看目录，但不能修改。如果你已使用菜单命令：“**File > Open**”命令，并且已选中“Read Only”选择盒，才会确切的有操作：“**File > View**”。

（见下图）



#### 7.4.4 保存文件

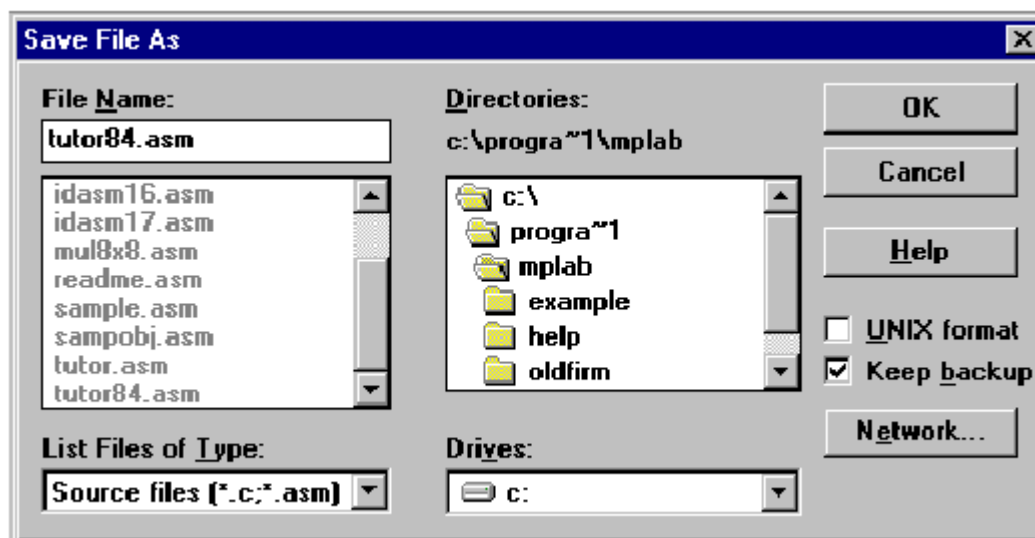
下面有三种方式可以用来保存文件：

选中菜单命令 **File>Save** 把当前窗口的内容保存在磁盘上。现存窗口的内容将替换原来的内容。如果现存的文件没被命名，MPLAB 编辑器会提示你，给此文件命名。如果文件名被重复使用，MPLAB 编辑器则会保存当前文件来替换原文件。

选择菜单命令 **File>Save As** 保存文件，你可以指定文件名。选择命令 “**File>Save As**” 可以把内容保存到磁盘文件里，允许用户指定文件名。MPLAB 编辑器会确认，改写原有的文件。

1. 通过驱动器和目录列表，选择你所要保存文件的驱动器和目录。
2. 也可以在文件名框中输入文件名，或从列表中选择要改写的已有文件。
3. 点击 “OK”，保存文件。如果你指定的文件名已存在，MPLAB 编辑器将会改写。
4. 通过察看左下脚的 “List Files of Type”（文件列表类型）可以查找你所需要类型的文件名。

（见下图）



选择菜单命令 **File>Save All**，保存所有已更改文件，保存所有剪切板里改变过的内容到剪切文件里，一次操作就可以保存所有改变过的剪切板文件。

注解：在任何对话框中点击 Cancel 按钮，将取消所有的保存命令。

#### 7.4.5 关闭文件

选择菜单命令 **File>Close**，关闭当前显示在窗口的文件。选择菜单命令：**File>Close All**，关闭所有打开的工作文件。

如果你修改的文件没有被保存到磁盘中，MPLAB 编辑器将提示你“保存，放弃或取消”被修改的文件。

#### 7.4.6 导入

使用菜单命令：**File>Import**，你可以从一个 PC 文件中移动资料至仿真器和目标存储器中或是移至仿真存储器。也可以把信息从目标传递到模拟器的存储器中，这个功能允许你将数据从目标板传送数据到仿真器进行调试。

##### 7.4.6.1 导入至存储器 (Import to Memory)

选择菜单命令 **File>Import>Import to Memory**，显示对话框，选中要导入（下载）的文件。选中的文件将被输入到仿真存储器或模拟器的存储器里面。文件必须是合法的“hex”十六进制文件。如果已准备了用来烧写的 PICmicro MCU 芯片的“hex”十六进制文件文件(比如：code.hex)，则可以通过这项操作把 hex 代码调到 MPLAB IDE 的程序存储器窗口中。这个文件也可以通过建立的一个 MPLAB IDE “项目”或“导出”程序存储器来建立。

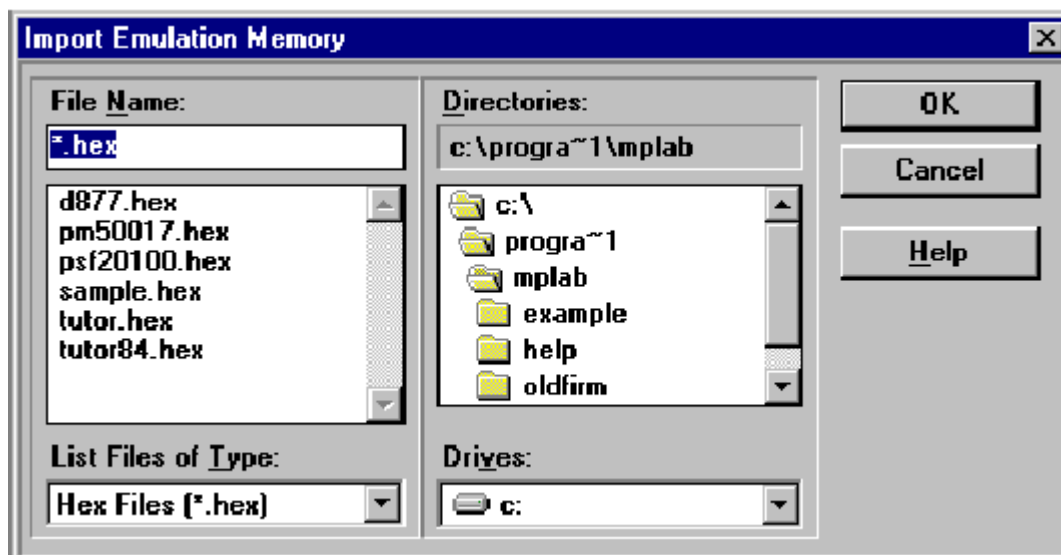


图 7.5：输入到仿真存储器 “Import Emulation Memory” 对话框

##### 7.4.6.1.1 PIC17CXXX 和 PIC18CXXX 系列芯片

如果现有的目标处理器 (target processor) 是 PIC17CXXX 或是 PIC18CXXX 系列芯片，文件扩展名应是 intel 文件格式 (INHX32) 的“hex”十六进制文件。如果目标文件被成功导入，符号 (symbols) 将从 \*.COD 文件中被自动输入。目标代码文件的默认扩展名为 .HEX。

如果没有 a\*.COD 文件，你则需要关闭 “source tracking”（源跟踪）开关选择项。选择菜单命令 **Options>Environment Setup**，点击 “General” 标签，在对话框中的 “Global Switches”（全局开关）设置源跟踪。

**注意：**如果应用程序的存储器寻址超过 64KB（PIC17CXXX 为 32K 字）或者包含配置位信息，可以使用 INHX32 格式。

#### 7.4.6.1.2 其他 PICmicro® 芯片

如果现有的目标处理器属于 PICmicro MCU 系列，而不是 PIC17CXXX 或 PIC18CXXX 系列芯片，文件格式必需是 intel 8-BIT “hex” 十六进制文件格式（INHX8M）。如果目标文件已成功导入，这时候相应的符号文件 “.COD” 也自动的被输入。目标代码默认的扩展名为 “.HEX”。

#### 7.4.6.2 导入到目标内存 (Import to Target Memory)

对于工作在微处理器模式或是扩展微控制器模式下的 PIC17CXXX 或 PIC18CXXX 系列芯片，使用仿真器进行调试的时候，你都可以选择从目标板上选择片外（off-chip）存储器。选择菜单命令 **Options>Development Mode**，并点击 **Configuration**，进入处理器模式（Processor Mode），选择菜单命令 **Options>Development Mode**，点击 “Memory” 标签，即可进入片外存储器。

为了将一个数据文件输入（下载）到目标内存，可以选择菜单命令 **File>Import>Import to Target Memory**，弹出对话框，选择一个通过在仿真器导入到目标内存中的文件。文件必须是合法的 intel 格式的十六进制 hex 文件。文件可以通过创建一个 MPLAB IDE “项目” 或者从程序存储器 “导出”（export）来建立。

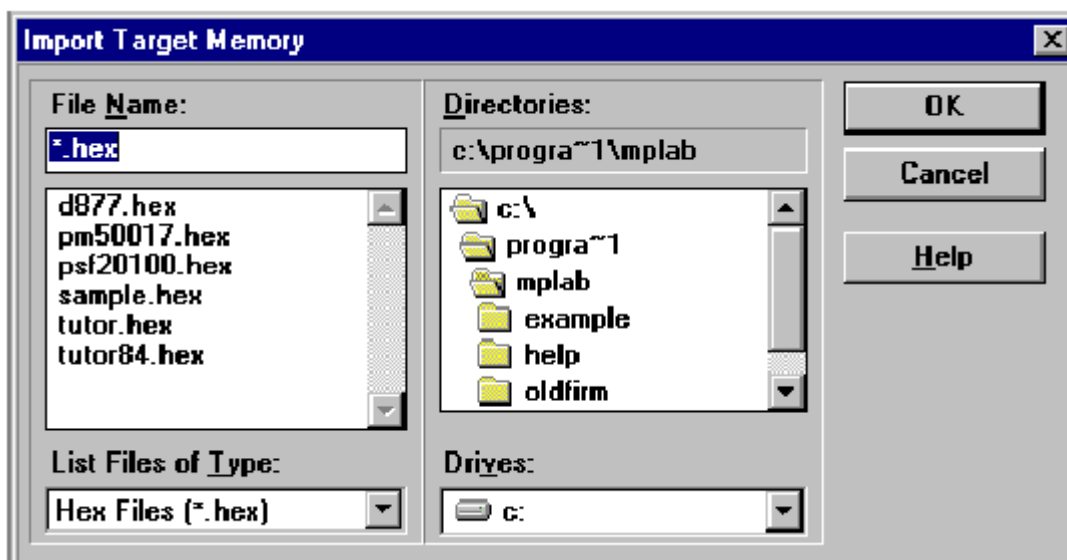


图 7.6: 导入目标存储器对话框 (MPLAB-ICE)

#### 7.4.6.3 从目标内存中复制

对于工作在微处理器模式或是扩展微控制器模式下的 PIC17CXXX 或 PIC18CXXX 系列芯片，使用仿真器进行调试的时候，你都可以选择从目标板上选择片外（off-chip）存储器。选择菜单命令 **Options>Development Mode**，并点击 **Configuration**，进入处理器模式

(Processor Mode)，选择菜单命令 **Options>Development Mode**，点击“Memory”标签，即可进入片外存储器。

为了把目标存储器里的内容复制到一个数据文件中，可以选择菜单命令 **File>Import>Copy form Target Memory**，将数据通过仿真器拷贝到计算机里的一个文件。

## 7.4.7 导出 (Export)

### 7.4.7.1 导出跟踪缓冲器

选择菜单命令 **File>Export>Export Trace Buffer**，显示对话框，你可以把模拟器或仿真器的跟踪缓冲器保存到已选文件中。

**注意：**对于 MPLAB-ICE 仿真器，如果你有一个具有最大标号深度的跟踪缓冲器来保存所有的跟踪点，可能需要 7MB 的容量（通常仅需 2MB）对于 MPLAB-SIM 模拟器来说，完整保存出现在跟踪窗口里的内容（地址，数据，反汇编代码和外部逻辑探头数据），可能需要超过 1MB 的存储空间。

#### 7.4.7.1.1 MPLAB-ICE 跟踪缓冲器

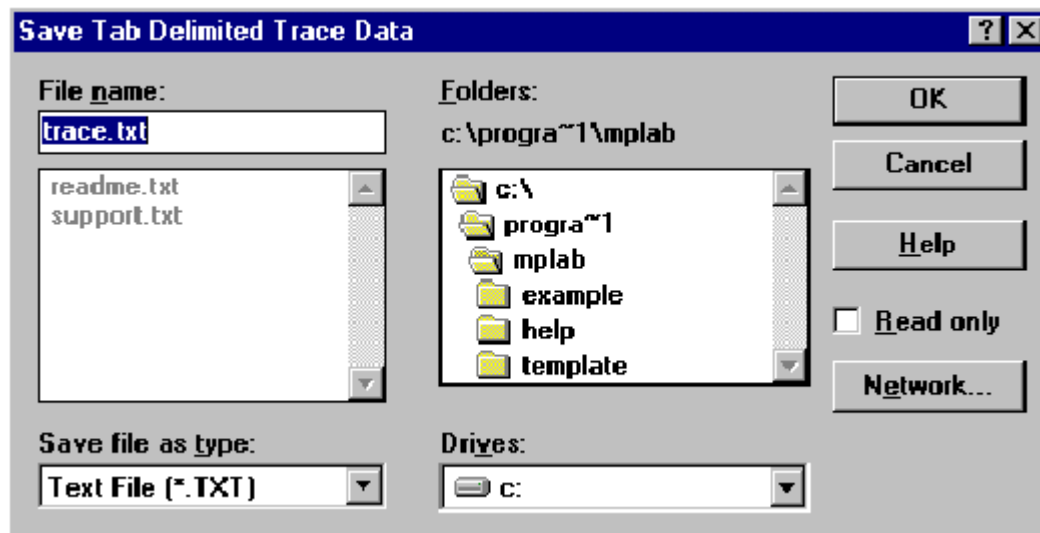


图 7.7: 导出跟踪缓冲器的对话框 (MPLAB-ICE)

- **File name:** 把文件名输入对话框里输入文件名或者在文件名列表中选择一个你想要改写的现有文件。
- **Save file as type:** 指定要保存的跟踪缓冲器文件的类型。显示的文件列表将限制你所指定的文件类型。
- **Folders:** 选择你想要存入的目录。
- **Drives:** 选择你想要存入的磁盘。
- **OK:** 点击 OK，保存文件。如你所指定的文件名已存在，MPLAB 编辑器将会确认，改写。
- **Cancel:** 注意：点击 Cancel，撤消整个保存操作。
- **Read Only:** 如果你希望防止日后加以改动，选择“Read Only”（只读）。

## 7.4.7.1.2 MPLAB-SIM 和 PICMASTER 跟踪缓冲器

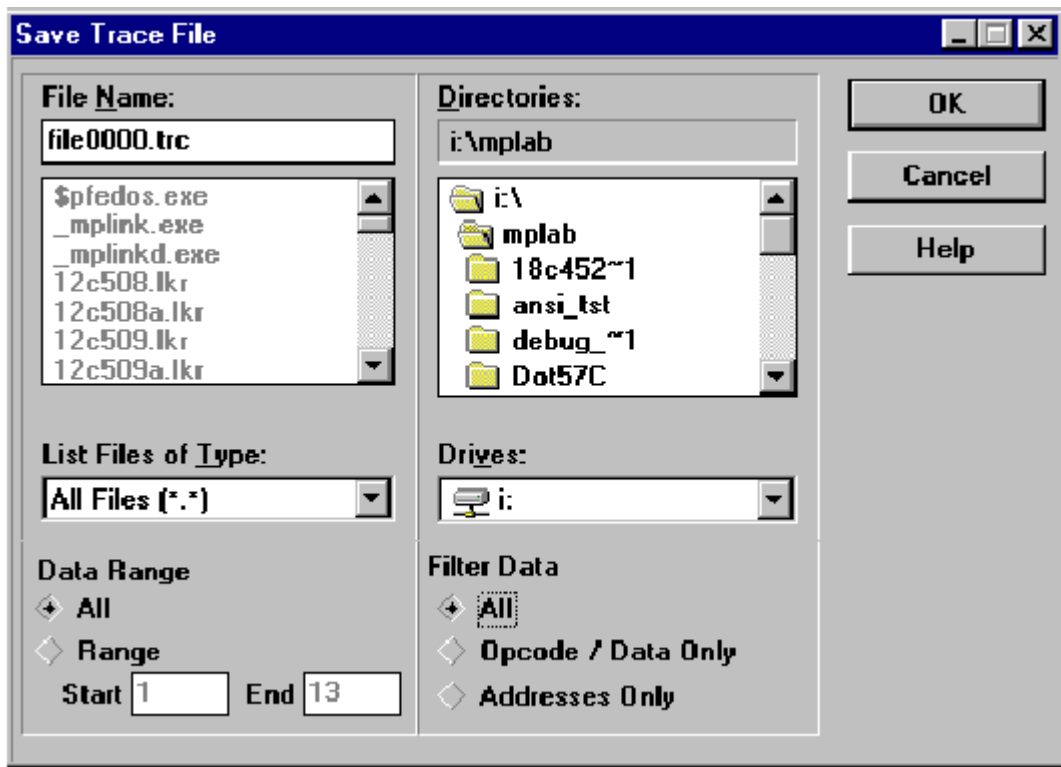


图 7.8: 导出跟踪缓冲器对话框 (MPLAB-SIM 或 PICMASTER)

- **Range:** 选择你想要保存的跟踪起的范围 (从 0 到 8190)。在 “Start” (开始) 和 “End” (结束) 框中输入所需的数值。
- **Filter Data: All :** 把完整的跟踪缓冲器内容写入到已选中的文件。
- **Filter Data: Opcode (Data Only)**  
只保存操作码或数据 (Opcode/Data)。(PIC17CXXX 外部读/写周期)。只能用于 MPLAB-ICE 仿真器。
- **Filter Data: Address Only :** 仅保存地址。

## 7.4.7.2 导出存储器 (Export Memory)

选择菜单命令 “**File>Export>Export Memory**”，显示 “Export Memory” (导出存储器) 对话框并将存储器内容导出到一个文件里文件。

如果你有一个已经编程的 PICmicro MCU 芯片，你可能想要复制到其他的芯片，你可以将芯片里的程序读入 MPLAB IDE (从编程菜单中使用 **Read Device** 命令)，保存此芯片的程序，然后使用命令: **File>Export>Export Memory**，把读出的结果输出到一个十六进制文件里。然后可以在芯片烧写菜单里使用 “Program/Verify” (烧写/校验) 选择项把文件导入到 MPLAB IDE 中并烧写/校验其他的芯片。

**注意:** 在你选择作为导出的存储器范围内，所有的位置(包括空位置)，在你以后再次调入数据时，将改写程序存储器。相比之下，在 “**Build Project**” (创建项目) 并建立一个十六进制文件的过程中，忽略了一些空位置。

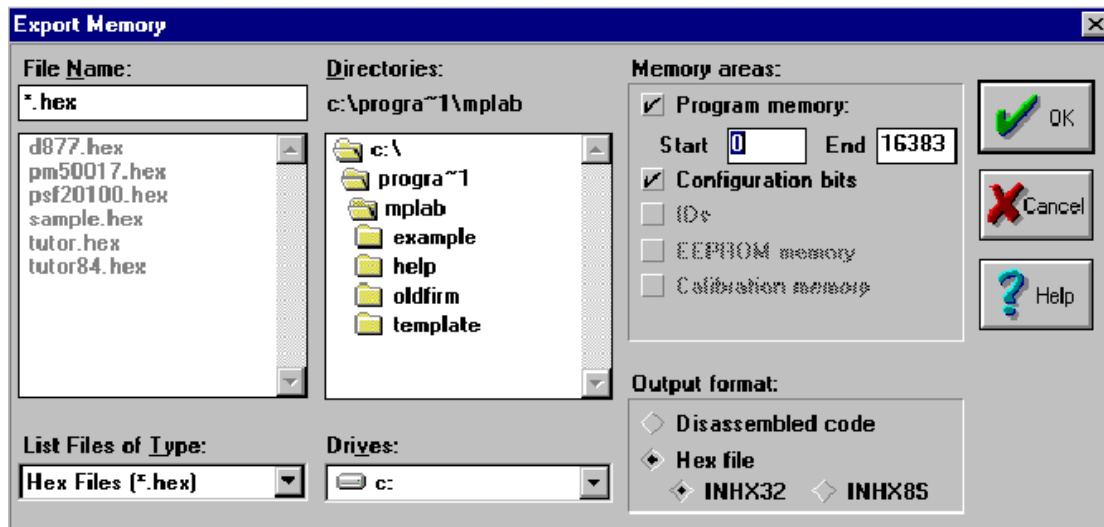


图 7.9: 导出存储器对话框

- **Memory areas:** 在建立输出文件中，确定你想要存入的存储器区域。
  - Program Memory:** 选择，保存程序存储器，同样也确定了程序存储器的“Start”（开始）和“End”（结束）的地址（整个范围是默认的）。
  - Configuration Bits:** 选择，保存配置位设定。
  - IDs:** 如果能使用，则选择保存 ID 信息。
  - EEPROM Memory:** 如果能使用，选择其则保存 EEPROM 存储器。
  - Calibration Memory:** 如果能使用，选择其则保存校正存储器。
- Output format:** 确定输出文件的格式化。
  - Disassembled Code:** 选择其，可以用反汇编格式保存。
  - Hex Code:** 选择十六进制格式的 INHX32 代码或 INHX8X 格式代码。

#### 7.4.8 打印 (Ctrl+P)

在你当前所选的打印机下，选择命令 **File>Print** 打印部分或所有当前文件。

可以通过打印对话框，详细指定哪种打印机，如何打印文件。

默认条件下，MPLAB IDE 将使用你最后一次打印文件时所指定的打印机设置。打印机的默认参数设置，列如，自动换行，页眉，TAB 大小和行号等参数的设置，都来自当前窗口里设置的编辑模式。你可以在 **Options>Current Editor modes** 菜单命令，设置这些参数。

MPLAB IDE 在当前的使用状态下，对话框显示打印机的名字。如果你没有指定，你的系统将被设为默认打印机。

##### 7.4.8.1 打印当前文件

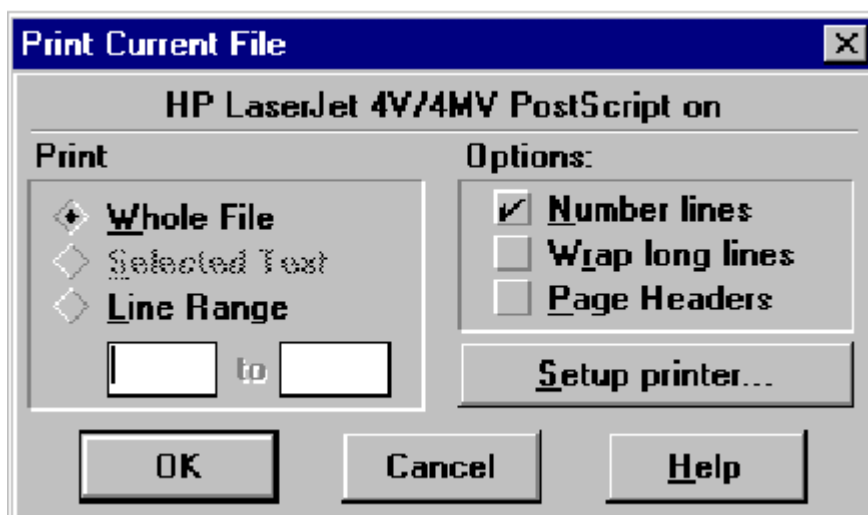


图 7.10: 打印当前文件对话框

- **Setup Printer:** 点击 Setup Printer，改变打印机，打印字体，页边距。
- **Whole File:** 点击 Whole File（默认），打印整个文件。
- **Selected Text:** 点击 Selected Text，只打印选中处。此操作仅适用于的文本内容。
- **Line Range:** 点击 Line Range，在框内填入要打印的首尾页码和末尾页码数。你可以使用“开始”和“结束”来表示文件的首页和末页。
- **Number Lines:** 选择 Number Lines，打印每一行时，都添加行号。
- **Wrap Long Lines:** 选择 Wrap Long Lines，较长的行会根据页面自动换行，而不是被切去。
- **Page Headers:** 选择 Page Header，在每一页的页眉打印文件名和其他信息。

#### 7.4.9 打印机设置

选择菜单命令 **File>Print Setup**，设置打印机的详细参数，运行一个单独的打印机设置的对话框，选择字体。MPLAB IDE 记录了你在“**Print Setup**”菜单选项里设置的打印机信息。这样，不同的打印机就有不同的设置。这些数值便成为默认的打印机设置。

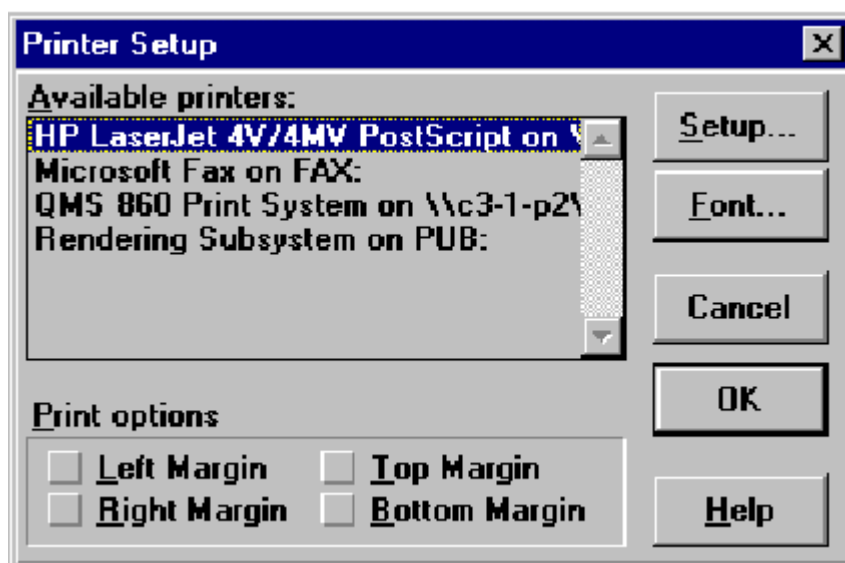


图 7.11: 打印机设置对话框



**File>Print Setup**具体讲述了：

- 你使用的是哪种打印机。
- 当你打印时，你运用的是哪种页边距。
- 打印机使用的是哪种字体。

如果用户菜单命令“**File>Print Setup**”，则可以运行打印机自己的设备对话框来设定具体设备的详细资料。

- **Available Printers:** 在“Available Printers”（当前支持的打印机）栏中是不同的打印机，滚动此列，在打印机名上点击左键，使之光亮。
- **Print Options:** 改变每页的边距。在“Print Options”（打印设置）区域，选取或不选取这一选择项。
- **Setup:** 运行被高亮选中的打印机的设备对话框。点击“Setup”（设置）。
- **Font:** 点击，改变打印机的字体，MPLAB IDE 将使用选中的打印机。

由于 MPLAB IDE 是一个文件编辑器，而不是一个字处理器，你被限制使用固定的字符间距（fixed pitch fonts），也就是说，所有的字符宽度都相同。

在选择字体后，如果你运行你的打印机设置程序对话框，你选择的字体将不再使用，一些打印机在不同操作模式下提供不同的字体。

#### 7.4.10 退出（Alt+F4）

选择菜单命令 **File>Exit**，终止 MPLAB IDE 任务。

如果你所使用的工作窗口中的多个文件被改变，而且没被存盘，MPLAB IDE 将会提示你保存每个文件，你可以选择保存已改写文件，放弃或取消全部操作。也可以被终止保存当前的项目。

#### 7.4.11 最近使用过的文件列表（Most---Recently---Used File List）

MPLAB 编辑器在 File Menu 的最后增加了 MRU（Most-Recently-Used）文件列表。无论你何时打开文件，MPLAB 编辑器都会在列表中记录文件名，无论何时你打开一个文件，MPLAB 编辑器将最近一次被使用过的文件名(MRU)记录在文件列表菜单的末尾，任何时候你打开一个文件，MPLAB 编辑器就在列表里记录文件名，调整列表使得最近使用过的文件排列在顶部。只需点击菜单里的选项，可以简单地重新打开列表中的任何文件。这个选项是使用者可配置的。参阅 7.8.5.3 节。

### 7.5 “项目”菜单

以下选项在项目菜单中被使用：

- 新项目（New Project）：创建一个新项目
- 打开项目（Open Project）：打开一个已有的 MPLAB IDE 项目
- 关闭项目（Close Project）：关闭当前打开的 MPLAB IDE 项目
- 保存项目（Save Project）：保存 MPLAB IDE 项目
- 编辑项目（Edit Project）：可以调节一些设置，添加文件等
- 创建项目（Make Project）：在项目中建立所有已改变的结点
- 全部创建（Build All）：在项目中建立所有的结点
- 创建结点（Build Node）：建立已选中的结点
- 安装语言工具（Install Language Tool）：可以设置 MPLAB IDE，使其可以认出一种语言工具
- （最近使用过的项目）(Most Recently Used Projects)：显示最近使用过的项目

关于项目菜单选项的详细资料，请参阅第三章：“开始使用 MPLAB IDE – 一个指导”。

## 7.6 编辑菜单

### 7.6.1 通用编辑选项

MPLAB 编辑器可以在插入模式 (insert) 或重叠模式 (strikeover) 下输入文本。MPLAB 编辑器在状态栏中显示 “INS” 或 “OVR” 模式。

#### 7.6.1.1 撤消——Ctrl+Z

选择菜单命令 **Edit>Undo**，撤消最后一次编辑。当没有编辑活动可撤消，则显示不能撤消，你不能选择此命令。

你可以设置撤消最近的编辑活动次数。请参阅 7.8.5.4 节。

#### 7.6.1.2 剪切——Ctrl +X

在当前窗口中删除选中处，寄予剪贴板中。在此操作后，可以将这个被删除的文本粘贴到其他 MPLAB 编辑器窗口中或其他窗口应用程序中。

#### 7.6.1.3 复制——Ctrl+C

在当前窗口中复制选中的文本到剪贴板中。在此操作后，可以将这个被删除的文本粘贴到其他 MPLAB 编辑器窗口中或其他窗口应用程序中。

#### 7.6.1.4 粘贴——Ctrl+V

粘贴剪贴板中的内容到当前窗口光标所在处。只有剪贴板包含的信息是文本格式的时候，你才可以执行此操作。MPLAB 编辑器不能支持位图或其他剪贴板格式化的粘贴。

#### 7.6.1.5 全选

使当前窗口的所有内容高亮。

#### 7.6.1.6 选中词 — 双击鼠标左键

双击鼠标左键，使光标所在的词高亮。

#### 7.6.1.7 删除行——Ctrl+Shift+K

删除光标所在的整行内容，移动光标至下一行的行首。

#### 7.6.1.8 删除 EOL——Ctrl+K

删除光标所在处至该行最后的文本。如果光标在空白行的行首，下一行将被移上来缩短间距。

#### 7.6.1.9 转向行 (Goto Line) - Ctrl+G

移动光标至具体一行的行首。

此菜单命令允许用户指定一个绝对或相对的行号。

#### 7.6.1.10 查找——F3

这个命令可以在当前的窗口中搜索用户指定的文章字符串。如果当前的窗口中有被高亮的文本，查找指令则使用这一文本作为搜索字符串。

如果要在当前窗口搜索文本，把光标放在搜索的起点，并且选择菜单命令 **Edit>Find**。



图 7.12: 查找对话框

在查找对话框中，你可以使用字符的 ESC 代码来搜索特殊的字符。你可以使用“\n”代表“回车”，“\t”代表 TAB 制表符，“\”代表反斜线。

#### 7.6.1.11 替换——F4

这个命令在当前的窗口中找一个文本字符串，并且选择性地与另外的字符串替换。这个命令让你指定搜索参数。如果在当前的窗口中有高亮的文本，替换操作会使用该文本作为搜索字符串。“

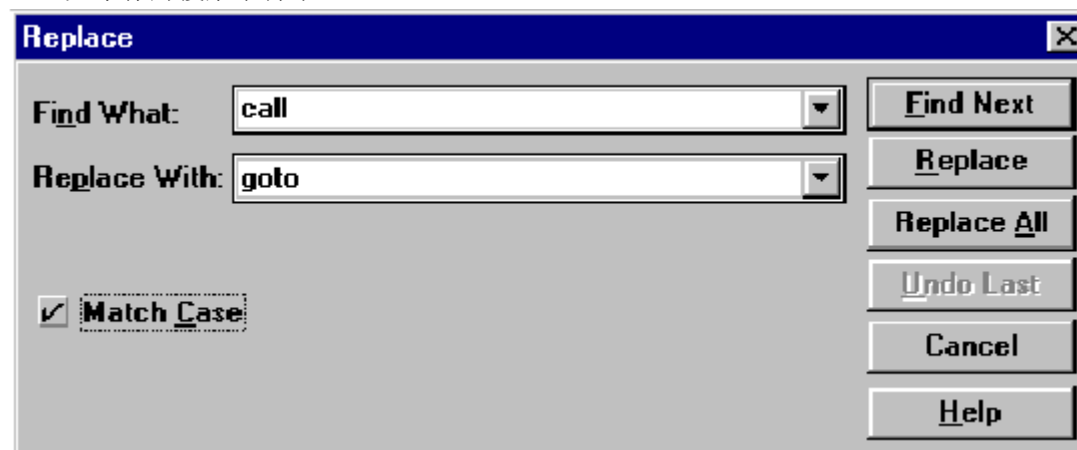


图 7.13: 替换对话框

在窗口中替换文本或特定的字符，把光标放在搜索的起点，选择菜单命令 **Edit>Replace**。在查找对话框中，你可以使用字符的 ESC 代码来搜索特殊的字符。你可以使用“\n”代表“回车”，“\t”代表 TAB 制表符，“\”代表反斜线。

#### 7.6.1.12 重复查找——Shift+F3

在不使用查找对话框的情况下，重复查找，选择菜单命令 **Edit>Repeat Find**。

#### 7.6.1.13 重复替换——Shift+F4

在没有详细提示的情况下，重复最后一次替换操作，可以选择菜单命令：***Edit>Repeat Replace***。

#### 7.6.1.14 括号匹配——Ctrl+B

MPLAB 编辑器允许用户使用括号字符，如方括号和圆括号，它通常用来限定文本或源程序的分段区域。

括号字符的定义有所不同，取决于在一个窗口的模式下设定的语言类型。对于 C 语言，括号是有相应的语法意义的符号。开括号有“{”，“[”，或“（”，和闭括号“}”，“]”，“）”。

语言类型选择为“none”的时候，括号被用做文本中所使用的一种符号或者为其他一些语言所定义——开括号有“{”，“[”，或“（”，和闭括号“}”，“]”，“）”。

如果想定位相匹配的括号，可以把光标移到一个括号上，然后使用菜单命令：***Edit>Match>Brace***。光标移至相匹配的括号处，并考虑在代码中的嵌套层。

### 7.6.2 模板选项

#### 7.6.2.1 添加模板文件

使用菜单命令 ***Edit>Template>Attach File***，添加一个已有的模板文件（.TPL），存放到存储器中，使其可以使用。选择你想要添加的文件，并点击 OK。

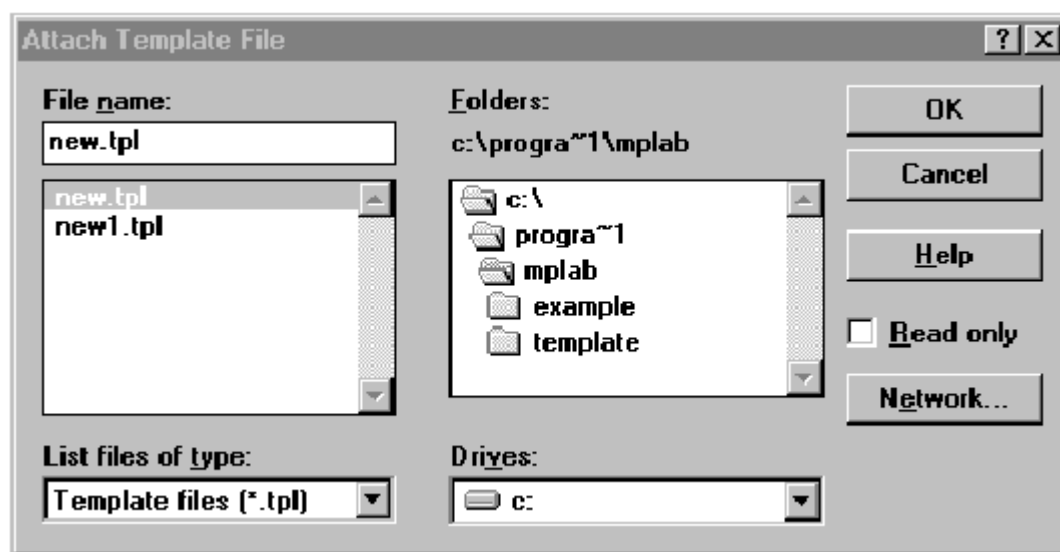


图 7.14: 添加模板文件对话框

#### 7.6.2.2 拆分模板文件

使用菜单命令 ***Edit>Template>Detach File***，从存储器中移走一个模板文件。模板文件一旦被分开，就不能使用此模板的内容。

选择菜单命令 ***Edit>Template>Detach File***，选择你想要拆分的模板文件，并点击 OK。但不能拆分一个你正在编辑的独立模板。

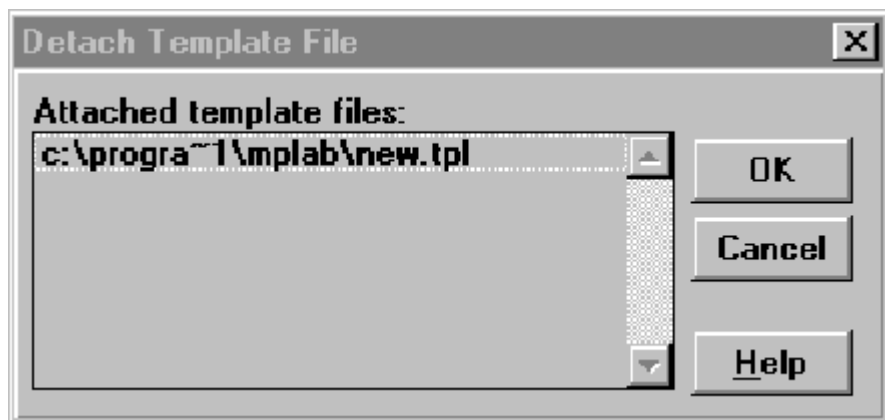


图 7.15: 拆分模板文件对话框

#### 7.6.2.3 创建模板文件

该模板是面向 MPLAB IDE 应用开发者所设计的，开发 MPLAB IDE 模板的第一步是创建一个包含模板的文件。

使用菜单命令 **Edit>Template>Creat Files**，可以使 MPLAB 编辑器创建一个包含单独模板的模板文件（.TPL）。在储存二进制文件格式的模板之前，(.TPL) 文件必须已经创建。填写文件名，驱动器和目录名，并点击“OK”。

如果你想使该模板文件自动地被所有 MPLAB IDE “项目”所接受，必须将文件名设为“auto.tpl”。当你在 MPLAB IDE 中编辑源文件时，在“auto.tpl”中的模板始终可被使用。

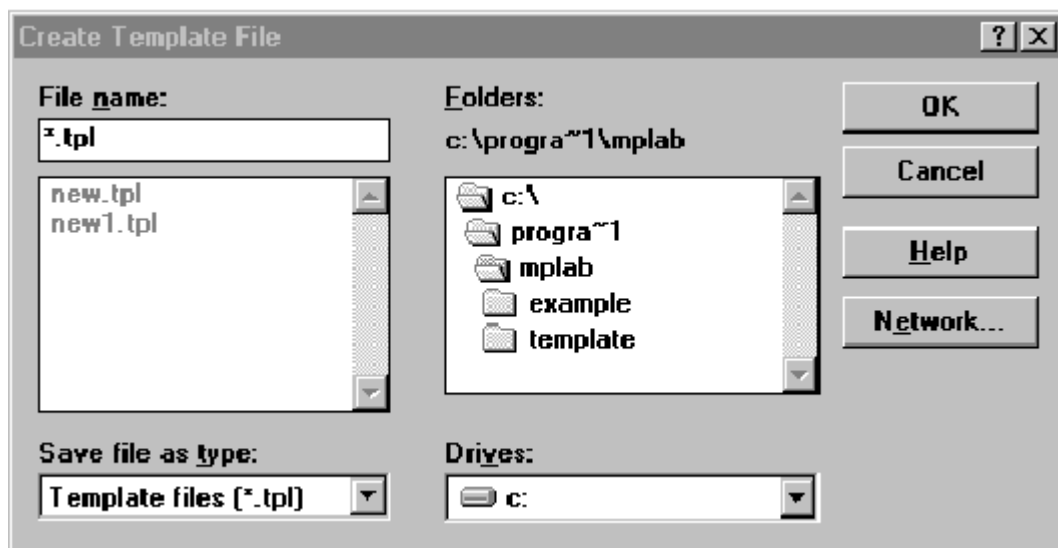


图 7.16: 创建模板文件对话框

#### 7.6.2.4 保存模板文件

当你在对模板进行编辑，创建或删除时，MPLAB 编辑器将对存储器中的模板副本进行修改，你退出 MPLAB IDE 时，也将被撤消修改。这些修改将不会被保存到磁盘中，除非使用使用菜单命令：**Edit>Template>Save File**。

如果你不想保存模板文件，而退出 MPLAB IDE 时，又希望保存对模板（.TPL）文

件的改变，回答 “Yes”。

#### 7.6.2.5 插入模板

使用菜单命令 **Edit>Template>Insert**，可以将现有的独立模板插入到源文件中，来用于程序的开发。在你插入模板之前，文件必须已被链接到 MPLAB IDE。把光标放置在源文件中你想要插入的模板文本上。选择菜单命令 **Edit>Template>Insert**，选择包括相应模板的模板文件，选择你想要插入的独立模板，点击 “OK”。

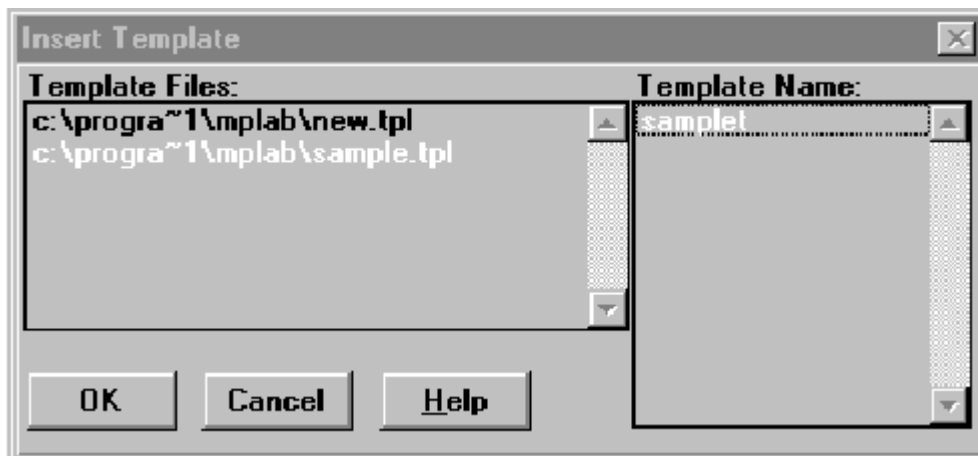


图 7.17: 插入模板对话框

#### 7.6.2.6 编辑模板

改变模板文件中的模板，使用 **Template>Edit** 命令，选择你想要更改的模板文件和独立模板，点击 OK。模板文件必须已经于 MPLAB IDE 链接（见 7.6.2.1），只有这样，你才能对包含于其中的一个文件进行编辑。

当你选择了一个模板，MPLAB 编辑器将此模板放置到编辑器窗口中。该窗口与普通的文本文件相同，但是其标题显示其实际上为模板。对模板进行编辑后，使用 **Edit>Template>Store** 命令更新该模板。

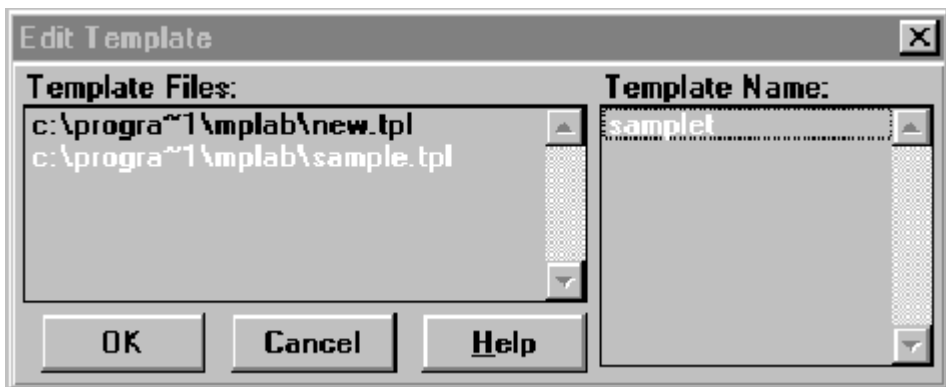


图 7.18: 编辑模板对话框

### 7.6.2.7 新模板

一旦建立了含有模板的.TPL 文件，就为个人模板建立了源和/或文本内容。你可能希望使用 MPLAB IDE 里面现成的模板而不是自己去建立。

使用菜单命令：**Edit>Template>New** 建立一个模板。MPLAB 编辑器打开一个编辑窗口，该窗口的标题显示它是一个模板。

建立自己的源，在未命名模板窗口中输入简单的文本。

如果想要使用 MPLAB IDE 模板文件中的某一个文件，则需要在路径“MPLAB\Templates\Code”或路径“MPLAB\Templates\Object”下打开 MPLAB IDE 模板文件。选择所要复制的文本（或使用命令 **Edit>Select All**），然后复制并将它粘贴到未命名模板窗口，分别使用 **Edit>Copy** 和 **Edit>Paste** 命令。

可以在模板中插入标记，表明任一用户源代码或开发工程师必须完成的文本（见 7.6.2.11 节）。当完成建立新模板后，切记使用菜单命令：**Edit>Template>Store** 或 **Edit>Template>Store Template As** 命令来保存（参见 7.6.2.8 节和 7.6.2.9 节）。

### 7.6.2.8 保存

选择菜单命令 **Edit>Template>Store**，将一个模板保存在模板文件中。该命令覆盖了模板文件中的原始模板。

注意：由于保存命令仅仅写入存储器内，因此退出 MPLAB IDE 时，改变的内容将被废除。为了保存模板的改变以备后用，可以使用菜单命令：**Edit>Template>Save File**。当你退出 MPLAB IDE 时，会有提示，会问你是否需要保存变化内容到模板（.TPL）文件中。回答是 **Yes**。

### 7.6.2.9 另存为

一旦完成了建立模板并插入标记，选择菜单命令 **Edit>Template>Store As** 命令将个人模板源代码/文本保存在模板文件中。也可以使用该命令将这些变化的信息以不同的文件名保存在一个新的模板文件中（列如，为原始模板的源/文本建立一个拷贝）。

选择菜单命令 **Edit>Template>Store As**，选择模板文件，指定文件名，并单击 OK。若选择一个已存在的模板名，该命令会覆盖原有的模板内容。在执行该命令前 MPLAB 编辑器会要求确认。

**注意：**由于“Store As”（另存为）命令仅仅写入存储器，所以当退出 MPLAB IDE 时所做的改变及添加内容会被破坏。为了保存模板的改变以备后用，可以使用 **Edit>Template>Save File**。当你退出 MPLAB IDE 时，会有提示，会问你是否需要保存变化内容到模板（.TPL）文件中。回答是 **Yes**。

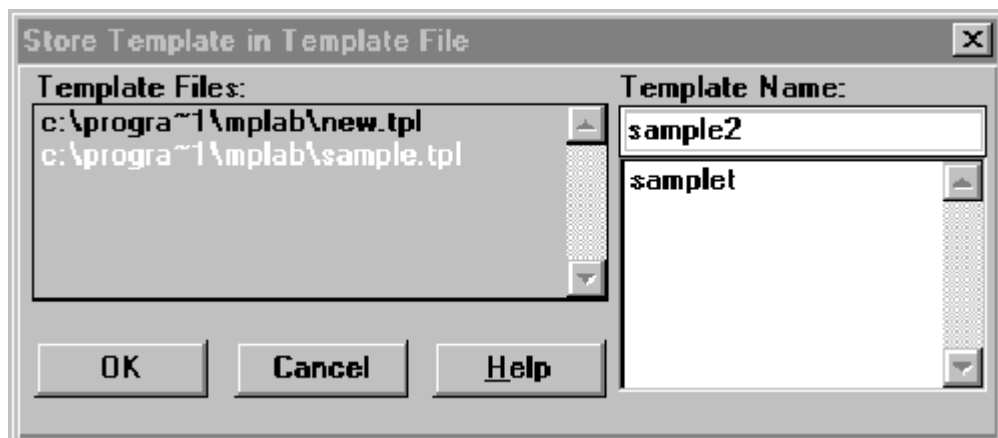


图 7.19: 模板保存对话框

#### 7.6.2.10 删除模板

如何从模板文件中删除一个模板，选择菜单命令 **Edit>Template>Delete**。选择包含你要删除的某个模板的模板文件，选择此模板，并点击 OK。此模板在 MPLAB IDE 中必须已被链接（见 7.6.2.1 节），以至于你可以选择它并删除。

注意：此删除命令仅从存储器中删除模板。想永久的从模板文件中转移，选择菜单命令 **Edit>Template>Save File**。当你退出 MPLAB IDE 时，如果你想另存为此模板（.TPL）文件，会有提示出现，回答 Yes。



图 7.20: 删除模板对话框

#### 7.6.2.11 插入模板标记

当你创建模板时，可能希望在需要自定义或要在应用程序插入的特殊代码的地方设置标记。

把光标分别放置到开发者需要插入标记的模板中的各个地方，选择菜单命令 **Edit>Template>Insert Mark**，在这些地方插入标记。这些标记将出现在源文件中，以便于开发者工作。



#### 7.6.2.12 查找模板标记

按照 7.6.2.5 的方法，插入标记后，用 **Edit>Template>Find Mark** 命令，查找模板标记。模板标记指出了你想要定制的编码或想要插入特殊的应用程序代码的地方。在每个标记处输入你想要的文本。

### 7.6.3 文本选项

此菜单显示与文本有关的命令列表。

#### 7.6.3.1 置换 “Transpose” ——Ctrl+T

选择菜单命令 **Edit>Text>Transpose**，置换左光标和右光标的特性。如果光标被放置在行首或行尾，此命令无效。

#### 7.6.3.2 大写

选择菜单命令：“**Edit>Text>Uppercase**” 改变所有当前高亮处的小写字母为大写字母。

#### 7.6.3.3 小写

选择菜单命令：“**Edit>Text>Uppercase**” 改变所有当前高亮处的大写字母为小写字母。

#### 7.6.3.4 缩进 (Indent)

要缩进一个单独的行，将光标放置到此行的任意位置，选择菜单命令 **Edit>Text Indent**。MPLAB 编辑器将此行所有的文本向右移一格。MPLAB 编辑器缩进所有的高亮行。如果没有被高亮的行，MPLAB 编辑器只缩进光标所在行。

#### 7.6.3.5 扩展 (Unindent)

回格一个单独的行，将光标放置在此行，选择菜单命令 **Edit>Text Unindent**。MPLAB 编辑器将文件向左回格一格。若不以 TAB 或空格键开头的行，MPLAB 编辑器则不能改变。对不以空格键开头的行没有影响。

## 7.12 汇编菜单

调试 (Debug) 菜单包括在你调试编码时，所需要的所有选项。

### 7.12.1 运行

运行 (Run) 菜单，允许用户控制在目标处理器上运行固件 (firmware) 程序。

#### 7.1.1.1 运行 (F9)

选择菜单命令 **Debug>Run>Run**，使处理器脱离停止状态，处理器将开始运行直到遇到一个断点或直到你暂停它。

从当前程序计数器 PC 开始执行开始程序（显示在状态栏）。在程序存储器窗口里，当前程序计数器单元被设为高亮度。当处理器运行时，单步 (Step) 和运行 (Run) 是不能用的。

#### 7.7.1.2 复位 (F6)

选择菜单命令 **Debug>Run>Reset**, 发送一个复位信号到目标处理器。它发出了一个 MCLR 信号, 将程序计数器 PC 复位到复位矢量位置。如果处理器正在运行, 它将从复位矢量地址开始继续运行。

#### 7.7.1.3 暂停 (F5)

选择菜单命令 **Debug>Run>Reset**, 迫使处理器进入停止状态。当你点击 **Halt**, 处理器将迫使进入停止状态 (程序计数器被停止), 处理器状态信息被更新。

#### 7.7.1.4 停止跟踪 (Shift+F5)

选择菜单命令 **Debug>Run>Halt Trace**, 将会停止跟踪缓冲器的捕捉数据工作, 但允许处理器继续运行。阅读关于仿真器的相应文档, 得到更多关于“暂停跟踪器”(Halt Trace)的信息。

#### 7.7.1.5 动画模式 (Animate)

选择菜单命令 **Debug>Run>Animate**, 在运行模式下, 使模拟器实际运行于连续的单步方式, 同时更新寄存器值。

使用动画 (Animate) 模式, 可以观察在特殊功能寄存器窗口或观察窗口里发生变化的寄存器数值。动画 (Animate) 模式比运行 (Run) 模式运行的慢, 但可以看见寄存器值的变化。

#### 7.7.1.6 单步运行 (F7)

选择菜单命令 **Debug>Run>Setp**, 单步执行程序。这个命令执行一个操作指令 (单周期或多周期指令), 然后使处理器进入暂停状态。在执行一个指令后, 所有窗口都会被当前的处理器状态更新。当处理器在实时 “real time” 模式下运行时, MPLAB IDE 会忽略 Step 按钮的操作。

#### 7.7.1.7 单步跳过运行 (F8)

选择菜单命令 **Debug>Run>Setp Over**, 执行当前程序计数器 PC 处的一条指令。在遇到 CALL 指令的时候, “单步跳过” (Step Over) 会运行被调用的子程序, 在 CALL 指令下面地址的一条语句暂停下来。

#### 7.7.1.8 更新所有寄存器

选择菜单命令 **Debug>Run>Update All Registers**, 更新当前指令中的所有寄存器数值。

#### 7.7.1.9 改变程序计数器

选择菜单命令 **Debug>Run>Change Program Counter**, 你可以改变当前程序计数器 PC 的数值。



图 7.21: 改变程序计数器对话框

- **PC:** 输入想要设置的程序计数器地址。
- **Change:** 点击 Change，可以设置新的计数器地址。处理器必须先停止，这样改动才会有效。
- **Close:** 从改变程序计数器会话框退出。

### 7.12.2 执行

Execute（运行）菜单选项允许用户对目标处理器中固件（firmware）的运行进行控制。

#### 7.12.2.1 执行一个操作代码

选择菜单命令 **Debug>Execute>Execute an Opcode**，可以在不修改项目代码或程序存储器内容的情况下执行一个或一组指令。执行完指令以后，你可以重新从当前程序存储器位置执行程序代码。



图 7.22: 执行操作代码对话框

- **Opcode:** 操作码。输入 16 进制的指令或符号指令(如 ADDWF Ox19)。点击 Opcode List，显示最后 8 个命令，每执行一个命令后，MPLAB IDE 会将此命令高亮度显示，你可以深入一个新的命令。(MPLAB IDE 会跟踪你输入的指令，这样，用户在操作代码列表（Opcode List）中不会得到同一个指令的两个拷贝。
- **Execute:** 执行。点击 **Eecute**，在不改变程序计数器的当前位置的情况下执行一条指令。

#### 7.12.2.2 条件断点(Conditional Break)

选择菜单命令 **Debug>Execute>Conditional Break**，会出现一个对话框，在这里可以显示处理器自动的单步执行。点击“Start”按钮，开始执行指令，直到遇见条件对话框中显示的条件才暂停下来。

**注意：**条件断点（Conditional Break）对话框在 MPLAB-ICE 或 MPLAB-ICD 开发模式中是不存在的。

有关更多的条件断点的信息，请参阅 6.8 章。

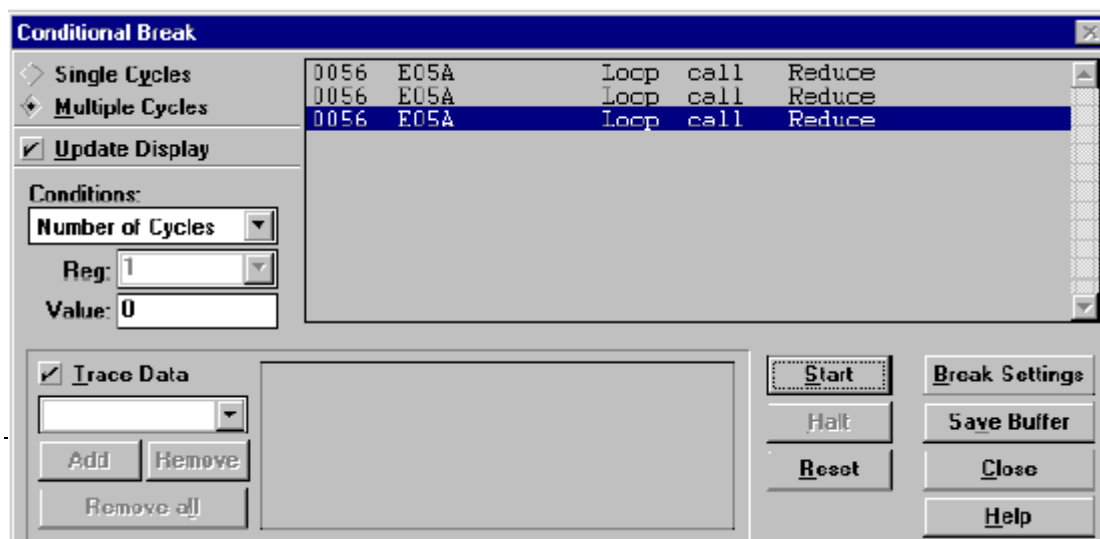


图 7-23: 条件断点对话框

- **Single Cycle:** 逐个指令的检索条件。Single Cycle 选项对每个指令抽取条件，使你可以捕获到一个特定的条件。
- **Multiple Cycles:** 多周期。只在用户定义的断点处检索条件。Multiple Cycles 选项除了在断点停止外，都是实时运行的。此选项可以允许处理中断。
- **Update Display:** 更新显示。执行条件断点，但是不更新窗口中的反汇编代码。MPLAB IDE 只存储最后的 1000 行。
- **Conditions:** 条件。你所创建的条件对你指定的寄存器的位置和输入的 8-bit 的连续值，进行测试。你可以测试以下条件：、
  1. **User Halt:** 在处理器运行时，点击 **Halt**，停止处理。
  2. **Number of Cycle:** 在 **Value** 会话框里键入周期数。
  3. **Register Value Conditions:** 寄存器数值条件：
    - Ram Addr Data Value = Equals Value Entered 等于输入的数值
    - Ram Addr Data Value < > Not Equals Value Entered 不等于输入的数值
    - Ram Addr Data Value > Greater Than Value Entered 大于输入的数值
    - Ram Addr Data Value < Less Than Value Entered 小于输入的数值
    - Ram Addr Data Value > = Greater or Equal Value Entered 大于等于输入的数值
    - Ram Addr Data Value < = Less or Equal Value Entered 小于等于输入的数值
 当测试到条件为真的时候，执行完下一个指令之前就会停止。在程序存储器窗口的下一个指令会以高亮显示。
 

**警告:** 所有的寄存器被作为 8-bit 无符号值。所以，条件<0 决不会是正确的。

**Reg:** 寄存器条件。输入一个你想在那一点上测试数据值的 RAM 地址。你所输入的地址必须是一个文件寄存器的地址。

**Value:** 在 Value 会话框中输入你想测试的 8-bit 数值。
- **Trace Data:** 跟踪数据在处理器停止时取样寄存器数值，并列显示列表里的寄存器数值。
- **Add, Remove, Remove All:** 编辑在每个断点处采集的数据变量。
- **Start:** 开始执行，并继续执行单步操作，直到按 **Halt 按钮** 处理器被停止运行，或者遇到设定的条件。
- **Halt:** 在条件中断（Conditional Break）处停止。
- **Reset:** 复位处理器。
- **Break Setting:** 打开断点设置对话框。
- **Save Buffer:** 打开保存文件（Save File）对话框，将列表框中的信息存储为一个“\*.TB”文件。

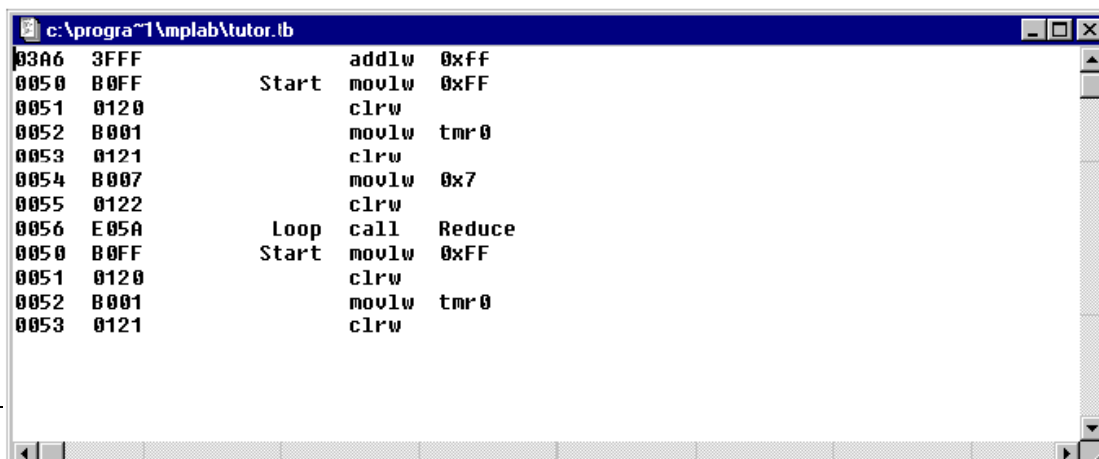


图 7-24: \*.TB 文件

- **Close:** 退出条件断点（**Conditional Break**）对话框。

### 7.12.3 模拟器激励

MPLAB-SIM 的模拟功能允许你设置一个常规时钟激励信号并允许模拟器对来自 PC 机上一个文件里设置的事件作出响应。这些文件可以通过 MPLAB 编辑器写成或其他相适的文本编辑器或字处理器，并且应当存储到当前项目的同一目录下：

模拟器功能包括：

- 异步脉冲（Asynchronous Stimulus）
- 管脚激励（Pin Stimulus）
- 时钟激励（Clock Stimulus）
- 寄存器激励（Register Stimulus）

有关模拟器激励的细节，请参阅 6.9 节的介绍。

### 7.12.4 调试位置对中

选择菜单命令 **Debug>Center Debug Location**，将当前程序计数器移动到调试窗口中央（对中）。

这个功能工作于源代码窗口（Source Code Window），程序存储窗口（Program Memory Window），以及绝对列表窗口（Absolute Listing Window）。

### 7.12.5 断点的设置

选择菜单命令 **Debug>Break Settings**，设置“断点”和“断点鉴别符”（breakpoint qualifiers）- 也即以此条件作为断点发生的条件。

有关断点的更多信息，请参阅 6.7 节。

**注意：**如果执行到断点而没有停止，选择菜单命令 **Option>Development Mode**，并点击 **Break Options** 标签。确认已选中 **Global Break Enable**（选择盒）。

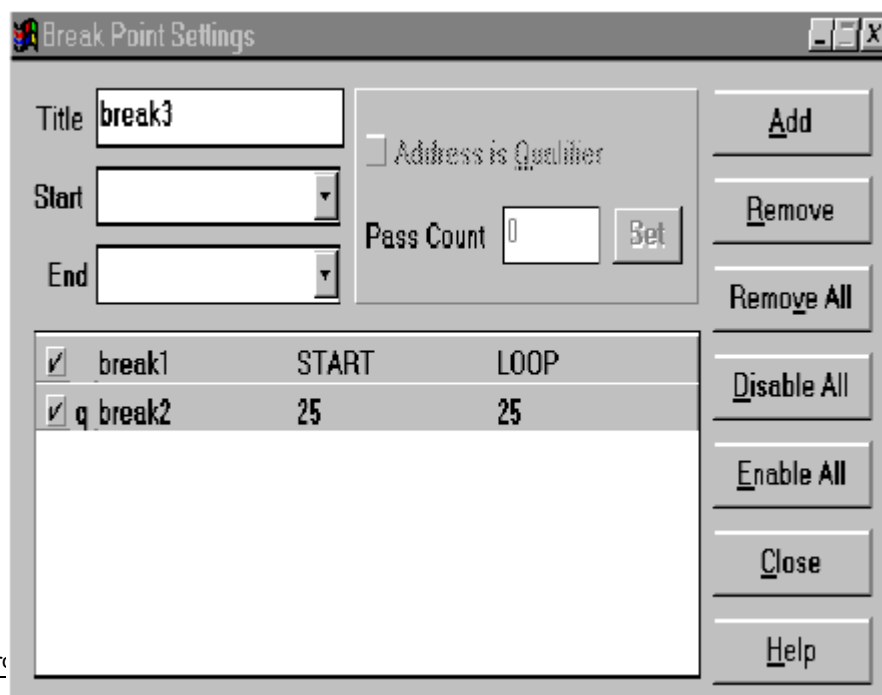


图 7-25: 断点设置对话框

7.7.5.1 常规断点设置

你最多可以定义 16 个不同名字的断点。首先，给断点命名。之后，输入起始和结束地址（可选）。点击 Add，确认断点范围的定义。

**注意：** 当使用 In-Circuit Debugger(ICD)时，只能设置一个断点设置地址。

7.7.5.2 保存断点设置

断点被作为项目中的一部分来保存。

- **Title:** 为每个断点输入一个唯一的标题（最多 32 个字符）。MPLAB IDE 接受下划线，但并不允许使用空格。

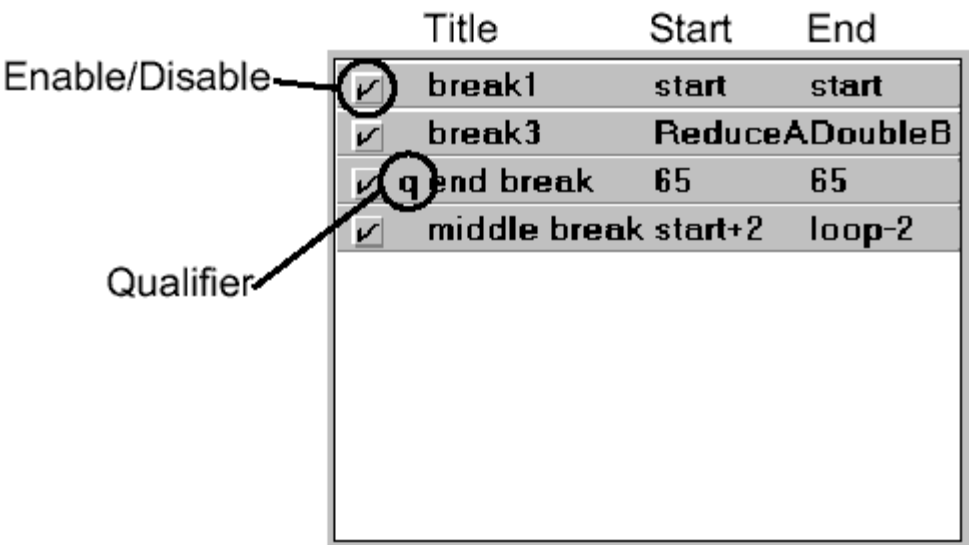
如果你没有输入标题名，MPLAB IDE 会自动输入一个默认的唯一标题名。断点范围必须有一个标题。

- **Start, End:** 输入 16 进制的开始和结束地址，或输入断点范围的标识。输入地址范围被限定在目标处理器的有效地址范围里。你可以只输入起始地址，MPLAB IDE 会自动填入同一数值的结束地址（而不是一个范围）。

你可以输入起始和结束地址或标号。如果你使用标号，MPLAB IDE 允许你通过使用命令“MAIN+2”或“EXECTIMR-10”来调整标号。当你使用标号并对一个项目再次编译时，MPLAB IDE 会分配断点到新的地址范围。

输入一个新的断点范围时，你可以使用一个已有的断点范围选项来作为中断起始点。点击对话框中你想要的选项。键入新标题，之后再点击 **Add** 键，确认你的选项。

- **Break Point List:** 此列表允许你输入大于 16 的断点范围。在选择范围上，断点设置框显示 Start, End 和 Title, 允许你编辑起始和结束地址。此列表包括以下内容：



	Title	Start	End
<input checked="" type="checkbox"/>	break1	start	start
<input checked="" type="checkbox"/>	break3	ReduceADoubleB	
<input checked="" type="checkbox"/>	end break	65	65
<input checked="" type="checkbox"/>	middle break	start+2	loop-2

图 7.26: 断点列表

**Enable/Disable:** 此按钮击活或禁止在程序存储器中使用断点地址范围。

**Qualifier:** 鉴别符。当一个选项范围显示字母“q”，则地址范围是一个“通过计数器”地址。

**Title, Start, End:** 点击列表框中的选项，显示标题，起始和结束数据。你可以通过编辑起始和结束地址，或改变标题来输入一个新的断点范围。

**Add:** 点击 **Add**，确认当前的范围并将之加入列表中。输入一个范围，激活在程序存储器中的断点。禁止断点范围，不会清除其他选项中的断点。

- **Remove:** 删除已选择的断点范围。如果不选中一个条目，则什么也不会删除。
- **Remove All:** 删除列表中所有选项断点范围。
- **Disable All:** 禁止所有选项的断点范围。
- **Enable All:** 激活所有选项的断点范围。
- **Close:** 接受已有的中断选项，并关闭断点设置对话框。
- **Help:** 显示帮助信息。

### 7.7.5.3 断点鉴别符 (Break Point Qualifiers)

该功能仅适用于 PICMASTER 和模拟器开发模式。

1. 在下拉列表中选择断点限定。
2. 从下拉菜单的列表里输入一个断点的“通过计数器鉴别点”(pass count qualifier)

MPLAB 编辑器有一个 16 位的通过计数器 (pass count)，在程序存储器中，只要有任何一个相匹配的地址，通过计数器 (pass count) 将会减少一。

当处理器处于停止状态时，你可以调整断点设置对话框中的计数值。设置通过计数器 (pass count) 的方法是：首先，设置想要的地址范围并输入所需计数值（可以达到 16-bit 长度）。计数器的值为零时，仿真器就会停止。

通过计数器 (pass count) 会在每个事件发生时减少，你可以利用该特性来为事件计数器。

- **Address is Qualifier:** “地址作为鉴别符”。你可以为断点逻辑或跟踪逻辑分配一个“通过计数器设定地址”。

你必须在列表中选择一个组件。当你作出选择后，对话框将会选中“**Address is Qualifier**”（将地址作为鉴别符）选择盒。

当你确认选中了“**Address is Qualifier**”（将地址作为鉴别符）选择盒，意味着你已经将“通过计数器”(pass counter) 分配给所选择范围内所有地址。通常将某一个地址分配给“通过计数器”是非常有用的。在选择了“**Address is Qualifier**”（将地址作为鉴别符）选择盒后，MPLAB IDE 会允许“**Pass Count Edit Box**”（通过计数器编辑框）和“**Set**”（设置）按钮。在允许了“**Pass Count Edit Box**”的情况下，你可以设置希望的“通过计数器”(pass counter) 的数值（可以达到 65534）

- **Pass Count:** 键入“通过计数器”(Pass Count) 的数值。通过计数器数值是指在处理器停止运行前，通过某一限定地址的次数。每次当程序运行遇到所定义地址时，通过计数器的数值就会减一。当通过计数器减到零的时候，处理器就被停止。
- **Set:** 点击 **Set**，输入“通过计数器”(Pass Count) 的计数值。直到你点击 **Set** 键，当前显示的“通过计数器”(Pass Count) 才会将其数值下载到仿真器的“跑表计数器”(pass counter) 里面。

### 7.7.6 跟踪设置

选择菜单命令 **Debug>Trace Setting**，显示跟踪点设置（**Trace Point Settings**）对话框，可以设置达到 16 个不同名称的跟踪点。在 MPLAB-ICE 或 MPLAB-ICD 开发模式中，跟踪点设置对话框无效。

有关跟踪点设置的详细信息，请参阅 6.7 节。

- 注意：**
1. “通过计数器限定地址”可以被分配到跟踪逻辑或断点逻辑。
  2. 跟踪点设置对话框在 MPLAB-ICE 及 MPLAB-ICD 模式中不能被使用。

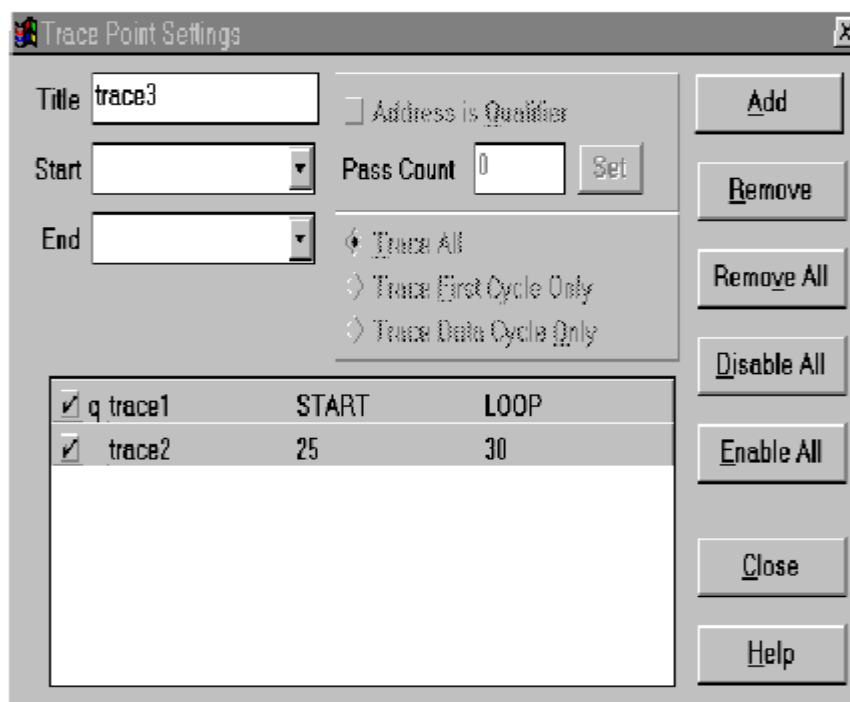


图 7.27：跟踪点设置对话框



### 7.7.6.1 常规跟踪设置

你可以定义达到 16 个不同名称的跟踪点。在输入了一个跟踪点的“标题”，“起始地址”和“结束地址”（可选项）之后，点击 **Add**（添加）按钮，确认跟踪点范围的设定。

**注意：** 当使用 In-Circuit Debugger（ICD）时，在跟踪设置对话框中只需设定一个地址。

### 7.7.6.2 保存跟踪点设置

“跟踪点”会作为“项目”中的一部分被存保存起来。

- **Title:** 标题。每个“跟踪点范围”（trace point range）都需要输入一个唯一的标题（可达 32 个字母）。MPLAB IDE 接受下划线，但不接受空格键。

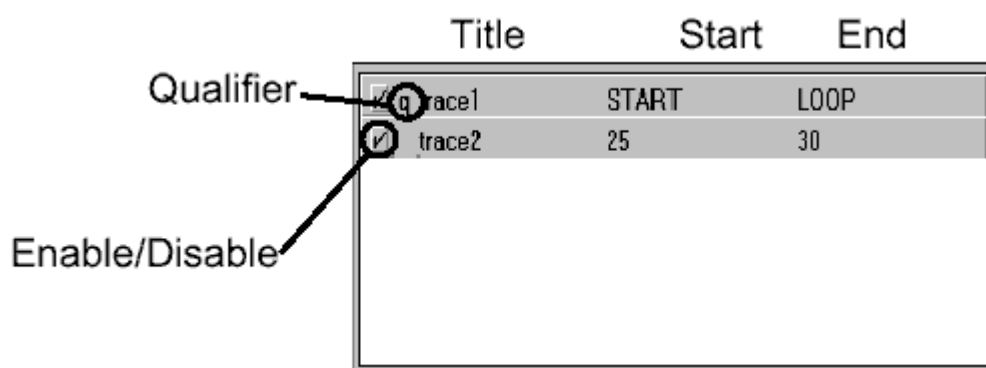
如果你选择不输入这一标题，MPLAB IDE 则自动进入默认状态，唯一的标题。每个“跟踪点范围”都需要一个标题。

- **Start, End:** 可以以十六进制方式或者以标号方式输入一个“Start”（开始）和“End”（终止）地址。地址范围被限制为目标处理器的有效的地址范围。你可以输入开始地址，MPLAB IDE 将在唯一的位置中填入相应的跟踪点终止地址（而不是一个地址范围）。

你可以用地址或标号的方式输入“开始”和“终止”的数值。如果你使用标号，MPLAB IDE 则允许你使用偏移来修改标号，比如：“MAIN+2”或“EXECTIMR-10”。当你使用标号和重新编译一个项目（标号则会因为编译而移动）MPLAB IDE 将给跟踪点分配新的地址范围。

你可以使用一个已有的跟踪点范围段，来作为输入一个新跟踪点范围的起点。在列表中，点击一个所需的条目，输入一个新的标题，然后点击 **Add**（添加），确认被定义的跟踪点范围。

- **Trace Point List:** 列表允许你输入可达 16 个的跟踪点范围。在选择一个范围时，跟踪设置对话框显示“Start, End”和“Title”，你可以编辑“开始”和“终



止”地址。列表框里包括以下几部分：

图 7.28: 跟踪点列表

**Enable/Disable:** 此按键击活或禁止在程序存储器中使用跟踪点范围。

**Qualifier:** 鉴别符。当一个选项范围显示字母“q”，则地址范围是一个“通过计数器”地址。

**Title, Start, End:** 点击列表框中的选项，显示标题，起始和结束数据。你可以通过编辑起始和结束地址，或改变标题来输入一个新跟踪点范围。

**Add:** 点击 **Add**，确认当前的范围并将之加入列表中。输入一个范围，激活在程序存储器中的跟踪点。禁止跟踪点范围，不会清除其他选项中的跟踪点。

- **Remove:** 删除已选择的跟踪点范围。如果不选中一个条目，则不删除任何东西。
- **Remove All:** 删除列表中所有选项跟踪点范围。
- **Disable All:** 禁止所有选项的跟踪点范围。
- **Enable All:** 激活所有选项的跟踪点范围。
- **Close:** 接受已有的中断选项，并关闭跟踪设置对话框。
- **Help:** 显示帮助信息。

#### 7.7.6.3 跟踪点鉴别符 (Trace Point Qualifiers)

此功能仅用于 PICMASTER 和模拟器开发模式。

1. 从下拉列表中选择一个“跟踪点鉴别符”。
2. 进入一个“跟踪点鉴别符”。

在程序存储器中，MPLAB IDE 有一个“16-bit”的“通过计数器”(Pass Counter)，遇到一个相匹配的地址它就减少一个。“

当处理器处于停止“Halt”状态中，你可以在跟踪点设置对话框中改变计数值。建立一个“通过计数器”的方法是：首先设置你希望的地址范围，然后装载一个希望的计数值计数器（可达 16 位宽度）。当计数器的数值减少至零，跟踪将开始启动。“

没发生一个事件，通过计数器减少一个。你可以使用此功能来计算一个事件所发生的次数。

- **Address is qualifier:** “将地址作为鉴别符”。你可以将一个“通过计数预置地址”分配给断点逻辑或跟踪逻辑。

在已选的列表中，必须有一个单元 (element)。选中这个单元以后，对话框将会选中“**Address is Qualifier**”选择盒。

当你选中了“**Address is Qualifier**”选择盒以后，你把通过计数器分配给了所选范围里的所有的地址。通常把通过计数分配给某一个地址是很有用的。选中了“**Address is Qualifier**”选择盒以后，MPLAB IDE 将允许“通过计数编辑框”(Pass Count Edit Box) 和设置 (Set) 按钮。选中“通过计数编辑框”以后，你可以把“通过计数”设置到一个希望的值 (可达 65,534)。

- **Pass Count:** 输入“通过计数”的数值。“通过计数”的数值是指在处理器停止之前，程序通过一个预置地址的次数。程序每遇到一个预置地址，则会减少计数器数值一次。当“通过计数”的数值达到零时，会使处理器停止运行。
- **Set:** 点击“Set”按钮，可以输入“通过计数”的数值。直到你点击“Set”按钮，当前显示的“通过计数”(pass count) 数值才会下载到仿真器的“通过计数器”(pass counter) 中。

#### 7.7.6.4 全局跟踪点环境选择项 (Global Trace Point Environment Options)

这些设置选项实用于 PICMASTER 仿真器和模拟器 (MPLAB-SIM)

- **Trace All:** 实用于所有 PICmicro MCU 芯片。允许你跟踪每一个被允许操作的跟踪地址范围。此操作显示跟踪缓冲器，仿佛可以看见处理器在每一时钟周期里取指令和执行指令的方式。
- **Trace First Cycle Only:** 适用于所有的 PICmicro MCU 芯片。滤除跟踪缓冲器里双周期指令的所有附加强制 NOP 周期 (或数据周期)。这样就可以允许跟踪缓冲器仅仅显

示确切的指令流序列。除去附加的周期也可以节省宝贵的跟踪缓冲器空间，也就可以提供更多的空间来捕捉有意义的信息。

- **Trace Data Cycle Only:** 只适用于 PIC17CXXX 芯片。只捕捉双周期的“写表”（TABLWT）和“读表”（TABLRD）指令的“数据周期”（第二个周期）。使用此特性可以用来读一个表格或从一个希望的 RAM 范围内捕捉数据，并可以实时地把捕捉到的数据以“写表”方式写入未使用的程序存储器内，以做调试功能之用。

使用“**File>Export>Export Trace Buffer**”，把捕捉的信息周期保存到一个文件里，以便进行打印图形或分析数据。

模拟器的跟踪存储器类似图 7-29 所示的情形。更多关于仿真器信息，请参阅相应型号仿真器的资料。

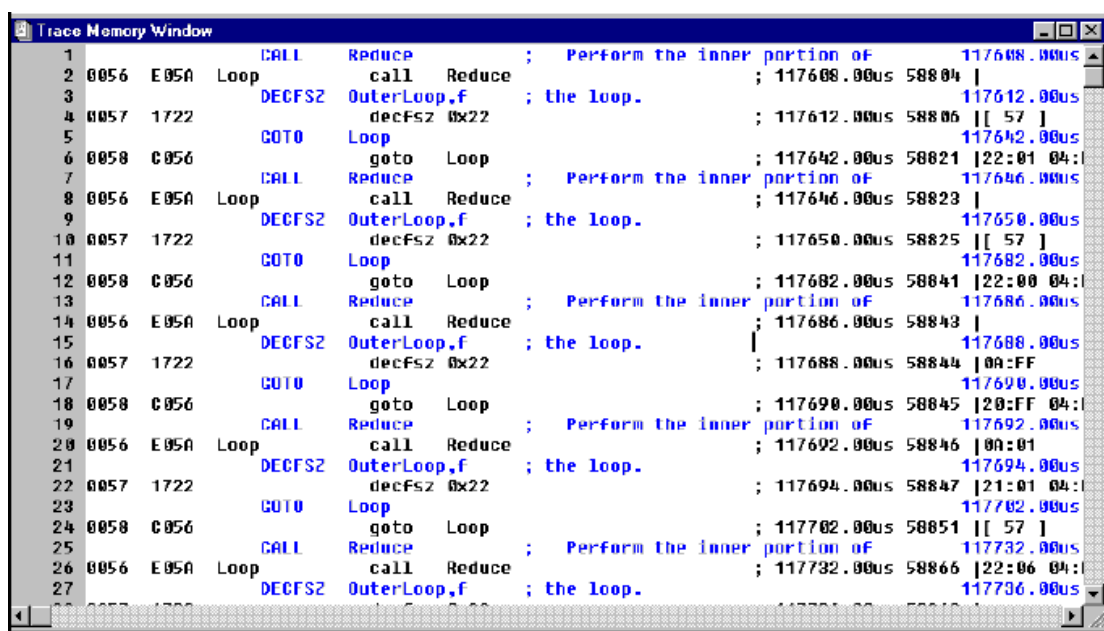


图 7-29: 跟踪存储器窗口

### 7.7.7 触发器输入/输出设置

选择菜单命令 **Debug>Trigger In/Out Setting**, 可以设置触发器输出和输入。

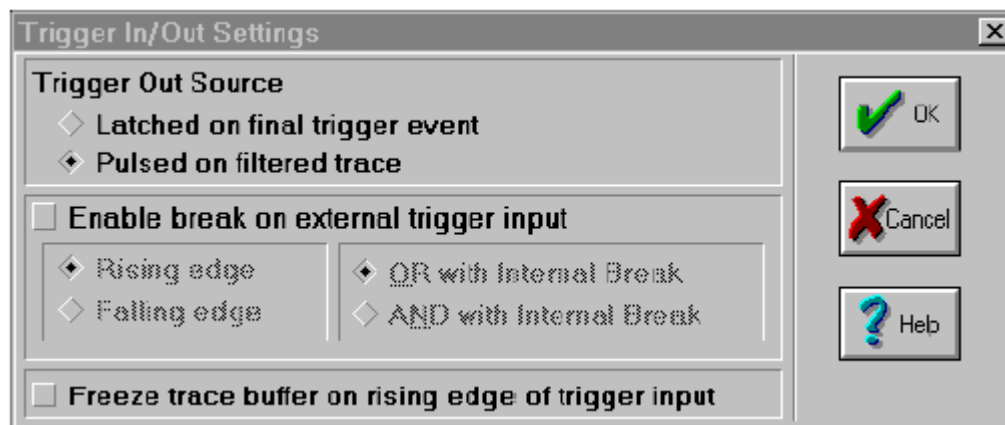


图 7.30: 触发器输入/输出设置对话框

此对话框适用于 MPLAB-ICE 和 PICMASTER 仿真器。

**MPLAB-ICE 仿真器选项:**

- **Trigger Out Source** - 触发器输出源
- **Enable break on external trigger input** - 允许外部触发输入断点
  - Rising edge : 上升沿
  - Falling edge : 下降沿
- **Freeze trace buffer on rising edge of trigger input** – 上升沿触发输入时冻结跟踪缓冲器。

**PICMASTER 仿真器选项:**

- **Enable break on external trigger input** – 外部触发输入断点
  - Rising edge : 上升沿
  - Falling edge : 下降沿
  - OR with Internal Break : 和内部断点相“或”
  - AND with Internal Break : 和内部断点相“与”
- **Freeze trace buffer on rising edge of trigger input** : 在触发输入的上升沿冻结跟踪缓冲器。

关于设定这些仿真器选项的更多信息, 请参阅《MPLAB-ICE 用户指南》或《PICMASTER 用户指南》。

### 7.7.8 触发器输出点

选择菜单命令 **Debug>Trigger Output Points**, 设置 PICMASTER 仿真器的触发输出。

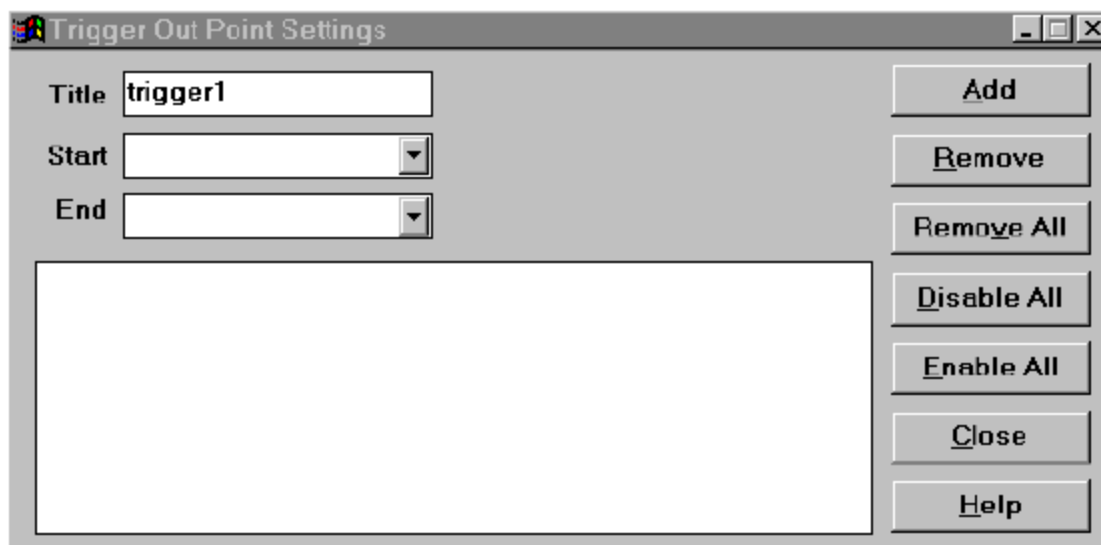


图 7-31: 触发器输出点设置对话框

更多关于设置这些仿真器的设置信息请参阅《PICMASTER 用户指南》。

#### 7.7.9 清除所有的点

选择菜单命令 ***Debug > Clear All Points***，清除所有的断点和跟踪点。

多关于这些仿真器的设置信息请参阅《PICMASTER 用户指南》。

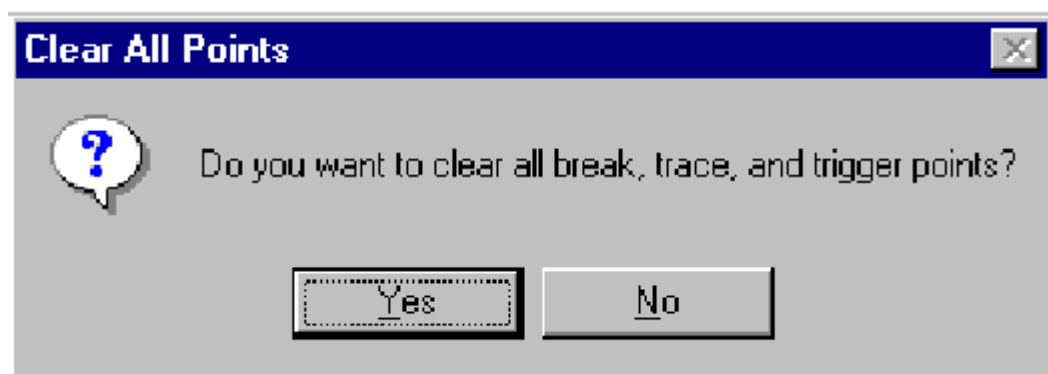


图 7.32: 清除所有点信息框

#### 7.7.10 复杂的触发器设置

选择菜单命令 **Debug > Complex Trigger Settings**，设置复杂的 MPLAB-ICE 仿真器触发事件。

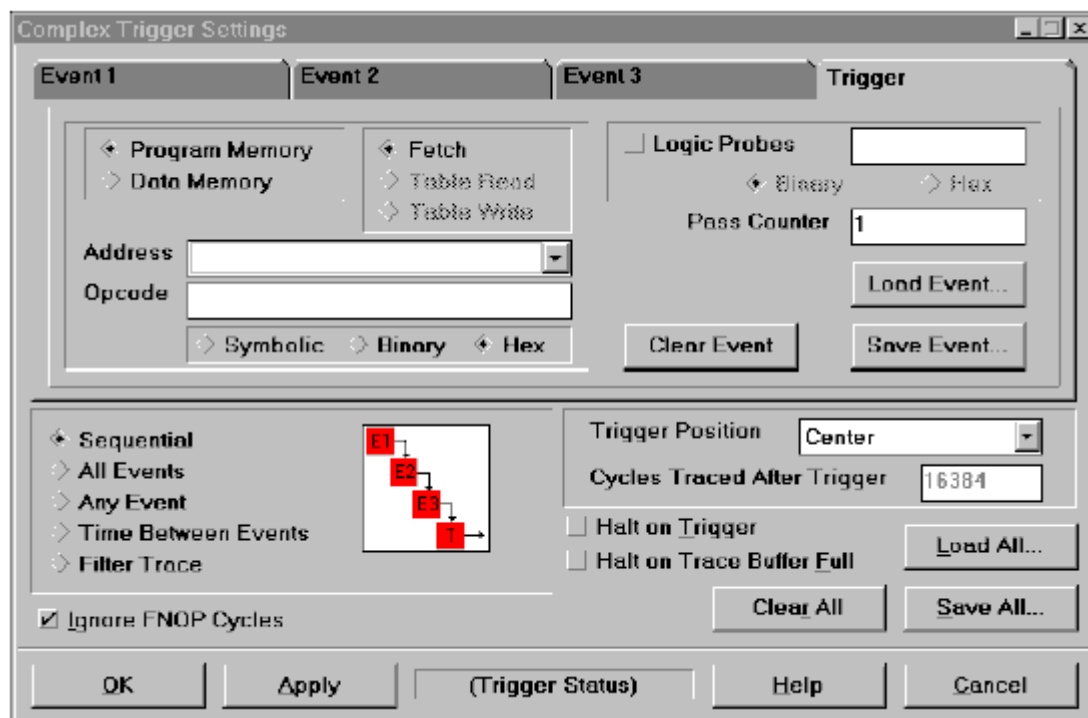


图 7-33: 复杂的触发器设置对话框

#### 7.7.11 代码区域

选择菜单命令: **Debug > Code Coverage**，允许/禁止操作 MPLAB-IDE 仿真器代码区域。

多关于这些仿真器的设置信息请参阅《PICMASTER 用户指南》。

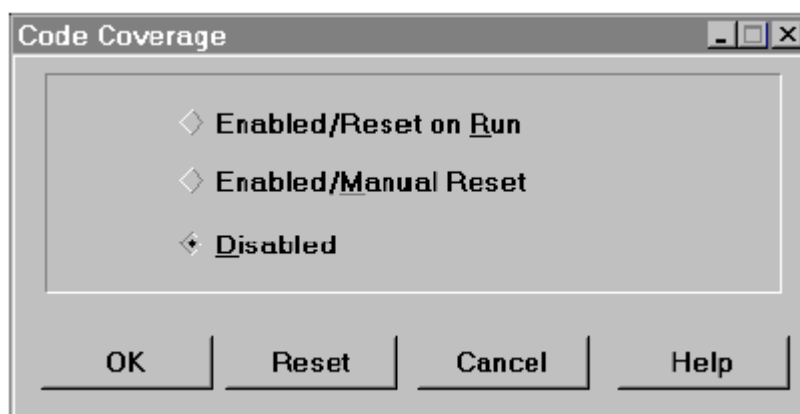


图 7-34: 代码转换会话窗口

#### 7.7.12 清除程序内存 (Ctrl+Shift+F2)

选择菜单命令 **Debug>Clear Program Memory**，设置所有的程序内存位至“1”。



图 7-35: 清除程序存储器信息框

#### 7.7.13 系统复位(Ctrl+Shift+F3)

选择菜单命令 **Debug>System Reset**，复位整个仿真器系统，包括 MPLAB-ICE 仿真器硬件（如果连通），软件和目标处理器。若是首次进入 MPLAB IDE，系统复位“System Reset”会执行相同的初始化。

**注意：** 1. 选择菜单命令 **Debug>Run>Reset**，运行一个处理器复位（MCLR）。  
2. 当更换探头或处理器模块的时候通常需要关掉仿真器电源，然后执行一个系统复位。如果你没有执行系统复位，MPLAB IDE 将不会正确的配置新的探头或处理器模块。

#### 7.7.14 上电复位（Ctrl+Shift+F5）

选择菜单命令 **Debug>Power-On-Reset**，显示“上电源复位”对话框，可以选择“POR”选项。

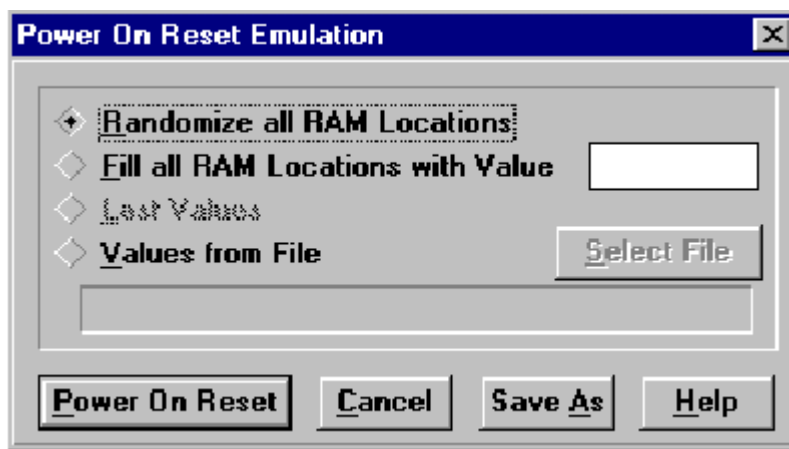


图 7-36: 上电复位对话窗口

“Power-On-Reset” 允许你用随机数或常数填满 RAM 位置。

通常,“未初始化寄存器”能导致一个程序的功能异常,很难跟踪。当应用程序首次启动,POR 将模拟随机化(Randomize)寄存器的操作。如果应用程序表现异常,Power-On-Reset 能帮助你隔离问题。

**注意:** 在芯片上 Power-On-Reset 的实现是将 MCLR 管脚连接到 VDD 上。尤其对仿真器来说, POR 没有必要复位所有 SFR。

Power-On-Reset 对话框有以下几种功能:

- “POR” (上电复位) 的时候对未知数值的寄存器进行随机化数值。
- 在寄存器里填入数值或清除寄存器。
- 将 MICROCHIP 数据手册里讲述的不同型号芯片 (只适合于 MPLAB-SIM 模拟器) 的寄存器设置为 “POR” (上电复位) 状态。“Fill with Value POR” (上电复位填充) 功能并不影响专门定义了复位初值的寄存器。
- 把当前的 POR 数值存入一个文件。
- 从一个文件中装载 POR 数值。

**注意:** 在使用 POR 对话框时, 程序存储器和断点都不受影响。

- **Randomize:** 选择 “Randomize” 以后, 在 POR 的时候, 把随机数值输入到未知数值的寄存器中。
- **Fill with Value:** 在 POR 的时候, 在 “Enter Value” (输入数值) 会话框里输入一个你想填充到寄存器中的值。
- **Last Value:** 在 POR 的时候, 选择 “Last Value”, 把最后的随机数或填充值输入到芯片中。
- **Power-On-Reset:** 在 POR 的时候, 点击 “Power-On-Reset”, 可以复位或设置选中的寄存器值。
- **Cancel:** 点击 “Cancel”, 关闭 “Power-On-Reset” 对话框, 不执行 POR。
- **Save As:** 打开对话框, 输入文件名 (\*.POR), 保存 “Power-On-Reset” 设置。
- **Values from File:** 从文件中调入数据值。点击 “Values from File”, 然后点击 “Select”, 打开一个对话框, 输入一个包含了 “Power-On-Reset” 数值的 “\*.POR” 的文件名。

## 7. 8 编程器菜单

选择一个编程器, 从 **Options>Programmer Options>Select Programmer** 菜单中, 打开 “**Select Programmer**” 对话框。一旦你改变编程器, MPLAB IDE 将被关闭。当你重新启动 MPLAB IDE 的时候, 编程器的设置将生效。你所选择的编程器菜单将出现在菜单栏中。

想获取更多关于编程器的操作信息, 请参阅编程器的有关文件。

### PICSTART Plus 菜单:

- **Enable/Disable Programmer:** 允许或禁止操作编程器。一旦编程器处于允许状态, 菜单选项会切换到 “Disable Programmer”。
- **Program/Verify:** 打开 “Program/Verify” 对话框, 允许你对所选的芯片编程 (例如, 烧写程序存储器的一部分, 或仅仅芯片的配置位)。或者校验已经正确烧写完毕的芯片。



- **Read Device:** 打开“Read Device”对话框，读取选择的芯片（例如，程序存储器的一部分，或仅仅芯片的配置位）。
- **Blank Check All:** 检查芯片是否完全为空（所有的二进制位被设置位“1”）。也可将检查的所有配置是否设置到“1”。（未编程状态）。
- **Blank Check OTP:** 该功能是为 OTP 芯片的使用所设置，OTP 芯片内有厂商已编程过的配置位。在使用该功能前，将当前显示的配置位设置为与厂商烧写的配置一样。该功能可验证是否所有的程序存储器单元被设置为“1”。并且配置位与“Configuration Bits”窗口内显示的设置一样。
- **Display Error Log:** 若有任何错误出现，在屏幕上显示错误记录。当烧写或校验结果出现芯片里的数据和程序存储器（**Program Memory**）窗口及编程器状态（**Programmer Status**）会话窗口中的数据不符合时，就会出现出错记录。
- **Erase Program Memory:** 将 MPLAB IDE 程序存储器，校正存储器和数据存储器（如果有的话）中的所有二进制数设置为“1”。
- **Erase Configuration Bits:** 将所有可以使用的配置位和 ID 位都设置为“1”。当你随后重装含有配置数据或含有配置位定义的“项目”的十六进制文件的时候，这些值在“Programmer Status”对话框中将会改变。当你装入十六进制文件或重建“项目”以后，你可以通过选择这一功能来改变你的代码的数值。
- **Reset:** 复位 PICSTART PLUS 硬件并重新建立 RS-232 通讯连接。若 PICSTART 断电时使用该选项。该选项不能复位“Program Memory”窗口中的信息以及配置位或 ID 位。

#### PRO MATE Programmer Menu:

- **Enable/Disable Programmer:** 允许或禁止操作编程器。一旦编程器处于允许状态，菜单选项会切换到“Disable Programmer”。
- **Program/Verify:** 打开“Program/Verify”对话框，使你能将显示在 **Program Memory** 窗口中的数据烧写到芯片上或 PRO MATE 插座上芯片中的数据进行校对，看是否与“Program Memory”窗口中的数据相符合。你可以选择烧写或校验目标芯片上的程序存储器或其他存储器区域。
- **Read Device:** 打开“Read Device”对话框，允许你读取选择的芯片程序存储器芯片的配置位。你可以设置程序存储器地址范围和其他读选择项。
- **Blank Check All:** 检查芯片完全为空（所有的位被设置位“1”）。也将检查所有的配置位是否为“1”（未编程状态）。
- **Blank Check OTP:** 该功能是为 OTP 芯片的使用所设置，OTP 芯片内有随同厂商已编程过的配置位。在使用该功能前，设置当前显示的配置与厂商烧写的设置一样。该功能可验证是否所有的程序存储器单元都被设置在“1”。并且配置位符合“Configuration Bits”窗口内显示的设置。
- **Display Error Log:** 当一个芯片编程或校对以后，会出现一个出错窗口。显示芯片中的存储器数据与相应 MPLAB IDE 中的数据不符合。
- **Erase Program Memory:** 将 MPLAB IDE 程序存储器，校正存储器和数据存储器（假若有的话）中的所有二进制位设置为“1”。
- **Erase Configuration Bits:** 将所有可以使用的配置位和 ID 位都设置为“1”。当你随后重装含有配置数据或含有配置位定义的“项目”的十六进制文件的时候，这些值在“Programmer Status”对话框中将会改变。当你装入十六进制文件或重建“项目”以后，你可以通过选择这一功能来改变你的代码的数值。

- **Reset Voltages** : 为所选的芯片设定默认值的 VDDMIN, VDDMAX 和 VPP。
- **Transfer to PRO MATE**: 将芯片信息传送给 PRO MATE。
- **Transfer from PRO MATE**: 从 PRO MATE 将芯片信息传回 PC 机。
- **Generate SQTP<sub>SM</sub> File** : 为芯片的系列化编号产生一个 SQTP 文件。
- **Load SQTP File** : 从计算机的磁盘里调入一个 SQTP 文件。
- **Download PRO MATE Operating System**: 下载最新版本的 PRO MATE 操作系统到烧写器里面。
- **Establish Communications**: 建立或重新建立芯片烧写器和计算机之间的 RS-232 通讯。当电源被重新连接到 PRO MATE II 的时候可以使用这个设置。但这并不复位烧写器的状态窗口, 配置位和 ID 位。

### 7. 8. 1 开发模式选择

选择命令: “**Options > Development Mode**” 可以打开一个会话窗口, 允许用户设置以下方面的参数:

- **Tools**: 工具
- **Ports**: 端口
- **Clock**: 时钟
- **Memory**: 存储器
- **Configuration**: 配置
- **Power**: 电源
- **Pins**: 管脚
- **Break options**: 断点

#### 7. 8. 1. 1 工具

选择命令: “**Options > Development Mode**” 并点击: “**Tools**” 按钮可以改变当前开发模式并可以选择处理器模块芯片类型。

点击 “**Details**” (详细) 并列双击表里面的高亮条款, 可以观察你所选择的芯片受限制的具体方面。

假如你想看当前配置的仿真器所支持的仿真芯片, 确认选择了正确的仿真器 (比如 MPLAB-ICE 或 PICMASTER 等) 并点击 “**Inquire**”, 这时候所选芯片将会出现在处理器窗口里。

假如你想保存当前在开发模式会话窗口里设置的所有信息, 可以点击 “**Apply**” 即可。

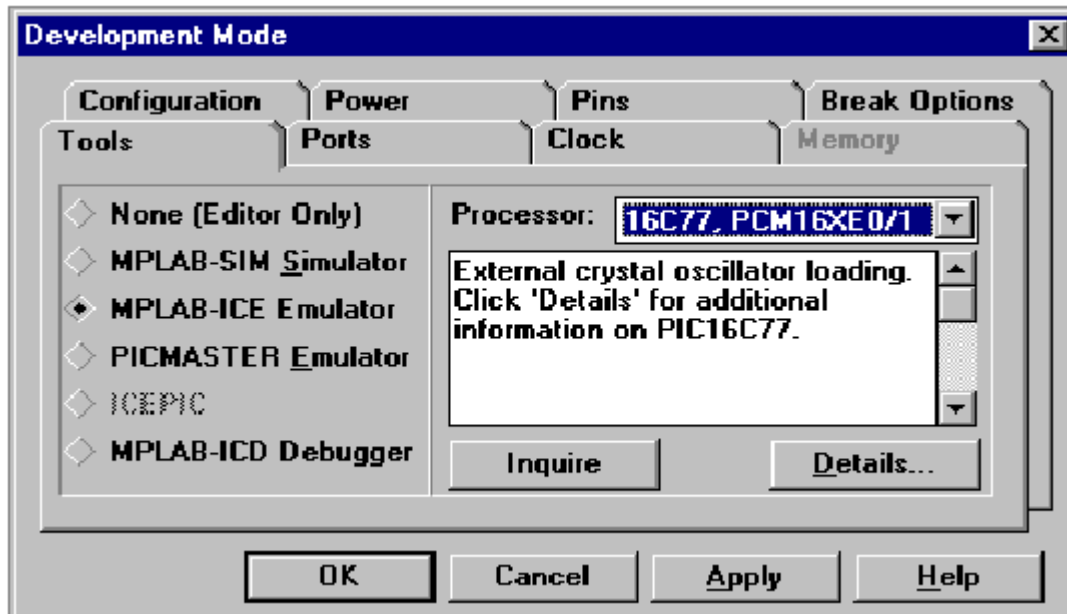


图 7-37: 开发模式会话窗口

- **None (Editor Only):** 选择纯文本方式的时候在状态栏上将会显示“EO”。在这种模式之下，模拟器和仿真器功能将会被屏蔽，无法使用。用户只能做编辑，编译，维护“项目”等操作。  
Processor: 选择你想开发的处理器型号。
- **MPLAB-SIM Simulator:** 选择 MPLAB-SIM 模拟器开发模式。  
Processor: 选择你想模拟的处理器型号。
- **MPLAB-ICE Emulator:** 选择 MPLAB-ICE 仿真器（假如已经连接了仿真器）开发模式。这时候状态栏将会显示“Ice”。  
Processor: 选择你想仿真的处理器型号。
- **PICMASTER Emulator:** 选择 PICMASTER 仿真器（假如已经连接了仿真器）开发模式。这时候状态栏将会显示“Em”。  
Processor: 选择你想仿真的处理器型号。
- **ICEPIC:** 选择 ICEPIC 仿真器（假如已经连接了仿真器）开发模式。  
Processor: 选择你想仿真的处理器型号。
- **MPLAB-ICD Debugger:** 选择 MPLAB-ICD 在线调试器（假如已经连接了仿真器）开发模式。这时候状态栏将会显示“ICD”。  
Processor: 选择你想调试的处理器型号。
- **Cancel:** 点击“Cancel”可以放弃当前的选择并退出这个窗口。
- **OK:** 点击“OK”可以确认当前的选择并退出这个窗口。

#### 7. 8. 1. 2 芯片端口

选择“**Options > Development Mode**”并点击“**Ports**”（端口）按钮为当前的操作设置端口。

假如你想保存当前在开发模式会话窗口里设置的所有信息，可以点击“**Apply**”即可。

**(Development Mode) Port:** 选择在这个开发模式下需要和计算机通讯的端口。这里的选择条款取决于在开发模式窗口下的工具选择盒里选择的开发模式。

**MPLAB-ICE 仿真器:**

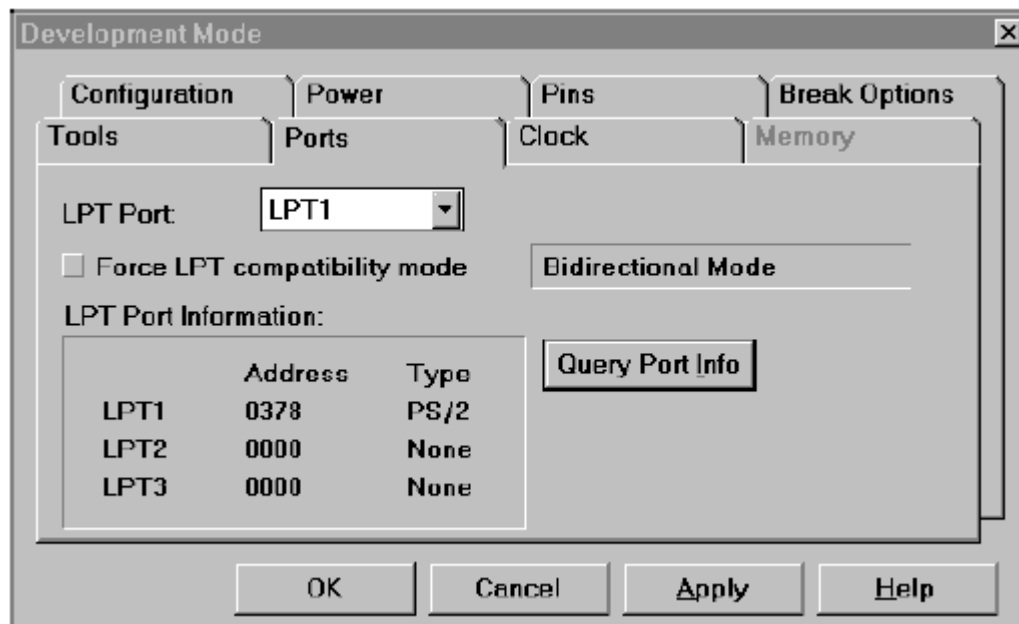


图 7-38: 端口会话窗口 - MPLAB-ICE

- **LPT Port :** 选择 LPT 并行口（比如并行 LPT1）作为 MPLAB-ICE 的通讯口。LPT 也叫做并行打印机口。
- **Force Compatibility Mode :** 选择这一选择项强迫 MPLAB-ICE 使用兼容模式通讯而非双向模式。当前工作模式在这一项的右边显示。
- **LPT Port Information :** 点击“Query Port Info”（申请端口信息）可以看见用户计算机上的 LPT 并行口设置信息。  
更多关于 MPLAB-ICE 的信息可以参考《MPLAB-ICE 用户指南》

**注意:** 假如并行口上安装了任何设备（比如打印机，扫描仪，ZIP 驱动器等设备），一定不要选择 MPLAB-ICE 仿真器模式，否则可能永久性损坏该设备。参考《MPLAB-ICE 用户指南》(DS51159)获取更多信息。

**PICMASTER 仿真器:**

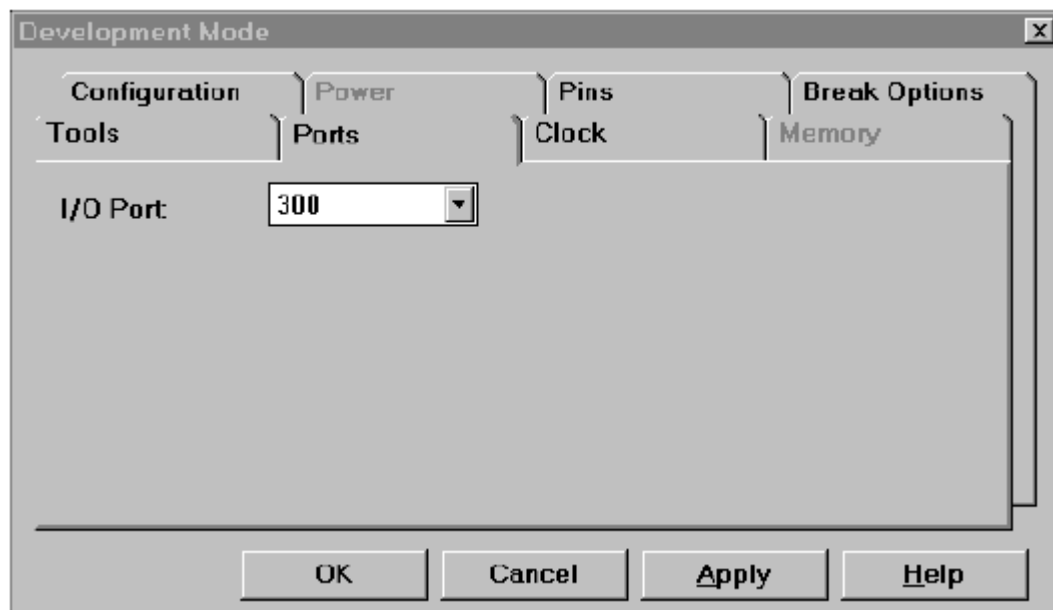


图 7-39: 端口会话窗口 – PICMASTER

- **I/O Port** : 可以选择与你的 PICMASTER 接口卡的 IO 端口地址相匹配的基本地址。

更多关于 PICMASTER 仿真器的信息可以参考《**PICMASTER 用户指南**》

#### ICEPIC 仿真器:

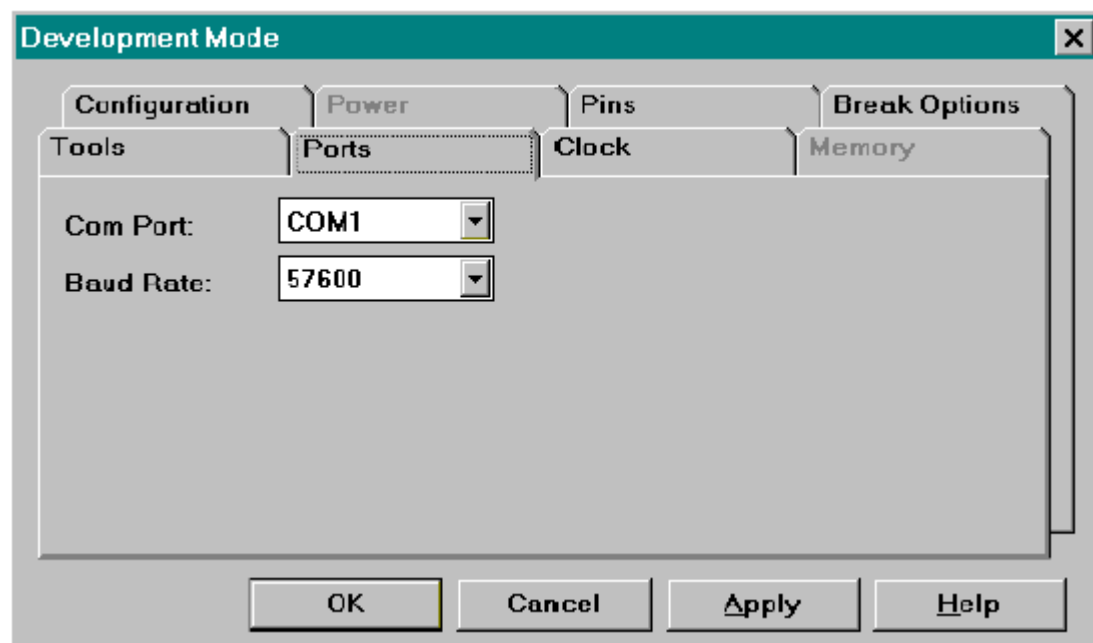


图 7-40: 端口会话窗口 – ICEPIC

- **Com Port** : 选择你的 ICEPIC 连接到计算机的串行端口。
- **Baud Rate** : 选择串行口里数据传送的波特率。

更多关于 ICEPIC 仿真器的通讯信息可以参考关于 ICEPIC 的文档资料。

### 7. 8. 1. 3 时钟频率

选择命令：“**Options > Development Mode**”并点击“**Clock**”按钮可以设置时钟频率。你可以设置频率为兆赫（MHZ），千赫（KHZ）或者赫（HZ）。参考相应的数据手册看芯片可以支持的频率大小范围。

假如你希望保存当前你在开发模式会话窗口下的所有设置，可以点击“**Apply**”按钮。

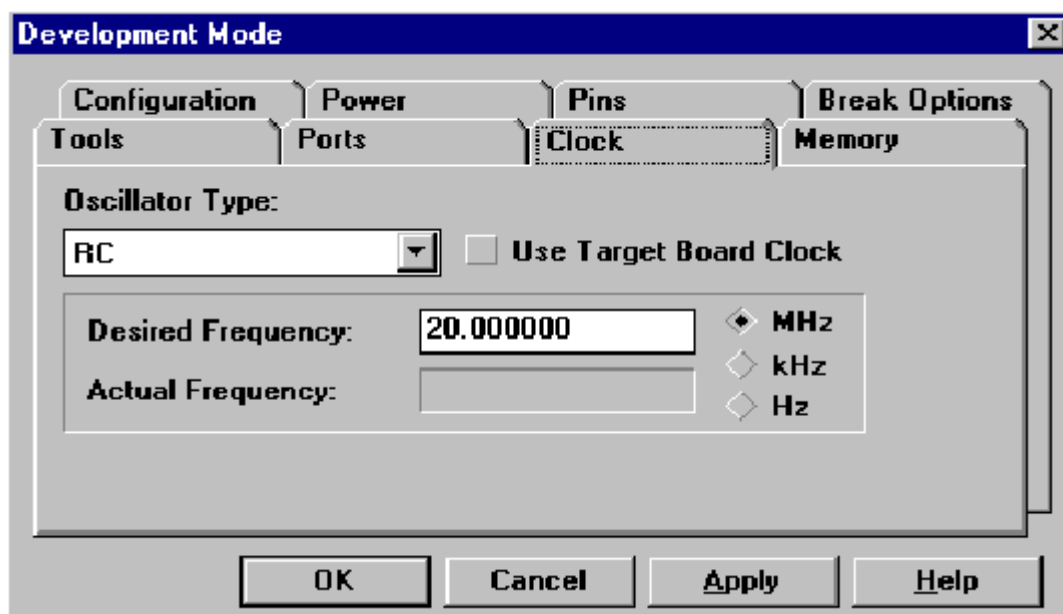


图 7-41：处理器时钟会话窗口

1. 为仿真系统选择一种振荡器类型。振荡器类型由所选择的处理器类型决定。
2. 对应你的 MPLAB-ICE 应用系统可以选择“**Use Target Board Clock**”-使用目标板上的时钟源。
3. 输入需要设置的时钟频率然后选择兆赫（MHZ），千赫（KHZ）或者赫（HZ）等频率单位。参考相应的数据手册看芯片可以支持的频率大小范围。

**注意：** 实际上的时钟频率是只为 MPLAB-ICE 所使用的。

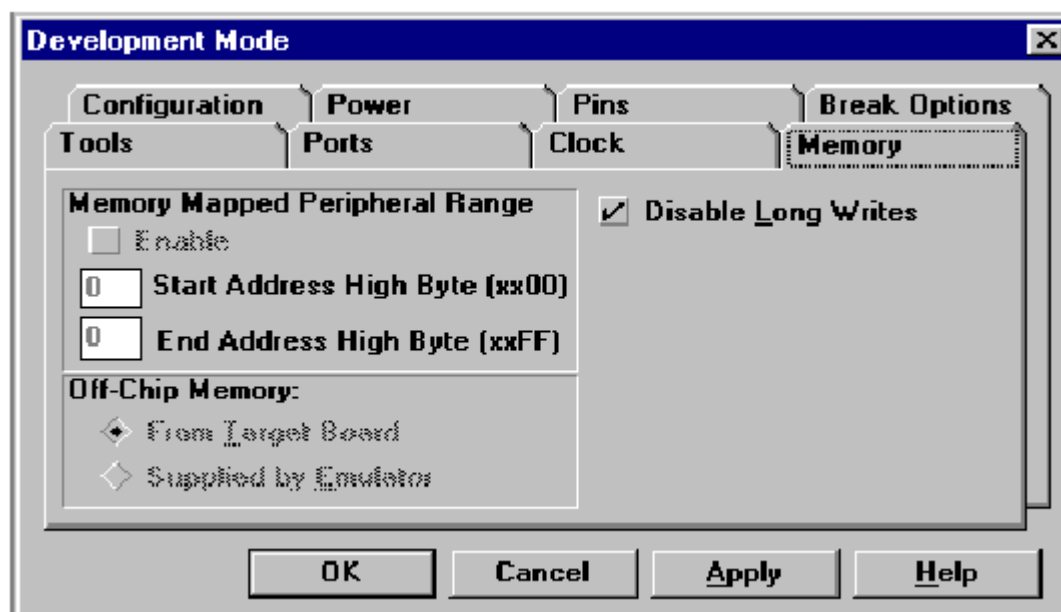
4. 点击“**Apply**”（接受），接受这些新的设置。

### 7. 8. 1. 4 存储器

选择命令：“**Options > Development Mode**”然后点击“**Memory**”选择盒，可以进入到存储器设置菜单，设置所使用的存储器。

**注意：** 这个选择项不适用于任何处理器。

假如你希望保存当前你在开发模式会话窗口下的所有设置，可以点击“**Apply**”按钮。



7-42: 存储器会话窗口

- **Memory Mapped Peripheral Range** : 允许/禁止存储器映射外设范围，并设置开始和结束的高字节地址。
- **Off-Chip Memory** : 从目标板或仿真器上选择片外存储器。
- **Disable Long Writes** : 允许/禁止长写 (long writes)。

#### 7. 8. 1. 5 设置 (Configuration)

选择命令: “**Options > Development Mode**” 并点击: “**Configuration**” 按钮盒, 可以进入设置看门狗定时器和处理器模式。

假如你希望保存当前你在开发模式会话窗口下的所有设置, 可以点击 “**Apply**” 按钮。

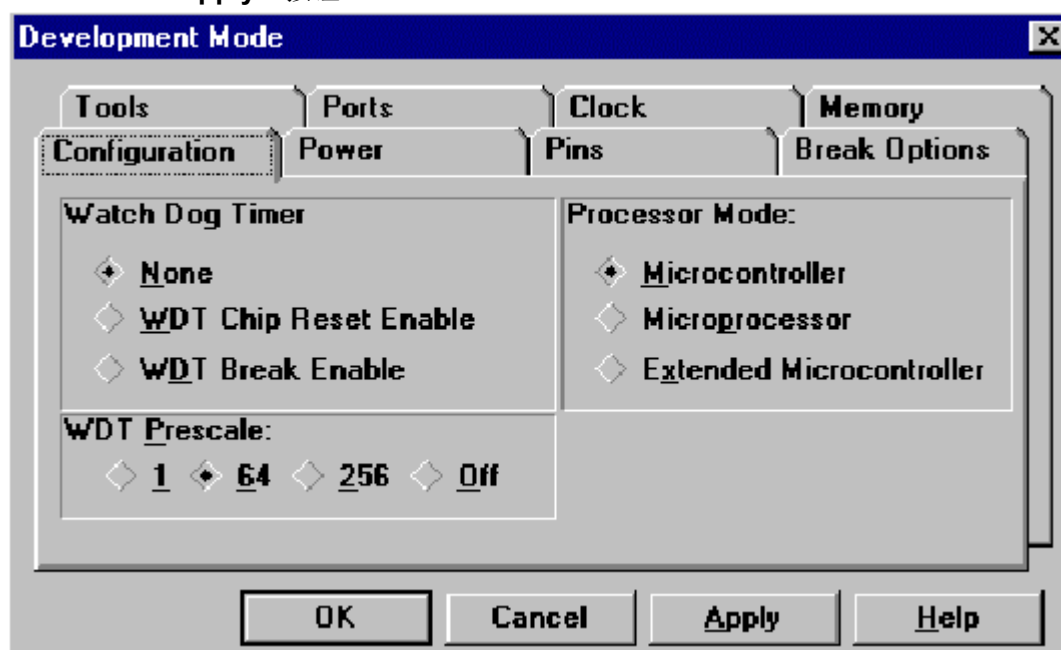


图 7-43: 设置会话窗口

- **Watchdog Timer:** 允许/禁止看门狗定时器。  
None: 禁止看门狗定时器。  
WDT Chip Reset Enable: 当看门狗定时器溢出的时候复位处理器。  
WDT Break Enable: 当看门狗定时器溢出的时候停止处理器运行。
- **WDT Prescale:** 对于 PIC17CXXX 和 PIC18CXXX 系列处理器: 假如用户使用了看门狗定时器, 记得设置其预分频器。这个设置是用来设置其预分频器的数值。  
对于所有的处理器, 看门狗定时器 (WDT) 的预分频器是通过软件设置的。(对于 PIC12CXXX, PIC14000 和 PIC16CXXX 处理器, 是在 options 寄存器里设置的: 参考相应的数据手册)。
- **Processor Mode:** 假如用户所选择的处理器允许的话, 可以选择处理器模式 (processor mode)。对于每种模式, 参考相应的数据手册。  
Microcontroller 模式: 只能操作内部存储器。  
Microprocessor 模式: 只能操作外部存储器。  
Extended Microcontroller 模式: 可以操作外部存储器和内部存储器。

#### 7. 8. 1. 6 电源

选择命令: “**Options > Development Mode**” 并点击: “**Power**” 按钮盒, 可以进入设置 处理器的电源模式和电压。

假如你希望保存当前你在开发模式会话窗口下的所有设置, 可以点击 “**Apply**” 按钮。

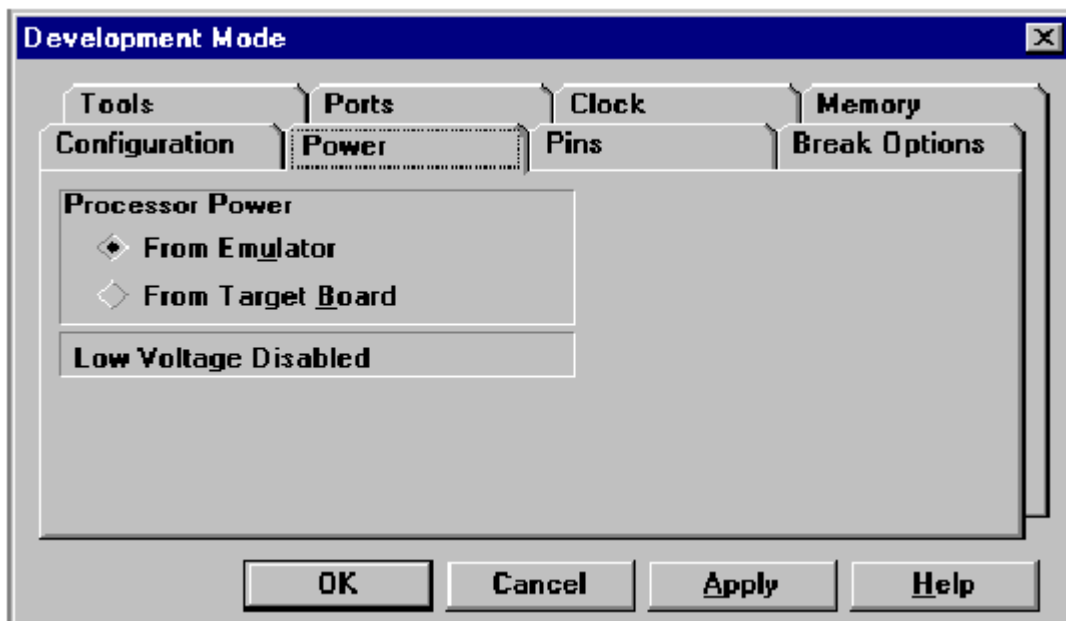


图 7-44: 电源会话窗口

- **Processor Power :** 对于仿真器系统, 请申明处理器的电源是来自仿真器 (From Emulator) 还是来自目标板 (From Target Board)。
- **Low Voltage Enabled/Disabled :** 这一行指出是否允许/禁止低电压。仿真器决定了是否允许/禁止低电压。



#### 7. 8. 1. 7 管脚

选择命令：“**Options > Development Mode**”并点击：“**Pins**”按钮盒，可以进入管脚设置。

假如你希望保存当前你在开发模式会话窗口下的所有设置，可以点击“**Apply**”按钮。

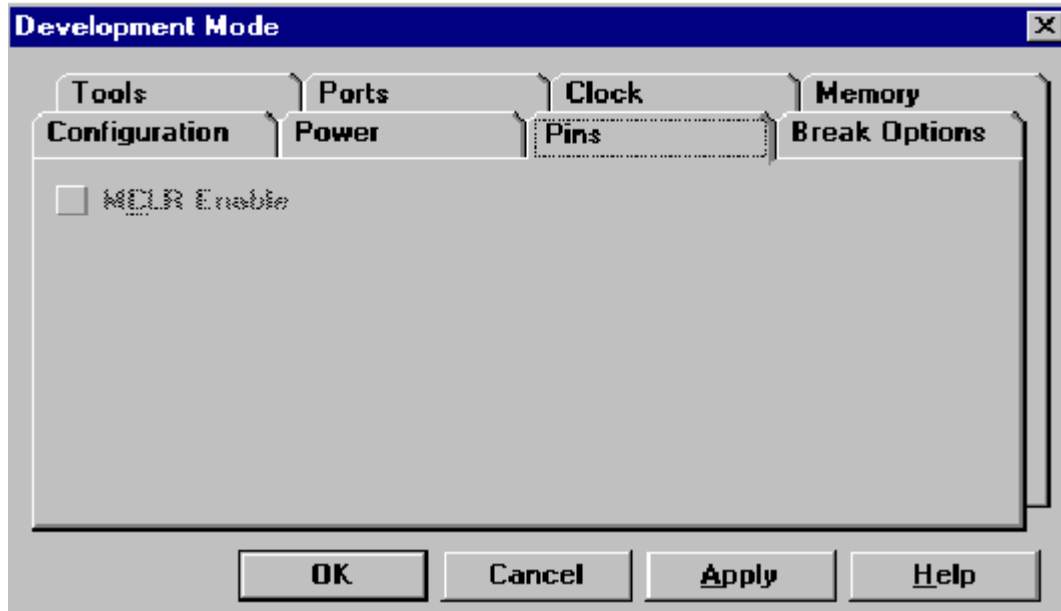


图 7-45: 管脚会话窗口

- **MCLR Enable:** 允许/禁止处理器主复位（Master Clear）。

#### 7. 8. 1. 8 断点

选择命令：“**Options > Development Mode**”并点击：“**Break Options**”按钮盒，可以进入设置全局断点和跟踪点的环境。

假如你希望保存当前你在开发模式会话窗口下的所有设置，可以点击“**Apply**”按钮。

假如没有指出的话，以下设置适合于除了纯编辑状态以外的任何开发模式。

- **Clear Break Points on Download:** 当选择这一项以后，当下载到仿真器的时候，将去除所有的断点，跟踪点，触发点和通过寄存器地址。
- **Global Break Enable:** 当选择这一项以后，允许所有的断点。假如全局断点允许（Global Break Enable）未被选择，则所有的断点都会被禁止。也可以在状态栏里面看见全局断点允许。
- **Stack Overflow Break Enable:** 当选择这一项以后，当堆栈溢出或 underflow 的时候，导致处理器暂停工作。假如你不想在某一指定的堆栈溢出或 underflow 的时候，使处理器暂停工作，就清除这一开关。选择项“Disable Stack Overflow Warning”（禁止堆栈溢出警告）可以决定在堆栈溢出或 underflow 的时候是否给出警告信息。假如你不想在堆栈溢出或 underflow 的时候是否给出警告信息，就可以清除这一开关。

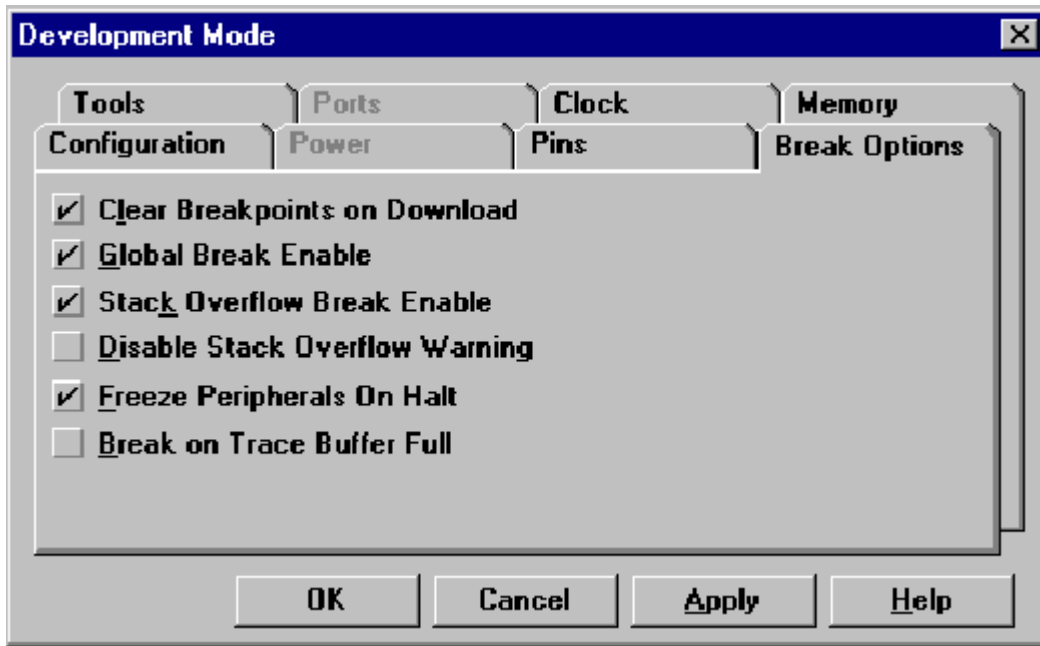


图 7-46: 断点会话窗口

- **Freeze Peripherals on Halt:** 这一选择项可以在暂停操作的时候冻结系统外设（定时器，端口，PWM 模块等）。默认模式下，这个选择是“开”的，以便用户可以得到一个正确的状态图示。当“Freeze Peripherals on Halt”（暂停的时候冻结外设）是“开”的状态，你可以在暂停以后向端口写数据，但在你执行一个单步指令之前是无法读取该端口的数据的。

注意：假如选择了“Freeze Peripherals On Halt”，当单步执行的时候 SFR 或观察窗口里的 IO 口数据将无法更新。

- **Break on Trace Buffer Full:** MPLAB-ICE 和 ICEPIC 仿真器没有此功能。当选择这一项以后，当跟踪缓冲器满的时候将导致处理器暂停运行。当跟踪缓冲器捕捉了 8K 深度的指令/周期以后就会满。对于 MPLAB-ICE 仿真器，这一功能是通过“复杂触发”（Complex Trigger）会话窗口来实现的。

## 7. 8. 2 窗口设置

### 7. 8. 2. 1 保存设置

选择命令：“**Options > Window Setup > Save Setup**”将当前配置保存到文件里。这个配置文件的默认文件名是“\*.CFG”。

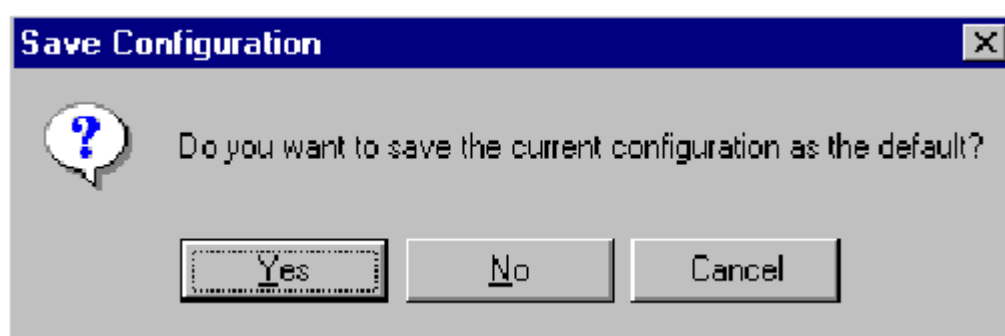


图 7-47：保存设置信息作为默认信息窗口

- **Yes:** 假如这个问题：“Do you want to save the current configuration as the default?”（保存设置信息作为默认信息），你的回答是“YES”，MPLAB-IDE 将会将当前设置做为默认的设置保存起来，下一次 MPLAB-IDE 启动的时候就可以将这个配置重新调出来。
- **No:** 假如你对这个问题的回答是“NO”的话，MPLAB-IDE 会显示出“Save Configuration”（保存配置）会话窗口。这里你可以输入想要保存配置信息的文件名和存盘目录。
- **Cancel:** 选择“CANCEL”可以退出“Save Configuration”（保存配置）会话窗口，不保存任何信息。

#### 7. 8. 2. 2 调入设置

选择命令：“**Options > Window Setup > Load Setup”可以调入一个前一次的由命令：“**Options > Window Setup > Save Setup”保存的配置信息文件。在会话窗口里面选择一个你想读入的文件并键入“Enter”。配置文件的默认文件名是“\*.CFG”。****

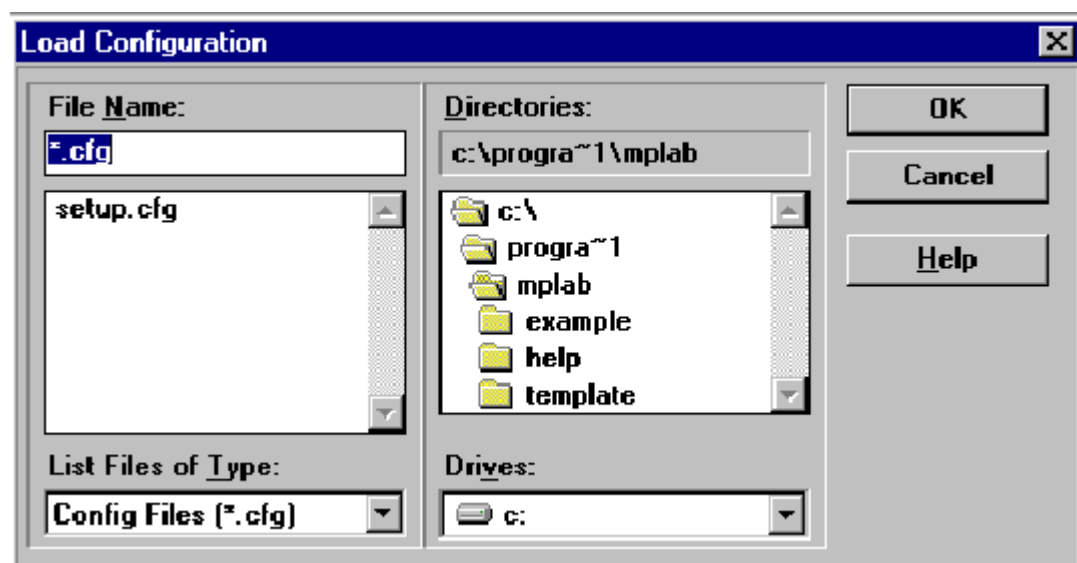


图 7-48：调入配置信息会话窗口

注意：文件“Setup.cfg”是一个由 PRO MATE II 使用的，用来烧写“安全数据产品”类器件的配置文件。不要覆盖这个文件。

### 7. 8. 2. 3 默认配置

选择命令：“**Options > Window Setup > Default Configuration**”可以调入默认用户配置信息，这个文件在你下次启动 MPLAB IDE 的时候有用。

### 7. 8. 3 当前编辑模式

选择命令：“**Options > Current Editor Modes**”可以配置当前窗口的编辑器模式。

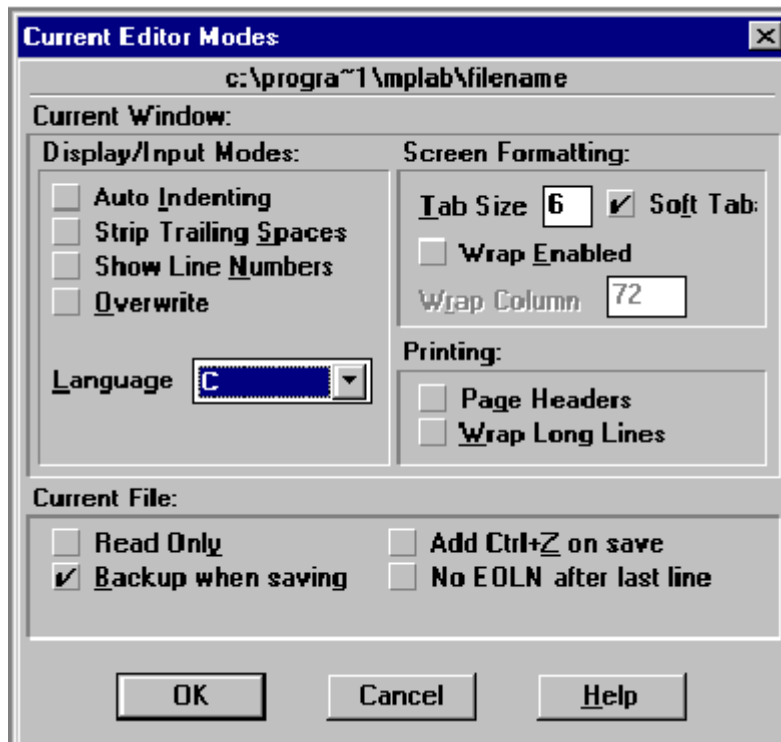


图 7-49：当前编辑模式会话窗口

#### 显示/输入模式

这些设置控制窗口里文本的显示方式以及按照你的习惯安排 MPLAB 编辑器的操作模式。

- **Auto Indenting:** 将新行缩排为与前面一行相同的等级。
- **Strip Trailing Spaces:** 删除用户按“Enter”（回车）键后，本行末尾的所有空白键。
- **Show Line Numbers:** 显示窗口里的所有行号。
- **Overwrite:** 覆盖用户输入的位于光标下的所有内容。否则 MPLAB 编辑器在光标处插入字符。
- **Language:** 旋转这个选项可以将编辑器和某一高级语言(比如 C 语言或 PARSCAL 语言)的文件编辑环境联系起来。这个设置模式可以使编辑器在书写程序的时候和该高级语言的文本布局规范一致，方

使用户编程。

#### 屏幕格式(Screen Formatting):

以下这些设置影响到关于 TAB 键和文本打包(text wrapping)。

- **Tab Size:** 定义一个 TAB 键的宽度。
- **Soft Tabs:** 当使用软 TAB 的时候插入一个空格而不是一个 TAB 宽度。  
**Wrap Enabled Wrap Column:** 如果要使用“行约束”(line wrapping), 可以在左边的“Wrap Enabled”(约束允许)选择盒加上选中标记。在“Wrap Column”(约束列)窗口里给出你想在一行里设置的约束列数。假如你不想行约束, 可以去掉选择盒里的选中标记。

#### 打印(Printing):

这个设置会影响 MPLAB 编辑器打印文件的方式。这些设置可以成为打印文件命令的默认设置, 但是在打印的时候依然可以不管这些设置。

- **Page Headers:** 在每一页的开头打印文件名, 页号和日期。
- **Wrap Long Lines:** 将超出一行宽度的太长的行打包, 转入下一行里打印。

#### 当前文件(Current File):

这些设置关系到文件模式。

- **Read Only:** 防止用户用这种方式的文件保存文件。
- **Backup When Saving:** 当用户写文件到磁盘的时候拷贝一个同样的备份文件。
- **Add Ctrl+Z On Save:** 当保存文件到磁盘的时候自动在文件的末尾加上一个 Ctrl+Z 符号。
- **No EOLN after last line:** 当保存文件到磁盘的时候自动在最后一行的末尾加上行结束标志(CR-LF 或 LF)。

### 7. 8. 4 复位编辑器模式:

选择命令: “**Options > Reset Editor Modes**” 将当前窗口的模式复位到默认得模式。

### 7. 8. 5 环境设置:

选择命令: “**Options > Environment Setup**” 可以打开一个会话窗口, 允许用户改变下来范围里的设置:

- **General Options - 基本选项**
- **Project Template Options - “项目” 剪贴板选项**
- **Files - 文件**
- **Default Editor Modes - 默认编辑器模式**
- **Key Mappings - 对应的默认键**
- **Colors - 颜色**

#### 7. 8. 5. 1 基本选项

选择命令: “**Options > Environment Setup**” 并点击 **General 按钮**, 可

以进入一个会话窗口，在这里可以设置屏幕字体，工具栏选择，符号显示宽度以及全局开关等参数。

假如用户想保存在环境设置会话窗口“**Environment Settings**”里的所有设置，只要按“**Apply**”按钮即可。

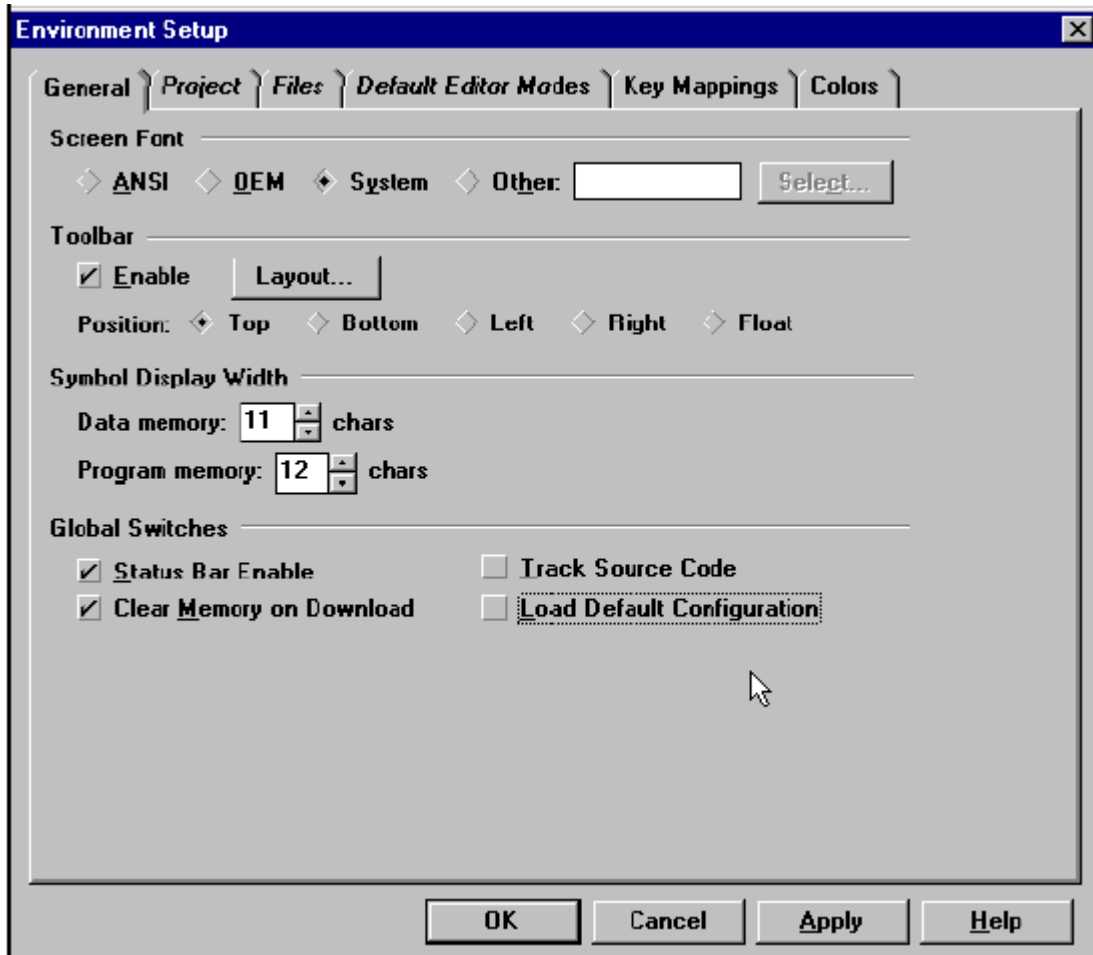


图 7-50：基本会话窗口

#### 7. 8. 5. 1. 1 屏幕字体

当使用命令：“Options > Environment Setup”并点击“**General**”按钮，会出现屏幕字体设置，允许用户选择一种 MPLAB IDE 屏幕显示字体。

1. 在环境设置会话窗口里面选择 ANSI, OEM, 系统或者其他的屏幕字体。
2. 假如用户选择的是其他模式(Other)的字体，点击“**Select**”可以出现字体(Font)会话窗口。
3. 在选择字体会话窗口里，选择用户所喜欢的字体，然后点击“**OK**”按钮返回到环境设置会话窗口(Environment Settings)。

假如用户想保存当前在环境设置会话窗口“**Environment Settings**”里的所有设置，只要按“**Apply**”按钮即可。

#### 7. 8. 5. 1. 2 工具栏设置

选择命令：“Options > Environment Setup”并点击“**General**”可

以进入到 MPLAB IDE 工具栏设置环境。

在这个环境下用户可以做以下工作：

- 为屏幕上的工具栏选择一个位置并改变其他显示设置。可以安排的位置有上部，底部，左边，右边和浮动(Float)。

- 选择允许 checkmark，使得 MPLAB IDE 可以显示工具栏。

在环境设置会话窗口里的通用(General)标签里面点击 “Layout”，可以看见工具栏设置会话窗口：

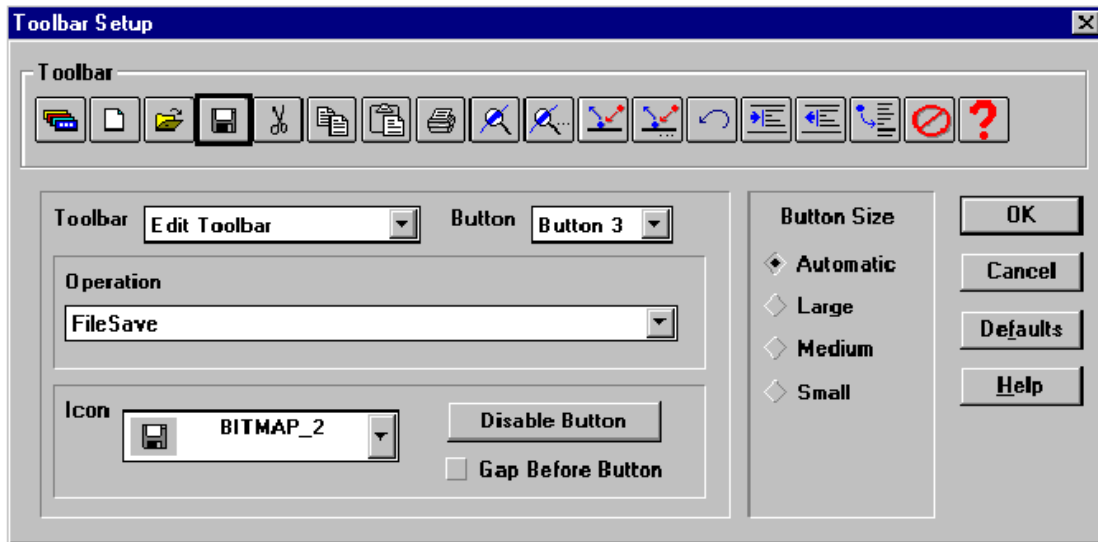


图 7-51：工具栏设置会话窗口

从工具栏设置会话窗口里面，用户可以：

- 增加一个按钮到当前工具栏里面或者编辑一个当前存在的按钮。
- 从工具栏里面移走一个按钮。
- 排列工具栏按钮。
- **Toolbar:** 选择将要编辑的工具栏：Edit, Debug, Project 或者 User。
- **Button:** 选择一个工具栏按钮的位置。工具栏有 16 种位置可以安排。
- **Operation:** MPLAB IDE 将要采用用户选择的按钮位置。
- **Icon:** 选择一个按钮放置在用户选择的按钮位置作为显示按钮图标。
- **Disable Button:** 在工具栏选择位置禁止某一个工具栏按钮。
- **Gap Before Button:** 在工具栏按钮前面设置一个隔离区域。
- **Button Size:** 改变工具栏按钮的尺寸。按钮尺寸的可选项有：自动 (Automatic)，大 (Large)，中 (Medium)，和小 (Small)。
- **Ok:** 确认当前显示的工具栏
- **Cancel:** 返回到工具栏以前的设置。
- **Defaults:** 恢复所有的工具栏默认设置，撤消用户对工具栏所作的修改。

假如用户想保存当前在环境设置会话窗口 “Environment Settings” 里的所有设置，只要按 “Apply” 按钮即可。

### 增加和编辑工具栏按钮(Adding and Editing Toolbar Buttons)

假如里想通过工具栏来执行一个操作，但当工具栏里没有这样的按钮，比如“File Save”这样的按钮的时候，用户可以增加一个新的按钮或者编辑一个当前有的按钮，使得它可以完成这个功能。

1. 选择命令：“Options > Environment Setup”并点击“General”察看当前有的工具栏按钮设置。点击 **Layout** 可以进入到工具栏设置环境。
2. 在工具栏下拉菜单里面选择你想增加按钮的工具栏。你选择的工具栏会出现在工具栏设置会话窗口上部的工具栏区域里面。
3. 在工具栏设置会话窗口上部的工具栏区域里面，在你希望分配新按钮的地方点击。

注意：假如当前位置没有一个红色的圈，你在下面选择的按钮和操作将会替换当前的按钮和操作。为了保护当前分配的按钮，将你希望的按钮放置在当前分配的按钮右边。你不能将按钮添加到工具栏上的第 17 个位置。

4. 从“Icon” (图标)下拉菜单里面，选择你希望在工具栏里显示的按钮。
5. 从“Operation” (操作)下拉菜单里面选择你希望在按这个按钮以后系统执行的操作命令(比如 EditText Indent)。
6. 点击“OK”保存你所作的修改(或者点击 **Cancel** 废除这些操作)然后返回到环境设置会话窗口。

### 禁止(去除)一个工具栏按钮

假如用户不想通过工具栏来执行某一个操作的话，可以禁止其工具栏按钮位置。这个操作会将按钮从工具栏里面去除。

1. 选择命令：“Options > Environment Setup”并点击“General”察看当前有的工具栏按钮设置。点击 **Layout** 可以进入到工具栏设置环境。
2. 在工具栏下拉菜单里面选择你想禁止(去除)的按钮。你选择的工具栏会出现在工具栏设置会话窗口上部的工具栏区域里面。
3. 在工具栏设置会话窗口上部的工具栏区域里面，在你希望禁止(去除)的按钮上点击。假如你选择了工具栏按钮，用户点击按钮的时候将会执行的相应操作就会出现在操作下拉菜单里。
4. 在图标下拉菜单的右边点击“Disable Button”，在这个按钮的边沿会出现红色的边框并替换这个按钮。
5. 点击“OK”保存你所作的所有操作(点击 **CANCEL** 会忽略所有的操作。

当你返回到 MPLAB-IDE 桌面的时候，这个按钮的位置将变为空白。

### 在工具栏上排列按钮

用户可以安排多组按钮在工具栏上，以便执行相关的一些操作。比如，你可能希望将编辑器相关的按钮排列在一起并与其他的功能的按钮分隔开来，组之间的按钮可以由工具栏上的一小段空格分隔



开。

1. 选择命令：“**Options > Environment Setup**”并点击“**General**”察看当前有的工具栏按钮设置。点击 **Layout** 可以进入到工具栏设置环境。
2. 在工具栏下拉菜单里面选择你想自定义安排的按钮。你选择的工具栏会出现在工具栏设置会话窗口上部的工具栏区域里面。
3. 在工具栏设置会话窗口上部的工具栏区域里面，点击没一组的第一个按钮(最左边)，然后选择“**Gap Before**”。点击这一组里余下的按钮并重新选择(清除)“**Gap Before**”。
4. 点击“**OK**”保存你所作的所有操作(点击 **CANCEL** 会忽略所有的操作)

#### 7. 8. 5. 1. 3 符号显示宽度

选择命令：“**Options > Environment Setup**”并点击“**General**”标签，“符号显示宽度”(Symbol Display Width)区域允许用户定制 MPLAB IDE 显示符号信息时的字符宽度。

- **Register Variables:** (寄存器变量)允许用户选择 6 个到 32 个字符宽度。
- **Address Labels:** (地址标号)允许用户选择 6 个到 32 个字符宽度。

#### 7. 8. 5. 1. 4 全局开关(Global Switches)

选择命令：“**Options > Environment Setup**”并点击“**General**”“全局开关”设置区域允许用户对下列选择项进行“开”或“关”。

- **Status Bar Enable:** 对状态栏进行“开”或者“关”。
- **Clear Memory on Download:** 当选择这个选择项的时候，这个全局开关将在 MPLAB IDE 下载数据到仿真器之前清除存储器。这个功能将所有的程序存储器空间里的位设置为“1”。
- **Load Default Configuration:** 当选择这个选择项的时候，MPLAB IDE 会在启动的时候调入用户默认设置。想改变默认窗口设置，打开你想在启动的时候调入的默认窗口，选择命令：“**Options > Window Setup > Save Setup**”然后点击“**Yes**”。MPLAB IDE 窗口设置可能包含任何 MPLAB IDE 所有的窗口。MPLAB.CFG 是用户配置的默认文件。
- **Track Source Code:** 当选择这个选择项的时候，执行单步操作的时候，MPLAB IDE 会跟踪当执行到的源程序行。假如你有\*.HEX 文件却没有\*.COD 文件，可能希望关调这个特性。

假如用户想保存当前在环境设置会话窗口“**Environment Settings**”里的所有设置，只要按“**Apply**”按钮即可。

#### 7. 8. 5. 2 “项目”剪切板选择项

选择命令：“**Options > Environment Setup**”并点击“**Project**”标签可以显示和编辑“项目”的语言和目录路径的默认选择项。你选择的默认设置将择参数将在随后建立的“项目”里被使用。已经建立了的“项目”将不受影响。

**注意:** 想单独改变某一个“项目”的设置，可以选择“**Project > Edit Project**”

“命令，按照第三章：“开始使用 MPLAB IDE – 一个辅导案例”里的介绍来设置。

假如用户想保存当前在环境设置会话窗口“**Environment Settings**”里的所有设置，只要按“**Apply**”按钮即可。

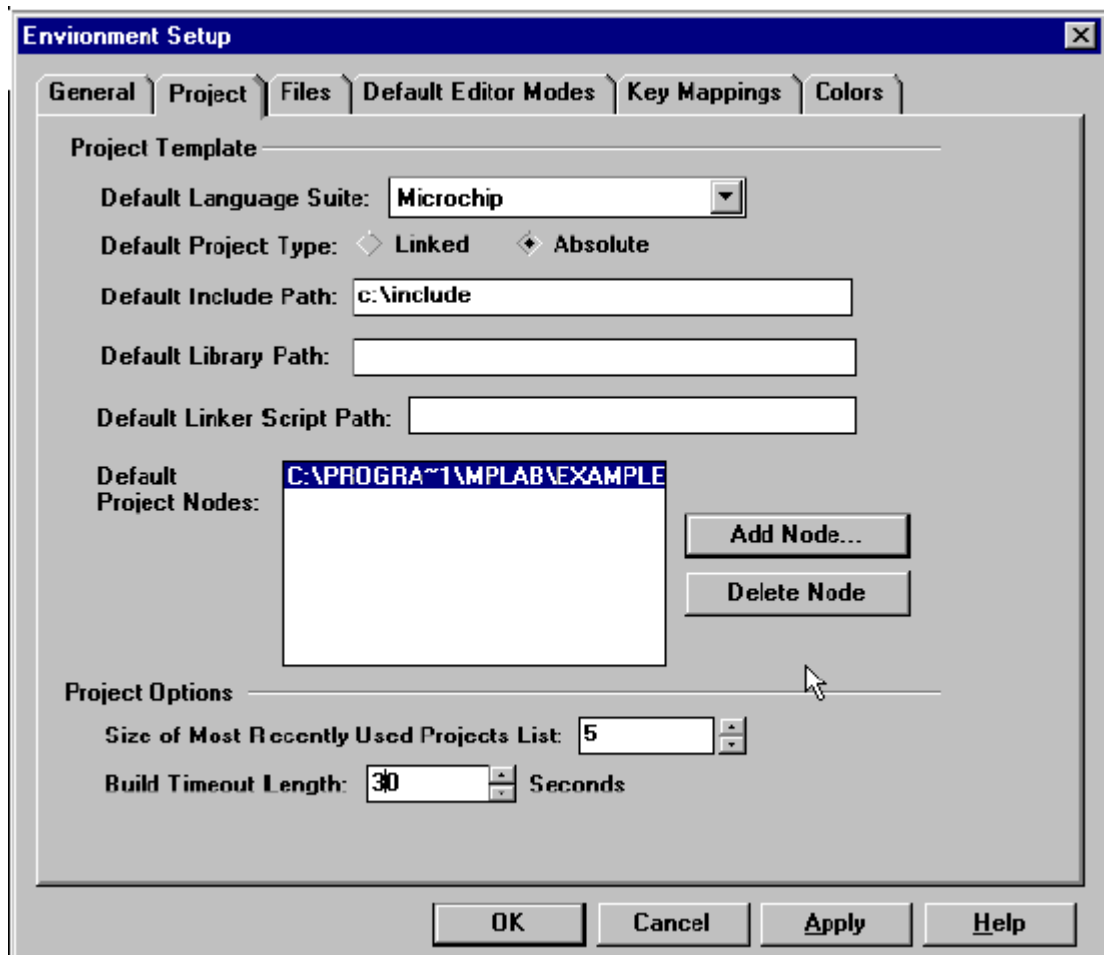


图 7-52: “项目”剪切板会话窗口

- **Default Language Suite:** 用户可以在“环境设置”会话窗口的“项目”剪切板标签里，位于顶部的下拉菜单里选择一个开发语言套件(比如汇编器，编译器等)。你可能需要为这个套件安装开发语言工具。你可以选择一个在你的 PC 机上没有安装的套件，但是“项目”将不会建立。  
假如你在安装 MPLAB IDE 之前安装过语言工具套件，你必须在安装语言工具会话窗口里面给出关于语言工具的 MPLAB IDE 信息。
- **Default Project Type:** 默认“项目”类型。在包含文件路径窗口里，输入关于包含“项目”文件(比如头文件)的默认目录。假如没有为一个“项目”设置包含文件路径，则“创建项目”(Make Project)将无法发现包含文件并完成创建整个“项目”。由于在 AUTOEXEC.BAT 文件里面包含了 MCC\_INCLUDE 变量，所以汇编器、编译器可以找到这些包含文件，从而可以顺利完成创建。然而这种方法的效率却很低。
- **Default Library Path:** 在编辑项目会话窗口里的库文件窗口里，输入默认的“项目”库文件所在目录的路径。这个选择项只有在用户选择了

“已链接项目” (Linked project)类型何时才设置。

- **Default Linker Script Path:** 在编辑“项目”会话窗口的链接器描述窗口里，输入项目的链接器描述文件所在目录的默认路径。这个选择项只有在用户选择了“已链接项目” (Linked project)类型的时候才设置。

每个路径窗口都可以包含一个和多个由分号隔离的绝对或相对路径。例如：

```
c:\mplab\projects\mproj\include;  
c:\include\h;..\sys
```

- **Default Project Nodes:** 默认项目节点。指出任何你想包含在你的项目剪切板里的没有编译(项目，库函数，链接器描述文件)的节点。点击“**Add Node**”可以指出你想添加的每个节点的驱动器，目录和文件名。若想从你的“项目”剪切板里删除某一个节点，可以选中这个节点并点“**Delete Node**”。

你选择文件的目录默认为“项目”剪切板里确定的默认浏览目录。假如你想从另外一个目录里选择文件，可以从“驱动器”的下拉菜单里面选择相应的驱动器(“增加节点”会话窗口的右下方)，双击“增加节点”会话窗口右上方的文件夹，确认你需要选择的源文件被选中，并出现在文件列表下面的窗口里面。

- **Size of Most Recently Used Projects List:** 最近使用过的“项目”列表大小。可以让用户选择在 MPLAB IDE 项目菜单的下面的最近使用过的项目列表长度。当你减少这个数目，文件名将会从列表里面被移走。当你以后又增加这个数值得时候，这些文件名将不会再被 MPLAB IDE “记住”(显示出来)。当你增加这个数值以后，以后所使用过的文件将会被记忆在系统里。默认的文件数目是 5 个。
- **Build Timeout Length:** 为创建功能设置最长的时间段。假如创建工作超过了设置的允许时间段，就会给出一个超时信号，也就不会生成十六进制文件。假如创建超时被设置为“关”，那么就不会产生超时信号。

#### 7. 8. 5. 3 文件选择项

选择命令：“**Options > Environment Setup**”并点击“**Files**”标签，可以为 MPLAB IDE 设置默认的浏览目录并可以调整“最近使用过的文件列表” (Most Recently Used File List)数目。假如用户想保存当前在环境设置会话窗口“**Environment Settings**”里的所有设置，只要按“**Apply**”按钮即可。

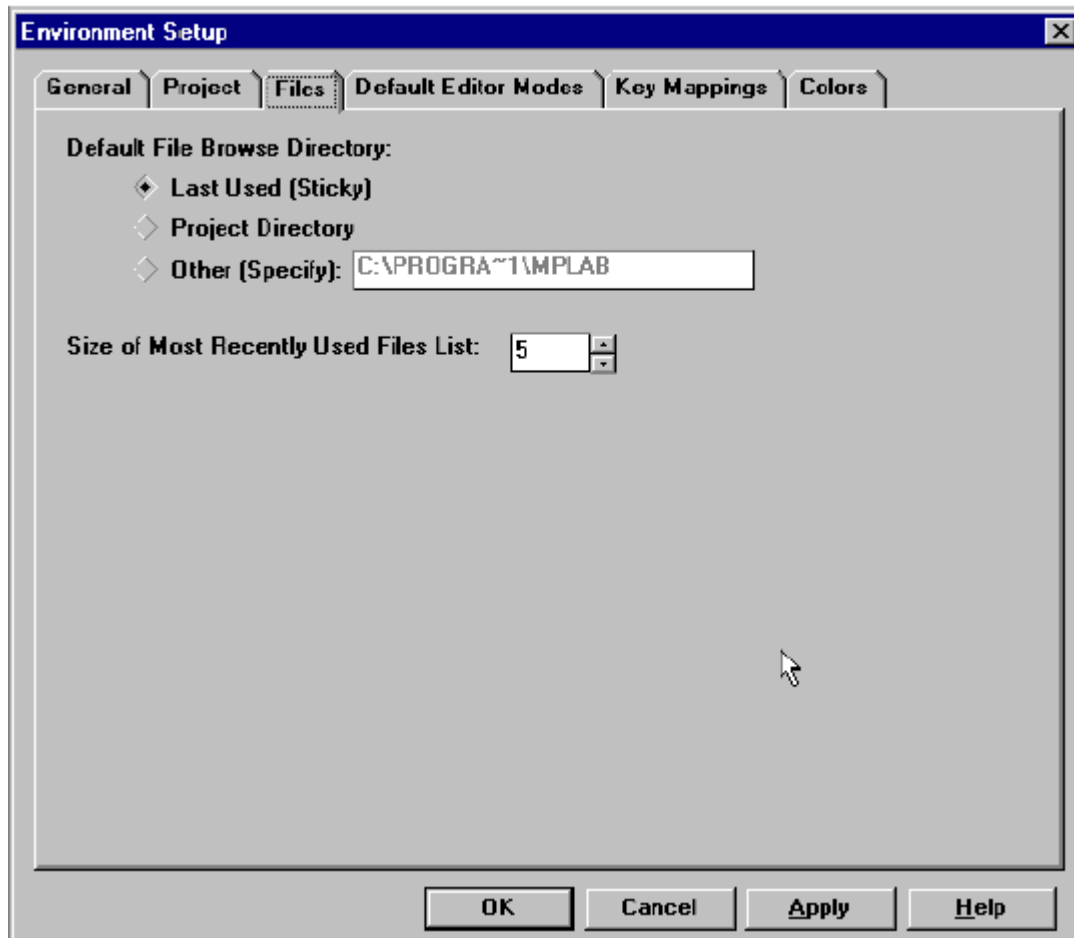


图 7-53: 文件选择项会话窗口

#### 7. 8. 5. 3. 1 默认文件浏览目录

MPLAB IDE 从“默认文件浏览目录”(Default File Browse Directory)开启一个文件浏览窗口。

要想从一个文件被加入的最后那个位置开始，选择“Last Used”。如果目录已经不存在了，MPLAB IDE 将会使用 MPLAB IDE 目录。

要想从“项目”目录的根目录开始，可以选择“Project Directory”(项目目录)。如果目录已经不存在了，MPLAB IDE 将会使用当前存在的最高的“父”目录。假如没有“项目”被打开，MPLAB IDE 将会使用 MPLAB IDE 目录。

想要指定一个开始浏览的目录，可以选择“Other”，然后输入相应得目录。这个设置的默认目录为“Project Directory”(“项目”目录)。

#### 7. 8. 5. 3. 2 “最近使用过的文件列表”大小

- “最近使用过的文件列表”大小，在 MPLAB IDE 文件菜单的底部，允许用户设置一个最近使用过得文件列表大小。当你减少这个数目，文件名将会从列表里面被移走。当你以后又增加这个数值得时候，这些文件名将不会再被 MPLAB IDE “记住”(显示出来)。当你增加这个数值以后，以后所使用过的文件将会被记忆在系统里。默认的文件数目是 5 个。

#### 7. 8. 5. 4 默认编辑器模式

MPLAB 编辑器为每一个新文件和旧文件的编辑窗口设置了一系列的 WINDOWS 模式。这个模式的设置可以通过使用菜单命令：“**Options > Environment Setup**”来实现。假如用户想保存当前在“环境设置”(Environment Settings)会话窗口里的所有设置，只要按“**Apply**”按钮即可。

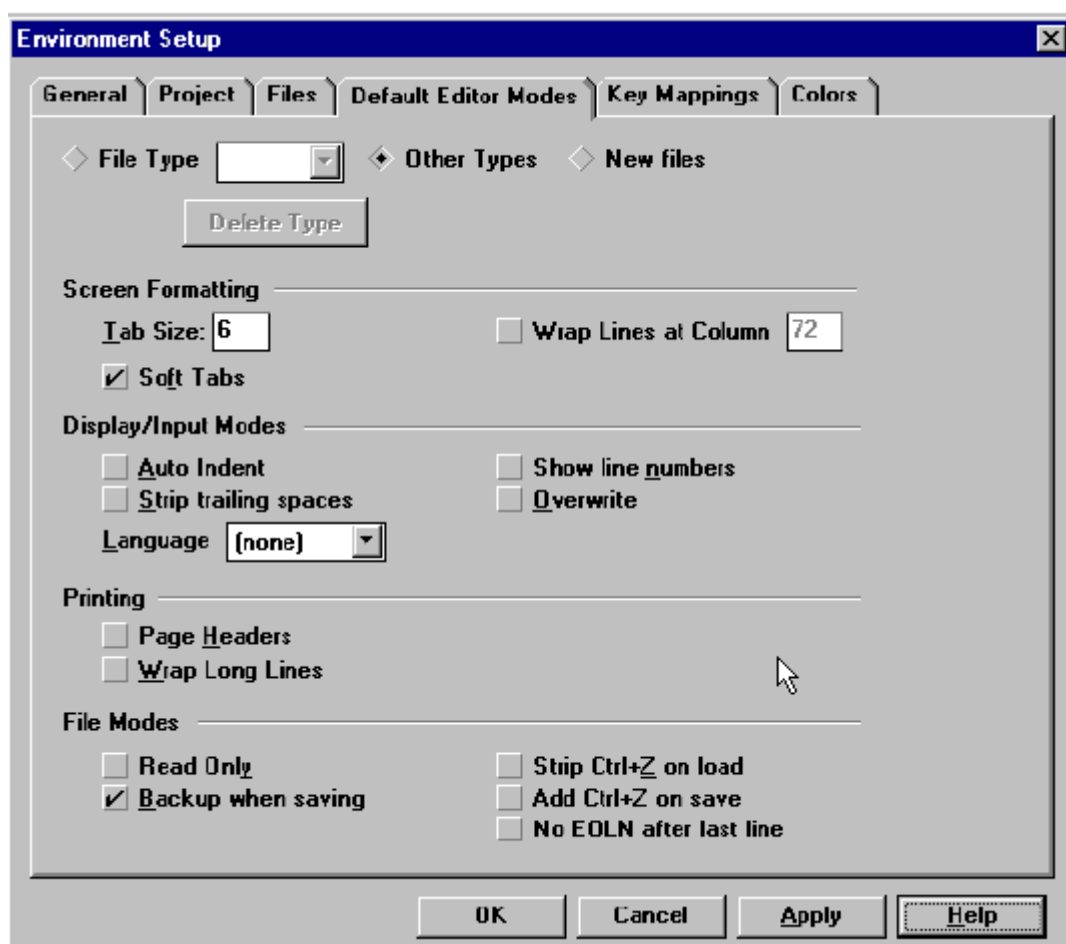


图 7-54: 编辑模式会话窗口

要想给一个指定类型的文件设置默认编辑器模式，点击“**File Type**”按钮，然后可以从相应的列表里面选择文件类型，或者自己在编辑控制里面输入一种新的类型(要以“.”开头)。使用菜单命令：“**File > Open Source**”，“**File > View**”和命令：“**File > Save As**”可以完成这种模式的设置。

1. 为用“**New File**”命令建立的文件设置默认编辑器模式的时候，选择“**New Files**”。或者点击“**Other Types**”按钮来为在文件类型(File Type)列表里没有的文件设置默认模式。
2. 检查会话窗口里的模式窗口，给出屏幕模式、显示及输入、打印和文件保存模式。
3. 点击“**Apply Settings**”来保存关于这个文件的设置。

## 4. 重复步骤 1 到步骤 3，为其他类型的文件设置编辑器模式。

可以在会话窗口里设置的模式有：

<b>屏幕格式</b> 这些设置影响到 TAB 键和文本约束(text wrapping)	
<b>TAB 键大小</b>	定义一个 TAB 字符的宽度
<b>软 TAB</b>	检查到是软 TAB 的时候插入空格而不是一个 TAB
<b>Wrap Lines at Column</b> 行约束列数	在这个选择项的右边的选择盒里作上标记，并给出每行希望约束的列数。假如你不希望有行约束，可以不选中该检查盒。

<b>显示及输入模式</b> 这些设置可以控制字符在窗口里的的显示方式，以及按你的任何输入，MPLAB 编辑器所作的特殊操作	
<b>Auto Indent</b> 自动缩进	将新的行缩进到与前面的行一样的程度
<b>Strip trailing spaces</b> 删除末尾空格	任何时候当你输入回车键 <b>Enter</b> 的时候将行末尾的空白区域字符去处
<b>Show line numbers</b> 显示行号	在窗口里面显示行号
<b>Overwrite</b> 覆盖	在当前光标处，用你键入的字符替换原先有的字符。否则 MPLAB 编辑器将会插入字符到当前光标处。
<b>Language</b> 语言	可以选择一种语言(比如 C 语言或 PASCAL 语言)和文件类型相联系。这种联系意味着编辑格式和你指定的语言的格式对应。

<b>打印(Printing)</b> 这些设置影响到在当前窗口里面 MPLAB 编辑器打印文本的方式。用命令 “ <b>File &gt; Print</b> ” 打印的时候这些设置成为默认设置。但在打印的时候可以忽略。	
<b>Page Headers</b> 页眉	每页里都打印的文件名、页号、日期
<b>Wrap Long Lines</b> 长行约束	打印时候，将太长的行约束，将超出约束长度的文本打印到下一行里

<b>文件模式(File Modes)</b> 这些设置是应用于文件的文件模式	
<b>Read only</b> 只读	禁止用户对这种类型的文件存盘
<b>Backup when saving</b> 存盘的时候备份	当用户保存文件到磁盘的时候给该文件创建一个相同的拷贝
<b>Strip Ctrl+Z on load</b> 调入的时候去除 Ctrl+Z	当调入文件的时候自动去除文件末尾的 Ctrl+Z 符号
<b>Add Ctrl+Z on save</b> 存盘的时候加上 Ctrl+Z	当保存文件到磁盘的时候自动在文件末尾加上 Ctrl+Z 符号
<b>No EOLN after last line</b> 最后一行没有 EOLN 标记	当保存文件到磁盘的时候自动在文件最后一行的最后一个字符后面行结束符加上当保存文件到磁盘的时候自动在文件末尾加上行结束符号 (CR-LF 或 LF)

### 7. 8. 5. 5 快捷键对应关系 Key Mappings (Key Mappings)

选择命令：“**Options > Environment Setup**”然后点击“**Key Mappings**”标签，可以进入显示和编辑 MPLAB IDE 的快捷键对应关系。通过将键盘的键和 MPLAB IDE 的某一功能对应，你可以很快速的执行操作命令。里可能会使用当前有的快捷键对应关系，也可以自己编辑快捷键对应关系来满足自己当前“项目”的需要。MPLAB IDE 使用二进制初始化文件“MPLAB.KEY”，里面存储了快捷键对应关系，可以在其他的任务里面使用这样的设置。假如用户想保存当前在“环境设置” (Environment Settings) 会话窗口里的所有设置，只要按“**Apply**”按钮即可。

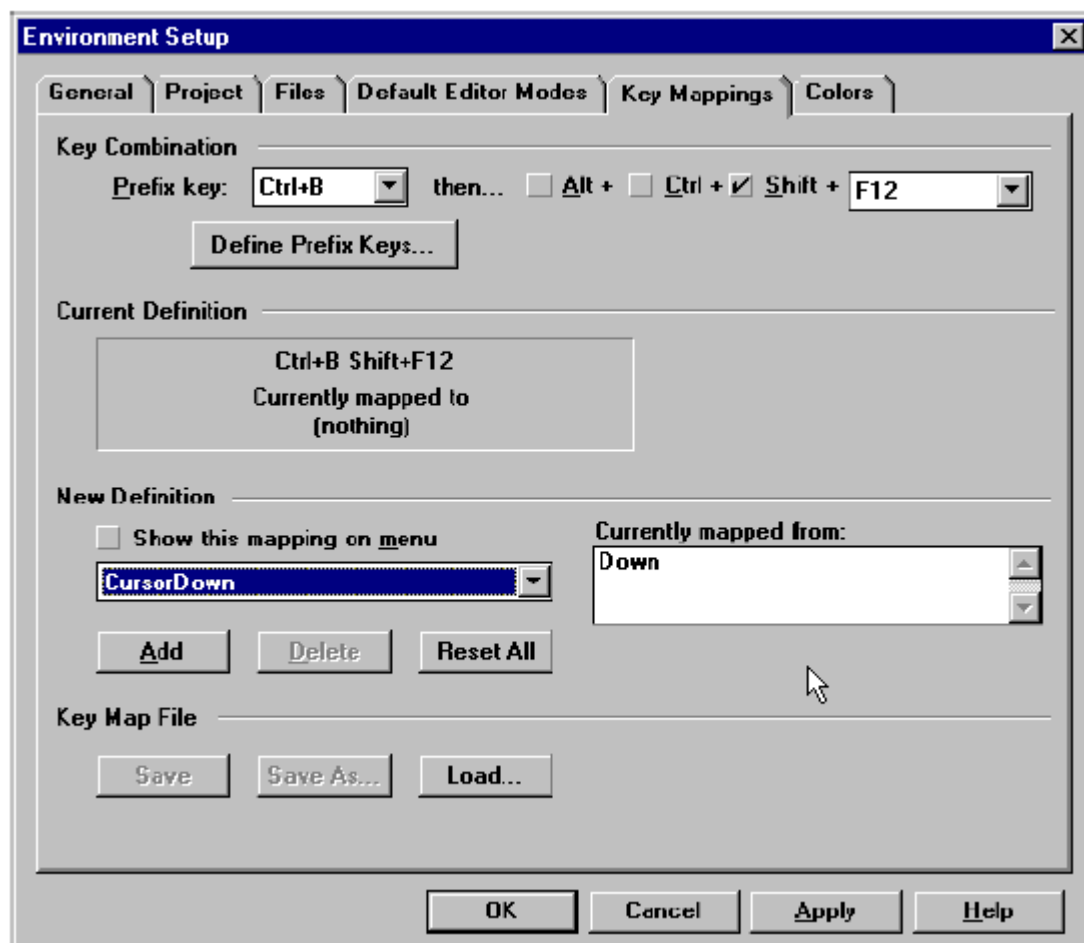


图 7-55: 快捷键对应关系会话窗口

默认情况下，不允许组合键。这意味着快捷键功能对应只有单键对应。

MPLAB IDE 使用二进制初始化文件“MPLAB.KEY”，里面存储了快捷键对应关系，可以在其他的任务里面使用这样的设置

如果想建立一个快捷键功能对应，按以下步骤：

1. 修改当前现成的快捷键功能对应文件，点击“**Load**”可以选择一个已经存在的快捷键功能对应文件。

2. 定义一个组合键(prefix key) (可选)。
3. 点击位于组合键下面的选择盒, 可以定义键或组合键(例如 Shift + F2 等)。当前定义窗口里显示的是当前功能所对应的键或键组合。
4. 假如你想这个快捷键出现在 MPLAB IDE 菜单的选项里的右边, 请在选择盒 “**Show this mapping on menu**” 上选中。
5. 从
6. 点击 “**Add**” 将这一对应关系添加到快捷键对应文件里。  
想改变一个现有的快捷键对应功能, 可以选择这个键或键的组合, 再选择相应得功能, 然后点击 “**Change**” 就可以了。想删除一个快捷键对应关系, 可以定位在这个组合键上, 然后点击 “**Delete**” 即可。
7. 点击 “**Save**” 或者 “**Save As**” 来保存你的快捷键对应表。

#### 7. 8. 5. 6 颜色 (Colors)

选择命令: “**Options > Environment Setup**” 并点击 “**Colors**” 可以进入改变显示数据颜色的界面。假如用户想保存当前在“环境设置” (Environment Settings) 会话窗口里的所有设置, 只要按 “**Apply**” 按钮即可

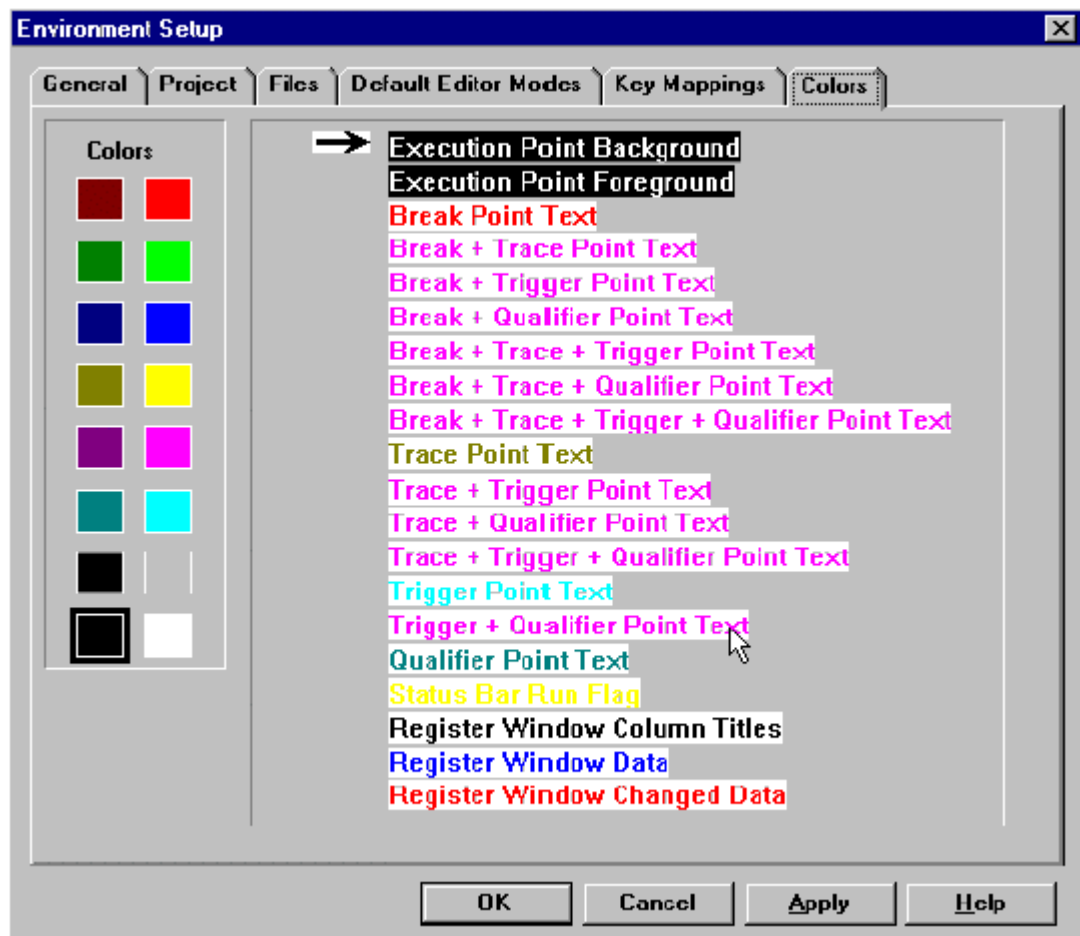


图 7-56: 设置颜色会话窗口

选择你想改变颜色的文本然后上面点击一下, 下一步选择新的颜色。就



可以改变相应的颜色。

#### 7. 8. 6 烧写器选择项

##### 7. 8. 6. 1 选择烧写器

1. 选择命令：“**Options > Programmer Options > Select Programmer**”可以从现有的可选项里面选择一种烧写器类型。
2. 从“**Select Programmer**”下拉菜单里面选择一种烧写器。
3. 点击“**OK**”即可。



图 7-57: 选择烧写器会话窗口

##### 7. 8. 6. 2 通讯口设置

选择命令：“**Options > Programmer Options > Communications Port Setup**”可以进入设置烧写器通讯口的界面。

1. 先选择菜单命令：“**Options > Programmer Options > Communications Port Setup**”。
2. 在通讯口设置会话窗口（**Communications Port Setup**）里面设置串行口。
3. 点击“**OK**”即可。

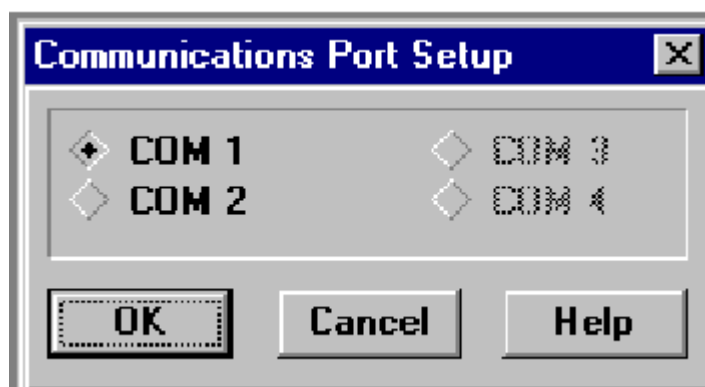


图 7-58: 通讯口设置会话窗口

#### 7. 9. 1 工具菜单 (F11)

菜单命令：“**Tools > DOS Command to Window**”允许用户可以在

WINDOWS 环境下在运行基于 DOS 的程序，比如编译器或内部命令（比如 DIR），还可以把输出捕捉到编辑窗口下面。

该命令启动一个会话窗口，并可以给你显示出命令行的执行情况。这可以是任何 DOS 命令，包括诸如 DIR 之类的 DOS 的内部命令。

点击“**Browse**”按钮，用户可以指定执行命令的工作目录。这个设置只影响 DOS 命令。MPLAB IDE 仍然将会使用以前的工作目录。

当你启动会话窗口，MPLAB IDE 会将命令串和工作目录名称设置为上一次设置的状态。

你同时只能运行一个基于 DOS 的程序。所以必须在一个 DOS 命令执行完以后才可以启动另外一个 DOS 程序。

注意：用户无法用这种方式打开基于 WINDOWS 的应用程序。

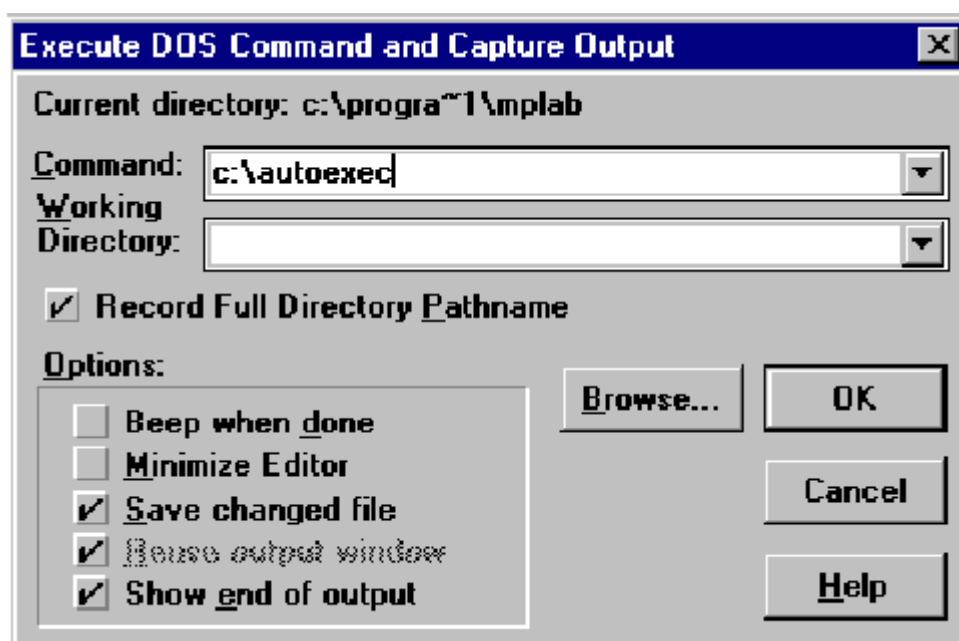


图 7-59：执行 DOS 命令会话窗口

1. 设置你想应用的各种选择项：

**Beep when done:** 可以在编辑器执行完 DOS 应用程序以后给出一声标准的系统“笛”声。

**Minimize Editor:** 在执行基于 DOS 的应用程序之前可以使编辑器最小化为一个图标。

**Save changed files:** 当执行基于 DOS 的应用程序的时候之前，编辑器会察看是否你在编辑的文件被改变过并给你相应的提示。假如你确认可以保存改变的信息，所有你做的修改都会被存入磁盘。

**Reuse output window:** 会导致编辑器将 DOS 应用程序输出到 WINDOWS 使用的你上次用过的会话窗口。假如没有设置，那么就会生成一个新的窗口。

**Show end of output:** 导致编辑器自动将窗口滚动到最后，让你看见输

出结果的最后的内容而不是开始的内容。

2. 点击“OK”运行 DOS 应用程序。

命令行, 工作目录和选择项都保存起来成为你下一次启动会话窗口时候的默认设置。

### 7. 9. 2 在 WINDOW 下重复 DOS 命令 (Ctrl+F11):

执行菜单命令: “**Tools > Repeat DOS Command to Window**” 可以重复上一次“在 WINDOW 下执行 DOS 命令”(Execute DOS Command To Window) 状态下执行的 DOS 命令。命令执行结束后会在窗口里显示执行结果。

假如你以前没有执行命令, 编辑器看起来好象执行了“在 WINDOW 下执行 DOS 命令”(Execute DOS Command To Window), 并将会打开“执行 DOS 命令并捕捉输出”(Execute DOS Command and Capture Output)。

### 7. 9. 3 检测 PICMASTER 仿真器

假如你在 MPLAB IDE 下使用 PICMASTER 仿真器, 选择菜单命令: “**Tools > Verify PICMASTER**” 来检测 PICMASTER 是否工作正常。参考《PICMASTER 用户指南》获取更多关于如何检测 PICMASTER 仿真器的信息。

### 7. 9. 4 检测 MPLAB-ICE

假如你在 MPLAB IDE 下使用 MPLAB-ICE 仿真器, 选择菜单命令: “**Tools > Verify MPLAB-ICE**” 来检测 MPLAB-ICE 是否工作正常。参考《MPLAB-ICE 用户指南》获取更多关于如何检测 MPLAB-ICE 仿真器的信息。

## 7. 10 窗口菜单

无论是模拟器模式还是仿真器模式, 所有的窗口选择项都可以选择。

在纯编辑器模式下, “绝对列表”(Absolute Listing) 和“显示符号列表”(Show Symbol List) 都可以使用。另外, 所有的窗口位置选择项和“打开窗口”“Open Windows”选择项都可以使用。

可以使用的窗口包括以下这些:

- Program Memory – 程序存储器
- Trace Memory – 跟踪存储器
- EEPROM Memory (device dependent) – EEPROM 存储器
- Calibration Data (device dependent) – 校正数据
- Absolute Listing – 绝对列表
- Map File – 映射文件
- Stack – 堆栈
- File Registers – 文件寄存器
- Special Function Registers – 特殊功能寄存器
- Show Symbol List – 显示符号列表
- Stopwatch – 跑表
- Project - 项目
- Watch Window - 观察窗口
- Modify – 修改

下列命令影响到 MPLAB IDE 里的窗口排列方式和外观特性。

- Tile Horizontal – 垂直排列
- Tile Vertical – 水平排列
- Cascade – 重叠
- Iconize All – 使所有图标化
- Arrange Icons – 排列图标
- (Open Windows) – 打开窗口

### 7. 10. 1 程序存储器

选择命令：“**Window > Program Memory**”可以显示程序存储器的内容。程序存储器显示的内容与用户选择的处理器所规定的范围一致。任何时候你都可以开着程序存储器窗口，可以移动它或者最小化。

程序存储器窗口只能在仿真器和模拟器模式下才是可以使用的。

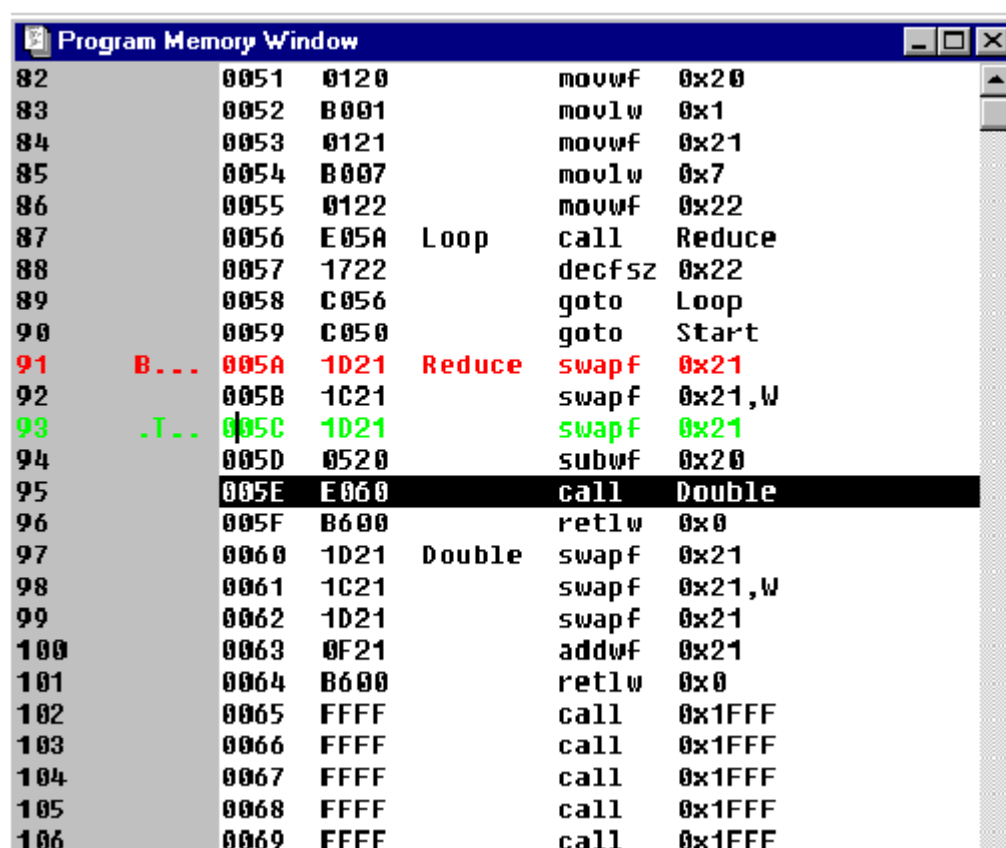


图 7-60: 程序存储器窗口 - 显示机器代码

#### 7. 10. 1. 1 系统按钮选择项

在程序存储器（Program Memory）窗口的左上方点击系统按钮，可以看见下列选择项：

- Toggle Line Numbers: 可以切换决定是显示还是不显示行号和鉴别符。
- Hex Code Display: 将程序存储器里的代码用十六进制显示出来。
- Machine Code Display: 显示不带符号的反汇编 HEX 代码。

- **Disassembly Display:** 显示带符号的反汇编 HEX 代码。

#### 7. 10. 1. 2 程序存储器显示模式

程序存储器内容可以有三种显示方式。可以从系统菜单里选择需要的显示格式。

- **Hex Code Display** – 这种方式显示的是程序存储器里内容的 HEX 数据。当使用芯片烧写器的时候这个选择非常有用（见图 7-61）。
- **Machine Code Display** – 这种方式显示不带符号的反汇编 HEX 代码。（见图 7-60）
- **Disassembly Display** – 这种方式显示带符号的反汇编 HEX 代码。当程序存储器窗口的显示方式是“机器代码显示模式”（Machine Code Display）的时候，当前 PC 指针地址处的指令将会高亮。其他 MPLAB IDE 的特性可以改变程序存储器窗口的显示。

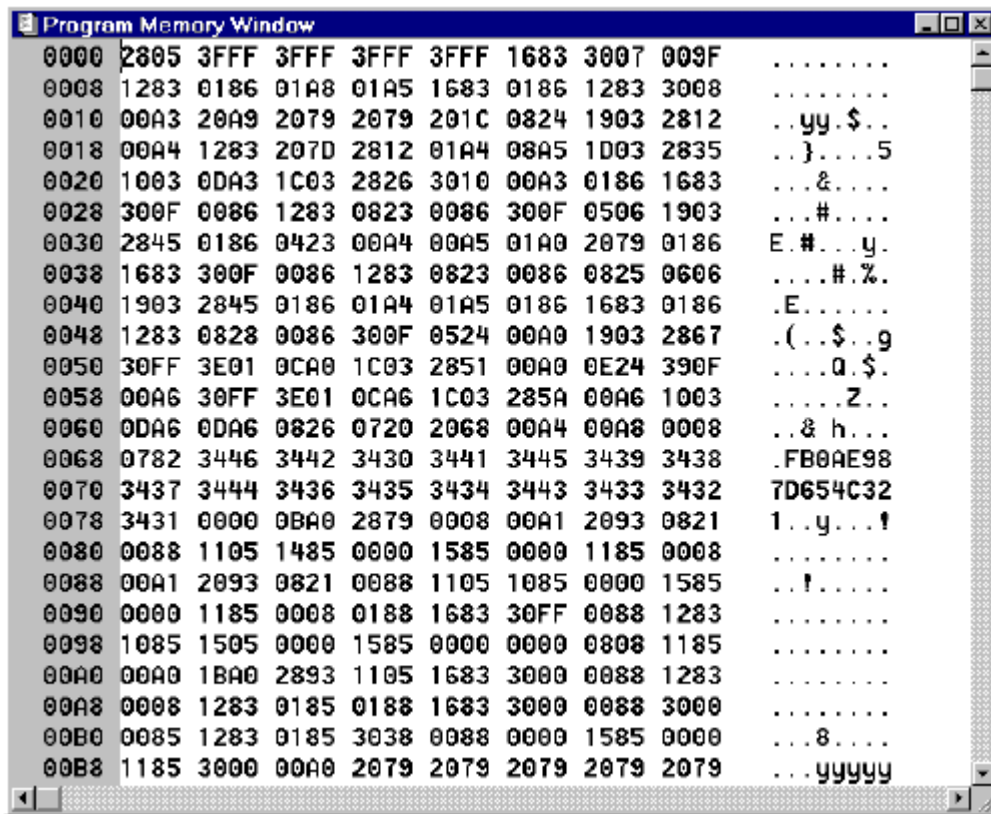


图 7-61: 程序存储器窗口 – 显示 HEX 代码

#### 7. 10. 1. 3 程序存储器窗口区域的描述

MPLAB IDE 直接从仿真器的存储器里读取代码并显示在程序存储器窗口里面。程序存储器窗口区域包含下列信息：

- 区域 1: 十六进制地址
- 区域 2: 十六进制操作码（或数据）
- 区域 3: 符号格式的程序标号。你可以增加标号的显示宽度，使用命令：“***Options > Environment Setup***”然后点击“OK”。
- 区域 4: 机器代码，反汇编代码或者源代码。

- 高亮栏：当前程序指针 PC 的位置。

程序存储器在每个存储器位置显示断点，跟踪点，触发输出状态：

#### 7. 10. 1. 3. 1 选择点 (Selecting Points)

符号	点类型	菜单选项 (RMB=鼠标右键)
B	断点	RMB > Break Point(s)
T	跟踪点	RMB > Trace Point(s)
O	触发输出	RMB > Trigger Point(s)
Q	通过计数器地址	在断点或跟踪点设置会话窗口里设置

可供使用的选择项取决于所选择的开发工具。

在程序存储器窗口里，MPLAB IDE 使用不同的颜色和符号来标识断点，跟踪点，触发点等。假如在某一地址没有设置任何“点”，这里的文本按正常方式显示。假如设置了一种“点”，这一行的颜色就会改变，同时行号的宽度也会增加，以便显示被激活的“点”。重新设置的点则按现在的方式显示。

#### 7. 10. 1. 4 建立一个临时的实时断点

如果要在程序存储器窗口里建立一个临时的实时断点，可以在任何合法的地址行上双击鼠标的左键。直到遇到下列状况出现，处理器会一直实时运行。

- 执行到的行设置了临时断点
- 遇到断点
- 用户自己点击了“Halt”

#### 7. 10. 2 跟踪存储器

选择命令：“**Window > Trace Memory**”，可以显示跟踪缓冲器里的内容。这个窗口可以在任何时候打开，移动，重新设置窗口大小。

跟踪存储器采用一种叫做“snapshot”的方法采样你程序的执行，对于有跟踪缓冲器的仿真器，这可以让你看到程序是如何全速运行的。

一些应用比如电机控制系统，是无法停止下来的，比如在单步运行的时候不会出现问题，只有程序整步运行的时候才能看到一些 BUG 的存在。跟踪缓冲器给出你另外一种调试工具用来调试这种类型的应用程序。关于在仿真器的硬件跟踪缓冲器里面收集的信息，请参考仿真器使用说明。

在模拟器模式下，跟踪缓冲器也是很有用的，可以收集程序执行时候的信息，以便以后分析。模拟器收集的信息和仿真器的信息差别很小。

如果想使用模拟器的跟踪缓冲器，首先你必须选择跟踪的代码。你可以使用鼠标来“点-拉”并选择一段程序代码进行跟踪，然后点击鼠标的右键。这时候会看见出现一个快捷菜单，从里面你可以选择“Trace Point(s)”（跟踪点）。

现在可以复位并运行程序，运行几秒钟后，停止运行。然后选择命令：“**Window > Trace**”察看收集的跟踪信息。

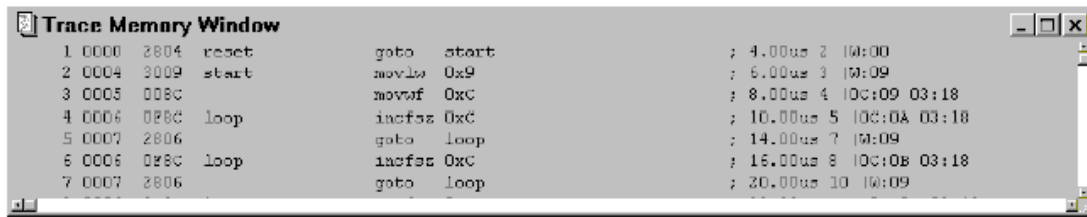


图 7-62: 跟踪存储器窗口

模拟器会在每一行加上时标，并且还会显示任何改变了数值的寄存器。

- 区域 1: 十六进制的地址
- 区域 2: 十六进制的操作数（或数据）
- 区域 3: 符号格式的程序标号。你可以增加标号的显示宽度，使用命令：  
“**Options > Environment Setup**” 然后点击 “**General**” 标签。
- 区域 4: 机器代码，反汇编代码或者源代码。
- 区域 5: 在窗口的最右边显示的是仿真器的“外部逻辑探头状态”（Status on External Logic Probe Lines）信息。

想保存所有的跟踪信息到一个文件里，可以使用菜单命令：“**File > Export > Export Trace Buffer**” 即可。

### 7. 10. 3 EEPROM 存储器

选择命令：“**Window > EEPROM Memory**” 可以显示内部带有 EEPROM 存储器的芯片里的 EEPROM 数据。比如 PIC16F84 就含有内部 EEPROM 存储器。

EEPROM 窗口可以在任何时候打开，移动，重新设置窗口大小。这个窗口只是用来显示的，无法在窗口里面改变 EEPROM 的数值。如果你想改变 EEPROM 的数值，可以使用菜单命令：“**Window > Modify**”，用“修改”会话窗口来实现修改 EEPROM 内容的功能。

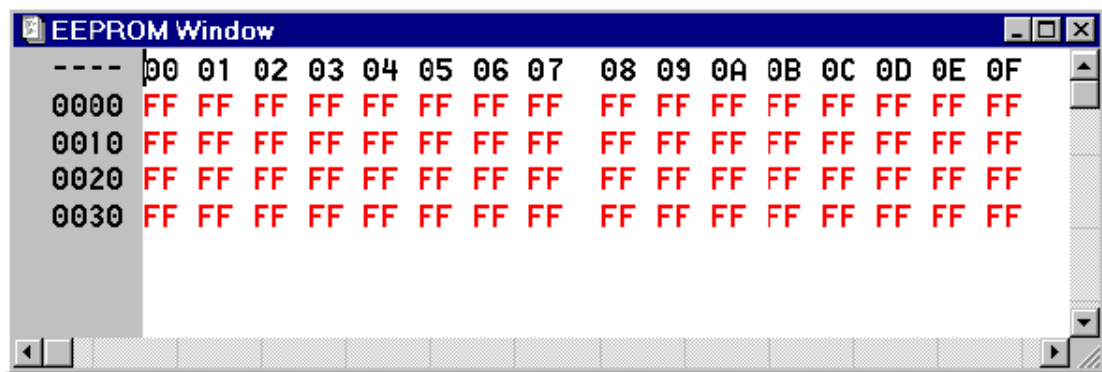


图 7-63: EEPROM 存储器窗口

EEPROM 窗口显示了某一被仿真的处理器里的“数据/操作代码”的十六进制信息。当 EEPROM 寄存器数值改变或者处理器暂停的时候，EEPROM 窗口里的数据会相应的更新。

#### 7. 10. 3. 0. 1 系统按钮选择项

点击位于 EEPROM 存储器窗口的最右上边的系统按钮，可以显示以下

选择项:

- **Toggle Line Numbers:** 切换行号的显示
- **Hex Display:** 以十六进制方式显示程序存储器信息
- **ASCII Display:** 以 ASCII 码的方式显示每个存储器区域里面信息

#### 7. 10. 4 校正数据

假如被仿真的芯片内含有校正存储器，校正存储器的内容可以通过命令：“**Window > Calibration Data.**”观察到。窗口的外形根据仿真器的不同而有差异。

“校正数据”（ Calibration Data）会话窗口是为 PIC12CXXX 或 PIC14000 系列处理器设计的。这里可以显示仿真头（Emulator Probe）里的浮点数据，方便用户修改。

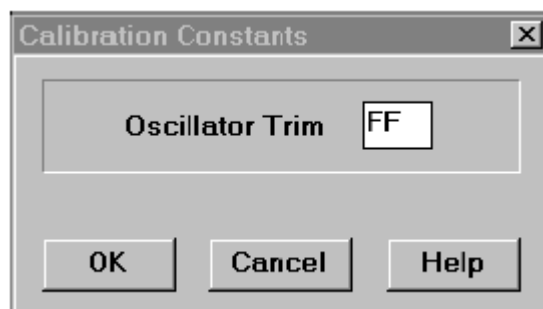


图 7-64: 校正常量会话窗口 - PIC12CXXX

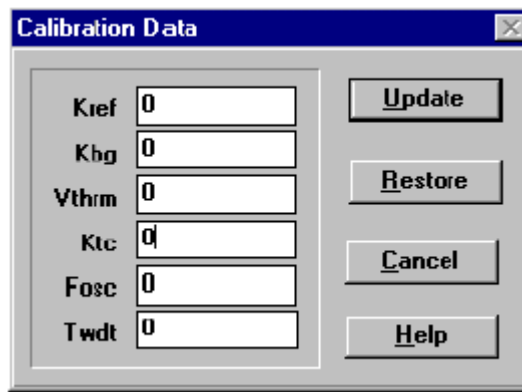


图 7-65: 校正数据会话窗口 - PIC14000

- **Update:** 从“校正数据”会话窗口里得到的信息，从 IEEE745 格式转化为 Microchip 版本的 IEEE 754 格式，并将其存储在存储器区域里作为嵌入代码以供使用。只有前四个数据是浮点数格式，这四个数据分别是：KREF，KBG，VTHRM 和 KTC。最后的两个 FOSC 和 TWDIT 是 8 位无符号数，数值范围是 0 到 255。
- **Restore:** 在系统复位的时候从探头“上载”，得到原始校正数据，并且将存储器区域的数值重新设置，作为嵌入代码使用。
- **Cancel:** 关闭会话窗口，不修改存储器区域的数值。



### 7. 10. 5 绝对列表

选择命令：“**Window > Absolute Listing**”可以显示和单步察看由 MPASM 汇编器或兼容的 C 编译器生成的列表文件 (\*.LST)。

使用这个命令还可以同时察看 C 源代码和与其相对应的汇编代码。绝对列表可以给你一个更好的对比，你可以看见你的 C 程序相应的汇编代码是如何对应的。假如你没有使用 C 语言，汇编器（比如 MPASM 汇编器）将会产生一个相应的列表文件。

“绝对列表”窗口显示了由汇编器或编译器产生的列表文件。“绝对列表”显示源代码和生成的目标代码。

```

c:\progra~1\mplab\tutor.lst
00070
00071 ;-----
00072 ; Main Program
00073 ;-----
00074
00075 ORG    H'50'
00076
00077 Start
00078      MOVLW 255      ; Initialize the variables to
00079      MOVWF CountDown ; their starting values.
00080      MOVLW 1
00081      MOVWF Doubler
00082      MOVLW 7
00083      MOVWF OuterLoop
00084 Loop
00085      CALL Reduce      ; Perform the inner portion of
00086      DECFSZ OuterLoop,f ; the loop.
00087      GOTO  Loop
00088
00089      GOTO  Start      ; Repeat the whole thing.
00090
00091 ;-----
00092 Reduce
00093      SWAPF Doubler,f      ; Reduce CountDown by the
00094      SWAPF Doubler,w      ; value of Doubler. Then
00095      SWAPF Doubler,f      ; call the doubling routine.
00096      SUBWF CountDown,f
00097      CALL Double
00098      RETLW 0
00099
  
```

图 7-66: 绝对列表窗口

### 7. 10. 6 映射文件

映射文件反映了链接以后使用过的和没有使用的存储器空间。要想看这个映射表，可以使用命令：“**Window > Map File**”进行查看。

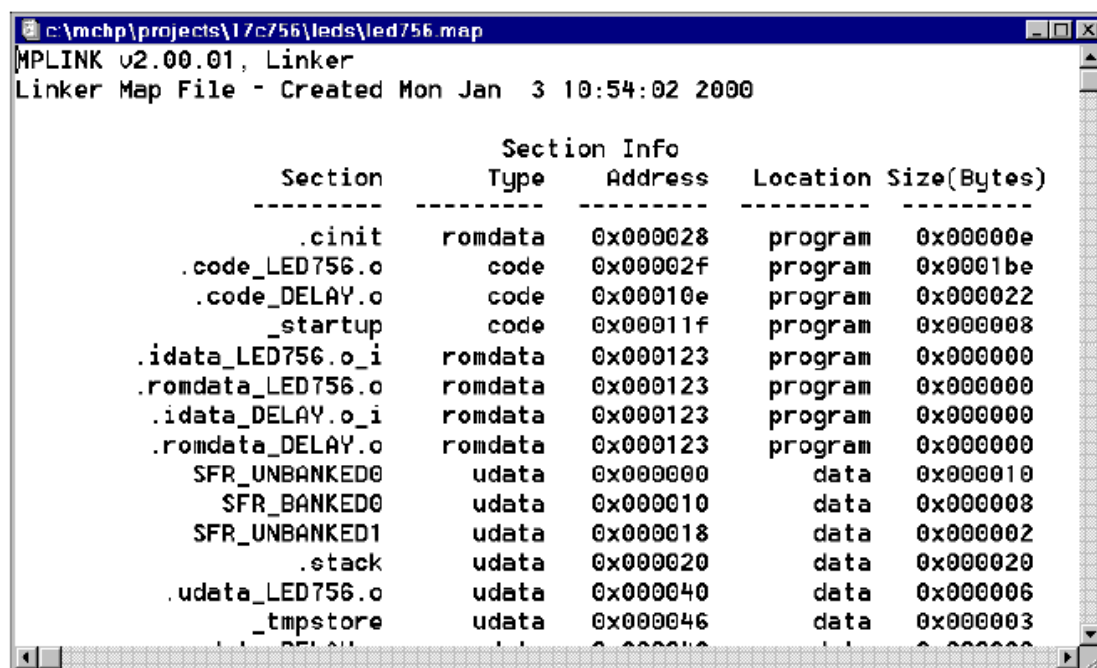


图 7-67: 映射文件窗口

要想生成一个已经链接项目的映射文件，可以使用命令：“**Project > Edit Project**”，进入“编辑项目”（Edit Project）会话窗口，然后在会话窗口的“项目文件”（Project Files）里选择（高亮）主节点[.hex]。点击“**Node Properties**”（节点属性），进入“节点属性”会话窗口。找到映射文件选择盒，点击并打开选择项即可。

关于映射文件的更多信息，请参考《MPASM 及 MPLINK, MPLIB 用户指南》。

#### 7. 10. 7 堆栈

选择命令：“**Window > Stack**”可以打开一个显示堆栈内容的窗口。堆栈的深度取决于所仿真的处理器类型。“堆栈”窗口可以随时被打开，移动，重新设置窗口的大小。

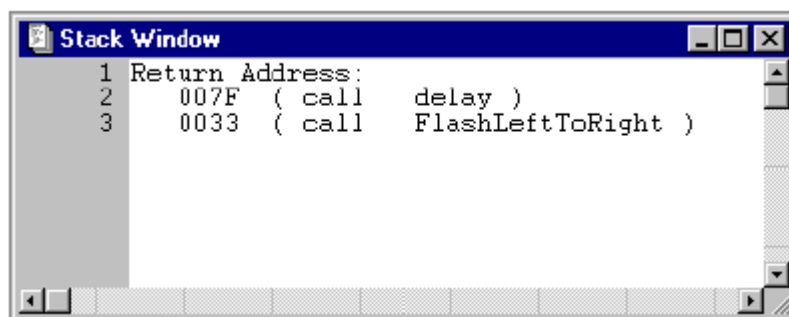


图 7-68: 堆栈窗口

注意：假如设置了堆栈溢出允许（**Debug>Break Settings**），当堆栈溢出的时候 MPLAB IDE 将显示堆栈溢出和 underflow 警告。

堆栈内容的显示可能有（显示）或没有行号。用户可以在系统菜单里选择喜欢的格式。

注意：要想操作系统菜单，可以点击程序存储器窗口左上角。

#### 7. 10. 7. 1 硬件堆栈深度

##### 7. 10. 7. 1. 1 12 位核芯片的硬件堆栈深度 – 2 级

有 12 位核的芯片，比如 PIC12CXXX 和 PIC16C5X 系列的芯片有两级硬件堆栈。

##### 7. 10. 7. 1. 2 14 位核芯片的硬件堆栈深度 – 8 级

有 14 位核的芯片，比如 PIC14000 和 PIC16CXX 系列的芯片有八级硬件堆栈。

注意：假如你清除了“禁止堆栈溢出警告”（Disable Stack Overflow Warning）（**Options > Develop-ment Mode**, Bread Options tab），程序里又对堆栈推入了过多的数据，MPLAB IDE 将显示一个 underflow 或 overflow 信息。

##### 7. 10. 7. 1. 3 16 位核芯片的硬件堆栈深度 – 16 级

有 16 位核的芯片，比如 PIC17CXXX 系列系列的芯片有十六级硬件堆栈深度。

注意：假如你清除了“禁止堆栈溢出警告”（Disable Stack Overflow Warning）（**Options > Develop-ment Mode**, Bread Options tab），程序里又对堆栈推入了过多的数据，MPLAB IDE 将显示一个 underflow 或 overflow 信息。

#### 7. 10. 7. 2 模拟器堆栈 - 12 位核芯片

MPLAB IDE 可以精确模拟 PIC12CXX 和 PIC16C5X 系列处理器的硬件堆栈，而且当堆栈 underflow 或者 overflow 的时候会给出警告信息。当遇到程序里的一个 CALL 指令的时候，或者出现了中断，这时候 PC+1 后被推进堆栈，当执行 RETLW 指令的时候又会弹出堆栈。假如在堆栈弹出之前压进了多于 2 个数据的时候，数据仍然会压进堆栈，但是会看到系统给出了堆栈溢出警告报告。假如用户想弹出一个空的堆栈，系统会产生一个错误信息。弹出一个空的堆栈会导致最后一个数据被弹出并放入程序指针（PC）里。

#### 7. 10. 7. 3 模拟器堆栈 - 14 位核芯片

MPLAB IDE 可以精确模拟 PIC14000 和 PIC16CXX 系列处理器的硬件堆栈，而且当堆栈 underflow 或者 overflow 的时候会给出警告信息。当遇到程序里的一个 CALL 指令的时候，或者出现了中断，这时候 PC+1 后被推进堆栈，当执行 RETLW, RETURN 或 RETFIE 指令的时候又会弹出堆栈。假如在堆栈弹出之前压进了多于八个数据的时候，数据仍然会压进堆栈，但是会看到系统给出了堆栈溢出警告报告。假如用户想弹出一个空的堆栈，系统会产生一个错误信息。弹出一个空的堆栈会导致堆栈指针指向堆栈的顶部（堆

栈满)，假如又执行了另外一个 POP 指令，将不产生错误报告。

#### 7. 10. 7. 4 模拟器堆栈 - 16 位核芯片

MPLAB IDE 可以精确模拟 PIC17CXXX 系列处理器的硬件堆栈，而且当堆栈 underflow 或者 overflow 的时候会给出警告信息。当遇到程序里的一个 CALL 或者 LCALL 指令的时候，或者出现了中断，这时候 PC+1 后被推进堆栈，当执行 RETLW，RETURN 或者 RETFIE 指令的时候又会弹出堆栈。假如在堆栈弹出之前压进了多于十六个数据的时候，数据仍然会压进堆栈，但是会看到系统给出了堆栈溢出警告报告，并且 STAKAVL 位被清除，直到出现复位信号。

#### 7. 10. 7. 5 模拟器堆栈 - 扩展 16 位核芯片

MPLAB IDE 可以精确模拟 PIC18CXXX 系列处理器的硬件堆栈，而且当堆栈 underflow 或者 overflow 的时候会给出警告信息。当遇到程序里的一个 CALL 或者 LCALL 指令的时候，或者出现了中断，这时候 CALL 指令的后面一条指令的地址被推进堆栈，当执行 RETLW，RETURN 或者 RETFIE 指令的时候又会弹出堆栈。假如在堆栈弹出之前压进了多于三十一个数据的时候，数据仍然会压进堆栈，但是会看到系统给出了堆栈溢出警告报告，并且 STAKAVL 位被清除，直到出现复位信号。

#### 7. 10. 8 文件寄存器 (File Registers)

选择命令：“**Window > File Registers**”可以显示被仿真的芯片内部所有的文件寄存器。当一个文件寄存器的数值改变的时候，或者处理器暂停下来以后，“文件寄存器”窗口将会更新。“文件寄存器”窗口可以随时被打开，移动，重新设置窗口的大小。

File Register Window

0000	HEX	DEC	BINARY	CHAR	SYMBOL NAME
0001	FF	255	11111111	.	TRMT
0002	00	0	00000000	.	I2C_READ
0003	1B	27	00011011	.	STATUS
0004	7B	123	01111011	{	CCP1Y
0005	10	16	00010000	.	T0CS
0006	00	0	00000000	.	RP1
0007	FE	254	11111110	.	PSPIE
0008	81	129	10000001	.	DAT
0009	00	0	00000000	.	PORTE
000A	00	0	00000000	.	PCLATH
000B	00	0	00000000	.	INTCON
000C	00	0	00000000	.	PIR1
000D	00	0	00000000	.	PIR2
000E	00	0	00000000	.	TMR1L
000F	00	0	00000000	.	TMR1H
0010	00	0	00000000	.	T1CON
0011	00	0	00000000	.	TMR2
0012	00	0	00000000	.	T2CON

图 7-69: 文件寄存器窗口 – 符号化的显示

文件寄存器的内容可以通过窗口来修改。如果想改变一些寄存器的内容（比如用某一数值填充某一区域的内容），你可以使用鼠标左键来“点-拉”并选择一段想改变数值的寄存器（假如只想改变一个寄存器的数值，简单地将光标放在你想改变数值的寄存器上），然后点击鼠标的右键。这时候会看见出现一个“填充寄存器”菜单，从里面你可以选择“**Fill Register**”（填充寄存器），这时候会出现“**Modify**”（修改）会话窗口（见图7-79所示），可以看到寄存器的地址范围已经自动输入。

#### 7. 10. 8. 1 文件寄存器窗口显示模式

文件寄存器的显示方式可以有三种。用户可以通过系统菜单选择所希望的格式。

- **Hex Display**（十六进制显示）：这个选择项可以使文件寄存器以1十六进制方式显示。（见图7-70所示）
- **Symbolic Display**（符号化显示）：可以符号化显示每个文件寄存器内容对应的十六进制格式，十进制格式，二进制格式和字符格式（见图7-69所示）。
- **ASCII Display**（ASCII符显示）：以ASCII文本格式显示文件寄存器的内容。

**注意 1：** 可以点击程序存储器窗口的左上角，进入系统菜单。

**注意 2：** 假如在“**Options > Development Mode**”里的“**Break Options**”标签（断点选择项）下选择了“**Freeze Peripherals On Halt**”（暂停时冻结外设），当单步执行的时候SFR里的IO口或观察窗口里的寄存器内容都不会更新。管脚的状态会被修改，可是“读”请求因为冻结而被阻断，直到下一步或执行“**RUN**”（运行）模式，其内容不会被更新。

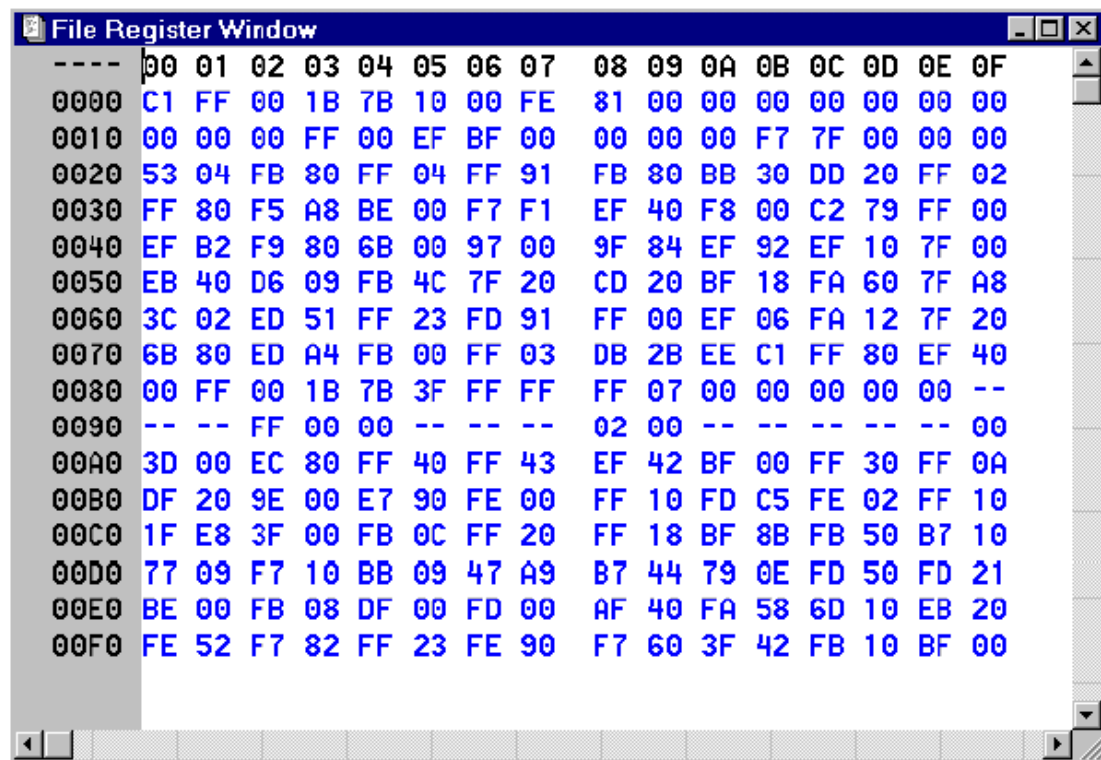


图7-70: 文件寄存器窗口 – 十六进制显示

#### 7. 10. 8. 2 系统按钮选择项

可以点击“文件寄存器存储器”（File Register Memory）窗口的左上角，进入系统菜单并可以显示下列选择项：

- **Toggle Line Numbers:** 切换行号的显示
- **Hex Display:** 以十六进制方式显示文件寄存器信息
- **Symbolic Display**（符号化显示）：可以符号化显示每个存储器内容对应的十六进制格式，十进制格式，二进制格式，ASCII字符格式，符号（Symbol）和名称（Name）。
- **ASCII Display（ASCII符显示）**：以ASCII文本格式显示存储器的内容。

#### 7. 10. 9 特殊功能寄存器

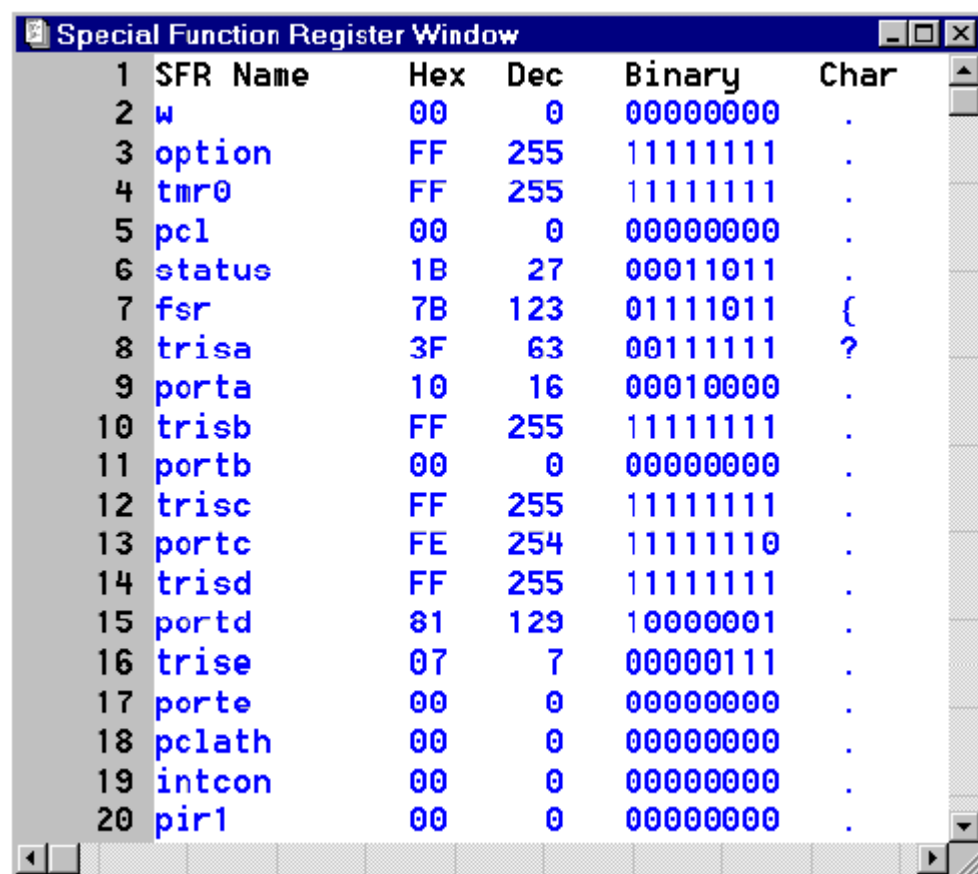
选择命令：“**Window > Special Function Registers**”可以显示被仿真的处理器内部特殊功能寄存器（SFR）里的内容。这个窗口提供的显示格式比其他普通文件寄存器窗口要更容易观察 SFR。因为这里显示出了每个特殊功能寄存器（SFR）的名称和好几种数值显示格式。任何时候出现断点中断以后，SFR 就会被更新。

特殊功能寄存器窗口可以随时被打开，移动，重新设置窗口的大小。

特殊功能寄存器（SFR）显示的时候可以带行号或不带行号。可以在系统菜单里选择希望使用的格式。

**注意 1:** 可以点击程序存储器窗口的左上角，进入系统菜单。

**注意 2:** 假如在 “**Options > Development Mode**” 里的 “Break Options” 标签（断点选择项）下选择了 “Freeze Peripherals On Halt”（暂停时冻结外设），当单步执行的时候 SFR 里的 IO 口或观察窗口里的寄存器内容都不会更新。管脚的状态会被修改，可是“读”请求因为冻结而被阻断，直到下一步或执行“RUN”（运行）模式，其内容不会被更新。



SFR Name	Hex	Dec	Binary	Char
w	00	0	00000000	.
option	FF	255	11111111	.
tmr0	FF	255	11111111	.
pcl	00	0	00000000	.
status	1B	27	00011011	.
fsr	7B	123	01111011	{
trisa	3F	63	00111111	?
porta	10	16	00010000	.
trisb	FF	255	11111111	.
portb	00	0	00000000	.
trisc	FF	255	11111111	.
portc	FE	254	11111110	.
trisd	FF	255	11111111	.
portd	81	129	10000001	.
trise	07	7	00000111	.
porte	00	0	00000000	.
pclath	00	0	00000000	.
intcon	00	0	00000000	.
pir1	00	0	00000000	.

图 7-71: 特殊功能寄存器

区域 1: SFR 的名称

区域 2: 数据的十六进制值

区域 3: 数据的十进制值

区域 4: 数据的二进制值

区域 5: 数据的 ASCII 值

如果想修改某一个特殊功能寄存器的数值，可以按以下步骤：

1. 在窗口里的一个寄存器上双击鼠标，这时候会出现 “**Modify**”（修改）会话窗口。符号/地址和数据区域都已经填充了数值。也可以：
2. 使用菜单命令：“**Window > Modify**” 进行修改。

**注意:** 对于不同的芯片，特殊功能寄存器（SFR）的名称和数据是不同的。

### 7. 10. 10 显示符号列表 (Ctrl+F8)

选择命令：“**Window > Show Symbol List**”可以显示 MPLAB IDE 的涉及的符号。符号包括常量和标号。“Show Symbol List”（显示符号列表）只是一个信息窗口。在这个窗口里显示的是从你的源程序经过编译或汇编后传入的符号（symbols）。这些符号来自“项目”里的\*.COD 文件。

注意：如果你想查看符号列表，那末“项目”必须打开和创建。

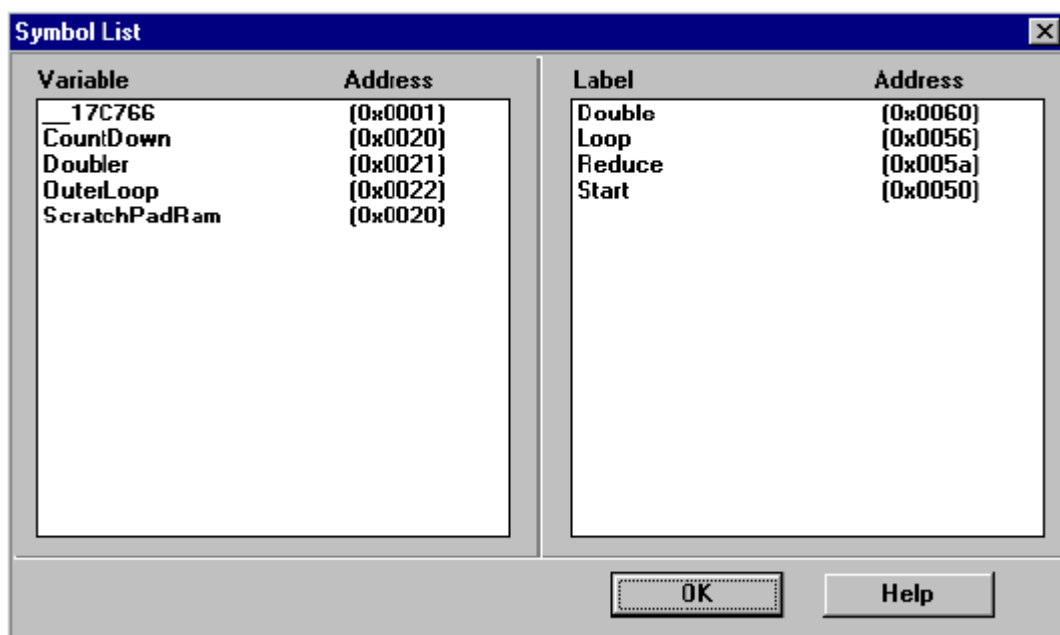


图 7-72: 显示符号列表

- **Variable, Address:** 显示文件寄存器里的变量和每个变量的地址。
- **Label, Address:** 显示程序存储器里的标号和每个标号的地址。
- **Constants:** 在源代码里的常量可以用做操作代码也可以在“**Modify**”（修改）里作为操作数。

### 7. 10. 11 “跑表” (STOPWATCH)

选择命令：“**Window > Stopwatch**”可以显示“Cycle counter”（周期计数器）的当前数值。系统“跑表” (STOPWATCH) 可以记录处理器执行的周期数。The counting occurs with real-time execution and with polled execution. 定时器在指令的每个周期都会触发。“跑表” (STOPWATCH) 允许用户测量代码执行时间。如果是单步执行，测量并经常做到完全精确。“跑表” (STOPWATCH) 根据 PICmicro MCU 的时钟频率测量时间。如果用户想设置时钟频率，可以使用命令：“**Options > Development Mode**”并电击“**Clock**”标签即可。



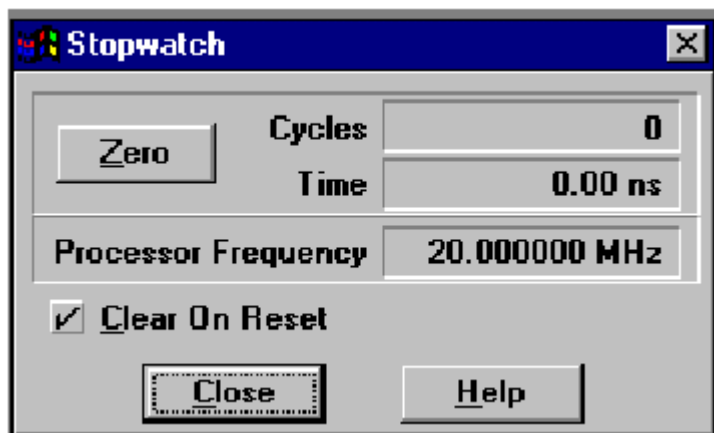


图 7-73: “跑表” (STOPWATCH) 会话窗口

- **Cycles:** 显示处理器执行指令的周期数。
- **Time:** 以秒为单位显示“跑表” (STOPWATCH) 的计时。这个数值是根据芯片时钟频率和统计的周期数计算出来的。
- **Zero:** 点击“Zero”可以将周期计数器复位为零。

**范例 7-1:** 你可以用这个定时器进行实际的时间测量。假如你想计算一个子程序执行的时间，只要在进入子程序之前复位定时器并在子程序的末尾放置一个断点。定时器会显示子程序执行的周期数并显示执行时间。

- **Processor Frequency:** 显示用户选择的处理器时钟频率。如果用户想改变时钟频率，可以选择命令：“***Options > Development Mode***”然后点击“Clock”标签即可。

●

#### 7. 10. 12 “项目”窗口

只有在“项目”被打开的时候才会看见“项目”窗口。这里显示的时当前项目涉及到的文件。假如项目已经被“创建” (compile)，项目窗口里将会显示所有包含的文件。否则项目窗口里只显示主项目文件。双击项目窗口里显示的任何文件可以打开该文件并编辑该文件。

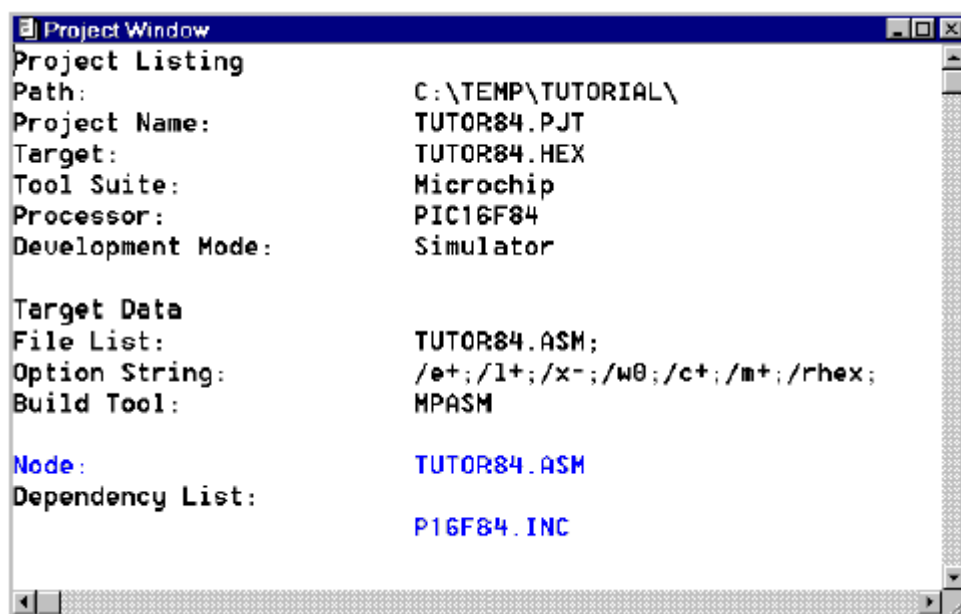


图 7-74: “项目” 窗口

#### 7. 10. 13 “观察” 窗口

MPLAB IDE 允许通过观察窗口(Watch window)来监控文件寄存器的变化情况。可以使用 MPLAB IDE 菜单或者在观察窗口里面，你可以添加或删除寄存器符号，也可以改变它们的显示属性。观察窗口里的显示内容可以有行号也可以没有行号。如果想选择希望的格式，可以在观察窗口里的系统菜单里选择命令：“**Toggle Line Numbers**” (切换行号)。

##### 7. 10. 13. 1 新“观察” 窗口

如果想建立一个“观察”窗口，可以选择命令：“**Window > Watch Window > New Watch Window**”。窗口“Add Watch Symbol”(图 7-76)和窗口“Watch\_1”(图 7-75)都会被打开。参考 7-10-13-4 节的信息，获得关于编辑“观察”窗口的知识。

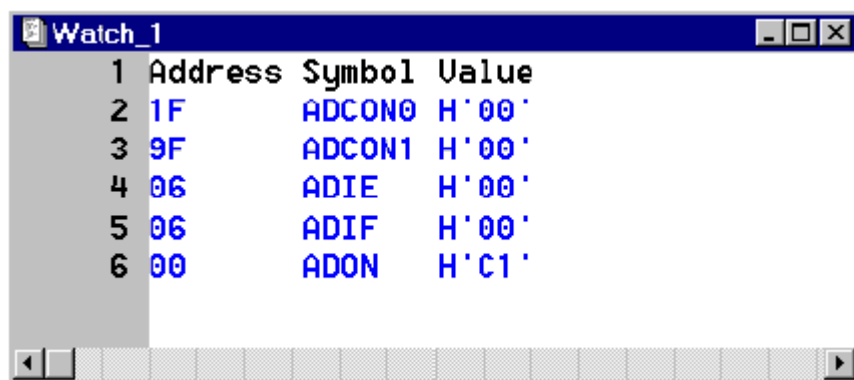


图 7-75: “观察” 窗口

### 7. 10. 13. 2 调入“观察”窗口

选择命令：“**Window > Watch Window > Load Watch Window**”可以调入一个你前一次建立并存盘的“观察”窗口。选择一个“观察”窗口文件并点击“OK”，或者双击该文件即可。

在“Load Watch” (调入“观察”)会话窗口里，在会话窗口右下方的下拉菜单里选择一个驱动器名称。点击在“Load Watch”窗口右上方的“Folders” (文件夹)里的文件夹，在所选择的磁盘上指定路径。在“Load Watch”窗口右边的“File Name”窗口下面的文件列表里面，选择你想打开的“观察”窗口文件。点击“OK”。这时候“Add Watch Symbol” (添加“观察”窗口符号)会话窗口和“Watch”窗口都会被同时打开。

在系统菜单里选择命令：“Add Watch Symbol”，可以添加符号到“观察”窗口里面。参考 7-10-13-4 节获得更多关于编辑“观察”窗口的信息。

### 7. 10. 13. 3 编辑“观察”窗口符号

用户可以添加用户自己正在使用的符号到“观察”窗口里面。

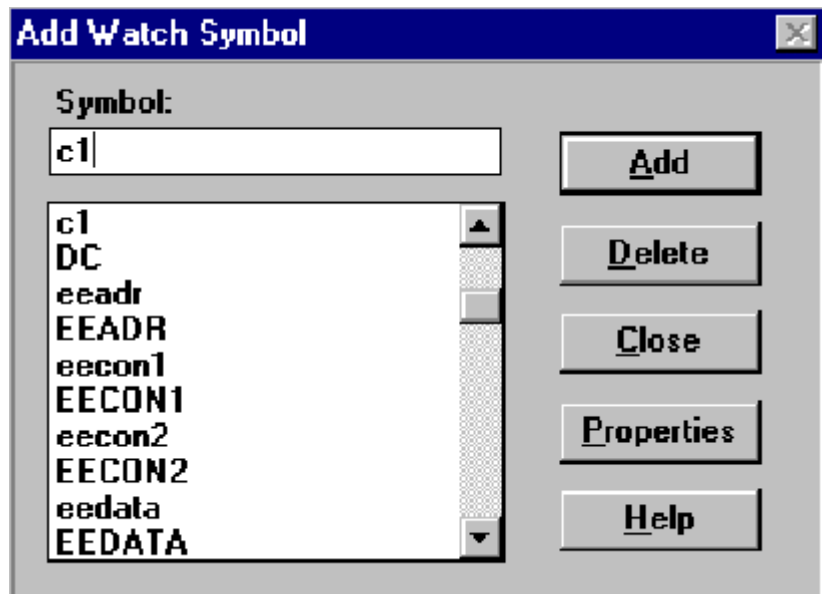


图 7-76: 添加“观察”符号会话窗口

选择命令：“**Window > Watch Window > Add to Active Watch**”，在“Add Watch Symbol”列表窗口里面点击所要添加的符号。或者你可以输入希望“观察”的符号的地址(比如：0x40)。点击“Add”就可以将它们加入到观察窗口里面。

**注意：**变量“w”的地址定义为“0”，同时“w”也表示是“w”寄存器的值。

如果希望改变一个符号在“观察”窗口里显示的方式，可以先选择这个符号，然后点击“Properties” (属性)，打开属性会话窗口(参考第 7-10-13-5 节)。

添加完符号以后点击“OK”。观察窗口将会显示你选择的变量当前值。

#### 7. 10. 13. 4 改变“观察”符号属性

如果用户想改变现在观察窗口里的符号的显示属性或者想删除观察窗口里的符号，可以使用 MPLAB IDE 菜单命令，也可以使用观察窗口里的系统菜单。

建立或打开一个观察窗口以后，从 MPLAB IDE 的菜单里选择命令：**“Window > Watch Window > Edit Active Watch”** 可以显示 **“Edit Watch Symbol”** (编辑“观察”符号)会话窗口。会话窗口里显示了你已经添加到观察窗口里的所有符号。

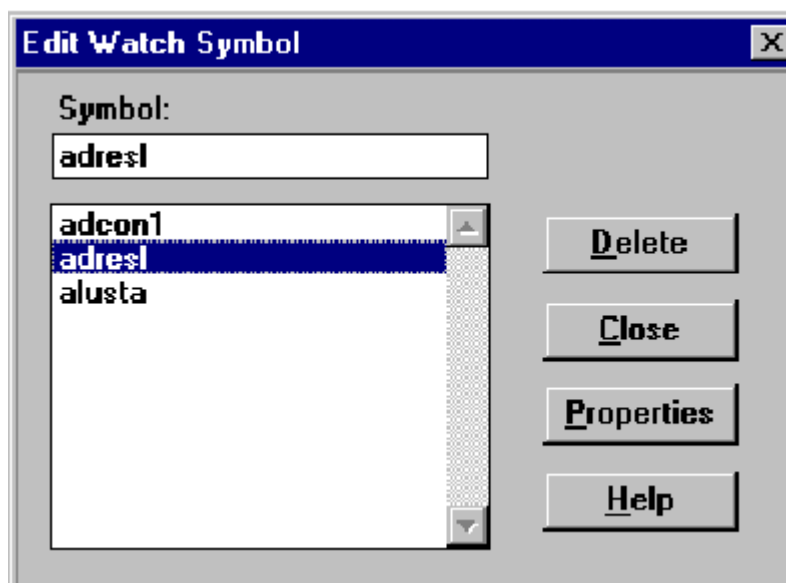


图 7-77: 编辑“观察”符号会话窗口

在 **“Edit Watch Symbol”** (编辑“观察”符号)会话窗口里的符号列表里面，滚动到你想要编辑的变量的符号上，使其高亮，然后点击 **“Properties”** (属性)。

如果用户想删除一个符号，在 **“Edit Watch Symbol”** (编辑“观察”符号)会话窗口里的符号列表里面，滚动到你想要删除的变量的符号上，使其高亮，然后点击 **“Delete”** (删除)。也可以简单的把光标放在观察窗口里的一个符号上，然后在观察窗口里的系统菜单里选择命令：**“Delete > Watch”** 即可。

完成了编辑或者删除“观察”符号的任务后，点 **“Close”** 可以关闭编辑观察符号会话窗口并返回到观察窗口界面。

#### 7. 10. 13. 5 保存“观察”窗口

属性会话窗口允许用户选择在观察窗口里显示的符号的格式。在 **“Add Watch Symbol”** (添加观察符号)窗口和 **“Edit Watch Symbol”** (编辑观察符号)里选择 **“Properties”** (属性)可以出现“属性”会话窗口。

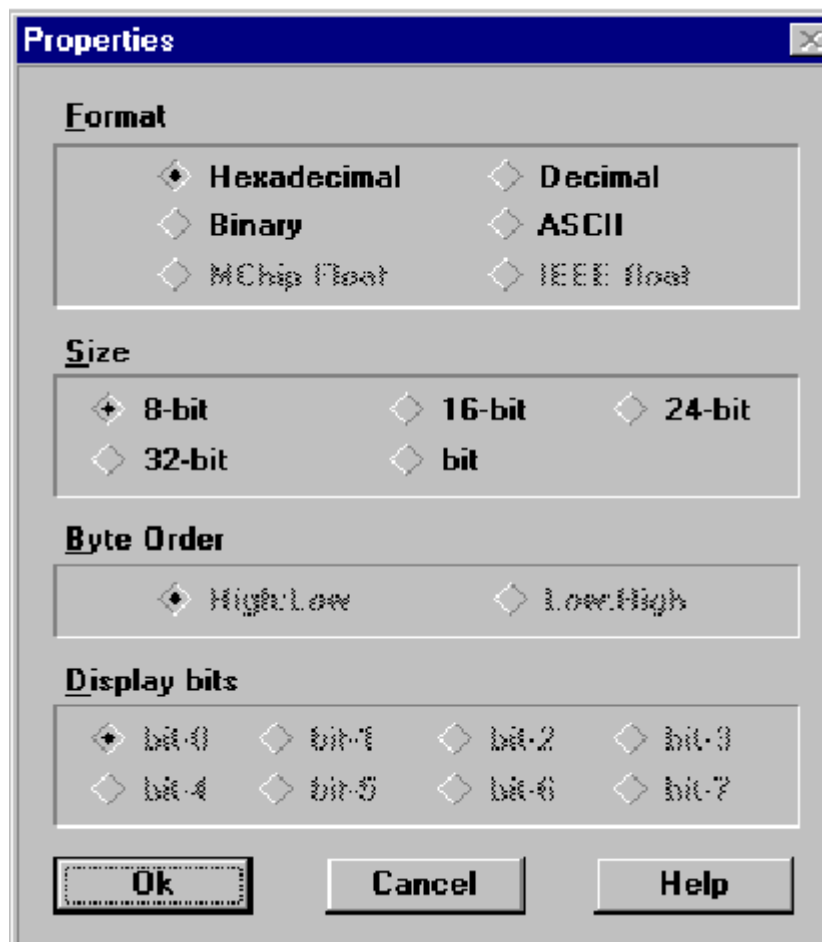


图 7-78: 属性会话窗口

- **Format:** 决定数值的显示类型，可以提供的选择包括 Hexadecimal，Binary，Mchip Float，Decimal，ASCII，IEEE Float，Mchip Float 和 IEEE Float 等，长度的选择有 24 位，32 位。
- **Size:** 决定窗口里显示的数值长度示多少位。
- **Byte Order:** 决定每个字节的显示顺序。可以适合 16 位，24 位和 32 位数据。
- **Display Bits:** 显示指定的位，在“Size”(长度)区域里选择“Bit”(位)，然后选择在这里显示的位。
- **OK:** 点击“OK”接受改变的参数返回到“Edit Watch Symbol”(编辑观察符号)会话窗口。
- **Cancel:** 点击“ ”可以不接受改变的参数，返回到“Edit Watch Symbol”(编辑观察符号)会话窗口。

#### 7. 10. 13. 6 保存观察窗口

如果用户想保存当前观察窗口，可以选择菜单命令：“**Window > Watch Window > Save Watch Window**”即可。

在“Save Watch”(保存观察)会话窗口里，在会话窗口右下方的下拉菜

单里选择一个驱动器名称。点击在 “**Save Watch**” 窗口右上方的 “**Folders**” (文件夹)里的文件夹，在所选择的磁盘上指定路径。在 “**Save Watch**” 窗口左上部的 “**File Name**” 窗口里面输入 “\*” 代表你要保存的的 “观察” 窗口文件名。点击 “**OK**” 即可保存当前观察窗口。

#### 7. 10. 14 修改(Modify)

如果用户想显示和(或)修改 ““数据存储器，程序存储器，堆栈或者 EEPROM 存储器的数值，可以使用菜单命令： “**Window > Modify**”。

修改功能可以允许用户读或者写指定地址，在用户增加地址到下一个的时候进行读或者写，或者填充一个地址块。MPLAB IDE 允许用户可以让 “修改” 窗口随时打开，移动。

MPLAB IDE 提供给用户四种打开 “修改” 会话窗口的方法：

- 选择菜单命令： “**Window > Modify**”。
- 在特殊功能寄存器窗口里双击想修改的对象。
- 在 “观察” 窗口里双击要修改的对象。
- 在文件寄存器窗口里面选择一个地址或地址范围，然后点击鼠标右键，会出现一个菜单，面包含了一个 “**Fill Register(s)**” (填充寄存器) 选择项。从弹出的菜单里选择命令 “**Fill Register(s)**” 可以显示出 “修改” 会话窗口。
- **Address:** 输入将被读或者修改数据的地址。用户可以输入具体的地址数字或者符号(标号)。
- **Data/Opcode:** 点击 “**Read**” 可以读出所选择地址和存储器空间里的数据或者操作代码。点击 “**Write**” 可以将数据或者操作代码保存到所选择的地址和存储器空间里。
- **Radix:** 十六进制或者十进制。
- **Memory Area:** 选择你想修改的存储器区域。  
Data Memory: RAM 存储器  
Program Memory: 仿真器上的ROM 存储器  
Stack: 芯片上的堆栈存储器  
EEPROM: EE 数据存储器。
- **End Address:** “**Fill Range**” (填充范围)的结束地址。
- **Fill Range:** 用在 “Data/Opcode” 里输入的数值填充由这里的两个地址决定的存储器空间。
- **Auto Increment:** 选择这个选择项后，当完成一次读或写操作以后，自动将地址加一。

**注意：** 自动增加功能可以自动增加到下一个地址，显示下一个地址并读出当前地址里的内容。假如你在使用自动增加功能去读一个区域的内容，请将输入的存储器地址减一，因为第一次读操作将会使得地址增加。

- **Write:** 在 Data/Opcode 区域里输入新数据，然后点击 “**Write**” 就可以修改指定地址里的数据。(
- **Read:** 点击 “**Read**” (读)可以读某指定地址的数据。
- **Close:** 点击 “**Close**” 可以退出 “修改” 会话窗口。

**注意：**用户修改堆栈寄存器或者程序指针寄存器的时候要小心。修改这些寄存器导致的后果在暂停的时候是看不见的。

#### 7. 10. 15 水平层叠 (Tile Horizontal)

命令：“**Window > Tile Horizontal**”可以使窗口按照水平层叠，这样可以允许你看见更多的窗口，方便选择相应的工作窗口。这个命令可以让窗口以层叠方式排列，把一个窗口相应的叠放在另外一个窗口的上面，多余的窗口可以在屏幕低一些的地方水平叠放。

含有使用命令“**Tools > DOS Command To Window**”运行 DOS 命令输出结果的窗口优先排列在屏幕的上部。

#### 7. 10. 16 垂直层叠 (Tile Vertical)

命令：“**Window > Tile Vertical**”可以使窗口按照垂直方向层叠，这样可以允许你看见更多的窗口，方便选择相应的工作窗口。

这个命令可以让窗口以层叠方式排列，把一个窗口相应的叠放在另外一个窗口的上面，以便可以使每个窗口尽可能深。

#### 7. 10. 17 Cascade

使用菜单命令：“**Window > Cascade**”可以使打开的窗口按照 cascade 模式排列。

#### 7. 10. 18 全部图标化

选择菜单命令：“**Window > Iconize All**”可以使得全部窗口图标化。

#### 7. 10. 19 排列图标

选择菜单命令：“**Window > Arrange Icons**”可以重新排列图标化后的窗口，这样所有的图标可以在桌面下面排列成一行。这个命令不影响已经打开的窗口。

#### 7. 10. 20 打开窗口

菜单命令：“**Window > Open Windows**”可以列出打开的窗口。

任何时候用户打开一个窗口的时候，MPLAB IDE 会把名称记录在列表里面，这样你最近使用过的窗口会一直出现在屏幕上部。

- **More Windows:** 假如窗口菜单里保存不了太多的文件，那么就会自动添加更多的窗口到窗口菜单里。该命令会打开一个会话窗口，这里显示了所有的窗口列表并可以让你选择你想使用得窗口。要想从列表里打开一个窗口，可以用鼠标左键双击列表里的窗口名称或者选择某一名称然后点击“**Open**”。更多的会话窗口还提供更多的选择项。这些选择项的设置读被记录下来并成为下一次使用这个会话窗口时的默认设置。

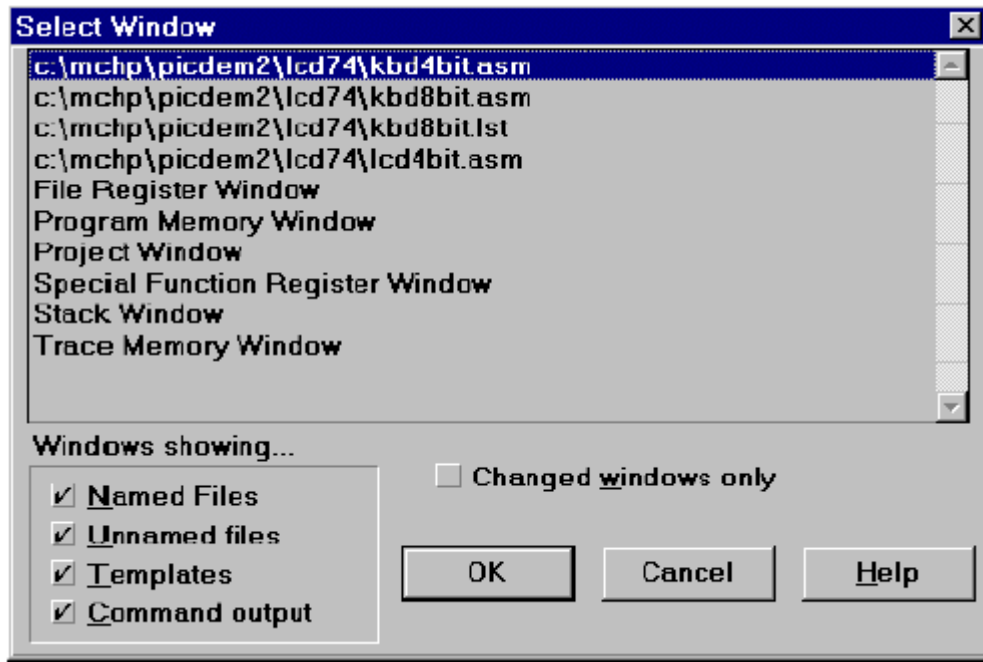


图 7-80: “更多窗口” 会话窗口

- **Changed Windows Only:** 显示出自窗口打开以来被修改过的窗口。
- **Named Files:** 列出所有具有相近文件名的窗口。
- **Unnamed Files:** 列出所有没有相近文件名的窗口。
- **Templates:** 列出所有剪切板窗口。
- **Command Output:** 列出所有包含了由于使用命令: “**Tools > DOS Command to Window**” 后的输出的窗口。

## 7. 11 帮助菜单

### 7. 11. 1 发布信息 (Shift+F1)

菜单命令: “**Help > Release Notes**” 可以打开和显示 MPLAB 背景 IDE 开发环境最近的设置改变历史。发布信息在文件 “README.LAB” 里面。

### 7. 11. 2 开发工具发布信息

假如你使用了另外的开发工具, 在帮助菜单里可能找到那些工具相应的帮助信息发布。

### 7. 11. 3 MPLAB 处理 IDE 帮助

命令: “**Help > MPLAB Help**” 可以看到关于 MPLAB IDE 的帮助信息。

### 7. 11. 4 编辑器帮助

命令: “**Help > Editor Help**” 可以看到关于 MPLAB 编辑器的帮助信息。

### 7. 11. 5 错误帮助

命令: “**Help > Error Help**” 可以看到关于 MPLAB IDE 错误消息帮助信息。



### 7. 11. 6 MPASM 帮助

命令: “**Help > MPASM Help**”可以看见在线《MPASM 以及 MPLINK 和 MPLIB 用户指南》和关于 MPASM 的信息。点击绿色带下划线的文字, 可以看见详细介绍。

MPASM 帮助还包含快速《参考指南》, 讲述的是汇编指示语言和各种不同芯片指令集。

### 7. 11. 7 MPLINK 帮助

使用菜单命令: “**Help > MPLINK Help**” 可以看见在线《MPASM 以及 MPLINK 和 MPLIB 用户指南》和关于 MPLIB 的信息。点击绿色带下划线的文字, 可以看见详细介绍。

### 7. 11. 8 开发工具帮助

假如你使用了其他的开发工具, 在帮助菜单里可能有这些开发工具的帮助信息。

### 7. 11. 9 关于

在菜单: “**Help > About**” 里面可以看到以下内容:

- MPLAB IDE 版本
- MICROCHIP 联系地址
- 处理器版本
- 反汇编器的版本
- 关于其他应用程序的信息

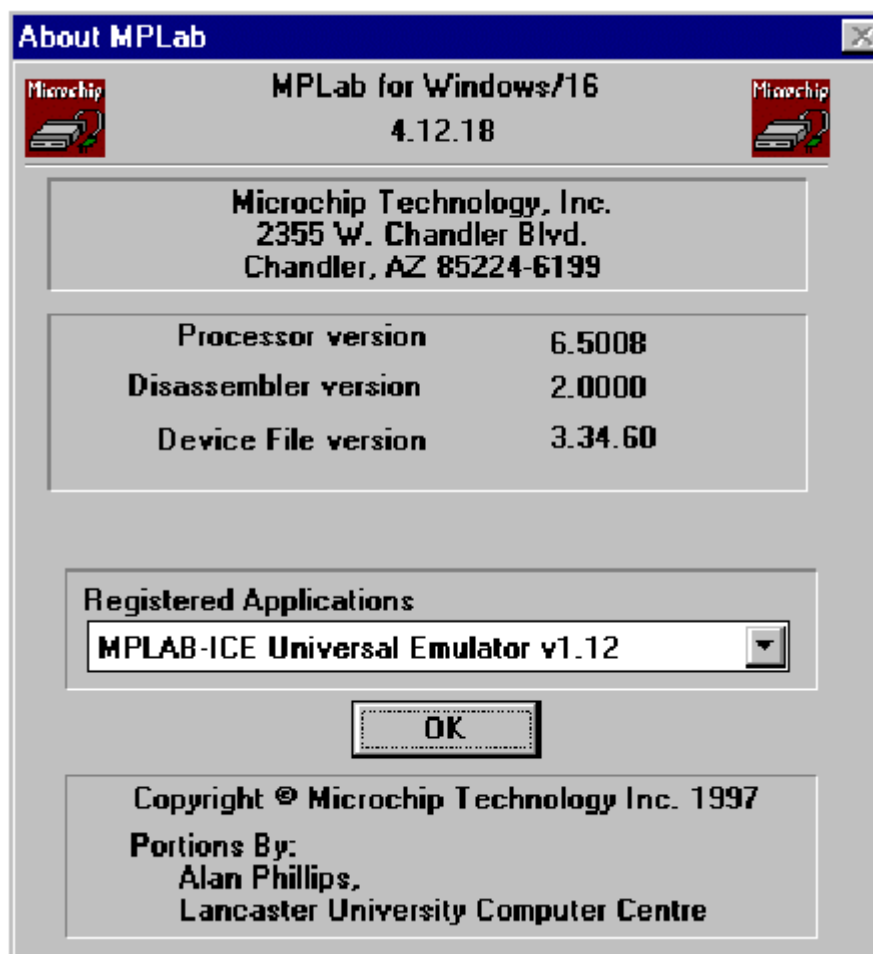


图 7-81: 关于帮助会话窗口

## 附录 A：MPLAB IDE 里使用的快捷键及其相应的功能

### A. 1：概述

本附录列出了 MPLAB IDE 里使用的快捷键及其相对应的功能。

### A. 2：MPLAB IDE 快捷键及其对应功能

组合键功能描述	功能定义	默认组合键
CursorBottomOfWindow	光标移到窗口底部	Ctrl+PgDn
CursorBottomOfWindowSelect	光标移到窗口底部并选择	Ctrl+Shift+PgDn
CursorDown	光标下移	Down
CursorDownSelect	光标向下移动并选择	Shift+Down
CursorEndOfFile	光标移动到文件的结尾	Ctrl+End
CursorEndOfFileSelect	光标移动到文件的结尾并选择	Ctrl+Shift+End
CursorEndOfLine	光标移动到行的末尾	End
CursorEndOfLineSelect	光标移动到行的末尾并选择	Shift+End
CursorLeft	光标左移动	Left
CursorLeftSelect	光标向左移动并选择	Shift+Left
CursorLeftWord	光标向左移动一个字	Ctrl+Left
CursorLeftWordSelect	光标向左移动一个字并选择	Ctrl+Shift+Left
CursorPageDown	光标下移动	PgDn
CursorPageDownSelect	光标向下移动并选择	Shift+PgDn
CursorPageUp	光标上移动	PgUp
CursorPageUpSelect	光标向上移动并选择	Shift+PgUp
CursorRight	光标右移动	Right
CursorRightSelect	光标向右移动并选择	Shift+Right
CursorRightWord	光标向右移动一个字	Ctrl+Right
CursorRightWordSelect	光标向右移动一个字并选择	Ctrl+Shift+Right
CursorStartOfFile	光标移动到文件开始	Ctrl+Home
CursorStartOfFileSelect	光标移动到文件开始并选择	Ctrl+Shift+Home
CursorStartOfLine	光标移动到行开始	Home
CursorStartOfLineSelect	光标移动到行开始并选择	Shift+Home
CursorStartOfText	光标移动到文本的开始	Alt+Home
CursorStartOfTextSelect	光标移动到文本的开始并选择	Alt+Shift+Home
CursorTopOfWindow	光标移动到窗口顶部	Ctrl+PgUp
CursorTopOfWindowSelect	光标移动到窗口顶部并选择	Ctrl+Shift+PgUp
CursorUp	光标往上	Up
CursorUpSelect	光标往上并选择	Shift+Up
DebugAnimate	“动画”执行	Ctrl+F9
DebugBreak	断点设置	F2
DebugCenterDebug	调试位置“对中”	

DebugChangePC	改变程序指针	
DebugClearAll	清除所有的鉴别符 (Qualifiers)	
DebugClearMemory	清除程序存储器	Ctrl+Shift+F2
DebugConditionalBreak	条件终止	
DebugExecuteOpcode	执行一个代码	
DebugHalt	停止处理器	F5
DebugHaltTrace	停止跟踪	Shift+F5
DebugPerformanceAnalysis	效能分析	
DebugPORReset	上电复位	Ctrl+Shift+F5
DebugStep	单步执行	F7
DebugReset	复位处理器	F6
DebugRun	运行	F9
DebugStepOver	单步跳过运行	F8
DebugStepTrace	单步执行跟踪	Shift+F7
DebugAsyncStim	异步激励	
DebugClockStim	时钟激励	
DebugPinStim	管脚激励	
DebugRegStim	寄存器激励	
DebugSystemReset	系统复位	Ctrl+Shift+F3
DebugTrace	跟踪设置	
DebugTrigger	触发器设置	
DebugUpdateRegisters	更新寄存器状态	
EditCancelSelection		keypad5
EditClearUndo	废除所有保存的“撤消”动作	
EditCopy	将高亮文本复制到剪切板	Ctrl+C Ctrl+Ins
EditCut	将高亮文本剪贴到剪切板	Ctrl+X Shift+Del
EditDeleteBackwards	往后删除字符	Backspace Ctrl+H
EditDeleteForwards	往后删除	Del
EditDeleteLine	删除光标所在行的内容	Ctrl+Shift+K
EditDeleteSelection	删除选择内容	
EditDeleteToEndOfLine	删除从光标处到本行结束的内容	Ctrl+K
EditFind	查找一个文本串	F3
EditGotoLine	将光标移动到指定的行	Ctrl+G
EditInsertHardTab	插入硬 TAB	
EditInsertSoftTab	插入软 TAB	
EditInsertTab	插入 TAB	Tab Ctrl+I
EditMarkUnchanged	标记文件为“未改变”	
EditNewLine	插入新行	Enter Ctrl+M
EditPaste	将剪贴板里的内容粘贴到光标所在的位置	Ctrl+V Shift+Ins
EditRepeatLastFind	重复查找上一次查找的内容	Shift+F3
EditRepeatLastReplace	重复替换上一次替换的内容	Shift+F4
EditReplace	替换文本串	F4
EditSelectAll	高亮当前窗口里的内容	
EditSelectLine	将窗口滚动到看得见光标的地方	

EditShowNextLine	查看下一行	
EditShowNextPage	查看下一页	
EditShowPreviousLine	查看前一列	
EditShowPreviousPage	查看前一页	
EditSplitLine	切断行	Ctrl+Shift+O
EditTextIndent	用 TAB 键移动字符	
EditTextInsertASCIIcode	插入一个特定的字符代码	Ctrl+Q
EditTextLowercaseSelection	将高亮的文本变为小写	
EditTextMatchBrace	移动到匹配的括号处	Ctrl+B
EditTextMatchBraceSelect	移动到匹配的括号处并选择	Shift+Ctrl+B
EditTextTransposeCharacters	交换光标左右的字符	Ctrl+T
EditTextUnIndent	将文本左移动一个 TAB	
EditTextUppercaseSelection	将高亮的文本变为大写	
EditTextWidenBraceSelect	高亮下一个最大的括号区域里的文本	Shift+Ctrl+W
EditUndo	撤消上一次编辑操作	Ctrl+Z
ExecDosCommand	执行 DOS 命令并捕获输出	F11
ExecRepeatDosCommand	重复执行 DOS 命令并捕获输出	Ctrl+F11
FileAbandon	废除文件	
FileClose	在当前窗口关闭文件	
FileCloseAll	关闭所有打开的文件	Shift+F9
FileExit	终止 MPLAB IDE 任务，退出	Alt+F4
FileImportDownloadToMemory	下载十六进制文件到内存里	
FileImportDownloadToTarget	下载十六进制文件到目标板里	
FileImportReadTarget	拷贝内存里的 16 进制数据到目标板里	
FileInsert	在当前光标处插入一个文件	
FileName	改变当前窗口里的文件名	
FileNew	创建一个新的空的编辑窗口	Ctrl+N
FileOpen	打开一个现存的文件	Ctrl+O
FilePrint	打印一个当前文件	Ctrl+P
FilePrintSetup	改变当前打印机的设置	
FileSave	将当前文件存储到磁盘里面	Ctrl+S
FileSaveAll	将所有打开的文件存到磁盘里面	
FileSaveAs	将当前文件存到磁盘里面	
FileSaveHex	保存十六进制文件	
FileSaveTrace	保存跟踪	
FileSimulatorStimulus	调入模拟激励文件	
FileView	将一个现有文件以只读方式打开	
FileWrite	将当前文件写入磁盘里面	
HelpAbout	给出关于 MPLAB IDE 的版本信息	
HelpBugs	给出关于 MPLAB IDE 软件漏洞列表	
HelpCommands	在 MPLAB IDE 命令行给出帮助信息	
HelpContents	在 MPLAB IDE 帮助内容里选择帮助	F1
HelpEditor	编辑器帮助	
HelpMpasm	MPASM 宏汇编帮助	
HelpMpc	MPLAB-C 帮助	
HelpOnHelp	使用帮助系统提供帮助	
HelpPICmicro	PICMICRO 用户指南	
HelpReleaseNotes	软件发布信息	Shift+F1

OptionsColors	编辑颜色选择设置	
OptionsCommunicationsPort	设置通讯地址	
OptionsCurrent	为当前窗口和文件设置模式	
OptionsDefault	为文件设置默认模式	
OptionsDevelopmentMode	设置开发模式	
OptionsEnvironmentSetup	设置环境	
OptionsHardware	选择硬件设置	
OptionsKeyMapping	改变快捷键设置	
OptionsLoadSetup	调入设置文件	
OptionsMultiProcessor	设置多个处理器	
OptionsPreferences	设置 MPLAB IDE 配置选择	
OptionsResetModes	设置当前文件/窗口模式为默认值	
OptionsSaveSetup	保存设置文件	
OptionsScreenFontANSI	从标准 ANSI 界面里选择屏幕界面	
OptionsScreenFontOEM	设置屏幕界面为标准 OEM 界面	
OptionsScreenFontOther	从所有现有的界面里面选择屏幕界面	
OptionsScreenFontSystem	将屏幕设置为标准系统界面	
OptionsSimulatorIOSetup	设置模拟器 I/O	
OptionsToggleInsertMode	切换插入模式	Ins
OptionsToggleLineNumbers	切换行号	
OptionsToggleStatusBar	隐藏或出现状态栏	
OptionsToolbarBottom	将工具栏移动到窗口的底面	
OptionsToolbarFloat	使工具栏变为浮动窗口	
OptionsToolbarHide	使工具栏不可见	
OptionsToolbarLeft	将工具栏移动到窗口的右面	
OptionsToolbarRight	将工具栏移动到窗口的左面	
OptionsToolbarShow	使工具栏可见	
OptionsToolbarTop	将工具栏移动到窗口的上面	
ProjectBuildAll	创建所有项目	Ctrl+F10
ProjectBuildNode	创建节点	Alt+F10
ProjectCloseProject	关闭项目	
ProjectCompileSingle	编译单个文件	Alt+F10
ProjectEditProject	编辑项目定义	Ctrl+F3
ProjectMakeProject	创建当前项目	F10
ProjectMakeSetup	设置关于项目“创建”选择项	
ProjectNewProject	建立新项目	
ProjectOpenProject	打开一个项目	
ProjectSaveProject	保存当前项目	
SwapToolbar	交换工具栏	
SysSetMenuMode	设置菜单模式	
TemplateDelete	从模板文件里删除一个模板	
TemplateEdit	在模板文件里编辑一个模板	
TemplateFileAttach	调入模板一个模板文件供使用	
TemplateFileCreate	建立一个空的模板文件里	
TemplateFileDetach	放弃一个附加的模板文件	
TemplateFileSave	保存一个修改过的模板文件	
TemplateFindMark	在当前窗口寻找一个作了标记的模板	
TemplateInsert	在光标处插入一个模板	
TemplateInsertMark	在光标处插入一个模板标记	

TemplateNew	为一个模板建立新的窗口	
TemplateStore	将模板保存到模板文件里	
TemplateStoreAs	将模板保存到模板文件里	
ToolsEmulatorConfiguration	设置仿真器配置	
ToolsProgramHeader	烧写仿真头	
ToolsProgramPod	烧写仿真 POD	
ToolsVerifyEmulator	仿真器校验	
WindowAbsoluteListing	绝对列表	
WindowArrangelcons	整齐排列窗口里所有图标	
WindowCascade	层叠方式排列图标	
WindowClose	关闭当前窗口	Ctrl+F4
WindowDuplicate	复制当前窗口	
WindowEeprom	EEPROM 存储器窗口	
WindowFileRegisters	文件寄存器存储器	
WindowIconize	图标化窗口	
WindowIconizeAll	将所有窗口变为图标	
WindowLoadWatch	调入观察窗口	
WindowMaximize	最大化窗口	
WindowModify	修改窗口	
WindowNewWatch	创建新的观察窗口	
WindowNext	激活下一个非图标化窗口	
WindowProgramMemory	程序存储器窗口	
WindowRestore	复位窗口	
WindowSelect	从列表里选择一个窗口并激活	
WindowSpecialFunctionRegisters	特殊功能寄存器窗口	
WindowStack	堆栈窗口	
WindowStopwatch	“跑表”窗口	
WindowSymbolList	符号列表窗口	
WindowTileHorizontal	排列窗口使其最大高度	
WindowTileVertical	排列窗口使其最大宽度	
WindowTrace	跟踪窗口	
WindowWiden	最小化当前窗口的宽度	

## 附录 B: MPLAB IDE 里使用的文件扩展名

MPLAB IDE 里面使用的默认扩展名有以下这些类型:

- .ASM      汇编语言源文件
- .C        C 语言源文件
- .CFG      配置/设置文件
- .CSV      跟踪保存文件（只有 MPLAB-ICE2000 适用）
- .COD      包含符号信息和目标代码
- .DAT      模拟数据文件
- .ERR      由汇编器/编译器产生的错误文件
- .H        C 语言包含文件
- .HEX      PICMICRO 的十六进制机器代码文件
- .HLP      帮助文件
- .INC      汇编语言包含文件
- .INI      MPLAB IDE 和语言工具设置文件
- .KEY      MPLAB IDE 快捷键文件
- .LKR      MPLINK 链接器描述文件
- .LST      由汇编器/编译器产生的绝对列表文件
- .MTC      语言工具设置文件
- .PJT      包含项目信息的文件
- .REG      寄存器激励文件
- .STI      管脚激励文件
- .TB       条件断点跟踪文件
- .TBR      工具栏文件
- .TPL      Template file
- .TRC      跟踪保存文件
- .TXT      Trace save file other than MPLAB-ICE 2000
- .EAT      观察窗口文件



## 附录 C：MPLAB IDE 工具栏和状态栏定义

### C. 1 MPLAB IDE 工具栏

#### C. 1. 1 编辑工具栏

编辑工具栏包含编辑源文件时候经常使用的按钮。

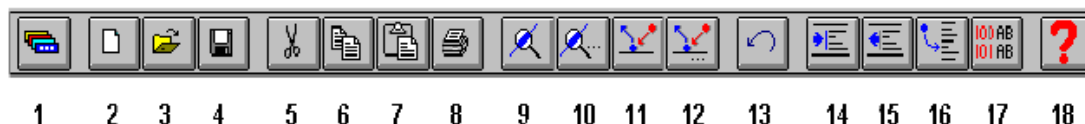


图 7-82：编辑工具栏默认的按钮有以下这些：

- |                         |              |
|-------------------------|--------------|
| 1. Change toolbar       | 改变工具栏        |
| 2. File New             | 建立新文件        |
| 3. File Open            | 打开文件         |
| 4. File Save            | 保存文件         |
| 5. Cut                  | 切除           |
| 6. Copy                 | 拷贝           |
| 7. Paste                | 剪切           |
| 8. Print                | 打印           |
| 9. Find                 | 查找           |
| 10. Repeat Last Find    | 重复查找上一次查找的内容 |
| 11. Replace             | 替换           |
| 12. Repeat Last Replace | 重复查找上一次查找的内容 |
| 13. Undo Edit           | 撤消上一次的编辑动作   |
| 14. Indent              |              |
| 15. Unindent            |              |
| 16. Goto Line           | 跳到某一行        |
| 17. Toggle Line Numbers | 切换行号         |
| 18. Help Contents       | 帮助包含的内容      |

#### C. 1. 2 调试工具栏

调试工具栏包含在运行和调试代码的时候通常用到的按钮

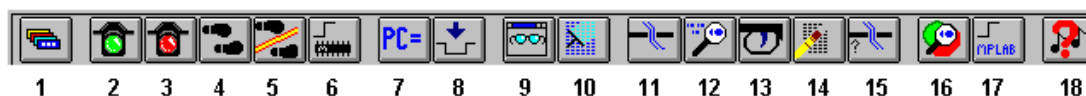


图 7-83：调试工具栏

默认的按钮有以下这些：

- |                         |          |
|-------------------------|----------|
| 1. Change toolbar       | 改变工具条    |
| 2. Run Program          | 运行程序     |
| 3. Halt Program         | 暂停程序     |
| 4. Step Through Program | 单步运行程序   |
| 5. Step Over            | 单步跳过运行程序 |
| 6. Reset System         | 复位系统     |
| 7. Change PC            | 改变程序指针   |
| 8. Execute Opcode       | 执行操作码    |
| 9. New Watch Window     | 新建观察窗口   |
| 10. Modify Window       | 修改窗口     |
| 11. Break Point         | 断点       |
| 12. Trace Point         | 跟踪点      |
| 13. Trigger             | 触发器      |
| 14. Clear All Breaks    | 清除所有断点   |
| 15. Conditional Break   | 条件断点     |
| 16. Halt Trace          | 暂停跟踪     |
| 17. System Reset        | 系统复位     |
| 18. Help Release Notes  | 帮助和发布信息  |

### C. 1. 3 项目工具栏

项目工具栏包含在运行和调试代码的时候通常用到的图标。



图 7-84：项目工具栏

默认的按钮有以下这些：

- |                   |       |
|-------------------|-------|
| 1. Change toolbar | 改变工具条 |
| 2. New Project    | 新建项目  |
| 3. Open Project   | 打开项目  |
| 4. Close Project  | 关闭项目  |
| 5. Save Project   | 保存项目  |
| 6. Edit Project   | 编辑项目  |
| 7. Make Project   | 创建项目  |
| 8. Build All      | 全部创建  |
| 9. Build Node     | 创建节点  |
| 10. Install Tools | 安装工具  |
| 11. Help          | 帮助    |

### C. 1. 4 用户定义工具栏

用户定义工具栏可以让用户根据自己要求定义各自的按钮：

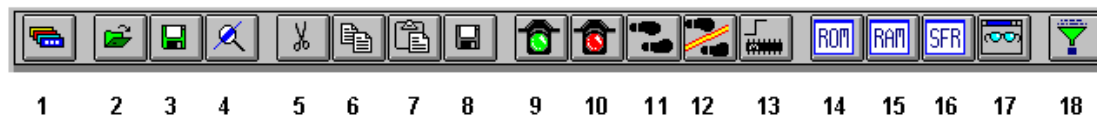


图7-85: 用户定义工具栏

起初按钮条里面包含如下按钮:

1. **Change Toolbar** : 改变工具条
2. **Open Project** : 打开项目
3. **Save Project** : 保存项目
4. **Find** : 查找
5. **Cut** : 剪切
6. **Copy** : 拷贝
7. **Paste** : 粘贴
8. **Save File** : 保存文件
9. **Run** : 运行
10. **Halt** : 暂停
11. **Step** : 单步
12. **Step Over** : 单步跳过
13. **Reset** : 复位
14. **Program Memory Window** : 程序存储器窗口
15. **File Registers Window** : 文件寄存器窗口
16. **Special Function Registers Window** : 特殊功能寄存器窗口
17. **New Watch Window** : 新建观察窗口
18. **Make Project** : 创建项目

## C. 2 MPLAB IDE 状态栏

状态栏 (Status Bar) 显示的是光标位置坐标, 开发模式, 芯片类型, 当前有效的工具栏 (active toolbar) 等当前信息。



图7-86: 状态栏 (Status Bar)

		描述	双击后结果
行，列号（窗口打开的时候） 或：MPLAB IDE版本号（没有窗口打开的时候）	Ln 1 Col 1 2.00.00	显示文件里当前行和列号 MPLAB称IDE版本号	打开“跳到某行”  无操作
文件里的行	72	显示当前文本文件里的行数	无操作
被修改的文件	#	打开的文件被修改过则显示“#”	无操作
只写/读	WR		无操作
文本交换			
插入/	INS		在INS和OVR间切换
当前处理器	PIC16C61	显示当前处理器	无操作
当前程序指针	pc:0x5f	显示当前程序指针	打开“改变程序指针”会话窗口
当前W寄存器值	W:0x00	显示当前W寄存器值	无操作
状态位	ov Z dc c	大写=置位（1） 小写=复位（0）	无操作
允许全局断点（Break）	Bk On	显示当前全局断点的状态为“允许”	使全局断点在ON和OFF之间切换
当前开发模式	Sim	显示当前开发模式： EO = 纯编辑状态 Sim=模拟-MPLAB-SIM Si=模拟器- SIMICE ICE=仿真器MPLAB-ICE Em=仿真器-PICMASTER	显示开发模式会话窗口
处理器时钟	4 MHz	显示当前处理器时钟	打开处理器时钟会话窗口
当前工具栏	Edit	显示当前工具栏	无操作

## 附录D MPLAB编辑器默认命令键

### D. 1 概述

本附录描述了MPLAB编辑器默认命令键，并且列出了相应的菜单命令（假如有的话）。

键命令执行了大多数的快速操作命令。这些键可以修改定义为不同的功能。默认状态下，没有定义组合键，只定义了单键命令。

参考附录A：MPLAB IDE功能键，可以查找到这些键对应的不同功能。

### D. 2 重点

这些默认键命令有：

- 功能键
- 移动键
- 控制键
- 格式和编辑键

## D. 3 功能键

<b>F1</b>	当经过一个窗口（如文件寄存器窗口）时候，显示相应的帮助标题
<b>F2</b>	执行命令 <b><u>Debug &gt; Break Settings</u></b> 打开断点设置会话窗口
<b>F3</b>	执行命令 <b><u>Edit &gt; Find</u></b> 来查找字符串
<b>F4</b>	执行命令 <b><u>Edit &gt; Replace</u></b> 来替换字符串
<b>F5</b>	执行命令 <b><u>Debug &gt; Halt</u></b> 来暂停调试
<b>F6</b>	执行命令 <b><u>Debug &gt; Reset</u></b> 来复位被模拟或仿真的处理器
<b>F7</b>	执行命令 <b><u>Debug &gt; Step</u></b> 来运行程序存储器里的一个代码
<b>F8</b>	执行命令 <b><u>Debug &gt; Step Over</u></b> 来运行程序存储器里的一个调用语句
<b>F9</b>	执行命令 <b><u>Debug &gt; Run</u></b> 来运行被模拟或仿真的处理器
<b>F10</b>	执行命令 <b><u>Project &gt; Make Project</u></b> 来为当前项目进行“创建”
<b>F11</b>	执行命令 <b><u>Execute &gt; DOS Command To Window</u></b> ，运行DOS命令并将结果输出到窗口里
<b>Shift+F3</b>	执行命令 <b><u>Edit &gt; Repeat Find</u></b> 来重复上一次查找操作
<b>Shift+F4</b>	执行命令 <b><u>Edit &gt; Repeat Replace</u></b> 来重复上一次替换操作
<b>Shift+F5</b>	执行命令 <b><u>Debug &gt; Halt Trace</u></b> 停止对被模拟处理器进行跟踪
<b>Shift+F9</b>	执行命令 <b><u>File &gt; Close All</u></b> 来关闭所有打开的文件和窗口
<b>Ctrl+F2</b>	执行命令 <b><u>Project &gt; Open Project</u></b> 打开“打开项目”会话窗口
<b>Ctrl+F3</b>	执行命令 <b><u>Project &gt; Edit Project</u></b> 打开“编辑项目”会话窗口
<b>Ctrl+F7</b>	执行命令 <b><u>Options &gt; Environment Setup</u></b> 打开“环境设置”会话窗口
<b>Ctrl+F8</b>	执行命令 <b><u>Windows &gt; Show Symbol List</u></b> 打开“符号列表”会话窗口
<b>Ctrl+F10</b>	执行命令 <b><u>Project &gt; Build All</u></b> 创建当前项目的所有源程序
<b>Alt+F4</b>	执行命令 <b><u>File &gt; Exit</u></b> 结束MPLAB任务
<b>Alt+F10</b>	执行命令 <b><u>Project &gt; Build Node</u></b> 创建当前节点
<b>Shift+Ctrl+F2</b>	执行命令 <b><u>Debug &gt; Clear Program Memory</u></b> 清除程序存储器为“erased”（清除）状态
<b>Shift+Ctrl+F3</b>	执行命令 <b><u>Debug &gt; System Reset</u></b> 复位所有的模拟系统
<b>Shift+Ctrl+F5</b>	执行命令 <b><u>Debug &gt; POR Reset Emulation</u></b> 打开“POR复位”会话窗口

## D. 4 移动键

这些移动键加上SHIFT键可以得到扩充的组合键。

需要指出的是：**Alt**键既可以由箭头键，又可以由**Home, End, PgDn, PgUp, Ins** 或 **Del**中某一个键组合而成。你必须使用扩展键盘区的键，而不是数字小键盘上的键。

<b>Up</b>	将光标往上移动一行
<b>Shift+Up</b>	将光标往上移动一行，并扩展高亮区
<b>Down</b>	将光标往下移动一行
<b>Shift+Down</b>	将光标往下移动一行，并扩展选择区
<b>Left</b>	将光标往左移动一行
<b>Shift+Left</b>	将光标往左移动一行，并扩展选择区
<b>Ctrl+Left</b>	将光标往左移动一个字
<b>Ctrl+Shift+Left</b>	将光标往左移动一个字，并扩展选择区
<b>Right</b>	将光标往右移动一行
<b>Shift+Right</b>	将光标往右移动一行，并扩展选择区
<b>Ctrl+Right</b>	将光标往右移动一个字
<b>Ctrl+Shift+Right</b>	将光标往右移动一个字，并扩展选择区
<b>PgDn</b>	将光标往下移动一页

<b>Shift+PgDn</b>	将光标往下移动一页，并扩展选择区
<b>Ctrl+PgDn</b>	将光标往下移动到窗口的最后一行的开始位置
<b>Ctrl+Shift+PgDn</b>	将光标往下移动到窗口的最后一行的开始，并扩展选择区
<b>PgUp</b>	将光标往上移动一页
<b>Shift+PgUp</b>	将光标往上移动一页，并扩展选择区
<b>Ctrl+PgUp</b>	将光标往上移动到窗口的最后一行的开始位置
<b>Ctrl+Shift+PgUp</b>	将光标往上移动到窗口的最后一行的开始，并扩展选择区
<b>Home</b>	将光标往 移动到行的开始位置
<b>Shift+Home</b>	将光标移动到行的开始位置，扩展选择区域
<b>Ctrl+Home</b>	将光标移动到文件的开始位置
<b>Ctrl+Shift+Home</b>	将光标移动到文件的开始位置，扩展选择区域
<b>Alt+Home</b>	将光标移动到本行第一个非空位置
<b>Alt+Shift+Home</b>	将光标移动到本行第一个非空位置，扩展选择区域
<b>End</b>	将光标移动到行的结束位置
<b>Shift+End</b>	将光标移动到行的结束位置，扩展选择区域
<b>Ctrl+End</b>	将光标移动到文件的结束位置
<b>Ctrl+Shift+End</b>	将光标移动到文件的结束位置，扩展选择区域

## D. 5 控制键 (Control Keys)

<b>Ctrl+Shift+B</b>	将光标移动到与当前光标位置处的括号相匹配的括号处，并将这两个相匹配括号之间的文本（包括括号）高亮
<b>Ctrl+C</b>	执行命令 <b>Edit &gt; Copy</b> 将选中的文本拷贝到剪切板里
<b>Ctrl+G</b>	执行命令 <b>Edit &gt; Goto Line</b> 将光标移动到指定的行
<b>Ctrl+H</b>	将光标左边的字符删除
<b>Ctrl+I</b>	插入一个TAB，或者指定的空格数，使光标移动到下一个TAB位置。空格数目决定于当前WINDOWS窗口模式是设置为硬TAB或软TAB
<b>Ctrl+K</b>	执行命令: <b>Edit &gt; Delete To End Of Line</b> 删除从光标起右边的所有字符
<b>Ctrl+Shift+K</b>	执行命令: <b>Edit &gt; Delete Line</b> 删除光标所在处的所有字符
<b>Ctrl+N</b>	执行命令: <b>File &gt; New</b> 建立一个新的空的编辑窗口
<b>Ctrl+O</b>	执行命令: <b>File &gt; Open</b> 打开一个现有的文件
<b>Ctrl+Shift+O</b>	在光标所在处切开当前行，光标位置不变
<b>Ctrl+P</b>	执行命令: <b>File &gt; Print</b> 打印当前窗口里的内容
<b>Ctrl+Q</b>	执行命令: <b>Edit &gt; Text Insert ASCII Code</b> 插入一个按ASCII代码指定的字符
<b>Ctrl+S</b>	执行命令: <b>File &gt; Save</b> 保存当前文件到磁盘
<b>Ctrl+T</b>	执行命令: <b>Edit &gt; Text Transpose Characters</b> 可以使光标所在处的字符和其左边的字符互换
<b>Ctrl+V</b>	执行命令: <b>Edit &gt; Paste</b> 可以将剪切板里的内容复制到当前窗口里
<b>Ctrl+W</b>	执行命令: <b>Window &gt; Select</b> 可以在很多打开的窗口之间选择
<b>Ctrl+Shift+W</b>	将光标放置在括号之间，该指令可以高亮最近的（最里面）一对括号之间文本
<b>Ctrl+X</b>	执行命令: <b>Edit &gt; Cut</b> 将当前选择的内容剪切到剪切板里面
<b>Ctrl+Z</b>	执行命令: <b>Edit &gt; Undo</b> 撤消上一个编辑动作

**D. 6 格式和编辑键（Formatting and Editing Keys）**

<b>Enter</b>	插入一个新的行
<b>BackSpace</b>	删除光标左边一个字符
<b>Del</b>	删除光标左边一个字符
<b>Shift+Del</b>	执行命令: <b>Edit &gt; Cut</b> 将选择的文本剪切到剪切板里面
<b>Ins</b>	将当前窗口的工作模式在“插入”和“替换”之间切换
<b>Shift+Ins</b>	执行命令: <b>Edit &gt; Paste</b> 将剪切板里面的文本复制到窗口里面
<b>Ctrl+Ins</b>	执行命令: <b>Edit &gt; Copy</b> 将选择的文本拷贝到剪切板里面
<b>Tab</b>	插入一个TAB, 或者指定的空格数, 使光标移动到下一个TAB位置。 空格数目决定于当前WINDOWS窗口模式是设置为硬TAB还是软TAB

**缩略语****概述**

本章给出通用的参考信息, 定义了很多MICROCHIP开发工具的缩略语和术语。

**重点**

本章包含了以下MICROCHIP开发工具里涉及到的术语。

- MPLAB IDE, MPLAB-SIM, MPLAB Editor (编辑器)
- MPASM, MPLINK, MPLIB
- MPLAB-CXX (C语言开发工具)
- MPLAB-ICE, PICMASTER Emulators (仿真器)
- MPLAB-ICD (基于FLASH的低成本开发工具)
- PICSTART Plus, PRO MATE programmer (烧写器)

**术语 – MPLAB IDE, MPLAB-SIM, MPLAB Editor****Access RAM (只针对PIC18CXXX系列MCU)**

PIC18CXXX系列MCU里面的通用功能寄存器, 允许在无须考虑堆选择位 (bank select bit - BSR) 的情况下对其任意访问。

**Application : 应用**

用户开发的一系列软件和硬件, 通常是由一个PICMICRO 微控制器控制下的一个产品。

**Break Point – Software**

是一个地址, 固件程序 (firmware) 运行到这里会停下来, 通常设置了一个特殊的断点操作代码。

**Build : 创建**

将对所有的源程序进行重新编译。

**Calibration Memory : 校正存储器**

是一个或多个特殊功能寄存器, 用来保存校准PICMICRO微控制器上RC振荡器的参数。

**Configuration Bits : 配置位**

用来设置PICMICRO微控制器 工作模式的可编程位。配置位可以或者不可以被

预先烧写。 可以使用命令：“***Options > Development Mode***” 来为模拟器或者仿真器设置这些位，用指示语言：“ \_ \_ CONFIG” 来为烧写器设置这些位。

**Data Memory : 数据存储**

PICMICRO微控制器里面被仿真的通用功能寄存器（GPRs）。文件寄存器窗口里显示的是数据存储内容。

**Download : 下载**

就是将数据从PC机发送到另外一个设备里面。比如仿真器，烧写器，目标板等。

**EEPROM : 电可擦除只读存储器**

就是：“***E***lectrically ***E***rasable ***P***rogrammable ***R***ead ***O***nly ***M***emory”，电可擦除只读存储器。一种特殊的PROM，可以用电擦除其内容。每次可以写入或擦除一个字节。EEPROM可以在电源撤消后依然保持数据不丢失。

**Event : 事件**

一种描述总线周期的术语，它可能包括诸如地址，数据，脉冲数目，外部输入，周期类型（取周期，读/写等）以及时间标签（time stamp）等术语。EVENTS通常用来描述触发和断点。

**Export : 导出**

用标准格式从MPLAB IDE 向往外发送数据。

**Extended Microcontroller Mode – 扩展处理器模式**

(只适用于PIC17CXXX 和 PIC18CXXX 系列产品)

在扩展处理器模式下，片上程序存储器和外部扩展存储器都存在。当程序存储器的地址超过了PIC17CXXX 或 PIC18CXXX系列微控制器的内部存储器时候，将自动切换到外部程序存储器。

**External RAM (只使用于PIC17CXXX 和 PIC18CXXX 系列)**

片外读/写存储器。

**File Registers : 文件寄存器**

片内通用和特殊功能寄存器。

**Flash : 闪存存储器**

也是一种EEPROM，其中的数据是按照块来擦写的，而不是按照字节擦写。

**FNOP**

**Forced No Operation**。一个双周期指令的第二个周期，是一个强制NOP指令。因为PICmicro微控制器的结构是双流水线，当微控制器执行当前指令的时候，会同时到物理地址空间里去取下一条指令代码。 然而，假如当前指令改变了程序指针的时候，这一预取指令将会被忽略，同时产生一个NOP周期。

**GPR**

参考数据存储。

**Halt : 暂停**

可以使仿真器停止运行的功能。执行**Halt** 与在断点停下来的功能一样，用户可以观察和改变寄存器的数值，并可以单步运行代码。

**IDE : 集成开发环境**

**Integrated Development Environment** –集成开发环境。支持进行固件开发的有多种功能的应用程序。MPLAB-IDE集成了编译器，汇编器，项目管理器，编辑器，调试器，模拟器，an assortment of other tools within one Windows application。用户不用离开MPLAB IDE 集成开发环境就可以进行编写源代码，编译源代码，调试代码，测试应用程序等工作。



**Import – 导入**

从外部将数据（比如一个十六进制文件）导入到MPLAB IDE集成开发环境里面。

**Make Project – 创建项目**

重新创建一个应用程序，但只重新编译那些自上次创建以来改变过的源文件。

**MCU**

**Microcontroller Unit**。微控制器的缩写。与mC类似。

**Microcontroller - 微控制器**

一种高度集成的芯片。包含了所有组成一个控制器的部件。典型的情况是包含了CPU，RAM，以及一些ROM，I/O口，定时器。和同样包含有以上部件的通用计算机不一样的是，微控制器是设计来满足特殊的应用，比如用来控制一个专用系统。因此微控制器可以省略很多的部件，大大节省产品成本。

**Microcontroller Mode (只适用于PIC17CXXX和PIC18CXXX系列MCU)**

这是PIC17CXXX和PIC18CXXX系列MCU程序存储器设置的一种可能的选择。在微控制器模式下，只允许执行内部程序存储器里的代码，由此可见，微控制器模式下只有片内程序存储器可以供用户使用。

**Microprocessor Mode (只适用于PIC17CXXX和PIC18CXXX系列MCU)**

这是PIC17CXXX和PIC18CXXX系列MCU程序存储器设置的一种可能的选择。在微处理器模式下，不允许执行内部程序存储器里的代码，由此可见，所有的程序存储器都映射到外部存储器里面。

**MPLAB-SIM**

MICROCHIP的集成开发环境MPLAB IDE.里面的模拟器（simulator）。

**MPLAB Software**

主要的可执行文件名，这里包含了集成开发环境里的编辑器，项目管理器，模拟/仿真调试器等。MPLAB IDE集成开发环境软件包驻留在PC机里面。文件名为：“MPLAB.EXE”，它可以调用很多的应用程序。

**MPSIM**

Microchip模拟器的DOS版本文件。MPLAB-SIM是Microchip最新的模拟器。

**MRU**

**Most Recently Used**。指MPLAB IDE集成开发环境里面的下拉菜单里选择的文件和窗口。

**Non Real-Time – 非实时**

指处理器工作于断点或执行单步指令，或者MPLAB IDE处于模拟器状态。

**Node – 节点**

MPLAB IDE集成开发环境的项目部件。

**NOP**

**No Operation** – 空操作。这条指令在执行的时候不执行任何操作，但是程序指针会加一。

**Off-Chip Memory (只适用于PIC17CXXX 和 PIC18CXXX 系列MCU)**

片外程序存储器是指PIC17CXXX 和 PIC18CXXX 系列MCU的存储器选择为在片外目标板上扩展。或者所有的程序存储器由仿真器提供。执行命令：

“**Options > Development Mode**” 可以进行片外存储器类型的选择。

**Pass Counter – 通过计数器**

当执行了某一种操作（比如执行了某一地址处的某一条指令）的时候，这个计数器会自动减一。当“通过计数器” Pass Counter的数值归零的时候，满足事

件要求，你可以在进行复杂的触发动作的时候将“通过计数器” Pass Counter 分配给断点，跟踪逻辑和任何顺序事件。

## **PC**

**Personal Computer** – 个人计算机，或者 **Program Counter** – 程序指针。

## **PC Host**

任何IBM®或兼容个人计算机，运行于Windows 3.1x, Windows 95/98, Windows NT，或者 Windows 2000操作系统之下。MPLAB IDE集成开发软件包运行于486以上的硬件平台。

## **PICmicro MCUs**

PICmicro微控制器。指MICROCHIP的所有微控制器。

## **Program Counter – 程序计数器**

一个可以表示当前程序执行地址的寄存器。

## **Program Memory – 程序存储器**

指存储着PICmicro微控制器的程序代码的存储器。仿真器或模拟器的存储器里面保存着已经下载的应用固件程序。

## **Project – 项目**

指一系列的源文件和指令，组成一个应用系统的项目和可执行代码。

## **Prototype System – 原型系统**

是一个描述用户目标应用或目标板的术语。

## **PWM Signals**

**Pulse Width Modulation Signals** – 脉宽调制信号。一些PICmicro微控制器里面含有PWM外设。

## **Qualifier – 鉴别符**

被“通过计数器” PASS COUNTER使用的一个地址或者一段地址，或者是在一个复杂的触发里面另外一个操作发生之前的事件。

## **Radix**

即数字进制，十六进制或者十进制，用命令：“**Window > Modify**”来指定一个地址或者输入的数据的进制类型。

## **RAM**

**Random Access Memory** – 随机存储器（数据存储器）。

## **Real-Time**

在仿真器或者MPLAB ICD 模式下，当结束暂停状态后处理器就运行于实时模式，表现的好象一颗通常状态下的芯片一样。运行于实时模式的时候，MPLAB ICE的实时跟踪缓冲被允许而且立即捕捉所有被选择的周期并且允许所有的断点逻辑。在仿真器或MPLAB ICD执行于实时模式状态下的时候，只有遇到合法的断点或者用户终止处理器运行的情况下才会暂停下来。

在模拟开发模式下，实时模式意味着其执行速度取决于PC机的CPU运行速度。

## **ROM**

**Read Only Memory**（只读存储器）

## **Run – 运行**

该命令使仿真器从暂停的状态下解脱，允许MCU以实时方式运行应用程序代码，改变I/O状态或显示当前I/O状态。

## **SFR**

**Special Function Registers** – 特殊功能寄存器。

**Simulator – 模拟器**

一个软件应用程序，用来模拟PICmicro系列微处理器的运行。

**Single Step – 单步运行**

该指令用来每次执行一个指令代码。每次执行以后，MPLAB IDE会更新寄存器窗口，观察变量窗口，状态窗口里的内容。以使用户分析和调试程序。

用户可以单步执行C程序代码，但是MPLAB IDE将会执行若干条与这条C语言程序相对应的汇编程序代码。

**Special Function Registers – 特殊功能寄存器**

用来控制处理器的I/O功能，I/O状态，定时器，或其他外设及其模式的寄存器。

**Stack – Hardware – 堆栈-硬件**

在PICmicro MCU内部的一些存储器区域用于存储函数参数，返回数值，局部变量，返回地址。比如针对调用例程操作的“压入”（Push-Down）操作。PICmicro 系列MCU在执行一个CALL操作或响应一个中断，软件会将程序的返回地址压入堆栈。一个调用返回命令会将堆栈里面的地址弹出来并送入程序指针里面。对于PIC18CXXX系列芯片，为适应快速中断还会有硬件堆栈来保存寄存器的值。

**Static RAM or SRAM—静态RAM或SRAM**

**Static Random Access Memory**，静态随机存储器。用户板上不用刷新就可以读/写的存储器。

**Status Bar—状态栏**

状态栏位于MPLAB IDE窗口的下方，在这里显示当前工作信息，包括光标位置，开发模式和处理器类型，以及有效的工具栏。

**Step Into – 单步进入**

这个命令和单步运行指令一样，Step Into（与Step Over相对应）会执行一个CALL指令所调用的子程序里面的指令。

**Step Over – 单步跳过**

该命令允许用户不用进入子程序就可以调试程序代码。当用“Step Over”来调试一个CALL指令的时候，则下一个断点将会设置在CALL指令的后面。假如因为某种原因而使程序进入死循环或者无法正常返回的时候，那么下一处断点将永远无法达到。

“Step Over”命令和“Single Step”命令执行的方式是一样的，区别在于前者处理CALL指令的方式不一样。

**Stimulus—激励**

是一系列数据，用来测试在外部信号激励下系统的反映。这些数据以文本方式将激励信号分别列出来。激励可以是异步信号，同步信号（管脚），时钟和寄存器。

**Stopwatch – 跑表**

一个用来测试系统执行周期数的计数器。

**System Button—系统按钮**

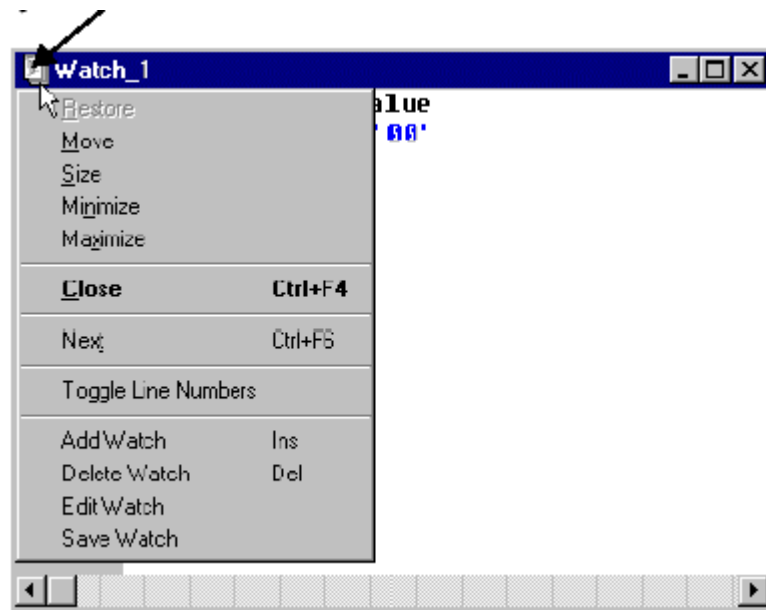
系统按钮“system button”是系统窗口控制的另外一个名称。点击系统按钮将会弹出系统菜单。

**System Window Control—系统窗口控制**

系统窗口控制“system window control”位于窗口的左上方，同时还有一些会话标题。点击这些控制，可以弹出一个窗口，里面有象“Minimize”，

“Maximize”和“Close”之类的选择项目。在一些MPLAB IDE窗口里面，还有更多的模式和功能选择项。

#### 系统窗口控制



图G1：系统窗口控制菜单 – 观察窗口

#### Target – 目标

指的是用户设计的硬件系统。

#### Target Application – 目标应用

位于用户目标板上的固件（Firmware）程序。

#### Target Board – 目标板

组成目标系统的电路和可编程序部件。

#### Target Processor – 目标处理器

被仿真的位于用户目标板上的微控制器。

#### Template – 模板

用户建立的一些将会插入到用户随后的文件里的文本行。MPLAB IDE编辑器将会将这些模板文本保存在模板文件里面。

#### Tool Bar – 工具栏

是一些图标，用户可以点击这些图标来执行MPLAB IDE的功能。

#### Trace – 跟踪

是一个模拟或仿真功能，可以记录程序的执行。仿真会将程序的执行记录到跟踪缓冲器里面，这些数据会被上载到MPLAB IDE 的跟踪窗口里面。

#### Trace Memory – 跟踪存储器

跟踪存储器包含在仿真器里面。通常跟踪存储器也称为跟踪缓冲器。

#### Upload – 上载

上载是将数据从一些开发工具里，比如仿真器，烧写器里面传送到PC机里面，或者从目标板传到仿真器里。

#### Warning – 警告

用来警告用户，指出其操作可能会引起器件的物理损坏，损坏软件文件或设备损坏。

**WatchDog Timer (WDT) – 看门狗计数器**

PICmicro 微控制器里面的一个计数器，它会在一段指定的时间之后将处理器复位。看门狗计数器可以通过修改配置位来进行允许，禁止和设置。

**Watch Variable – 观察变量**

在调试的时候可以通过观察窗口的变量来监视变量的变化。

**Watch Window – 观察窗口**

观察窗口包含一系列的观察变量，这些变量每次断点都会更新数据。

## 术语 – MPASM, MPLINK, MPLIB

**● Absolute Section : 绝对区段**

有固定绝对地址的区段，MPLINK 无法改变其地址。

**● Alpha Character : 字母**

字母表中所有包括大小写的字母 (a,b,...,z,A,B...,Z)。

**● Alphanumeric : 字符**

阿拉伯数字 (0,1,...,9)

**● Application : 应用**

用户开发的一系列软件和硬件，通常是用 PICmicro 控制的产品。

**● Assemble : 汇编**

用 MPASM 对源程序进行汇编，生成目标代码的过程。

**● Assembler Source Code : 汇编源文件**

由汇编器处理的文本文件，使汇编指令和机器码之间一一对应。

**● Assigned Section : 已分配区段**

在链接器命令文件里被分配了目标存储器的区段。链接器指定一块已分配区段到指定的存储器块里。

**● Build : 创建**

将应用系统的所有源文件重新编译。

**● COFF(Common Object File Format) : 共用目标文件格式**

共用目标文件格式 - 一种目标/可执行文件的格式。用于 MPASM 或 MPLAB-C17/C18 生成的可重定位目标文件。

**● Command Line Interface : 命令行接口**

命令行界面用于执行带参数的程序。带命令行参数执行 MPASM 或者直接执行 MPASM 都将调用汇编器。如果缺少命令行参数，则会有输入界面提示用户输入相应的参数值。

**● Control Directives : 控制指示符**

控制指示符允许条件汇编代码。

**● Data Directives : 数据指示符**

数据指示符控制存储器的分配，提供数据和符号之间的对应，即：用具有含义的符号代表数据。

**● Data RAM : 数据 RAM**

仿真 PICmicro 中的通用 RAM 文件存储器。文件寄存器窗口显示数据 RAM 的

值。

- **Directives : 指示符**

指示符控制汇编器的操作过程，它指示 MPASM 如何处理助记符，定义数据，如何格式化输出列表文件。指示符使得代码更容易理解，而且为特殊的需求提供用户自定义的输出。

- **Expressions : 表达式**

表达式用于源文件中的操作数区域，它可以包含常量，符号，或者任何用算术运算符组合起来的常量和符号的组合。每个常量或符号都可以冠以一个正号或负号，以表示其正负性质。

**注意：**表达式被用 32 位整型求值运算（目前尚不支持浮点运算）。

- **External Linkage : 外部链接**

如果在其他模块定义了可以操作一个函数或变量，则称它具有外部连接的功能。

- **External Symbol : 外部符号**

给有外部连接功能的变量或函数定义的符号。

- **External Symbol Definition : 外部符号定义**

在当前模块定义的变量或函数的外部符号。

- **External Symbol Reference : 外部符号参考**

一个外部符号，它指向一个在其他模块定义的函数或变量。

- **External Symbol Resolution : 外部符号判断**

链接器的一个处理过程。在此过程中，来自所有输入模块的外部符号定义都试图去更新所有外部符号参考。任何没有得到更新的外部符号参考将引起链接错误报告。

- **Hex Code : 十六进制代码**

可执行的 16 进制指令代码。是由源文件通过汇编器或编译器的处理生成的。16 进制代码可以直接转化为目标代码。它包含在一个 16 进制文件中。

- **Hex File : 十六进制文件**

一个 ASCII 文件。它包含 16 进制地址和代码值，便于编程器使用。该格式的文件可以由编程器读出。

- **Identifier : 定义符**

一个函数或变量的名称。

- **Initialized Data : 已初始化数据**

数据被初始化为特定的值。在 C 语言里，`int myVar=5` 这条语句定义了一个变量，该变量将驻留在一个已初始化的区段里。

- **Internal Linkage : 内部链接**

不能从其他模块访问一个在该模块定义的函数或变量，则称函数或变量内部可连接。

- **Library : 库**

库是一些可重定位的目标模块的集合。它由多个源文件经汇编生成目标文件，然后使用库管理程序将这些目标模块和库文件链接起来。一个库文件可以和目标模块和其他库链接，生成可执行代码。MICROCHIP 的库管理程序 MPLIB 可以建立库并维护库。

- **Link : 链接**

将目标模块和库文件链接起来，生成可执行代码的过程。链接工作有 MICROCHIP 的链接器 MPLINK 完成。

- **Listing Directives : 列表指示符**

列表指示符是用来控制 MPASM 列表文件的格式。它们允许指定标题、分页及其他控制。

- **Listing File : 列表文件**

列表文件是一个 ASCII 格式的文件，它显示出源文件中每条汇编指令、MPASM 指示符、宏语言和机器码之间的对应情况。

- **Local Label : 局部标号**

局部标号是用 LOCAL 指示符定义的标号。这些标号在宏指令的某一特定的时间内存在，换言之：定义为局部性质的标号和符号在执行完 ENDM 宏指令后就从符号表里撤出了。

- **Macro : 宏**

宏是一系列汇编指令的集合。当在源代码中遇到宏时，这些指令便会插入到汇编序列里。宏在使用前必须定义，宏不允许前向引用。

所有跟在 MACRO 指示符后的参数都是宏定义的一部分，宏里使用的标号必须是局部类型，

- **Macro Directives : 宏指示符**

这些指示符在宏体内控制执行和数据的分配。

- **Make Project : 创建项目**

一条命令，它控制重新创建 (REBUILD) 一个应用程序，只编译那些自上次建立以来曾经改变过的源文件。

- **Mnemonics : 助记符**

这些指令被直接转化为机器码。对 MCU 内部的 RAM 或 ROM 里的数据执行算术或逻辑运算。它们还能把数据从寄存器和存储器里放入、取出，也能改变程序执行流程。也称为 OPCODES。

- **MPASM**

MICROCHIP 可重定位宏汇编器。

- **MPLAB**

主要的可执行程序，支持集成调试环境 (IDE)，内涵文本编辑器 (EDITOR)、项目管理器、仿真/模拟调试器。MPLAB 软件驻留在用户的 PC 主机里。可执行文件名是 MPLAB.EXE，它可以调用许多其他的文件。

- **MPLAB-C17/C18**

MICROCHIP 为 PIC17CXX 和 PIC18CXX 设计的 C 编译器。

- **MPLIB**

和 MPLINK 链接程序一起使用的 MICROCHIP 库文件管理器。

- **MPLINK**

MICROCHIP 链接器。

- **Nesting Depth : 嵌套深度**

宏嵌套深度可达 16 级。

- **Node : 节点**

MPLAB 的项目元件。

- **Object Code : 目标代码**

汇编器或编译器对源程序处理后生成的机器码。该代码驻留在 PICmicro 的存储器里面，执行用户的应用程序。可重定位代码是由 MPASM 或 MPLAB-C17/C18 生成，通过 MPLINK 链接程序后生成可执行代码。目标代码放置在目标文件中。

- **Object File** : 目标文件

包含可重定位代码或数据及外部代码和数据的参考目标模块。多个目标模块可以链接为一个可执行文件。当生成目标文件时，源代码里需要有特殊的指示符。目标代码放置在目标文件中。

- **Object File Directives** : 目标文件指示符号

只有生成目标文件时，才使用的指示符。

- **Operators** : 操作符号

运算符号是一系列形如加号“+”、减号“-”的符号，用以组成运算表达式。每个运算符号都有自己的优先级。

- **PC 翻译: 个人计算机**

任何 IBM 及兼容的计算机。MPLAB 要求 486 以上的计算机。

- **PC Host** : 主机

计算机运行于 WINDOWS3.X 或 WINDOWS95/98 环境。

- **PICmicro**

PICmicro 指 MICROCHIP 的 PIC12CXX、PIC14000、PIC16CXX、PIC17CXX、PIC18CXX 系列微控制器。

- **Precedence** : 优先级

优先级指表达式里一些部件比另外一些先进行求值运算。相同优先级别的运算符其运算顺序由左到右。

- **Program Memory** : 程序存储器

下载了目标系统支持软件的仿真器或者模拟器的存储器。

- **Project** : 项目

一系列源文件和指令，为一个应用系统建立目标代码。

- **Radix** : 进制

进制基数是汇编器求值的时候使用的基本计数机制。默认的进制基数是 16 进制。用户可以使用基数操作符改变默认进制基数。

- **RAM** : 随机存储器

随机存取存储器 (RANDOM ACCESS MEMORY)，即数据存储器。

- **Raw Data** : 原始数据

与一个区段相关的二进制代码或数据。

- **Recursion** : 递归

一个已经定义的宏可以调用它自己。当书写递归宏的时候应当特别留意，很容易导致无穷循环而无法退出。

- **Relocatable Section** : 可重定位区段

地址还未最后确定的区段。链接器通过重定位过程给可重定位区段分配地址。

- **Relocation** : 重定位

链接器的处理过程，在此过程中，地址被分配给可重定位区段；所有可重定位区段里的标识符定义 (identifier symbol definitions) 被更新为新的地址。

- **ROM** : 只读存储器

只读存储器 (READ ONLY MEMORY)。

- **Script** : 描述 (脚本)

链接器描述文件是 MPLINK 的控制文件。

- **Section** : 区段

具有名称、大小和地址的一些数据或代码的集合。



- **Shared Section : 共享区段**

驻留在共享（未分堆）数据 RAM 里的区段。

- **Shell : 接口界面**

MPASM SHELL 是提供给宏汇编器的输入接口界面。有两种界面类型：一种是 DOS 界面，另一种是 WINDOWS 界面。

- **Software Stack : 软件堆栈**

MPLAB-C17/C18 的软件堆栈。

- **Source Code : 源代码**

源代码是由将会翻译成机器码的 PICmicro 指令系统、MPASM 指示符和宏组成。这种代码适应于 PICmicro 系列 MCU 或者像 MPLAB 之类的 MICROCHIP 开发系统。

- **Source File : 源文件**

是由将会翻译成机器码的 PICmicro 指令系统、MPASM 指示符和宏的 ASCII 文本文件的源代码。它可以用任何 ASCII 文本编辑器进行编辑。

- **Stack : 堆栈**

是数据存储器里的一块区域。用以存储函数参数、返回值、局部变量和返回地址。

- **Symbol : 符号**

用来描述程序里的各种“部件”的通用方法。这些“部件”包括：函数名、变量名、区段名、文件名、结构名等。

- **Unassigned Section : 未分配区段**

在链接器命令文件里没有分配特定的目标存储器块的区段。链接器必须找到一个目标存储器块，在这里安排未定义区段。

- **Uninitialized Data : 未初始化数据**

没有初始化赋值的数据。在 C 语言里，`<int myVar;>` 语句的含义是：定义一个位于未初始化数据区段的变量。

## 术语 - MPLAB-CXX

### C

一种高级可编程语言，可以用来生成 PICmicro 系列微控制器特别是高端处理器的机器代码。

### Compile – 编译

编译器所做的工作，见编译器的解释。

### Compiler – 编译器

一种语言工具，可以将 C 语言源程序翻译为机器代码。MPLAB-C17 和 MPLAB-C18 是 Microchip 为 PIC17CXXX 系列和 PIC18CXXX 系列处理器开发的 C 编译工具。

### High Level Language – 高级语言

一种用来编写程序的，比汇编语言更概括和抽象的高级语言。高级语言（比如 C 语言）可以使用编译器源程序翻译为机器指令代码。

### Initialized Data – 初始化数据

预先被赋予数值的数据。在 C 语言里，`int myVar=5` 意味着定义了一个驻留在初始化数据区段里的变量。

### Memory Models – 存储器模块

与某一型号微控制器（具有一定存储器 RAM 或 ROM 空间和结构）相对应的库文件和（或）预编译目标文件。

### MPLAB-CXX

指 MPLAB-C17 和 MPLAB-C18 C 编译器。

### Source Code – C

用高级语言“C 语言”编写的程序，该程序可以用 C 编译器编译为 PICmicro 系列微控制器的机器代码。机器代码可以为 PICmicro 系列微控制器直接使用，也可以用来在 MPLAB IDE 开发系统里运行。

### Source File – C

在一个 ASCII 文本格式的 C 源文件里的程序可以用编译器将其转换为机器代码。该文件可以用 ASCII 文本编辑器进行编辑。

### Stack – Software: 堆栈 - 软件

编译器使用软件堆栈来保存局部变量和函数的入口参数以及返回参数

### Uninitialized Data: 未初始化数据

没有初始化数值的数据，在 C 语言里，语句“`int myVar`”表示定义一个驻留在未初始化数据区段里的变量。

## 术语 – MPLAB-ICE, PICMASTER 仿真器

### Break Point – Hardware : 断点 – 硬件

一个将会导致处理器暂停运行的事件。

### Emulation : 仿真

在仿真器存储器里运行仿真软件，仿佛是在开发着的处理器上运行固件（firmware）一样。

### Emulation Memory : 仿真存储器

仿真器里的程序存储器。

**Emulator : 仿真器**

执行仿真功能的硬件。

**Emulator System : 仿真器系统**

MPLAB IDE 仿真系统包括 POD，处理器模块，芯片适配器以及 MPLAB IDE 集成开发软件包。PICMASTER 仿真器系统包括 POD，针对不同芯片的仿真头，电缆和 MPLAB 软件包。

**External Input Line (MPLAB-ICE only) : 外部输入线（只适合于 ICE）**

外部输入信号逻辑探头（TRIGIN），可以设置基于外部信号的事件。

**ICE : 在线仿真器**

**In-Circuit Emulator**，在线仿真器。是 MICROCHIP 的在线仿真器在 MPLAB IDE 集成开发环境下运行。

**Logic Probes : 逻辑探头**

可达 14 个逻辑探头连接到仿真器。逻辑探头可以提供外部跟踪输入，触发输出信号，+5V 电源，公共地。

**MPLAB-ICE**

MICROCHIP 的工作于 MPLAB IDE 集成开发环境软件包下的在线仿真器。

**PICMASTER Emulator**

一个硬件仿真器。可以用来仿真和调试固件应用程序。该装置包括仿真存储器，断点逻辑，计数器，定时器以及一些工具的跟踪分析仪。MPLAB ICE 是最新的 MICROCHIP 仿真器。

**Pod**

外部仿真器盒包括仿真存储器，跟踪存储器，事件和周期定时器以及跟踪/断点逻辑。通常是 MPLAB ICE 仿真器的缩写。

**Power-on-Reset Emulation : 上电复位仿真器**

一个软件随机处理过程，在数据 RAM 区域里填入随机数据，用来模拟上电时候 RAM 中的未初始化数据。

**Skew**

指令运行的信息表现为在不同时间里处理器总线上的时序。例如，在总线上执行的操作代码是在上一个代码在执行的时候取出来的。当操作代码在执行的时候，源数据地址和数值以及目标数据地址就会出现，当下一个指令执行的时候，目标数据将会出现。跟踪缓冲器会捕捉某时刻总线上的信息。因此，一个缓冲条目里包含三个指令的执行信息。从指令执行的一条信息到另外一条信息里捕捉的周期数，称为“SKEW”。

**Skid**

当硬件断点被设置来暂停处理器的运行，在处理器停止运行之前可以执行一个或者多个附加的指令。在预先设置的断点处暂停之后执行的附加指令数可以叫做“SKID”。

**Trigger Output : 触发输出**

触发器输出是指可以在任何地址或地址范围生成，并独立于跟踪点和断点的仿真器输出信号。触发输出点可以设置为任意数字。

**MPLAB-ICD**

**ICD**

在线调试器（**In-Circuit Debugger**）。MPLAB-ICD 是 MICROCHIP 为 16F87X 系列处理器开发的在线调试器。MPLAB-ICD 嵌入 MPLAB-ICE 集成开发软件包里面。

**MPLAB-ICD**

MPLAB-ICD 是 MICROCHIP 为 16F87X 系列处理器开发的在线调试器。MPLAB-ICD 嵌入 MPLAB-ICE 集成开发软件包里面。和 MPLAB-IDE 一起工作。该系统硬件由模块，仿真头，DEMO 板（可选），电缆和 MPLAB 软件组成。

**术语 – PICSTART PLUS, PRO MATE****PICSTART Plus**

为 Microchip 的 PICmicro 系列 8 脚，14 脚，28 脚，40 脚单片机开发的烧写器。必须和 MPLAB 软件包一起使用。

**Programmer : 烧写器**

一种可以为可程序的半导体器件，比如微控制器烧写资料的电子设备。

**PRO MATE**

为 Microchip 的 PICmicro 系列 8 脚，14 脚，28 脚，40 脚单片机和 KEELOQ 跳码芯片开发的烧写器。可以和 MPLAB 软件包一起使用，也可以单独使用。