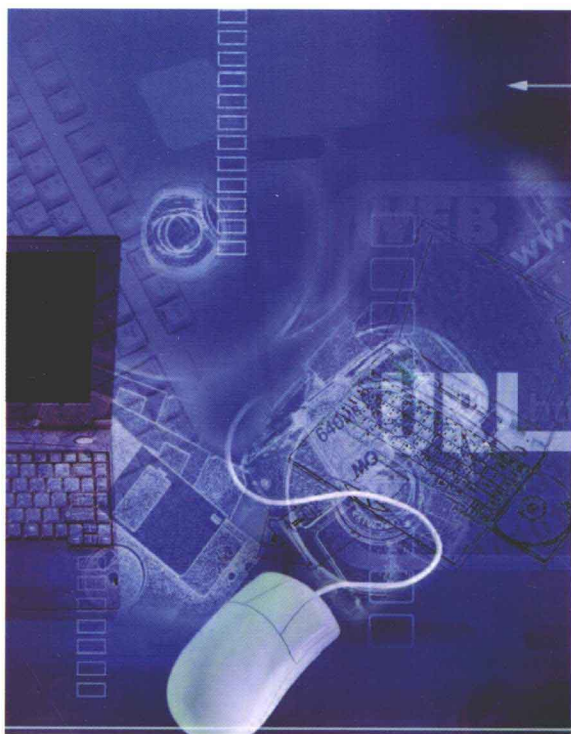


ARM7嵌入式系统 实践教程

- ◆ ADS1.2开发环境
- ◆ 混合编程技术
- ◆ 串行接口技术
- ◆ GPIO接口技术
- ◆ 定时器技术
- ◆ 外部中断技术
- ◆ A/D转换技术
- ◆ LCD显示技术
- ◆ 触摸屏技术
- ◆ μ CLinux嵌入式操作系统
- ◆ 无线通信技术
- ◆ GPRS技术



高等学校计算机应用规划教材

ARM7 嵌入式系统实践教程

田志忠 赵宏伟 李树秋 编著

清华大学出版社

北 京

内 容 简 介

本书以 MagicARM2200 实验平台为基础,以 PHILIPS 公司嵌入式处理器 LPC2290 为核心,以 ADS1.2 为集成开发环境,以 $\mu\text{C}/\text{OS-II}$ 操作系统和 μCLinux 操作系统及各种中间件为软件平台,构建一个 ARM 嵌入式系统实验教学体系。本书共 12 个实验项目,包括基础性实验、提高性实验和设计性实验。实验内容涉及程序设计、 $\mu\text{C}/\text{OS-II}$ 操作系统环境搭建与应用、 μCLinux 操作系统环境搭建与应用以及基于嵌入式处理器 LPC2290 的键盘、数码管、步进电机、触摸屏、GPRS 通信等接口设计与应用。

本书对应的电子教案和实例源文件可以到 <http://www.tupwk.com.cn/downpage> 网站下载。

本书封面贴有清华大学出版社防伪标签,无标签者不得销售。

版权所有,侵权必究。侵权举报电话:010-62782989 13701121933

图书在版编目(CIP)数据

ARM7 嵌入式系统实践教程 / 田志忠, 赵宏伟, 李树秋 编著. —北京: 清华大学出版社, 2014
(高等学校计算机应用规划教材)

ISBN 978-7-302-34744-6

I. ①A… II. ①田… ②赵… ③李… III. 微处理器—系统设计—高等学校—教材 IV. ①TP332

中国版本图书馆 CIP 数据核字(2013)第 298466 号



责任编辑: 胡辰浩 袁建华

装帧设计: 思创景点

责任校对: 曹 阳

责任印制: 李红英

出版发行: 清华大学出版社

网 址: <http://www.tup.com.cn>, <http://www.wqbook.com>

地 址: 北京清华大学学研大厦 A 座 邮 编: 100084

社 总 机: 010-62770175 邮 购: 010-62786544

投稿与读者服务: 010-62776969, c-service@tup.tsinghua.edu.cn

质 量 反 馈: 010-62772015, zhiliang@tup.tsinghua.edu.cn

印 刷 者: 北京富博印刷有限公司

装 订 者: 北京市密云县京文制本装订厂

经 销: 全国新华书店

开 本: 185mm×260mm	印 张: 17.25	字 数: 427 千字
版 次: 2014 年 1 月第 1 版		印 次: 2014 年 1 月第 1 次印刷
印 数: 1~3000		
定 价: 32.00 元		

前 言

本书是针对嵌入式开发工具 ADS1.2 和 MagicARM2200 嵌入式教学平台的实践教材。考虑到方便学生嵌入式实践和日新月异的嵌入式系统创新设计,我们编写了这本教材。在教材编写中,我们选取了一些平台配套教材中比较典型的实例作为基础实践题目,另外设计了一些提高性实践题目,尤其以大学本科生研究项目为基础,设计了综合性实践题目和开发性实践题目,为学习嵌入式系统创新设计提供了思路和实例。

本书内容包括 4 个部分,第一部分讲述 ADS1.2 开发工具的使用方法;第二部分以 MagicARM2200 教学实验平台为基础,以 PHILIPS 公司 ARM7TDMI-S 微控制器 LPC2290 为核心,以 μ C/OS-II 操作系统和 μ CLinux 操作系统及各种中间件为软件平台,构建一个 ARM 嵌入式系统实践教学体系,包括基于 ARM 技术的键盘、数码管、步进电机、触摸屏、无线通信、GPRS 通信等嵌入式系统的开发实例;第三部分给出了一些嵌入式练习题目;第四部分讲述 ADS1.2 使用过程中经常出现的问题和解决方法。

本书在编写过程中,嵌入式系统课程梯队的张仲明、臧雪柏、刘萍萍、裴士辉、任凤娟、卢欣华、江明等教师做了大量的工作,为本书的顺利出版提供了强有力的支持和帮助,在此,向这些老师表示衷心感谢。同时,还要感谢嵌入式系统实验室张仲明等老师提供的良好实验环境和资源,感谢吉林大学计算机科学与技术学院和吉林大学软件学院教务办公室的老师和领导为本教材顺利出版提出的建议和作出的努力。

本书编写过程中参考了有关的优秀教材、专著、应用成果,以及优秀的网络站点,恕不一一列举。能够领略众多新颖的观点和技术,是原创者的无私贡献,是读者的集粹之想。本书编者在此向提供各种观点和技术的各位编著者表示最真诚的谢意。

我们对在本书编写、出版过程中给予支持和帮助的所有领导及朋友们,一并表示衷心的感谢。

由于编著者水平和经验所限,书中不足之处在所难免,恳请读者批评指正。我们的电话是 010-62796045,邮箱是 huchenhao@263.net。

编 者

2013 年 7 月于长春

目 录

第 1 章	ADS1.2 集成开发环境	1
1.1	ADS1.2 简介	1
1.2	CodeWarrior IDE	2
1.2.1	文件管理	2
1.2.2	工程管理	6
1.2.3	编译器	13
1.2.4	链接器	20
1.2.5	文本编辑器	22
1.3	AxD Debugger	24
1.3.1	启动与关闭 AxD Debugger	25
1.3.2	与调试有关的概念	25
1.3.3	镜像文件	26
1.3.4	调试辅助工具	27
1.3.5	调试参数	34
1.4	简单应用实例	38
1.4.1	创建工程	38
1.4.2	创建源程序文件	39
1.4.3	配置编译器、链接器参数	42
1.4.4	编译、修改程序	43
1.4.5	链接程序	45
1.4.6	调用调试器、配置调试参数	45
1.4.7	调试方案	46
1.4.8	调试程序	48
第 2 章	实验项目	53
2.1	编程实验	53
2.1.1	汇编语言编程实验	53
2.1.2	C 语言编程实验	56
2.1.3	混合编程实验	61
2.2	UART 实验	68
2.2.1	UART0 数据发送实验	68
2.2.2	UART0 中断接收实验	86
2.2.3	串行通信协议实验	104
2.3	GPIO 实验	106
2.3.1	蜂鸣器实验	106
2.3.2	LED 显示与按键识别实验	114
2.3.3	计算机控制 LED 检测实验	122
2.4	外部中断与定时器实验	125
2.4.1	外部中断实验	125
2.4.2	定时器实验	132
2.4.3	计算机控制 KEY1 按键识别实验	140
2.5	模数转换实验	142
2.5.1	直流电压测量实验	142
2.5.2	多路直流电压测量实验	157
2.5.3	计算机控制蜂鸣器检测实验	161
2.6	LCD 显示实验	163
2.6.1	LCD 绘图实验	163
2.6.2	LCD 汉字显示实验	175
2.6.3	LCD 汉字串显示实验	183
2.7	触摸屏实验	184
2.7.1	触摸屏屏幕定位实验	184
2.7.2	按钮触摸实验	192
2.8	μCLinux 基础实验	202
2.9	模拟量强度表现实验	207
2.10	键盘综合控制实验	211
2.11	无线通信实验	219
2.12	GPRS 通信实验	234
第 3 章	附加实验	257
第 4 章	常见错误分析与处理	263
	参考文献	269

第1章 ADS1.2集成开发环境

1.1 ADS1.2 简介

ADS(ARM Developer Suite)是一套嵌入式开发工具，它是由英国 ARM 公司提供的面向 ARM 处理器的嵌入式通用开发软件，目前最常用的版本是 ADS1.2，它支持的嵌入式处理器最高为 ARM10 系列。

使用 ADS1.2 可进行软件编程调试和硬件仿真调试。ADS1.2 采用的是混合编程技术，它在一个工程内可包含 C、C++、汇编等多种类型的源程序文件。使熟悉不同编程语言的技术人员都能参与到嵌入式系统的开发，有效地提高了技术人员的利用率和嵌入式系统的开发效率。

ADS1.2 可运行在 Windows 或 Linux 操作系统上。它由 CodeWarrior for ARM Developer Suite(Code Warrior IDE)和 AxD Debugger 两个软件组成，包括以下几个部分。

1. 编译器

ADS1.2 包括 5 个编译器，由 CodeWarrior IDE 统一管理调用。

- (1) 汇编编译器，用于编译基于 ARM/Thumb 指令的汇编源程序；
- (2) ARM C 编译器，用于编译基于 ARM 32 位指令的 C 源程序；
- (3) ARM C++编译器，用于编译基于 ARM 32 位指令的 C++源程序；
- (4) Thumb C 编译器，用于编译基于 Thumb 16 位指令的 C 源程序；
- (5) Thumb C++编译器，用于编译基于 Thumb 16 位指令的 C++源程序。

2. 链接器

ADS1.2 包括 1 个链接器，用于将目标文件和相关库链接生成可执行的 ELF 镜像文件，由 CodeWarrior IDE 管理调用。

3. Code Warrior IDE

Code Warrior IDE 就是 ADS1.2 的集成开发环境，它是一个基于窗口的管理工具，常用功能包括工程的创建与管理；源程序的创建、编辑、编译、链接管理等。

4. AxD Debugger

AxD Debugger 是 ADS1.2 的调试软件，用于提供仿真调试功能。

5. 指令模拟器

指令模拟器指的是 ARMulator 软件，它是一个 Window 应用程序，为基于 ARM 32 位和 Thumb 16 位代码的程序提供仿真调试环境。

6. ARM 库

包括 C、C++ 函数库，供用户程序调用。

1.2 CodeWarrior IDE

本节将讲述利用 CodeWarrior IDE 集成开发软件创建工程、创建源程序文件、编辑程序、编译程序、创建可执行镜像文件的操作方法。

CodeWarrior IDE 为集成开发环境，它通过提供简单、直观、灵活的图形界面来管理工程。一个工程实际就是一些文件的集合，其中包括源程序文件、库文件、目标文件、镜像文件等。CodeWarrior IDE 是一个 Window 应用程序，在桌面上通过双击其图标即可启动该软件，其主窗口如图 1.2.1 所示。

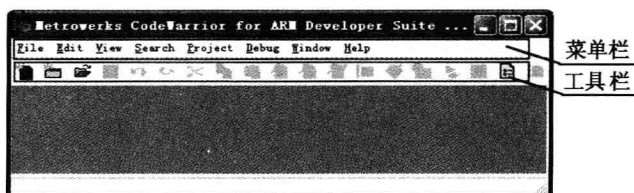


图 1.2.1 CodeWarrior IDE 主窗口

CodeWarrior IDE 主窗口为工程实施提供了很多工具和方法，包括菜单、工具栏以及管理窗口等，使用这些工具就可以有效地管理工程。下面将详细介绍管理工具和实施方法。

1.2.1 文件管理

CodeWarrior IDE 文件管理包括文件的创建、打开、关闭、存储和打印等操作。常用的文件操作如下。

1. 创建文件

(1) 文件创建命令

调用文件创建命令，会执行文件创建操作。常用的文件创建命令有以下几种。

- ① 菜单：在 CodeWarrior IDE 主窗口中，菜单 File 下提供了创建文件的命令 New；
- ② 按钮：在 CodeWarrior IDE 主窗口中，工具栏上提供了创建文件按钮；
- ③ 快捷键：CodeWarrior IDE 提供了创建文件的快捷键命令 Ctrl+Shift+N。

(2) 使用 CodeWarrior IDE 可创建的文件类型

经常使用 CodeWarrior IDE 创建工程文件、源程序文件、头文件等，这些文件的类型如表 1.2.1 所示。

表 1.2.1 使用 CodeWarrior IDE 可创建的文件类型

文 件	文 件 类 型	文件扩展名
汇编源文件	文本文件	*.s
C 语言源文件	文本文件	*.c
C++源文件	文本文件	*.cpp
头文件	文本文件	*.h
工程文件	二进制文件	*.mcp

(3) Build Targets(生成目标)

对源程序文件进行编译时,会使用生成目标创建输出文件,一种生成目标用于创建一种类型的输出文件。默认情况下有以下 3 种生成目标。

① Debug

依据 Debug 创建的目标文件和可执行的 ELF 镜像文件包含全部的调试信息,此生成目标适用于新工程的开发。

② Release

依据 Release 创建的目标文件和可执行的 ELF 镜像文件不包含任何调试信息,此生成目标适用于产品发布。

③ DebugRel

依据 DebugRel 创建的目标文件和可执行的 ELF 镜像文件包含部分调试信息,此生成目标适用于具有多个版本的工程开发。

(4) 创建文件的操作过程

我们以 main.c 文件为例,讲述文件的创建过程。

① 显示文件创建选项卡

选择 File | New 命令,打开 New 对话框。在 New 对话框中单击 File 标签,打开 File 选项卡(即文件创建选项卡),如图 1.2.2 所示。



图 1.2.2 文件创建窗口

② 填写文件创建信息

a. 文件名

将源程序文件命名为 main.c, 在 File name 项的编辑框内输入文件名 main.c。

b. 文件存储路径

在 Location 项的编辑框内输入文件存储路径 F:\ARM_Exp\Exp, 即工程存储路径。

c. 将文件加入工程

首先, 选中 Add to Project 复选框, 使其处于选中状态; 然后, 单击 Project 项的  按钮, 显示下拉列表框, 从中选择文件要加入的工程。默认情况下, Project 项显示的就是当前文件要加入的工程, 如 Exp.mcp, 所以一般不需再执行工程选择操作。

d. 生成目标选择

在 Targets 项的列表框内, 单击 Debug 复选框, 使其处于选中状态; 单击 DebugRel 复选框, 使其处于选中状态; 单击 Release 复选框, 使其处于选中状态, 即选中所有的生成目标。

③ 单击“确定”按钮, 执行 main.c 文件创建操作。操作的结果是在工程管理窗口中增加文件名 main.c, 同时打开一个文本编辑器编辑 main.c。

至此, 文件创建结束。随后就可以使用文件编辑器对 main.c 文件进行代码录入、程序修改和文件保存等操作。

2. 打开文件

使用 CodeWarrior IDE 可以打开多种类型文件, 比如工程文件、文本文件、CW 文件和 XML 文件等。

(1) 使用 CodeWarrior IDE 可以打开的文件类型如表 1.2.2 所示。

表 1.2.2 使用 CodeWarrior IDE 可打开的文件类型

序 号	文 件 后 缀	文 件 类 型
1	*.txt	文本文件
2	*.mcp	工程文件
3	*.cat , *.ctlg, *.elf, *.eld, *.pch++, *.axp	CW 文件
4	*.xml	XML 文件
5	*.Exe	可执行文件
6	*.dll , *.lib	库文件
7	*.gSYM , *.iSYM	符号文件
8	*.c, *.cp, *.cpp, *.c++, *.cc, *.cxx, *.h, *.hpp, *.h++, *.hh,	C/C++文件
9	*.c, *.cp, *.cpp, *.c++, *.cc, *.cxx	C/C++文件
10	*.h, *.hpp, *.h++, *.hh, *.hxx	C/C++头文件
11	*.Java	Java 文件

(2) 文件打开命令

调用文件打开命令, 执行文件打开操作。常用打开文件的操作命令有以下几种。

- ① 菜单: 在 CodeWarrior IDE 主窗口中, 菜单 File 下提供了打开文件的命令 Open。
- ② 按钮: 在 CodeWarrior IDE 主窗口中, 工具栏上提供了打开文件的  按钮。
- ③ 快捷键: CodeWarrior IDE 提供了打开文件的快捷键命令 Ctrl+O。

(3) 打开文件的操作方法 1

使用菜单可打开工程文件、源程序文件等。以 main.c 文件为例，说明使用菜单打开文件的操作过程。

① 显示文件“打开”对话框

选择 File | Open 命令，显示“打开”对话框，如图 1.2.3 所示。

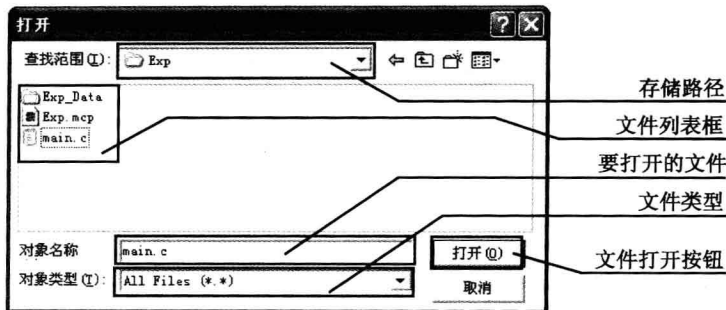


图 1.2.3 文件打开窗口

② 选择文件类型

单击“对象类型”项的 ▾ 按钮，显示下拉列表框。下拉列表框内显示了所有的文件类型，从中选择要打开的文件类型，例如 C/C++ Files (*.c, *.cp, *.cpp...)。

③ 配置文件路径

单击“查找范围”项的 ▾ 按钮，显示下拉列表框。单击下拉列表框的 ▾ 按钮，从下拉列表框内选择文件存储路径 F:\ARM_Exp\Exp，则与指定文件类型相符的所有文件和子目录都会显示在文件列表框内，包括文件 main.c。

④ 选中文件

在文件列表框内，单击要打开的 main.c 文件，文件名 main.c 会显示在“对象名称”栏内。

⑤ 文件打开

单击“打开”按钮，执行 main.c 文件打开操作。操作的结果是打开一个文本编辑器编辑文件 main.c，如图 1.2.4 所示。随后就可以使用文本编辑器对文件 main.c 进行代码录入、程序修改和文件保存等操作。

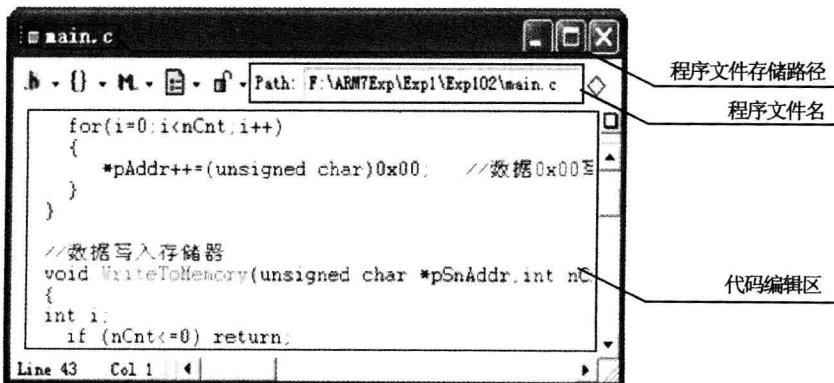


图 1.2.4 代码编辑窗口

(4) 打开文件的操作方法 2

在工程管理窗口内，经常需要打开源程序文件和与程序有关的其他文本文件，具体操作过程如下。

① 显示工程管理窗口

在 CodeWarrior IDE 主窗口中，选择 File | Open Recent | Exp.mcp 命令，执行工程打开操作。操作的结果是打开工程 Exp.mcp 并显示在工程管理窗口内，如图 1.2.5 所示。

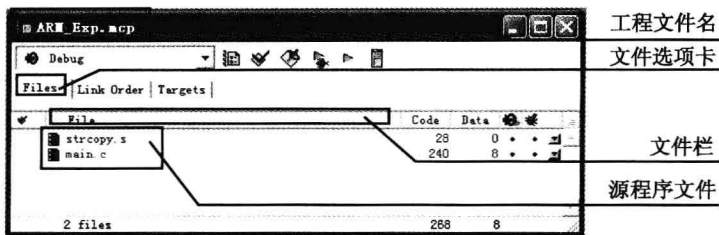


图 1.2.5 工程管理窗口

② 打开文件

从工程管理窗口中打开文件有两种操作方法，一种方法是双击文件名，打开文件；另一种方法是使用鼠标左键或使用 ↑ 键、↓ 键选中文件，然后按回车键打开文件。

以打开 main.c 文件为例，说明文件打开过程。

参考图 1.2.5，在工程管理窗口中，双击文件名 main.c，执行文件打开操作。操作的结果是打开文本编辑器编辑文件 main.c，其中，标题栏显示文件名 main.c，编辑区显示的 main.c 的内容，如图 1.2.4 所示。

3. 保存文件

在文本编辑器编辑窗口内，执行代码录入和修改操作后，需要及时保存文件。通常有两种保存文件的操作方法：一是选择 File | Save 命令或按快捷键 Ctrl+S 或单击  按钮，执行文件保存操作；二是选择 File | Save All 命令或按快捷键 Ctrl+Shift+S，执行文件保存操作。

4. 打开最近操作过的文件

工程开发过程中，经常需要使用最近操作过的工程或源程序文件。CodeWarrior IDE 提供了一种方法，使用这种方法可以快速打开最近操作过的工程或源程序文件。

CodeWarrior IDE 将最近操作过的文本文件和工程文件以菜单的方式挂接在菜单 Open Recent 下，作为其子菜单。选择 File | Open Recent | “文件名”命令，就可以打开指定文件。需要说明的是：打开文本文件时，使用文本编辑器编辑文件；打开工程文件时，使用工程管理窗口显示工程。

1.2.2 工程管理

CodeWarrior IDE 依据工程来管理所有的嵌入式项目。工程管理主要包括创建工程、打开工程、向工程添加文件、保存工程、从工程中删除文件和打开最近操作过的工程等。

1. 工程管理窗口

CodeWarrior IDE 是基于窗口的软件平台, 依据窗口来管理工程, 工程管理窗口如图 1.2.6 所示。

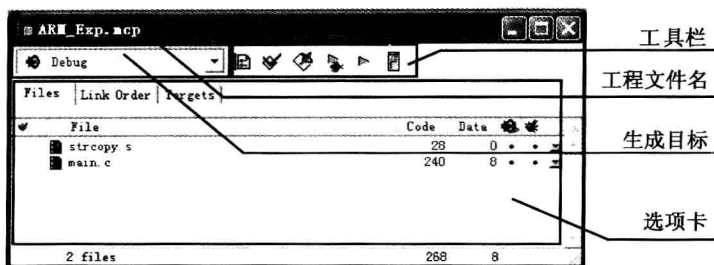


图 1.2.6 工程管理窗口

工程管理窗口由标题栏、生成目标栏、工具栏和选项卡 4 部分组成。

(1) 标题栏

用于显示当前正在管理的工程文件名。

(2) 生成目标栏

此栏用于指定一生成目标作为当前“生成目标”。当对工程内的源程序文件进行编译操作时, 使用此生成目标创建输出文件, 即目标文件。一个生成目标对应一种类型的目标文件, 默认情况下有 3 种生成目标, 分别是 Debug、DebugRel 和 Release。一般根据项目的进展情况, 选择符合项目进度的一种作为当前工程生成目标。

参考图 1.2.6, 配置当前“生成目标”的操作如下。单击生成目标栏内的 ▾ 按钮, 显示一个下拉列表框。下拉列表框内包含所有可用的生成目标, 从中选择一种需要的生成目标即可。选中的生成目标显示在生成目标栏内, 作为工程当前“生成目标”。

(3) 工具栏

提供了一些快速操作按钮, 比如 Make 按钮、Run 按钮和 Debug 按钮等。

(4) 选项卡

工程管理窗口用于显示与工程有关的所有信息, 包括 Files、Link Order 和 Targets 3 个选项卡。

Files 选项卡由标题栏和列表框组成, 用于显示工程内包含的文件、组、子工程等信息。其中, 标题栏包括文件栏、代码栏、数据栏、生成目标栏和调试栏; 列表框用于显示文件的详细信息, 如图 1.2.6 所示。

文件栏(File)用于显示文件名、组名、子工程名。

代码栏(Code)用于显示目标文件中代码空间的大小。

数据栏(Data)用于显示目标文件中数据空间的大小。

目标栏(Target)的作用是决定文件是否属于当前“生成目标”。如果文件属于当前“生成目标”, 则在工程管理窗口内能够对文件执行编译操作, 否则就不能对文件执行编译操作。文件是否属于当前“生成目标”, 是通过配置操作来实现的。在 Target 栏内, 将鼠标移动到

对应某个文件位置,单击指定位置,如果出现圆点“.”,说明文件属于当前“生成目标”;如果没有出现圆点“.”,说明文件不属于当前“生成目标”。默认情况下都有圆点“.”,说明文件都属于当前“生成目标”。

调试栏(Debug)的作用是:如果当前栏对应某个文件的位置有圆点“.”,则编译此文件时,生成的目标文件带有调试信息;如果没有圆点“.”,则编译此文件时,生成的目标文件不带调试信息。编译文件时是否生成调试信息,是通过配置操作来实现的,配置操作过程与 Target 栏配置相同。

需要说明的是,如果某文件不属于当前“生成目标”,则文件不能被编译,更谈不上生成带有调试信息的输出文件了。因此,在 Target 栏和 Debug 栏内对应文件位置都无圆点。如果 Target 栏对应某文件位置有圆点,而 Debug 栏没有,则编译此文件时,生成的输出文件不带调试信息。

Link Order 选项卡按照一定次序列表显示工程内的程序文件,程序文件的排列次序就是创建*.axf 文件时目标文件的链接次序。

在 Link Order 选项卡内,文件排列顺序不是固定的,只要拖动要变更顺序的文件,即可改变文件的排列次序。

注意:

不建议使用 Link Order 选项卡决定输出文件的结构,输出文件的结构是由专项配置文件来实现。

Targets 选项卡用于列表显示所有可用的生成目标。默认情况下可用的生成目标包括 DebugRel、Debug、Release,列表框内黑色箭头指示的生成目标就是当前“生成目标”。

2. 创建工程

CodeWarrior IDE 使用工程模板创建工程。工程模板本身也是一个工程,它预先配置了工程开发所需的各种参数。CodeWarrior IDE 提供了很多工程模板,每种工程模板都适合一类应用,用户还可以根据需求扩充工程模板。

(1) 工程模板

ADS1.2 安装目录下有个 Stationery 子目录,它保存了可供工程开发使用的工程模板,下面就简单介绍这些工程模板。

① ARM Executable Image

使用此模板创建的工程,编译链接后生成一个可执行的 ELF 镜像文件,此文件全部由 ARM 32 位代码组成。工程模板配置如下。

- 使用 ARM C 编译器编译后缀为“.c”的源程序文件。
- 使用 ARM C++编译器编译后缀为“.cpp”的源程序文件。
- 使用 ARM 汇编编译器编译后缀为“.s”的源程序文件。
- 使用 ARM 链接器链接目标文件,生成可执行的 ELF 镜像文件。
- 使用 AxD Debugger 调试和运行可执行的 ELF 镜像文件。

② ARM Object Library

使用此模板创建的工程，编译链接后生成一个 `armar` 格式的目标库，此目标库全部由 ARM 32 位代码组成。工程模板配置如下。

- a. 用 `armar` 工具创建一个目标库。
- b. 能单独调试和运行库文件。

③ Thumb Executable Image

使用此模板创建的工程，编译链接后生成一个可执行的 ELF 镜像文件，此文件全部由 Thumb 16 位代码组成。工程模板配置如下。

- a. 用 Thumb C 编译器编译后缀为“.c”的源程序文件。
- b. 用 Thumb C++编译器编译后缀为“.cpp”的源程序文件。
- c. 用 Thumb 汇编编译器编译后缀为*.s 的源程序文件。
- d. 用 ARM 链接器链接目标文件，生成可执行的 ELF 镜像文件。
- e. 用 AxD Debugger 调试和运行可执行的 ELF 镜像文件。

④ Thumb Object Library

使用此模板创建的工程，编译链接后生成一个 `armar` 格式的目标库，此目标库全部由 Thumb 16 位代码组成。工程模板配置如下。

- a. 用 `armar` 工具创建一个目标库。
- b. 能单独调试和运行库文件。

⑤ Thumb ARM Interworking Image

使用此模板创建的工程，编译链接后生成一个可执行的 ELF 镜像文件，此文件由 ARM 32 位代码和 Thumb 16 位代码组成。工程模板配置如下。

- a. 用 Thumb C 编译器编译后缀为“.c”的源文件。
- b. 用 Thumb C++编译器编译后缀为“.cpp”的源文件。
- c. 用 Thumb 汇编编译器编译后缀为“.s”的源文件。
- d. 用 ARM 链接器链接目标文件，生成可执行的 ELF 镜像文件。
- e. 用 AxD Debugger 调试和运行可执行的 ELF 镜像文件。

(2) 创建工程

以工程 Exp 为例，说明工程创建的过程。

① 调用工程创建选项卡

在 CodeWarrior IDE 主窗口中，选择 File | New 命令，显示创建对话框。单击 Project 标签，显示 Project(工程)创建选项卡，如图 1.2.7 所示。

② 工程创建信息

a. 工程模板

在工程模板列表框内单击 ARM Executable Image 项，选中 ARM Executable Image 工程模板。

b. 工程名

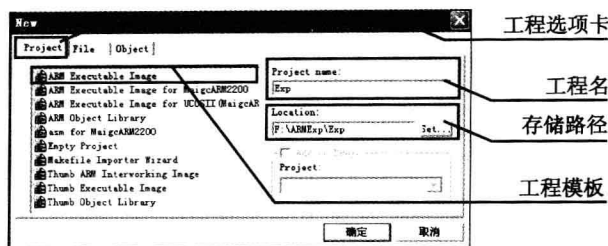


图 1.2.7 工程创建选项卡

在 Project name 项的编辑框内输入工程名 Exp。

c. 工程存储路径

在 Location 项内输入工程存储路径 F:\ARM_Exp\Exp。

③ 创建工程

单击“确定”按钮，执行工程创建操作。工程创建完毕后，CodeWarrior IDE 打开一个新窗口用于显示新建的工程 Exp，如图 1.2.6 所示。

④ 查看工程存储路径 F:\ARM_Exp\Exp

在桌面上，双击“我的电脑”图标，显示“我的电脑”窗口。将地址栏配置为工程存储路径 F:\ARM_Exp\Exp，显示 F:\ARM_Exp\Exp 路径下的信息，包括工程文件 Exp.mcp 和工程数据存储路径 Exp_Data，这就是工程创建后的内容，其中，mcp 为工程文件后缀，是工程文件的标识，如图 1.2.8 所示。

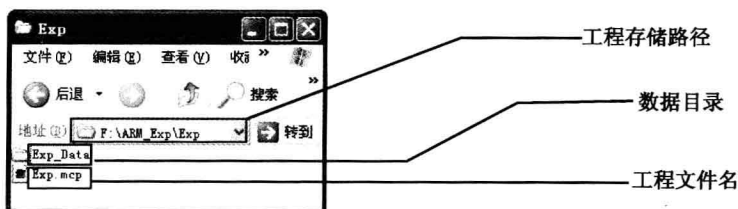


图 1.2.8 工程存储路径

3. 打开工程

常用打开工程的操作方法有 3 种，下面以工程 Exp.mcp 为例，简单介绍打开工程的操作过程。

(1) 使用 Open 菜单打开工程

① 显示工程“打开”对话框

在 CodeWarrior IDE 主窗口中，选择 File | Open 命令，显示“打开”对话框，如图 1.2.9 所示。

② 选择工程存储路径

单击“查找范围”项的 ▾ 按钮，将“查找范围”项配置为工程存储路径 F:\ARM_Exp\Exp。

③ 选择工程文件类型

单击“对象类型”项的 ▾ 按钮，将“对象类型”项配置为工程文件类型 Project Files(*.mcp)，则在文件列表框内显示了所有的工程文件，如图 1.2.9 所示。

④ 选择工程文件

在文件列表框内单击工程文件名 Exp.mcp，工程文件名 Exp.mcp 会显示在“对象名称”项的显示栏内。

⑤ 打开工程文件

单击“打开”按钮，执行工程打开操作。工程打开后显示在工程管理窗口内，如图 1.2.6 所示。

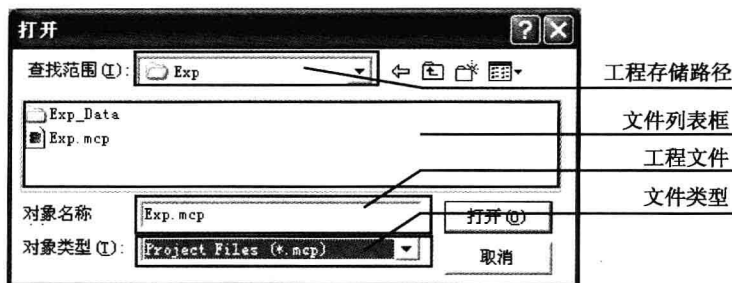


图 1.2.9 工程打开窗口

(2) 使用 Open Recent 命令打开工程

通常情况下，最近操作过的工程文件和程序文件会以菜单的方式挂接在菜单 File | Open Recent 下，如图 1.2.10 所示。

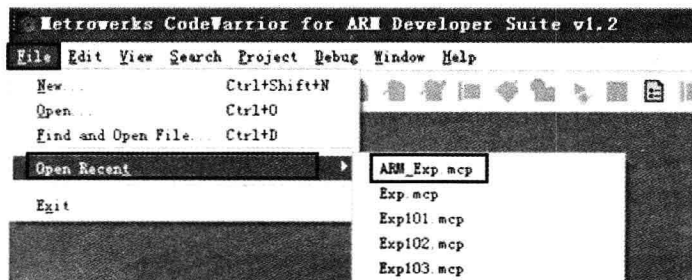


图 1.2.10 打开最近操作过的工程

在 CodeWarrior IDE 主窗口中，选择 File | open Recent | Exp.mcp 命令，执行工程打开操作。操作的结果是打开工程 Exp.mcp 并显示在工程管理窗口内，如图 1.2.6 所示。

(3) 使用鼠标操作打开工程

① 找到工程存储路径

在桌面上单击“我的电脑”，显示“我的电脑”窗口；将地址栏配置为工程存储路径 F:\ARM_Exp\Exp，则工程文件显示在文件列表框内。

② 打开工程文件

双击工程文件名 Exp.mcp，默认情况下会调用 CodeWarrior IDE 打开工程 Exp.mcp，并显示在工程管理窗口内，如图 1.2.6 所示。

4. 关闭工程

在 Code Warrior IDE 主窗口中, 选择 File | Close 命令或单击工程管理窗口中的关闭按钮即可关闭工程。

5. 向工程内添加文件

通常有两种情况需要向工程添加文件, 一种是使用 CodeWarrior IDE 创建的文件没有加入工程; 另一种是使用其他编辑器编写的程序文件。

以向工程 Exp.mcp 添加文件 main.c 为例, 说明向工程添加文件的操作过程。

(1) 复制文件到工程目录下

将文件 main.c 复制到工程存储路径 F:\ARM_Exp\Exp 下, 即与 Exp.mcp 文件同在一个路径下, 如图 1.2.11 所示。

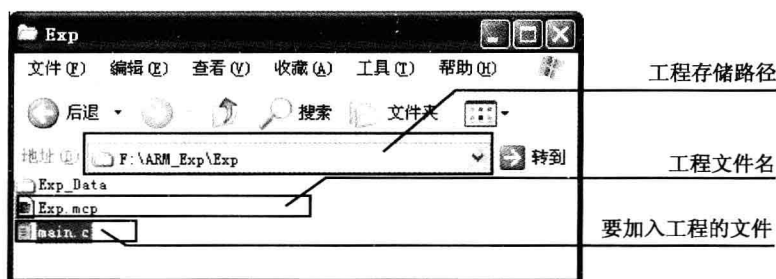


图 1.2.11 Exp 工程存储路径

(2) 打开工程

选择 File | Open 命令, 显示“打开”对话框; 将“查找范围”项配置为工程存储路径 F:\ARM_Exp\Exp; 单击工程文件名 Exp.mcp, 工程文件名显示在“对象名称”项的编辑框内; 单击【打开】按钮, 打开工程 Exp.mcp 并显示在工程管理窗口内。

(3) 显示添加文件菜单

在工程管理窗口内右击, 弹出菜单, 如图 1.2.12 所示。

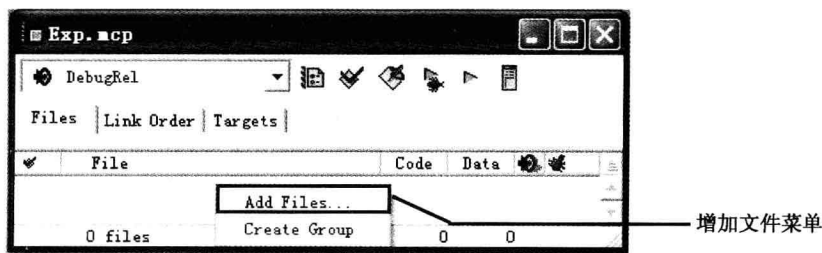


图 1.2.12 增加文件菜单

(4) 向工程添加文件

① 选择 Add Files 命令, 显示“打开”对话框。

② 将“查找范围”项配置为工程存储路径 F:\ARM_Exp\Exp; 单击文件名 main.c, 文件