IAR EWARM 安装使用指南

—— For Luminary Micro Stellaris 系列

—— Use J-link 仿真器

北京精仪达盛科技有限公司

目录

第 1 章 EWARM 集成开发环境

1.1 IAR EWARM 简介

IAR Embedded Workbench for ARM（下面简称IAR EWARM）是一个针对ARM 处理器的集成开发环境，它包含项目管理器、编辑器、C/C++编译器和ARM 汇编器、连接器XLINK和支持RTOS 的调试工具C-SPY。在EWARM 环境下可以使用C/C++和汇编语言方便地开发嵌入式应用程序。比较其他的ARM 开发环境，IAR EWARM 具有入门容易、使用方便和代码紧凑等特点。

目前IAR EWARM支持ARM Cortex-M3 内核的最新版本是 5.4，该版本支持Luminary全系列的MCU。为了方便用户学习评估，IAR 提供一个限制32K 代码的免费试用版本。用户可以到**IAR**公司的网站**www.iar.com**下载。

1.2 J-LINK 仿真器介绍

全功能版 J-LINK 配合 IAR EWARM、ADS、KEIL、WINARM、Real View 等集成开发环境支持所有 ARM7/ARM9/Cortex 内核芯片的仿真，通过 RDI 接口和各集成开发环境无缝连接，操作方便、连接方便、简单易学，是学习开发 ARM 最好最实用的开发工具。最显著的特点:速度快，FLASH 断点不限制数量，支持 IAR、KEIL、RV、ADS 等环境。

* USB 2.0 接口；
* 支持任何 ARM7/ARM9 核 Cortex M3 supported, 包括 ithumb 模式；
* 下载速度达到 600k byte/s;
* DCC 速度到达 800k byte/s;
* 与 IAR Workbench 可无缝集成；
* 通过 USB 供电，无需外接电源；
* JTAG 最大时钟达到 12M；
* 自动内核识别；
* 自动速度识别；
* 支持自适应时钟；

* 所有 JTAG 信号能被监控，目标板电压能被侦测；

* 支持 JTAG 链上多个设备的调试；

* 完全即插即用；

* 20Pin 标准 JTAG 连接器；

* 宽目标板电压范围：1.2V-3.3V (可选适配期支持到 5V)；

* 多核调试；

* 包括软件：J-Mem,可查询可修改内存；

* 包括 J-Link Server (可通过 TCP/IP 连接到 J-Link);

* 可选配 J-Flash,支持独立的 Flash 编程；

* 选配 RDI 插件使 J-Link 适合任何 RDI 兼容的调试器如 ADS、Relview 和 Keil 等；

* 选配 RDI Flash BP，可以实现在 RDI 下，在 Flash 中设置无限断点；

* 选配 RDI Flash DLL，可以实现在 RDI 下的对 Flash 的独立编程；

* 选配 GDB server，可以实现在 GDB 环境下的调试。

图 1.1 J-LINK仿真器

1.3 IAR EWARM 的安装

1.3.1 IAR EWARM 的安装步骤如

1. 从IAR 的官方网站上www.iar.com/ewarm 下载IAR 5.4，32K 代码试用评估版本，文件名为： EWARM-KS-WEB-5.4.exe。

2. 运行EWARM-EV-WEB-5.4.exe

3. 点击Install the IAR Embedded Workbench，开始安装。如图 1.2 所示。

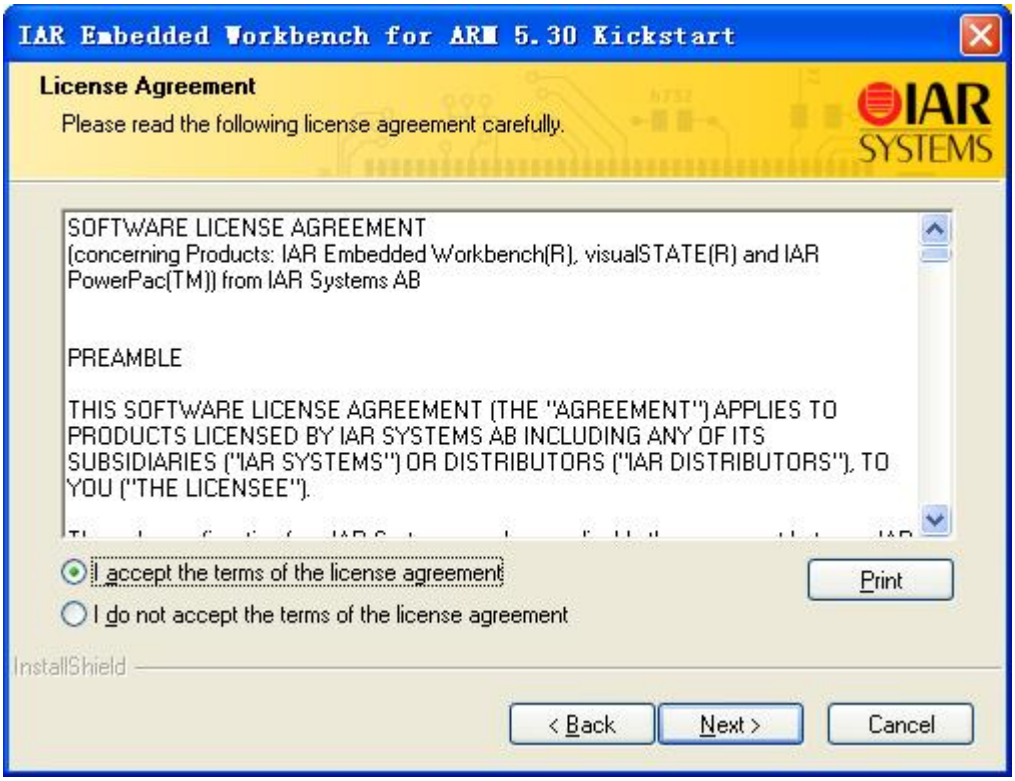

图 1.2 EWARM 安装

4. 输入许可证号（License）和密钥（License key）用户从下载的软件包中的文本文件中提取许可证号（License）和密钥（License key），分别输入下面两个窗口如图 1.3 和图 1.4 所示。许可接受后建议按默认设置安装。

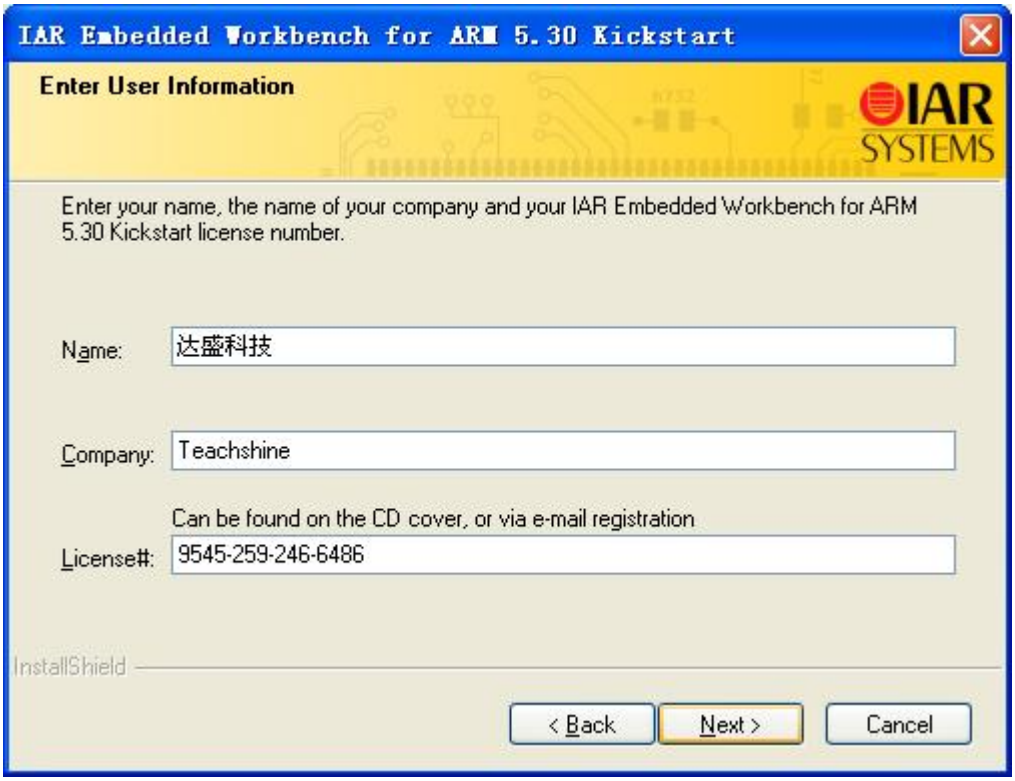

图 1.3 License 输入

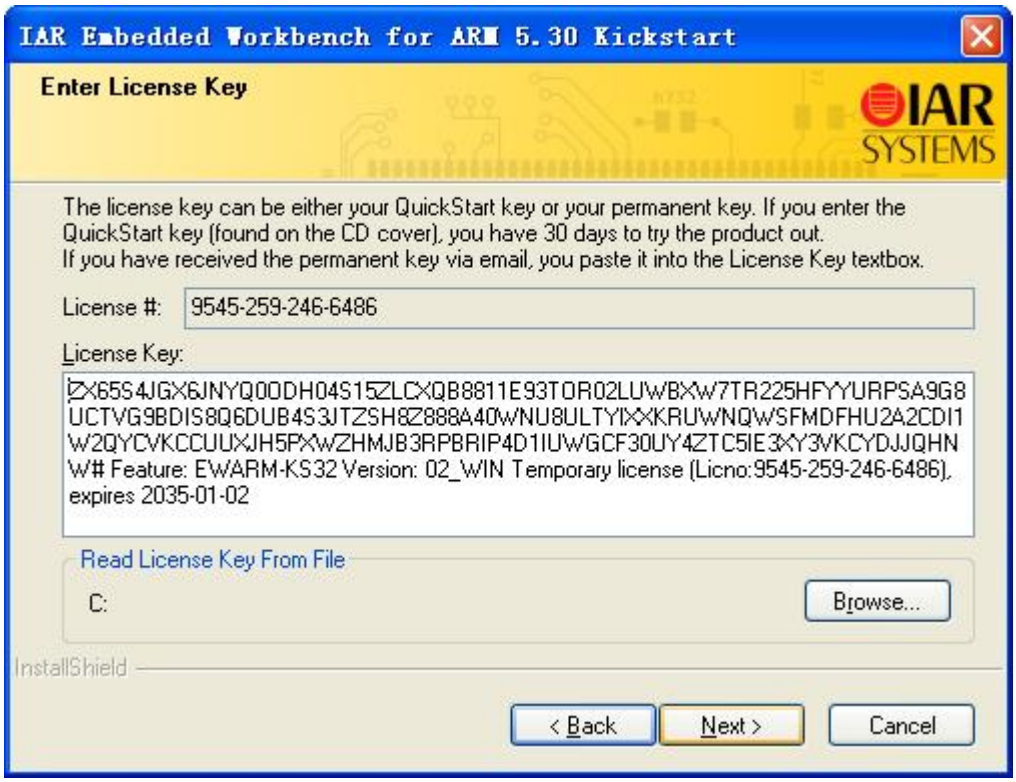

图 1.4 密钥输入

5. 点击“下一步”直到软件安装完成。

1.3.2 安装 J-LINK 驱动

1、双击J-LINK驱动目录下的安装文件开始安装，选择默认路径即可，出现如下对话框时打钩，直至安装完成。如图1.5所示：

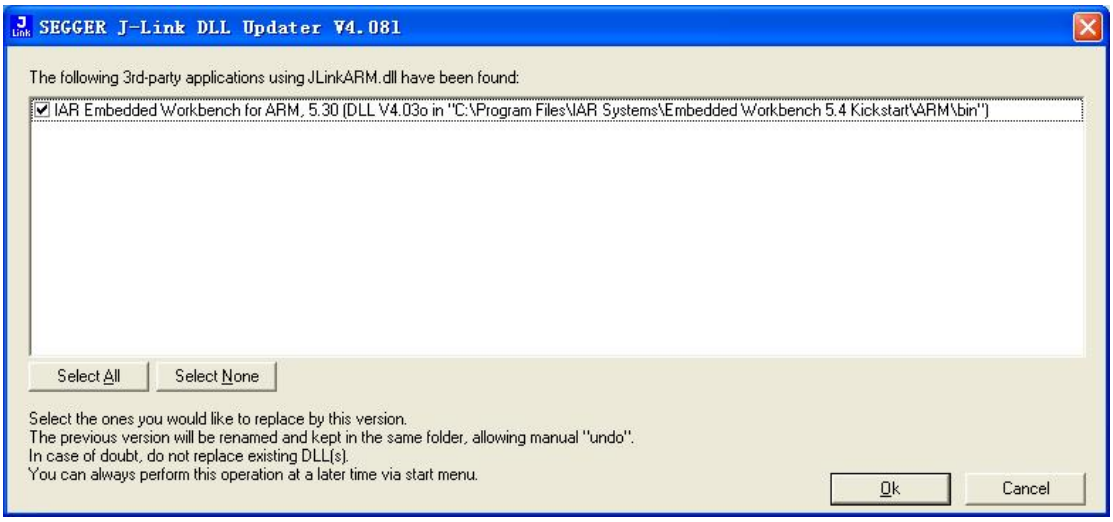

图1.5 选择开发环境

2、用USB电缆把仿真器与开发板连接上后，在我的电脑设备管理器的通用串行总线控制器下能找到J-link driver。如图1.6所示：

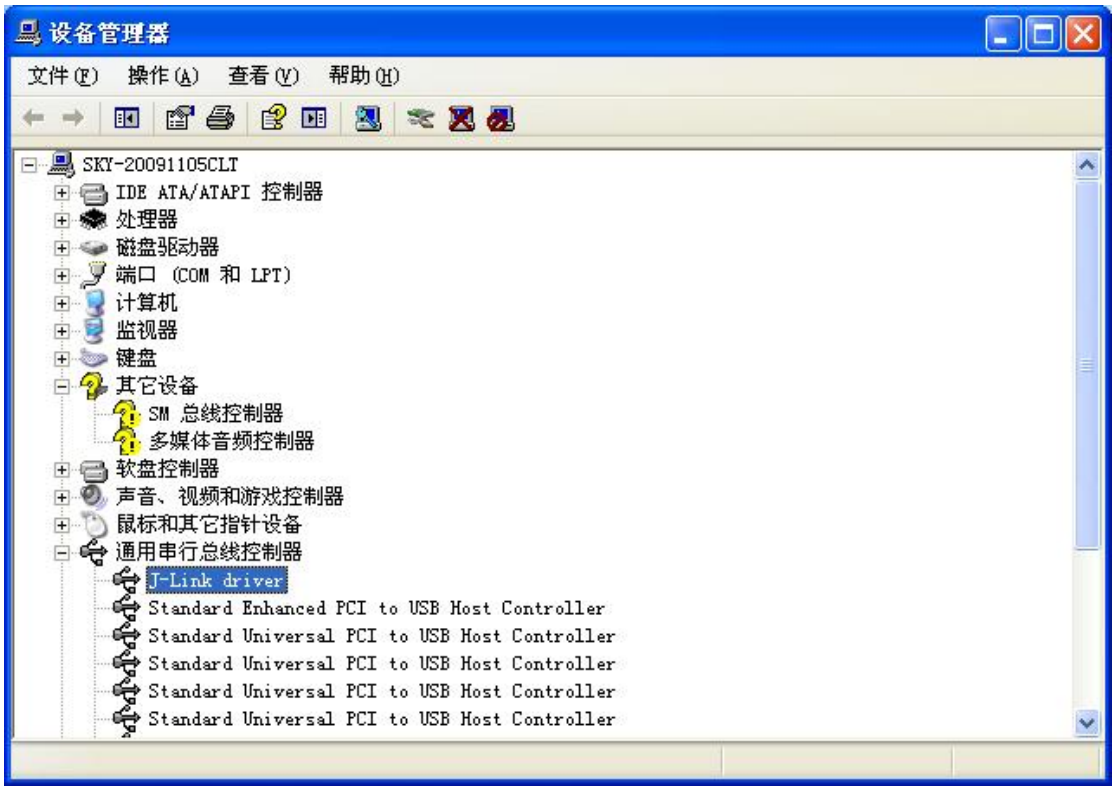

图1.6 安装完成后显示的驱动信息

第 2 章 安装流明诺瑞驱动库

在安装好EWARM 集成开发环境后，就可在该环境下新建工程了。但在新建工程之前，为了使以后的工程更便于管理、工程中的设置更加简单化，在这里就需要一些准备工作，将某些文件拷贝到指定路径下，具体的操作方式将在随后介绍。至于为什么要这样做，在工程的设置时就会体会出其优越性。

注意：本文是以32K 的试用版为例作讲解。如果用正式版可以参照本文进行设置。

2.1 下载最新库文件

从流明诺瑞官方网站http://www.luminarymicro.com 下载最新的驱动库文件。假设保存于“D:\”，如图 2.1 所示。

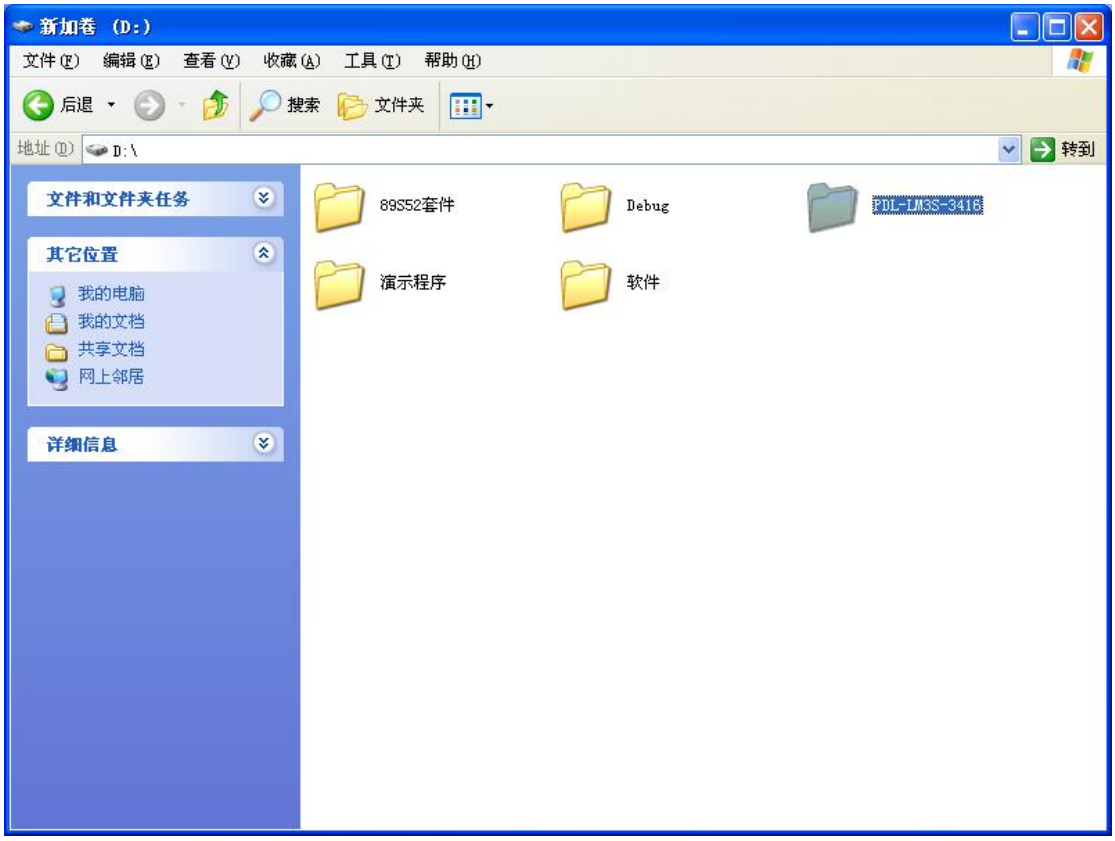

图 2.1 驱动库文件存放目录

2.2 拷贝连接器命令文件

这一步是将连接器命令文件复制到IAR的默认路径下面，节省了每次在选择连接器命令

文件时的查找步骤。

1. 打开目录“D:\PDL-LM3S-3416\DriverLib\ewarm”如图 2.2 所示。

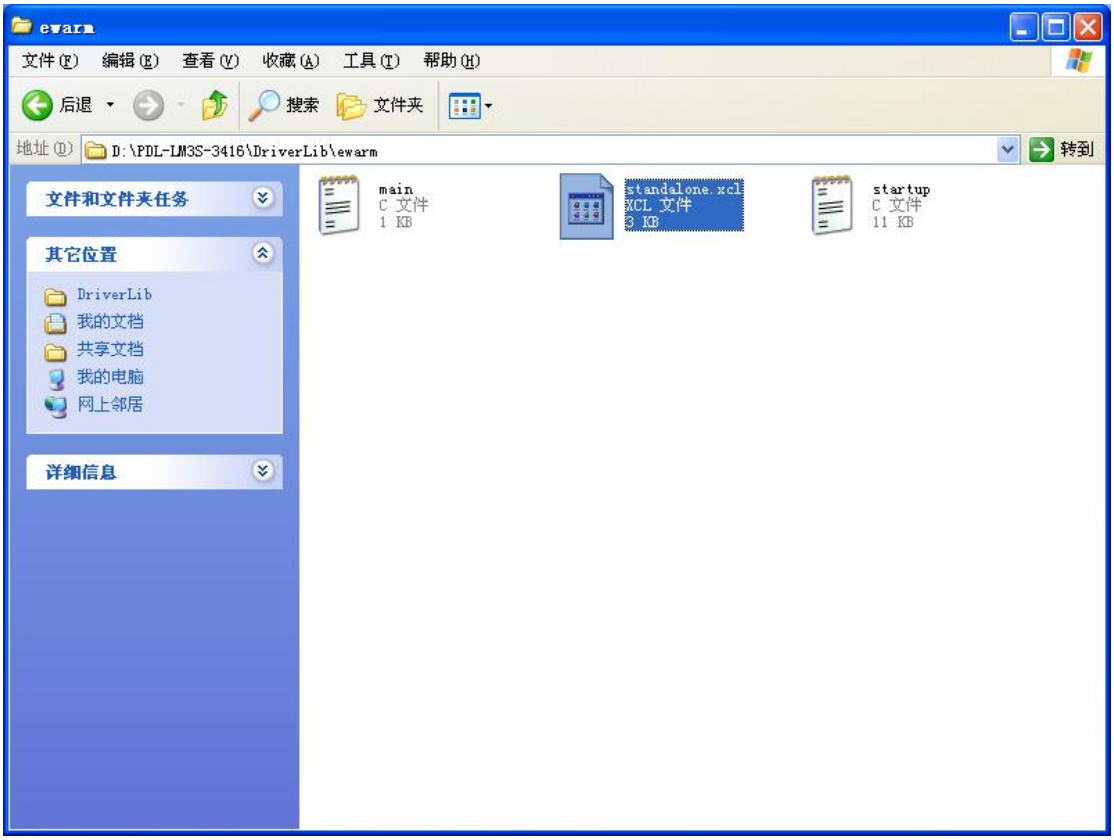

图 2.2 原连接器命令文件

2. 将图 2.2 中所示的“standalone.xcl”文件复制一份，然后粘贴到“C:\Program Files\IARSystems\Embedded Workbench 5.30 Kickstart\arm\config”目录下。并改名为“lnk—LM3.xcl”如图2.3所示

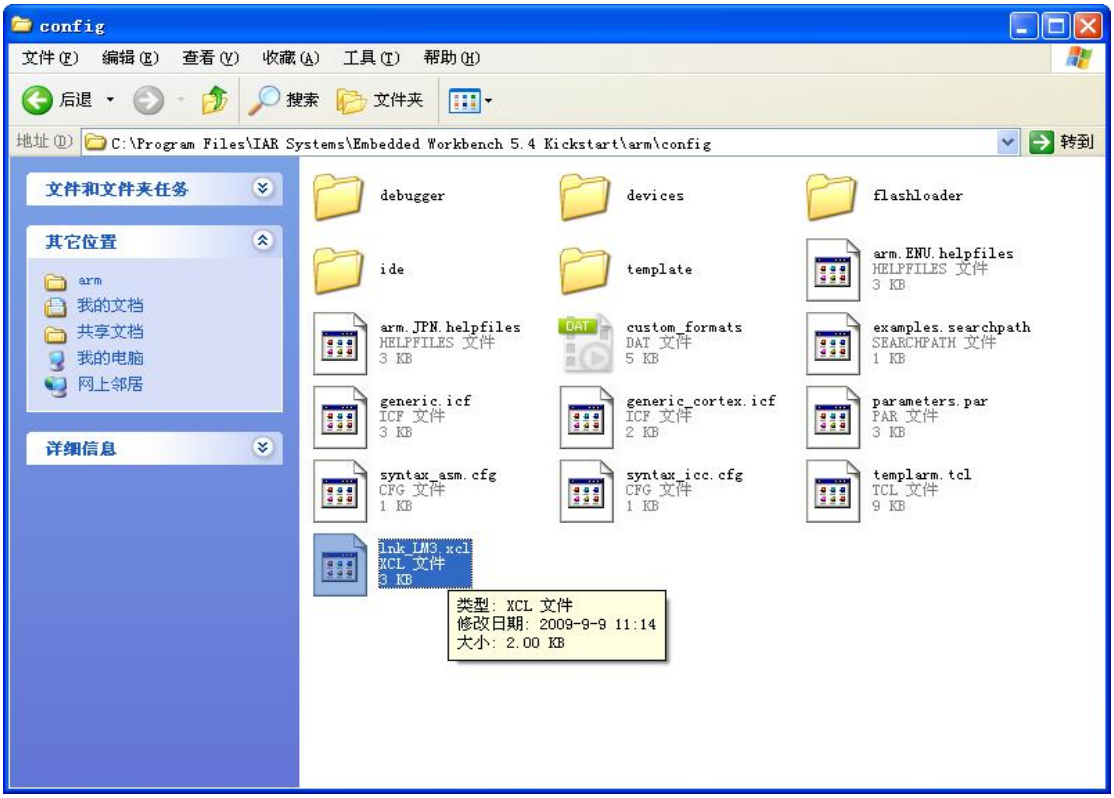

图 2.3 连接器命令文件存放的目录

2.3 拷贝驱动库头文件

这一步是将库文件复制到IAR 的默认路径下面，减轻了每次在选择库文件时的添加库文件步骤。

1. 打开目录“D:\PDL-LM3S-3416\DriverLib”如图 2.4 所示。

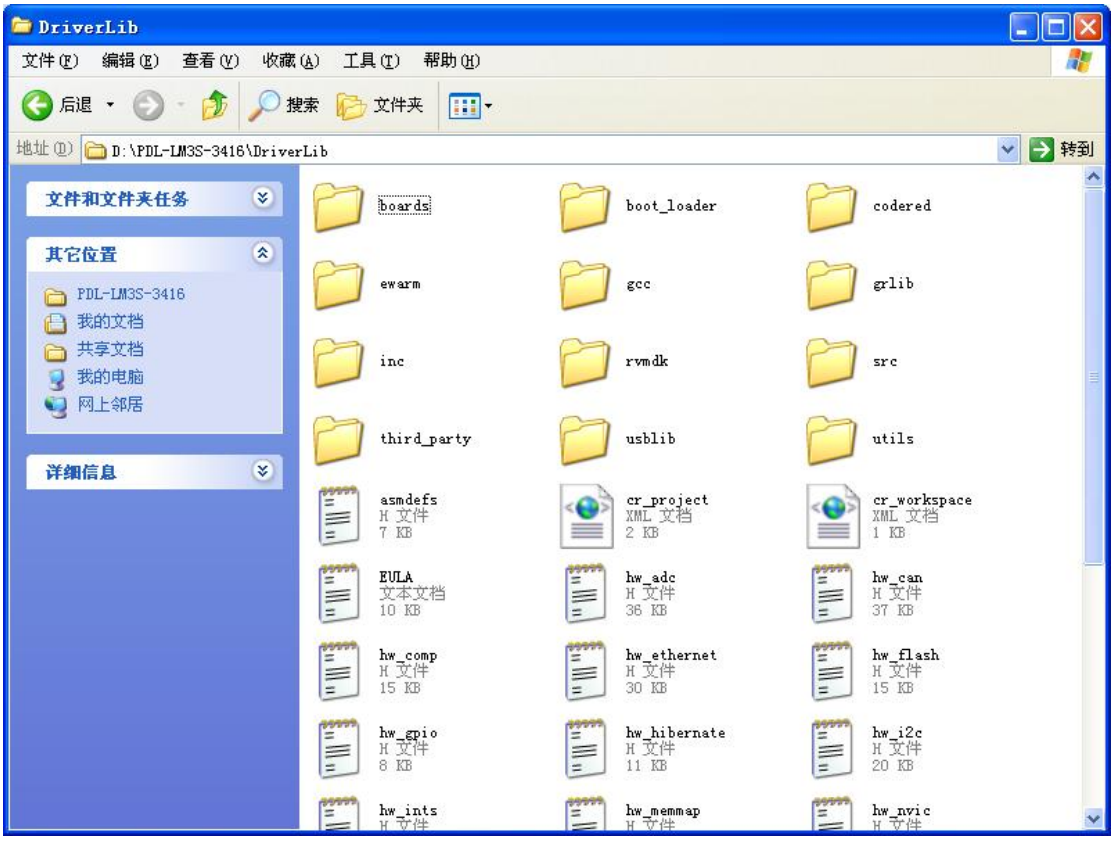

图 2.4 原驱动库头文件目录

2. 在“C:\Program Files\IAR Systems\Embedded Workbench 5.30 Kickstart\arm\inc”下，新建一个“Luminary”文件夹，如图 2.5 所示。

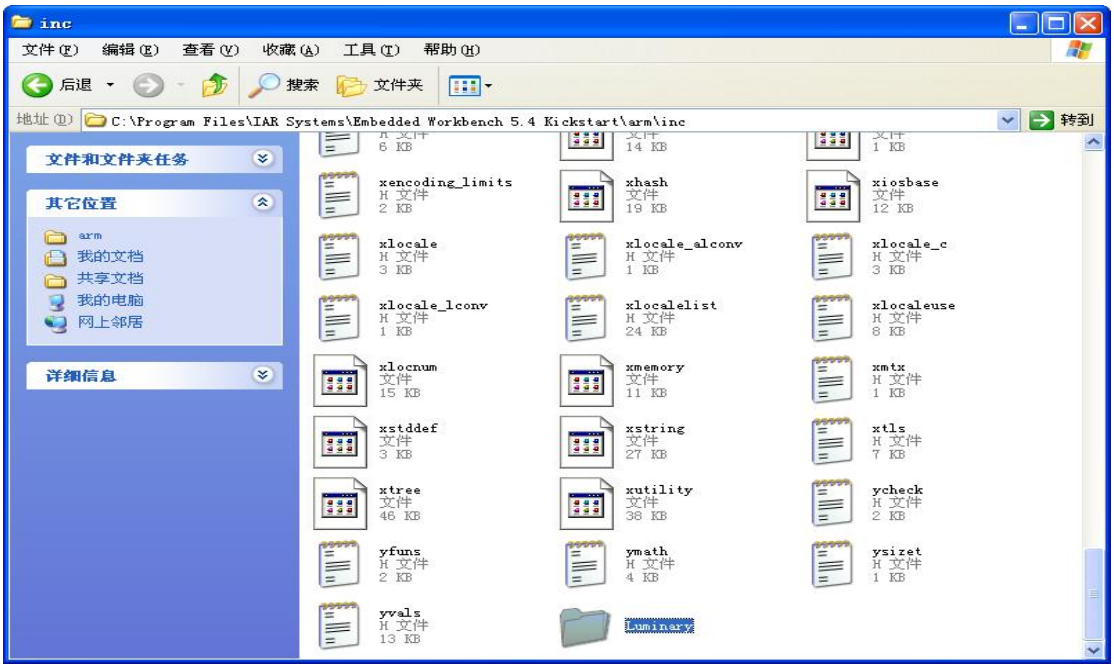

图 2.5 新建“Luminary”文件夹

3. 复制驱动库头文件，然后粘贴到新建的“Luminary”文件夹下，即“C:\ProgramFiles\IAR Systems\Embedded Workbench 5.30 Kickstart\arm\inc\Luminary”目录。如图 2.6 所示。

这里包括三个步骤：

第一步是：将图 2.4 中的所有.h 文件，拷贝到新建的“Luminary”文件夹下。

第二步是：打开图 2.4 中的“src”文件，将该文件下的所有.h 和.c 文件，拷贝到新建的“Luminary”文件夹下。

第三步是：打开图 2.4 中的“inc”文件，将该文件下的所有.h 文件，拷贝到新建的“Luminary”文件夹下。

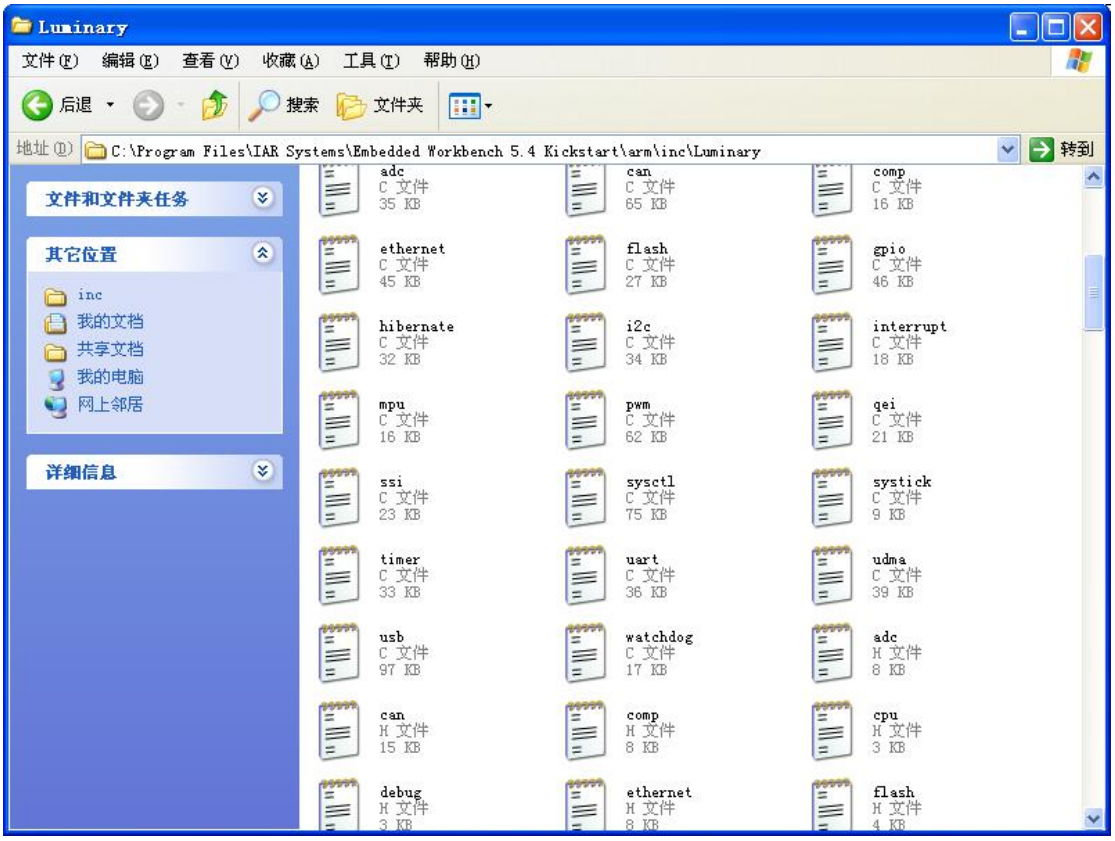

图 2.6 驱动库头文件存放目录

2.4 拷贝底层驱动函数库

1. 打开目录“D:\PDL-LM3S-3416\DriverLib\src\ewarm\Exe”如图 2.7 所示。

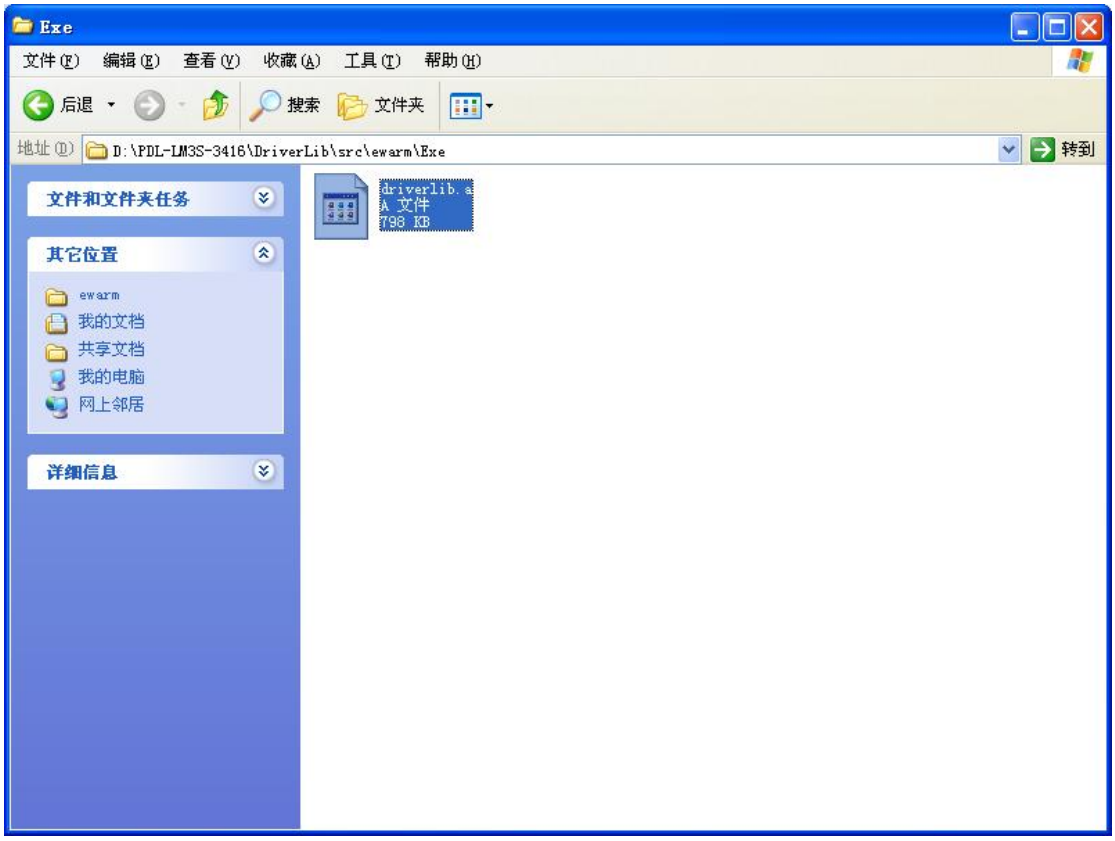

图 2.7 底层驱动函数库目录

2. 在“C:\Program Files\IAR Systems\Embedded Workbench 5.30 Kickstart\arm\lib”下，新建一个“Luminary”文件夹，如图 2.8 所示。

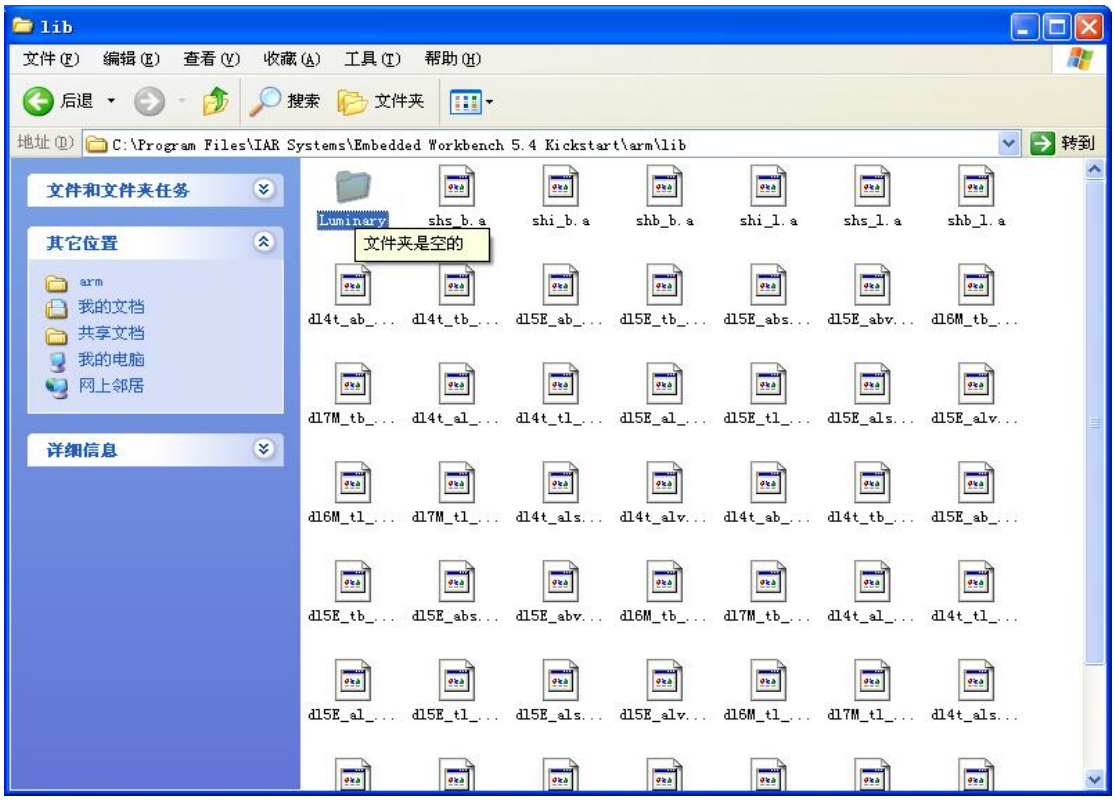

图 2.8 新建“Luminary”文件夹

3. 将图 2.4 中的“driverlib.r79”复制一份，然后粘贴到新建的“Luminary”文件夹下，即“C:\Program Files\IAR Systems\Embedded Workbench 5.30 Kickstart\arm\lib\Luminary”目录下。如图 2.9 所示。

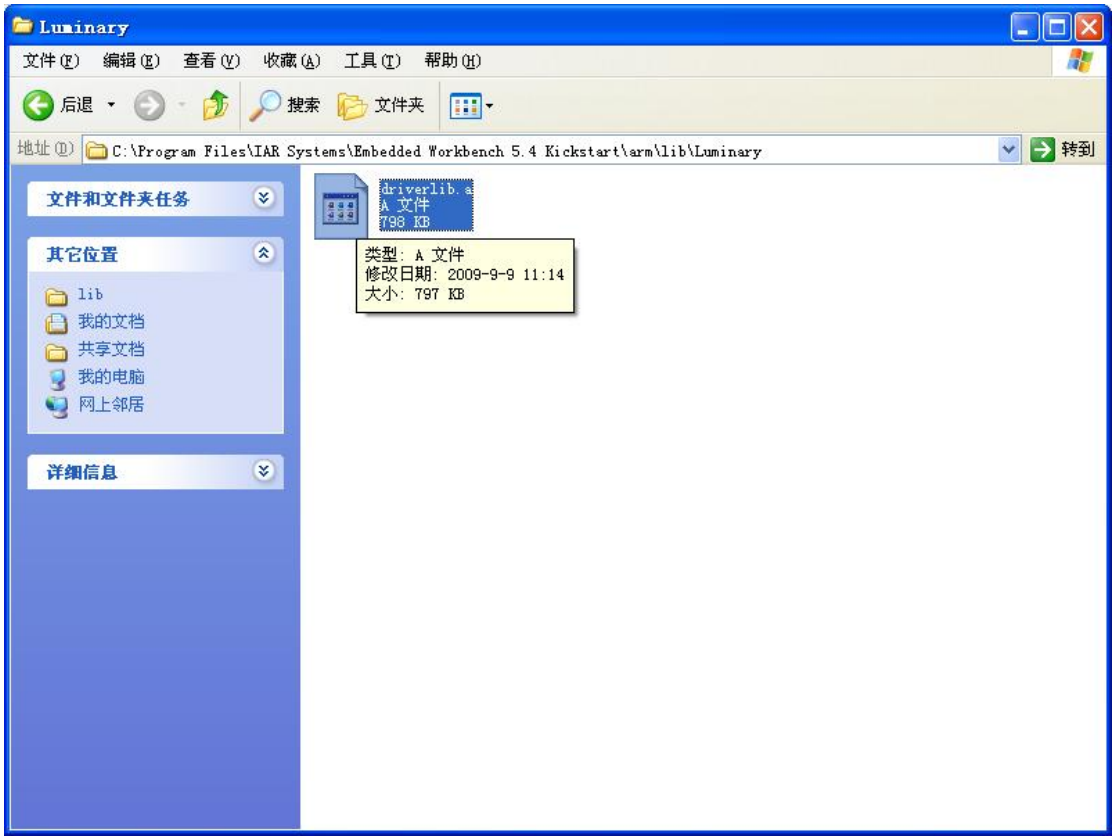

图 2.9 底层驱动函数库存放目录

到此，要做的准备工作已经完成。

第 3 章 在 EWARM 中新建一个新项目

要为某个目标系统开发一个新应用程序，必须先新建一个新项目。新建项目具体步骤下面将作详细介绍。

3.1 建立一个项目文件目录

首先应该为新项目创建一个目录，用来存放与项目有关的各种文件。项目开发过程中生成的一系列文件，如：工作区文件，开发环境的配置，编译、连接和调试选项配置，各种列表文件和输出文件等都将被存放在这个目录下。用户也可以选择把各种源文件也放在这个目录下。在下面的例子中我们生成一个D:\DEMO 目录。

3.2 新建工作区

EWARM 虽然是按项目进行管理，但是要求把所有的项目都放在工作区（Workspace）。用户如果是第一次使用EWARM 开发一个新项目，必须先创建一个新工作区，然后才能在工作区中创建新项目。一个工作区中允许存放一个或多个项目。如果用户过去已经建立了一个工作区并且希望把目前要建的新项目放在老工作区内，则可以直接打开老工作区并执行第三步生成新项目。创建新工作区方法如下：

启动EWARM 开发环境，如图 3.1 所示。

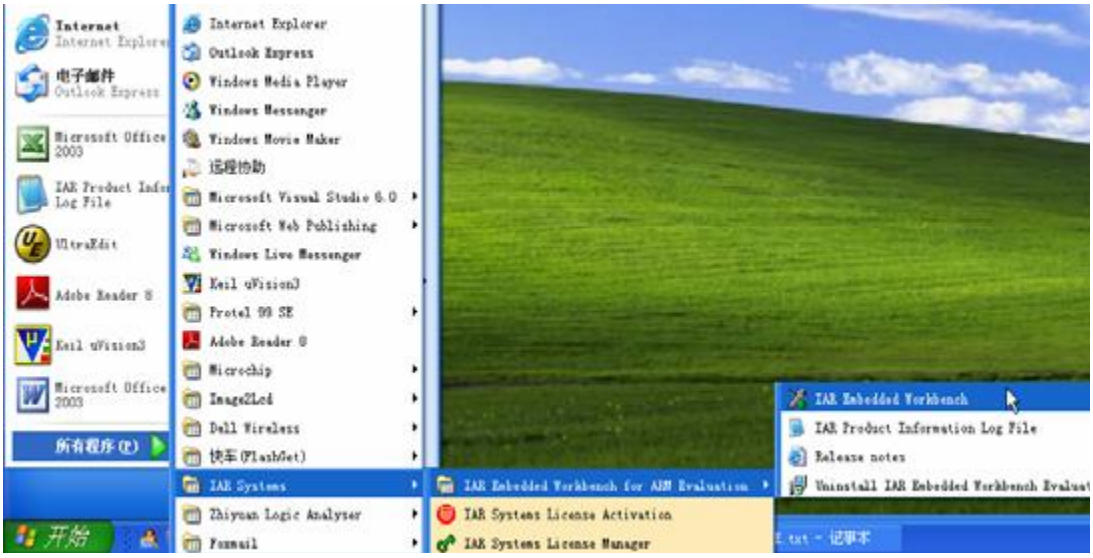

图 3.1 启动EWARM 开发环境

选择主菜单的 File > New > Workspace 命令，然后开启一个空白工作区窗口，如图 3.2所示。

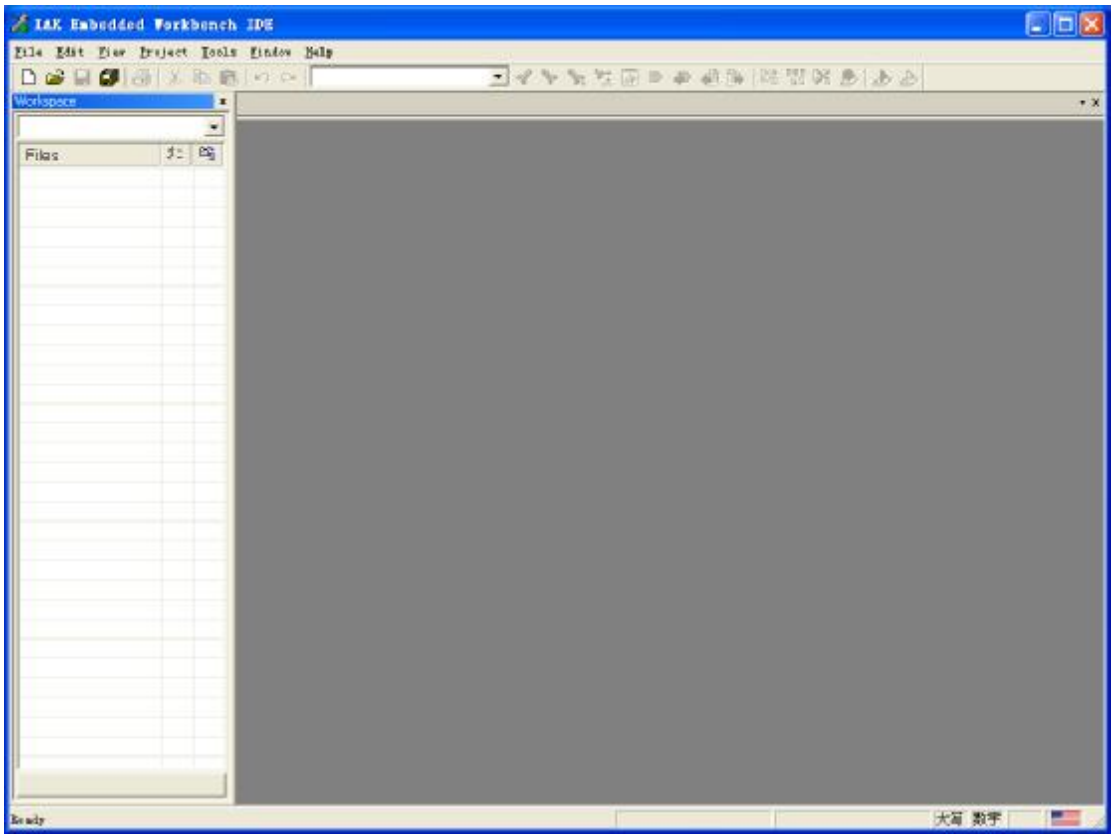

图 3.2 空白工作区窗口

3.3 生成新项目

下一步就是在工作区中创建新项目，方法如下：

1. 选择主菜单 Project > Create New Project，弹出生成新项目窗口。EWARM 提供几种应用程序和库程序的项目模板。如果选择Empty project，表示采用默认的项目选项设置，为一个空工程。在本例中我们选择Empty project，如图 3.3 所示。

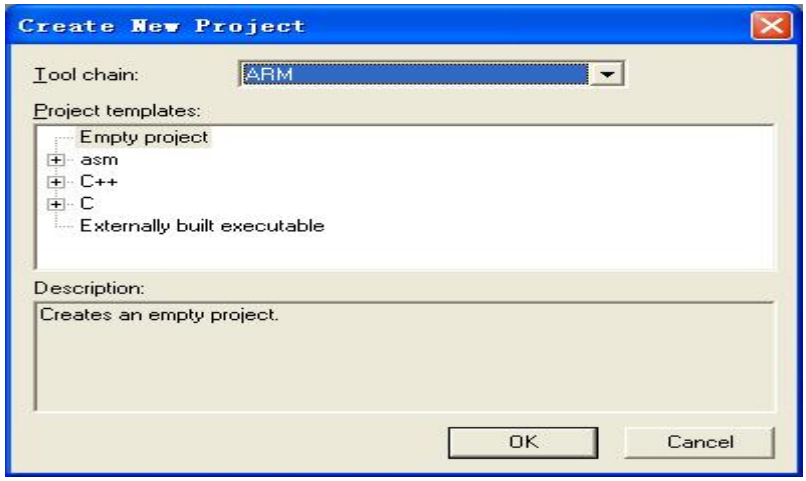

图 3.3 生成新项目窗口

2. 在Tool chain 栏中选择ARM，点击OK 按钮，弹出“另存为”窗口。如图 3.4 所示。

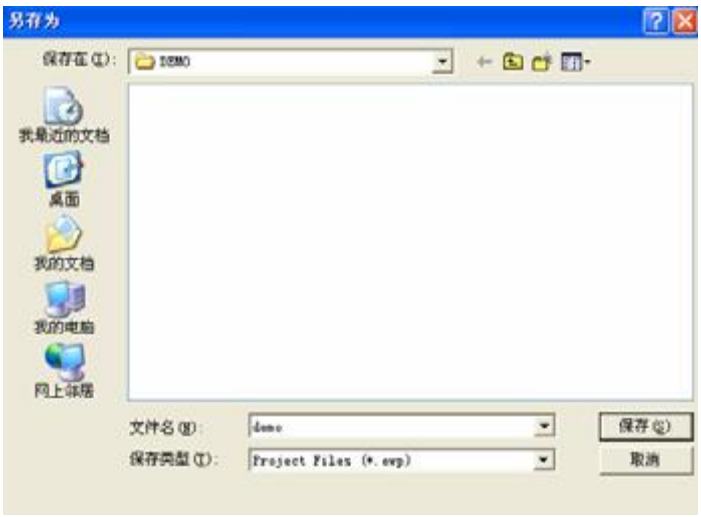

图 3.4 “另存为”窗口

3. 在“另存为”窗口中浏览和选择新建的D:\DEMO 目录，输入新项目的文件名为demo，然后保存。这时在屏幕左边的Workspace 窗口中将显示新建的项目名和输出代码模式，如图 3.5 所示。

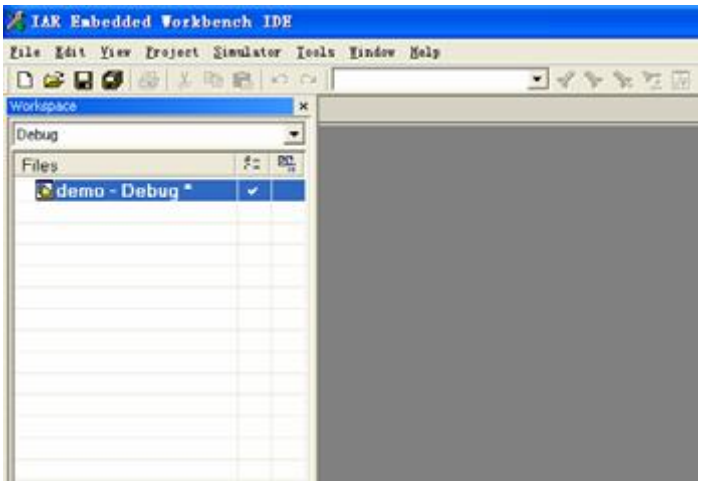

图 3.5 新建的项目名

项目名后面的Debug 表示输出含调试信息的代码文件。EWARM 能为项目提供两种输出代码模式：Debug 和Release。Debug 模式生成含调试信息的程序代码，用户利用它可以在EWARM 环境下调试应用程序。而Release 模式生成不含调试信息的发行版本的程序代码，其代码比较紧凑。用户可以从Workspace 窗口顶部的下拉菜单中选择两种项目配置之一，本例我们选择Debug。现在DEMO 目录下已生成一个demo.ewp 文件，该文件中将包含与 demo 项目设置有关的信息，如编译、连接（build）的选项等。

注意：demo-Debug 后的 * 号表示当前的工作区和项目经修改后还没有被保存。

4. 保存工作区

新生成的工作区需保存才有效，所以在添加项目后EWARM 要求执行保存工作区操作。保存工作区选择主菜单 File > Save Workspace，浏览并选择D:\DEMO 目录。然将工作区取名为demo 输进File name 输入框，按保存按钮退出，如图 3.6 所示。这时在D:\DEMO 目录下又生成一个 demo.eww 文件。同时在D:\DEMO 目录下还生成一个settings 子目录，这个目录下存放保存窗口设置和断点设置等与当前操作有关信息的其他文件。

图 3.6 保存工作区

注意：保存操作完成后项目名后的 * 号已经消失。

3.4 添加/新建文件

保存工作区后，下一步就是在项目中新建文件或添加已有文件。项目中的文件允许分组，用户可以根据项目的需要和自己的习惯来组织源文件。为举例说明，这里新建以下几个文件组：一个startup 文件组，一个 src 文件组，一个 lib 文件组。注意：往项目中添加文件时只需添加汇编语言和C 语言的源程序，不需要添加头文件(即.h 头文件)。但是用户必须在配置项目的编译器、连接器选项时指明包含头文件的路径和目录。关于项目配置选项的设定我们会在后面详细介绍。

3.4.1 建立文件组

右击“demo-Debug”然后选择 ADD > ADD Group…，如图 3.7 所示。

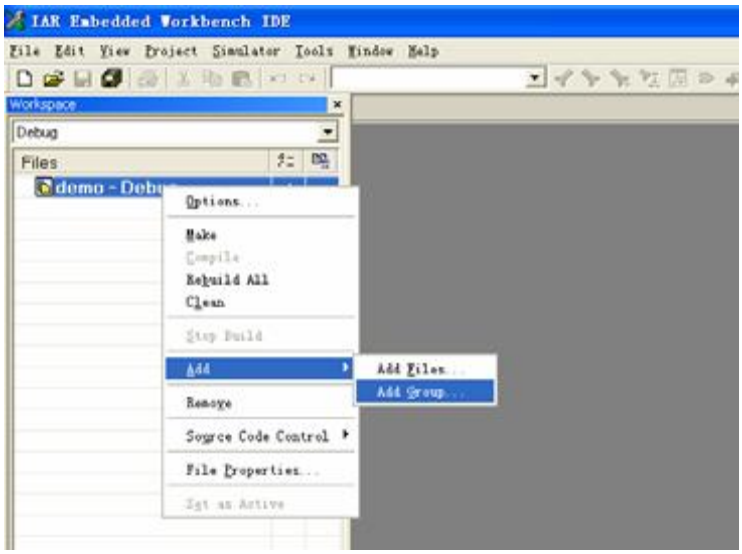

图 3.7 建立文件组

新建3 个文件组：startup 文件组，src 文件组，lib 文件组，如图 3.8 所示。

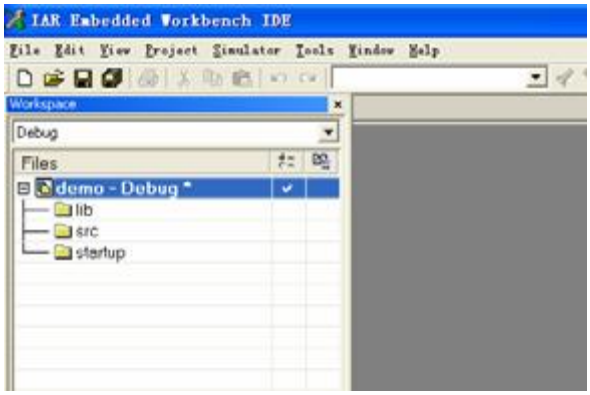

图 3.8 新建3 个文件组

3.4.2 添加对应文件

向文件组添加对应文件，如图 3.9 所示。

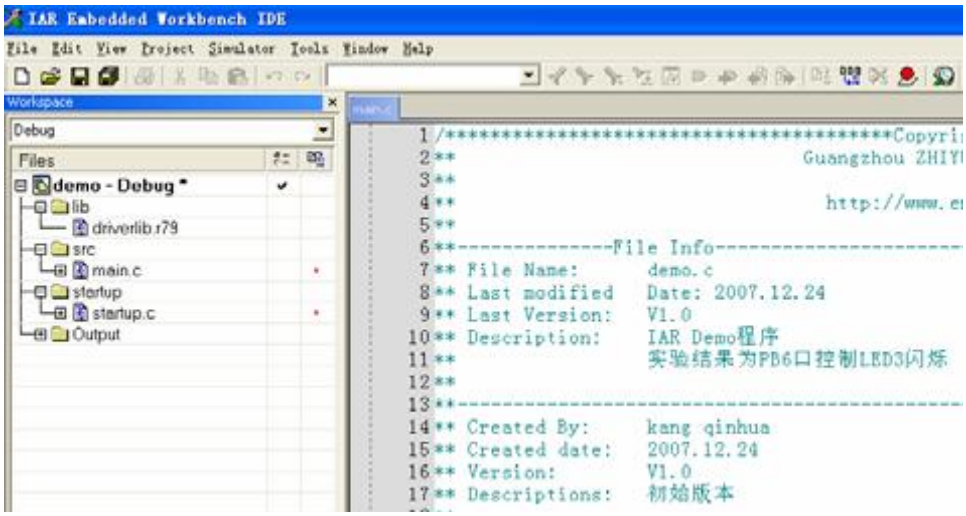

图 3.9 向文件组添加对应文件

□ 在lib 组添加driverlib.r79 文件。

添加方法：右击lib，选择ADD > ADD Files…，在弹出的对话框中选择目录：“C:\ProgramFiles\IAR Systems\Embedded Workbench 5.30 Kickstart\arm\lib\Luminary”，选择需要添加的库文件driverlib.r79，如图 3.10 所示。

图 3.10 选择需要添加的库文件

□ 在startup 组添加startup.c 文件。将“D:\PDL-LM3S-3416\DriverLib\ewarm”下的startup.c 文件复制到工程目录D:\DEMO下面。然后右击startup，选择ADD > ADD Files…，在弹出的对话框中选择目录D:\DEMO，添加startup.c 文件，如图 3.11 所示。

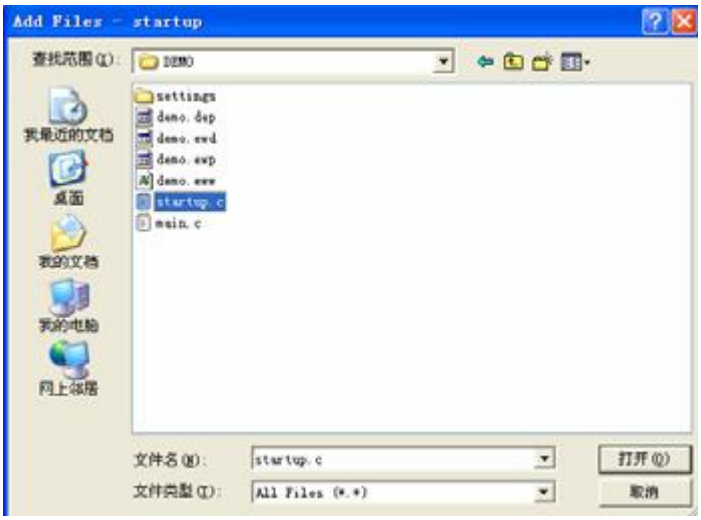

图 3.11 添加startup.c 文件

□ 在src 组中新建需要的main.c 文件或添加已有的main.c 文件，即主程序在这里编辑。这里新建一个main.c，首先单击src 组，选择File > New > File（也可以选择Newdocument），将在窗口中出现一个空白页，再选择File > Save，弹出另存为对话框，保存在D:\DEMO，保存为main.c，如图 3.12 所示。

图 3.12 另存为对话框

然后右击src 组，选择ADD > ADD Files…，在弹出的对话框中选择目录D:\DEMO，添加main.c 文件。此时，便可以在该main.c 文件中编辑需要的程序，这里编写了一个LED 灯闪烁的示例程序。如图 3.9 所示。

3.5 项目选项设置

生成新项目和添加文件后的下一步是为项目设置选项。设置项目选项是非常重要的一

步，如果设置不当，编译、连接就会出错，就无法生成正确的代码文件。大家记得，在创建新项目时我们选择了Empty project 模板，表示采用默认的项目选项设置。但是这些默认的设置还要根据具体项目的需要进行修改。IAR EWARM 提供的项目选项内容繁多，初学者可能会感觉到摸不着头脑、无从下手。实际上关键的选项并不多，只要把它们设置正确了，其它的采用默认设置就不会出错。下面我们把这些关键选项设置逐条介绍。

注意：文中没有提及的选项均采用默认设置。

3.6 通用选项设置

IAR EWARM 允许为工作区中的任何一级目录和文件单独设置选项，但是用户必须首先为整个项目设置通用的选项General Option。设置方法是：选中工作区中的项目名demo – Debug，按鼠标右键在弹出菜单中选择Options…或选择主菜单 Project > Options…。在弹出的Options 窗口左边的目录（Category）中选择第一项General Options。然后分别在：

□ Target 设置

□ 在Processor Variant 框中选择Device。并点击右边的器件选择按钮，选择芯片型号Luminary LM3Sx9xx。同时Endian mode 选择Little，Stack align 选择4 byte。如图3.13 所示；

□ 其它选项采用默认值。

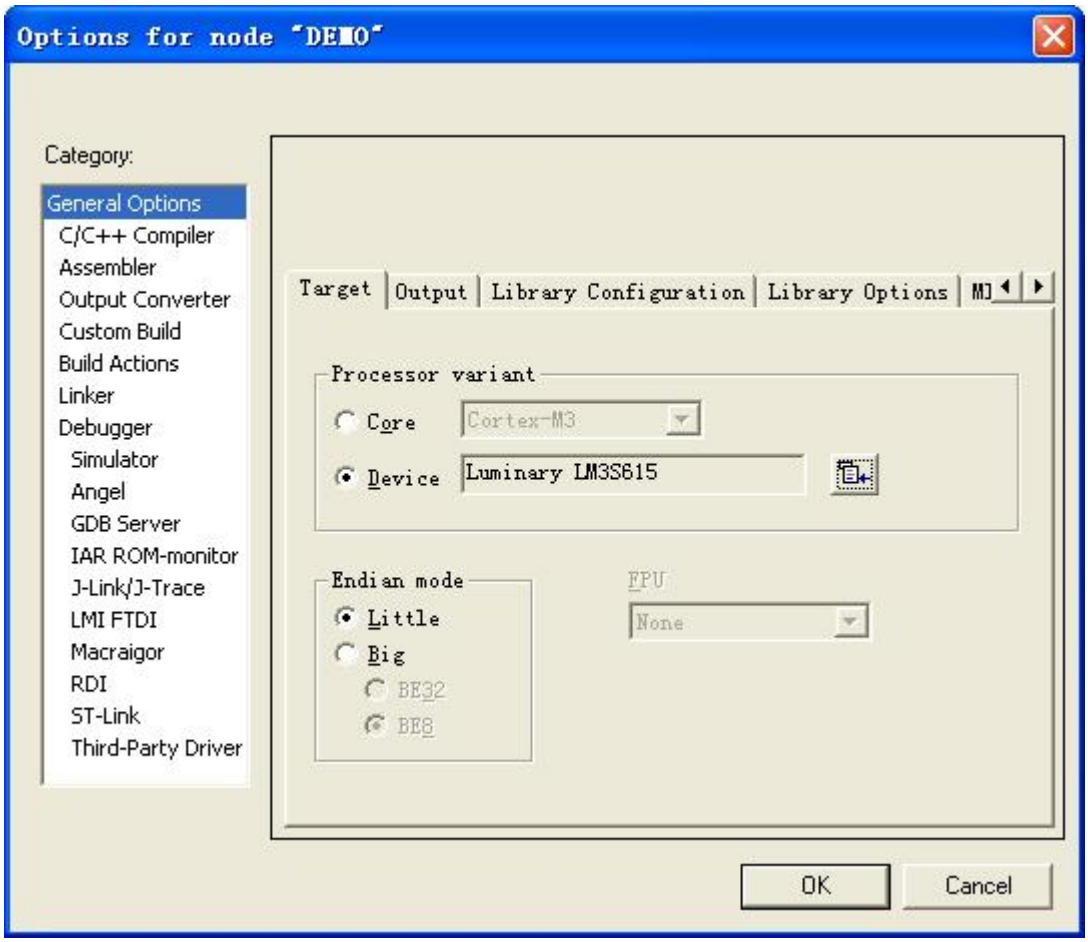

图 3.13 General Option 选项设置

3.7 C/C++编译器选项设置

在Options 窗口的目录Category 中选择第二项C/C++ Compiler。C/C++编译器的选项设置如下：

□ Preprocessor 设置

Preprocessor 页面中，列有标准的include 文件的目录。如果用户的include 文件不在标准目录下时，必须在Additional include directories 输入包含该项目include 文件的目录。一个目录用一行描述，有多个目录时允许用多行。在Preprocessor 框中的Additional include directories(one per line) 项目中输入“$TOOLKIT_DIR$\INC\Luminary”，前面的拷贝库文件目的就在此。如图 3.14 所示。

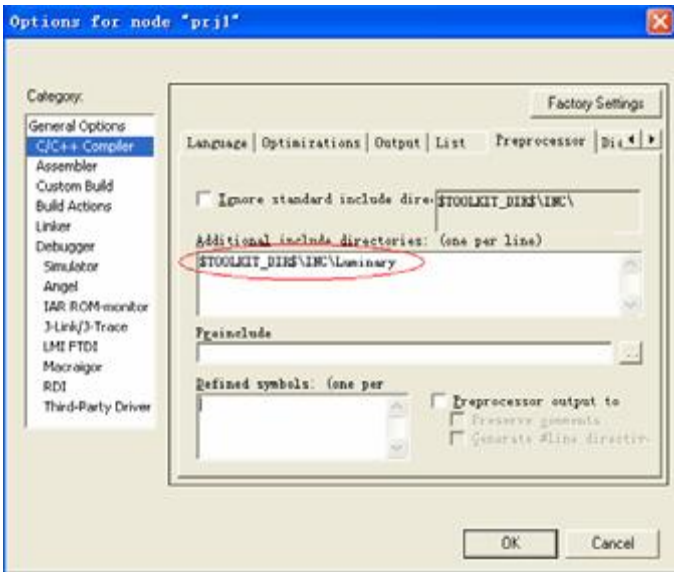

图 3.14 C/C++编译器选项设置

其它的选项采用默认值。

3.7.1 Assembler 选项设置

在Options 窗口的目录Category 中选择第三项Assembler。汇编器的选项设置采用默认设置。

3.7.2 Linker 选项设置

在Options 窗口的目录Category 中选择第一项Config。

□ Config 设置

主要是定义连接器命令文件（Linker Command File）。这是连接器选项中最重要同时也是最复杂的设置。连接器命令文件中包含连接器的各项命令行参数，主要用于控制程序各种代码段和数据段在存储器中如何分布。用户一定要吃透和掌握如何生成正确的连接器命令文件。为了帮助初学者理解，我们增加了下面一段介绍。用户会采用不同半导体厂家的产品，每种芯片内部SRAM 和FLASH 的大小和地址分布都不同，另外用户目标系统配置的外部存储器也不同，用户应用软件要求的存储器分配也不相同。以上所有的不同最后落实到在运行时不同的代码段和数据段的存储器地址分配方案。而这种运行时存储器分配必需在连接器命令文件中说明，并由连接器IAR XLINK 生成。经XLINK 连接生成的代码文件下载到目标板时的地址，由FlashLoader 执行，后面将介绍。IAR EWAR 提供默认的连接器命令文件，它在IAR EWAR 安装目录的ARM\config 目录下，名字叫lnkarm.xcl。但是默认的连接器命

令文件lnkarm.xcl 不能完全适用特定的目标系统，必须加以修改。standalone.xcl 为LM3S 系列MCU 在EWARM 集成开发环境下的连接器命令文件。之前我们把standalone.xcl 文件拷贝到默认的ARM\config 目录下，并命名为“lnk_LM3.xcl”，就是为了这一步很方便的选择lnk_LM3.xcl。在Link Comamnd file 中，选中Override default，点击右边选择按钮，打开选项选择lnk_LM3.xcl。如图 3.18 所示。

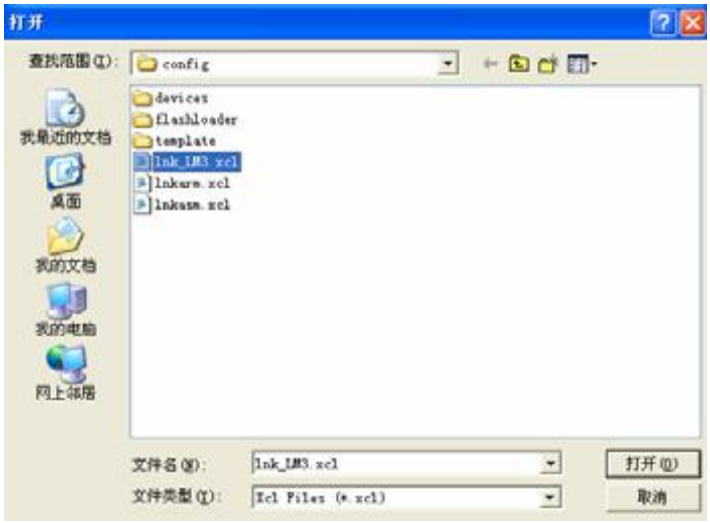

图 3.18 选择**standalone.xcl**

在Entry lab 输入 ResetISR，如图 3.19 所示。

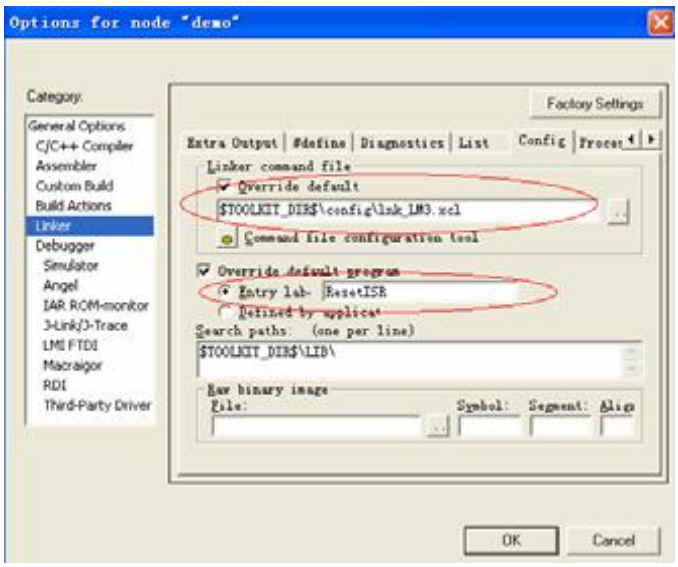

图 3.19 Linker 选项的Config 设置

注：ResetISR 为启动文件startup_ewarm.c 中程序复位时的入口。

3.7.3 Debugger 选项设置

在Options 窗口的目录Category 中选择第七项Debugger。调试器的选项设置如下：

□ Setup 页面设置

本项选择所用的调试工具，我们选择的是**J-Link/J-Trace**，如图 3.20 所示。

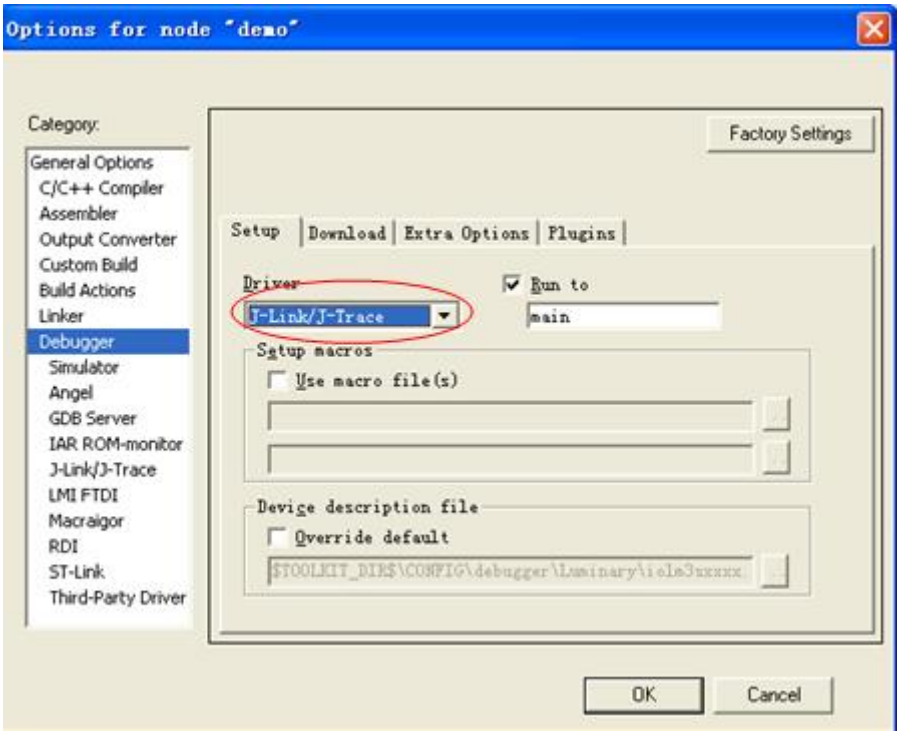

图 3.20 调试工具选择

□ Download 页面设置

选择 Verify download 和 use flash load。如图 3.21 所示。

要进行应用程序的调试，必须将生成的 demo.d79 文件下载到目标系统MCU 的Flash或RAM中。调试器C-SPY 是通过一个叫做Flash Loader 的程序完成下载任务的。FlashLoader的详细工作原理以及它和C-SPY 的互动机理我们不在这里介绍，用户可以参阅IAR 的FlashLoader Guide。前面我们在设置General Options 选项时，已经指定目标MCU 是LM3Sx9xx。所以EWARM 已经提供了该芯片默认的Flash Loader。如果用户选用的MCU 不在EWARM的Device 清单中，那就必须自己去编写该芯片的Flash Loader 了。由于我们使用的EWARM提供的LM3SXXXX 芯片默认的Flash Loader，按Download 页面(图 3.21)中的Edit 按钮，在弹出的Flash Loader Overview 对话框（如图 3.22 所示），选中 default ，按OK 即可。

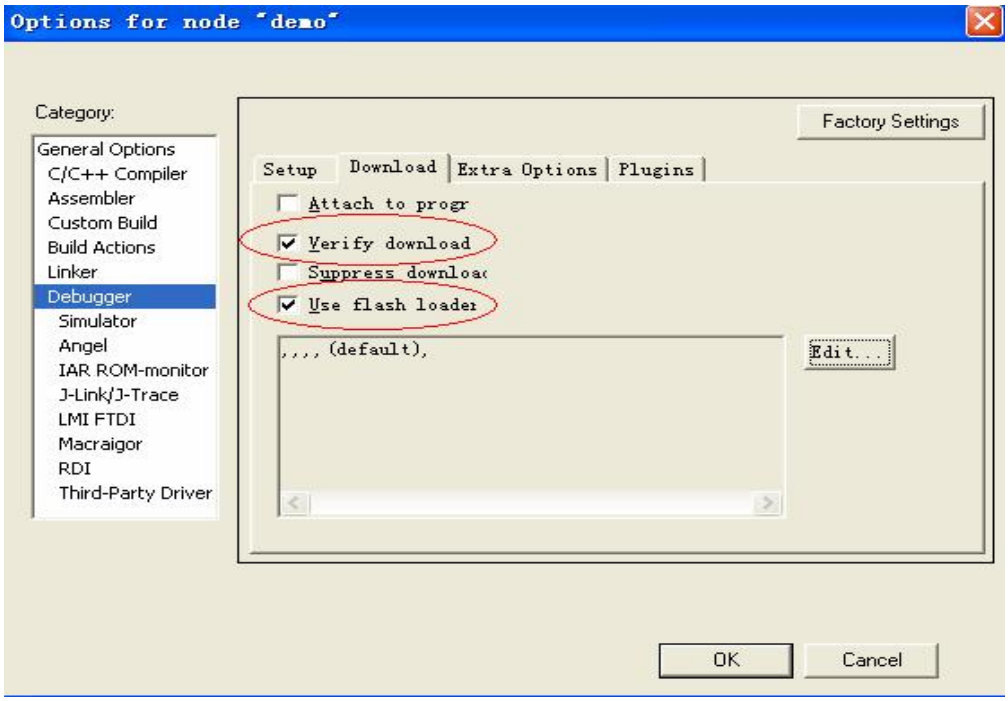

图 3.21 下载程序选项设置

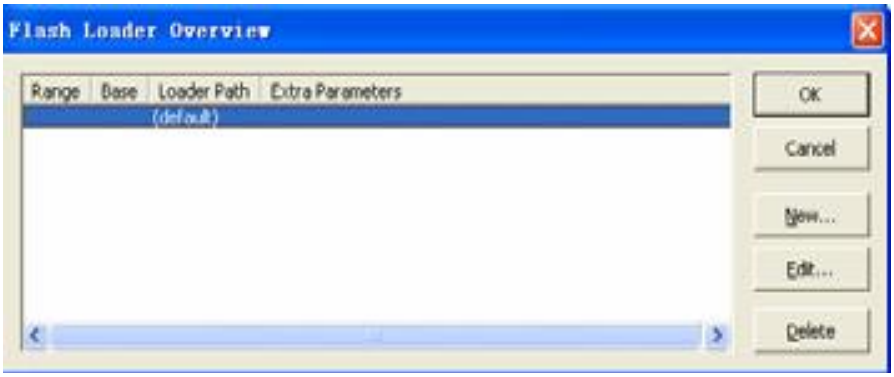

图 3.22 Flash Loader 设置

注：到此，工程已经建好，各项设置也完成了。

第 4 章 编译和运行应用程序

按上述步骤完成所有的工程设置以后就可以开始编译程序了。

4.1 编译连接处理

选择主菜单Project > Make，或选中工作区中的项目名demo– Debug，按鼠标右键在弹出菜单中选择Make。如果你想重新编译所有的文件，选择主菜单Project > Rebuild All，或选中工作区中的项目名demo – Debug，按鼠标右键在弹出菜单中选择Rebuild All。EWARM 将执行编译连接处理，生成可调试代码文件。Build 消息窗口中将显示连接处理的消息。连接的结果将生成一个带调试信息的代码文件demo.d79 和一个存储器分配（MAP）文件demo.map。从编译连接后的工作区窗口中树结构中，我们可以看到每个源文件访问关联了哪些头文件，同时生成了哪些输出文件。因为我们在建立新项目时选择Debug 配置，所以在DEMO目录下自动生成一个Debug 子目录。Debug 子目录下又包含另3 个子目录，名字分别为List、Obj、Exe。在Obj 目录下后缀为.r79 的文件，用作IAR XLINK 连接器的输入文件。在Exe 目录下后缀为.d79 的文件，用作IAR C-SPY 调试器的输入文件，注意在执行连接处理之前这个目录是空的。

4.2 查看 MAP 文件

双击Workspace 中的demo.map 文件名，编辑器窗口中将显示该MAP 文件。从MAP 文件中我们可以了解以下内容：

 文件头中显示连接器版本，输出文件名以及连接命令使用的选项。

 CROSS REFERENCE 部分显示程序入口地址。

 RUNTIME MODEL 部分显示使用的运行时模块的属性。

 MODULE MAP 部分显示所有被连接的文件。每个文件中，作为应用程序一部分加载的有关模块的信息，包括各段和每个段中声明的全局符号都列出来。

 SEGMENTS IN ADDRESS ORDER 部分列出了组成应用程序的所有段的起始地址和结束地址，字节数，类型和对齐标准等。

 END OF CROSS REFERENCE 部分显示总的代码和数据字节数。如果编译连接没有

任何错误，则生成demo.d79 应用程序代码，并可以用于在IAR C-SPY中调试。

4.3 加载应用程序

选择主菜单Project > Debug 或工具条上的Debugger 按钮或者按键CTL+D，C-SPY 将开始装载demo.d79。屏幕上将显示PC 机通过 LM LINK 加载的过程。屏幕上除了已经原先已经打开的窗口外，将显示一组C-SPY 专用窗口。如Debug Log和Disassembly 窗口。如图 4.1 所示。

注意：如果在下载程序时，有提示信息出现，直接选择“否”就可以了。

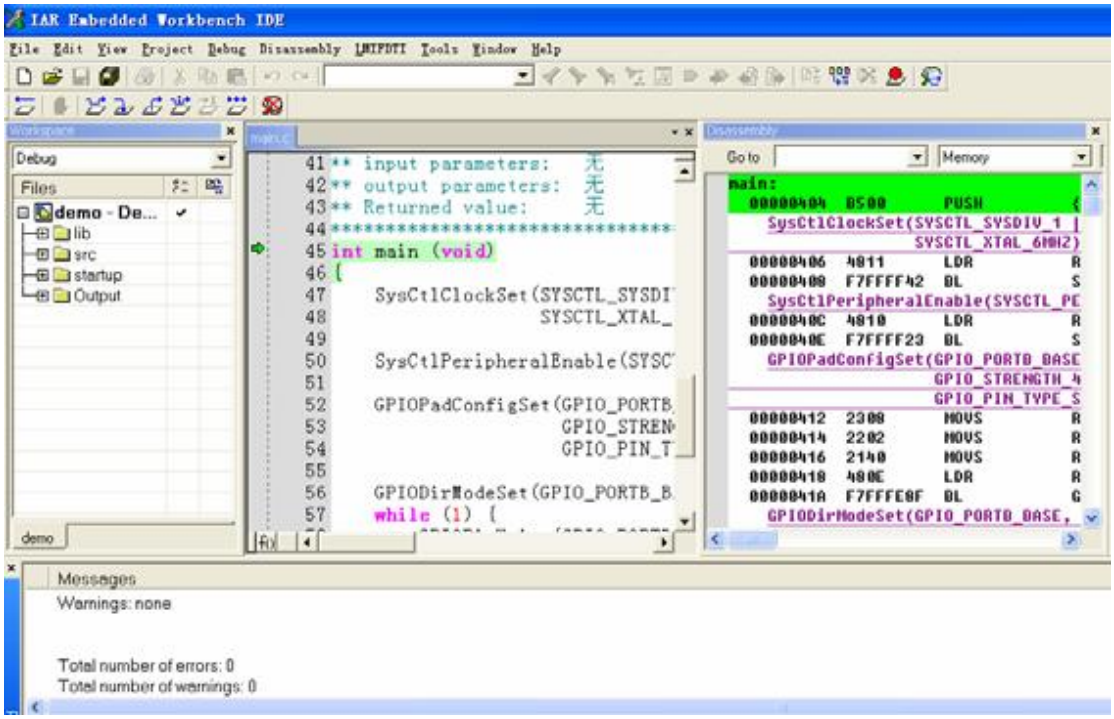

图 4.1 EWARM 的Debug 窗口

注：到此，程序已经下载到Flash，也可以进行程序的调试了。

第 5 章 生成 hex 文件

在有些场合需要生成**.hex** 文件，因此，这里介绍在IAR 中如何生成**.hex** 文件。

□ 生成方法

在Options 窗口的目录Category 中选择第六项Linker。在Output 选项中，选中Format复选框的Other，然后在Output 下拉菜单中选择msd-i，如图 5.1 所示。

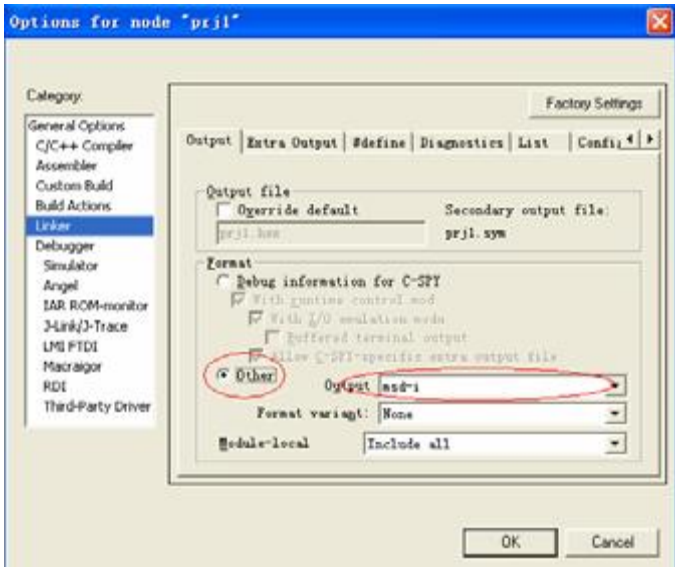

图 5.1 hex 生成选项设置

注意：在选择了Format 复选框的Other 后，在C-SPY 调试器下的Debug information forC-SPY 的调试信息不可用，在这里只生成了.hex 的输出文件。在Output 下选择msd-i 还会生成第二输出文件.sym 文件，如果需要生成其它输出文件，则可以选择Output 下的其它选项。其选项内容的解释可以参见“C:\Program Files\IAR Systems\Embedded Workbench 5.30Kickstart\common\doc”下的xlink.ENU.pdf 文档。

□ 生成结果

生成的**.hex** 文件在工程的Debug\Exe 下，如图 5.2 所示。

图 5.2 生成的.hex 文件__