ADS 集成开发环境及 EasyJTAG 仿真器应用

ADS 集成开发环境是 ARM 公司推出的 ARM 核微控制器集成开发工具，英文全称为 ARM Developer Suite，成熟版本为 ADS1.2。ADS1.2 支持 ARM10 之前的所有 ARM 系列微控制器，支持软件调试及 JTAG 硬件仿真调试，支持汇编、C、C++源程序，具有编译效率高、系统库功能强等特点，可以在 Windows98、Windows XP、Windows2000 以及 RedHat Linux 上运行。

这里将简单介绍使用 ADS1.2 建立工程，编译连接设置，调试操作等等。最后还介绍了基于 LPC2100 系列 ARM7 微控制器的工程模板的使用，EasyJTAG 仿真器的安装与使用。

一、ADS 1.2 集成开发环境的组成

ADS 1.2 由 6 个部分组成，如表 1 所示。

表 1 ADS 1.2 的组成部分

名称	描述	使用方式
代码生成工具	ARM 汇编器， ARM 的 C、C++编译器， Thumb 的 C、C++编译器， ARM 连接器	由 CodeWarrior IDE 调用
集成开发环境	CodeWarrior IDE	工程管理，编译连接
调试器	AXD， ADW/ADU， armsd	仿真调试
指令模拟器	ARMulator	由 AXD 调用
ARM 开发包	一些底层的例程， 实用程序(如 fromELF)	一些实用程序由 CodeWarrior IDE 调用
ARM 应用库	C、C++函数库等	用户程序使用

由于用户一般直接操作的是 CodeWarrior IDE 集成开发环境和 AXD 调试器，所以这一章我们只介绍这两部分软件的使用，其它部分的详细说明参考 ADS 1.2 的在线帮助文档或相关资料。

1. CodeWarrior IDE 简介

ADS 1.2 使用了 CodeWarrior IDE 集成开发环境，并集成了 ARM 汇编器、ARM 的 C/C++编译器、Thumb 的 C/C++编译器、ARM 连接器，包含工程管理器、代码生成接口、语法敏感(对关键字以不同颜色显示)编辑器、源文件和类浏览器等等。CodeWarrior IDE 主窗口如图 1 所示。

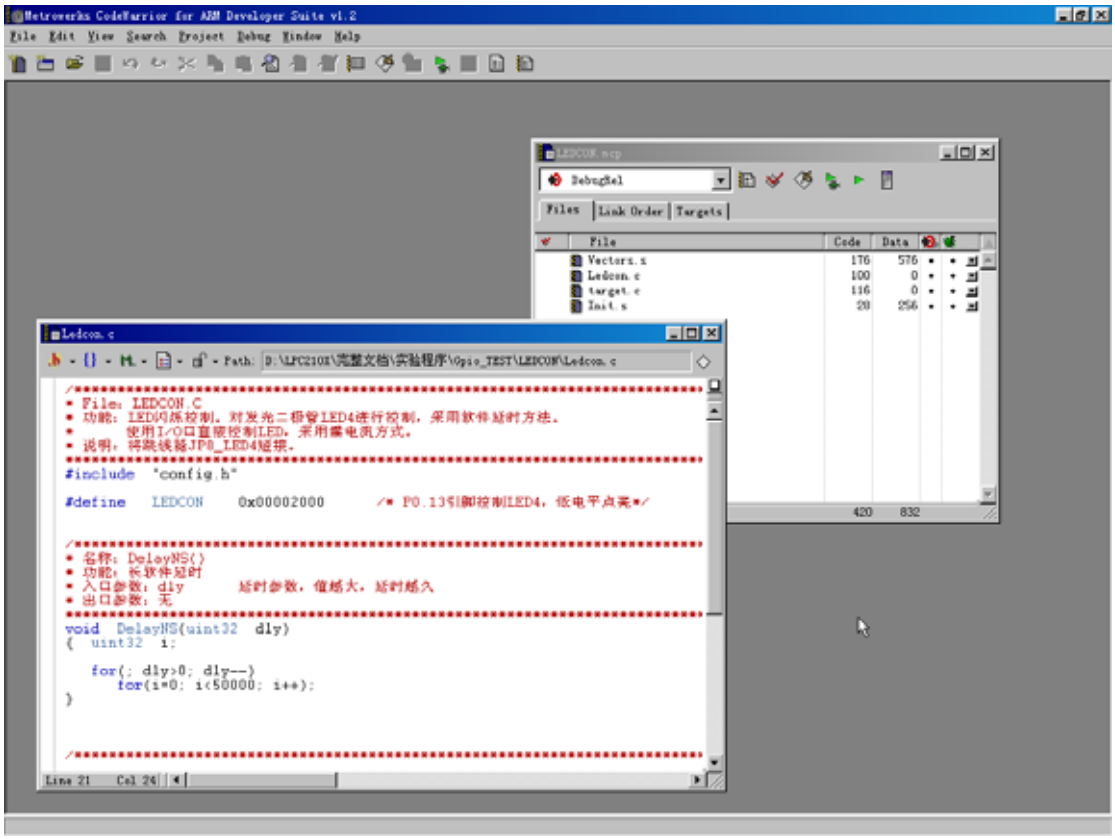

图 1 CodeWarrior 开发环境

2. AXD 调试器简介

AXD 调试器为 ARM 扩展调试器(即 ARM eXtended Debugger)，包括 ADW/ADU 的所有特性，支持硬件仿真和软件仿真(ARMulator)。AXD 能够装载映像文件到目标内存，具有单步、全速和断点等调试功能，可以观察变量、寄存器和内存的数据等等。AXD 调试器主窗口如图 2 所示。

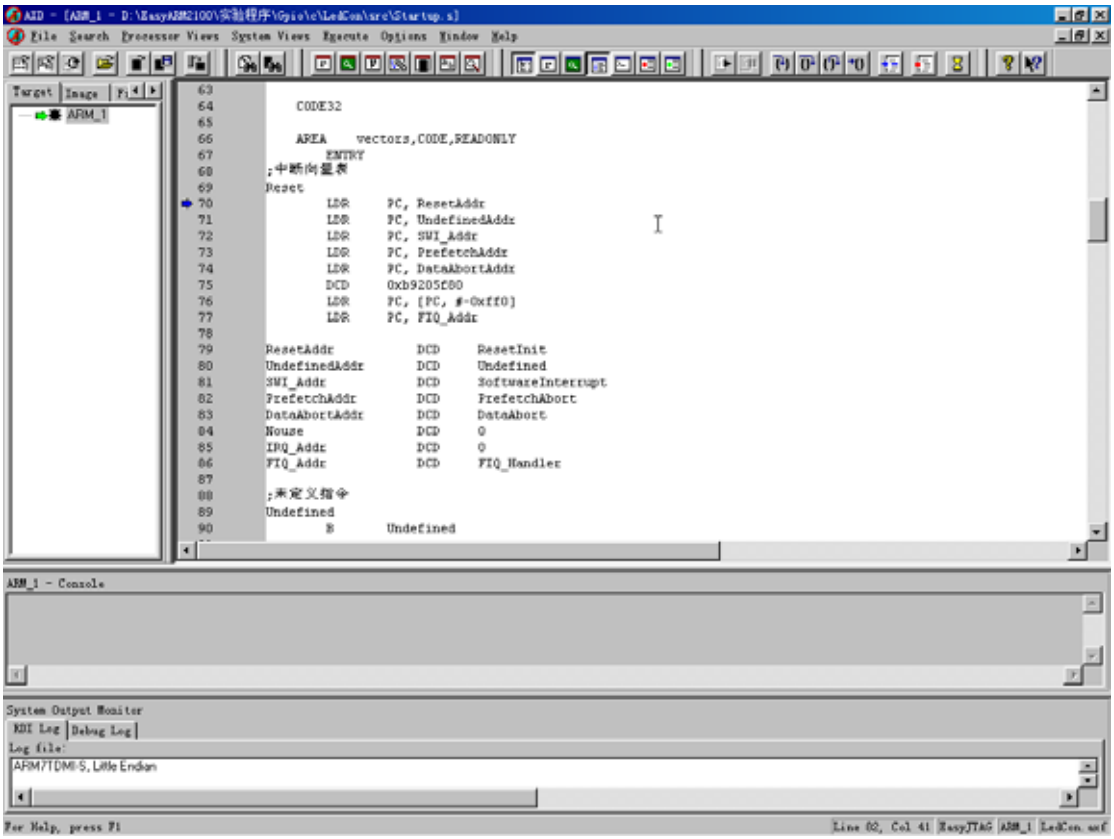

图 2　AXD 调试器

二、工程的编辑

1. 建立工程

点击 WINDOWS 操作系统的【开始】->【程序】->【ARM Developer Suite v1.2】->【CodeWarrior for ARM Developer Suite】起动 Metrowerks CodeWarrior，或双击“ CodeWarrior for ARM Developer Suite ”快捷方式起动。启动 ADS1.2 IDE 如图 3 所示。

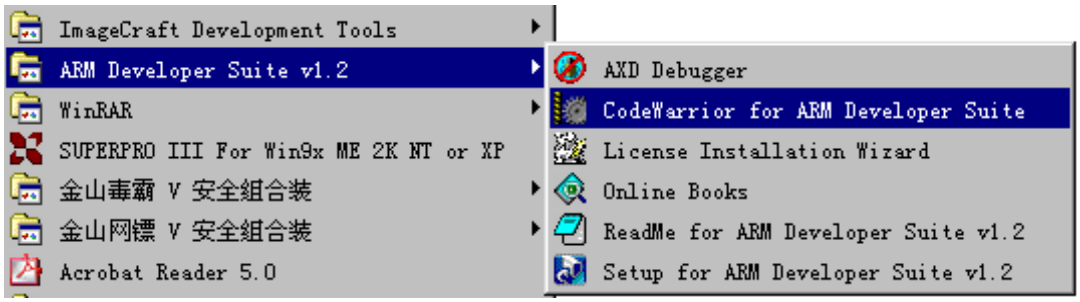

图 3　启动 ADS1.2 IDE

点击【File】菜单，选择【New…】即弹出 New 对话框，如图 4 所示。

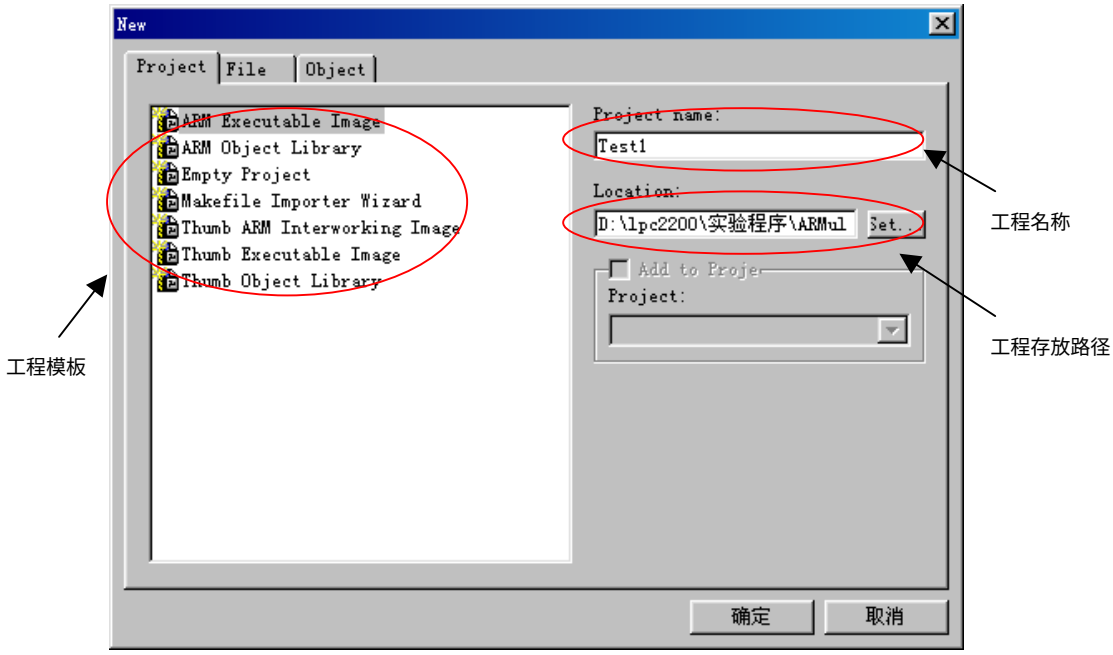

图 4 New 对话框

选择工程模板为 ARM 可执行映象（ARM Executable Image）或 Thumb 可执行映象(Thumb Executable Image)，或 Thumb、ARM 交织映象(Thumb ARM Interworking Image)，然后在【Location】项选择工程存放路径，并在【Project name】项输入工程名称，点击【确定】按钮即可建立相应工程，工程文件名后缀为 mcp(下文有时也把工程称为项目) 。

2. 建立文件

建立一个文本文件，以便输入用户程序。点击“ New Text File ”图标按钮，如图 5 所示。

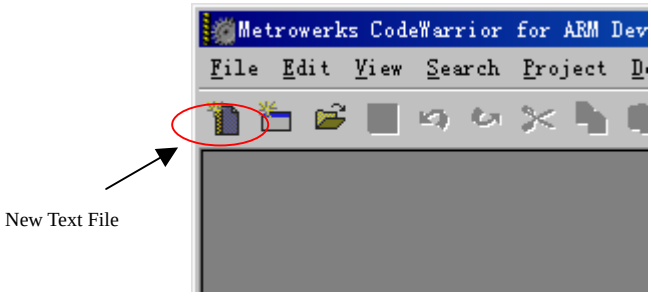

图 5 “ New Text File ”图标按钮

然后在新建的文件中编写程序，点击“ Save ”图标按钮将文件存盘(或从【File】菜单选择【Save】)，输入文件全名，如 TEST1.S。注意，请将文件保存到相应工程的目录下，以便于管理和查找。

当然，您也可以 New 对话框选择【File】页来建立源文件，如图 4 所示，或使用其它文本编辑器建立或编辑源文件。

3. 添加文件到工程

如图 6 所示，在工程窗口中【Files】页空白处右击鼠标，弹出浮动菜单，选择“ Add Files… ”即可弹出“ Select files to add… ”对话框，选择相应的源文件(可按着 Ctrl 键一次选择多个文件)，点击【打开】按钮即可。

另外，用户也可以在【Project】菜单中选择【Add Files…】来添加源文件，或使用 New

对话框选择【File】页来建立源文件时选择加入工程(即选中“Add to Project”项)。添加文件操作如图 6、图 7 所示。

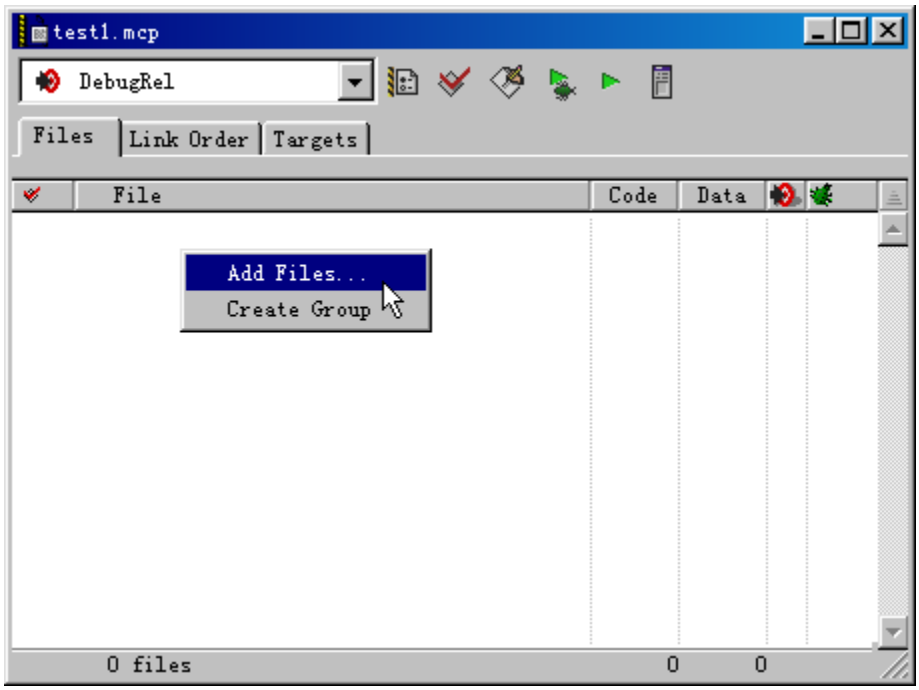

图 6 在工程窗口中添加源文件

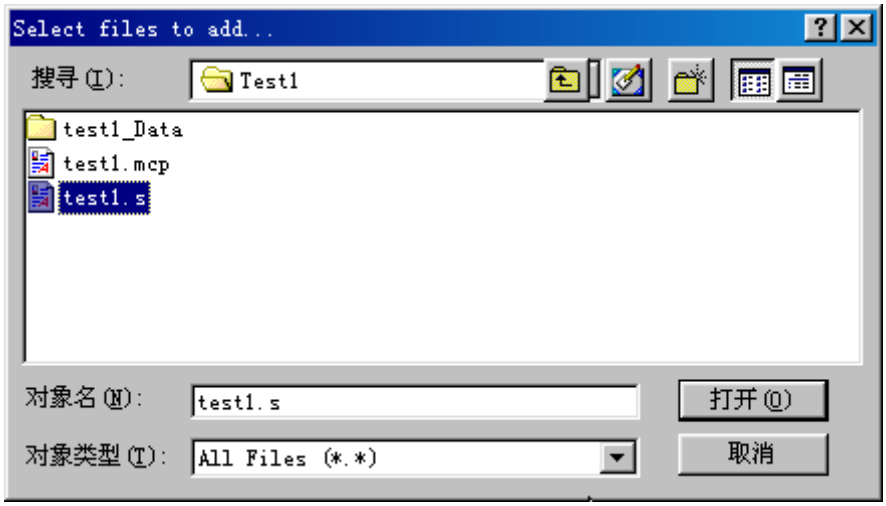

图 7 Select files to add…对话框

4. 编译连接工程

如图 8 所示为工程窗口中的图标按钮，通过这些图标按钮，您可以快速的进行工程设置、编译连接、启动调试等等(在不同的菜单项上可以分别找到对应的菜单命令)。它们从左至右分别为：

DebugRel Settings…	**工程设置，如地址设置、输出文件设置、编译选项等，其中 DebugRel 为当前的生成目标(target system)。**
Synchronize Modification Dates	同步修改日期，检查工程中每个文件的修改日期，若发现有更新(如使用其它编辑器编辑源文件)，则在

	Touch 栏标记“√”。
Make	**编译连接(快捷键为 F7)。**
Debug	**启动 AXD 进行调试(快捷键为 F5)。**
Run	启动 AXD 进行调试，并直接运行程序。
Project Inspector	工程检查，查看和配置工程中源文件的信息。

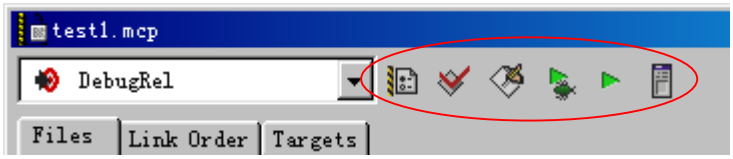

图 8 工程窗口中的图标按钮

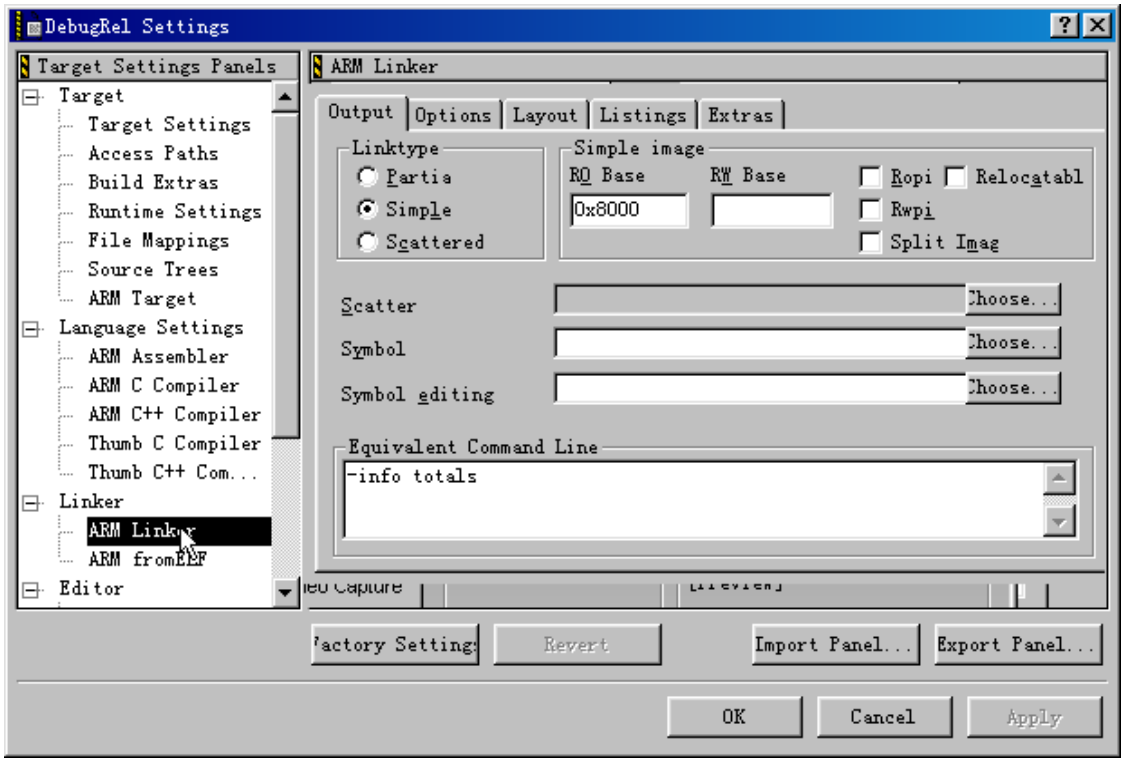

图 9 DebugRel Settings 窗口

点击“DebugRel Settings…”图标按钮，即可进行工程的地址设置、输出文件设置、编译选项等，如图 9 所示。在“ARM Linker”对话框设置连接地址，在“Language Settings”中设置各编译器的编译选项。

对于简单的软件调试，可以不进行连接地址的设置，直接点击工程窗口的“Make”图标按钮，即可完成编译连接。**若编译出错，会有相应的出错提示，双击出错提示行信息，编辑窗即会使用光标指出当前出错的源代码行**，编译连接输出窗口如图 10 所示。同样，您可以在【Project】菜单中找到相应的命令。

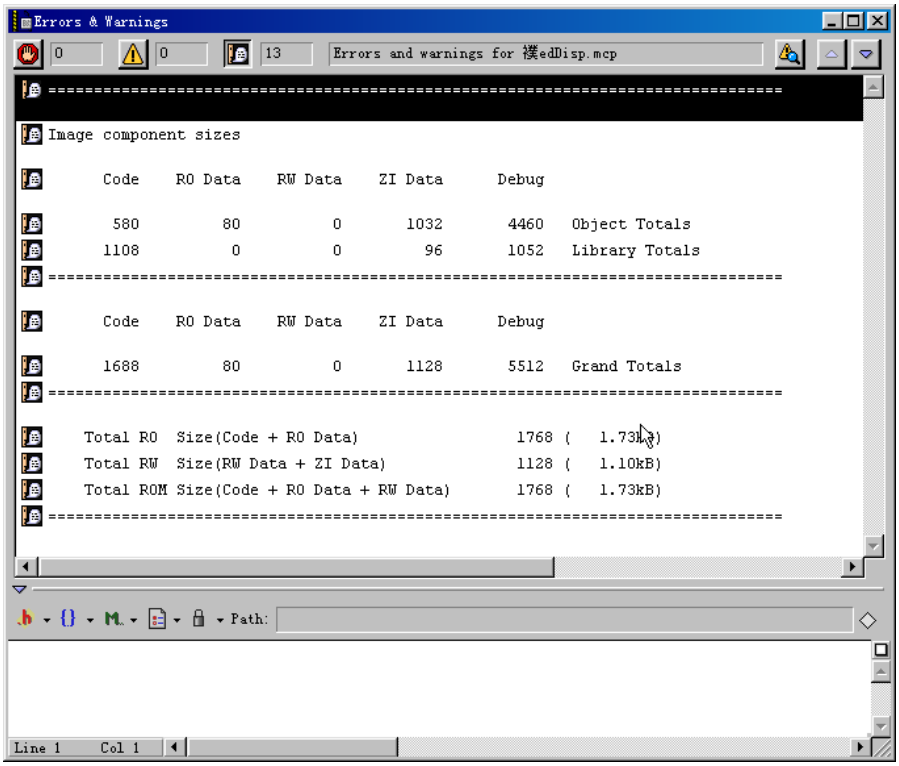

图 10 编译连接输出窗口

如图 11 所示，Touch 栏用于标记文件是否已编译，若打上“√”则表明对应文件需要重新编译。Touch 栏用于标记文件是否已编译，若打上“√”则表明对应文件需要重新编译。可以通过单击该栏位置来设置/取消符号“√”，或将工程目录下的*.tdt 文件删除也可以使整个工程源文件均打上“√”。

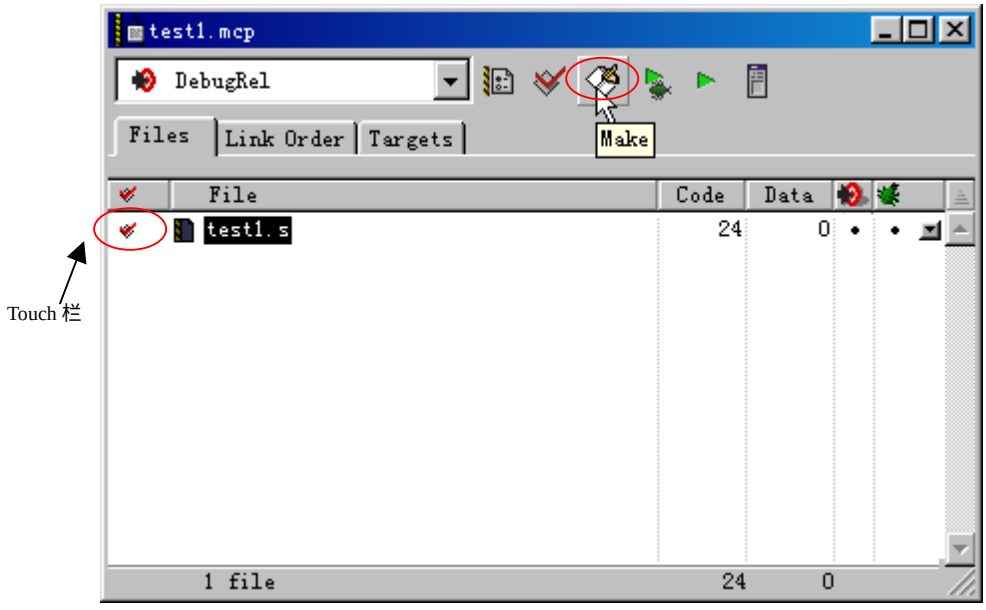

图 11 工程窗口中 Make 操作

5. 打开旧工程

点击【File】菜单，选择【Open…】即弹出“打开”对话框，找到相应的工程文件(*.mcp)，单击【打开】即可。在工程窗口的【Files】页中，双击源程序的文件名即可打开该文件进行

编辑。

三、工程的调试

1. 选择调试目标

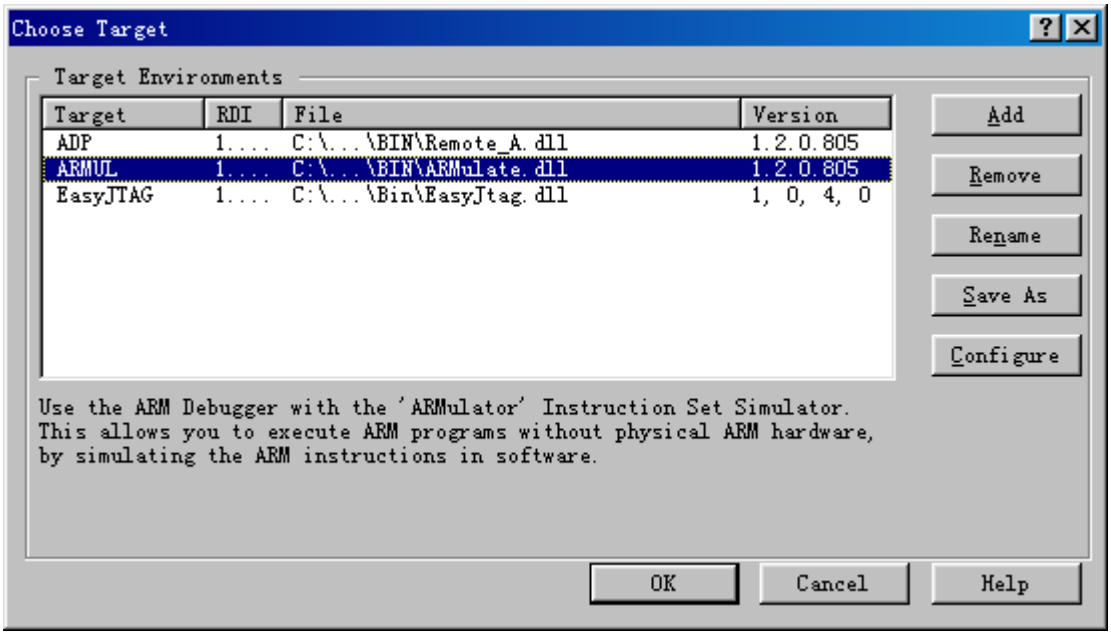

图 12 Choose Target 窗口

当工程编译连接通过后，在工程窗口中点击“ Debug ”图标按钮，即可启动 AXD 进行调试(也可以通过【开始】菜单起动 AXD)。点击菜单【Options】选择【Configure Target…】，即弹出 Choose Target 窗口，如图 12 所示。在没有添加其它仿真驱动程序前，Target 项中只有两项，分别为 ADP(JTAG 硬件仿真)和 ARMUL(软件仿真)。

选择仿真驱动程序后，点击【File】选择【Load Image…】加载 ELF 格式的可执行文件，即*.axf 文件。说明：当工程编译连接通过后，在“ 工程名\工程名_Data\当前的生成目标 ”目录下就会生成一个*.axf 调试文件。比如工程 TEST，当前的生成目标 Debug，编译连接通过后，则在…\TEST\TEST_Data\Debug 目录下生成 TEST.axf 文件。

2. 调试工具条

AXD 运行调试工具条如图 13 所示，调试观察窗口工具条如图 14 所示，文件操作工具条如图 15 所示。

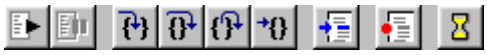

图 13 运行调试工具条

 全速运行(Go)

 停止运行(Stop)

 单步运行(Step In)，与 Step 命令不同之处在于对函数调用语句，Step In 命令将进入

该函数。

单步运行(Step)，每次执行一条语句，这时函数调用将被作为一条语句执行。

单步运行(Step Out)，执行完当前被调用的函数，停止在函数调用的下一条语句。

运行到光标(Run To Cursor)，运行程序直到当前光标所在行时停止。

设置断点(Toggle BreakPoint)

图 14 调试观察窗口工具条

打开寄存器窗口(Processor Registers)

打开观察窗口(Processor Watch)

打开变量观察窗口(Context Variable)

打开存储器观察窗口(Memory)

打开反汇编窗口(Disassembly)

图 15 文件操作工具条

加载调试文件(Load Image)

重新加载文件(Reload Current Image)。由于 AXD 没有复位命令，所以通常使用 Reload 实现复位(直接更改 PC 寄存器为零也能实现复位)。

四、LPC2100 系列 ARM7 微控制器工程模板

在第二节介绍新建立工程时，我们已经接触了 ADS1.2 提供的几个标准工程模板，使用各个模板建立的工程，它们的各项设置均有不同之处，方便生成不同结构的代码，如 ARM 可执行映象（生成 ARM 指令的代码）或 Thumb 可执行映象(生成 Thumb 指令的代码)，或 Thumb、ARM 交织映象(生成 Thumb、ARM 指令交织的代码)。

针对 LPC2100 系列 ARM7 微控制器，我们定义了 6 个工程模板，这些模板一般包含的设置信息有 FLASH 起始地址 0x00000000、片内 RAM 起始地址 0x40000000、编译连接选项及编译优化级别等等 模板中包含了 LPC2100 系列 ARM7 微控制器的起动文件，包括 IRQ.S、STARTUP.S、TARGET.C；模板还包含了 LPC2100 系列 ARM7 微控制器的头文件(如：LPC2294.h 和 LPC2294.inc，LPC2294 的寄存器是向下兼容的)，分散加载描述文件(如：mem_a.scf、mem_b.scf、mem_c.scf)等等。

1. 为 ADS1.2 增加 LPC2100 专用工程模板

将工程模板的所有目录（工程模板可以在 http://www.zlgmcu.com/tools/kaifaban/EasyARM2100.asp 网页下载得到）拷贝到“ <ADS1.2 安装目录>\Stationery\ ”即可，操作如图 16 和图 17 所示。这个步骤只需 1 次，以后就可以直接使用工程模板了。

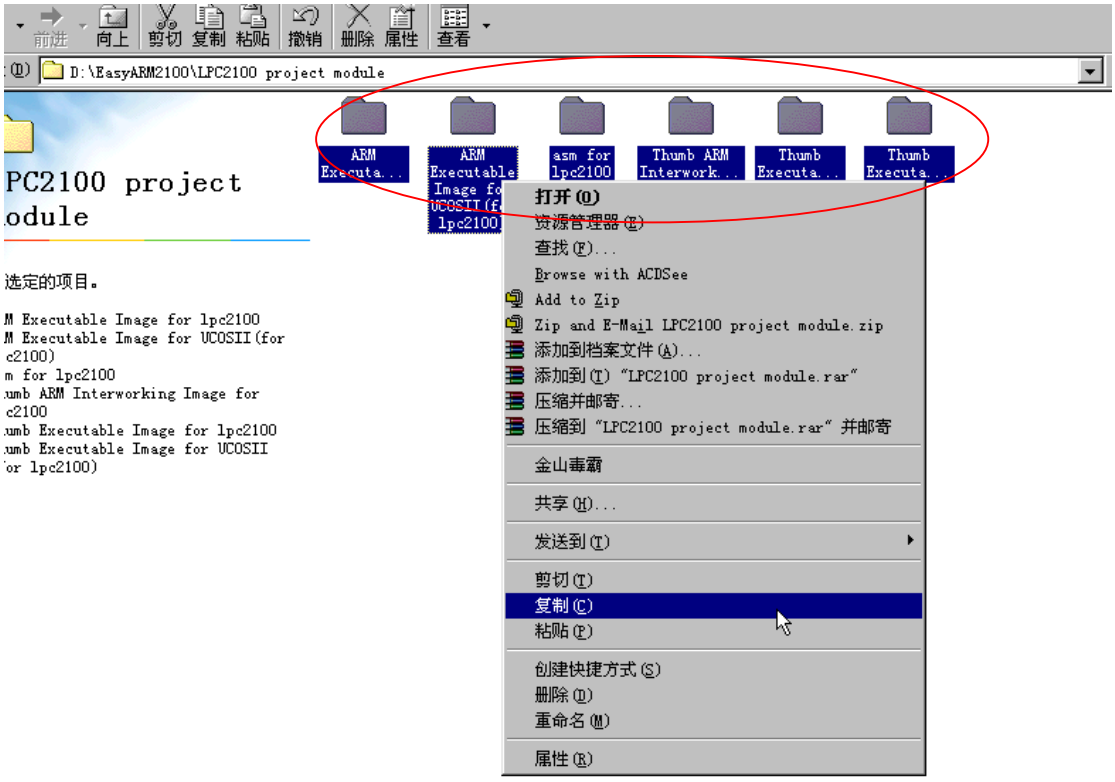

图 16 选择拷贝的文件和目录

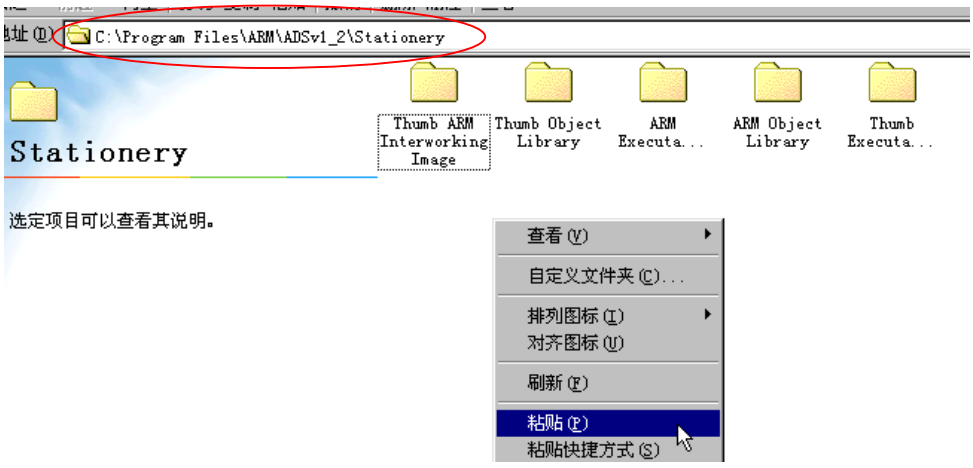

图 17 复制文件目录

2. 使用 LPC2100 专用工程模板建立工程

启动 ADS1.2 IDE，点击【File】菜单，选择【New...】即弹出 New 对话框，如图 18 所示。由于事先增加了 LPC2100 专用工程模板，所以在工程模板栏中多出几项工程模板选项。

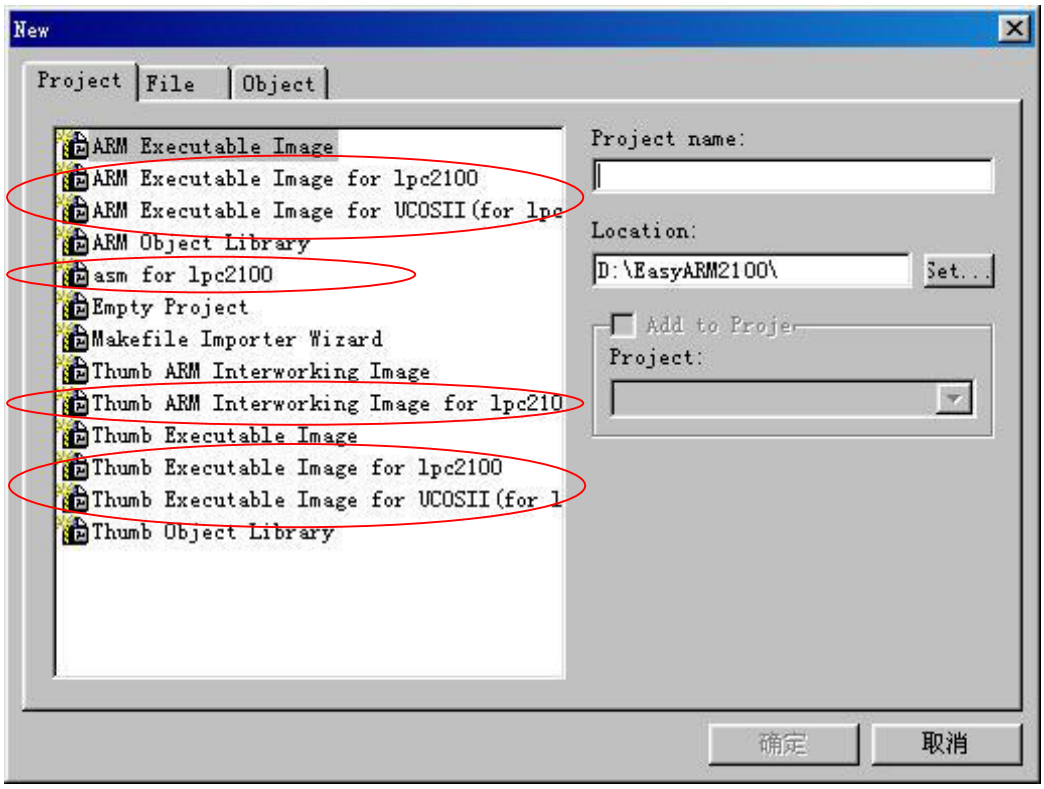

图 18 增加的工程模板

其中：

ARM Executable Image for lpc2100 ： 无操作系统时所有 C 代码均编译成 ARM 指令的工程模板。

asm for lpc2100 ：汇编程序工程模板。

Thumb ARM Interworking Image for lpc2100 ：无操作系统时部分 C 代码编译为 ARM 指令，部分 C 代码编译为 Thumb 指令的工程模板。

Thumb Executable Image for lpc2100 ：无操作系统时所有 C 编译成 Thumb 指令的工程模板。

ARM Executable Image for UCOSII(for lpc2100) ：所有 C 代码均编译为 ARM 指令的 µC/OS-II 工程模板

Thumb Executable Image for UCOSII(for lpc2100) ：部分 C 代码编译为 ARM 指令，部分 C 代码编译为 Thumb 指令的µC/OS-II 工程模板（使用µC/OS-II 时，不可能所有代码均编译成 Thumb 指令）。

用户选择相应的工程模板建立工程，如图 19 所示为使用 ARM Executable Image for lpc2100 工程模板建立的一个工程。工程有三个生成目标（target system）：DebugInRAM、DebugInFLASH 和 RelInFLASH，它们的配置如表 2 所示。工程模板已经将相应的编译参数设置好了，可以直接使用即可。

注意：选用 RelInFLASH 目标时，将会对 LPC2100 芯片(除 LPC2106/2105/2104 外)进行加密。加密的芯片只能使用 ISP 进行芯片全局擦除后，才能恢复 JTAG 调试及 ISP 读/写操作。

注意：模板是按片内 RAM 为 16K 字节设计的，对于 LPC2106/2105 芯片，需要更改堆栈区(STACKS)起始地址为 0x40010000/0x40008000(文件 mem_a.scf、mem_b.scf 和 mem_c.scf)；使用 DebugInRAM 目标时，可以更改变量区(IRAM)起始地址(文件 mem_b.scf)，比如 0x40006000，以获得合适的执行区大小(即代码区)。

表 2 LPC2100 专用工程模板 target system 配置

target system	分散加载描述文件	调试入口点地址	C 优化等级	应用说明
DebugInRAM	mem_b.scf	0x40000000	Most	片内 RAM 调试模式，程序在片内 RAM 中
DebugInFLASH	mem_c.scf	0x00000000	Most	片内 FLASH 调试模式，程序在片内 FLASH 中
RelInFLASH	mem_a.scf	0x00000000	Most	片内 FLASH 工作模式，程序在片内 FLASH 中

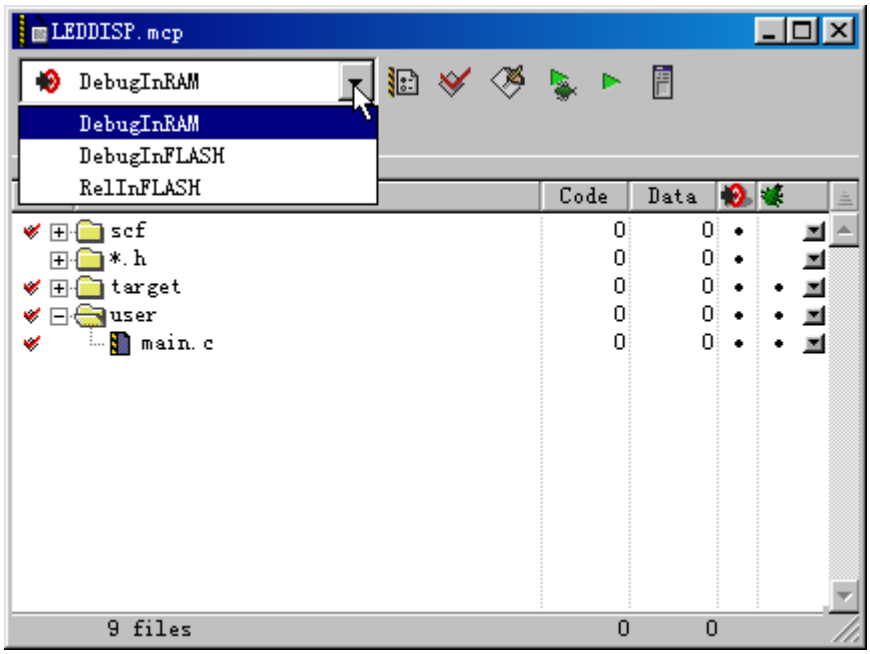

图 19 用 LPC2100 专用工程模板建立的工程

五、EasyJTAG 仿真器的安装与应用

EasyJTAG 仿真器是广州周立功单片机发展有限公司开发的 LPC2000 系列 ARM7 微控制器的 JTAG 仿真器，支持 ADS1.2 集成开发环境，支持单步、全速及断点等调试功能，支持下载程序到片内 FLASH 和特定型号的片外 FLASH，采用 ARM 公司提出的标准 20 脚 JTAG 仿真调试接口。其主要特点如下：

采用 RDI 通讯接口，无缝嵌接 ADS1.2 和其它采用 RDI 接口的 IDE 调试环境。

高达 1M 速率的 JTAG 时钟驱动。

采用同步 Flash 刷新技术（synFLASH），同步下载用户代码到 Flash 中，即下即调。

采用同步时序控制技术（synTIME），仿真可靠稳定。

支持 32 位 ARM 指令/16 位 THUMB 指令的混合调试。

增加映射寄存器窗口，方便用户查看/修改寄存器数值。

微型体积设计，方便用户灵活使用。

EasyJTAG 仿真器外观如图 20 所示，其驱动程序可在 http://www.zlgmcu.com/tools/kaifaban/EasyARM2100.asp 网页下载获得)。

图 20 EasyJTAG 仿真器实物外观

1. 安装 EasyJTAG 仿真器

首先，将 EasyJTAG 仿真器的驱动程序(比如 EasyJTAG_drive 目录下的所有文件)复制到 ADS 的 BIN 目录，如 C:\Program Files\ARM\ADSv1_2\BIN。

接着，将 EasyJTAG 仿真器的 25 针接口通过并口延长线与 PC 机的并口连接，将 EasyJTAG 仿真器的 20 针接口通过 20 PIN 连接电缆接到目标板(使用 LPC2000 系列微控制器的目标板)的 JTAG 接口上，然后给目标板供电。LPC2100 系列 ARM7 微控制器 JTAG 电路设计参考图 21，LPC2106/2105/21004 微控制器的 JTAG 接口电路有些不同，如图 22 所示（使用主 JTAG 接口）。

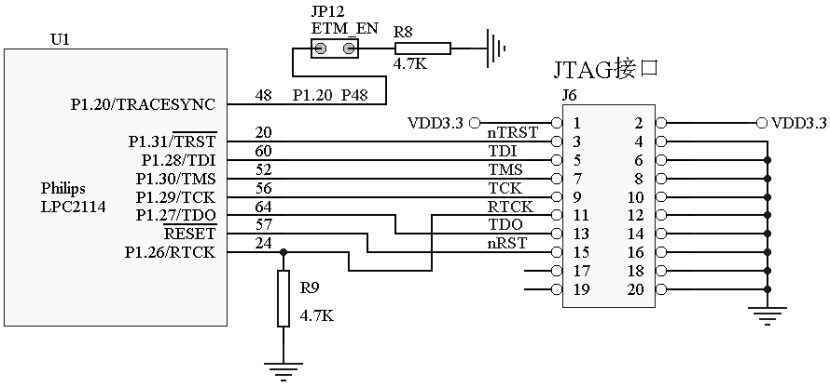

图 21 LPC2100 的 JTAG 电路原理

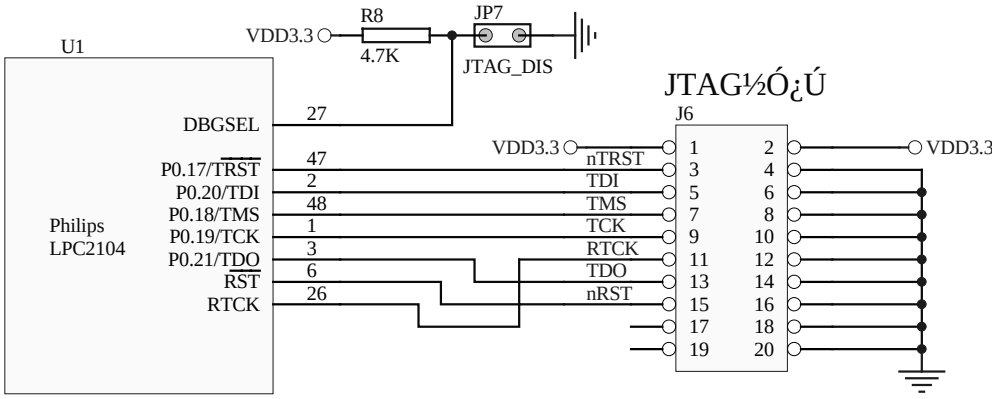

图 22 LPC2104 的 JTAG 电路原理

然后，进入 AXD 调试环境，打开【Options】->【Configure Target...】，弹出 Choose Target 窗口，如图 12 所示。点击“ ADD ”添加仿真器的驱动程序，在添加文件窗口选择如 C:\Program Files\ARM\ADSv1_2\BIN 目录下的 EasyJTAG.dll，点击 “ 打开 ” 即可。

说明：点击 Windows 系统的【开始】->【程序】->【ARM Developer Suite v1.2】->【AXD Debugger】可以直接运行 AXD 软件。

注意：若在添加文件窗口中没有显示 DLL 文件，请设置 WINDOWS 文件浏览窗口的“ 文件夹选项(O)... ”，将查看页中的“ 隐藏文件 ”项选用“ 显示所有文件 ”。

2. 使用 EasyJTAG 仿真器

将计算机并口与 EasyJTAG 仿真器连接，并将仿真器 JTAG 口接头插入目标板的 JTAG 接口上，通过 AXD 软件的设置即可进行仿真调试。

(1) 仿真器设置

在 AXD 调试环境，打开【Options】->【Configure Target...】，弹出 Choose Target 窗口，在“ Target Environments ”框中选择“ EasyJTAG... ”项。

点击“ Configure ”按钮，进入“ EasyJTAG Setup ”设置窗口，见图 23。在"ARMcore"项中选取 CPU 类型，在“ Options ”项中选择 Halt and reset。然后点击“ OK ”，再点击“ OK ”，此时 EasyJTAG 将会进行连接(目标板)的操作。

注意：有时，AXD 会弹出如图 25 所示的错误对话框，或者类似的对话框，此时可以点击“ Connect mode... ”，然后选择“ ATTACH ... ”项确定，再点击“ Restart ”。若 EasyJTAG 正确连接目标板，AXD 代码窗口将显示空白，接下来就可以使用【File】->【Load Image...】加载调试文件，进行 JTAG 调试。

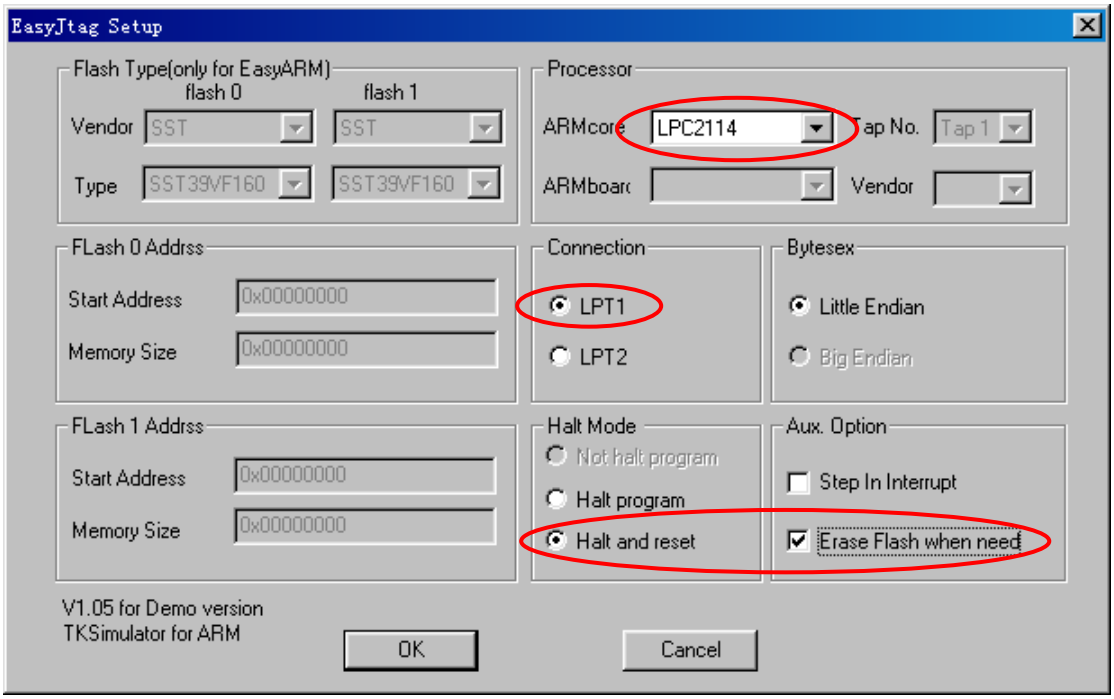

图 23 “ EasyJTAG Setup ”设置窗口

EasyJTAG 设置选项说明：

ARMcore 项，选择 CPU 型号；

Tap No.项，当 CPU 为 LPC2106/2105/2104 时，选择主/从 JTAG 调试口，Tap1 为主，Tap2 为从；

Connection，硬件连接接口选择；

Halt Mode，停机模式选择，包含 Halt program(停止 CPU)和 Halt and reset(复位然后停

止 CPU)两项；

Aux. Option，辅助选项，包含 Step In Interrupt(允许单步运行进入中断)和 Erase Flash when need(允许 EasyJTAG 擦除 Flash)两项；

Flash Type，片外 FLASH 型号选择，可支持两块 FLASH 芯片，当 ARMcore 选择 LPC2200 系列 CPU 时此项才有效。当程序需要下载到片外 FLASH 时，EasyJTAG 仿真器会按所选芯片型号进行擦除/编程。

Flash 0 Addrss，第一块 Flash 的地址设置，当 ARMcore 选择 LPC2200 系列 CPU 时此项才有效。

Flash 1 Addrss，第二块 Flash 的地址设置，当 ARMcore 选择 LPC2200 系列 CPU 时此项才有效。

(2) 仿真器的应用问题

在 ADS1.2 IDE 环境中按 F5 键或 Debug 图标按钮即可直接进入 AXD，但有时会出现如图 24 所示的提示，处理方法是点击“ 确定 ”，然后在弹出的 Load Session 窗口中点击“ 取消 ”。若进入 AXD 后，主调试窗口没有任何代码，且【File】->【Load Image…】菜单项无效时，此时需要重新打开【Options】->【Configure Target...】点击“ OK ”，再点击【File】选择【Load Image…】加载调试文件。

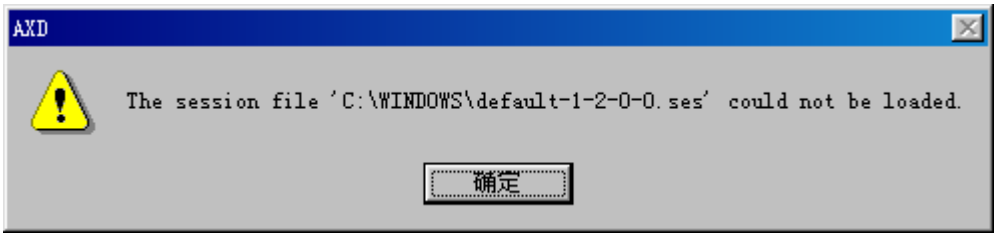

图 24 session 文件错误提示

在进入 AXD 调试环境后，有时会弹出 Fatal AXD Error 窗口，如图 25 所示，此时可以点击“ Connect mode... ”，然后选择“ ATTACH ... ”项确定，再点击“ Restart ”。接下来就可以使用【File】->【Load Image...】加载调试文件，进行 JTAG 调试。

注意：对于有的 PC 机，EasyJTAG 不能正确连接目标板，总是弹出错误对话框，这时可以检查并口连接是否可靠，检查并口上是否接有软件狗，或者重新给目标板上下电。另外，在 PC 机的 CMOS 设置里将并口模式设为 SPP 模式，设置并口的资源为 378H～37FH。

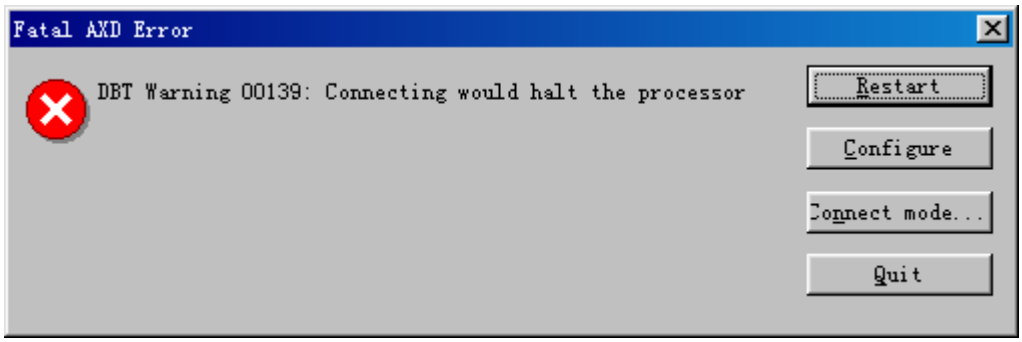

图 25 Fatal AXD 错误提示

片内外设的寄存器观察。在【System Views】->【Debugger Internals】即可打开 LPC2000

系列 ARM7 微控制器的片内外设寄存器窗口。有些寄存器是不能读出显示或读操作会影响其它寄存器的值，所以在片内外设寄存器窗口中不能找到，如果需要观察这些寄存器，可以使用存储器观察窗口(Memory)来实现。

使用 JTAG 下载程序到 FLASH。进入 AXD 调试环境，打开【Options】->【Configure Target...】，弹出 Choose Target 窗口，点击“ Configure ”按钮，进入“ EasyJTAG Setup ”设置窗口，在“ FLASH ”项中选择"Erase Flash when need"，然后确定退出。这样，每次装载 FLASH 地址的调试文件时，将会擦除 FLASH 并下载代码到 FLASH 中。

六、固化程序

对于 LPC2100 系列 ARM7 微控制器芯片来说，固化程序到片内 FLASH 可通过两种方式实现：JTAG 接口下载和使用 ISP 功能下载。 不管使用哪一种方式，用户均要先设置编译链接的地址，即代码地址从 0x00000000 地址开始，比如使用 LPC2100 专用工程模板时，在生成目标选用 RelInChip，其分散加载描述文件 mem_c.scf 如程序清单 1 所示。

其中，ROM_LOAD 为加载区的名称，其后面的 0x00000000 表示加载区的起始地址（存放程序代码的起始地址），也可以在后面添加其空间大小，如“ ROM_LOAD 0x00000000 0x20000 ”表示加载区起始地址为 0x00000000，大小为 128K 字节；ROM_EXEC 描述了执行区的地址，放在第一块位置定义，其起始地址、空间大小与加载区起始地址、空间大小要一致。从起始地址开始放置向量表(即 Startup.o(vectors, +First)，其中 Startup.o 为 Startup.s 的目标文件，vectors 代表 vectors 段定义的代码)，接着放置其它代码(即* (+RO)) ；变量区 IRAM 的起始地址为 0x40000000，首先放置 Startup.o(MyStacks)，接着放置其它文件的变量(即* (+RW,+ZI))；紧靠变量区之后的是系统堆空间(HEAP)，放置描述为 Startup.o(Heap)；由于 ARM 的堆栈一般采用满递减堆栈，所以堆栈区(STACKS)起始地址设置为 0x40004000，放置描述为 Startup.o(Stacks)。

程序清单 1 用于固化程序的分散加载描述文件 mem_c.scf

```
ROM_LOAD    0x00000000
{
    ROM_EXEC    0x00000000
    {
        Startup.o (vectors, +First)
        * (+RO)
    }

    IRAM    0x40000000
    {
        Startup.o (MyStacks)
        * (+RW,+ZI)
    }

    HEAP    +0    UNINIT
    {
        Startup.o (Heap)
    }

    STACKS    0x40004000    UNINIT  ; LPC2106 改为 0x40010000，LPC2105 改为 0x40008000
```

```
    {
        Startup.o (Stacks)
    }
}
```

1. 使用 JTAG 接口下载

使用 JTAG 接口下载程序到 FLASH 是需要 JTAG 仿真器的支持。EasyJTAG 仿真器可支持 LPC2000 系列 ARM7 微控制器的片内 FLASH 下载，这样就可以使用这一功能将程序下载到 FLASH 中，以便脱机运行。

首先设置 EasyJTAG 仿真器，参见图 26，注意 ARMcore 项一定要选择正确的 CPU 型号，否则可能会导致编程出错。

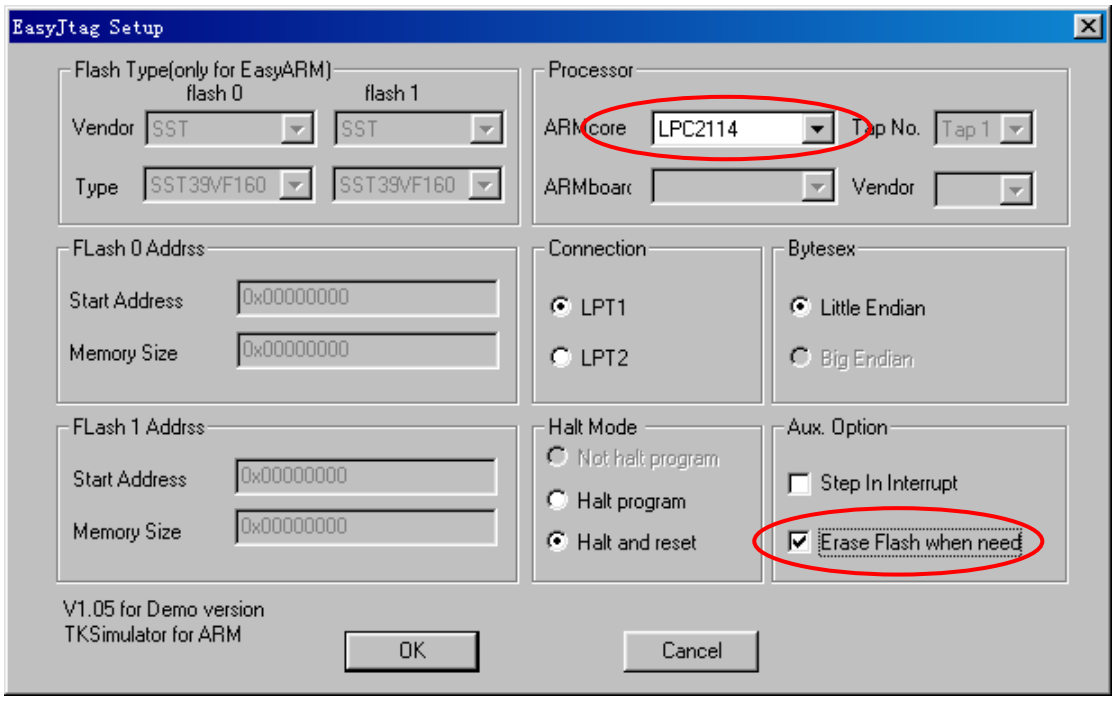

图 26 下载片内 FLASH 的 EasyJTAG 设置

然后将工程的生成目标选用 DebugInFLASH 或 RelInChip，编译链接，再按 F5 键进入 AXD 调试环境，在加载调试映像文件时即会下载程序到 FLASH 中。

实际上，只要你加载调试映像文件，且代码的地址设置为 FLASH 的地址，EasyJTAG 仿真器即把程序下载到指定的 FLASH 空间。

2. 使用 ISP 下载

LPC2100 系列 ARM7 微控制器芯片具有 ISP 功能，可以通过串口进行程序下载。

首先，选用工程的生成目标为 DebugInFLASH 或 RelInFLASH，这样工程连接时将会使用 mem_c.scf 或 mem_a.scf 分散加载描述文件，生成可固化到 FLASH 中的二进制代码。

然后，打开工程的 DebugInFLASH Settings 窗口，在 Target Settings 项中设置 Post-linker 选取 ARM fromELF（如图 27 所示）。

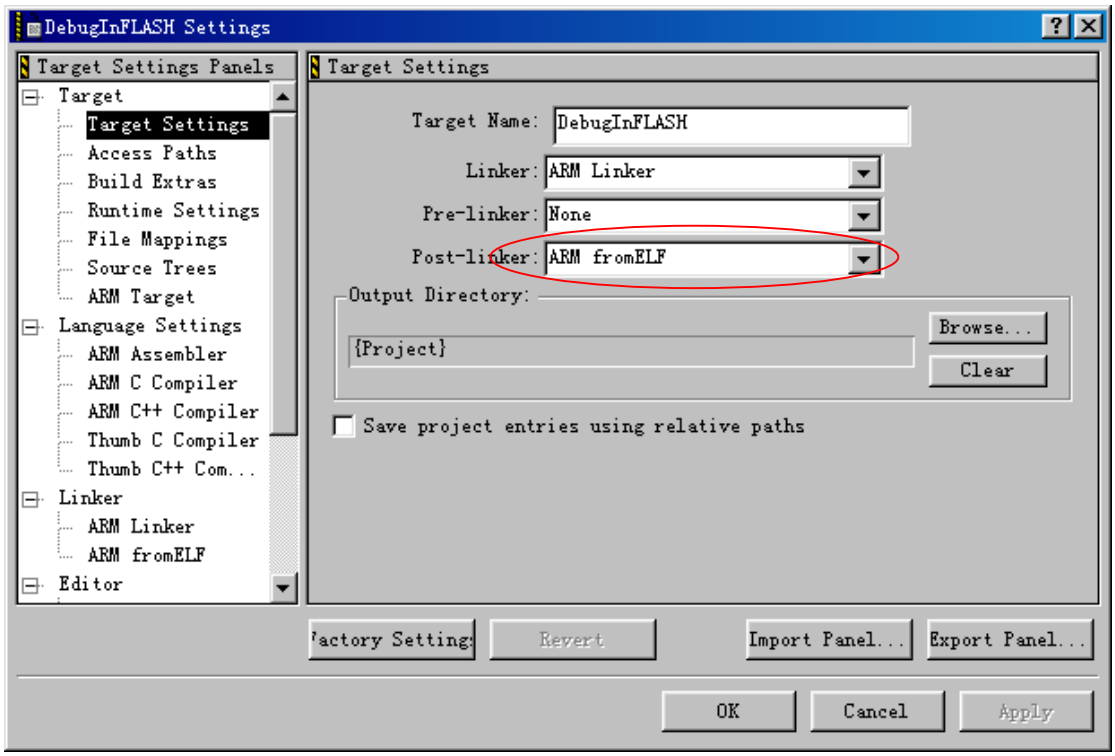

图 27　设置 Post-linker

接着，在 ARM formELF 项中设置输出文件类型，如设置为 Intel 32 bit Hex，然后设置输出文件名，也可指定目录，若不指定目录，则生成文件存放在当前工程的目录中（如图 28 所示）。重新编译连接，编译通过即会生成指定的输出文件（比如：leddisp.hex）。

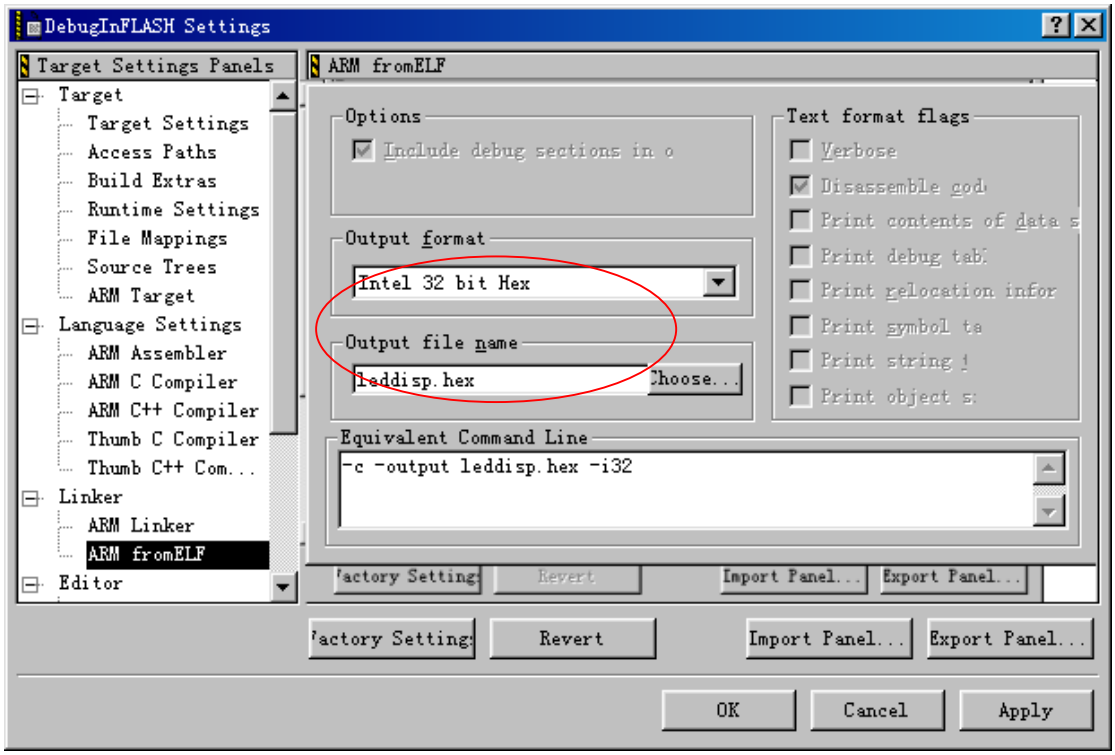

图 28　生成文件设置

生成 HEX 文件后，接下来使用串口延长线连接 PC 串口(如 COM1)和目标板 UART0(需要转换为 RS232 电平，如图 29 所示)，并将目标板上的 ISP（JP1）跳线短接。打开 LPC2000 Flash Utility 软件，并设置串口、波特率、系统晶振(注意，晶振频率项单位为 kHz)等，如图 30 所示。

设置好参数后，点击 Read Device ID 按钮，读取芯片 ID 号，若读取成功(状态栏显示“ Read Part ID Successfully! ”)，则表明 ISP 连接成功。否则，当出错提示为复位 LPC2000 信息时，如图 31 所示，首先目标板上的复位键，然后再单击图 31 的“ 确定 ”按钮。

连接成功后，先使用“ Erase ”按钮擦除选定扇区的 FLASH，然后在 Filename 项中输入要下载的 HEX 文件全名，点击“ Upload to Flash ”按钮即开始下载程序。程序固化后，将 ISP（JP1）跳线断开，重新复位系统即可运行程序。

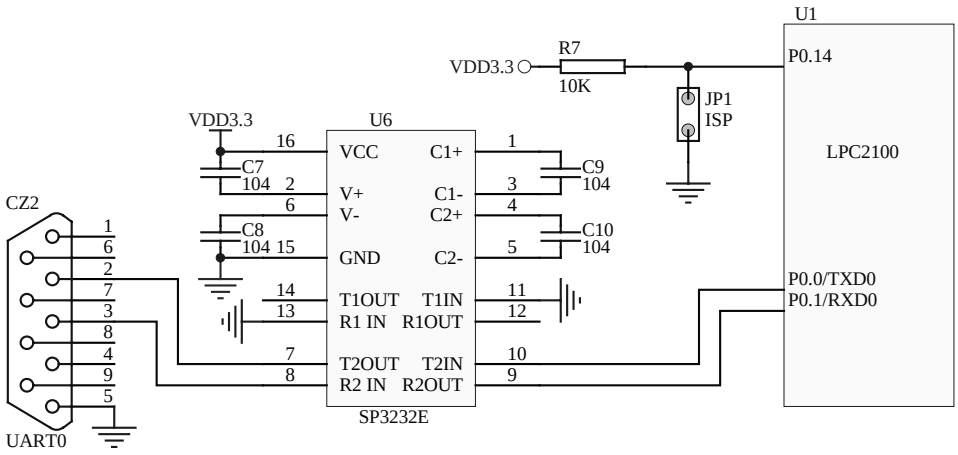

图 29 LPC2100 的 ISP 电路原理

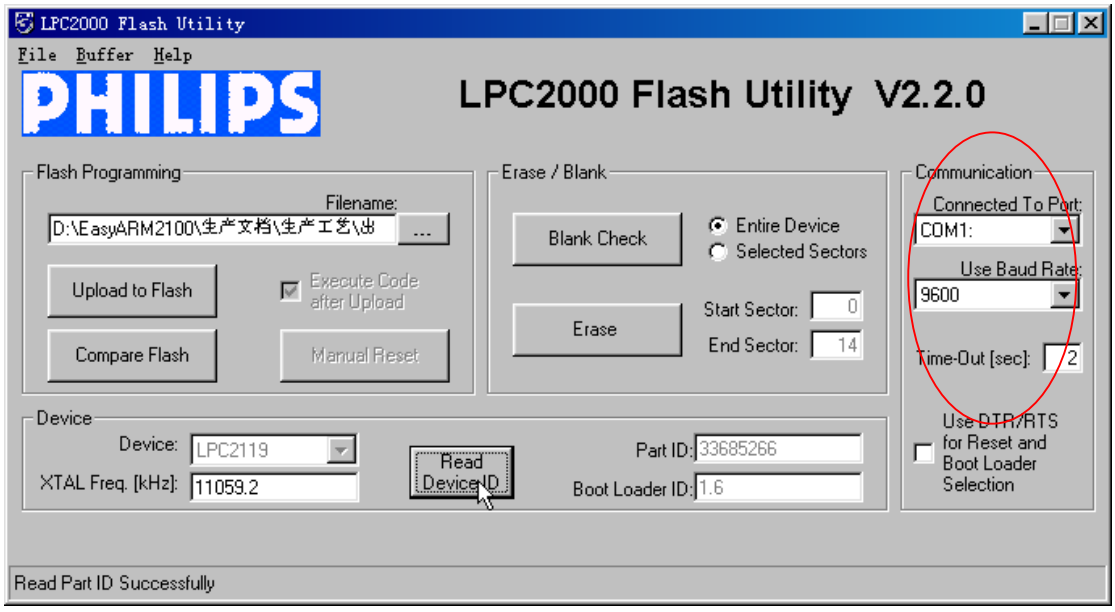

图 30 LPC2000 Flash Utility 软件设置

图 31　复位 LPC2000 提示