English-Chinese Series of Science & Technique of Electronic Information for University Labs

大学电子信息科学与技术英汉实验丛书

嵌入式系统设计实验

何菁　詹洪陈　戴瑾　编著

南京大学出版社

大学电子信息科学与技术英汉实验丛书

嵌入式系统设计实验

何 菁 詹洪陈 戴 瑾 编著

南京大学出版社

图书在版编目(CIP)数据

嵌入式系统设计实验 / 何菁，詹洪陈，戴瑾编著.
—南京：南京大学出版社，2014.9
（大学电子信息科学与技术英汉实验丛书）
ISBN 978-7-305-13699-3

Ⅰ.①嵌… Ⅱ.①何… ②詹… ③戴… Ⅲ.①微型计算机—系统设计—实验—高等学校—教材—英、汉 Ⅳ.①TP360.21-33

中国版本图书馆 CIP 数据核字（2014）第 178861 号

出版发行 南京大学出版社
社　　址 南京市汉口路 22 号　　　　邮编 210093
出 版 人 金鑫荣

丛 书 名 大学电子信息科学与技术英汉实验丛书
书　　名 嵌入式系统设计实验
编　　著 何　菁　詹洪陈　戴　瑾
责任编辑 惠　雪　沈　洁　　　　编辑热线 025-83686531

照　　排 江苏南大印刷厂
印　　刷 宜兴市盛世文化印刷有限公司
开　　本 787×1092 1/16 印张 12 字数 277 千
版　　次 2014 年 9 月第 1 版　　2014 年 9 月第 1 次印刷
ISBN 978-7-305-13699-3
定　　价 30.00 元

网　　址：http://www.njupco.com
官方微博：http://weibo.com/njupco
官方微信号：njupress
销售咨询热线：(025)83594756

大学电子信息科学与技术英汉实验丛书

编 委 会

序

为了培养高质量的人才，在高等学校的教学计划中，应该充分重视实验课的设置，并把课程的各个环节抓紧、抓好。尤其在本科生阶段，更要让学生接受全面的训练，从常见仪器的使用、基本量的测量，到按照一定的要求搭建系统以满足特定的性能指标，到有意地探寻和隔离非主要因素、探寻和突出主要因素，并观察其对于最终结果的影响。这样，由易而难、由简单而复杂，环环相扣、步步升高，教学工作才能扎实有效。

南京大学电子科学与工程学院李元教授长期从事电子技术教学工作，对于实验教学一向十分重视。他转往南京大学金陵学院后，一度执掌全院教务处和信息科学与工程学院的领导岗位。在他的带领下，陈孝桢教授、吴宗森教授、王均义教授、沈一骑高工、何菁博士等一批在大学电子信息科学与技术领域内有重要影响的专家学者齐心协力，先后开设了模拟电路实验、数字电路实验、电工学综合实验、通信原理实验、嵌入式系统实验、大学物理实验等课程，并编写了相应的教材，集结为丛书。经过多年的使用和实践，证明这些课程和教材在基础的层面上恰如其分地反映了学科发展的趋势，符合当前学生的实际，对于培养学生的独立工作能力发挥了很好的作用，也得到了国内同行的高度认同。

这些课程和教材也吸引了国外同行的注意，有英国 ESSEX 大学、法国勒芒大学、加拿大罗里尔大学、日本北海道情报大学等学校主动要求交流。可惜，由于教材原来都用中文写成，语言的障碍使有关部门无法把这些教材送往国外，或者送出后没有起到其应有的作用。与此同时，随着学校国际化程度的日益提高，在学生这一层面上的涉外活动也不断增多，要求学校在关心学生能否通过四级或六级英语考试的同时，注意加强对他们专业英语能力的培养。

这样，李元等教授就萌发了一个想法，将原有的实验教材进行重新整理、出新，并进行初译，然后交由吴宗森教授、加拿大罗里尔大学 Sturtevant 教授进行英语文字上的加工与译审，使之成为英汉实验丛书，正式出版以飨国内外广大读者。

我很高兴地得知，经过有关教授一年多的努力，这套英汉实验丛书即将完成，并陆续付印。希望这套丛书的出版，对于培养我国电子信息科学与技术领域的专门人才、对于促进国际交流都能有所裨益。

吴培亨

南京大学教授

中国科学院院士

2013. 12

前　言

现代电子技术的发展日新月异，各类新型、智能的电子设备已经渗透进我们生活的各个角落。其中的技术主角就是嵌入式系统设计。嵌入式系统技术已成为智能型产品设计的基础，在现代电子设计中扮演着无法替代的角色。

作为理工科大学电子通信类专业的基础课程之一，“嵌入式系统设计”是建立在电子学基础上，综合应用模拟和数字电路、微处理器原理和程序设计语言等一系列课程知识的一门技术型课程，具有涉及知识面广、实践性强的特点。

学习和掌握嵌入式系统设计的基本原理和方法，熟悉各种常用的电子器件和功能模块、熟练地将编程语言应用到电子产品设计环境中，这将对学生日后走向社会，从事各类电子设计、测试、生产等工作奠定良好的基础。

在学习嵌入式系统设计的过程中，实验环节显得尤为重要。本教材就是针对嵌入式系统设计课程配套的实验所编写的。

本教材的结构

课程包括 12 个基础型的实验。其中包括：

1. 开发环境和开发工具
2. GPIO 与 LED 控制
3. 键盘与中断
4. 串行通信
5. 利用串行接口实现调度台
6. 定时器与计数器
7. 使用有限状态机实现键盘控制
8. 实时时钟
9. I2C 接口与设备
10. LCD 的控制与显示
11. 模数转换实验
12. 数模转换实验

针对每一个单元实验，本教材对实验涉及的基础知识进行适当地梳理，以此为基础，对实验原理进行深入地讲解，对实验的关键电路进行分析，并对重点代码做了详细地解释。

所有实验都是基于 NXP LPC1768 芯片以及实验开发板设计的，因此教材在涉及 LPC1768 内部寄存器的细节描述时，只对与代码最直接关联的部分进行解释和说明。更

详细的内容则需要读者直接阅读 NXP 公司提供的相关数据手册。

本教材的特点

本教材的突出特点是强调基础性和实用性。规划的实验都是嵌入式系统设计中最基础最核心的知识。程序实现的方法都是在日后的工程实际中可以直接加以利用的。在具体的教学过程中，各个实验由浅入深，逐步推进。讲述在力求简洁易懂的同时，对关键点进行仔细分析，使读者可以全面理解并灵活运用。读者完成全部实验后，将对嵌入式系统所涉及的各种基础电路模块和软件编程方法有一个全面的掌握，并可以开始进行初步的设计工作。

教材主体采用英文讲述，每个实验都配有中文注解，对单元实验中的关键术语和段落进行说明，以利读者的理解。

本教材的实验使用的是市面上非常流行的以 NXP LPC1768 为核心的开发板。读者很容易按照自己的需要购置实验器材开展实验。

本教材配有完整的工程代码提供给教师，如果需要可以向作者索取。在教学过程中，我们鼓励学生在理解教材中关键代码的基础上，自己将工程补充完整。这样更加有利于学生对相关知识的理解。

本书可作为普通大学电子、通信、自动控制、智能仪器和物联网等相关专业本科学生的嵌入式课程配套的实验教材，也可作为嵌入式系统爱好者的学习实验参考用书。

本教材由何菁负责实验的总体设计，实验 5～12 的文字工作和全书的统调，詹洪陈负责实验 1～4 的设计、代码编写和文字工作以及全书插图的制作，戴瑾负责本书其他所有实验代码的编写与调试。

本教材在编写过程中，参考了 NXP 公司为 LPC1768 提供的设计参考和有关代码，以及维基百科的部分内容，文中没有逐一标注，在此表示感谢。

何菁

2014 年 5 月

于南京大学金陵学院

CONTENTS
目 录

Lab 1 Development and Experiment Tools

实验1 开发环境和开发工具

Purpose

❶ Learning how to use Keil MDK-ARM IDE.

❷ Understanding about the process of developing an embedded software project.

❸ Build the first project.

Setup the lab environment

Our development environment includes a Personal Computer and a Target Development Board, or simply a target board. The codes editing, compiling are done in PC. The generated executable image will be downloaded to the target board to test and execute.

Target board is connected to PC as shown in Figure 1-1. An USB cable connects the Emulator on the target board to PC and, in most cases, a RS-232 cable is needed to connects PC and target as the communication channel.

The integrated development environment (IDE), Keil MDK-ARM, is installed on PC as the basic tools chain for compiling, linking, debugging your software project.

Source codes of a project consist of one or more C language files and/or assembly code files, which you can edit and modify them with editor. Then the source codes are compiled by the compilers (C or assembly) to object files. Final, all the object files are linked together by Linker to generate the executable codes for specific CPU/MCU.

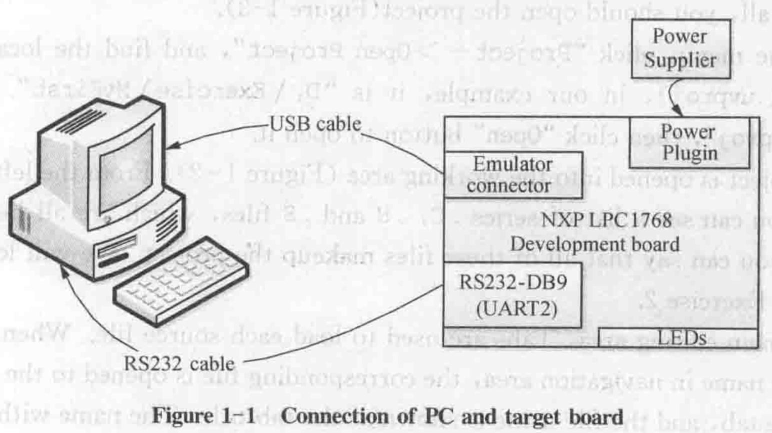


Figure 1-1 Connection of PC and target board

Exercise 1. Compile an exist project

The IDE we installed is Keil uVision version 4. 5. Double click the icon of uVision4 will open it and you will see a blank working area. (If it is not the first time you open uVision4, the previous project you worked with is loaded. You may select from the menu, "Project—>Close Project", to close it.)

The whole working area could be classified into 4 parts (Figure 1-2). Part 1 on the top is the menu and shortcut keys area. Part 2 on the left is the project files and the help files navigation area. Part 3 on the bottom is an output information area in which the building information will be displayed when IDE build the project. The maximum area, Part 4, in the middle is for editing your codes.

Following description will show you how to build an existing project step by step.

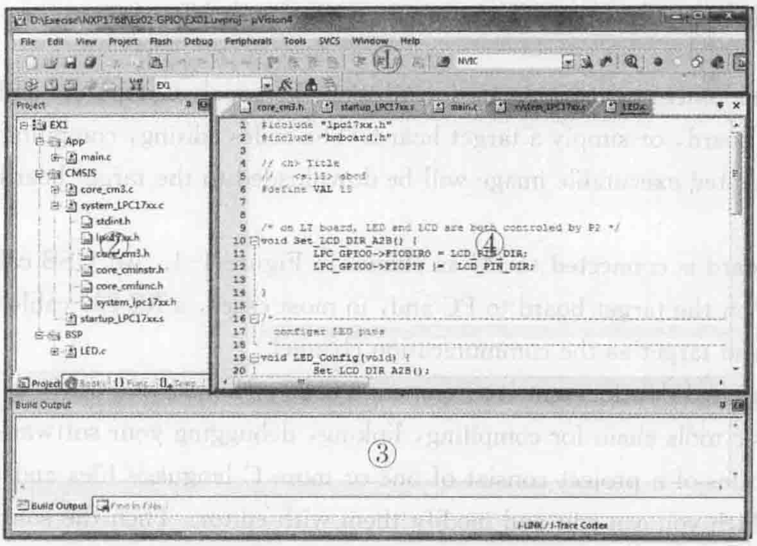


Figure 1-2 Working area of Keil uVision

First of all, you should open the project(Figure 1-3).

From the menu, click "Project—>Open Project", and find the location of the project file (.uvproj), in our example, it is "D: \Exercise\MyFirst". Select file "MyFirst.uvproj", then click "Open" button to open it.

The project is opened into the working area (Figure 1-2). From the left navigation windows, you can see a list of series .C, .H and .S files, which are all belong to the project, or you can say that all of these files makeup the project. We will look into the files later in Exercise 2.

In the main editing area, Tabs are used to load each source file. When you double click the file name in navigation area, the corresponding file is opened to the editing area as a separate tab, and the file name is shown as the tab title. The name with underscore means this tab is where you are editing.

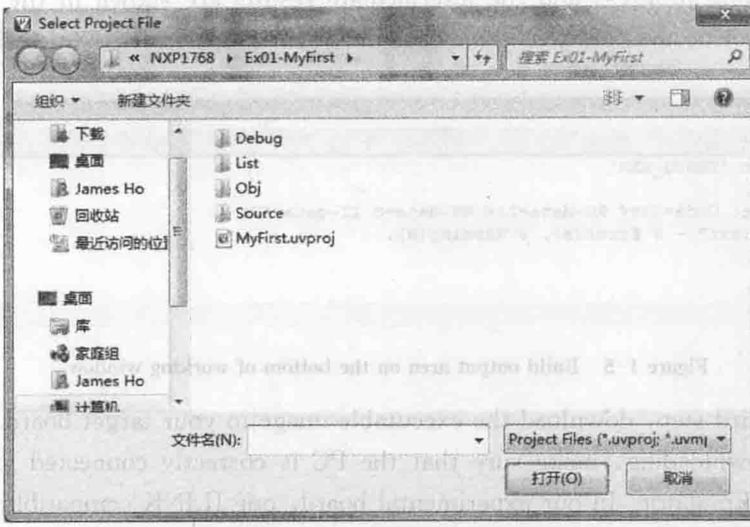


Figure 1-3 Window of select project file

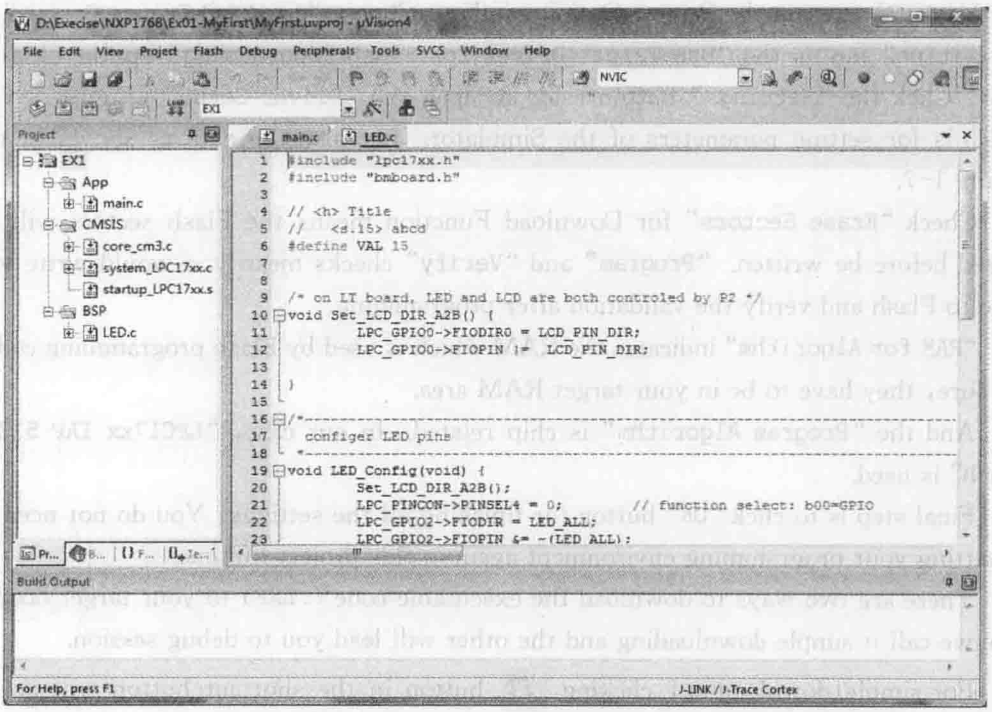


Figure 1-4 Working Window after open a project

Secondly, to build the project.

From the menu, select "Project—>Build Target", or press F7 directly, Keil will build the project. That means IDE will compile every source file in the project and link them together as a whole executable image file in an appropriate format.

The working progress and the intermediate results are shown in the information area. If no error occurs, an executable code image file is generated, EX01.AXF, as shown in Figure 1-5.

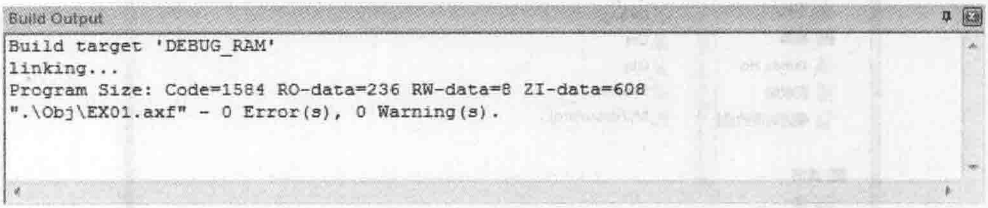


Figure 1-5 Build output area on the bottom of working window

As the third step, download the executable image to your target board.

Before downloading, make sure that the PC is correctly connected to the target board via the Emulator. In our experimental board, one JLINK compatible emulator is embedded which can be used to connect to PC via a USB cable.

Select "Project - >Options for Target 'Ex1'" from Keil menu (or press combined key Alt + F7) to open the Project Option window. Choose "J-LINK/J-Trance Cortex" in "Utilities" tab as the "Use Target Driver for Flash Programming" option (Figure 1-6). Click the "Settings" button aside to open the "Driver Setup" (Figure 1-7), which is for setting parameters of the Simulator. Just use the data as shown in the Figure 1-7.

Check "Erase Sectors" for Download Function means the Flash sectors will be erased before be written. "Program" and "Verify" checks mean you would write you code to Flash and verify the validation after programming.

"RAM for Algorithm" indicates the RAM which is used by Flash programming code, for sure, they have to be in your target RAM area.

And the "Program Algorithm" is chip related. In our case, "LPC17xx IAP 512KB Flash" is used.

Final step is to click "OK" button for finishing all the settings. You do not need to re-setting your programming environment again in this project lifetime.

There are two ways to download the executable code (.AXF) to your target board. One we call it simple downloading and the other will lead you to debug session.

For simple downloading, clicking LOAD button in the shortcut button will do all things for you. You will see a series of information scrolling in the output area, which show the progress of downloading. In the end, "Erase Done", "Programming Done" and "Verify OK" indicates all went well and your codes have been programmed to the target board (i. e. Flash memory inside LPC1768) successfully. It's ready to run!

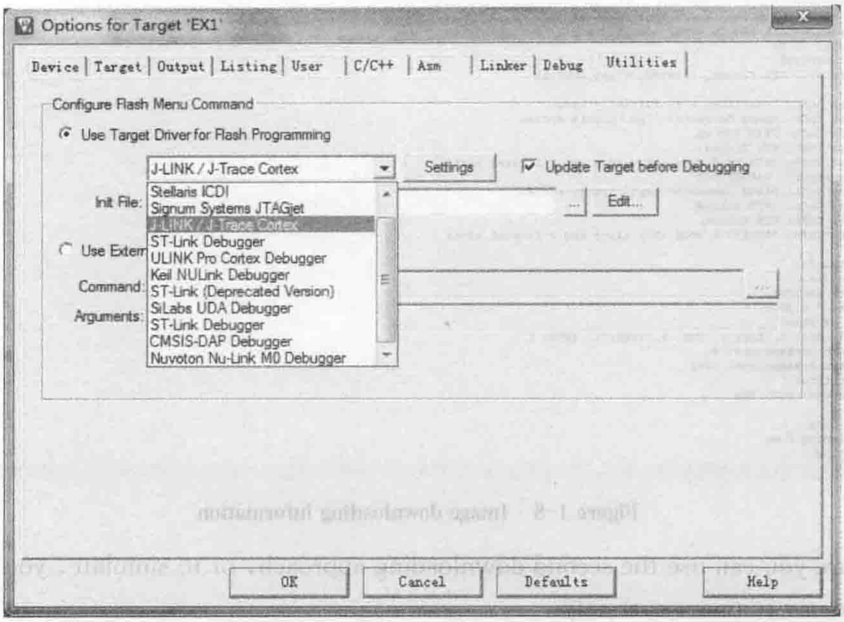


Figure 1-6 Target Flash driver selection on 'Option for Target'

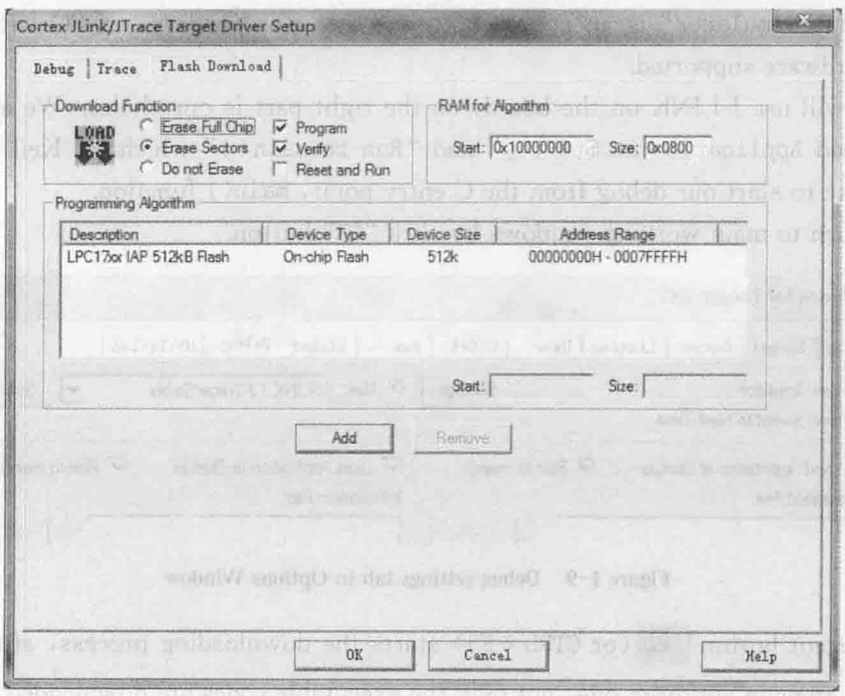
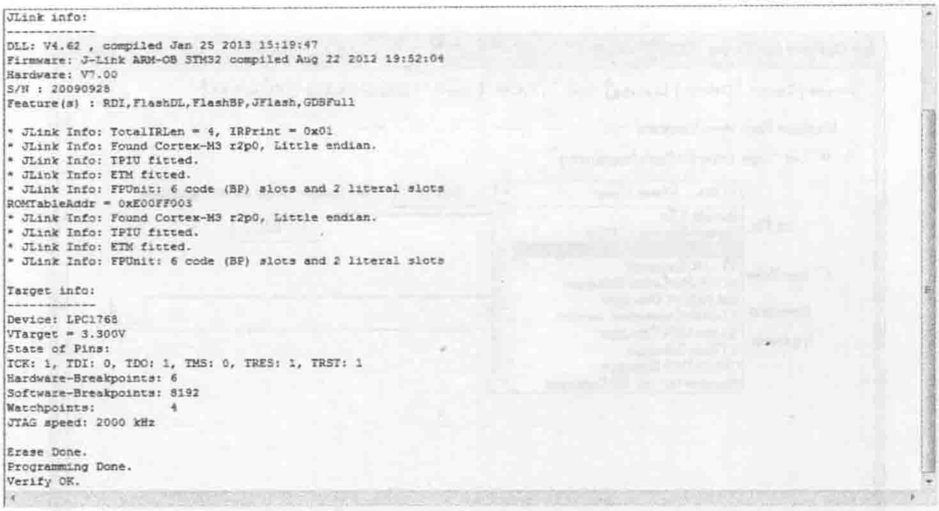


Figure 1-7 Simulator settings



```
JLink info:
------------
DLL: V4.62 , compiled Jan 25 2013 15:19:47
Firmware: J-Link ARM-OB STM32 compiled Aug 22 2012 19:52:04
Hardware: V7.00
S/N : 20090928
Feature(s) : RDI,FlashDL,FlashBP,JFlash,GDBFull

* JLink Info: TotalIRLen = 4, IRPrint = 0x01
* JLink Info: Found Cortex-M3 r2p0, Little endian.
* JLink Info: TPIU fitted.
* JLink Info: ETM fitted.
* JLink Info: FPUnit: 6 code (BP) slots and 2 literal slots
ROMTableAddr = 0xE00FF003
* JLink Info: Found Cortex-M3 r2p0, Little endian.
* JLink Info: TPIU fitted.
* JLink Info: ETM fitted.
* JLink Info: FPUnit: 6 code (BP) slots and 2 literal slots

Target info:
------------
Device: LPC1768
VTarget = 3.300V
State of Pins:
TCK: 1, TDI: 0, TDO: 1, TMS: 0, TRES: 1, TRST: 1
Hardware-Breakpoints: 6
Software-Breakpoints: 8192
Watchpoints:          4
JTAG speed: 2000 kHz

Erase Done.
Programming Done.
Verify OK.
```

Figure 1-8　Image downloading information

Before you can use the second downloading approach, or to simulate, you also need to set the environment correctly.

Back to Options window, in "Debug" tab, you could see two Debug options stand parallel, "Use Simulator" on the left and Use one of JTAG hardware simulator on the right. "Use Simulator" is an approach to execute your program on PC without real ARM hardware supported.

We will use J-LINK on the board, so the right part is our choice. We also check both "Load Application at Startup" and "Run to main()" which let Keil know we would like to start our debug from the C entry point, main() function.

Return to main working windows by click "OK" button.

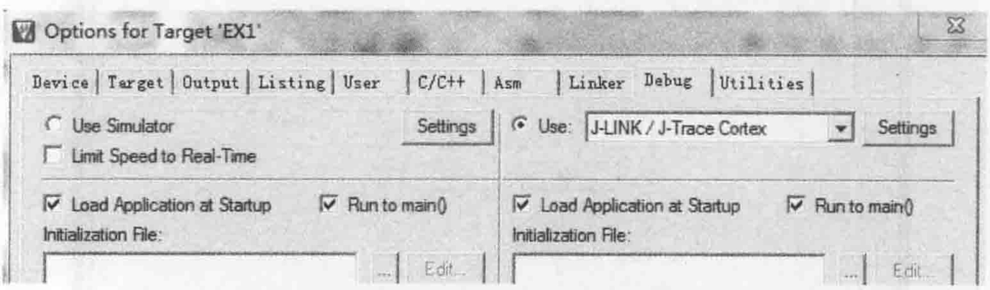


Figure 1-9　Debug settings tab in Options Window

Shortcut button (or CTRL + F5) starts the downloading process, after similar progress steps in approach one, not only the executable codes are downloaded to Flash, but the whole working window changes. Now what you see is a "Debugging" interface (Figure 1-10).

Navigation area in the left switches to new tab, "Registers", which shows all the