低引脚数 **USB** 开发工具包用户指南

请注意以下有关 Microchip 器件代码保护功能的要点：

- Microchip 的产品均达到 Microchip 数据手册中所述的技术指标。
- Microchip 确信：在正常使用的情况下， Microchip 系列产品是当今市场上同类产品中最安全的产品之一。
- 目前，仍存在着恶意、甚至是非法破坏代码保护功能的行为。就我们所知，所有这些行为都不是以 Microchip 数据手册中规定的操作规范来使用 Microchip 产品的。这样做的人极可能侵犯了知识产权。
- Microchip 愿与那些注重代码完整性的客户合作。
- Microchip 或任何其他半导体厂商均无法保证其代码的安全性。代码保护并不意味着我们保证产品是 “牢不可破”的。

代码保护功能处于持续发展中。 Microchip 承诺将不断改进产品的代码保护功能。任何试图破坏 Microchip 代码保护功能的行为均可视为违反了《数字器件千年版权法案 （Digital Millennium Copyright Act）》。如果这种行为导致他人在未经授权的情况下，能访问您的软件或其他受版权保护的成果，您有权依据该法案提起诉讼，从而制止这种行为。

QUALITY MANAGEMENT SYSTEM
CERTIFIED BY DNV
ISO/TS 16949:2002

Microchip 位于美国亚利桑那州 Chandler 和 Tempe 与位于俄勒冈州 Gresham 的全球总部、设计和晶圆生产厂及位于美国加利福尼亚州和印度的设计中心均通过了 ISO/TS-16949:2002 认证。公司在 PIC® MCU 与 dsPIC® DSC、KEELOQ® 跳码器件、串行 EEPROM、单片机外设、非易失性存储器和模拟产品方面的质量体系流程均符合 ISO/TS-16949:2002。此外，Microchip 在开发系统的设计和生产方面的质量体系也已通过了 ISO 9001:2000 认证。

目录

前言

客户须知

所有文档均会过时，本文档也不例外。Microchip 的工具和文档将不断演变以满足客户的需求，因此实际使用中有些对话框和 / 或工具说明可能与本文档所述之内容有所不同。请访问我们的网站（www.microchip.com）获取最新文档。

文档均标记有 “DS”编号。此编号位于每页底部的页码之前。 DS 编号的命名约定为 “DSXXXXXA”，其中 “XXXXX”为文档编号，“A”为文档版本。

欲了解开发工具的最新信息，请参阅 MPLAB® IDE 在线帮助。在 Help （帮助）菜单中选择 Topics （主题），打开现有在线帮助文件列表。

简介

本章包含使用低引脚数 USB 开发工具包前需要了解的一般信息。本章讨论的内容包括：

- 文档编排
- 本指南中使用的约定
- 推荐读物
- Microchip 网站
- 客户支持
- 文档版本历史

文档编排

本文档描述了如何使用低引脚数 USB 开发工具包作为开发工具在目标板上仿真和调试固件。本手册的内容编排如下：

- **第 1 章 “概述”**
- **第 2 章 “项目实验入门”**
- **附录 A “原理图”**

本指南中使用的约定

本手册采用下列文档约定：

文档约定

说明	涵义	示例
Arial 字体：		
斜体字	参考书目	*MPLAB® IDE User's Guide*
	需强调的文字	*... 仅有*的编译器 ...
首字母大写	窗口	Output 窗口
	对话框	Settings 对话框
	菜单选项	选择 Enable Programmer
引用	窗口或对话框中的字段名	“Save project before build”
带右尖括号前有下划线的斜体文字	菜单路径	*File>Save*
粗体字	对话框按钮	点击 **OK**
	选项卡	点击 **Power** 选项卡
N‘Rnnnn	verilog 格式的数字，其中 N 是总位数， R 是基数， n 是其中一位。	4‘b0010， 2‘hF1
尖括号 < > 括起的文字	键盘上的按键	按 <Enter> 和 <F1>
Courier New 字体：		
常规 Courier New	源代码示例	`#define START`
	文件名	`autoexec.bat`
	文件路径	`c:\mcc18\h`
	关键字	`_asm, _endasm, static`
	命令行选项	`-Opa+, -Opa-`
	位值	`0, 1`
	常数	`0xFF, ‘A’`
斜体 Courier New	可变参数	*`file`*`.o`，其中 *`file`* 可以是任一有效文件名
方括号 []	可选参数	`mcc18 [options]` *`file`* `[options]`
花括号和竖线：{|}	选择互斥参数； “或”选择	`errorlevel {0|1}`
省略号 ...	代替重复文字	`var_name [, var_name...]`
	表示由用户提供的代码	`void main (void) { ... }`

推荐读物

本用户指南描述了如何使用低引脚数 USB 开发工具包。下面列出了其他有用的文档。以下 Microchip 文档均已提供，并建议读者作为补充参考材料。

自述文件

如需关于使用其他工具的最新信息，请阅读 MPLAB® IDE 安装目录的 Readmes 子目录下的相应工具的自述文件。这些自述文件包含了本用户指南中可能未包括的更新信息和已知问题。

设计中心

Microchip USB 设计中心的网址为：**www.microchip.com/usb**。

以下 Microchip 应用笔记均已提供，并建议读者作为补充参考材料。

MICROCHIP 网站

Microchip 网站 （www.microchip.com）为客户提供在线支持。客户可通过该网站方便地获取文件和信息。只要使用常用的因特网浏览器即可访问，网站提供以下信息：

- **产品支持**——数据手册和勘误表、应用笔记和示例程序、设计资源、用户指南以及硬件支持文档、最新的软件版本和存档软件
- **一般技术支持**——常见问题 （FAQ）解答、技术支持请求、在线讨论组以及 Microchip 顾问计划成员列表
- **Microchip 业务**——产品选型和定购指南、最新 Microchip 新闻稿、研讨会与活动安排表、 Microchip 销售办事处、代理商以及工厂代表列表

客户支持

Microchip 产品的用户可以通过以下渠道获得帮助：

- 代理商或代表
- 当地销售办事处
- 应用工程师 （FAE）
- 技术支持

客户应联系其代理商、代表或应用工程师 （FAE）寻求支持。当地销售办事处也可为客户提供帮助。本文档后附有销售办事处的联系方式。

也可通过 http://support.microchip.com 网站获取技术支持。

文档版本历史

版本 A （2008 年 9 月）

- 本文档的初始版本。

版本 B （2009 年 2 月）

- 纠正了文档错误。

第 1 章 概述

1.1 简介

低引脚数 USB 开发工具包为评估 Microchip 的 PIC18F14K50 和 PIC18F13K50 20 引脚 USB 单片机的功能提供了一种简易的低成本方法。这款全功能的工具包包含了将新一代 USB 设计从概念变为第一个样机所需的硬件、软件和代码示例。创建时考虑到 USB 初学者，它还包含了一本自学指导课程和实验材料 《**Microchip 低引脚数 USB 解决方案入门**》，以帮助用户尽快掌握与将 USB 连接添加到嵌入式系统相关的知识。

1.2 重点

本章讨论了：

- 低引脚数 USB 开发工具包内容
- 低引脚数 USB 开发板的结构和布局

1.3 低引脚数 USB 开发工具包内容

低引脚数 USB 开发工具包包含以下各项：

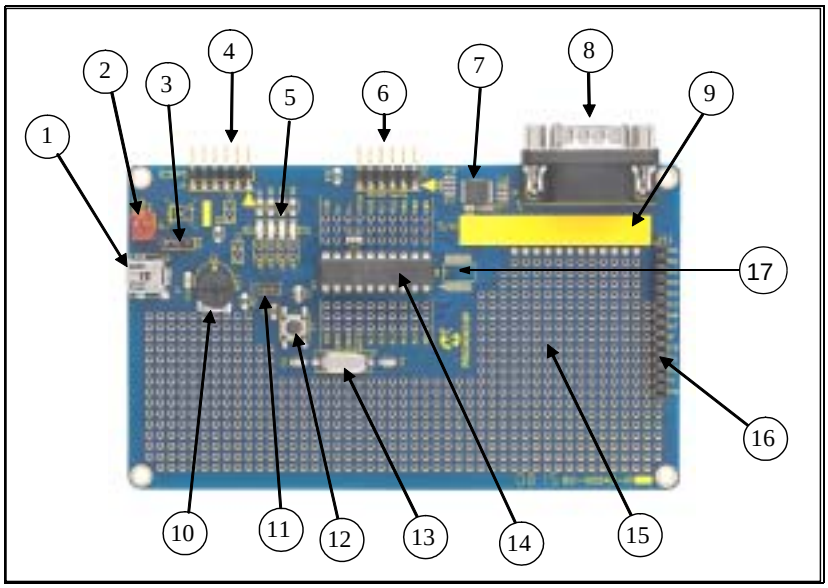
- (1) 完全组装的低引脚数 USB 开发板
- (1) 未组装的备用开发板
- (1) PIC18F14K50 ICD 已组装扩展连接头
- (1) 一张包含用户指南、课程材料和产品文档的光盘
- (1) PICkit™ 2 调试器 / 编程器及电缆

图 1-1： 低引脚数 USB 开发工具包

1.4 低引脚数 USB 开发板的结构和布局

图 1-2 给出了低引脚数 USB 开发板及已组装元件。

图 1-2： 低引脚数 USB 开发板



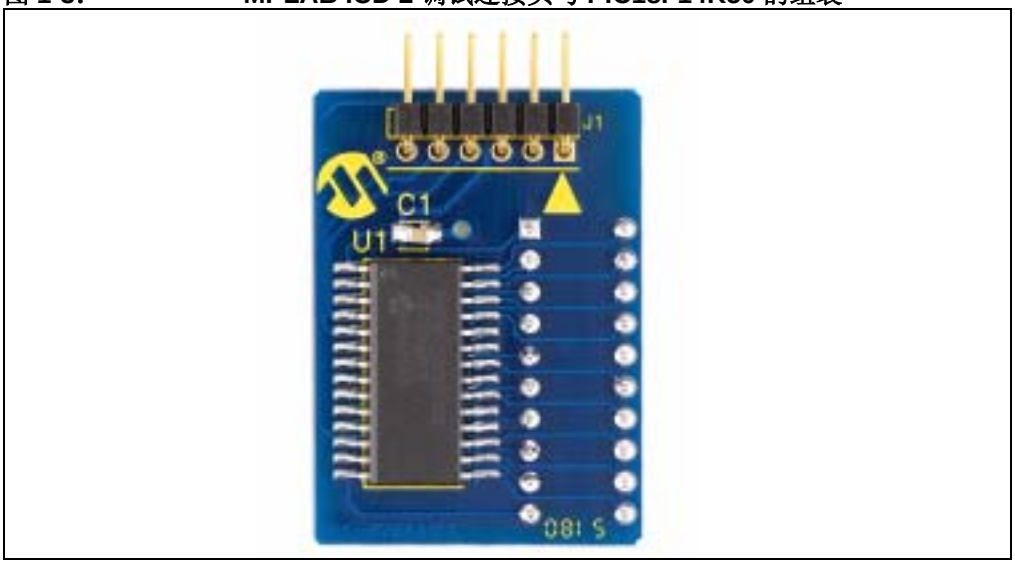
1. USB 微型 B 连接器
2. J9 稳压 5V 连接头
3. J14，用于连接 V_{BUS} （USB 提供）或 J9 稳压 5V 到 PIC18F14K50 V_{DD} 电源
4. PICkit™ 2 调试器 / 编程器连接头
5. 与 PORTC （RC0、 RC1、 RC2 和 RC3）相连的 LED
6. PICkit™ 串行分析器连接头
7. **MAX3232** RS-232 线路驱动器 / 接收器
8. RS-232 连接器
9. 用户 PID/VID 信息存储区
10. 电位器
11. J12，用于与 PIC18F14K50 上的 V_{USB} 建立 / 断开连接
12. 按钮
13. 12 MHz 晶振
14. PIC18F14K50 MCU
15. 实验布线区
16. PICtail™ 子板扩展连接头
17. SSOP 扩展

注： J2-J5、 J7 和 J8 在开发板底侧并联在一起，因此即使未安装跳线器也还是默认这些功能连接在一起。切除跳线是为了禁止每个引脚所连接的电路。

1.5 PIC18F14K50 ICD 调试连接头

低引脚数 USB 开发工具包包含一个调试连接头，它与 PIC18F14K50 ICD MCU 组装在一起，以使用 PICkit™ 2 调试器 / 编程器。

图 1-3： MPLAB ICD 2 调试连接头与 PIC18F14K50 的组装



要使用调试连接头，只需除去低引脚数 USB 开发板上 MCU 插座 （U2）中贴装的 PIC18F14K50。然后使用引脚连接头 （未包含）将调试连接头连接到 MCU 插座并将 PICkit™ 2 编程器 / 调试器连接到所提供的连接头。

1.6 《MICROCHIP 低引脚数 USB 解决方案入门》自学指导课程

低引脚数 USB 开发工具包包含自学指导课程 **《Microchip 低引脚数 USB 解决方案入门》**。此课程提供了 USB 2.0 协议的简要介绍及其在 PIC18F14K50 MCU 中的实现。Microchip 的 USB 设备固件框架作为资源被引入，它提供了 USB 操作所需的固件代码库，用以处理低级别的任务和许多参考项目。用户通过遵循指示信息逐步完成 “动手” 实验来加深对所涉及概念的理解。

注：

第 2 章 项目实验入门

2.1 简介

用户指南的此部分将向用户演示一组项目实验，有助于简化原始 USB 设计应用程序的开发。它们具有固定的格式，用户通过遵循指示信息对某些代码段取消注释或进行复制粘贴来逐步完成每个项目的源代码。选择这种格式是为了强制开发人员探讨框架的重要代码段和熟悉所给源文件的整体结构。

实验文件位于文件夹 `C:\LPCUSBDK_Labs\Labx_files` 中。每个实验文件夹都包含实验的源文件以及一个包含解决方案及其工作代码的文件夹。

提供了 4 个实验：

1. 项目实验 1 （枚举）：此实验向用户介绍如何在其应用程序中开发主机 （PC） 用来枚举和最终配置 PIC18F14K50 的独特描述符。
2. 项目实验 2 （HID 鼠标）：此实验使用定义的描述符进一步阐述在项目实验 1 中学到的概念。用户将在 *mouse.c* 固件文件内逐步开发应用特定函数。最终应用通过在 PC 屏幕上移动指针，实现与人机接口设备类 （Human Interface Device， HID）鼠标类似的操作。
3. 项目实验 3 （HID 键盘）：在此实验中，要求用户通过修改描述符来实现基于 HID 键盘的应用。通过旋转低引脚数 USB 开发板上的电位器来更改通过 USB 传送到 PC 的符合 HID 规范的 Unicode 值，该值经过 ADC 转换后以字符形式输出。
4. 项目实验 4 （CDC 串行仿真器）：最后，用户将使用通信设备类 （Communication Device Class， CDC）驱动程序实现通信协议转换器。

2.2 前提条件

这些实验假定用户：

1. 完成了随附光盘上提供的自学指导课程 《**Microchip 低引脚数 USB 解决方案入门**》。
2. 熟悉 MPLAB IDE 和 C18 编译器。
3. 有使用 C 语言编程的经验。
4. 熟悉 Microchip 的 PIC18F 系列单片机。

2.3 完成项目实验所需的资源

为了完成这组项目实验，用户应该：

1. 已在 PC 上安装了最新版本的 MPLAB IDE 和 C18 编译器。MPLAB IDE 可至以下网站下载：
   http://www.microchip.com/stellent/idc-plg?IdcService=SS_GET_PAGE&nodeId=1406&dDoc-Name=en019469&part=SW007002

2. 可至以下网站下载 C18 编译器：
   http://www.microchip.com/stellent/idcplg?Idc-Service=SS_GET_PAGE&nodeId=1406&dDocName=en010014
3. 最新版本的 PICkit 2 编程器软件。
4. 下载并安装 Microchip 全速 USB 固件框架。该框架可从以下网站免费下载：
   http://www.microchip.com/stellent/idcplg?IdcSer-vice=SS_GET_PAGE&nodeId=2651&param=en534494
5. 具有“*Microchip USB Device Firmware Framework User's Guide*”（DS51679）的副本
6. PIC18F13K50/14K50 数据手册 （DS41350B_CN）的副本
7. USB R2.0 规范副本，可从以下网站下载：
   http://www.usb.org/developers/docs/
   整个实验过程中，它都作为参考使用。
8. 通用串行总线 （Universal Serial Bus， USB） HID 用法表副本，可从以下网站下载：
   http://www.usb.org/developers/devclass_docs/Hut1_12.pdf
9. USB HID 描述符工具，可从以下网站免费下载：
   http://www.usb.org/developers/hidpage/dt2_4.zip
10. 通信设备的通用串行总线类定义文档副本，可从以下网站下载：
    http://www.usb.org/developers/devclass_docs
11. 将 LPCUSBDK_Labs.zip 文件解压缩到 C: 目录下。

2.4 项目实验 1 （枚举）

注： 在本项目实验以及随后的所有项目实验中，使用 PICkit 2 编程器编程时，必须断开 USB 电缆与低引脚数 USB 开发板上 USB 微型 B 连接器的连接。

2.4.1 实验目的

此实验的目的旨在指导用户使用 Microchip 的全速 USB 固件框架在 MPLAB IDE 中创建项目。尽管框架中的许多应用程序示例都可以用来创建原始代码，但是从零开始构建框架仍然不失为熟悉此多任务工具整体功能的一个好方法。

用户将创建一个项目、确保 IDE 进行了相应的配置并更改 *usb_descriptor.c* 文件以使 PIC18F14K50 作为 HID 鼠标设备进行枚举。

2.4.2 实验步骤

2.4.2.1 构建框架

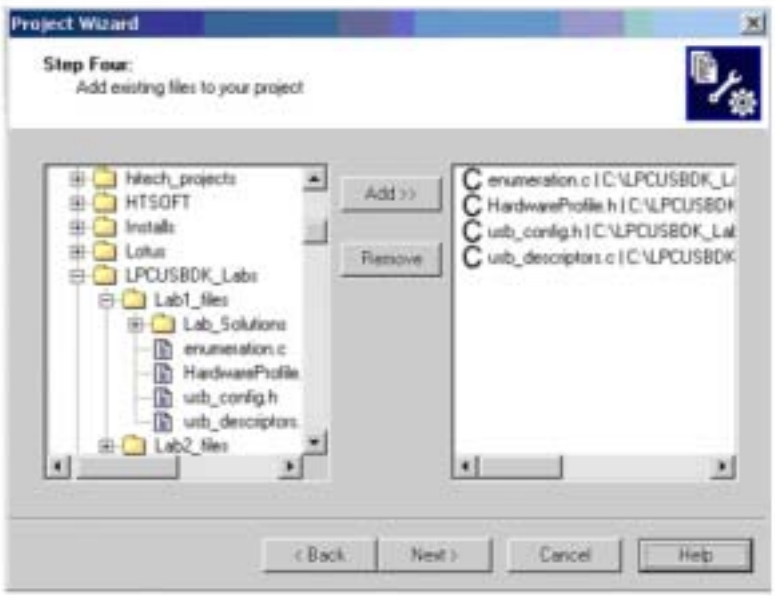
1. 通过选择 *Start>Programs>Microchip>MPLAB IDE vx.xx>MPLAB IDE* （开始 > 程序 >Microchip>MPLAB IDE vx.xx>MPLAB IDE）打开 MPLAB IDE。
2. 打开 MPLAB IDE 后，通过选择 *Project>Project Wizard* （项目 > 项目向导）启动 Project Wizard。
3. 选择 **PIC18F14K50** 作为器件，选择 **MPLAB C18 C Compiler** （MPLAB C18 C 编译器）作为语言工具套件，并在下列目录中创建一个新的项目文件夹：
   `C:\Microchip Solutions\Project Lab 1\Project Lab 1`

4. 在 “Add Existing Files to Your Project”（添加现有文件到项目）窗口中，导航到 C:\LPCUSBDK_Labs\Lab1_files 并复制以下文件：
   a) enumeration.c
   b) usb_descriptors.c
   c) HardwareProfile.h
   d) usb_config.h

注： 这些文件是用户文件，需要对它们进行更改以实现任何应用程序。

要复制某个文件，请单击窗口右窗格中文件路径前的大写字母 A，直到出现大写字母 C 为止。通过此方法可将此文件复制到新的项目目录下，而原来的文件保持不变 （见图 2-1）。

图 2-1：



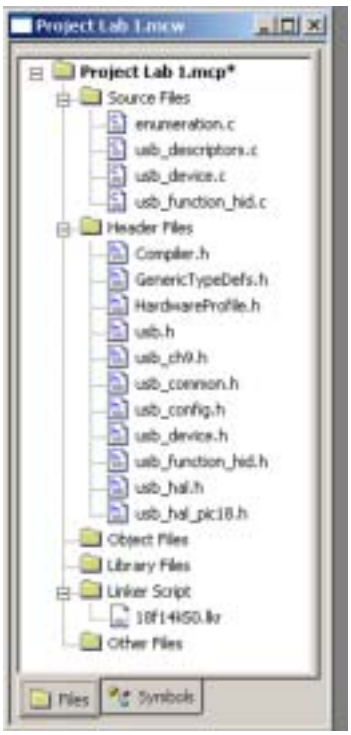
下一部分将添加构建框架的文件，该框架实现低级别的 USB 功能。不是每个应用程序都需要此步骤。包含的示例项目很容易进行转换以满足定制应用程序的需要。但是，为了提供框架的直观介绍，此实验从零开始构建应用程序。

注： 这些文件由框架中所有应用程序使用。因此，更改这些文件将影响所有应用程序。如果不小心更改了这些文件中的任意一个，建议重新安装框架。

5. 然后，导航到 C:\Microchip Solutions\Microchip\Include 并复制以下文件：
   a) Compiler.h
   b) GenericTypeDefs.h

6. 导航到 C:\Microchip Solutions\Microchip\Include\Usb 并复制以下文件：
   a) usb.h
   b) usb_ch9.h
   c) usb_common.h
   d) usb_device.h
   e) usb_function_hid.h （文件定义了特定于 HID 类的组件）
   f) usb_hal.h
   g) usb_hal_pic18.h （文件定义了特定于 PIC18 构架的组件）
7. 然后，导航到 C:\Microchip Solutions\Microchip\Usb 并复制以下文件：
   a) usb_device.c
8. 然后，导航到 C:\Microchip Solutions\Microchip\Usb\HID Device Driver 以复制特定于 HID 的源文件：
   a) usb_function_hid.c
9. 最后，导航到 C:\MCC18\lkr 并复制 18f14k50.lkr 链接描述文件。
10. 单击 *Next>Finish*（下一步 > 完成）退出 Project Wizard。项目窗口现在应如图 2-2 所示。

图 2-2： 实验 1 的项目窗口



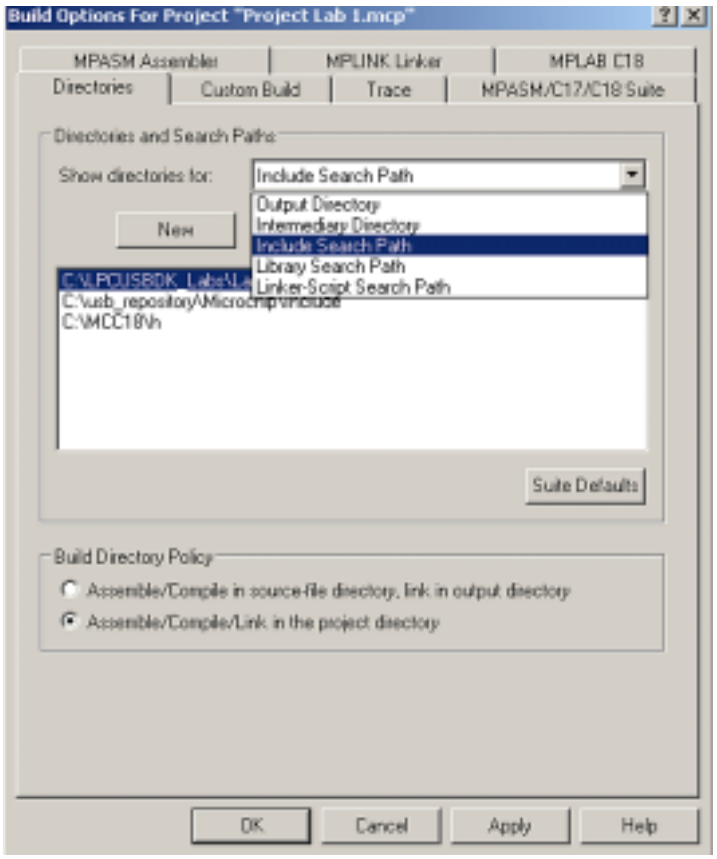
如果未在 MPLAB IDE 工作空间内打开项目窗口，请选择 *View> Project* （视图 > 项目）。

然后需要针对框架配置 MPLAB IDE，方法是将 C18 编译器定向到相关的文件位置。

11. 在 MPLAB IDE 中选择 *Project>Build Options...>Project*（项目 > 编译选项 ...> 项目）。将出现 Build Options （编译选项）对话框 （图 2-3）。

12. 单击 **Directories** （目录）选项卡，选择 **Output Directory** （输出目录）并单击 **New** （新建）以添加新路径。单击 ... 按钮，导航到 `C:\LPCUSBDK_Labs\Lab1_files` 并创建一个名为 **output** 的新文件夹。突出显示该文件夹并单击 **OK** （确定）。这就是保存输出文件的文件夹。
13. 在 “Show directories for”（显示目录）下拉菜单内选择 “Include Search Path”（包含搜索路径）目录。确保列出了 `C:\MCC18\h` 目录路径。选择 **New** 并导航到 `C:\Microchip Solutions\Microchip\Include`，单击 **OK** 以添加此目录。重复这些步骤以添加应用程序文件夹 `C:\LPCUSBDK_Labs\Lab1_files`。
14. 在 “show directories for:”下拉菜单中选择 “Library Search Path”（库搜索路径）。确保列出了 `C:\MCC18\lib`。然后，选择 “Linker-Script Path”（链接描述路径）并确保路径指向 `C:\MCC18\lkr` 目录。

图 2-3： 针对 MICROCHIP USB 固件框架的配置



15. 单击 **Apply** （应用），然后单击 **OK** 以应用这些设置并关闭 Build Options 窗口。

现在已构建了框架。

定义项目描述符

双击项目窗口中的 `enumeration.c` 源文件，将其打开。向下滚动到 `ProcessIO()`。注意它是空的。因此，此应用程序将不执行任何操作。此实验的目的是指导用户正确配置固件以使 PIC18F14K50 在连接到主机 PC 时作为 HID 鼠标进行枚举。因此，需要对 `usb_descriptors.c` 文件进行相应的修改。应将作为参考的

USB R2.0 规范打开至**第 9-5 节 “描述符”**。此部分详细描述了每种描述符类型（设备、配置和接口等）所需的各种组件。此 usb_descriptor.c 文件是以下框架文件夹中找到的同名源文件的副本：

C:\Microchip Solutions\USB Device - HID - Mouse\HID - Mouse – Firmware\usb_descriptor.c

调试期间此文件用作参考。

16. 在 MPLAB IDE 项目窗口中，选择 usb_descriptors.c 源文件并将其打开。

    请注意文件开头的 #include "./USB/usb_function_hid.h"。如果为给定应用定义了其他的设备类，需要在这里将相应的类头文件包括进来。

17. 向下滚动到设备描述符部分，将例 2-1 中的代码复制粘贴到标有以下内容的段内（用花括号括起来）：

    //ADD DEVICE DESCRIPTOR CODE HERE

注：	用户可能希望代码整洁，可以将代码部分与注释段部分适当分隔。为了确保此文档的完整性，忽略了制表符。

例 2-1：　　实验 1 的设备描述符

```
0x12,                   // Size of this descriptor in bytes
USB_DESCRIPTOR_DEVICE,  // DEVICE descriptor type
0x0110,                 // USB Spec Release Number in BCD format
0x00,                   // Class Code
0x00,                   // Subclass code
0x00,                   // Protocol code
USB_EP0_BUFF_SIZE,      // Max packet size for EP0, see
                        // usbcfg.h
MY_VID,                 // Vendor ID
MY_PID,                 // Product ID
0x0003,                 // Device release number in BCD format
0x01,                   // Manufacturer string index
0x02,                   // Product string index
0x00,                   // Device serial number string index
0x01                    // Number of possible configurations
```

18. 向下滚动到配置、特定类和接口描述符段。将例 2-2 中的代码复制粘贴到标有以下内容的段内 （用花括号括起来）：

    //ADD CONFIGURATION, CLASS SPECIFIC AND
    //INTERFACE DESCRIPTOR CODE HERE

例 2-2：　　配置、特定类和接口

```
0x09,//sizeof(USB_CFG_DSC),    // Size of this descriptor in bytes
USB_DESCRIPTOR_CONFIGURATION,  // CONFIGURATION descriptor type
DESC_CONFIG_WORD(0x0022),      // Total length of data for this cfg
1,                             // Number of interfaces in this cfg
1,                             // Index value of this configuration
0,                             // Configuration string index
_DEFAULT|_SELF,                // Attributes, see usbd.h
50,                            // Max power consumption (2X mA)

/* Interface Descriptor */
0x09,//sizeof(USB_INTF_DSC),   // Size of this descriptor
                               // in bytes
USB_DESCRIPTOR_INTERFACE,      // INTERFACE descriptor type
0,                             // Interface Number
0,                             // Alternate Setting Number
1,                             // Number of endpoints in this intf
HID_INTF,                      // Class code
BOOT_INTF_SUBCLASS,            // Subclass code
HID_PROTOCOL_MOUSE,            // Protocol code
0,                             // Interface string index

/* HID Class-Specific Descriptor */
0x09,//sizeof(USB_HID_DSC)+3,  // Size of this descriptor in bytes
DSC_HID,                       // HID descriptor type
DESC_CONFIG_WORD(0x0111),      // HID Spec Release Number in BCD
                               // format(1.11)
0x00,                          // Country Code (0x00 for Not
                               // supported)
HID_NUM_OF_DSC,                // Number of class descriptors, see
                               // usbcfg.h
DSC_RPT,                       // Report descriptor type
DESC_CONFIG_WORD(50),          // sizeof(hid_rpt01),
                               // Size of the report descriptor
/* Endpoint Descriptor */
0x07,/*sizeof(USB_EP_DSC)*/
USB_DESCRIPTOR_ENDPOINT,       // Endpoint Descriptor
HID_EP | _EP_IN,               // EndpointAddress
_INT,                          // Attributes
DESC_CONFIG_WORD(3),           // size
0x01                           // Interval
```

鼓励用户花些时间将上述代码示例中的代码与 USB R2.0 规范**第 9-5 节 “描述符”**中的描述符定义进行比较。为了使代码直观且更容易参考 USB 规范，所以包含了注释。可通过在 MPLAB IDE 中的文件中突出显示定义名并选择 ***Edit>Find*** （编辑 > 查找）来查找定义源。这会找到项目内所有定义的实例并在 Output （输出）窗口中将它们列出来。

19. 向下滚动到字符串描述符段。字符串描述符用于向用户描述设备。枚举时，字符串描述符将出现在屏幕的右下角，告知用户设备的预期用途。将例 2-3 和例 2-4 中表示制造商字符串描述符和产品字符串描述符的代码分别复制粘贴到标有以下内容的段内 （用花括号括起来）：

    //ADD MANUFACTURER STRING DESCRIPTOR CODE HERE

    和

    //ADD PRODUCT STRING DESCRIPTOR CODE HERE

例 2-3： 实验 1 的制造商字符串描述符

```
'M','i','c','r','o','c','h','i','p',
' ','T','e','c','h','n','o','l','o','g','y',
' ','I','n','c','.'
```

例 2-4： 实验 1 的产品字符串描述符

```
'M','o','u','s','e',
' ','E','n','u','m','e','r','a','t','i','o','n ',
' ','D','e','m','o'
```

最后，添加最后一个描述符即报告描述符。

HID 类规范开发的早期使用子类标识其他 HID 设备类别。但是，由于能用此类实现的设备繁多，很明显我们需要一个更强大的定义方法。因此，此类并不使用子类来定义大部分协议。而是使用一个称为报告描述符的机制来标识数据协议和为设备提供的数据类型。在这里定义了诸如鼠标上的按键数、键盘的 Unicode 键值范围以及其他一些特性。

USB 组织提供了许多专为实现报告描述符而设计的文档和工具，可通过以下网站访问：

http://www.usb.org/developers/hidpage/

鼓励用户下载定义了某个给定设备的报告描述符的 HID 用法表。而且， USB 组织还进一步开发了一个 HID 描述符工具，用于协助开发人员设计他们自己的报告描述符。该工具还包含了常用 HID 设备 （例如鼠标和键盘）的预定义描述符。可在**第 2.3 节 “完成项目实验所需的资源”**中列出的链接处找到此文档和工具。

20. 向下滚动到报告描述符段，将例 2-5 中的代码复制粘贴到标有以下内容的段内 （用花括号括起来）：

    //ADD REPORT DESCRIPTOR CODE HERE

例 2-5：　　实验 1 的报告描述符

```
0x05, 0x01, /*  Usage Page (Generic Desktop)*/
0x09, 0x02, /*  Usage (Mouse)*/
0xA1, 0x01, /*  Collection (Application)*/
0x09, 0x01, /*  Usage (Pointer)*/
0xA1, 0x00, /*  Collection (Physical)*/
0x05, 0x09, /*  Usage Page (Buttons) */
0x19, 0x01, /*  Usage Minimum (01)*/
0x29, 0x03, /*  Usage Maximum (03)*/
0x15, 0x00, /*  Logical Minimum (0)*/
0x25, 0x01, /*  Logical Maximum (0)*/
0x95, 0x03, /*  Report Count (3)*/
0x75, 0x01, /*  Report Size (1)*/
0x81, 0x02, /*  Input (Data, Variable, Absolute)*/
0x95, 0x01, /*  Report Count (1)*/
0x75, 0x05, /*  Report Size (5)*/
0x81, 0x01, /*  Input (Constant)    ;5 bit padding  */
0x05, 0x01, /*  Usage Page (Generic Desktop)*/
0x09, 0x30, /*  Usage (X)*/
0x09, 0x31, /*  Usage (Y)*/
0x15, 0x81, /*  Logical Minimum (-127)*/
0x25, 0x7F, /*  Logical Maximum (127)*/
0x75, 0x08, /*  Report Size (8)*/
0x95, 0x02, /*  Report Count (2)*/
0x81, 0x06, /*  Input (Data, Variable, Relative)*/
0xC0, 0xC0
```

现在完成了描述符定义。

21. 编译项目。应准确无误地完成。

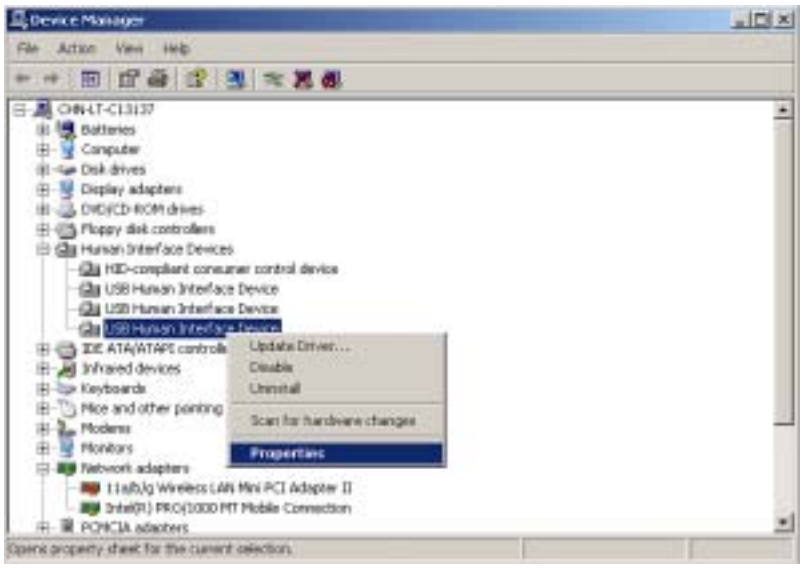
测试应用

22. 配置低引脚数 USB 开发板，使 J14 跳线位于最右边的两个引脚上。此应用将通过 USB 电缆由 V_{BUS} 供电。
23. 断开 J12 跳线的连接。
24. 将 PICkit 2 编程器连接到 PC USB 端口，然后连接到低引脚数 USB 开发板上的 J6 连接器。
25. 通过选择 *Start>Programs>Microchip PICkit 2 vx.xx* 打开 PICkit 2 编程器环境。
26. PICkit 2 编程器软件应识别 PICkit 2 已连接并指定 PIC18F14K50 器件。
27. 在 PICkit 2 编程器软件内，导航到 `C:\LPCUSBDK_Labs\Lab1_files\output` 文件夹并下载 `Project Lab 1.hex` 文件到 PIC18F14K50。
28. 断开 PICkit 2 编程器与 J6 的连接，然后将 USB 电缆插入微型 B 连接器 J1。

    连接后，将启动枚举过程。主机 PC 应识别连接的新设备并在屏幕的右下角显示一个通知，显示此实验早期在产品字符串中设置的 “Mouse Enumeration Demo”文本。

29. 接下来，将检查主机 PC 上的设备驱动程序。
    导航到主机 PC 上的设备管理器，方法是选择 Start>Settings>Control Panel>System（开始 > 设置 > 控制面板 > 系统）打开 System Properties（系统属性）窗口。选择 **Hardware** （硬件）选项卡并单击 **Device Manager** （设备管理器）按钮。
30. 在 Device Manager 窗口中展开 “Human Interface Devices”。右键单击每个驱动程序并选择 **Properties** （属性）直到找到 “Mouse Enumeration Demo”驱动程序。请参见图 2-4。

例 2-6： 设备管理器窗口和安装的 HID 驱动程序



鼓励用户熟悉 HID 用法表和 HID 描述符工具。更改描述符的各个方面然后重复此枚举实验步骤将有助于树立这些概念。

2.5 项目实验 2 （HID 鼠标）

2.5.1 实验目的

此实验在 HID 鼠标应用中实现低引脚数 USB 演示板功能，它在主机 PC 上移动鼠标指针做画圆动作。

注：	此处开发的源代码是在框架中找到的 USB 设备 （HID）鼠标应用程序的完整副本。

```
C:\Microchip Solutions\USB Device - HID - Mouse\HID - Mouse – Firmware
```

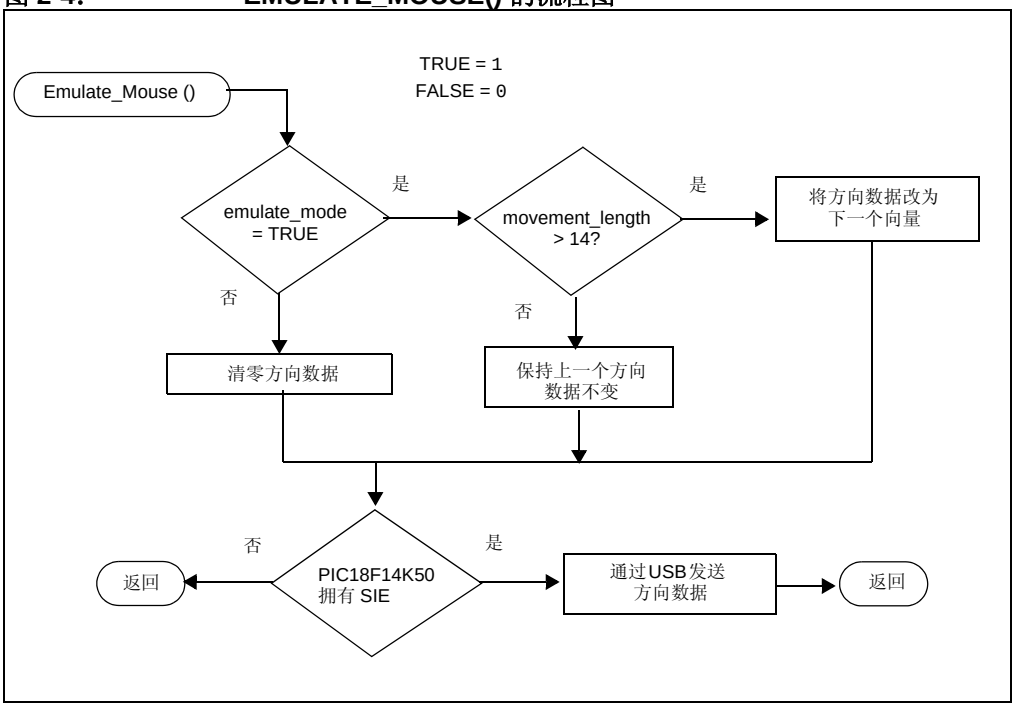
鼓励用户在需要时使用这些文件作为参考。

2.5.2 HID 鼠标固件概述

与大部分框架应用程序一样，用户定义的源代码是从 <application>.c 文件中的 ProcessIO() 函数调用的。用户定义的固件通过主循环指示主机 PC 鼠标指针单向移动 14 次。14 次之后，鼠标指针改变方向，最终移动完整的一圈。初始化名为 emulate_mode 的位标志，该标志会在低引脚数 USB 开发板上的按钮按下时，切换为置 1 或清零状态。此标志的状态会启动或停止屏幕上的指针移动，而不是通过调用用户定义的、用于处理鼠标移动例程的函数 emulate_mouse() 来实现。

图 2-4 给出了用户定义的函数的流程图。

图 2-4： EMULATE_MOUSE() 的流程图



在图 2-4 的流程图中，可以看出如果 emulate_mode 标志为 0，那么将清零通过 USB 发送的方向数据。请注意，不管该标志是否置 1，数据都将通过 PIC18F14K50 器件发送。只有在 SIE 能够发送数据的情况下才发送数据。通过在代码中使用条件语句 if(HIDTxHandleBusy(lastTransmission) == 0) 实现此检查。在发送时加载 lastTransmission 并由 “if” 条件语句中的 HIDTxHandleBusy 宏进行处理。

如果 emulate_mode 标志置 1，那么函数进入鼠标指针移动算法。这可以通过跟踪计数器变量 movement_length 实现。当该变量超过 14 时，缓冲数组将装入在 mouse.c 文件开头定义的 dir_table 数组提供的新方向数据。可通过递增向量变量计数器来访问数组的每个元素。然后将该缓冲数组装入 hid_report_in[] 缓冲数组，HIDTxPacket 宏使用 hid_report_in[] 缓冲数组通过 USB 发送数据到主机 PC。

2.5.3 实验步骤

此实验进一步阐述实验 1 中开发的描述符代码。

1. 创建一个新项目 “Project Lab 2”，并按照上一个实验中的第 1-15 步构建框架，不过这一次我们使用在 C:\LPCUSBDK_Labs\Lab2_files 下找到的源文件。
2. 确保 Project Build 选项按照实验 1 的第 11-15 步进行了配置。
3. 在项目窗口中打开 mouse.c 源文件。
4. 向下滚动找到变量定义，取消对变量和数组的注释，它们将要被源文件段中用户定义的函数所使用：

   ```
   /**VARIABLES*****************************/
   ```

   a) `BYTE old_sw2,old_sw3;`
   b) `BOOL emulate_mode;`
   c) `BYTE movement_length;`
   d) `BYTE vector = 0;`
   e) `char buffer[3];`
   f) `USB_HANDLE lastTransmission;`
   g) `ROM signed char dir_table[]={-4,-4,-4, 0, 4, 4, 4, 0};`

5. 向下滚动到私有函数原型段，取消对用户函数原型的注释
   `void Emulate_Mouse(void);`
6. 向下滚动到 UserInit()。在这里初始化与手头应用相关的所有组件。请注意已初始化的变量和函数调用。找到 `// emulate_mode = TRUE;`，取消对标志初始化代码的注释。
7. 向下滚动到 ProcessIO() 函数。请注意检查用于翻转 emulate_mode 标志的按钮。取消对以下代码的注释：
   `//emulate_mode = !emulate_mode;`
8. 最后，向下滚动到 Emulate_Mouse 函数，在标有以下内容的段中插入例 2-7 中的代码 （用花括号括起来）：

   //ADD EMULATE MOUSE CODE HERE

例 2-7： EMULATE_MOUSE() 的用户定义的函数代码

```
if (emulate_mode == TRUE)
    {
      //go 14 times in the same direction before changing
      if (movement_length > 14)
      {
        buffer[0] = 0;
        buffer[1] = dir_table[vector & 0x07];
        //X-Vector
        buffer[2] = dir_table [(vector+2) & 0x07];
        //Y-Vector
        // go to the next direction in the table
       vector++;
        //reset the counter for when to change again
        movement_length = 0;
      }//end if (movement_length > 14)
    }
else
    {
        //don't move the mouse
        buffer[0] = buffer[1] = buffer[2] = 0;
    }
    if(HIDTxHandleBusy(lastTransmission) == 0)
    {
        //copy over the data to the HID buffer
        hid_report_in[0] = buffer[0];
        hid_report_in[1] = buffer[1];
        hid_report_in[2] = buffer[2];
       //Send the 3 byte packet over USB to the host.
       lastTransmission = HIDTxPacket(HID_EP, (BYTE*)hid_report_in,
       0x03);

        //increment the counter to change the data sent
        movement_length++;
    }
```

请注意 HID 鼠标以 3 字节数据包的形式发送。此数据包的格式如下：

- 第一个字节通常用于标识鼠标按键。由于此应用无需任何鼠标点击操作，所以此字节始终为零。
- 第二个字节和第三个字节分别表示横轴 （X）和纵轴 （Y）坐标。

编译项目。应准确无误地完成。

测试应用

> **注：** 需从 Device Manager 中除去上一个实验中创建的设备，以使在本实验中创建的新设备能够正确枚举。在 Device Manager 窗口中，右键单击此设备然后选择 **Uninstall** （卸载）。

9. 确保低引脚数 USB 开发板如实验 1 进行配置。
10. 连接 PICkit 2 编程器并打开 PICkit 2 编程软件。
11. 在 `C:\LPCUSBDK_Labs\Lab2_files` 中找到此实验的 *.hex* 文件并下载到 PIC18F14K50。

12. 断开 PICkit 2 的连接，并将低引脚数 USB 演示板连接到主机 PC 端口。
13. 该设备应已枚举，且板上 LED 应按照自学指导课程中讨论的那样根据 `BlinkUSBStatus()` 闪烁。
14. 按下按钮应启动鼠标指针在主机 PC 屏幕上移动一圈，然后停止。

鼓励用户使用此应用程序进行进一步实验，方法是修改鼠标指针移动的时间长度或以下语句中的值来更改方向：

```
ROM signed char dir_table[]={-4,-4,-4, 0, 4, 4, 4, 0};
```

2.6 项目实验 3 （HID 键盘）

警告

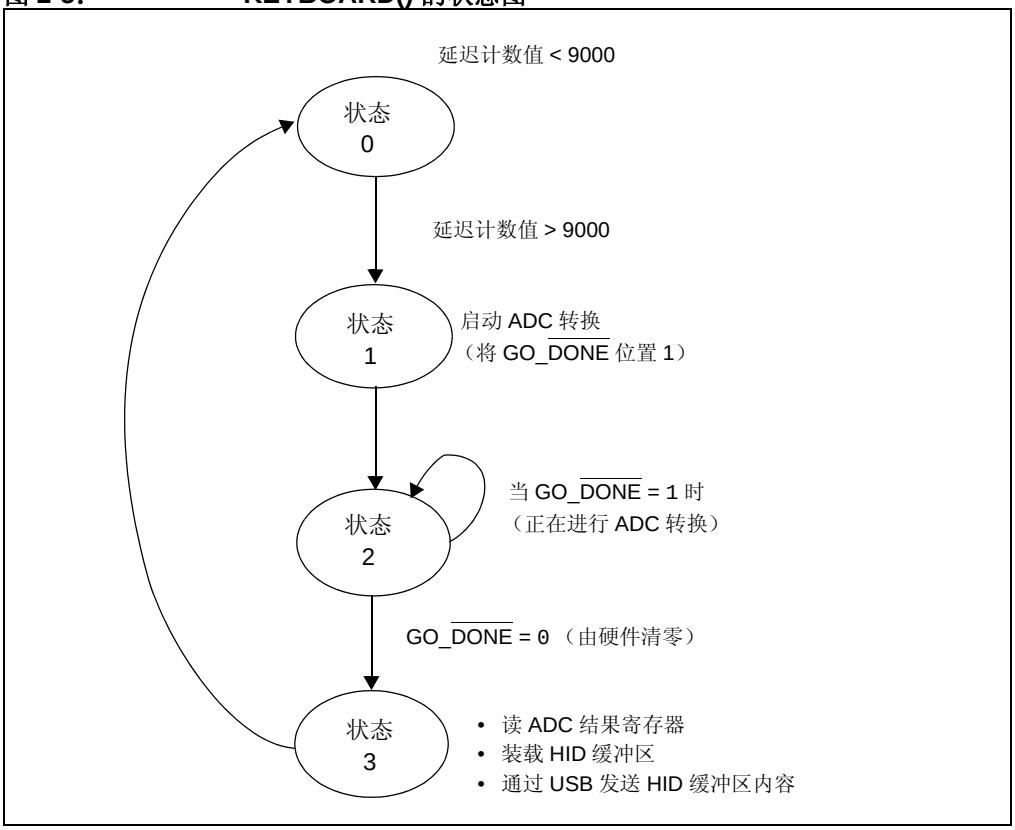
提醒一句，如果使用了可启动 Windows® 快捷方式的键组合，此 HID 设备可能会有破坏作用。请小心使用以确保发送缓冲区只包含用户确信不会产生任何破坏性键组合的键代码。

在本实验中， PIC18F14K50 被用作 HID 键盘设备。配置 ADC 外设使之根据与低引脚数 USB 开发板上的电位器相连的端口引脚上的电平进行转换。然后使用 ADC 结果寄存器中的值创建一个 4 到 29 之间的数字值，该值将使主机 PC 屏幕上显示一个 “a” 到 “z” 之间的字母字符 （见 HID 用法表文档**第 10 节 “键盘 / 键盘页”**（0x07））。随着电位器的旋转，输出到屏幕上的字符也将发生相应的变化。用户应注意此应用还适用于数据记录器应用，该应用使用低引脚数 USB 开发板上的电位器模拟混合信号接口以监视片外应用。通过 USB 产生和发送的数据可连接并通过转换呈现至主机 PC 上的图形用户界面，以监视实时应用操作，或用于保存重要数据留作以后分析。

2.6.1 HID 键盘固件概述

`keyboard()` 是用户定义的函数，可从 `ProcessIO()` 调用以解析从ADC模块接收到的数据、通过 USB 发送数字值并在屏幕上显示相应的字符。此函数作为状态机实现。此函数的状态图如图 2-5 所示。

图 2-5：　　KEYBOARD() 的状态图



从状态图可以看出，各个状态分别执行以下这些常规任务：

1. 状态 0：这是延时状态，用于确保 ADC 外设上的保持电容在启动转换之前有足够的时间充电。

2. 状态 1：此状态通过将 ADCON0 控制寄存器中的 GO_$\overline{\text{DONE}}$ 位置 1 启动转换过程。

3. 状态 2：检查转换是否完成，如果 GO_$\overline{\text{DONE}}$ = 0，则允许状态机进入下一个状态。

4. 状态 3：此最终状态读 ADC 结果寄存器、将结果转换为一个 4 （“a”）到 29 （“z”）之间的值、装载 HID 缓冲区并通过 USB 传送结果数据到主机 PC 上，以将其输出到打开的 *.txt* 文档中。

用户从此实验中得出的主要观点是：由于 USB 框架是多任务环境，所以应使用不阻塞的代码。因此，如本实验所示，状态机被普遍用于许多应用。每次通过主循环调用 `keyboard()` 状态机。如果未满足离开当前状态进入下一个状态的条件，状态机只会从函数返回而不会更改状态。下一次执行主循环时，将再次检查此条件。如果满足了状态条件，状态更改为随后的下一个状态。状态变量作为 `static` 类型声明，因为这会允许状态变量中的当前值在从 `keyboard()` 返回后保持不变。

2.6.2 实验步骤

此应用程序需要对 *usb_descriptor.c* 文件进行少量更改以将 PIC18F14K50 配置为 HID 键盘。注意：可使用从 http://www.usb.org/devel-opers/hidpage/ 下载的 HID 描述符工具对报告描述符进行更改。鼓励用户花些时间阅读此页的内容并了解开发 HID 应用所使用的资源。

1. 与之前实验一样创建一个名为“Project Lab 3”的新项目。实验 3 的源文件可在此路径下找到：
   `C:\LPCUSBDK_Labs\Lab3_files`
2. 确保 Project Build 选项按照实验 1 的第 11-15 步进行配置。
3. 打开 `usb_descriptor.c` 文件并向下滚动到接口描述符。取消对协议代码定义 `//HID_PROTOCOL_KEYBOARD,` 的注释
4. 向下滚动到 HID 类特定描述符，取消对 HID 报告宏的大小 `//DESC_CONFIG_WORD(63),` 的注释
   更改上一个实验中的报告描述符，使之包含 63 个组件，主机 PC 将需要这些组件来标识设备的键盘属性。
5. 向下滚动到报告描述符处并在标有以下内容的段中添加例 2-8 中的代码：

   //ADD REPORT DESCRIPTOR HERE

例 2-8： KEYBOARD() 的报告描述符

```
0x05, 0x01,   //   USAGE_PAGE (Generic Desktop)
0x09, 0x06,   //   USAGE (Keyboard)
0xa1, 0x01,   //   COLLECTION (Application)
0x05, 0x07,   //   USAGE_PAGE (Keyboard)
0x19, 0xe0,   //   USAGE_MINIMUM (Keyboard LeftControl)
0x29, 0xe7,   //   USAGE_MAXIMUM (Keyboard Right GUI)
0x15, 0x00,   //   LOGICAL_MINIMUM (0)
0x25, 0x01,   //   LOGICAL_MAXIMUM (1)
0x75, 0x01,   //   REPORT_SIZE (1)
0x95, 0x08,   //   REPORT_COUNT (8)
0x81, 0x02,   //   INPUT (Data,Var,Abs)
0x95, 0x01,   //   REPORT_COUNT (1)
0x75, 0x08,   //   REPORT_SIZE (8)
0x81, 0x03,   //   INPUT (Cnst,Var,Abs)
0x95, 0x05,   //   REPORT_COUNT (5)
0x75, 0x01,   //   REPORT_SIZE (1)
0x05, 0x08,   //   USAGE_PAGE (LEDs)
0x19, 0x01,   //   USAGE_MINIMUM (Num Lock)
0x29, 0x05,   //   USAGE_MAXIMUM (Kana)
0x91, 0x02,   //   OUTPUT (Data,Var,Abs)
0x95, 0x01,   //   REPORT_COUNT (1)
0x75, 0x03,   //   REPORT_SIZE (3)
0x91, 0x03,   //   OUTPUT (Cnst,Var,Abs)
0x95, 0x06,   //   REPORT_COUNT (6)
0x75, 0x08,   //   REPORT_SIZE (8)
0x15, 0x00,   //   LOGICAL_MINIMUM (0)
0x25, 0x65,   //   LOGICAL_MAXIMUM (101)
0x05, 0x07,   //   USAGE_PAGE (Keyboard)
0x19, 0x00,   //   USAGE_MINIMUM (Reserved (no event indicated))
0x29, 0x65,   //   USAGE_MAXIMUM (Keyboard Application)
0x81, 0x00,   //   INPUT (Data,Ary,Abs)
0xc0
```

然后，配置 *keyboard.c* 源文件。

6. 在项目窗口中，打开 *keyboard.c* 源文件并向下滚动到 `UserInit()`。取消对端口和 ADC 初始化代码的注释。鼓励用户查阅 PIC18F14K50 数据手册以找到这些相应外设的初始化含义。

   ```
   // ADCON0=0x29;
   // ADCON1 = 0X00;
   // ADCON2=0x3F;
   ```

   向下滚动到 `keyboard()` 并将例 2-9 中的代码复制粘贴到以下内容的语句中（用花括号括起来）：

   ```
   //ADD STATE MACHINE CODE HERE
   ```

例 2-9： KEYBOARD() 状态机代码

```
//delay to allow the hold capacitor on the ADC to charge
case 0:       if(++delaycounter>9000)
              {
                     delaycounter = 0;
                     state = 1;
              }
              break;

case 1:       ADCON0bits.GO_DONE = 1; //Start an ADC conversion
              state = 2;
              break;


case 2:        if(ADCON0bits.GO_DONE == 0) //Check if conversion is
                                           //completed
              {
                     state = 3;
              }
              break;


case 3:       HIDoutput = ADRESH>>3;//shift the result in ADRESH
                                    //left by three

              if(HIDoutput<=4) HIDoutput = 4;
              if(HIDoutput>=29) HIDoutput = 29;

              //Can the SIE transmit?
              if((HIDTxHandleBusy(lastTransmission) == 0))

              {
                     //Load the HID buffer
                     hid_report_in[0] = 0;
                     hid_report_in[1] = 0;
                     hid_report_in[2] = HIDoutput;
                     hid_report_in[3] = 0;
                     hid_report_in[4] = 0;
                     hid_report_in[5] = 0;
                     hid_report_in[6] = 0;
                     hid_report_in[7] = 0;

              //Send the 8 byte packet over USB to the host.
               lastTransmission = HIDTxPacket(HID_EP,
              (BYTE*)hid_report_in, 0x08);
              state = 0;
              }
              break;
```

请注意 HID 键盘以 8 字节数据包的形式通过 USB 发送。此数据包的格式如下：

- 第一个字节用于修改符，例如 shift 或 ctrl 字符。（即同时按下典型键盘上的 shift 和字符键。）
- 保留第二个字节。
- 其余字节用于承载包含按下的所需键盘字符的数字值。

编译项目。应准确无误地完成。

测试应用

7. 连接 PICkit 2 编程器并打开 PICkit 2 编程软件。
8. 找到此实验的 *.hex* 文件并下载到 PIC18F14K50。
9. 断开 PICkit 2 的连接。打开新的记事本、 word 文档或其他文件编辑器程序并在文档内单击以设置光标。
10. 将低引脚数 USB 演示板连接到主机 PC 端口。设备应枚举为 “Keyboard Demo”。
11. 在所选的文本编辑器内，应出现一组字符条目。
12. 旋转低引脚数 USB 演示板上的电位器以更改屏幕上的字符。

鼓励用户使用此应用程序进行进一步实验，方法是参阅键盘用法页，更改状态机中发送的键盘数据包。

2.7 项目实验 4（CDC 串行仿真器）

在本实验中， PIC18F14K50 用作串行仿真器，它使用增强型通用同步 / 异步收发器（Enhanced Universal Asynchronous Synchronous Receiver Transmitter， EUSART）外设获得通过 RS-232 发送的数据并根据固件内的 USB 协议对其进行转换。许多嵌入式应用仍在使用 RS-232 接口与外部系统通信。但是，随着 USB 的普及，新的 PC 正逐渐取消 RS-232 端口。一个简单的解决方案就是通过 USB 仿真 RS-232。在此示例中，创建了一个虚拟 COM 端口，使得 USB 连接看起来好像 RS-232 COM 连接。而且，此示例还使用了已存在的 Windows 驱动程序 （例如超级终端应用程序），从而避免了修改现有软件。

对低引脚数 USB 开发板上的 RS-232 连接器进行配置，以使 PIC18F14K50 能够用作数据终端设备 （Data Terminal Equipment， DTE）来与数据通信设备 （Data Communications Equipment， DCE）（例如报警系统和调制解调器等）接口。为了满足此实验的需要并使用户无需创建自己的 DCE 接口电路，工具包中已提供了一个非调制解调器对接转换头 （Null-Modem Gender Changer）来交叉连接发送和接收线，以使低引脚数 USB 开发板从 DTE 设备转换到 DCE 设备。通过这种方法，可使用一台 PC 上的两个超级连接终端以及一个 RS-232 串行 COM 端口和一个 USB 连接来阐释串行仿真的主要概念。

注意

提供了 RS-232 引脚校正器 （p/n 04-02087R1）的工具包不需要非调制解调器对接转换头，而是需要不交叉连接发送线和接收线的公 / 母对接转换头。在此实验中不使用引脚校正器，这是因为此版本的低引脚数 USB 开发板上的 RS-232 连接器被配置为 DCE 设备。将低引脚数 USB 开发板用作 DTE 设备的应用需要使用引脚校正器。

Microchip 的全速 USB 固件框架提供了其所有 CDC 应用程序示例的信息文件（*.inf*），这些应用程序示例自动执行 Windows 驱动程序更改，避免了用户手动操作。在 PIC18F14K50 被编程并连接到 PC USB 端口后， Windows 的 “New Hardware Found Wizard”（找到新硬件向导）将提示用户其他驱动程序信息。这时，用户只需要将 Windows 定向到包含相应 *.inf* 文件的目录即可。

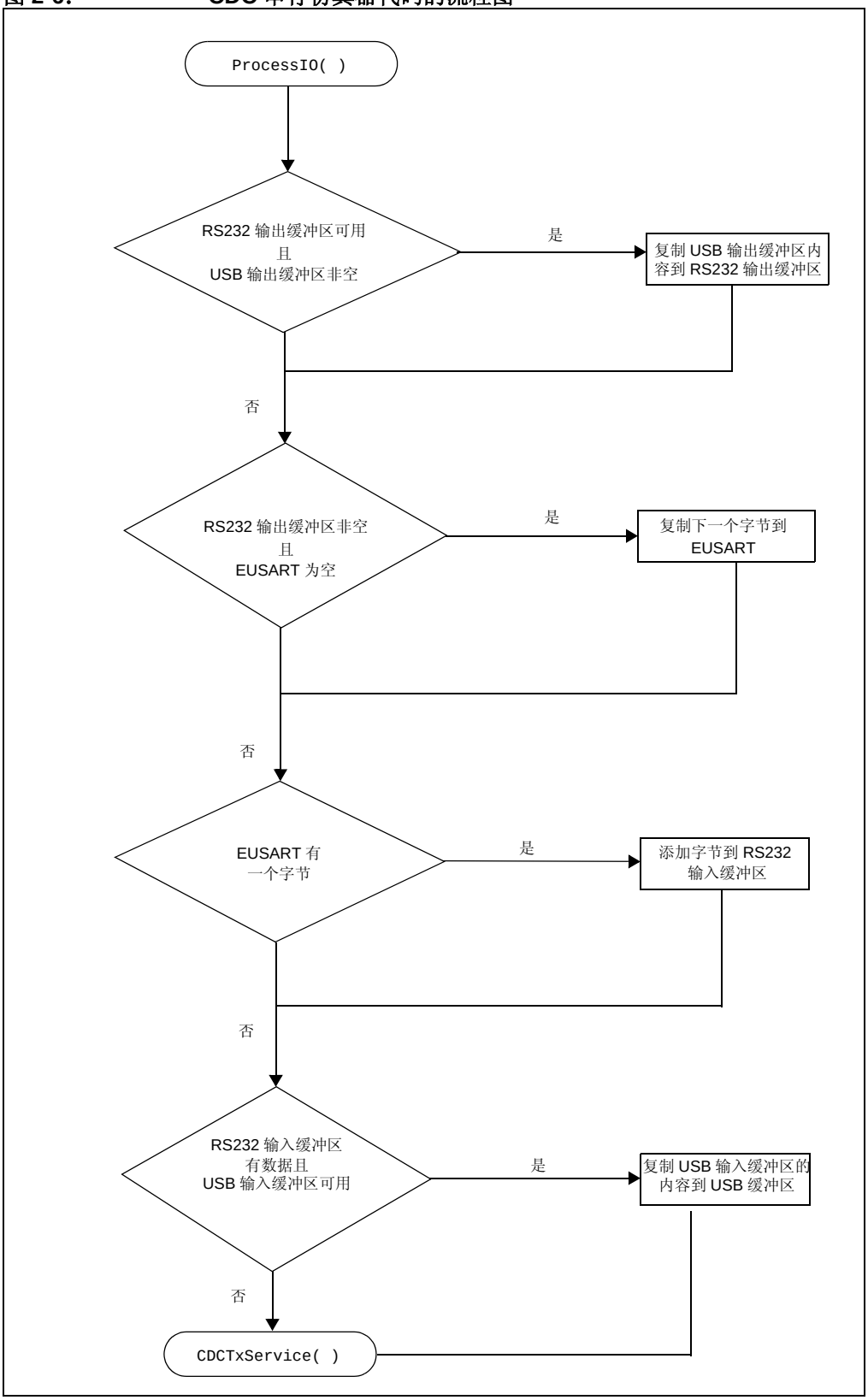
注： 在 *.inf* 中用户需要的唯一信息是特定于其原始设计的制造商标识 （VID）和产品标识 （PID）号。

Microchip的全速USB固件框架提供了执行低级别RS-232功能所需的所有源代码，用户可从中提取这些源代码。

2.7.1 CDC 串行仿真器固件概述

图 2-6 给出了 CDC 串行仿真器固件的流程图。

图 2-6： CDC 串行仿真器代码的流程图



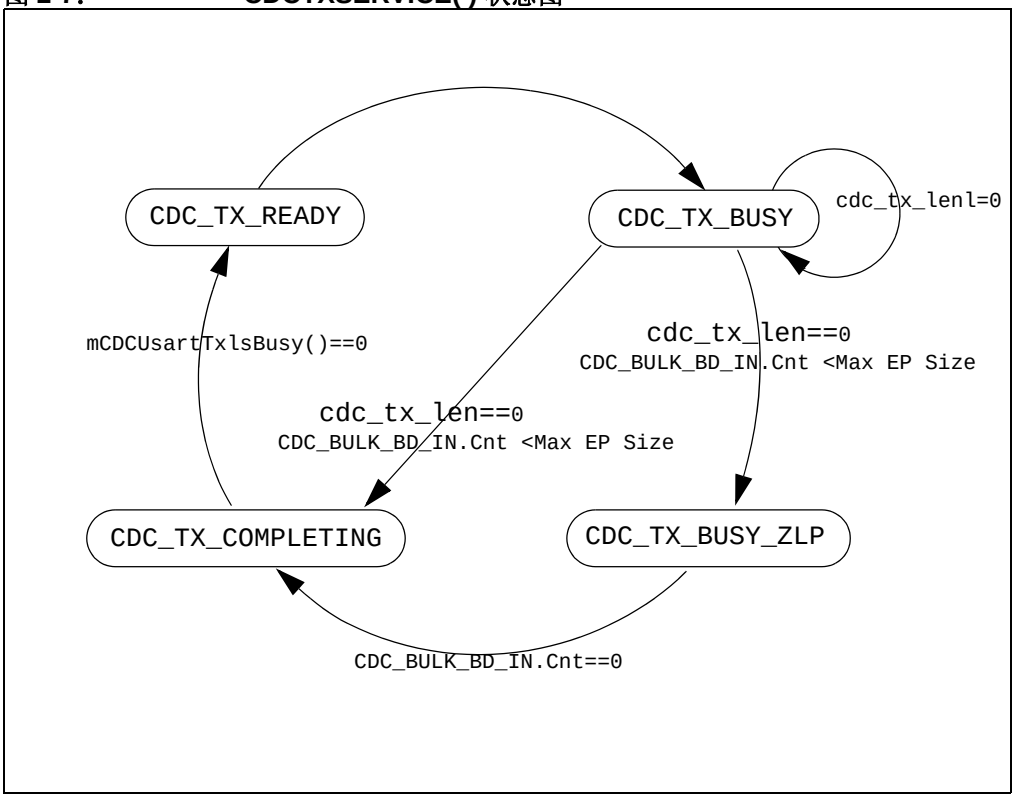
参见图 2-6 中的流程图， `ProcessIO()` 中的固件首先使用 `RS232_Out_Data_Rdy` 标志检查是否已通过 USB 完成上一次 RS-232 发送。如果该标志清零 （表示已完成上一次发送），那么固件将检查是否从 RS-232 连接发送了任何新数据并已准备好使用 `getsUSBUSART()` 通过 USB 进行发送。此函数将数据复制到缓冲区然后返回缓冲区包含的字节数。它可以确保实际上只复制了需要的字节数 （这里是 64）到此缓冲区。而且，如果不存在数据的话，函数返回零值，表示不存在数据。通过这个方法，函数不必等待数据，因而不会发生数据阻塞，记住所有固件必须适合此多任务环境。

在 RS-232 数据检查之后，固件将检查 EUSART 发送寄存器 TXREG 是否为空。使用 `mTxRdyUSART()` 宏来完成此操作，它将检查 EUSART 外设中 TXSTA （发送状态控制）寄存器中的 TRMT 位。如果 TRMT 位清零，则 TXREG 为满，如果 TRMT 位置 1，则 TXREG 为空。请注意，在从 TXREG 成功发送后，该位自动置 1。如果置 1，那么每次执行主循环时会将 `getsUSBUSART()` 收集到缓冲区中的数据一次传送一个字节到 TXREG 中。而且，此类宏以不阻塞的方式进行低级别 RS-232 通信，所以用户不必手动执行。如果 TXREG 非空，表示前一个数据尚未通过 USB 发送，因此不应被改写。

然后固件检查 CDC 类设备是否准备好发送数据。通过 `mUSBUSARTIsTxrfReady()` 标志来完成此操作。用户必须确保在调用 `putUSBUSART()` 函数之前此标志置为 `1`。为了安全起见，此函数检查状态多次以确保它没有覆盖任何等待处理的事务。此函数才向 USB 写入数据。

`CDCTxSevice()`用于传送数据给主机。此函数跟踪状态机并将长数据字符串分成多个 USB 数据包。每次执行主程序循环时调用该函数一次。`CDCTxService()` 的状态机如图 2-7 所示，在 `usb_function_cdc.c` 源文件中可找到相应的源代码。鼓励读者参考此固件并将其与空闲时的状态图进行比较。

图 2-7： CDCTXSERVICE( ) 状态图



2.7.2 实验步骤

此应用需要对 *usb_descriptor.c* 文件进行大量更改以将 PIC18F14K50 配置为 CDC 设备。请注意没有报告描述符。鼓励用户花些时间查看 *usb_descriptor.c* 文件并将其与本章开头引用的通信设备的通用串行总线类定义文档中找到的信息进行比较。请注意此实验采用的固件与 Microchip 的全速 USB 固件框架应用程序示例的 CDC 串行仿真器应用程序示例相同，可供此实验参考。

1. 与之前实验一样，使用 Project Wizard 为实验 4 创建一个名为 “Project Lab 4” 的新项目。这里仅需添加的只有 `usb_descriptor.c`、`main.c` 源文件以及在以下目录中找到的实验 4 `rm18f14k50.lkr` 的新的唯一描述符：
   `C:\LPCUSBDK_Labs\Lab4_files`

   单步执行后面的操作以关闭 Project Wizard。这次将使用子文件夹把项目设置为类似于框架中的应用程序示例，以区别和组织项目窗口中的其他源文件。所有其他源文件 / 头文件都将从项目窗口添加。
2. 右键单击项目窗口中的 **Source Files** （源文件）文件夹并选择 **Create Subfolder**...（创建子文件夹）

图 2-8：　　在项目窗口中创建子文件夹

将新文件夹命名为 “USB Stack”，然后单击 **OK**。

3. 右键单击此新 USB Stack 文件夹并选择 **Add Files**（添加文件）。在 Add Files to Project（添加文件到项目）窗口中，导航到 `C:\Microchip Solutions\Microchip\Usb` 并选择 `usb_device.c` 源文件。确保选中 “System: File(s) are External to Project, Use Absolute Path”（系统：文件是项目外部文件，请使用绝对路径），然后单击 **Open**（打开）。

图 2-9：　　添加 *USB_DEVICE.C* 文件到 USB STACK 文件夹

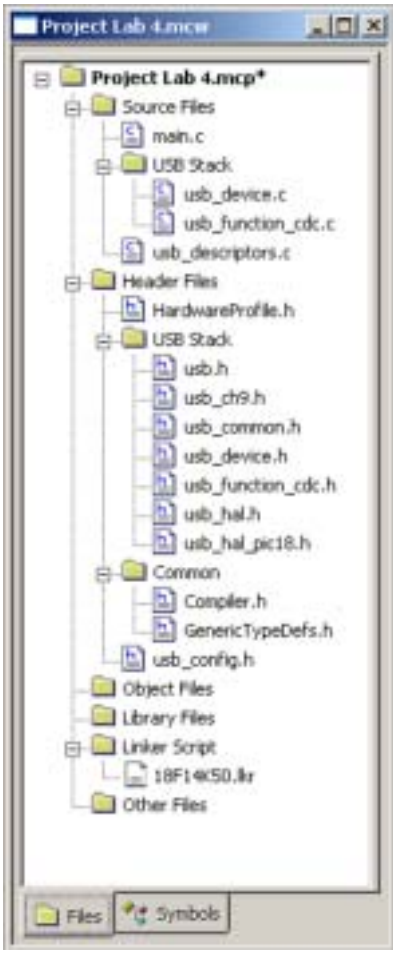
这会在项目窗口中将 *usb_device.c* 添加到 “USB Stack”文件夹。

重复此步骤以添加 C:\Microchip Solutions\Microchip\USB\CDC Device Driver 目录下的 usb_function_cdc.c 文件到同一 “USB Stack” 文件夹。

4. 在项目窗口中，按照此实验的第 2 步在 “Header Files” 文件夹下创建两个分别名为 “Common” 和 “USB Stack” 的新子文件夹。
5. 右键单击 “USB Stack” 文件夹并按照此实验的第 3 步添加 C:\Microchip Solutions\Microchip\Include\Usb 目录下的以下文件：
   - usb.h
   - usb_ch9.h
   - usb_common.h
   - usb_device.h
   - usb_function_cdc.h
   - usb_hal.h
   - usb_hal_pic18.h
6. 右键单击 “Common” 文件夹并按照此实验的第 3 步添加 C:\Microchip Solutions\Microchip\Include 目录下的以下文件：
   - Compiler.h
   - GenericTypeDefs.h
7. 确保 Project Build 选项按照实验 1 的第 11-15 步进行配置。
8. 编译项目。应准确无误地完成。

项目窗口现在应如图 2-10 所示。

图 2-10：　　实验 4 的项目窗口



9. 接下来需要初始化 EUSART 外设以使能波特率为 19200 bps 的异步通信。要执行此操作，请打开 *main.c* 文件并向下滚动到 `InitializeUSART()`。此函数由 `UserInit()` 调用。请注意在此函数中，代码具有固定的格式，允许根据所使用的特定器件和编译器进行配置。将例 2-10 的内容复制粘贴到 `InitializeUSART()` 函数中标有以下内容的段内：

   //ADD C18 PIC18F14K50 EUSART INITIALIZATION CODE HERE

例 2-10： INITIALIZEUSART() 代码

```
#if defined(__18CXX)
      unsigned char c;
    #if defined(__18F14K50)
      ANSELHbits.ANS11 = 0;      // Make RB5 digital so USART can
                                 //use pin for Rx
    #endif
    UART_TRISRx=1;               // RX
    UART_TRISTx=0;               // TX
    TXSTA = 0x24;                // TX enable BRGH=1
    RCSTA = 0x90;                // Single Character RX
    SPBRG = 0x70;
    SPBRGH = 0x02;               // 0x0271 for 48MHz -> 19200 baud
    BAUDCON = 0x08;              // BRG16 = 1
    c = RCREG;                   // read
#endif
```

鼓励读者查阅 PIC18F14K50 数据手册的 EUSART 章节，以了解更多关于配置代码的特定信息。

然后，将添加实现图 2-6 所示流程图的应用特定代码。

10. 向下滚动到 `ProcessIO()`，然后复制粘贴例 2-11 的内容到标有以下内容的段内：

```
/*****************************************************
ADD CODE TO CHECK IF RS232 HAS
BEEN SENT ALONG USB
AND THE CODE TO CHECK IF
ANY NEW RS232 TRANSMISSION
HAS BEEN RECEIVED AND STORED
*****************************************************/
```

正如图 2-6 中的流程图所示，此代码将确保包含从 RS-232 发送的数据的缓冲区内容被发送到 USB 固件。如果这样，代码将检查缓冲区中存储的任何新 RS-232 数据。

例 2-11： RS-232 缓冲区检查

```
// only check for new USB buffer if the old RS232 buffer is
// empty.
// Additional USB packets will be NAK'd
// until the buffer is free.

if (RS232_Out_Data_Rdy == 0)
       {
              LastRS232Out = getsUSBUSART(RS232_Out_Data,64);
              if(LastRS232Out > 0)
              {
                     RS232_Out_Data_Rdy = 1;  // signal
                                              //buffer full
                     RS232cp = 0;// Reset the current position
              }
       }
```

11. 然后，输入检查 EUSART 发送移位寄存器 （TXREG）并装入 RS-232 缓冲区内容的代码。复制粘贴例 2-12 的内容到标有以下内容的段内：

```
/*******************************************************
ADD THE CODE THAT WILL CHECK
IF THE EUSART TXREG IS EMPTY.
IF SO, BEGIN SENDING DATA
FROM RS232 TRANSMISSION INTO
THE TXREG ONE BYTE AT A TIME
*******************************************************/
```

例 2-12： TXREG 检查并装载代码

```
       if(RS232_Out_Data_Rdy && mTxRdyUSART())
       {
              putcUSART(RS232_Out_Data[RS232cp]);
              ++RS232cp;
              if (RS232cp == LastRS232Out)
                     RS232_Out_Data_Rdy = 0;
}
```

12. 最后，输入检查 CDC 类设备是否准备好发送数据到 USB 发送缓冲区的代码。复制粘贴例 2-13 的内容到标有以下内容的段内：

```
/*******************************************************
ADD THE CODE THAT WILL CHECK
IF THE CDC CLASS DEVICE IS
READY TO LOAD THE USB BUFFER
*******************************************************/
```

例 2-13：　　检查 CDC 类设备代码

```
if((mUSBUSARTIsTxTrfReady()) && (NextUSBOut > 0))
{
   putUSBUSART(&USB_Out_Buffer[0], NextUSBOut);
   NextUSBOut = 0;
}
```

13. 到此为止，运行CDC串行仿真器应用程序所需的所有代码都已完成。编译项目。应准确无误地完成。

安装应用驱动程序

14. 连接 PICkit 2 编程器并打开 PICkit 2 编程软件。
15. 找到此实验的 *.hex* 文件并下载到 PIC18F14K50。
16. 断开与 PICkit 2 的连接。
17. 将低引脚数USB演示板连接到主机PC端口。Windows应将PIC18F14K50识别为“CDC RS-232 Emulation Demo”。

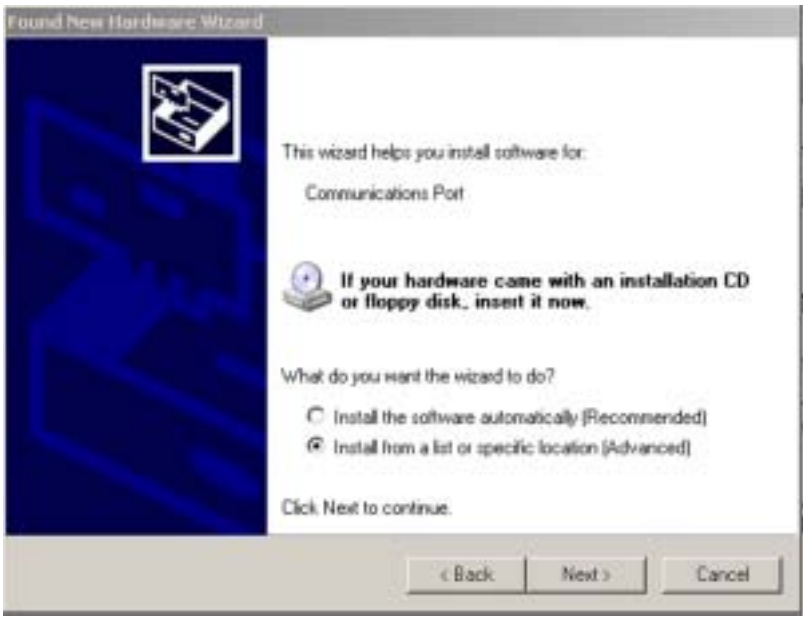
Windows 现在提示用户驱动程序信息。

18. 在 “Welcome to the Found New Hardware Wizard”（欢迎使用找到新硬件向导）窗口中，选择 “No, not this time”（不，这次不需要）然后选择 **Next** （见图 2-11）。

图 2-11：　　FOUND NEW HARDWARE WIZARD 窗口

19. 然后向导提示用户装入通信端口软件的出处。选择“Install from a list or specific location (Advanced)”（从列表或指定位置安装 （高级）），然后单击 **Next** （见图 2-12）。

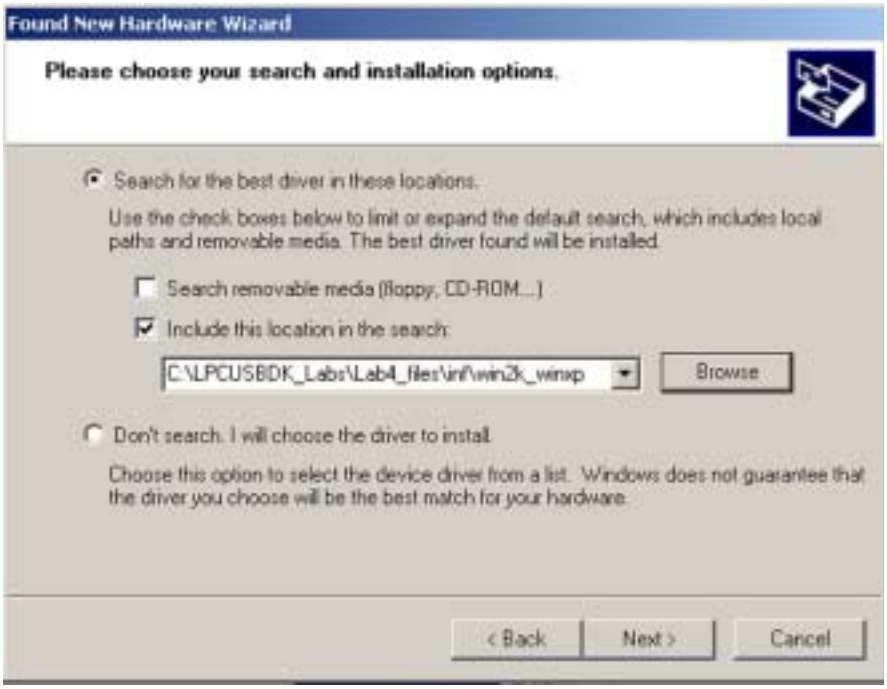
图 2-12： 选择通信端口的软件位置



向导现在提示用户 *.inf* 文件的位置， Windows 将使用该文件自动配置二进制驱动程序文件（*.sys* 文件），来创建 USB 的虚拟 COM 端口连接。确保选中了“Search for the best driver in these locations”（在这些位置上搜索最佳驱动程序）和 “Include this location in the search”（在搜索中包括这个位置）。选择 **Browse** （浏览）并浏览到 `C:\LPCUSBDK_Labs\Lab4_files\inf\win2k_winxp` 目录下的实验 4 源文件（见图 2-13）。

选中 `win2k_winxp` 文件夹并单击 **OK**。

图 2-13：　　将 WINDOWS 定向到 CDC 串行仿真器应用的 *.INF* 文件



单击 **Next**。

窗口现在应该开始装载软件。如果发出任何警告，请选择 **Continue Anyway**（继续安装）。

20. 向导应指示已成功安装通信端口的软件。选择 **Finish**（完成）（见图 2-14）。

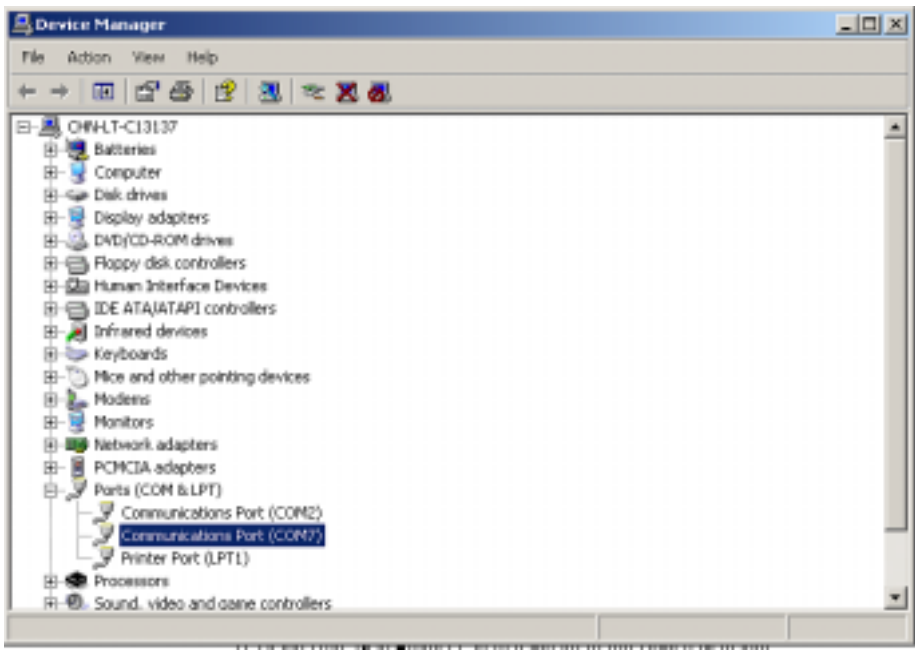
图 2-14：　　成功安装软件窗口

然后，将在设备管理器中检查虚拟 COM 端口以标识 COM 端口号。

建立通信

21. 使用项目实验 1 的第 29 步，导航到 Device Manager 并展开端口（COM 和 LPT）。在 Device Manager 窗口的 Ports (COM & LPT) （端口（COM & LPT））下拉列表中应能找到 *.inf* 文件所创建的新虚拟 COM 端口 （在大多数 PC 上通常为 COM 5 及更高编号）。如果难以找到虚拟 COM 端口，请右键单击每个驱动程序并选择 **Properties** 直到找到 CDC RS-232 Emulation Demo。请注意虚拟 COM 端口和用于 RS-232 连接 （主机 PC 上的串行端口连接）的 COM 端口的 COM 端口号。图 2-15 给出了可用 COM 端口的列表。该表可能与读者实际所见有所不同。

图 2-15： 设备管理器中可用 COM 端口的列表示例



在超级终端程序中使用此 COM 端口号建立低引脚数 USB 开发板和主机 PC 之间的 USB 连接和 RS-232 连接。

22. 通过在 Windows 内进行以下选择打开 Hyper Terminal （超级终端）窗口：

    Start>Programs>Accessories>Communications>HyperTerminal（开始 > 程序 > 附件 > 通讯 > 超级终端）。

超级终端程序应立即提示用户输入 “Connection Description”（连接描述）。

23. 将此第一个连接命名为 “USB Connection”并单击 **OK** （见图 2-16）。

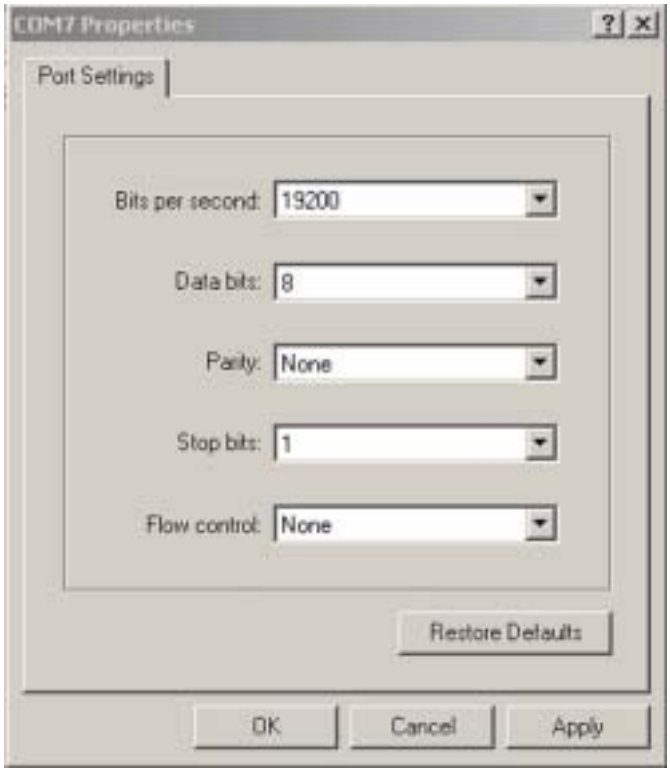
图 2-16：　　超级终端连接描述

24. 在 “Connect To”（连接到）窗口的 “Connect Using”（连接方式）下拉菜单中选择第 21 步中标出的虚拟 COM 端口并选择 **OK**。在图 2-17 中，虚拟 COM 端口在 COM7 上。请注意这可能与用户的 PC 不同。

图 2-17：　　超级终端 CONNECT TO 窗口

25. 在COM属性窗口中，按照图2-18所示配置连接且波特率为19200，单击**Apply**然后单击 **OK**。

图 2-18：　　超级终端 COM 属性窗口



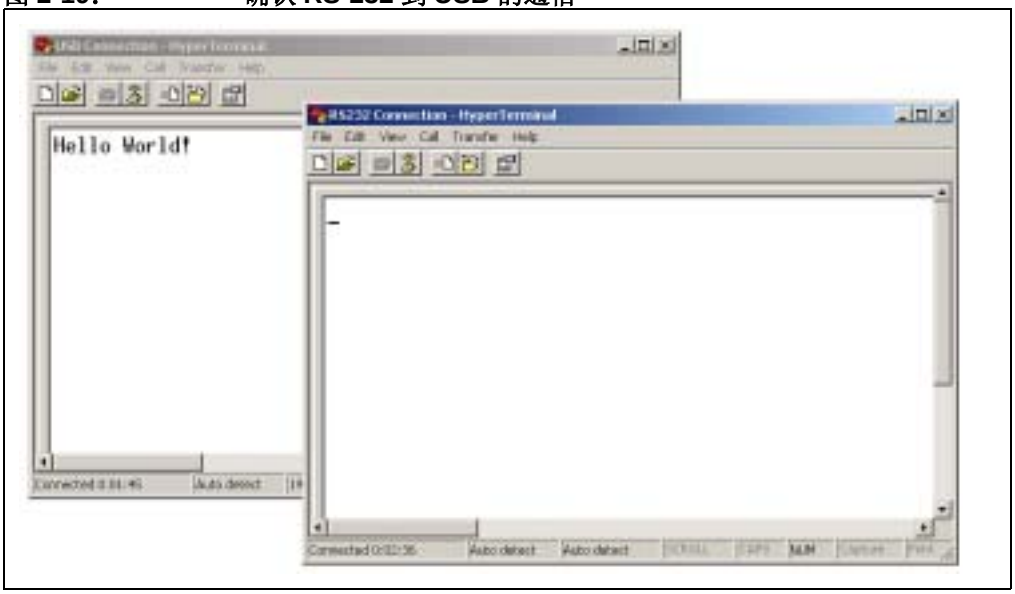
现在已经在低引脚数 USB 开发板 USB 连接器和主机 PC COM 端口之间建立了连接。

26. 然后，将 RS-232 串行电缆的一端连接到低引脚数 USB 开发板的连接器，另一端连接到主机 PC 的串行端口连接器 （使用 “对接转换头”适配器）。
27. 重复第 22 至 25 步，使用第 21 步中标出的与主机 PC 的串行端口连接器相连的端口建立 RS-232 连接。将此连接命名为 “RS232 Connection”并确保 COM 属性与图 2-18 类似。

测试应用

28. 连接后，在“RS232 Connection” COM 窗口中单击，然后输入一条消息。请注意，除非配置为本地回送，否则原始消息 COM 窗口将不会输出此消息。应在“USB Connection” COM 窗口中输出此消息，如图 2-19 所示。

图 2-19：　　确认 RS-232 到 USB 的通信



这将确认从主机 PC 通过 RS-232 连接到 PIC18F14K50 的通信， PIC18F14K50 反过来将接收到的数据发送回主机 PC。也就是说，进行了 RS-232 到 USB 的转换。

注：

附录 A 原理图

A.1 简介

此附录包含了低引脚数 USB 开发工具包的硬件图。

图 A-1： 低引脚数 USB 开发板元器件清单

数量	说明
1	IC， PIC18F14K50， 20P DIP
1	贴片 IC， MAX3232CPWR， DRVR/RCVR MLTCH RS232 16TSSO （U3）
5	贴片陶瓷电容， 0.1uF， 0603， 16V， 10%， X7R （C6 - C10）
2	贴片陶瓷电容， 0.1uF， 0805， 50V， 10%， X7R （C1， C3）
1	贴片陶瓷电容， 0.47uF， 0805， 16V， 10%， X7R （C2）
2	贴片陶瓷电容， 22pF， 0805， 100V， 5%， C0G （C4， C5）
4	贴片电阻， 330Ω， 1/16W， 1%， 0603 （R8 - R11）
2	贴片电阻， 1.0KΩ， 1/10W， 1%， 0805 （R1， R3）
1	贴片电阻， 10KΩ， 1/10W， 1%， 0805 （R2）
1	贴片电阻， 150KΩ， 1/10W， 1%， 0805 （R12）
4	贴片电阻， 470Ω， 1/10W， 1%， 0805 （R4 - R7）
1	THUMBWH 陶瓷 ST 变阻器， 10KΩ， 1/2W （POT 1）
4	亮绿色 LED， 565NM， 0805， T/R （D1 - D4）
1	表面贴装型贴片晶振， 12.000MHz， 18PF，基频 （HC49）（Y1）
1	贴片开关，单刀单掷按键， MOM， 6MM， 160GF/230GF （S1）
1	贴片板接连接器， USB 微型 B， 5POS， RA （J1）
1	D-SUB 连接器， 9P，直角插头，带插孔螺丝 （J15）
1	弯式板接连接器， 1x14 针， 0.100" （J11）
2	HDR 连接器，1x6 分离点，0.100" 排针，0.025 SQ，RA（0.230/0.090）（J6， J13）
1	HDR 连接器， 1x2， 0.100" 排针，0.025 方孔，锡焊 （0.135"/0.380"），POL （J9）
1	HDR 连接器， 1x3 分离点， 0.100" 排针， 0.025 方孔， GD （0.100"/0.230"）（J14）
1	插座， 20P DIP， 0.300W，弹簧套式筒夹 （@XU1）
1	备用位置 （U2， J2 - J5， J7， J8）

A.2 原理图

图 A-2： PICkit™ 2 USB 开发板原理图

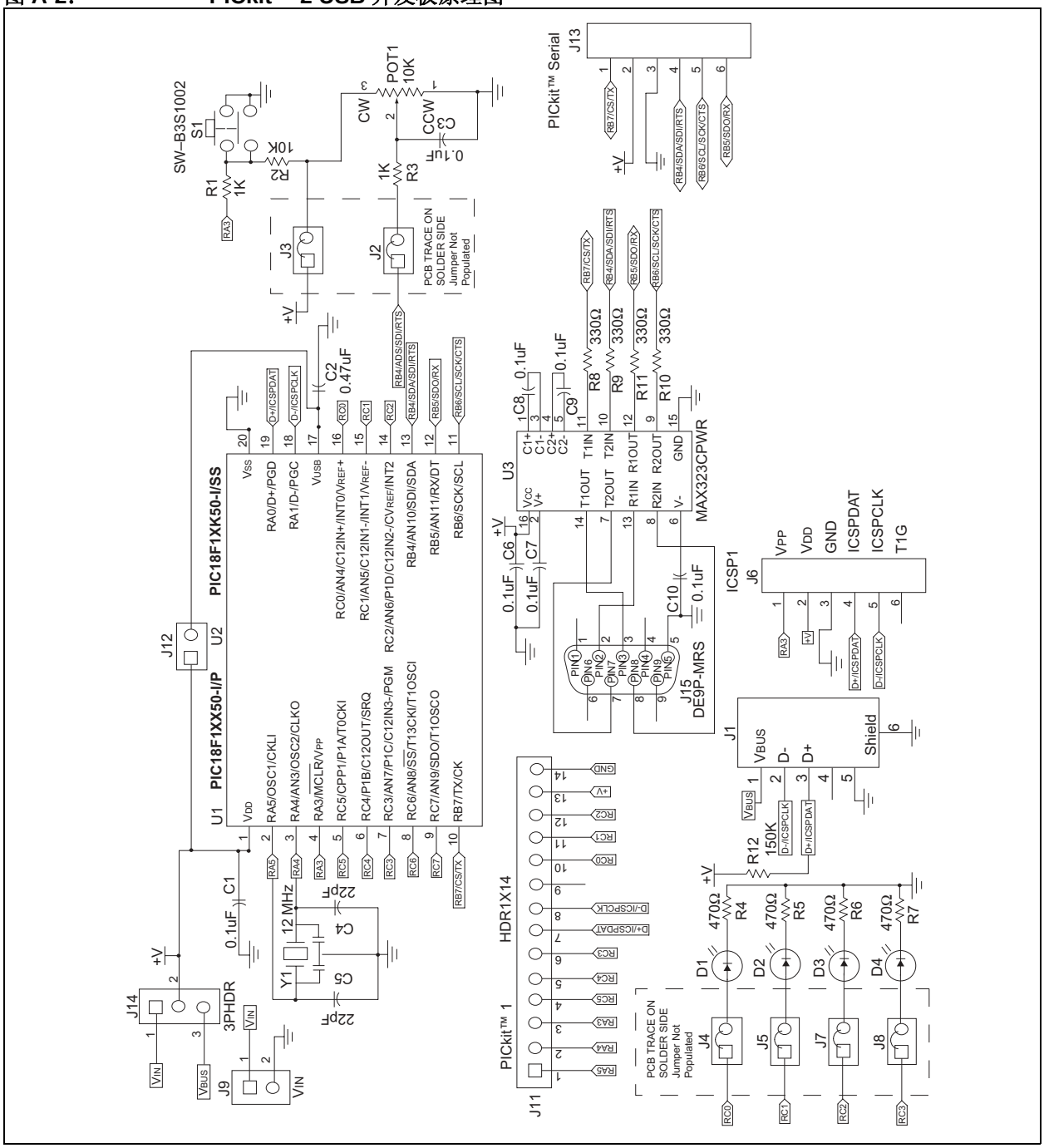


图 A-3： PICkit™ 2 调试连接头

注：

注：

全球销售及服务网点

美洲

公司总部 Corporate Office
2355 West Chandler Blvd.
Chandler, AZ 85224-6199
Tel: 1-480-792-7200
Fax: 1-480-792-7277
技术支持：
http://support.microchip.com
网址：www.microchip.com

亚特兰大 Atlanta
Duluth, GA
Tel: 678-957-9614
Fax: 678-957-1455

波士顿 Boston
Westborough, MA
Tel: 1-774-760-0087
Fax: 1-774-760-0088

芝加哥 Chicago
Itasca, IL
Tel: 1-630-285-0071
Fax: 1-630-285-0075

克里夫兰 Cleveland
Independence, OH
Tel: 216-447-0464
Fax: 216-447-0643

达拉斯 Dallas
Addison, TX
Tel: 1-972-818-7423
Fax: 1-972-818-2924

底特律 Detroit
Farmington Hills, MI
Tel: 1-248-538-2250
Fax: 1-248-538-2260

科科莫 Kokomo
Kokomo, IN
Tel: 1-765-864-8360
Fax: 1-765-864-8387

洛杉矶 Los Angeles
Mission Viejo, CA
Tel: 1-949-462-9523
Fax: 1-949-462-9608

圣克拉拉 Santa Clara
Santa Clara, CA
Tel: 408-961-6444
Fax: 408-961-6445

加拿大多伦多 Toronto
Mississauga, Ontario,
Canada
Tel: 1-905-673-0699
Fax: 1-905-673-6509

亚太地区

亚太总部 Asia Pacific Office
Suites 3707-14, 37th Floor
Tower 6, The Gateway
Harbour City, Kowloon
Hong Kong
Tel: 852-2401-1200
Fax: 852-2401-3431

中国 – 北京
Tel: 86-10-8528-2100
Fax: 86-10-8528-2104

中国 – 成都
Tel: 86-28-8665-5511
Fax: 86-28-8665-7889

中国 – 香港特别行政区
Tel: 852-2401-1200
Fax: 852-2401-3431

中国 – 南京
Tel: 86-25-8473-2460
Fax: 86-25-8473-2470

中国 – 青岛
Tel: 86-532-8502-7355
Fax: 86-532-8502-7205

中国 – 上海
Tel: 86-21-5407-5533
Fax: 86-21-5407-5066

中国 – 沈阳
Tel: 86-24-2334-2829
Fax: 86-24-2334-2393

中国 – 深圳
Tel: 86-755-8203-2660
Fax: 86-755-8203-1760

中国 – 武汉
Tel: 86-27-5980-5300
Fax: 86-27-5980-5118

中国 – 厦门
Tel: 86-592-238-8138
Fax: 86-592-238-8130

中国 – 西安
Tel: 86-29-8833-7252
Fax: 86-29-8833-7256

中国 – 珠海
Tel: 86-756-321-0040
Fax: 86-756-321-0049

台湾地区 – 高雄
Tel: 886-7-536-4818
Fax: 886-7-536-4803

台湾地区 – 台北
Tel: 886-2-2500-6610
Fax: 886-2-2508-0102

台湾地区 – 新竹
Tel: 886-3-6578-300
Fax: 886-3-6578-370

亚太地区

澳大利亚 Australia - Sydney
Tel: 61-2-9868-6733
Fax: 61-2-9868-6755

印度 India - Bangalore
Tel: 91-80-3090-4444
Fax: 91-80-3090-4080

印度 India - New Delhi
Tel: 91-11-4160-8631
Fax: 91-11-4160-8632

印度 India - Pune
Tel: 91-20-2566-1512
Fax: 91-20-2566-1513

日本 Japan - Yokohama
Tel: 81-45-471- 6166
Fax: 81-45-471-6122

韩国 Korea - Daegu
Tel: 82-53-744-4301
Fax: 82-53-744-4302

韩国 Korea - Seoul
Tel: 82-2-554-7200
Fax: 82-2-558-5932 或
82-2-558-5934

马来西亚 Malaysia - Kuala Lumpur
Tel: 60-3-6201-9857
Fax: 60-3-6201-9859

马来西亚 Malaysia - Penang
Tel: 60-4-227-8870
Fax: 60-4-227-4068

菲律宾 Philippines - Manila
Tel: 63-2-634-9065
Fax: 63-2-634-9069

新加坡 Singapore
Tel: 65-6334-8870
Fax: 65-6334-8850

泰国 Thailand - Bangkok
Tel: 66-2-694-1351
Fax: 66-2-694-1350

欧洲

奥地利 Austria - Wels
Tel: 43-7242-2244-39
Fax: 43-7242-2244-393

丹麦 Denmark-Copenhagen
Tel: 45-4450-2828
Fax: 45-4485-2829

法国 France - Paris
Tel: 33-1-69-53-63-20
Fax: 33-1-69-30-90-79

德国 Germany - Munich
Tel: 49-89-627-144-0
Fax: 49-89-627-144-44

意大利 Italy - Milan
Tel: 39-0331-742611
Fax: 39-0331-466781

荷兰 Netherlands - Drunen
Tel: 31-416-690399
Fax: 31-416-690340

西班牙 Spain - Madrid
Tel: 34-91-708-08-90
Fax: 34-91-708-08-91

英国 UK - Wokingham
Tel: 44-118-921-5869
Fax: 44-118-921-5820

03/26/09