



工业和信息化部“十三五”
人才培养规划教材



嵌入式技术与应用开发 项目教程 | (STM32 版)

Embedded Technology and Application Development


郭志勇 © 编著



名师打造**精品教材**，包括 **8** 个项目、**19** 个任务、**15** 个技能训练。

采用**企业真实任务**、贴近职业**岗位需求**，改善**学习体验**。

“任务驱动、做中学”，全新的**仿真教学模式**，案例全部采用 **C 语言** 编程实现。

 中国工信出版集团

 人民邮电出版社
POSTS & TELECOM PRESS

“十三五”
规划教材



嵌入式技术与应用开发 项目教程 | (STM32 版)

Embedded Technology and Application Development

郭志勇 编著

人民邮电出版社
北京

图书在版编目 (CIP) 数据

嵌入式技术与应用开发项目教程：STM32版 / 郭志勇编著. — 北京：人民邮电出版社，2019.5
工业和信息化“十三五”人才培养规划教材
ISBN 978-7-115-50826-3

I. ①嵌… II. ①郭… III. ①微处理器—系统设计—高等学校—教材 IV. ①TP332

中国版本图书馆CIP数据核字(2019)第054346号

内 容 提 要

本书基于 ST 公司的 STM32 芯片，包括 8 个项目、19 个任务，分别介绍 LED 控制设计与实现、跑马灯控制设计与实现、数码管显示设计与实现、按键控制设计与实现、定时器应用设计与实现、串行通信设计与实现、模数转换设计与实现以及嵌入式智能车设计与实现等内容，涵盖了嵌入式系统的基本知识和嵌入式应用开发的基本内容。

本书引入 Proteus 仿真软件，采用“任务驱动、做中学”的编写思路，每个任务均将相关知识和职业岗位技能融合在一起，将知识、技能的学习结合任务完成过程来进行。

本书可作为嵌入式技术与应用、物联网应用技术等电子信息类专业嵌入式课程的教材，也可作为广大智能电子产品制作爱好者的自学用书。

-
- ◆ 编 著 郭志勇
责任编辑 祝智敏
责任印制 马振武
 - ◆ 人民邮电出版社出版发行 北京市丰台区成寿寺路 11 号
邮编 100164 电子邮件 315@ptpress.com.cn
网址 <http://www.ptpress.com.cn>
山东百润本色印刷有限公司印刷
 - ◆ 开本：787×1092 1/16
印张：19.75 2019 年 5 月第 1 版
字数：497 千字 2019 年 5 月山东第 1 次印刷
-

定价：59.80 元

读者服务热线：(010)81055256 印装质量热线：(010)81055316

反盗版热线：(010)81055315

广告经营许可证：京东工商广登字 20170147 号

本书紧随嵌入式技术领域的最新发展趋势,顺应现代高等教育指导思想的发展潮流,突出技能培养在课程中的主体地位,用任务引领理论,使理论从属于技能实践。

本书以解决实际项目为主线,连贯多个知识点,每个项目均由若干个具体的典型任务组成,每个任务均将相关知识和职业岗位基本技能融合在一起,把知识、技能的学习融入到任务完成的过程中。任务均是一个完整的嵌入式系统实际工作过程,既拉近了教学与职业岗位需求之间的距离,又兼顾了知识的系统性和完整性。同时本书还引入了 Proteus 仿真软件,使读者从 STM32 复杂的硬件结构中解放出来,实现了在计算机上完成 STM32 电路设计、软件设计、调试与仿真等一系列工作,便于读者掌握从设计到产品的完整过程。

本书基于 ST 公司的 STM32 芯片,共有 8 个项目 19 个任务。采用“项目引导、任务驱动”的编写模式,突出“做中学”的基本理念。前 7 个项目主要介绍嵌入式系统的基本概念、基本知识,嵌入式应用系统的编程入门以及用 C 语言进行程序设计、运行、调试等内容,培养读者分析问题和解决问题的能力。最后一个项目主要围绕全国职业院校技能大赛“嵌入式技术与应用开发”赛项的竞赛平台(嵌入式智能车),介绍嵌入式智能车的停止、前进、后退、左转、右转、速度和寻迹等控制,以及对道闸、LED 显示(计时器)、立体旋转、隧道风扇、烽火台报警等标志物控制,并完成光强度测量和超声波测距等任务,培养读者嵌入式技术与应用开发的能力。

本书建议教学学时为 60~90 学时,参考学时分配为:项目一 6~10 学时、项目二 6~10 学时、项目三 6~8 学时、项目四 8~10 学时、项目五 8~10 学时、项目六 6~10 学时、项目七 4~8 学时、项目八 16~24 学时。

本书已获安谋科技(中国)有限公司(Arm China)和百科荣创(北京)科技发展有限公司认可,可作为全国职业院校技能大赛“嵌入式技术与应用开发”赛项的培训教材。课程配套资源丰富,有自主学习的嵌入式技术与应用开发资源包、电子教案、课件、源代码和仿真电路、技能大赛案例以及校企合作资源及相关其他素材等。

本书由安徽电子信息职业技术学院省级教学名师郭志勇编著,由百科荣创(北京)科技发展有限公司张明伯、石浪、黄文昌、杨贵明等技术人员提供全国职业院校技能大赛“嵌入式技术与应用开发”赛项中的典型应用项目,并对本书的编写提供了宝贵的参考意见和相关课程资源。参加本书电路调试、程序调试、校对等工作的还有郑其、王顺顺、钱政、彭瑾、杨振宇、郭丽等,在此一并表示衷心感谢。

由于时间紧迫和编者水平有限,书中难免会有疏漏和不妥之处,敬请广大读者和专家批评指正。

编者

2018 年 10 月

项目一 LED 控制设计与实现..... 1	2.2 Cortex-M3 的编程模式.....50
1.1 任务 1 新建一个基于 STM32 固件库的工程模板..... 1	2.2.1 Cortex-M3 工作模式及状态.....50
1.1.1 新建基于 STM32 固件库的 Keil μ Vision4 工程模板..... 1	2.2.2 Cortex-M3 寄存器组.....51
1.1.2 认识 STM32 固件库..... 11	2.2.3 Cortex-M3 特殊功能寄存器组.....54
1.1.3 STM32 固件库关键字目录和文件..... 12	2.3 任务 5 跑马灯设计与实现.....56
1.2 任务 2 点亮一个 LED..... 15	2.3.1 跑马灯电路设计.....56
1.2.1 用 Proteus 设计第一个 STM32 的 LED 控制电路..... 15	2.3.2 跑马灯程序设计、运行与调试.....57
1.2.2 开发第一个基于工程模板的 Keil μ Vision4 工程..... 23	2.3.3 C 语言中的预处理.....59
1.2.3 位操作..... 25	2.3.4 结构体.....61
1.3 认识 Arm-STM32..... 26	【技能训练 2-2】结构体应用——GPIO 端口初始化.....62
1.3.1 嵌入式系统..... 26	2.4 STM32 结构.....63
1.3.2 Arm Cortex-M3 处理器..... 28	2.4.1 Cortex-M3 处理器结构.....63
1.3.3 STM32 系列处理器..... 29	2.4.2 STM32 系统结构.....65
1.4 任务 3 LED 闪烁控制..... 31	2.4.3 STM32 时钟配置.....67
1.4.1 LED 闪烁控制设计与实现..... 31	【技能训练 2-3】基于寄存器的跑马灯设计.....72
1.4.2 extern 变量声明..... 32	关键知识点小结.....73
1.4.3 Keil μ Vision4 文本美化..... 33	问题与讨论.....75
【技能训练 1-1】音频产生器..... 35	
关键知识点小结..... 36	
问题与讨论..... 38	
项目二 跑马灯控制设计与实现.....39	项目三 数码管显示设计与实现..... 76
2.1 任务 4 LED 循环点亮控制..... 39	3.1 任务 6 数码管静态显示设计与实现.....76
2.1.1 认识 STM32 的 I/O 口..... 39	3.1.1 认识数码管.....76
2.1.2 STM32 的 GPIO 初始化和输入输出库函数..... 45	3.1.2 数码管静态显示电路设计.....78
2.1.3 LED 循环点亮控制设计..... 47	3.1.3 数码管静态显示程序设计.....79
【技能训练 2-1】GPIO_SetBits ()和 GPIO_ResetBits ()函数应用..... 49	【技能训练 3-1】共阳极 LED 数码管应用.....81
	3.2 STM32 存储器映射.....83
	3.2.1 认识 Cortex-M3 存储器.....83
	3.2.2 Cortex-M3 存储器映射.....84
	3.2.3 STM32 存储器映射.....86
	【技能训练 3-2】编写外部设备文件.....89
	3.3 任务 7 数码管动态扫描显示设计与实现.....91

3.3.1 数码管动态扫描显示电路设计	92	关键知识点小结	144
3.3.2 数码管动态扫描显示程序设计、 运行与调试	92	问题与讨论	145
3.3.3 Keil μ Vision4 代码编辑	94		
3.4 I/O 口的位操作与实现	99	项目五 定时器应用设计与实现	146
3.4.1 位带区与位带别名区	99	5.1 任务 10 基于 SysTick 定时器的 1 秒延时设计与实现	146
3.4.2 位带操作	100	5.1.1 SysTick 定时器	146
3.4.3 I/O 口位带操作的宏定义	102	5.1.2 库函数中的 SysTick 相关函数	149
3.4.4 I/O 口的位操作实现	105	5.1.3 SysTick 的关键函数编写	151
【技能训练 3-3】I/O 口的位操作应用	106	5.1.4 基于 SysTick 定时器的 1 秒延时 设计与实现	153
关键知识点小结	107	5.2 任务 11 STM32 定时器的 定时设计与实现	156
问题与讨论	108	5.2.1 认识 STM32 定时器	156
项目四 按键控制设计与实现	109	5.2.2 STM32 定时器与定时相关的 寄存器	157
4.1 任务 8 按键控制 LED 设计 与实现	109	5.2.3 STM32 定时器相关的库函数	162
4.1.1 认识嵌入式应用技术与 开发的核心板	109	5.2.4 STM32 定时器的定时设计	165
4.1.2 按键控制 LED 电路设计	110	【技能训练 5-1】基于寄存器的 STM32 定时器定时设计与实现	167
4.1.3 按键控制 LED 程序设计	113	5.3 任务 12 PWM 输出 控制电机	169
4.1.4 按键控制 LED 运行与调试	118	5.3.1 STM32 的 PWM 输出相关寄存器	169
【技能训练 4-1】一键多功能按键 识别设计与实现	120	5.3.2 STM32 的 PWM 输出编程思路	172
4.2 GPIO 和 AFIO 寄存器 地址映射	122	5.3.3 STM32 的 PWM 输出相关库函数	174
4.2.1 GPIO 寄存器地址映射	122	5.3.4 PWM 输出控制电机设计	177
4.2.2 端口复用使用	125	【技能训练 5-2】基于寄存器的 PWM 输出控制电机设计与实现	180
4.2.3 端口复用重映射	126	关键知识点小结	181
【技能训练 4-2】串口 1 (USART1) 重映射实现	129	问题与讨论	183
4.3 任务 9 中断方式的按键 控制设计与实现	130	项目六 串行通信设计与实现	184
4.3.1 STM32 中断	130	6.1 STM32 的串行通信	184
4.3.2 STM32 外部中断编程	134	6.1.1 串行通信基本知识	184
4.3.3 中断方式的按键控制程序设计	137	6.1.2 认识 STM32 的 USART 串口	186
4.3.4 中断方式的按键控制工程搭建、 编译与调试	141	6.1.3 STM32 串口的相关寄存器	187
【技能训练 4-3】中断方式的声光报警器	141	6.2 任务 13 USART 串口 通信设计	190

6.2.1	STM32 串口的相关函数	190
6.2.2	STM32 的 USART1 串口通信设计	194
6.2.3	STM32 串行通信设计与调试	198
	【技能训练 6-1】基于寄存器的 STM32 串行通信设计	198
6.3	任务 14 STM32 串口无线传输设计与实现	202
6.3.1	基于 Wi-Fi 的 STM32 串口无线传输电路设计	202
6.3.2	基于 ZigBee 的 STM32 串口无线传输电路设计	204
6.3.3	嵌入式智能车通信协议	205
6.3.4	基于寄存器的 STM32 串口无线传输程序设计	207
	关键知识点小结	214
	问题与讨论	215
项目七 模数转换设计与实现 216		
7.1	STM32 的模数转换	216
7.1.1	STM32 的模数转换简介	216
7.1.2	ADC 相关的寄存器	218
7.2	任务 15 基于寄存器的 STM32 模数转换设计	223
7.2.1	STM32 的 ADC 设置	223
7.2.2	基于寄存器的 STM32 模数转换设计	225
7.2.3	基于寄存器的 STM32 模数转换运行与调试	233
7.3	任务 16 基于库函数的 STM32 模数转换设计	234
7.3.1	ADC 相关的库函数	234
7.3.2	基于库函数的 STM32 模数转换程序设计	235
7.3.3	基于库函数的 STM32 模数转换运行与调试	236
	关键知识点小结	237
	问题与讨论	238

项目八 嵌入式智能车设计与实现		239
8.1	嵌入式智能车	239
8.1.1	认识嵌入式智能车	239
8.1.2	嵌入式智能车任务板	241
8.1.3	嵌入式智能车循迹板	246
8.2	任务 17 嵌入式智能车巡航控制设计	248
8.2.1	嵌入式智能车电机驱动电路	249
8.2.2	电机正反转和速度控制程序设计	252
8.2.3	嵌入式智能车停止、前进和后退程序设计	256
8.2.4	嵌入式智能车循迹、左转和右转程序设计	260
	【技能训练 8-1】嵌入式智能车巡航控制	268
8.3	任务 18 嵌入式智能车标志物控制设计	270
8.3.1	道闸标志物控制设计	270
8.3.2	LED 显示标志物控制设计	271
8.3.3	基于红外线的标志物控制设计	274
8.3.4	智能路灯控制设计	278
8.3.5	超声波测距设计	285
8.3.6	双色灯控制程序设计	288
	【技能训练 8-2】嵌入式智能车标志物控制	289
8.4	任务 19 嵌入式智能车综合控制设计	292
8.4.1	语音播报标志物控制设计	293
8.4.2	嵌入式智能车控制运输车标志物设计	296
8.4.3	编写嵌入式智能车的任务文件	297
8.4.4	编写嵌入式智能车的主文件	303
8.4.5	嵌入式智能车综合控制工程搭建、编译、运行与调试	306
8.4.6	嵌入式智能车综合控制设计经验和技巧	307
	关键知识点小结	308
	问题与讨论	309
参考文献		310

Chapter

1

项目一

LED 控制设计与实现



学习目标

能力目标

建立基于 STM32 固件库的工程模板，通过 C 语言程序完成嵌入式 STM32 芯片输出控制，实现对 LED 控制的设计、运行与调试。

知识目标

1. 知道嵌入式系统基本概念，认识 STM32 固件库；
2. 会新建 Keil μ Vision4 工程、工程配置与编译，能构建任何一款 STM32 的最基本框架；
3. 掌握 STM32 应用程序开发经常用到的 C 语言编程知识；
4. 利用 STM32 的 GPIO 引脚，实现点亮一个 LED 和控制一个 LED 闪烁。

1.1 任务 1 新建一个基于 STM32 固件库的工程模板



任务要求

建立一个基于 V3.5 版本固件库的 Keil μ Vision4 工程模板，方便以后每次新建工程时，可以直接复制使用该模板。

1.1.1 新建基于 STM32 固件库的 Keil μ Vision4 工程模板

本书使用的是 Keil μ Vision4 版本。Keil μ Vision4 源自德国的 Keil 公司，集成了业内最领先的技术，包括 μ Vision4 集成开发环境与 RealView 编译器，支持 Arm 7、Arm 9 和最新的 Cortex-M3 核处理器，可以自动配置启动代码，集成了 Flash 烧写模块，具备强大的 Simulation 设备模拟、性能分析等功能。

1. 新建工程模板目录

下面介绍怎样建立基于 V3.5 版本固件库的工程模板目录，以后新建工程时，可以直接复制

使用该模板。

(1) 先在计算机的某个盘符下新建一个 STM32_ Project 目录，作为基于 STM32 固件库的工程模板目录。

(2) 在 STM32_Project 工程模板目录下，新建 USER、CORE、OBJ 以及 STM32F10x_FWLib 4 个子目录，如图 1-1 所示。

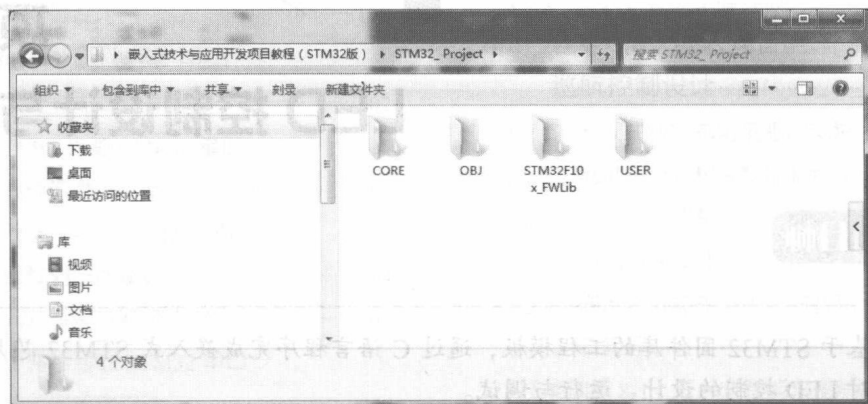


图1-1 工程STM32_Project模板目录

其中，CORE 用来存放核心文件和启动文件；OBJ 用来存放编译过程文件以及 hex 文件；STM32F10x_FWLib 用来存放 ST 公司（意法半导体公司）官方提供的库函数源码文件；USER 除了用来存放工程文件，还用来存放主函数文件 main.c，以及 system_stm32f10x.c、STM32F10x.s 等文件。另外，很多人喜欢把子目录“USER”取名为“Project”，将工程文件都保存到“Project”子目录下面，也是可以的。

(3) 把官方固件库“Libraries\STM32F10x_StdPeriph_Driver”下面的 src 和 inc 子目录复制到子目录 STM32F10x_FWLib 下面，如图 1-2 所示。

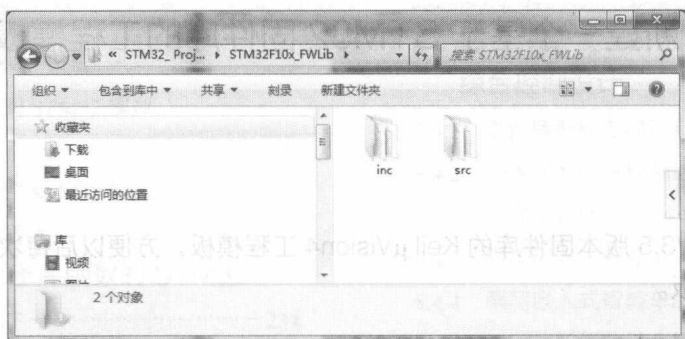


图1-2 STM32F10x_FWLib子目录

其中，src 存放的是固件库的“.c”文件，inc 存放的是其对应的“.h”文件。每个外设都对应一个“.c”文件和一个“.h”头文件。

(4) 把官方固件库里相关的启动文件复制到 CORE 文件夹里面。

把官方固件库“Libraries\CMSIS\CM3\CoreSupport”下面的 core_cm3.c 和 core_cm3.h 文件复制到子目录 CORE 下面；把“Libraries\CMSIS\CM3\DeviceSupport\ST\STM32F10x\

startup\arm” 下面的 startup_stm32f10x_ld.s 文件复制到 CORE 文件夹下面, 如图 1-3 所示。

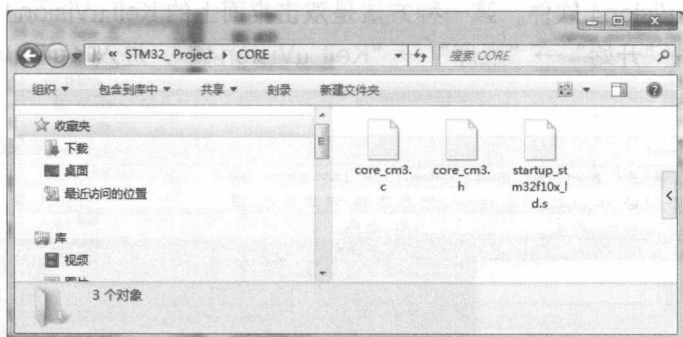


图1-3 CORE子目录



注意

本项目采用 STM32F103R6 芯片, 该芯片的 FLASH 大小是 32KB, 属于小容量产品, 所以启动文件使用 startup_stm32f10x_ld.s 文件。若采用其他容量的芯片, 可以使用 startup_stm32f10x_md.s 或 startup_stm32f10x_hd.s 启动文件。

(5) 先把官方固件库 “Libraries\CMSIS\CM3\DeviceSupport\ST\STM32F10x” 下面的 stm32f10x.h、system_stm32f10x.c、system_stm32f10x.h 文件复制到子目录 USER 下面, 然后把官方固件库 “Project\STM32F10x_StdPeriph_Template” 下面的 stm32f10x_conf.h、stm32f10x_it.c、stm32f10x_it.h 文件复制到子目录 USER 下面, 如图 1-4 所示。

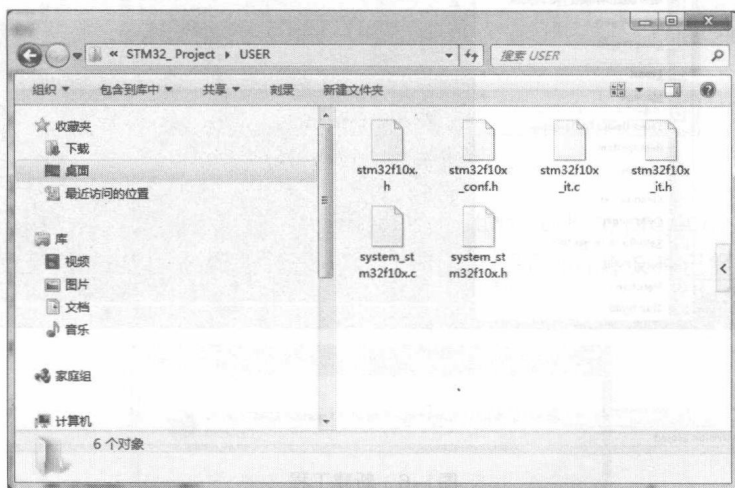


图1-4 USER子目录

通过前面几个步骤, 就把需要的官方固件库相关文件复制到了工程目录模板 “STM32_Project” 下面。在以后的任务中, 直接复制工程模板目录, 然后修改成需要的名字即可使用。

2. 新建 Keil μ Vision4 工程模板

在建立工程之前, 先在计算机的某个盘符下新建一个子目录 “任务 1 STM32_Project 工程模板”; 然后把工程目录模板 “STM32_Project” 复制到 “任务 1 STM32_Project 工程模板” 子目

录里面，并修改工程目录模板名为“STM32_Project 工程模板”。

(1) 运行 Keil μ Vision4 软件。第一种方法是双击桌面上的 Keil μ Vision4 图标；第二种方法是单击桌面左下方的“开始”→“程序”→“Keil μ Vision4”，进入 Keil μ Vision4 集成开发环境，如图 1-5 所示。

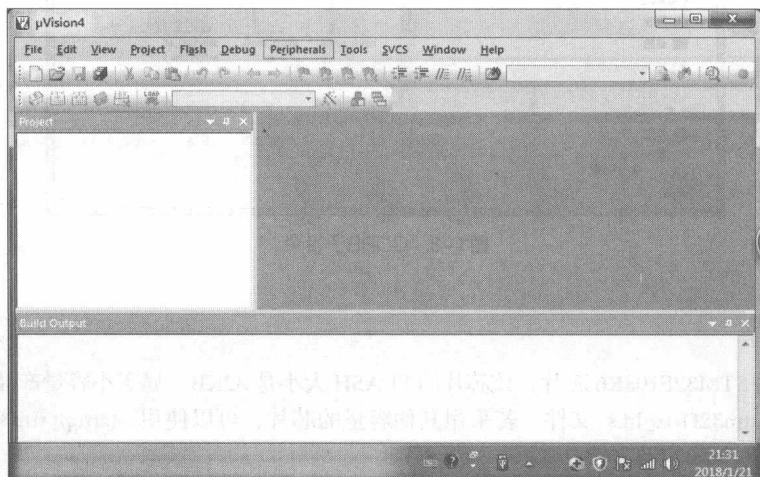


图 1-5 Keil μ Vision4 集成开发环境

(2) 单击 Project→New uVision Project，如图 1-6 所示。

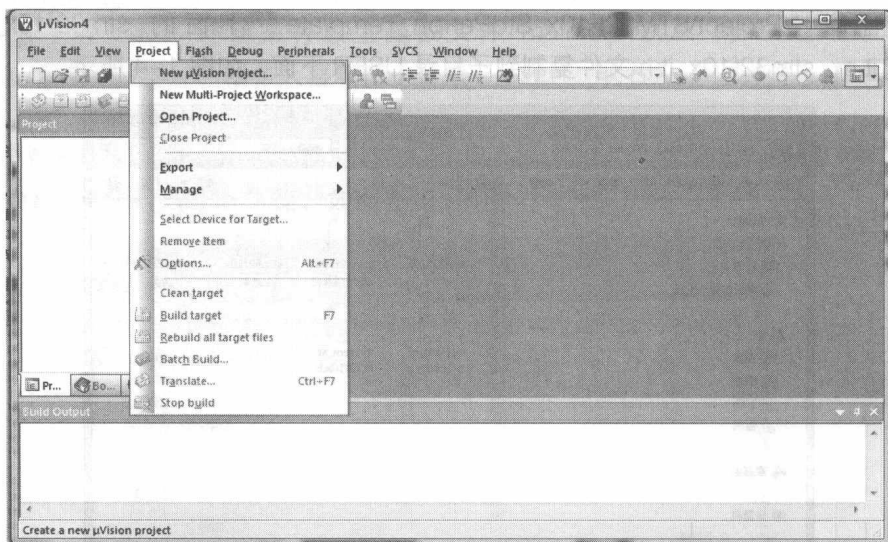


图 1-6 新建工程

然后把目录定位到“STM32_Project 工程模板\USER”下面，我们的工程文件都保存在这里。将工程命名为“STM32_Project”，单击“保存”按钮，如图 1-7 所示。

(3) 接下来弹出选择芯片“Select Device Target ‘Target1’”对话框，本项目使用的是 STM32F103R6 芯片，选择 STMicroelectronics 下面的 STM32F103R6 即可。如果使用的是其他系列的芯片，选择相应的型号就可以了，如图 1-8 所示。

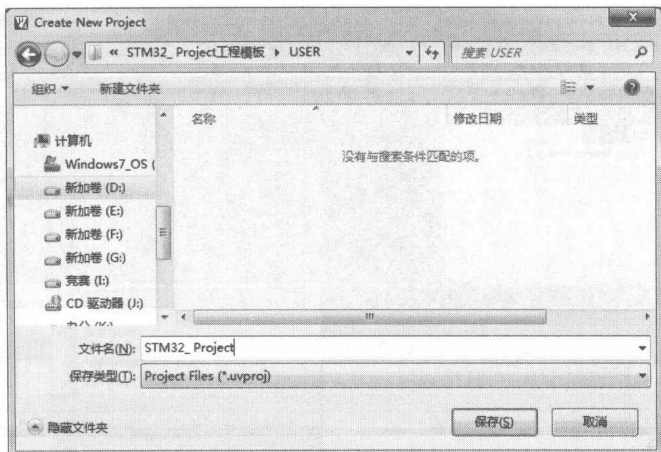


图1-7 保存新建工程

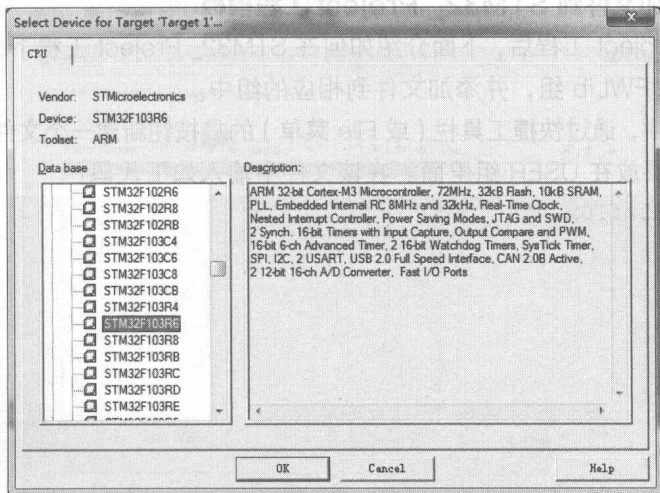


图1-8 选择芯片型号

(4) 单击 OK 按钮, 弹出对话框“Copy STM32 Startup Code to project...”, 询问是否添加启动代码到我们的工程中, 这里选择“否”, 因为我们使用的 ST 固件库文件已经包含了启动文件, 如图 1-9 所示。

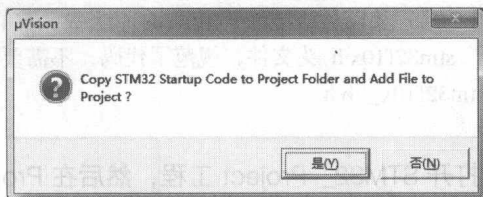


图1-9 是否添加启动代码对话框

启动代码是一段和硬件相关的汇编代码, 是必不可少的! 这段代码具体如何工作, 我们不必太关心。

现在我们可以看到新建工程后的界面, 如图 1-10 所示。

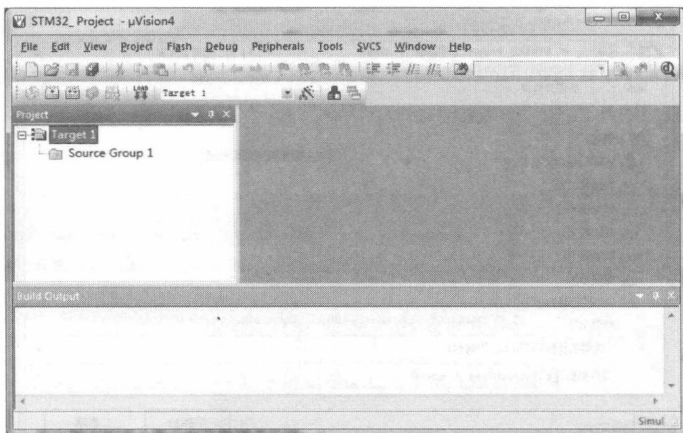



图1-10 新建工程后的界面

3. 新建组和添加文件到 STM32_Project 工程模板

建好 STM32_Project 工程后,下面介绍如何在 STM32_Project 工程下新建 USER、CORE、OBJ 和 STM32F10x_FWLib 组,并添加文件到相应的组中。

(1) 在图 1-10 中,通过快捷工具栏(或 File 菜单)的  按钮新建一个文件,并保存为 main.c,主文件 main.c 一定要放在 USER 组里面。在该文件中输入如下代码:

```
#include "stm32f10x.h"
int main(void)
{
    while(1)
    {
        ;
    }
}
```

“#include " stm32f10x.h ”语句是一个“文件包含”处理语句。stm32f10x.h 是 STM32 开发中最为重要的一个头文件,就像 C51 单片机的 reg52.h 头文件一样,在应用程序中是至关重要的,通常包括在主文件中。这里的 main()函数是一个空函数,方便以后添加需要的代码。




注意

stm32f10x.h 是 ST 公司 V3.5 及以后版本统一使用的库函数头文件,也就是把 V2.0 版本的 stm32f10x_lib.h 头文件换成了 stm32f10x.h 头文件,规范了代码,不需要再包含那么多的头文件了。使用高版本编译,将找不到 stm32f10x_lib.h。

(2) 在 USER 组里面,打开 STM32_Project 工程,然后在 Project 窗格的 Target1 上单击鼠标右键,选择 Manage Components 选项,如图 1-11 所示。

(3) 弹出图 1-12 所示 Components Environment and Books 对话框。

(4) 先把 Project Targets 栏下的 Target1 修改为 STM32_Project,把 Groups 栏下的 Source Group1 删除。然后在 Groups 栏(中间栏)单击新建  按钮(也可以双击下面的空白处),新建 USER、CORE 和 STM32F10x_FWLib 组,如图 1-13 所示。

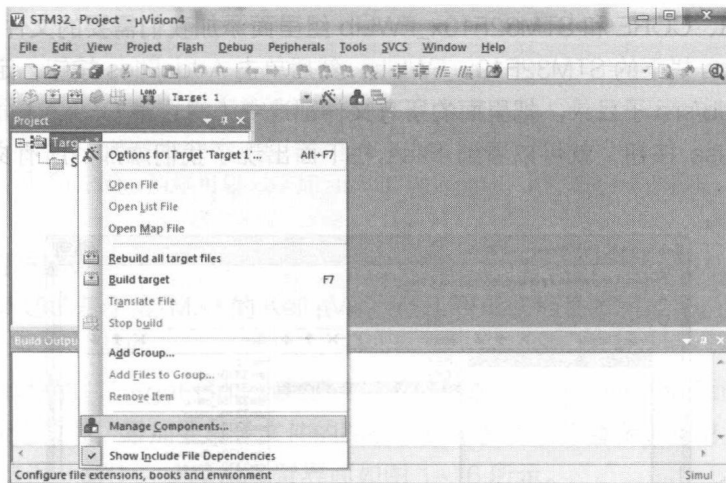


图1-11 调出Manage Components

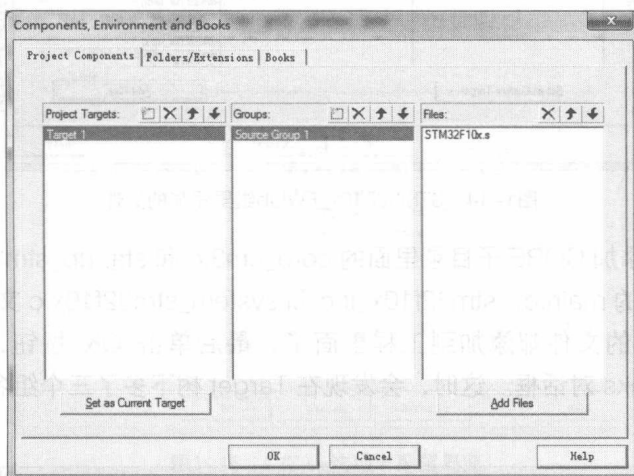


图1-12 Components Environment and Books对话框

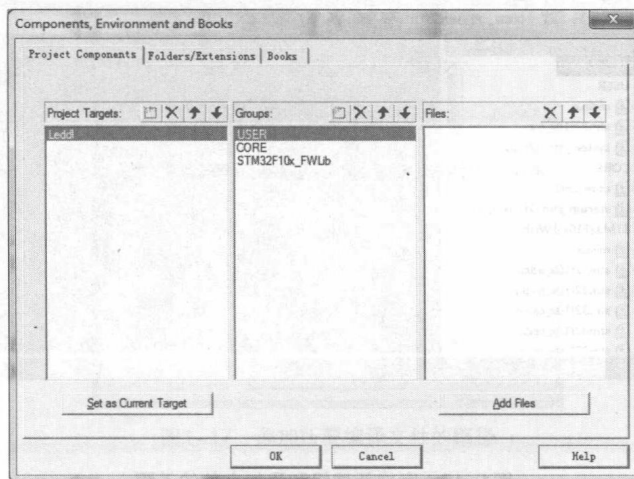


图1-13 新建的组

(5) 往 USER、CORE 和 STM32F10x_FWLib 组里面添加我们需要的文件。

先选中 Groups 栏下的 STM32F10x_FWLib，然后单击 Add Files 按钮，定位到工程目录的 STM32F10x_FWLib/src 子目录。把里面的所有文件都选中（组合键 Ctrl+A），然后单击 Add 按钮，最后单击 Close 按钮，就可以看到 Files 栏下面出现了我们添加的所有文件，如图 1-14 所示。

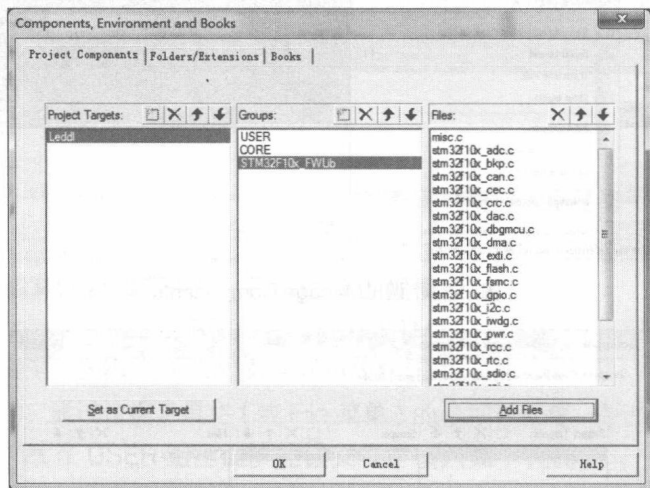


图1-14 STM32F10x_FWLib组里添加的文件

用同样的方法，添加 CORE 子目录里面的 core_cm3.c 和 startup_stm32f10x_ld.s 文件，添加 USER 子目录里面的 main.c、stm32f10x_it.c 和 system_stm32f10x.c 文件。

这样，需要添加的文件都添加到工程里面了，最后单击 OK 按钮，退出 Components Environment and Books 对话框。这时，会发现在 Target 树下多了三个组名和其中添加的文件，如图 1-15 所示。

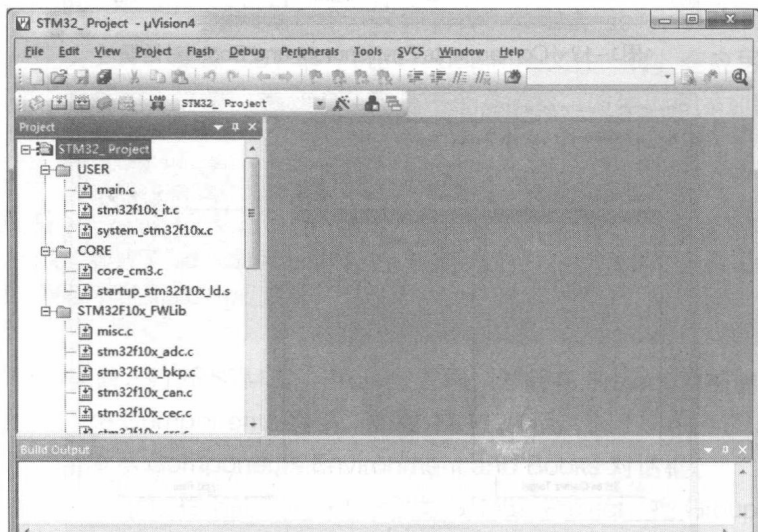



图1-15 完成新建组和添加文件的工程

**注意**

- 为方便后面使用，我们把所有外设的库文件都添加进工程了，不用每次增加外设时都要再添加相应的库文件，这样做的坏处就是当工程太大时，编译速度全变慢；
- 本任务只用了 GPIO，所以可以只添加 stm32f10x_gpio.c，其他的不用添加。

4. 工程配置与编译

到此为止，新建的基于 STM32 的 Keil μ Vision4 工程就已经基本完成了。接下来进行工程配置和编译。

(1) 单击工具栏的“Target Options...”按钮，弹出“Options for Target 'Ledd1'”对话框，选择 C/C++选项卡，添加要编译文件的路径。这个步骤非常重要，务必添加正确的路径，否则编译会出现错误。C/C++选项卡配置界面如图 1-16 所示。

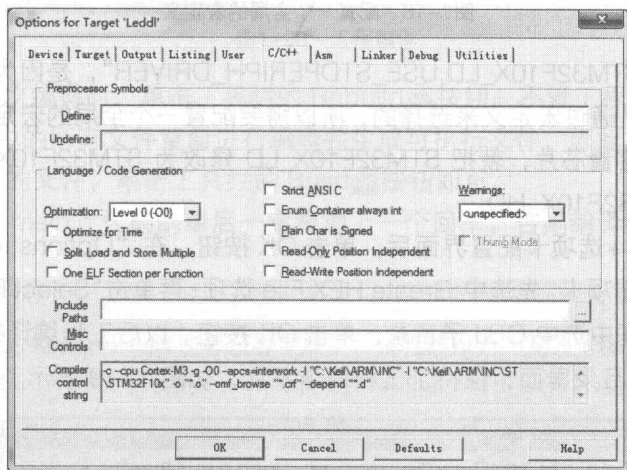


图 1-16 C/C++选项卡配置界面

(2) 单击 Include Paths 最右边的方块按钮，弹出添加路径的 Folder Setup 对话框，然后把 STM32F10x_FWLib\inc、CORE 和 USER 子目录都添加进去。此操作是为了设定编译器的头文件包含路径，在以后的任务中会经常用到，如图 1-17 所示。

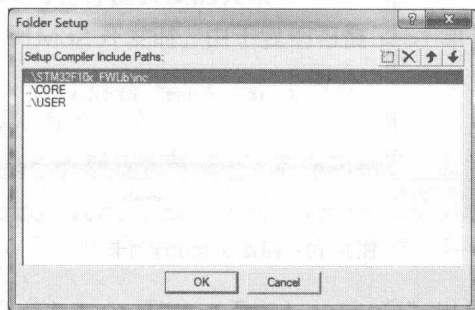


图 1-17 添加所要编译文件的路径

在这里，还需要在 C/C++选项卡配置界面中，填写“STM32F10X_LD,USE_STDPERIPH_DRIVER”到 Define 输入框里，如图 1-18 所示。

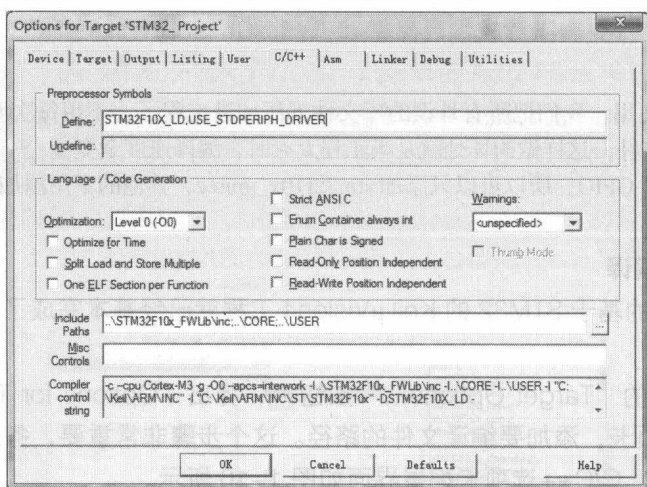


图1-18 配置一个全局的宏定义

之所以要填写“STM32F10X_LD,USE_STDPERIPH_DRIVER”，是因为 V3.5 版的库函数在配置和选择外设时，是通过宏定义来选择的，所以需要配置一个全局的宏定义变量，否则工程编译会出错。若使用中容量芯片，就把 STM32F10X_LD 修改为 STM32F10X_MD；若使用大容量芯片，就修改为 STM32F10X_HD。

(3) 设置完 C/C++ 选项卡配置界面后，单击 OK 按钮，在“Options for Target 'Leddl'”对话框中，选择 Output 选项卡。先选中 Create HEX File 选项；再单击“Select Folder for Objects...”按钮，在弹出的对话框中选中 OBJ 子目录，单击 OK 按钮。以后工程编译的 HEX 文件以及垃圾文件就会放到 OBJ 子目录里面，保持了工程简洁不乱，如图 1-19 所示。

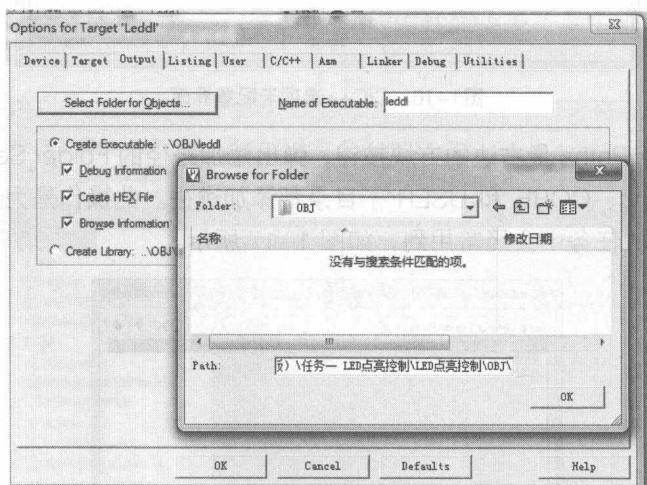



图1-19 配置Output选项卡

(4) 单击 OK 按钮，退出“Options for Target 'Target 1'”对话框，然后单击工具栏的 Rebuild  按钮，对工程进行编译。若编译发生错误，要进行分析检查，直到编译正确为止，如图 1-20 所示。