

唐思超 编著

嵌入式系统 软件设计实战

——基于IAR Embedded Workbench



北京航空航天大学出版社

嵌入式系统软件设计实战

——基于 IAR Embedded Workbench

唐思超 编著

北京航空航天大学出版社

内 容 简 介

全书分为13章。第1~4章为基础知识部分,讲述IAR Embedded Workbench开发环境的特点、功能、使用方法以及项目管理、参数配置等;第5~10章为本书的重点内容,结合处理器的相关结构讲述IAR Embedded Workbench开发环境的实用工作机制与应用,如启动代码与编译系统的关系、代码优化等;第11~13章是实例应用,详细介绍使用IAR Embedded Workbench开发环境进行开发的过程。

本书可作为软、硬件开发人员以及系统架构人员等相关工程技术人员的参考书,也可作为相关培训机构的教材或相关专业在校学生及教师的教学辅助教材,还可作为IAR Embedded Workbench开发环境的自学读物。

图书在版编目(CIP)数据

嵌入式系统软件设计实战:基于IAR Embedded Workbench/唐思超编著. --北京:北京航空航天大学出版社,2010.4

ISBN 978-7-5124-0045-0

I. ①嵌… II. ①唐… III. ①微处理器—系统设计
IV. ①TP332

中国版本图书馆CIP数据核字(2010)第047605号

© 2010,北京航空航天大学出版社,版权所有。

未经本书出版者书面许可,任何单位和个人不得以任何形式或手段复制本书内容。
侵权必究。

嵌入式系统软件设计实战 ——基于IAR Embedded Workbench

唐思超 编著

责任编辑 董六娟

*

北京航空航天大学出版社出版发行

北京市海淀区学院路37号(100191) 发行部电话:010-82317024 传真:010-82328026

www.buaapress.com.cn E-mail:casbook@gmail.com

北京市媛明印刷厂 印装 各地书店经销

*

开本:787×960 1/16 印张:27.5 字数:616千字

2010年4月第1版 2010年1月第1次印刷 印数:4 000册

ISBN 978-7-5124-0045-0 定价:49.00元

序

对于如今从事电子产品设计的工程师们来说,MCU(即单片机或微控制器)已经是十分常用的器件了。在 MCU 逐渐被各行各业的电子产品采用的同时,生产 MCU 的芯片厂商也从当初屈指可数的几家发展到现在的上百家甚至更多。各家厂商基于不同的 MCU 内核(根据数据总线的宽度,通常可划分为 8 位、16 位和 32 位内核)辅之以存储器、I/O 接口等片上资源,设计出具备不同特点的 MCU,使得工程师们可以按照各自应用的具体需求挑选成本和功能都适合的芯片进行开发。根据 2008 年的统计数据,8 位、16 位和 32 位 MCU 占据的市场份额分别为 33.9%、32.3%和 33.8%,近乎是三分天下的格局。

然而种类繁多的 MCU 也给工程师们带来了不少困扰,因为不同的 MCU 所要求的软件开发工具往往各不相同。一款 MCU 软件开发工具通常需要包含 C/C++ 编译器、汇编器、链接器、调试环境以及辅助的文本编辑器、工程管理等模块。用户获得软件工具的来源主要有独立的软件厂商、开源软件组织和一些芯片厂商本身。来源不同的软件工具在功能完整性和编译效率上参差不齐,其用户介面、控制命令、调试风格以及对标准 C 语言的功能扩展等方面也往往完全不同,从而使得用户为一种 MCU 开发的代码很难移植到其他 MCU 上去,而且在一款软件工具下所积累的使用经验在开发另一种 MCU 时也很难有所帮助。

IAR Systems 是目前嵌入式软件开发工具方面的领导企业之一,在该行业有着二十余年的服务经验,与各大著名的 MCU 芯片厂商保持着长期而紧密的合作关系,致力于为多达 20 余类不同架构的 MCU 提供风格统一、编译高效、使用方便的软件开发工具;其主力产品 IAR Embedded Workbench 系列中的各款软件可分别支持不同架构 MCU 上的代码开发、下载和调试,如 32 位 MCU 中的 ARM 系列、16 位 MCU 中的 TI MSP430 系列、8 位 MCU 中的 MCS-51 系列、Atmel AVR 系列等,最大程度地简化了代码的跨平台移植,方便了需要在不同的 MCU 之间进行转换的用户。此外,IAR Systems 的产品还包括基于状态机的软件建模和设计工具 visualSTATE;微内核、硬实时的嵌入式操作系统和中间件家族 PowerPac(含文件系统、TCP/IP 协议栈、USB 协议栈和 GUI)等,能够为 MCU 开发者提供 All-in-One 的解决方案。



目前, IAR Systems 的软件产品在全球拥有十万多授权的企业用户, 广泛分布在无线通信、工业控制、汽车电子、仪器仪表、消费电子等各大行业中; 而用于个人学习和评估的授权比这个数字还要多得多。然而国内一直缺乏一本能够结合几种有代表性的常用 MCU 系统地讲述基于 IAR Embedded Workbench 工具进行软件开发的书籍, 这也在某种程度上给国内的开发者造成了一些不便。

本书的作者在嵌入式系统开发方面有着多年的从业经历, 曾长期使用 IAR Embedded Workbench 软件和多种 MCU, 开发过形形色色的产品。本书集中介绍了各款 IAR Embedded Workbench 软件所共有的概念和使用方法, 也理清了对于不同的 MCU 在参数配置等方面的差异, 可以说是作者丰富经验的总结。作为一本软件书籍, 本书既包含“常规性”的内容(如功能介绍、操作方法等), 也包含了从作者对 MCU 软件的理解出发, 结合工具本身整理而成的一些很有价值和特点的内容(如代码和数据的定位、启动和初始化流程、存储器重映射、Flash 下载、C 与汇编混合编程、程序分析与优化等)。书中还给出了基于 Atmel AVR 单片机和 STMicroelectronics STR912 单片机 (ARM 内核) 进行具体系统设计的思路和实例, 对于理解前面的概念很有帮助。相信无论是刚刚接触 MCU 的初学者还是已经具备一些开发经验的读者, 都能够从本书中获益良多。

盛 磊

IAR Systems(上海)

前言

编写本书的目的

随着计算机技术的日益发展,嵌入式应用已经渗透到社会生活中的方方面面。从我们每天使用的手机、PMP 等消费电子产品到汽车电子、工业控制、国防科技等领域,无一例外都能找到嵌入式系统的身影。所谓嵌入式系统(Embedded System),就是可以嵌入到其他系统中的微处理器应用系统。嵌入式系统本身是一个可独立执行的系统,但更为重要的是它可以作为一个部件嵌入到其他应用系统中。

在组成上,嵌入式系统以微处理器及应用软件为核心组件。对于一个嵌入式系统,按其微处理器的类型可以分为以单片机为核心的嵌入式系统、以工业计算机板卡为核心的嵌入式系统、以 DSP 为核心的嵌入式数字信号处理系统、以 FPGA 及软 CPU 核为核心的嵌入式 SOPC 系统等;按其运行的软件类型可以分为没有操作系统的前后台应用系统(或称超循环系统)和基于嵌入式操作系统的应用系统。另外,嵌入式操作系统又可以分为面向控制、通信等领域的实时操作系统和面向消费电子的非实时操作系统。常用的实时操作系统有 IAR 公司的 PowerPac、Micrium 公司的 $\mu\text{C}/\text{OS}$ 、WindRiver 公司的 VxWorks、Express Logic 公司的 ThreadX 以及免费的 FreeRTOS 等。最常见的非实时操作系统有微软的 Windows XP Embedded、Windows CE 以及开源的嵌入式 Linux 等。

一般来说,大部分基于实时操作系统的应用系统需要将以源代码或库形式提供操作系统本身和应用程序部分一同编译;而基于非实时操作系统的应用系统则首先需要裁减、定制操作系统并生成新的系统映像文件,而后将其载入外部非易失性存储器,启动时先由相关的引导程序将操作系统映像加载至外部 SDRAM 或 DDR 等并完成相关初始化操作后,再启动操作系统。基于非实时操作系统的应用程序是单独编译的,这点和实时操作系统不同,这种方式更接近于传统 PC 机的应用程序开发过程。

当然,技术不是一成不变的。Raisonance 公司的开发环境 Ride 及其为 STM32 系列微控制器开发的嵌入式操作系统 CircleOS 就是一个介于上述两者之间的产物。CircleOS 不像 Linux 等非实时操作系统那样功能强大,但又不算实时操作系统,其本身更接近于实时系统。使用 Ride 开发环境可以开发 CircleOS,同时也可以单独开发可运行于 CircleOS 上的应用程序。CircleOS 具有应用程序管理功能,这些应用程序可以单独



下载至 STM32 的片内 Flash 中,并由 CircleOS 系统所管理。

嵌入式系统以微处理器及应用软件为核心组件,所以嵌入式开发人员需要深刻理解的两个方面就是处理器的架构和编译系统的机制。鉴于目前市面上缺乏此类书籍,本书即为解决此问题而编写。本书一方面介绍了 IAR Embedded Workbench 开发环境的实用机制及应用,另一方面简要介绍了处理器的相关架构,可以帮助读者在日后的开发工作中学会如何结合处理器的架构来更高效、更简捷地使用 IAR Embedded Workbench 进行嵌入式系统开发。同时,帮助读者增加对编译器系统相关机制的认知,不断提升开发境界,掌握分析解决问题的方法和能力,使得读者对于其他编译系统也能够从容自如地应用。例如,了解系统启动机制后,就可以在某些 ARM 架构上使用纯 C 语言来写启动代码,并且不需要编译系统的运行库支持。

本书的组织结构及特色

本书从工程实用的角度出发,分别以目前流行的 8 位单片机 AVR 系列、16 位单片机 MSP430 系列和 32 位单片机 ARM7、ARM9 以及 Cortex-M3 系列为例,结合其处理器架构特点介绍了相应 IAR Embedded Workbench 开发环境的机制及应用。本书的内容适用于前后台系统、基于实时操作系统的应用系统以及用于启动非实时操作系统的引导程序的开发。

全书分为三大部分,共 13 章。第 1~4 章为第一部分,讲述 IAR Embedded Workbench 开发环境的特点、功能、使用方法以及项目管理、参数配置等基础知识;第 5~10 章为第二部分,也是本书的重点内容,结合相应处理器的结构详细讲述了 IAR Embedded Workbench 开发环境的机制与应用,如编写启动代码、代码优化等;第三部分为第 11~13 章,讲述 IAR Embedded Workbench 开发环境的实例应用。

与市面的其他书籍相比,本书的最大特色在于它既不是单纯讲述某个方面的应用,也不是纯粹讲述某种软件的使用,而是将编译器、链接器、下载器和调试器的原理、机制与各种处理器的架构相结合,从实际应用出发,把理论和实践紧密联系起来,深入浅出地讲述了嵌入式系统的运作。本书可以帮助开发人员了解教科书中大多不曾提及的编译系统的实用机制、了解这些机制如何通过具体的处理器硬件来实现,从而熟练掌握 IAR Embedded Workbench 开发环境。

相关说明

本书涉及了许多相关基础知识和原理,受篇幅等限制,不能一一详述,读者可参阅相关资料。

本书在术语方面尽量使用中文,对于某些无法找到合适中文的术语则直接使用了英文。为了讲述的方便,书中某些地方将 IAR Embedded Workbench 开发环境简写为 IAR EW, IAR Embedded Workbench for ARM 简写为 IAR EWARM, IAR Embedded

Workbench for AVR 简写为 IAR EWAVR, IAR Embedded Workbench for MSP430 简写为 IAR EW430。

为了阐述方便,在不至于引起歧义的情况下,书中没有对单片机、微控制器、微处理器等术语进行严格区分。通常情况下单片机泛指微控制器,但严格说来,微控制器与微处理器是有差别的,它们代表了不同的理念。前者更侧重于一个完整的微型计算机系统;而后者则侧重于强大的处理能力,典型代表就是单芯片 CPU。近些年来,由于社会需求的发展和技术的进步,这两个概念已经发生交叉。某些微控制器内部具有非常强大的 CPU,而微处理器内部也集成了多种外设。因此,不断有新的名词出现,比如应用处理器、数字信号控制器等。另外,芯片的种类划分也与其面向的市场和厂商的定位有关,比如 TI 将其基于 ARM+DSP 核的应用处理器划分为 DSP 类,而 Atmel 则将相同的产品划分为微控制器。

读者对象

本书理论阐述与实用性相结合,既有针对性也有大众性,适合下列人员阅读:

- 相关专业的在校学生及教师;
- 想学习或初学嵌入式开发的人员;
- 硬件开发人员、软件开发人员以及系统架构人员;
- 想深入了解编译系统或从事编译系统开发的人员;
- 想学习 IAR Embedded Workbench 开发环境的人员。

本书的适用人群广泛,建议初学人员先通过其他方式获取一些基础知识,例如,处理器架构的基础知识,C 语言中函数指针、const 指针的用法及基础计算机操作等知识。

致 谢

本书在编写过程中得到了很多人的支持和热心关注。尤其感谢 IAR 公司中国代表处叶涛先生、盛磊先生的大力支持,感谢北京航空航天大学出版社的热心帮助,感谢家人和朋友的支持和鼓励。由于笔者年轻,水平有限以及编写书稿时间限制等原因,书中难免存在不妥、遗漏甚至错误,希望广大读者批评指正。有兴趣的读者可以发送电子邮件至 strongtang@gmail.com,与作者进一步交流;也可以发送电子邮件到:xdhydc5@sina.com,与本书策划编辑进行交流。

本书中部分章节的原始内容由 IAR 公司中国代表处提供,在此表示衷心感谢。

本书中的所有源代码以及工程文件,包括书中用于分析相关工作机制的分析案例文件和应用实例文件,均可以从北京航空航天大学出版社的网站下载,网址是:

唐思超
2010 年 1 月

目 录

第 1 章 IAR Embedded Workbench 基础知识	1
1.1 IAR Embedded Workbench 嵌入式集成开发环境简介	1
1.2 IAR Embedded Workbench 的菜单及工具栏	3
1.2.1 菜单栏	4
1.2.2 工具栏	13
1.2.3 状态栏	14
第 2 章 IAR Embedded Workbench 快速入门	15
2.1 项目的创建	15
2.1.1 建立一个项目文件目录	15
2.1.2 生成新工作区	15
2.1.3 生成新项目	16
2.1.4 给项目添加文件	17
2.1.5 配置项目选项	18
2.2 编译和链接应用程序	20
2.2.1 编译和链接	20
2.2.2 查看 MAP 文件	21
2.3 用 C-SPY 下载和调试应用程序	22
2.3.1 配置 Debugger 选项	22
2.3.2 下载应用程序	22
2.3.3 源代码级调试	24
2.3.4 查看变量	24
2.3.5 设置和监视断点	26
2.3.6 在反汇编窗口上进行调试	27
2.3.7 监视寄存器	27



2.3.8 查看存储器·····	27
2.3.9 观察 Terminal I/O ·····	28
2.3.10 执行和暂停程序 ·····	29
第3章 项目管理 ·····	30
3.1 项目组织模型·····	30
3.2 项目创建与管理·····	32
3.2.1 工作区及其内容的创建和管理·····	33
3.2.2 拖拽操作·····	35
3.2.3 源文件路径·····	36
3.3 项目文件导航·····	36
3.3.1 查看工作区·····	36
3.3.2 显示源代码浏览信息·····	37
3.4 使用库模块·····	39
第4章 IAR Embedded Workbench 项目参数配置 ·····	43
4.1 General Options——基本选项配置·····	43
4.1.1 Target 选项卡 ·····	43
4.1.2 Target 选项卡(适用于 IAR for AVR) ·····	46
4.1.3 Target 选项卡(适用于 IAR for MSP430) ·····	48
4.1.4 Output 选项卡 ·····	50
4.1.5 Library Configuration 选项卡 ·····	51
4.1.6 Library Options 选项卡 ·····	52
4.1.7 Heap Configuration 选项卡 ·····	53
4.1.8 Stack/Heap 选项卡 ·····	53
4.1.9 System 选项卡 ·····	54
4.1.10 MISRA C 选项卡 ·····	55
4.2 C/C++ 编译器配置 ·····	55
4.2.1 Language 选项卡 ·····	56
4.2.2 Code 选项卡(适用于 IAR for AVR) ·····	58
4.2.3 Code 选项卡(适用于 IAR for MSP430) ·····	59
4.2.4 Optimizations 选项卡 ·····	60
4.2.5 Output 选项卡 ·····	61
4.2.6 List 选项卡·····	63

4.2.7	Preprocessor 选项卡	63
4.2.8	Diagnostics 选项卡	65
4.2.9	MISRA C 选项卡	66
4.2.10	Extra Options 选项卡	67
4.3	汇编器配置	67
4.3.1	Language 选项卡	67
4.3.2	Output 选项卡	69
4.3.3	List 选项卡	69
4.3.4	Preprocessor 选项卡	70
4.3.5	Diagnostics 选项卡	72
4.3.6	Extra Options 选项卡	72
4.4	自定义创建配置	73
4.5	项目生成配置	74
4.6	链接器配置	75
4.6.1	Output 选项卡	75
4.6.2	Extra Output 选项卡	78
4.6.3	#define 选项卡	78
4.6.4	Diagnostics 选项卡	79
4.6.5	List 选项卡	81
4.6.6	Config 选项卡	82
4.6.7	Processing 选项卡	86
4.6.8	Extra Options 选项卡	90
4.7	库生成器配置	90
4.8	调试器配置	91
4.8.1	Setup 选项卡	91
4.8.2	Download 选项卡	93
4.8.3	Extra Options 选项卡	94
4.8.4	Plugins 选项卡	94
4.9	IAR J-Link 驱动配置	95
4.9.1	Setup 选项卡	96
4.9.2	Connection 选项卡	97
4.9.3	Breakpoints 选项卡	98



第 5 章 存储方式与段定位	103
5.1 数据存储方式	103
5.1.1 存储空间	103
5.1.2 栈与自动变量	104
5.1.3 堆中的动态存储分配	105
5.2 代码与数据的定位	106
5.2.1 段的定义	106
5.2.2 段的作用	106
5.2.3 段存储类型	106
5.2.4 段在存储器中的定位	108
5.2.5 数据段	110
5.2.6 代码段	114
5.2.7 C++ 动态初始化	115
5.2.8 变量与函数在存储器中的定位	115
第 6 章 IAR C-SPY 宏系统	119
6.1 C-SPY 宏系统	119
6.1.1 宏语言	120
6.1.2 宏函数	125
6.1.3 宏文件	134
6.2 使用 C-SPY 宏	136
6.2.1 使用设置宏函数和设置文件来注册、运行宏	136
6.2.2 使用 Macro Configuration 对话框注册宏文件	138
6.2.3 使用 Quick Watch 界面运行宏函数	138
6.2.4 将宏函数与断点相连以执行宏函数	139
6.3 使用 C-SPY 模拟器进行中断仿真	141
6.3.1 C-SPY 中断仿真系统	141
6.3.2 中断仿真系统的使用	143
6.4 中断仿真实例	147
6.4.1 添加中断句柄	147
6.4.2 设置仿真环境	148
6.4.3 运行仿真中断	151
6.4.4 使用系统宏定义中断和设置断点	152

第 7 章 IAR Embedded Workbench 的工作机制与应用	153
7.1 系统的初始化过程	154
7.2 微处理器的启动与重映射	156
7.2.1 映射的概念	156
7.2.2 存储器映射与存储器重映射	156
7.2.3 微控制的片内存储器	156
7.2.4 ARM 处理器的 Boot 技术	157
7.2.5 与映射和重映射相关的实例	159
7.3 重映射的意义与实现过程	163
7.3.1 软件断点与硬件断点	164
7.3.2 重映射的作用与实现举例	165
7.4 程序入口与启动代码	175
7.4.1 程序入口的概念	175
7.4.2 程序入口的实例分析	175
7.4.3 系统的启动代码	186
7.4.4 在 IAR 中设置程序的入口	188
7.5 ARM 处理器启动代码的深入研究	190
7.5.1 需要 IAR 运行库支持的纯 C 语言启动代码	190
7.5.2 不需要 IAR 运行库支持的纯 C 语言启动代码	197
7.5.3 纯 C 语言启动代码的适用情况	204
7.5.4 使用纯 C 语言启动代码的注意事项	205
7.6 全局变量运行时定位的实例分析	208
7.6.1 变量的简单分类	208
7.6.2 变量定位至 RAM 的时间	208
7.6.3 变量在只读存储器中的存储方式	209
7.6.4 全局变量的运行时定位分析	210
7.6.5 全局变量的运行时定位过程分析	213
7.7 在 RAM 中运行的函数	216
7.7.1 RAM 函数	216
7.7.2 RAM 函数的实现	216
7.8 RAM 调试与实现机制	227
7.8.1 MAC 文件的概念	227
7.8.2 RAM 调试的基础知识	227



7.8.3	RAM 调试的工作机制	228
7.9	Flash Loader 与 Flash 调试	237
7.9.1	Flash Loader 概述	237
7.9.2	可选的 Flash Loader C-SPY 宏文件	238
7.9.3	与 Flash Loader 框架程序的接口	238
7.9.4	Flash Loader 驱动程序实例	239
7.9.5	创建 Flash Loader 的过程举例	240
7.9.6	调试 Flash Loader	241
7.9.7	将应用程序下载至 Flash 中	242
7.9.8	Flash Debug 的流程及实例分析	243
7.10	应用程序的完整性校验	258
7.10.1	设置链接器产生 checksum	258
7.10.2	在用户代码中加入校验和计算函数	261
7.11	Flash Loader 的使用	263
7.11.1	设置 Flash Loader	264
7.11.2	Flash 装载机制	264
7.11.3	生成程序时需要考虑的事情	264
7.11.4	Flash Loader Overview 对话框	265
7.11.5	Flash Loader 配置对话框	266
7.12	使用 IAR EW 直接下载二进制文件到目标 Flash 存储器	267
7.13	将 MSP430 系列单片机的片内 Flash 拟作 EEPROM	273
7.13.1	MSP430 系列单片机的内部存储器组织	273
7.13.2	Flash 的擦除	274
7.13.3	演示程序分析	275
7.13.4	修改和使用 XCL 文件	279
第 8 章	IAR EWARM 版本迁移	281
8.1	版本迁移概述	281
8.1.1	EWARM 版本 4.xx 与 5.xx 的区别	281
8.1.2	迁移工作	281
8.2	链接器和链接器的配置	282
8.2.1	EWARM 4.xx 的链接器 XLINK 及其配置文件	282
8.2.2	XLINK 选项	282
8.2.3	XCL 文件举例	284

8.2.4	EWARM 5. xx 的链接器 ILINK 及其配置文件	286
8.2.5	ICF 格式概述	286
8.2.6	ICF 文件举例	290
8.2.7	图形化工具 ICF Editor 的使用	292
8.3	有关版本迁移的其他信息	292
第 9 章	C 与汇编的混合编程	294
9.1	AVR 单片机 C 语言与汇编语言的混合编程	294
9.1.1	在 C 语言函数和汇编语言函数间传递变量	295
9.1.2	C 代码调用汇编函数	296
9.1.3	汇编代码调用 C 函数	297
9.1.4	使用汇编语言编写中断程序	298
9.1.5	汇编代码访问全局变量	299
9.2	MSP430 单片机 C 语言与汇编语言的混合编程	300
9.2.1	调用内部函数	300
9.2.2	直接嵌入	300
9.2.3	调用汇编模块	300
9.2.4	新的函数调用协议	302
9.2.5	实例分析	303
第 10 章	程序分析与性能优化	308
10.1	应用程序分析	308
10.1.1	函数级剖析	308
10.1.2	代码覆盖	310
10.2	调整 IAR Embedded Workbench 以获取最佳性能	312
10.2.1	优化设置——代码容量与速度	312
10.2.2	存储模型选择	313
10.2.3	运行库设置	314
10.2.4	数据类型选择	315
10.2.5	目标处理器专有设置	315
10.3	为嵌入式应用编写高效率代码	315
10.3.1	合理利用编译系统	316
10.3.2	选择数据类型以及数据在存储器中的定位	319
10.3.3	编写高效代码	322



第 11 章 基于 CAN 协议的 Boot Loader	327
11.1 硬件电路设计	328
11.1.1 电源电路	328
11.1.2 CAN 收发器电路	328
11.1.3 单片机电路	330
11.2 软件设计概述	330
11.2.1 Boot Loader 运行环境	332
11.2.2 Boot Loader 实现	335
11.3 存储空间定义	337
11.3.1 Flash 存储空间	338
11.3.2 EEPROM 数据存储区	338
11.3.3 签名存储区	339
11.3.4 Boot Loader 信息存储区	339
11.3.5 Boot Loader 配置存储区	340
11.3.6 设备寄存器	343
11.4 CAN 协议和 ISP 命令	343
11.4.1 CAN 协议	343
11.4.2 CAN ISP 命令数据流协议	345
11.5 API 应用程序编程接口	351
11.5.1 API 的定义	351
11.5.2 使用 API	351
11.5.3 API 的使用限制	351
11.5.4 API 细节介绍	351
11.5.5 API 入口点	352
11.5.6 IAR 环境中的 API 调用示例	352
11.5.7 使用其他 C 编译器的 API 调用	352
11.6 使用 Flip 软件与 CAN 结点通信	355
第 12 章 基于 AVR 单片机的数码录放模块	356
12.1 系统工作原理	356
12.1.1 语音采样的理论依据	356
12.1.2 数据存储和读取	357
12.1.3 PWM 声音回放	358

12.2 硬件电路设计.....	359
12.2.1 微控制器和存储器电路.....	360
12.2.2 麦克风和扬声器电路.....	361
12.3 软件设计.....	362
12.3.1 初始化设置.....	362
12.3.2 主循环.....	362
12.3.3 擦 除.....	364
12.3.4 录 音.....	366
12.3.5 存 储.....	368
12.3.6 回 放.....	370
12.4 调试和优化.....	372
第 13 章 基于 STR912 的 USB 声卡	374
13.1 硬件设计.....	374
13.1.1 处理器概述.....	374
13.1.2 电源电路.....	374
13.1.3 JTAG 及复位电路.....	376
13.1.4 液晶显示电路.....	376
13.1.5 USB 接口电路	377
13.1.6 微控制器电路.....	378
13.1.7 音频接口电路.....	378
13.2 软件设计.....	380
13.2.1 启动程序.....	380
13.2.2 驱动程序.....	390
13.2.3 应用程序.....	396
13.3 调试和使用.....	402
13.3.1 硬件电路的调试.....	402
13.3.2 软件部分的调试.....	402
附录 A 为 MSP430 系列单片机编写高质量代码	410
附录 B 为 AVR 系列单片机编写高质量代码	412
附录 C 编译指南	414
附录 D 选择合适的微控制器	416
参考文献.....	422