



教育部高等学校电子信息类专业教学指导委员会规划教材
高等学校电子信息类专业系列教材

嵌入式与工业控制技术

P rinciples and Applications of Embedded Controller
Based on ARM Cortex-M3 (STM 32)

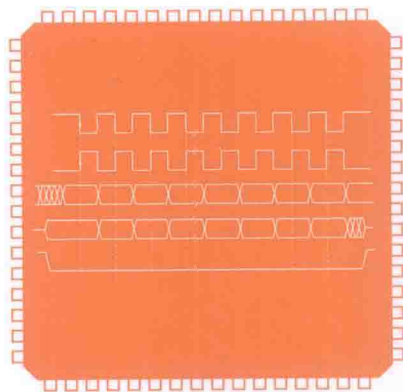
嵌入式微处理器 原理与应用

—— 基于ARM Cortex-M3微控制器

(STM32系列)

严海蓉 薛涛 曹群生 时昕 编著

Yan Hairong Xue Tao Cao Qunsheng Shi Xin



清华大学出版社



教育部高等学校电子信息类专业教学指导委员会
高等学校电子信息类专业系列教材

Principles and Applications of Embedded Controller
Based on ARM Cortex-M3 (STM 32)

嵌入式微处理器 原理与应用

—— 基于ARM Cortex-M3微控制器

(STM32系列)

严海蓉 薛涛 曹群生 时昕 编著

Yan Hairong Xue Tao Cao Qunsheng Shi Xin

TP332
343

清华大学出版社

内 容 简 介

本书系统论述了 ARM Cortex-M3 嵌入式微处理器的原理、架构、编程与系统开发方法,并以 STM32 微处理器为样本,给出了丰富的设计示例与综合实例。本书共分为 8 章,分别介绍了嵌入式微处理器的开发方法、Cortex-M3 的体系结构、Cortex-M3 的指令集、Cortex-M3 的功能特性、C 语言与汇编语言混合编程、Cortex-M3 的外设、Cortex-M3 的驱动、Cortex-M3 应用综合实例等内容。

本书适合作为高等学校电子信息类专业与计算机类专业本科生及研究生的“嵌入式系统原理及应用”课程的教材,也适合作为相关领域工程技术人员的参考用书。

本书封面贴有清华大学出版社防伪标签,无标签者不得销售。

版权所有,侵权必究。侵权举报电话:010-62782989 13701121933

图书在版编目(CIP)数据

嵌入式微处理器原理与应用:基于 ARM Cortex-M3 微控制器(STM32 系列)/严海蓉等编著. —北京:清华大学出版社,2014

高等学校电子信息类专业系列教材

ISBN 978-7-302-38378-9

I. ①嵌… II. ①严… III. ①微处理器—高等学校—教材 IV. ①TP332

中国版本图书馆 CIP 数据核字(2014)第 250966 号

责任编辑:盛东亮

封面设计:李召霞

责任校对:梁毅

责任印制:刘海龙

出版发行:清华大学出版社

网 址: <http://www.tup.com.cn>, <http://www.wqbook.com>

地 址:北京清华大学学研大厦 A 座 邮 编:100084

社总机:010-62770175 邮 购:010-62786544

投稿与读者服务:010-62776969, c-service@tup.tsinghua.edu.cn

质量反馈:010-62772015, zhiliang@tup.tsinghua.edu.cn

课 件 下 载: <http://www.tup.com.cn>, 010-62795954

印 装 者:北京密云胶印厂

经 销:全国新华书店

开 本:185mm×260mm 印 张:16.25 字 数:408 千字

版 次:2014 年 12 月第 1 版 印 次:2014 年 12 月第 1 次印刷

印 数:1~3000

定 价:34.50 元

高等学校电子信息类专业系列教材

— 顾问委员会

谈振辉	北京交通大学	(教指委高级顾问)	郁道银	天津大学	(教指委高级顾问)
廖延彪	清华大学	(特约高级顾问)	胡广书	清华大学	(特约高级顾问)
华成英	清华大学	(国家级教学名师)	于洪珍	中国矿业大学	(国家级教学名师)
彭启琮	电子科技大学	(国家级教学名师)	孙肖子	西安电子科技大学	(国家级教学名师)
邹逢兴	国防科技大学	(国家级教学名师)	严国萍	华中科技大学	(国家级教学名师)

— 编审委员会

主 任	吕志伟	哈尔滨工业大学			
副主任	刘 旭	浙江大学	王志军	北京大学	
	隆克平	北京科技大学	葛宝臻	天津大学	
	秦石乔	国防科学技术大学	何伟明	哈尔滨工业大学	
	刘向东	浙江大学			
委 员	王志华	清华大学	宋 梅	北京邮电大学	
	韩 焱	中北大学	张雪英	太原理工大学	
	殷福亮	大连理工大学	赵晓晖	吉林大学	
	张朝柱	哈尔滨工程大学	刘兴钊	上海交通大学	
	洪 伟	东南大学	陈鹤鸣	南京邮电大学	
	杨明武	合肥工业大学	袁东风	山东大学	
	王忠勇	郑州大学	程文青	华中科技大学	
	曾 云	湖南大学	李思敏	桂林电子科技大学	
	陈前斌	重庆邮电大学	张怀武	电子科技大学	
	谢 泉	贵州大学	卞树檀	第二炮兵工程大学	
	吴 瑛	解放军信息工程大学	刘纯亮	西安交通大学	
	金伟其	北京理工大学	毕卫红	燕山大学	
	胡秀珍	内蒙古工业大学	付跃刚	长春理工大学	
	贾宏志	上海理工大学	顾济华	苏州大学	
	李振华	南京理工大学	韩正甫	中国科学技术大学	
	李 晖	福建师范大学	何兴道	南昌航空大学	
	何平安	武汉大学	张新亮	华中科技大学	
	郭永彩	重庆大学	曹益平	四川大学	
	刘缠宇	西安工业大学	李儒新	中科院上海光学精密机械研究所	
	赵尚弘	空军工程大学	董友梅	京东方科技集团	
	蒋晓瑜	装甲兵工程学院	蔡 毅	中国兵器科学研究院	
	仲顺安	北京理工大学	冯其波	北京交通大学	
	黄翊东	清华大学	张有光	北京航空航天大学	
	李勇朝	西安电子科技大学	江 毅	北京理工大学	
	章毓晋	清华大学	谢凯年	赛灵思公司	
	刘铁根	天津大学	张伟刚	南开大学	
	王艳芬	中国矿业大学	宋 峰	南开大学	
	苑立波	哈尔滨工程大学	靳 伟	香港理工大学	
丛书责任编辑	盛东亮	清华大学出版社			

序

FOREWORD

我国电子信息产业销售收入总规模在 2013 年已经突破 12 万亿元,行业收入占工业总体比重已经超过 9%。电子信息产业在工业经济中的支撑作用凸显,更加促进了信息化和工业化的高层次深度融合。随着移动互联网、云计算、物联网、大数据和石墨烯等新兴产业的爆发式增长,电子信息产业的发展呈现了新的特点,电子信息产业的人才培养面临着新的挑战。

(1) 随着控制、通信、人机交互和网络互联等新兴电子信息技术不断发展,传统工业设备融合了大量最新的电子信息技术,它们一起构成了庞大而复杂的系统,派生出大量新兴的电子信息技术应用需求。这些“系统级”的应用需求,迫切要求具有系统级设计能力的电子信息技术人才。

(2) 电子信息系统设备的功能越来越复杂,系统的集成度越来越高。因此,要求未来的设计者应该具备更扎实的理论基础知识和更宽广的专业视野。未来电子信息系统的设计越来越要求软件和硬件的协同规划、协同设计和协同调试。

(3) 新兴电子信息技术的发展依赖于半导体产业的不断推动,半导体厂商为设计者提供了越来越丰富的生态资源,系统集成厂商的全方位配合又加速了这种生态资源的进一步完善。半导体厂商和系统集成厂商所建立的这种生态系统,为未来的设计者提供了更加便捷却又必须依赖的设计资源。

教育部 2012 年颁布了新版《高等学校本科专业目录》,将电子信息类专业进行了整合,为各高校建立系统化的人才培养体系,培养具有扎实理论基础和宽广专业技能的、兼顾“基础”和“系统”的高层次电子信息人才给出了指引。

传统的电子信息学科专业课程体系呈现“自底向上”的特点,这种课程体系偏重对底层元器件的分析与设计,较少涉及系统级的集成与设计。近年来,国内很多高校对电子信息类专业课程体系进行了大力度的改革,这些改革顺应时代潮流,从系统集成的角度,更加科学合理地构建了课程体系。

为了进一步提高普通高校电子信息类专业教育与教学质量,贯彻落实《国家中长期教育改革和发展规划纲要(2010—2020 年)》和《教育部关于全面提高高等教育质量若干意见》(教高【2012】4 号)的精神,教育部高等学校电子信息类专业教学指导委员会开展了“高等学校电子信息类专业课程体系”的立项研究工作,并于 2014 年 5 月启动了《高等学校电子信息类专业系列教材》(教育部高等学校电子信息类专业教学指导委员会规划教材)的建设工作。其目的是为推进高等教育内涵式发展,提高教学水平,满足高等学校对电子信息类专业人才培养、教学改革与课程改革的需要。

本系列教材定位于高等学校电子信息类专业的专业课程,适用于电子信息类的电子信

息工程、电子科学与技术、通信工程、微电子科学与工程、光电信息科学与工程、信息工程及其相近专业。经过编审委员会与众多高校多次沟通,初步拟定分批次(2014—2017年)建设约100门课程教材。本系列教材将力求在保证基础的前提下,突出技术的先进性和科学的前沿性,体现创新教学和工程实践教学;将重视系统集成思想在教学中的体现,鼓励推陈出新,采用“自顶向下”的方法编写教材;将注重反映优秀的教学改革成果,推广优秀的教学经验与理念。

为了保证本系列教材的科学性、系统性及编写质量,本系列教材设立顾问委员会及编审委员会。顾问委员会由教指委高级顾问、特约高级顾问和国家级教学名师担任,编审委员会由教育部高等学校电子信息类专业教学指导委员会委员和一线教学名师组成。同时,清华大学出版社为本系列教材配置优秀的编辑团队,力求高水准出版。本系列教材的建设,不仅有众多高校教师参与,也有大量知名的电子信息类企业支持。在此,谨向参与本系列教材策划、组织、编写与出版的广大教师、企业代表及出版人员致以诚挚的感谢,并殷切希望本系列教材在我国高等学校电子信息类专业人才培养与课程体系建设中发挥切实的作用。

吕志伟 教授

前言

FOREWORD

虽然嵌入式系统与计算机系统的起源一样久远,但由于嵌入式系统的微处理器受到体积、功耗、造价等因素的制约,一直没有一统江湖的某一个品牌出现。

然而随着近年来智能手机、手持设备的快速发展,ARM 微处理器逐渐占据了嵌入式系统的大部分阵地。而且随着物联网概念的出现,嵌入式系统包含了功能从复杂到简单的各种需求。为了适应这种趋势,ARM 也将其微处理器构架重新打造,产生了 Cortex 系列,包括了 A、R 和 M 三种不同性能,面向不同应用需求的微处理器系列。

从教授 ARM7 开始,为了满足教课的需要,就一直想自己编写一本专门的嵌入式微处理器原理的教材。随着 ARM 平台进一步朝着 Cortex 系列发展,更加萌发了将最新的微处理器系列介绍给学生的念头。本书选取 Cortex M3 系列的 STM32 作为主要的介绍对象,主要是考虑到嵌入式系统主要的一些基本功能就是做控制,如控制小车行进等。同时 M3 也具有一定的扩展性,可以放进去一个不大不小的嵌入式操作系统。最后,STM32 也是目前得到较广泛应用的一个微处理器芯片,而且教材并不丰富。

本书重点是讲述嵌入式微处理器原理,尤其是怎样对指令集进行编程,并且以微处理器为核心来设计连接硬件。

本书内容可分四大部分。

第 1~2 章为第一部分,主要是微处理器的概述,第 1 章介绍基础知识,第 2 章介绍 ARM 核体系结构。

第 3~5 章为第二部分,属于微处理器编程。第 3 章介绍 ARM 指令集,第 4 章介绍 Cortex-M3 特性,第 5 章介绍 C 语言与汇编语言混编。

第 6~8 章为第三部分,属于微处理器外设连接和驱动编写。以 STM32 为例,讲解了从硬件连接到软件驱动编写的设计过程。实验小车平台上的核心就是 STM32。

第 9 章是一个综合案例。在实验小车上实现了用 ZigBee 进行连接,接收计算机给的遥控指令,让小车前后左右行进。

本书内容新颖,立足点高。同时力求重点突出,层次清晰,语言通俗易懂,内容覆盖面广。学习本书需要有一定的 C 语言阅读能力和硬件的入门知识。本书可作为高等院校本科、研究生各相关专业(如嵌入式系统、物联网、计算机、电子信息、通信)的程序设计教材,也适合编程开发人员培训、广大嵌入式系统技术爱好者自学使用。

根据我们的教学体会,本书的教学可以安排为 32~48 学时。如果安排的学时数较少,可以根据学生的水平适当删减第二部分的部分内容。

本书所提供的实验实例全部在目标硬件上调试通过。

尽管在写作过程中投入了大量的时间和精力,但由于水平有限,错误和不足之处仍在所

难免,敬请读者批评指正(任何建议可以发至邮箱 yanhairong@bjut.edu.cn)。我们会在适当时间对本书进行修订和补充。

本书第6~7章由薛涛编写和实验验证,第8章由蒋思瑛提供素材。实验平台小车由北京芯博华科技有限公司提供。还要感谢北京工业大学嵌入式系统与软件系的全体师生,本书的最终出版得到了许多老师和同学的帮助。还要感谢我的家人对我的支持。清华大学出版社为本书的编写和出版付出了辛勤劳动。另外,还要感谢刘宏刚的帮助。在本书完成之际,一并向他们表示诚挚的感谢。

严海蓉

2014年12月于北京工业大学

目录

CONTENTS

第 1 章 绪论	1
1.1 微处理器定义	1
1.2 ARM 发展历程	3
1.3 ARM 体系结构与特点	5
1.4 处理器选型	9
1.4.1 嵌入式微处理器选型的考虑因素	9
1.4.2 嵌入式微处理器选型示例	10
第 2 章 ARM 核体系结构	11
2.1 寄存器	15
2.2 ALU	21
2.3 存储部件	21
2.4 中断控制	26
2.5 总线	27
2.6 外围接口 I/O	28
2.7 流水线	28
2.8 ARM 协处理器接口	29
第 3 章 ARM 指令集	30
3.1 指令简介	31
3.2 ARM 寻址方式	34
3.2.1 数据处理指令的操作数的寻址方式	34
3.2.2 字及无符号字节的 Load/Store 指令的寻址方式	35
3.2.3 杂类 Load/Store 指令的寻址方式	36
3.2.4 批量 Load/Store 指令的寻址方式	37
3.2.5 协处理器 Load/Store 指令的寻址方式	39
3.3 Cortex 指令集	39
3.3.1 数据传送类指令	40
3.3.2 数据处理指令	44
3.3.3 其他计算类指令	48
3.3.4 饱和运算	48
3.3.5 无条件跳转指令	49
3.3.6 标志位与条件转移指令	49
3.3.7 IF-THEN 指令块	52
3.3.8 Barrier 隔离指令	53

3.3.9	其他一些有用的指令	53
3.3.10	对内存的互斥访问	54
3.4	伪指令	55
3.5	内嵌汇编	57
第4章	Cortex-M3 特性	58
4.1	特殊功能寄存器	58
4.2	中断建立全过程的演示	66
4.3	复位序列	67
4.4	中断咬尾	67
4.5	晚到异常	68
4.6	位带操作	69
4.7	互斥访问	74
第5章	C语言与汇编语言混编	76
5.1	ATPCS与AAPCS	76
5.2	嵌入式C编写与编译	80
5.3	C语言与汇编语言混编规范	88
5.3.1	内嵌汇编	88
5.3.2	汇编程序中访问C全局变量	92
5.3.3	C语言与汇编语言的相互调用	92
5.4	C语言与汇编语言混编实践	94
5.4.1	Keil使用和STM32固件库	95
5.4.2	建立自己的第一个Keil程序	98
第6章	STM32输入/输出	101
6.1	pin配置	101
6.1.1	单独的位设置或位清除	106
6.1.2	外部中断/唤醒线	106
6.1.3	复用功能	106
6.1.4	软件重新映射I/O复用功能	106
6.1.5	GPIO锁定机制	106
6.1.6	输入配置	107
6.1.7	输出配置	107
6.1.8	复用功能配置	107
6.1.9	模拟输入配置	107
6.2	输入/输出基本概念(寄存器、输入/输出类型)	107
6.2.1	基本概念	107
6.2.2	寄存器详解	108
6.2.3	复用I/O配置寄存器	113
6.2.4	通用I/O和AFIO使用的配置步骤	118
6.3	通用I/O锁定机制	122
6.4	系统时钟	123
6.4.1	时钟配置	123
6.4.2	时钟输出的使能控制	125

6.5	输入/输出常用固件库函数	127
6.5.1	GPIO_DeInit 函数	127
6.5.2	GPIO_AFIODeInit 函数	127
6.5.3	GPIO_Init 函数	128
6.5.4	GPIO_StructInit 函数	129
6.5.5	GPIO_ReadInputDataBit 函数	130
6.5.6	GPIO_ReadInputData 函数	130
6.5.7	GPIO_ReadOutputDataBit 函数	131
6.5.8	GPIO_ReadOutputData 函数	131
6.5.9	GPIO_SetBits 函数	131
6.5.10	GPIO_ResetBits 函数	132
6.5.11	GPIO_WriteBit 函数	133
6.5.12	GPIO_Write 函数	134
6.5.13	GPIO_PinLockConfig 函数	134
6.5.14	GPIO_EventOutputConfig 函数	135
6.5.15	GPIO_EventOutputCmd 函数	137
6.5.16	GPIO_PinRemapConfig 函数	137
6.5.17	GPIO_EXTILineConfig 函数	138
6.6	GPIO 控制 LED 灯	139
6.6.1	硬件设计	139
6.6.2	软件设计	141
6.7	GPIO 控制蜂鸣器	142
6.7.1	硬件设计	142
6.7.2	软件设计	144
6.8	跑马灯实验	145
6.8.1	硬件设计	145
6.8.2	软件设计	146
6.9	LCD1602 驱动	149
6.9.1	硬件设计	150
6.9.2	软件设计	159
第 7 章	串行通信模块与中断程序	160
7.1	接口与通信标准	160
7.1.1	I2C 接口	160
7.1.2	通用同步/异步收发器(USART)	163
7.1.3	串行外设接口(SPI)	167
7.1.4	控制器区域网络(CAN)	174
7.1.5	通用串行总线(USB)	182
7.2	串口 USART 实例	190
7.3	扫描键盘	195
7.4	继电器	203
7.5	脉宽调制	204
7.6	步进电动机	209

第 8 章 STM32 带操作系统编程	213
8.1 RAM,Flash 启动	214
8.2 小型操作系统 STM32 移植	216
8.2.1 μ COS-II 内核简介	217
8.2.2 开始移植	217
8.2.3 程序开发模式讨论	221
第 9 章 综合案例	223
9.1 硬件连接方式	226
9.2 驱动软件编写	227
9.2.1 串口通信	227
9.2.2 数据采集	228
9.2.3 小车的方向控制	229
9.2.4 小车的速度控制	232
9.3 Z-Stack 软件框架	233
9.3.1 任务调度	233
9.3.2 ZigBee 无线传输系统开发	234
9.4 计算机端程序开发	236
附录 A STM32F10x.h 中的定义	239
附录 B LCD1602 程序	241

就像 PC 有一颗“奔腾的芯”一样，微处理器是嵌入式电子设备的“心”。这个比喻虽然说出了心的重要性，但是并不是非常恰当。无论 Intel 还是 ARM，它们都该是大脑，思考着逻辑，里面运行着程序代码。

如果拿嵌入式电子设备比做人体，微处理器就像是电子设备的大脑，指挥设计出电子设备的所有行动。

微处理器一般是由计算单元+存储单元+总线+外部接口构成。另外，晶振和电源管理部分当然是不可少的。随着集成度越来越高，有更多的东西可以放到微处理器芯片中。

从动态的角度看，晶振是微处理器工作的核心，所传出的像脉搏一样的时钟周期。

总线像神经一样连接起各个部分，并传送数据和指令。指令和数据在本质上是没多大区别的，之所以这样划分是为了能够说得更清楚。一般而言，指令能够让处理器产生动作；数据只是指令的后果，或者指令执行的一些资源。

存储单元则如大脑的记忆体，它们存放着指令或者数据。不同的是，大脑的记忆体多记忆的是过去发生过的事情，微处理器的数据存储单元里放的是某个运算的结果，而指令存储单元里放的是未来的一些安排好的动作序列，更像我们安排的计划表。

计算单元就像大脑的科学计算区域，负责一些有规则的数学的运算。这种运算是由指令安排的，并且计算操作数和结果都是存放在存储单元里的。

外部接口像传到四肢的神经接口，把微处理器的指令传到四肢，让四肢做一些工作，或者从四肢传回来一些感觉信息，让微处理器判断要做什么。

微处理器与人脑的主要不同在于微处理器里的所有指令都是事先安排好的，而人脑是可以自己学习、自己安排动作的。

电源就像我们的血液带着丰富的养分，在某些紧急情况下，为了能够更长久地存活，我们会调解身体，让血液耗氧量下降。微处理器也一样，有时需要电源管理，在没有重要工作时，可以进入休眠状态。

1.1 微处理器定义

微处理器(micro processor, 缩写为 μP)是可编程化的特殊集成电路；是一种处理器，其所有组件小型化至一块或数块集成电路内；是一种集成电路，可在其一端或多端接受编码

指令,执行此指令并输出描述其状态的信号。有时又称半导体中央处理机(CPU),是微型计算机的一个主要部件。微处理器的组件常安装在一个单片上或同一组件内,但有时分布在一些不同芯片上。在具有固定指令集的微型计算机中,微处理器由算术逻辑单元和控制逻辑单元组成。在具有微程序控制指令集的微型计算机中,它包含另外的控制存储单元。用作处理通用数据时,叫作中央处理器,这也是最为人所知的应用(如 Intel Pentium CPU);专用作图像数据处理的,叫作图形处理器(graphics processing unit);用于音频数据处理的,叫作音频处理单元(audio processing unit)等。

总之,从物理性来说,它就是一块集成了数量庞大的微型晶体管与其他电子组件的半导体集成电路芯片。

之所以会称为微处理器,并不只是因为它比迷你计算机所用的处理器还要小。最主要的原因,还是因为当初各大芯片厂的工艺,已经进入了 $1\mu\text{m}$ 的阶段,用 $1\mu\text{m}$ 的工艺,所产制出来的处理器芯片,厂商就会在产品名称上用“微”字,强调它们是高科技。就如同现在的许多商业广告一样,很喜欢用“纳米”字眼。

早在微处理器问世之前,电子计算机的中央处理单元就经历了从真空管到晶体管以及再后来的离散式 TTL 集成电路等几个重要阶段。甚至在电子计算机以前,还出现过以齿轮、轮轴和杠杆为基础的机械结构计算机。文艺复兴时期的著名画家兼科学家列奥纳多·达·芬奇就曾做过类似的设计,但那个时代落后的制造技术根本没有能力将这个设计付诸实现。微处理器的发明使得复杂的电路群得以制成单一的电子组件。

从 20 世纪 70 年代早期开始,微处理器性能的提升就基本上遵循着 IT 界著名的摩尔定律。这意味着在过去的 30 多年里每 18 个月,CPU 的计算能力就会翻番。大到巨型机,小到笔记本电脑,持续高速发展的微处理器取代了诸多其他计算形式而成为各个类别各个领域所有计算机系统的计算动力之源。

目前常常听到的微处理器是微处理机的一种变体,它包括了 CPU、一些内存以及 I/O 接口,所有都集成在一块集成电路上。微处理器的专利号为 4074351,授予了德州仪器的 Gary Boone 和 Michael J. Cochran。当时他们是以微计算机的名称申请专利的。这算不算“微处理器”尚有争议。

根据麻省理工学院出版的《现代计算史》第 200 页到 221 页,英特尔同圣安东尼奥的一家叫作计算机终端的公司(后改名为数点公司)签署了一份合同,合作设计一块用于终端的芯片。数年后决定不用这块芯片了,而采用 TTL 逻辑电路设计运行相同的指令集,英特尔随即将该芯片命名为 8008,并于 1972 年 4 月上市销售。这是世界上第一块 8 位微处理器,也是后来《无线电电子》杂志卖的著名的马克-8 计算机的主要部件。8008 及其后继产品 8080 开创了微处理器的市场。

微处理器已经无处不在,无论录像机、智能洗衣机、移动电话等家电产品,还是汽车引擎控制,以及数控机床、导弹精确制导等都要嵌入各类不同的微处理器。微处理器不仅是微型计算机的核心部件,也是各种数字化智能设备的关键部件。国际上的超高速巨型计算机、大型计算机等高端计算系统也都采用大量的通用高性能微处理器建造。

1.2 ARM 发展历程

ARM 在 1990 年成立,最初的名字是 Advanced RISC Machines Ltd.,当时它是三家公司的合资——苹果电脑公司、Acorn 电脑公司以及 VLSI 技术(公司)。在 1991 年,ARM 推出了 ARM6 处理器家族,VLSI 则是第一个制造 ARM 芯片的公司。后来,TI、NEC、Sharp、ST 等陆续都获取了 ARM 授权,使得 ARM 处理器应用在手机、硬盘控制器、PDA、家庭娱乐系统以及其他消费电子中。

不像很多其他的半导体公司,ARM 从不制造和销售具体的处理器芯片。取而代之的,ARM 把处理器的设计授权给相关的商务合作伙伴,让他们去根据自己的强项设计具体的芯片。基于 ARM 低成本和高效的处理器设计方案,得到授权的厂商生产了多种多样的处理器、微处理器以及片上系统(SoC)。这种商业模式就是所谓的“知识产权授权”。

除了设计处理器,ARM 也设计系统级 IP 和软件 IP。ARM 开发了许多配套的基础开发工具、硬件以及软件产品。使用这些工具,合作伙伴可以更加舒心地开发他们自己的产品。

ARM 十几年如一日地开发新的处理器内核和系统功能块。功能的不断进化,处理水平的持续提高,年深日久造就了一系列的 ARM 架构。常说 ARM 架构 7,这里的数字表示架构。

以前,ARM 使用一种基于数字的命名法。在早期(20 世纪 90 年代),为了进一步明细该处理器支持的特性,数字后面添加字母后缀。就拿 ARM7TDMI 来说,T 代表 Thumb 指令集,D 表示支持 JTAG 调试(debugging),M 意指快速乘法器,I 则对应一个嵌入式 ICE 模块。后来,这 4 项基本功能成了任何新产品的标配,于是就不再出现这 4 个后缀——相当于默许了。但是新的后缀不断加入,包括定义存储器接口的,定义高速缓存的,以及定义“紧耦合存储器(TCM)”的,这套命名法也一直在使用。

要说明的是,架构版本号 and 名字中的数字并不是一码事。例如,ARM7TDMI 是基于 ARMv4T 架构的(T 表示支持“Thumb 指令”),ARM7TDMI 并不是一款 ARMv7 的产品,而是 v4T 架构的产品。

后来出的 ARM11 是基于 ARMv6 架构建成的。基于 ARMv6 架构的处理器包括 ARM1136J(F)-S、ARM1156T2(F)-S 及 ARM1176JZ(F)-S。ARMv6 是 ARM 进化史上的一个重要里程碑。从那时起,许多突破性的新技术被引进,存储器系统加入了很多崭新的特性,单指令流多数据流(SIMD)指令也是从 v6 开始首次引入的。而最前卫的新技术,就是经过优化的 Thumb-2 指令集,它专门针对低成本的微处理器及汽车组件市场。

ARMv6 的设计中还有另一个重大的决定:虽然这个架构要能上能下,从最低端的 MCU 到最高端的“应用处理器”都包含,但仍需定位准确,使处理器的架构能胜任每个应用领域。结果就是要使 ARMv6 能够灵活地配置和剪裁。对于成本敏感的市场,要设计一个低门数的架构,让其有极强的确定性;另外,在高端市场上,不管要有丰富功能的还是要有高性能的,都要有拿得出手的好东西。

到了架构 7 时代,ARM 改革了一度使用的、冗长的、需要“解码”的数字命名法,转到另一种看起来比较整齐的命名法。例如,ARMv7 的三个款式都以 Cortex 作为主名。这不仅更加澄清并且“精装”了所使用的 ARM 架构,也避免了新手对架构号和系列号的混淆。

ARMv7 架构的版本中,内核架构首次从单一款式变成三种款式。

(1) A(ARMv7-A): 设计用于高性能的“开放应用平台”——越来越接近计算机,需要运行复杂应用程序的“应用处理器”。支持大型嵌入式操作系统(不一定实时),如 Symbian(诺基亚智能手机用)、Linux,以及微软公司的 Windows CE 和智能手机操作系统 Windows Mobile。这些应用需要强的处理性能,并且需要硬件 MMU 实现的完整而强大的虚拟内存机制,还基本上会配有 Java 支持,有时还要求一个安全程序执行环境(用于电子商务)。典型的产品包括高端手机和手持仪器、电子钱包以及金融事务处理机。

(2) R(ARMv7-R): 硬实时且高性能的处理器。目标是高端实时市场,像高档轿车的组件、大型发电机控制器、机器人手臂控制器等,它们使用的处理器不但要很好、很强大,还要极其可靠,对事件的反应也要极其敏捷。

(3) M(ARMv7-M): 认准了旧时代微处理器的应用而量身定制。在这些应用中,尤其是对于实时控制系统,低成本、低功耗、极速中断反应以及高处理效率,都是至关重要的。

Cortex 系列是 v7 架构的第一次亮相,其中 Cortex-M3 就是按款式 M 设计的。

根据 ARM7TDMI-S 和 Cortex-M3 比较(见表 1-1)可以看出,这两种处理器在架构上不同,Cortex-M3 有更多的中断,延迟也大大减小。总的来说,各方面的性能都有大幅度的提高。

表 1-1 ARM7TDMI-S 和 Cortex-M3 比较(采用 100MHz 频率和 TSMC 0.18G 工艺)

特 性	ARM7TDMI-S	Cortex-M3
架构	ARMv4T(冯·诺依曼)	ARMv7-M(哈佛)
ISA 支持	Thumb/ARM	Thumb/Thumb-2
流水线	3 级	3 级+分支预测
中断	FIQ/IRQ	NMI+1 到 240 个物理中断
中断延迟	24~42 个时钟周期	12 个时钟周期
休眠模式	无	集成
内存保护	无	8 区域内存保护单元
Dhrystone	0.95 DMIPS/MHz(ARM 模式)	1.25DMIPS/MHz
功耗	0.28mW/MHz	0.19mW/MHz
面积	0.62mm ² (仅内核)	0.86mm ² (内核+外设) ^①

注: ①不包含可选系统外设(MPU 和 ETM)或者集成的部件。

表 1-2 所示是 ARM 处理器的命名和架构之间的关系。Jazelle 是 ARM 处理器的硬件 Java 加速器。MMU 为存储器管理单元,用于实现虚拟内存和内存的分区保护,这是应用处理器与嵌入式处理器的分水岭。把 MPU 认为是 MMU 的功能子集,它只支持分区保护,不支持具有“定位决定性”的虚拟内存机制。

表 1-2 ARM 处理器的命名与架构

处理器名字	架构版本号	存储器管理特性	其他特性
ARM7TDMI	v4T		
ARM7TDMI-S	v4T		
ARM7EJ-S	v5E		DSP, Jazelle
ARM920T	v4T	MMU	
ARM922T	v4T	MMU	
ARM926EJ-S	v5E	MMU	DSP, Jazelle
ARM946E-S	v5E	MPU	DSP
ARM966E-S	v5E		DSP
ARM968E-S	v5E		DMA, DSP
ARM966HS	v5E	MPU(可选)	DSP
ARM1020E	v5E	MMU	DSP
ARM1022E	v5E	MMU	DSP
ARM1026EJ-S	v5E	MMU 或 MPU	DSP, Jazelle
ARM1136J(F)-S	v6	MMU	DSP, Jazelle
ARM1176JZ(F)-S	v6	MMU+TrustZone	DSP, Jazelle
ARM11MPCore	v6	MMU+多处理器缓存支持	DSP
ARM1156T2(F)-S	v6	MPU	DSP
Cortex-M3	v7-M	MPU(可选)	NVIC
Cortex-R4	v7-R	MPU	DSP
Cortex-R4F	v7-R	MPU	DSP+浮点运算
Cortex-A8	v7-A	MMU+TrustZone	DSP, Jazelle

1.3 ARM 体系结构与特点

从学习人脑的过程也会了解到,看微处理器是多角度的。首先从宏观角度看,它是一个有着丰富引脚的芯片,个头一般比较大,比较方正。再进一步看到其组成结构,就是上面提到的计算单元+存储单元+总线+外部接口的架构了。再细化些,计算单元中会有 ALU 和寄存器组。再细些,ALU 是由组合逻辑构成的,有与门有非门;寄存器是由时序电路构成的,有逻辑有时钟。再细,与门就是一个逻辑单元。

如图 1-1 所示,任何微处理器都至少由内核、存储器、总线、I/O 构成。ARM 公司的芯片特点是内核部分都是统一的,由 ARM 设计,但是其他部分各个芯片制造商可以有自己的设计。有的甚至包含一些外设在里面。

Cortex-M3 处理器内核是微处理器的中央处理单元(CPU)。完整的基于 Cortex-M3 的 MCU 还需要很多其他组件。在芯片制造商得到 Cortex-M3 处理器内核的使用授权后,它们就可以把 Cortex-M3 内核用在自己的硅片设计中,添加存储器、外设、I/O 以及其他功能块。不同厂家设计出的微处理器会有不同的配置,包括存储器容量、类型、外设等都各具特色。本书主讲处理器内核本身。如果想要了解某个具体型号的处理器,还需查阅相关厂家提供的文档。

如果把处理器内核更加详细地画出来,图 1-2 表示出了处理器内核中包含中断控制器、