

高校嵌入式教学系列丛书
中国电子学会嵌入式专委会推荐教材

ARM嵌入式 程序设计

主 编 张 瑜 王益涵
副主编 张崇明 郭 荣
主 审 刘启中



北京航空航天大学出版社

高校嵌入式教学系列丛书

中国电子学会嵌入式专委会推荐教材

ARM 嵌入式程序设计

主 编	张 瑜	王益涵
副主编	张崇明	郭 荣
主 审	刘启中	

北京航空航天大学出版社

内 容 简 介

本书从理论和实验两方面系统介绍了 ARM 体系结构、开发环境、ARM 汇编程序设计、ARM 工作模式切换程序、异常处理程序设计、汇编和 C 语言混合编程以及 $\mu\text{C}/\text{OS}-\text{II}$ 和 Linux 内核在 ARM 上的操作系统移植等。每章最后都附有实验思考,以巩固所学内容,激发读者的创新意识。

本书可作为普通高校计算机、自动化、电子信息类专业学生的实验教材,也可作为从事嵌入式系统相关领域技术人员的参考用书。

图书在版编目(CIP)数据

ARM 嵌入式程序设计/张瑜,王益涵主编. —北京:
北京航空航天大学出版社,2009. 1
ISBN 978-7-81124-482-3

I. A… II. ①张…②王 III. 微处理器, ARM—系统设计—高等学校—教材 IV. TP332

中国版本图书馆 CIP 数据核字(2008)第 176671 号

© 2009, 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。

ARM 嵌入式程序设计

主 编 张 瑜 王益涵

副主编 张崇明 郭 荣

主 审 刘启中

责任编辑 孔祥雯 范曼华 范仲祥

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

<http://www.buaapress.com.cn> E-mail: bhpres@263.net

北京时代华都印刷有限公司印装 各地书店经销

*

开本: 787×960 1/16 印张: 16.75 字数: 375 千字

2009 年 1 月第 1 版 2009 年 1 月第 1 次印刷 印数: 4 000 册

ISBN 978-7-81124-482-3 定价: 28.00 元

前 言

随着嵌入式系统的发展和普及,吸引越来越多的大学电类专业的学生和业余爱好者投入到 ARM 的学习和开发中。ARM 是一门应用性和实践性很强的技术,然而,目前市场上的 ARM 书籍主要侧重于理论介绍,而与之相应的实验类教材非常少。针对这个问题,本书以实例为主线,从初识 ARM 体系结构、开发环境及编写第一个 ARM 程序开始,到最后完成异常处理程序设计以及 $\mu\text{C}/\text{OS}-\text{II}$ 和 Linux 内核在 ARM 上的操作系统移植,由易到难,由简到繁,循序渐进,从而使嵌入式初学者能轻松入门,为以后进一步从事嵌入式系统的开发打下坚实的基础。

本书具有以下特点:

(1) 方法新颖

面向 ARM 的初学者,采用了“比较”的方法,即从熟悉的 Intel 8086 处理器入手,引导读者逐步深入学习 ARM。通过对 Intel 8086 与 ARM 的层层比较和分析,帮助 ARM 初学者在总结 Intel 8086 体系结构和编程模式的基础上能够较快地掌握 ARM 编程。

(2) 通俗易懂

文字表述浅显易懂,所选的 35 个 ARM 编程和操作系统在 ARM 上的移植实例,由浅入深,循序渐进,通用性强,并配以大量图表说明,尤其适合于没有硬件基础的嵌入式系统的初学者,对于具有一定基础的 ARM 开发人员也有相当大的参考价值。

全书共分 13 章。

第 1 章介绍了 ARM 基础知识及处理器体系结构实验。

第 2 章初步介绍了 ARM 开发工具 ADS 及 ADS 下简单 C 语言程序实验。

第 3 章介绍了简单的 ARM 汇编程序设计及 ADS 下简单 ARM 汇编程序实验。

第 4 章介绍了寻址方式及 ARM 寻址方式实验。

第 5 章介绍了数据处理指令及 ARM 数据处理指令实验。

第 6 章介绍了数据加载与存储指令及 ARM 数据加载与存储指令程序实验。

第 7 章介绍了控制转移指令及 ARM 控制转移指令程序实验。

第 8 章介绍了子程序设计及 ARM 子程序设计实验。

第 9 章介绍了处理器的工作模式及 ARM 工作模式切换程序实验。

第 10 章介绍了汇编语言和 C 语言混合编程及其实验。

第 11 章介绍了中断处理及 ARM 异常处理程序实验。

第 12 章介绍了 $\mu\text{C}/\text{OS}-\text{II}$ 基础及 $\mu\text{C}/\text{OS}-\text{II}$ 在 ARM 上的移植实验。



第 13 章介绍了嵌入式 Linux 基础及 Linux 在 ARM 上的移植与开发实验。

本书由张瑜、王益涵任主编,张崇明、郭荣任副主编。第 1、8、11 章由张瑜、王益涵编写,第 9、10、12 章由王益涵编写,第 2~4 章及第 13 章由张崇明编写,第 5~7 章及 8.1.3 小节由郭荣编写。张瑜对各章进行了修改和统稿。刘启中教授对全书进行了审定。

本书编写中,作者参考了大量的有关书籍,从同行中学到了很多知识,并参考了相关网站资料,而且得到了上海工程技术大学、上海师范大学、中北大学、北京航空航天大学等单位教师 and 领导的大力支持,中国电子学会嵌入式专委会和博创科技为本书组稿做了大量工作,汪春梅副教授对本书的编写提供了技术方面的协作,并提出了许多宝贵的建议,在此一并表示衷心的感谢。

作者虽然从事嵌入式系统的教学和研究工作多年,但由于 ARM 的发展极为迅速及作者研究方向的限制,本书难免有疏漏和错误之处,敬请读者和同行批评指正。

作 者

2008 年 9 月

目 录

第 1 章 ARM 基础知识	1
1.1 发展历史	1
1.1.1 Intel 80x86 的发展历史	1
1.1.2 ARM 的发展历史	3
1.2 功能结构	5
1.2.1 Intel 8086 功能结构	5
1.2.2 ARM 功能结构	6
1.3 寄存器组	7
1.3.1 Intel 8086 寄存器组	7
1.3.2 ARM 寄存器组	8
1.4 存储器组织	8
1.4.1 Intel 8086 存储器组织	8
1.4.2 ARM 存储器组织	8
1.5 指令系统	9
1.5.1 一般计算机的指令系统	9
1.5.2 Intel 8086 指令系统	9
1.5.3 ARM 指令系统	10
1.6 实验 1 处理器体系结构实验	12
1.6.1 实验目的	12
1.6.2 实验环境	12
1.6.3 实验内容	12
1.6.4 实验过程	13
1.6.5 实验思考	20
第 2 章 初识 ARM 开发工具 ADS	21
2.1 从 MASM 到 ADS	21
2.1.1 MASM 简介	21
2.1.2 ADS 简介	22



2.1.3	MASM 与 ADS 命令行开发工具的比较	22
2.2	ARM 图形化界面 CodeWarrior	23
2.3	ARM 调试工具 AXD	26
2.4	实验 2 ADS 下简单 C 语言程序实验	29
2.4.1	实验目的	29
2.4.2	实验环境	29
2.4.3	实验内容	29
2.4.4	实验过程	30
2.4.5	实验思考	41
第 3 章 简单的 ARM 汇编程序设计		42
3.1	ARM 汇编程序设计的一般过程	42
3.2	ARM 程序的编辑	43
3.3	ARM 编译器设置	44
3.4	ARM 链接器设置	45
3.5	ARM 程序的编译和链接	48
3.6	ARM 程序执行过程的跟踪和调试	50
3.7	实验 3 ADS 下简单 ARM 汇编程序实验	52
3.7.1	实验目的	52
3.7.2	实验环境	52
3.7.3	实验内容	52
3.7.4	实验过程	52
3.7.5	实验思考	57
第 4 章 寻址方式		58
4.1	寻址方式的基本概念	58
4.2	Intel 8086 的寻址方式	58
4.3	ARM 的寻址方式	59
4.4	实验 4 ARM 寻址方式实验	62
4.4.1	实验目的	62
4.4.2	实验环境	62
4.4.3	实验内容	62
4.4.4	实验过程	62
4.4.5	实验思考	72

第 5 章 数据处理指令	73
5.1 Intel 8086 的数据处理指令	73
5.1.1 数据传送指令	73
5.1.2 算术运算指令	74
5.1.3 逻辑运算指令	74
5.1.4 移位指令	75
5.2 ARM 数据处理指令	75
5.2.1 寄存器传送指令	76
5.2.2 移位指令	77
5.2.3 算术运算指令	79
5.2.4 逻辑运算指令	79
5.2.5 比较和测试指令	80
5.2.6 乘法指令	81
5.3 实验 5 ARM 数据处理指令实验	83
5.3.1 实验目的	83
5.3.2 实验环境	83
5.3.3 实验内容	83
5.3.4 实验过程	83
5.3.5 实验思考	92
第 6 章 数据加载与存储指令	93
6.1 ARM 的 MOV 指令与 LDR/STR 指令比较	93
6.2 ARM 的 Load/Store 存储体系结构	93
6.2.1 字和无符号字节单寄存器的加载和存储指令	94
6.2.2 半字和有符号字节单寄存器的加载和存储指令	95
6.2.3 多寄存器的加载和存储指令	97
6.2.4 需要注意的问题	101
6.3 实验 6 ARM 数据加载与存储指令实验	106
6.3.1 实验目的	106
6.3.2 实验环境	106
6.3.3 实验内容	106
6.3.4 实验过程	106
6.3.5 实验思考	114



第 7 章 控制转移指令	115
7.1 Intel 8086 的控制转移指令	115
7.1.1 无条件转移和条件转移指令	115
7.1.2 循环控制指令	116
7.2 ARM 控制转移指令	116
7.2.1 转移和转移链接指令 B 和 BL	116
7.2.2 转移交换指令 BX	120
7.3 实验 7 ARM 控制转移指令实验	122
7.3.1 实验目的	122
7.3.2 实验环境	122
7.3.3 实验内容	122
7.3.4 实验过程	123
7.3.5 实验思考	130
第 8 章 子程序设计	132
8.1 Intel 8086 的子程序调用和返回指令	132
8.1.1 子程序调用指令 CALL	132
8.1.2 子程序返回指令 RET	132
8.2 ARM 子程序的调用和返回	133
8.2.1 ARM 子程序的调用	133
8.2.2 ARM 子程序的返回	133
8.2.3 ARM 子程序的调用和返回过程	134
8.2.4 参数传递规则	136
8.3 实验 8 ARM 子程序设计实验	137
8.3.1 实验目的	137
8.3.2 实验环境	137
8.3.3 实验内容	137
8.3.4 实验过程	137
8.3.5 实验思考	147
第 9 章 处理器的工作模式	148
9.1 Intel 80x86 的工作模式	148
9.2 ARM 的工作模式	149

9.2.1 ARM 工作模式相关寄存器	150
9.2.2 ARM 工作模式间的切换	151
9.3 实验 9 ARM 工作模式切换程序实验	152
9.3.1 实验目的	152
9.3.2 实验环境	152
9.3.3 实验内容	152
9.3.4 实验过程	152
9.3.5 实验思考	166
第 10 章 汇编和 C 语言混合编程	167
10.1 C 程序嵌入汇编指令	167
10.2 C 程序调用汇编程序函数	169
10.3 汇编程序调用 C 程序函数	170
10.4 汇编程序访问 C 程序变量	170
10.5 实验 10 汇编和 C 语言混合编程实验	170
10.5.1 实验目的	170
10.5.2 实验环境	170
10.5.3 实验内容	170
10.5.4 实验过程	171
10.5.5 实验思考	181
第 11 章 中断处理	182
11.1 中断的基本概念	182
11.2 Intel 8086 的中断处理	182
11.3 ARM 的异常处理	183
11.3.1 ARM 的异常	183
11.3.2 ARM 的异常类型及其模式	183
11.3.3 异常处理	184
11.4 实验 11 ARM 异常处理程序	184
11.4.1 实验目的	184
11.4.2 实验环境	184
11.4.3 实验内容	185
11.4.4 实验过程	185
11.4.5 实验思考	192



第 12 章 $\mu\text{C}/\text{OS}-\text{II}$ 基础	194
12.1 $\mu\text{C}/\text{OS}-\text{II}$ 概述	194
12.1.1 $\mu\text{C}/\text{OS}-\text{II}$ 的特点	194
12.1.2 $\mu\text{C}/\text{OS}-\text{II}$ 的内核结构	196
12.2 $\mu\text{C}/\text{OS}-\text{II}$ 移植条件	197
12.2.1 对目标处理器的要求	197
12.2.2 对移植者的要求	199
12.3 $\mu\text{C}/\text{OS}-\text{II}$ 移植相关文件	200
12.3.1 INCLUDES. H	200
12.3.2 OS_CPU. H	200
12.3.3 OS_CPU_C. C	202
12.3.4 OS_CPU_A. ASM	203
12.4 实验 12 $\mu\text{C}/\text{OS}-\text{II}$ 在 ARM 上的移植实验	206
12.4.1 实验目的	206
12.4.2 实验环境	206
12.4.3 实验内容	206
12.4.4 实验过程	206
12.4.5 实验思考	214
第 13 章 嵌入式 Linux 基础	216
13.1 嵌入式 Linux 概述	216
13.1.1 Linux 简介	216
13.1.2 Linux 的内核结构	217
13.1.3 嵌入式 Linux 的主要版本	218
13.1.4 嵌入式 Linux 的应用	218
13.2 嵌入式 Linux 的开发流程	219
13.3 嵌入式 Linux 的开发环境	220
13.3.1 宿主机开发环境的搭建	220
13.3.2 嵌入式 Linux 开发环境的使用	223
13.4 实验 13 Linux 在 ARM 上的移植与开发实验	229
13.4.1 实验目的	229
13.4.2 实验环境	229
13.4.3 实验内容	229

13.4.4 实验过程.....	229
13.4.5 实验思考.....	239
附录 A ARM 指令表	240
附录 B ARM 伪指令表	243
附录 C ARM 伪操作表	244
附录 D 实验思考参考答案.....	247
参考文献.....	254

第 1 章

ARM 基础知识

80x86 是美国 Intel 公司开发的微处理器系列芯片。ARM (Advanced RISC Machines) 既是成立于英国剑桥专门从事出售芯片设计技术授权的公司名字,也是对一类微处理器的通称。基于 ARM 体系结构的处理器,除了具有 ARM 体系结构的共同特点以外,每一个系列的 ARM 微处理器都还具有各自的特点及应用领域。

1.1 发展历史

1.1.1 Intel 80x86 的发展历史

1971 年,Intel 公司开发出了第一代微处理器 4004,它是一款 4 位微处理器;1972 年,Intel 公司推出了第一款 8 位微处理器 8008;1974 年,Intel 公司又推出了为多种应用而设计的 8 位微处理器 8080,它是 Intel 公司的第二代微处理器,也是第一款通用微处理器,其功能相当强。8080 为 Intel 公司成为当今 CPU 的主流打下了坚实的基础。1976 年,Intel 公司公布了 8080 的变种 8080A,之后还公布了作为 8080A 增强型的 8085。

1978 年,Intel 公司率先推出了第三代微处理器——16 位微处理器 8086。它具有两个关键的结构概念:存储器分段和指令译码表。Intel 公司的 80x86 家族也由此开始诞生。8086 内部分成两部分:总线接口部件 BIU(Bus Interface Unit)和执行部件 EU(Execution Unit)。

为了方便与 8 位外部接口或设备相连,Intel 公司于 1979 年又推出了 8088。8088 是 8086 的 8 位版,它具有与 8086 相同的内部结构,包括 EU、BIU 和 16 位的寄存器,所不同的是 8088 对外只有 8 根数据线,总是按字节取内存单元。8088 也称为准 16 位微处理器。

1981 年,Intel 公司又推出了 80186。除了 8086 所具有的特性外,80186 还集成了若干通用系统所需的部件,包括 1 个片选逻辑部件、2 个独立的高速直接存储器访问通道、3 个可编程时钟和 1 个可编程中断控制器,这些部件使 80186 功能更强。

1982 年 2 月,Intel 公司又推出了一种超级 16 位微处理器——80286。它在速度和性能上比 8086/8088 和 80186 都有了很大的提高,具有 24 条地址线,可寻址的最大物理空间达 16 MB。由于它具有大批量数据处理、存储保护和多道程序处理能力,并支持迫切需要的虚拟存储系统,因此它已成为多任务、多用户系统的核心。80286 有 4 个独立的处理部件,分别是



执行部件 EU、总线部件 BU、指令部件 IU 和地址部件 AU,这 4 个部件能同时并行工作。与 8086 相比,80286 效率更高。

第四代微处理器是 32 位微处理器,Intel 80x86 家族的 32 位微处理器开始于 80386。1985 年 10 月,Intel 公司推出了 32 位微处理器 80386。它是微处理器发展进程中的里程碑。80386 兼容先前的 8086/8088、80186 和 80286。它全面支持 32 位数据类型和 32 位操作,通用寄存器等也扩展到 32 位。80386 支持实方式和保护方式两种运行模式。在实方式下,80386 相当于一个可进行 32 位处理的快速的 8086。只有在保护模式下,80386 才能真正地发挥其全部强大的功能。80386 在保护方式下会支持称为虚拟 8086 方式的运行模式。

1989 年 4 月,Inter 公司推出了 80486。它是 80x86 家族中继 80386 之后的又一种功能更强大的 32 位微处理器,兼容先前的 8086/8088、80186、80286 和 80386。简单地说,80486 是在微处理器 80386 的基础上集成数值协处理器 80387 和 Cache 而构成,它采用流水线的方式执行指令。

1993 年 3 月,Intel 公司推出了新一代微处理器 Pentium。它的性能比 80486 又有较大幅度的提高,而且兼容 8086/8088、80186、80286、80386 和 80486。Pentium 支持的数据总线位数达到 64 位,支持的物理地址位数是 32 位,内部寄存器仍是 32 位。Pentium 采用超标量体系结构,拥有两条“流水线”——U 流水线和 V 流水线。U 流水线和 V 流水线都能执行整数指令,U 流水线还能执行浮点指令。Pentium 还开始支持动态分支预测。它有两个独立的 Cache,即使 Pentium 以 80486 相同的频率工作,其整数运算性能仍可提高 1 倍,浮点性能可提高 5 倍。

1995 年 11 月,Intel 公司推出了更新一代的微处理器 Pentium Pro。它支持的数据总线位数是 64 位,支持的物理地址位数是 36 位,内部寄存器是 32 位。除了像 Pentium 那样具有两个独立的容量为 8 KB 的 L1 级 Cache 分别作为指令 Cache 和数据 Cache 外,Pentium Pro 还集成了一个 256 KB 或 512 KB 的 L2 级 Cache。L2 级 Cache 能以处理器的全速运行。Intel 公司在 Pentium Pro 上实现了既是超标量的,又是超流水线的;既能支持乱序执行,又能支持寄存器重命名;既开发了分支预测,又开发了推测执行。所有这些统称为动态执行。动态执行时的 Pentium Pro 在大多数情况下处理指令比 Pentium 的效率更高。

1997 年,Intel 公司推出了 Pentium II 处理器。它包括了 Intel 公司许多最新的技术。Pentium II 处理器集成了 750 万个晶体管,并整合了 MMX 指令集技术,可以更快、更流畅地播放影音 Video、Audio 以及图像等多媒体数据。

1999 年,Intel 公司推出了 Pentium III 处理器。Pentium III 处理器集成了从 Compaq 公司购买的 P6 动态执行体系结构,采用双独立系统总线(DIB)架构、多路数据传输系统总线和 MMX 多媒体增强技术。除此之外,Pentium III 处理器还提供了专为增强互联网流媒体应用的 70 条指令,可以更好地支持图像、3D 流媒体和音视频应用。

2000 年,Intel 公司推出了 Pentium IV 处理器。它采用了高级传输缓存(advanced trans-

fer cache)、高级动态执行(advanced dynamic execution)、执行追踪缓存(execution trace cache)、高级分支预测(enhanced branch prediction)、超长管道处理技术(hyper pipelined technology)、快速执行引擎(rapid execution engine)、高级浮点以及多媒体指令集等多种新技术。

2001 年, Intel 公司发布了 Xeon 处理器。Xeon 处理器的市场定位也更加瞄准高性能、均衡负载、多路对称处理等特性, 而这些特性是台式计算机的 Pentium 品牌所不具备的。Xeon 处理器实际上还是基于 Pentium IV 的内核, 而且同样是 64 位的数据带宽, 但由于其采用了与 AGP 4X 相同的原理——“四倍速”技术, 因此其前端总线有了巨大的提升。一方面, Xeon 处理器基于 Intel 公司的 NetBurst 架构, 有更高级的网络功能及更复杂、更卓越的 3D 图形性能; 另一方面, 支持 Xeon 的芯片组也在并行运算、支持高性能 I/O 子系统(如 SCSI 磁盘阵列、千兆网络接口)、支持 PCI 总线分段等方面更好地支持服务器端的运算。

2005 年 4 月, Intel 公司正式发布了至尊版双核处理器 Pentium Extreme Edition 840, 标志着 Intel 处理器进入了一个全新的双核和多核时代。2005 年 5 月, Intel 公司又发布了 3 款双核处理器 Pentium D820/830/840, 将 PC 用户带入了一个双核和多核的新时代。

1.1.2 ARM 的发展历史

ARM (Advanced RISC Machines)既可以认为是一个公司的名字, 也可以认为是对一类微处理器的通称, 还可以认为是一种技术的名字。

1985 年 4 月, 第一个 ARM 原型在英国剑桥的 Acorn 计算机有限公司诞生, 由美国加州 SanJose VLSI 技术公司制造。20 世纪 80 年代后期, ARM 很快开发成 Acorn 的台式机产品, 形成英国的计算机教育基础。

1991 年, ARM 公司成立于英国剑桥, 主要出售芯片设计技术的授权。ARM 公司是专门从事基于 RISC 技术芯片设计开发的公司, 作为知识产权供应商, 本身不直接从事芯片生产, 而是靠转让设计许可由合作公司生产各具特色的芯片。

ARM 微处理器目前包括 ARM7、ARM9、ARM9E、ARM10E、SecurCore 和 Intel 的 Xscale 几个系列, 以及其他厂商基于 ARM 体系结构的处理器。除了具有 ARM 体系结构的共同特点以外, 每一个系列的 ARM 微处理器都具有各自的特点及应用领域。

ARM7 系列微处理器为低功耗的 32 位 RISC 处理器, 其内核是 3 级流水线, 采用冯·诺伊曼结构。ARM7 系列微处理器包括 ARM7TDMI、ARM7TDMI-S、ARM720T 和 ARM7EJ 几种类型的核。其中, ARM7TDMI 是目前使用最广泛的 32 位嵌入式 RISC 处理器, 属低端 ARM 处理器核。TDMI 的基本含义: T——支持 16 位压缩指令集 Thumb; D——支持片上 Debug; M——内嵌硬件乘法器(Multiplier); I——嵌入式 ICE, 支持片上断点和调试点。

ARM9 系列微处理器在高性能、低功耗特性方面提供最佳的性能。ARM9 内核是 5 级流水线, 采用哈佛结构; 带有 MMU、CACHE 等; 时钟频率比 ARM7 更高; 采用哈佛结构区分数



据总线和指令总线；指令执行效率比 ARM7 约高 27%。ARM9 系列微处理器包含 ARM920T、ARM922T 和 ARM940T 三种类型。

ARM9E 系列微处理器为可综合处理器，使用单一的处理器内核就提供了微控制器、DSP 及 Java 应用系统的解决方案，极大地减小了芯片的面积，降低了系统的复杂度。ARM9E 系列微处理器提供了增强的 DSP 处理能力，很适合那些需要同时使用 DSP 和微控制器的应用场合。ARM9E 系列微处理器包含 ARM926EJ - S、ARM946E - S 和 ARM966E - S 三种类型。

ARM10E 系列微处理器具有高性能、低功耗的特点，由于采用了新的体系结构，与同等的 ARM9 器件相比较，在同样的时钟频率下，性能提高了近 50%；同时，由于采用了两种先进的节能方式，使其功耗极低。ARM10E 系列微处理器包含 ARM1020E、ARM1022E 和 ARM1026EJ - S 三种类型。

SecurCore 系列微处理器专为安全需要而设计，提供了完善的 32 位 RISC 技术的安全解决方案，因此，该系列微处理器除了具有 ARM 体系结构的低功耗、高性能的特点外，还具有其独特的优势，即提供了对安全解决方案的支持。SecurCore 系列微处理器包含 SecurCoreSC100、SecurCoreSC110、SecurCoreSC200 和 SecurCoreSC210 四种类型。

ARM 公司推出嵌入式核心之后，DEC 公司获得许可并在此基础上开发出增强版的 StrongARM 处理器，后来 DEC 公司被康柏所合并，而 StrongARM 核心则被 Intel 公司买走。属于该体系的 StrongARM SA - 1110 处理器被长时间用于 Pocket PC(PDA 中的一种，采用 Windows CE 操作系统)中。StrongARM SA - 1110 处理器融合了 Intel 公司的设计和处理技术以及 ARM 体系结构的电源效率，采用在软件上兼容 ARMv4 体系结构，同时采用具有 Intel 技术优点的体系结构。该处理器也是 Microsoft 公司的 Pocket PC 战略的奠基石。而在此之前，市场上的 WinCE 设备正被不同的 CPU 造成的软件兼容性问题困扰着。Intel 公司的 StrongARM 处理器是便携式通信产品和消费类电子产品的理想选择，已成功地应用于多家公司的掌上电脑系列产品中。其代表产品：康柏 iPAQ 3630 和 NEC MP300 等一些基于 PocketPC/WinCE3.0 的机型。另外，许多中国台湾的厂商也推出了很多基于该系列处理器的产品。

在 PDA 领域，Intel 公司的 StrongARM 和 XScale 处理器占据着举足轻重的地位。这两者在架构上都属于 ARM 体系，相当于 ARM 的一套实际应用方案。StrongARM - SA1100 的集成度较低，许多功能都必须借助第三方芯片实现，而且存在一些老接口及无用的功能。Intel 公司在接手之后对其进行改进，并在 2002 年 2 月正式推出基于 StrongARM 的架构——XScale。XScale 处理器是基于 ARMv5TE 体系结构的解决方案，是一款全性能、高性价比、低功耗的处理器。它支持 16 位的 Thumb 指令和 DSP 指令集。

所有 ARM 内核，包括流行的 ARM7、ARM9 和 ARM10E 等，都是单周期指令机。下一代 ARM 将向每周期能处理多重指令的超标量机发展，并将继承传统(比如保留以往内核的代

码兼容性),工作效率将大大提高。

1.2 功能结构

1.2.1 Intel 8086 功能结构

8086 微处理器的内部组成结构从功能上讲,它由两个独立的逻辑单元组成,即总线接口单元 BIU 和执行单元 EU。

1. 总线接口单元

总线接口单元 BIU 的功能是负责完成 CPU 与存储器或 I/O 设备之间的数据传送,即 BIU 负责从存储器的指定单元预取指令,送至指令流队列缓冲器中排队或直接传送给 EU,或者把执行单元的操作结果传送到指定的存储单元或外设端口中。

总线接口单元 BIU 内部设有 4 个 16 位段地址寄存器:代码段寄存器 CS、数据段寄存器 DS、堆栈段寄存器 SS 和附加段寄存器 ES,以及 1 个 16 位指令指针寄存器 IP、1 个 6 字节指令队列缓冲器、20 位地址加法器和总线控制电路。

(1) 指令队列缓冲器

8086 的指令队列为 6 字节(8088 为 4 字节)。当 EU 正在执行指令且不占用总线时,总线接口单元 BIU 会自动地将预取的指令按先后次序存入指令队列缓冲器中排队,再由 EU 按顺序取出执行。当 EU 执行完转移、调用和返回指令后,自动将指令队列缓冲器清空,BIU 就从新的地址重新取指令,将取到的第一条指令直接经指令队列送到 EU 执行,并将后取的指令填入指令队列缓冲器。

(2) 地址加法器

地址加法器用于将逻辑地址转换成 20 位的物理地址。

(3) 指令指针寄存器 IP

IP 用于存放 BIU 将要读取的下一条指令的段内偏移地址。IP 在执行过程中能够完成自动加 1 操作,使得能够不断地指向要执行的下一条指令。

2. 执行单元

执行单元 EU 的功能是从 BIU 的指令队列中取出指令代码,然后执行指令所规定的全部功能。EU 由 1 个 16 位的算术逻辑单元、8 个 16 位的寄存器、1 个 16 位的标志寄存器、1 个数据暂存寄存器和 EU 的控制电路组成。

一个 16 位的 ALU 总线将 EU 中的各部件连接在一起,在其内部实现快速数据传输。这个内部总线与 CPU 外接的总线之间是隔离的,因此这两个总线可以并行工作。

EU 主要完成两种类型的操作:算术/逻辑运算;计算指令要求寻址的所在单元地址的 16 位偏移地址(有效地址 EA)并送至 BIU,由 BIU 最后形成一个 20 位的物理地址。