

ARM SoC 体系结构

ARM System-on-Chip Architecture
second edition

[英] Steve Furber 著

田 泽 于敦山 盛世敏 译



北京航空航天大学出版社

<http://www.buaapress.com.cn>

内 容 简 介

介绍了一般微处理器核的设计原理、基于微处理器核的 SoC 设计的基本概念和方法,通过对 ARM 系列处理器核和 CPU 核的详尽描述,来说明微处理器及外围接口的设计原理和方法。同时也综述了 ARM 系列处理器核和最新 ARM 核的研发成果,以及 ARM 和 Thumb 编程模型,对 SoC 设计中涉及到的存储器层次、Cache、存储器管理、片上总线、片上调试和产品测试等主要问题进行了论述。在此基础上给出了几个基于 ARM 核的 SoC 嵌入式应用的实例。最后对基于异步设计的 ARM 核 AMULET 及异步 SoC 子系统 AMULET3H 的研究进行了介绍。

本书的特点是将基于 ARM 微处理器核的 SoC 设计和实际嵌入式系统的应用集成于一体,对于基于 ARM 核的 SoC 设计和嵌入式系统开发者来说是一本很好的参考手册。可用作计算机科学与技术、电气工程、电子科学与技术专业本科生及硕士研究生的教材。也可作为从事集成电路设计的工程技术人员、基于 ARM 的嵌入式系统应用开发技术人员的参考书。

图书在版编目(CIP)数据

ARM SoC 体系结构 / (英) 弗伯 (Furber, S.) 著; 田
泽等译. — 北京: 北京航空航天大学出版社, 2002. 9
ISBN 7-81077-170-1

I. A… II. ①弗… ②田… III. 微处理器, ARM
系统结构—教材 IV. TP332

中国版本图书馆 CIP 数据核字(2002)第 072396 号

本书英文版原名: ARM System-on-Chip Architecture, second edition by Steve Furber.

Copyright © Pearson Education Limited 2000. All rights reserved.

This translation of ARM System-on-Chip Architecture, second edition is published by arrangement with Pearson Education Limited.

本书中文简体字版由 Pearson Education Limited 授权北京航空航天大学出版社在中华人民共和国境内(不包括香港特别行政区)独家出版发行。版权所有。

北京市版权局著作权登记号: 图字: 01-2001-4587

ARM SoC 体系结构

ARM System-on-Chip Architecture, second edition

[英] Steve Furber 著

田 泽 于敦山 盛世敏 译

责任编辑 马广云

责任校对 戚 爽

*

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(100083) 发行部电话: (010)82317024 传真: (010)82328026

<http://www.buaapress.com.cn>

E-mail: pressell@publica.bj.cninfo.net

北京宏文印刷厂印装 各地书店经销

*

开本: 787×1 092 1/16 印张: 24 字数: 614 千字

2002 年 10 月第 1 版 2002 年 10 月第 1 次印刷 印数: 5000 册

ISBN 7-81077-170-1 定价: 55.00 元

中文版前言

欢 迎

第一片 ARM 处理器的开发距今已近 20 年了,在这 20 年中它已经发展成为世界领先的 32 位嵌入式处理器核,已向大多数世界领先的半导体公司授权或由他们制造。由于中国在全球半导体业界的影响日益增长,能够专门为我的书的中文版撰写前言令我异常激动。我希望更多的读者在以本国语言阅读本书后,能够鉴赏 ARM 体系结构一流的简易特性,以及它为基于 ARM 核的、先进的低功耗 SoC(System-on-Chip)创造的许多商机。

ARM 的起源

第一片 ARM 核是由 Acorn Computers Limited 公司在 1983—1985 年间开发的。这是一个总部设在剑桥的小型英国公司,当时大约有 400 名职员。处理器是使用很简单的工具设计的。行为模型是以 BASIC 写的寄存器传输级描述,而且用手工转换成逻辑图。VLSI 设计使用的是运行于 Apollo 工作站的 VLSI Technology, Inc. 的工具。全部硬件和 VLSI 的设计小组只有 4 个人,还有相当多的软件工程师从事程序测试和指令仿真。

小尺寸、低功耗和简易性都源于拥有非常有限资源的小规模开发小组。英语中有个说法:“需要是发明之母”。这句话不再适用于开发第一片 ARM 处理器的情形。开发小组很小,获得的资源也很少,以前在处理器设计方面几乎没有经验。在这样的环境下没有其他选择,只能尽力使设计在所有方面尽可能简单,否则,成功的机会非常小。这种简单化的动力直接导致了一个很小因而很适合低功耗 SoC 应用的核。它在 20 世纪 90 年代促成了便携式消费产品新的冲击波。

当然,技术只是事业成功的一个方面,使 ARM 今天享有优势地位的还有许多其他因素。但是从一开始,技术很好地迎合了市场的需要,并不是因为资源和经验丰富的小组实行了认真的计划和深思熟虑的决定来产生优化的解决方案;而是因为弱小的、无经验的小组别无选择,只有进行小型且简单的设计,而这恰好是正确答案。ARM 能获得今天这样全面的成功,运气起到了主要作用。

体系结构的发展趋势

为迎合市场的要求,ARM 处理器原有的简单性不得不为某些扩展让路。最初,体系结构最重要的扩展是提供对片上调试的支持,以及为小型嵌入式应用改善代码密度;后来又增加了指令来改善处理器支持信号处理算法的能力;最近又作了进一步的扩展

(在本书中未予讨论)来提供一些 SIMD 功能。对体系结构的这些扩展并未影响处理器的基本操作,尽管它们明显地在细节上增加了实现的复杂性。

伴随着指令集结构的进展,微结构也出现一些变化以增强性能。写本书时,销售最好的 ARM7TDMI 采用基于 3 级流水线的微结构,这与最初 ARM 核使用的结构是一样的。但是对于像 5 级流水线 ARM9TDMI 这样更高速芯核的市场正在快速发展。以今天多数处理器的标准来看,这些依然是很简单的机器。这种简单性是这些机器实现低功耗和小硅片面积的关键。

在处理器核的外部,在支持系统的功能部件上有相当多的发展,例如存储器管理单元、Cache(高速缓存)组织和片上总线结构等。第一片 ARM 处理器既没有存储器管理单元,也没有 Cache(尽管它支持外部存储器管理硬件)。但是,如果现在要处理器实现全部潜在性能和支持现代操作系统,那么这些都是必不可少的功能部件。

最近的系统级发展的是专用“平台”——一个包括处理器、Cache、存储器管理或保护、片上总线和关键存储器与外围部件的扩展子系统。它为特定产品的快速开发提供高级模块。随着硅技术的进步和芯片上可利用的晶体管数量的增长,平台方法代表着控制设计费用和上市时间的系统方法。ARM 公司提供的 IP 块,从 20 世纪 90 年代早期简单的整数处理器核起走过了漫长的路。这还仅仅只说了硬件。ARM 还得到了范围不断扩大的软件的支持,包括支持早期处理器核所必需的开发工具,但现在已经远远超出这个范围了。

这种趋势今后会把我们引向何处?显然,高度集成的 SoC 是实现多数电子装置的理想方式。单芯片器件在功耗、性能、大小和可靠性上的优势是无法抵挡的。但是,随着复杂性的提高,设计的成本也提高了,而且最小的经济加工量也提高了。这些经济问题将很快开始排除专用 SoC,除非它的产量很高。需要的是多用途的 SoC 平台,或许包括可编程逻辑部分,它能使芯片配置为若干不同的应用,从而增加加工量并使开发成本由若干产品分摊。问题是确定处理器性能、存储器容量、总线带宽和可编程逻辑的最佳平衡,以便不用多余的开销即可覆盖大范围的应用。这是主要的设计挑战!

模块化设计

设计复用的关键是模块化。让我们来看一看未来技术将支持的非常复杂的 SoC。很明显,片上互联(将不同模块连接在一起)是一个越来越重要的问题。单一的片上总线已经不能适应很多应用,已使用层次化的总线来提供必要的带宽。有一条新术语是“芯片区网络(chip area network)”,它在我们心中描绘了一幅灵活的互联系统的画面。这个互联系统可以使用全部种类的网络拓扑(环形、星形、总线形、交叉形和交换形)来服务片上互联要求。

目前的片上总线(例如 ARM 的 AMBA)是时钟型的。总线上所有的客户模块必须使用共同的全局时钟来实现接口。在本书的最后一章,介绍了由我所在的 Manchester(曼彻斯特)大学的研究小组作为研究样机而开发的 AMULET 异步微处理器。最近的设计采用的是异步片上总线的完整的 SoC。从那时以来,我们开发了全异步的芯片区网络技术,它支持完全的模块互联策略。该互联策略支持 GALS(全局异步本地同步)SoC 设计。在 GALS 系统中,所有基本模块可以都是传统的时钟型系统。

它们的设计能确保全局同步,并很好地控制时钟偏斜。而这些模块之间则采用异步互联,使得它们能使用任意的时钟(或使用同一时钟,但是不关心时钟的位相或任何两个模块之间的时钟偏斜)。异步互联使用对延迟不敏感的规范,以确保其鲁棒性,网络拓扑和连接带宽可以根据应用来裁剪。长线的连接可以采用流水线,使网络中任何两个模块之间获得最优的连接特性。

尽管 GALS 方法还只是一个活跃的研究领域,我深信在几年之内将成为公认的构筑复杂 SoC 的方法。我相信,在本书最后一章描述的异步设计技术将在主流产品中发挥重要作用,而 GALS 是实现这一目标的途径之一。

SoC 的挑战

设计复杂的 SoC 是当前世界最具挑战性的工程任务之一。在像这样一本简短的书中,我仅能开始描述可能遇到的问题和可能的解决方法。这是一项快速发展的学科,新思想和新方法不断翻新。你不能希望只是阅读本书就变为 SoC 设计的专家。但是我的希望是,本书将使一代工程师至少开始理解学科的基本原理和 ARM 处理器在形成当前技术发展水平中发挥的重要作用。

然而,成功的工程师需要的不仅是现代化的知识。工程是一个创造性的学科,在像半导体工业这样高速发展的产业中,最好的工程师确实是非常具有创造性的。每个新的设计都比以前的更复杂,且要求更高的性能、更低的功耗和更多的功能。在如此快速发展的工艺基础上,没有一种常规的设计方法能给出最佳的设计能力。

有许多世界最优秀的工程师以汉语为母语,在我的学生中就有几个这样的人。我希望本书的中文版本将鼓励这些工程师学习 ARM 处理器以及它在设计许多现代消费产品中的作用;我还希望我的中国读者在未来面对任何设计挑战时都能成功地应用本书提供的原理。

我期盼着及时获悉他们成功的消息。

Steve Furber
2002 年 4 月

前 言

目 标

本书中介绍的一些概念和方法,可以用于基于微处理器核的片上系统(SoC, System-on-Chip)设计,也可用于微处理器核设计。通过对 ARM 的详尽描述来具体说明微处理器的设计原理。

本书的目的是帮助读者理解 SoC 和微处理器是怎样设计和使用的,以及为什么先进的微处理器要这样来设计。对于那些只想了解微处理器一般原理的读者,通过对 ARM 的解说使得那些或许很微妙的问题变得容易理解了。想了解 ARM 设计的读者可以发现,这些基本原理恰如其分地阐明了 ARM 的理论基础。

本书没有介绍其他微处理器的结构。希望对体系结构进行比较研究的读者,可从本书找到所需要的与 ARM 相关的资料,但是关于其他微处理器设计的资料必须从其他读物中查找。

读 者

本书针对两个不同的读者群体,即

- 对那些设计基于嵌入 ARM 微处理器的 SoC 产品,或者正在考虑使用 ARM 产品的专业硬件和软件工程师的工作会很有帮助。尽管本书中的很多内容在其他 ARM 技术资料中介绍过,但本书的内容更广泛,并提供了更多的背景资料。
- 计算机科学、计算机工程以及电子工程专业的学生会从中找到对他们学业不同阶段有价值的资料。一些章节几乎是以本科阶段使用的课程教材为基础的,另一些章节是从研究生课程教材中截取的。

预备知识

本书不作为计算机结构或计算机逻辑设计的启蒙性教材。我们针对的读者应在相关领域具有相当于计算机科学或计算机工程专业大学本科二年级学生水平。书中有些内容是一年级的教材,但这些内容是以复习的方式而不是以讲新课的方式讲述的。

阅读本书不需要事先熟悉 ARM 处理器。

ARM

1985 年 4 月 26 日,ARM 的第一批样片送到英国剑桥的 Acorn 公司。这些样片是在加州圣何塞的 VLSI Technology 公司制造的。几个小时后,它们开始运行程序。在 20 世纪 80 年代后期,ARM 悄悄地发展以支持 Acorn 公司的台式计算机产品。该产品

成为英国教育界计算机的基础。20 世纪 90 年代,在 ARM 公司的精心经营下,ARM 步入世界舞台,在高性能、低功耗和低价格的嵌入式应用领域确立了市场的领先地位。

优越的市场地位增加了 ARM 的资源,加速了基于 ARM 产品的开发。

最近 10 年 ARM 开发的突出成果包括:

- 开发了称为 Thumb 的新型压缩指令格式,这种格式在小型系统中可降低成本和功耗;
- ARM9、ARM10 和 StrongARM 系列处理器的开发显著地提高了 ARM 的性能;
- 软件开发和调试环境更好;
- 基于 ARM 处理器核的嵌入式应用领域更为广阔。

现代 SoC 和处理器设计的大多数原理都在 ARM 系列处理器设计中得到了应用,并且 ARM 也开创了一些新的概念(例如指令流的动态解压缩)。基础的 3 级流水线 ARM 核固有的简单性使其成为实际处理器设计的优秀教学范例。同时,嵌入在复杂系统芯片中基于 ARM 核的调试系统代表着当今技术的前沿。

本书的结构

第 1 章首先复习了本科一年级与处理器设计相关的内容,通过复习逻辑级和门级描述方法说明硬件设计的抽象原理。然后介绍了精简指令集计算机(RISC, Reduced Instruction Set Computer)的重要概念作为以后章节的基础。最后介绍一些有关低功耗设计的问题。

第 2 章用第 1 章引入的概念介绍 ARM 处理器的体系结构。

第 3 章详细介绍用户级汇编语言编程。这部分可以作为大学一年级的教学内容。

第 4 章讲述 3 级和 5 级流水线 ARM 处理器核的组织及实现,内容适合于大学二年级的教学。本章还涉及了一些与实现相关的问题。

第 5 章和第 6 章逐渐深入地介绍 ARM 指令集。第 5 章对第 3 章中讲过的指令集作进一步的介绍,包括每条指令的二进制表示,可以使我们对指令集的理解更为深入。本章内容最好是先读一遍,然后作为参考。第 6 章主要考虑高级语言(这里指 C 语言)需求以及 ARM 指令集如何满足这些需求。这一章是以大学二年级的教材为基础写成的。

第 7 章介绍 Thumb 指令集。Thumb 指令集是 ARM 公司为满足小型嵌入式系统的代码密度和功耗要求而提出的。这对一般的计算机科学研究来说属于边缘问题,但是作为研究生课程还是比较有意思的。

第 8 章提出在处理器核嵌入式应用系统的调试和板级系统的产品测试中会遇到的一些问题。这些问题是第九章的背景资料。

第 9 章介绍一些不同的整数 ARM 核。对第 4 章的内容进一步扩展,包括带有 Thumb 指令的核、调试硬件以及更复杂的流水操作。

第 10 章引入存储器层次概念,讨论存储器管理和 Cache 的原理。

第 11 章回顾在大学二年级学习的现代操作系统对处理器的要求,并介绍 ARM 针对这些要求采取的方法。

第 12 章介绍 ARM CPU 核芯片(包括 StrongARM),包括了对存储器管理的全面支持。

第 13 章涉及使用嵌入式处理器核设计 SoC 的有关问题。ARM 在这方面位于技术的前沿。将复杂的专用系统集成到芯片上必然遇到许多问题,本章列举一些嵌入式系统芯片产品的例子来介绍针对这些问题所采取的解决方案。

第 14 章介绍 20 世纪 90 年代在英国曼彻斯特大学开发的、异步 ARM 兼容的处理器。经过 10 年的研究,到撰写本书时,AMULET 技术即将迈向商业化的第一步。这一章通过讲解 DRACO SoC 的设计,总结了第一个商业应用的 32 位异步微处理器。

附录简短地介绍了计算机逻辑设计的基本原理及第 1 章用到的术语。

书的末尾收录了本书用到的术语和进一步学习的参考文献,其后是详细的索引。

相应的课程安排

当用于大学本科教材时,本书各章节应作如下安排:

第一年:第 1 章(基本的微处理器设计);第 3 章(汇编语言编程);第 5 章(指令的二进制码和汇编语言编程的参考资料)。

第二年:第 4 章(简单的流水线微处理器设计);第 6 章(体系结构对高级语言的支持);第 10 章和第 11 章(存储器层次和体系结构对操作系统的支持)。

第三年:第 8 章(嵌入式系统的调试和测试);第 9 章(先进的流水线处理器设计);第 12 章(先进的 CPU);第 13 章(嵌入式系统举例)。

研究生课程可以跨越几个章节来组织题目,例如处理器设计(第 1、2、4、9、10 及 12 章)、指令集设计(第 2、3、5、6、7 及 11 章)和嵌入式系统(第 2、4、5、8、9 及 13 章)。

第 14 章包含相当于本科三年级或研究生课程关于异步设计的教材,但是需要大量的附加背景材料(本书没有提供)。

支持资料

本书中的许多图表可无偿地从互联网得到,但不能用于商业目的。使用它们的惟一限制是使用这些资料的任何课程都应将本书作为推荐教材。这些信息和其他支持资料可以从以下网页得到:

<http://www.cs.man.ac.uk/amulet/publications/books/ARMSysArch>

任何与商业用途有关的查询都应通过出版社。关于本书版权的声明不受上述内容影响。

反 馈

作者欢迎对本书形式和内容以及发现的任何错误的反馈意见。请将此类信息用 E-mail 发至:

sfurber@cs.man.ac.uk

致 谢

在过去的 10 年间,许多人为 ARM 的成功做出过贡献。作为一条原则的决定,我

没有在书中列出对 ARM 各项发展负有主要责任的人们,这是因为这个名单太长了,而且要想删减这个名单是很困难的。历史有一个习惯,就是只关注一两个伟人的荣誉,而常常忽略那些埋头工作的人们。但是,写关于 ARM 的书不能不提到 Sophie Wilson,他提出的最初的指令集结构沿用至今,虽有扩展但没有重大改变。

我还要感谢 ARM 公司的帮助,使我可以接触他的员工和设计文件。我还要感谢 ARM 的半导体合伙人的帮助,特别是 VLSI Technology 公司。这个公司现在已全部归 Philips 半导体公司所有。

初稿的审阅者提出了许多有益的意见,使本书定稿受益匪浅。我要感谢初稿受到的赞同和认可,以及审阅者返回的直言不讳的修改建议。Addison Wesley 出版社指导我答复这些建议和完成作者应尽的其他事宜,使我得到很大帮助。

最后,我想感谢我的妻子 Valerie、我的女儿 Alison 和 Catherine,她们使我从家庭义务中解脱出来,有时间写作本书。

Steve Furber

2000 年 3 月

商标公告

- Acorn™、Risc PC™、Acorn Archimedes™ 和 Online Media™ 是 Acorn Computer Ltd. 公司的商标。
- ARM®、StrongARM® 和 Thumb® 是 ARM Ltd. 公司的注册商标。AMBA™、Angel™、ARMMulator™、EmbeddedICE™、ARM7TDMI™、AEM7TDMI-S™、ARM9TDMI™、Piccolo™ 和 STRONG™ 是 ARM Ltd. 公司的商标。
- Apple™ 和 Newton™ 是 Apple Computer, Inc. 公司的商标。
- Digital™、Digital Semiconductor™、PDP-8™ 和 Alpha™ 是 Digital Equipment Corporation 公司的商标。
- TimeMill™、Piccolo™ 是 EPIC Design Technology, Inc. 公司的商标。
- Hagenuk® 是 Hagenuk GmbH 公司的注册商标。
- IBM® 是 International Business Machines Corporation 公司的注册商标。
- Inmos® 是 Inmos Group of Companies 公司的注册商标, transputer™ 是 Inmos Group of Companies 公司的商标。
- Intel® 是 Intel Corporation 公司的注册商标。
- MS-DOS®、Microsoft® 和 Windows® 是 Microsoft Corporation 公司的注册商标。
- MC68000™ 是 Motorola Corporation 公司的商标。
- UNIX® 是 Novell, Inc. 公司的注册商标。
- I²C Bus® 是 Philips Semiconductors 公司的注册商标, JumpStar™、N-Trace™、OneC™、Rapid Silicon Prototyping™、Ruby II™ 和 VLSI ISDN Subscriber Processor™ 是 Philips Semiconductors 公司的商标。
- Psion Series 5MX™ 是 Psion plc 公司的商标。
- Samsung SGH2400™ 是 Samsung 公司的商标。
- SPARC® 是 SPARC International, Inc. 公司的注册商标。
- SUN™ 是 Sun Microsystems, Inc. 公司的商标。
- Bluetooth™ 是 Telefonaktiebolaget LM Ericsson 公司拥有的技术商标。

本书中, BBC 一词用作 British Broadcasting Corporation 的缩写。

目 录

第 1 章 处理器设计导论

1.1 处理器体系结构和组织	1
1.2 硬件设计中的抽象	3
1.3 MU0——一个简单的处理器	6
1.4 指令集的设计	11
1.5 处理器设计中的权衡	16
1.6 精简指令集计算机	20
1.7 低功耗设计	23
1.8 例题与练习	26

第 2 章 ARM 体系结构

2.1 Acorn RISC 机器	29
2.2 体系结构的继承	30
2.3 ARM 编程模型	32
2.4 ARM 开发工具	35
2.5 例题与练习	39

第 3 章 ARM 汇编语言编程

3.1 数据处理指令	40
3.2 数据传送指令	45
3.3 控制流指令	51
3.4 编写简单的汇编语言程序	56
3.5 例题与练习	59

第 4 章 ARM 的组织和实现

4.1 3 级流水线 ARM 的组织	62
4.2 5 级流水线 ARM 的组织	65
4.3 ARM 指令执行	68
4.4 ARM 的实现	71
4.5 ARM 协处理器接口	83
4.6 例题与练习	85

第 5 章 ARM 指令集

5.1 引 言	87
5.2 异 常	89

5.3	条件执行.....	92
5.4	转移及转移链接(B,BL)指令	94
5.5	转移交换和转移链接交换(BX,BLX)指令	96
5.6	软件中断(SWI)指令	98
5.7	数据处理指令.....	99
5.8	乘法指令	103
5.9	前导 0 计数(CLZ——仅用于 v5T 体系结构)	105
5.10	单字和无符号字节的数据传送指令.....	105
5.11	半字和有符号字节的数据传送指令.....	108
5.12	多寄存器传送指令.....	110
5.13	存储器和寄存器交换指令(SWP)	111
5.14	状态寄存器到通用寄存器的传送指令.....	112
5.15	通用寄存器到状态寄存器的传送指令.....	113
5.16	协处理器指令.....	115
5.17	协处理器的数据操作.....	116
5.18	协处理器的数据传送.....	117
5.19	协处理器的寄存器传送.....	118
5.20	断点指令(BKPT——仅用于 v5T 体系结构).....	120
5.21	未使用的指令空间.....	120
5.22	存储器故障.....	122
5.23	ARM 体系结构的各种版本	126
5.24	例题与练习.....	128
第 6 章 体系结构对高级语言的支持		
6.1	软件设计中的抽象	129
6.2	数据类型	130
6.3	浮点数据类型	135
6.4	ARM 浮点体系结构	139
6.5	表达式	143
6.6	条件语句	145
6.7	循 环	148
6.8	函数与过程	150
6.9	使用存储器	154
6.10	运行环境.....	158
6.11	例题与练习.....	159
第 7 章 Thumb 指令集		
7.1	CPSR 中的 Thumb 指示位.....	161
7.2	Thumb 编程模型.....	162
7.3	Thumb 转移指令.....	164
7.4	Thumb 软中断指令.....	166

7.5	Thumb 数据处理指令	167
7.6	Thumb 单寄存器数据传送指令	169
7.7	Thumb 多寄存器数据传送指令	171
7.8	Thumb 断点指令	172
7.9	Thumb 的实现	173
7.10	Thumb 的应用	174
7.11	例题与练习	175
第 8 章 体系结构对系统开发的支持		
8.1	ARM 存储器接口	178
8.2	AMBA 总线	185
8.3	ARM 参考外围规范	189
8.4	建立硬件系统原型的工具	191
8.5	ARM 仿真器 ARMulator	192
8.6	JTAG 边界扫描测试结构	193
8.7	ARM 调试结构	198
8.8	嵌入式跟踪	202
8.9	对信号处理的支持	204
8.10	例题与练习	209
第 9 章 ARM 处理器核		
9.1	ARM7TDMI	210
9.2	ARM8	217
9.3	ARM9TDMI	220
9.4	ARM10TDMI	223
9.5	讨 论	226
9.6	例题与练习	227
第 10 章 存储器层次		
10.1	存储器容量及速度	228
10.2	片上存储器	229
10.3	Cache	230
10.4	Cache 设计示例	235
10.5	存储器管理	240
10.6	例题与练习	243
第 11 章 体系结构对操作系统的支持		
11.1	操作系统简介	245
11.2	ARM 系统控制协处理器	248
11.3	保护单元寄存器 CP15	249
11.4	ARM 保护单元	251
11.5	CP15 MMU 寄存器	252

11.6	ARM MMU 结构	255
11.7	同 步	260
11.8	上下文切换	261
11.9	输入/输出	262
11.10	例题与练习	266
第 12 章 ARM CPU 核		
12.1	ARM710T/720T/740T	267
12.2	ARM810	272
12.3	StrongARM SA-110	275
12.4	ARM920T 和 ARM940T	282
12.5	ARM946E-S 和 ARM966E-S	285
12.6	ARM1020E	286
12.7	讨 论	289
12.8	例题与练习	291
第 13 章 嵌入式 ARM 的应用		
13.1	VLSI Ruby II 先进通信处理器	292
13.2	VLSI ISDN 用户处理器	294
13.3	OneC™ VWS22100 GSM 芯片	296
13.4	爱立信-VLSI 蓝牙基带控制器	300
13.5	ARM7500 和 ARM7500FE	303
13.6	ARM7100	306
13.7	SA-1100	310
13.8	例题与练习	313
第 14 章 AMULET 异步 ARM 处理器		
14.1	自定时设计	315
14.2	AMULET1	318
14.3	AMULET2	321
14.4	AMULET2e	323
14.5	AMULET3	326
14.6	DRACO 电信控制器	329
14.7	自定时系统的未来	334
14.8	例题与练习	335
附录 计算机逻辑		337
术 语		342
参考文献		347
索 引		349

第 1 章 处理器设计导论

本章内容综述

同多数工程任务一样,通用处理器的设计需要对很多方面进行权衡和折衷。在这一章中,将讨论处理器指令集与逻辑设计的基本原理,以及设计工程师可采用的、有助于达到设计目标的各种技术。

抽象是理解复杂计算机的基础。本章介绍计算机硬件设计师使用的抽象方法,其中最重要的是逻辑门。本章介绍了一个简单微处理器的设计,从指令集、寄存器传输级描述,一直到逻辑门设计。

精简指令集计算机(RISC, Reduced Instruction Set Computer)的思想起源于1980年斯坦福大学的一项处理器研究项目,而其中一些核心思想可以追溯到更早的计算机。在这一章中我们将介绍RISC产生的思想。这些思想也影响了ARM处理器设计。ARM处理器的设计将在第2章介绍。

随着基于计算机的便携式产品市场的快速发展,数字电路的功耗越来越重要。在本章的末尾,我们将介绍低功耗、高性能设计的原理。

1.1 处理器体系结构和组织

所有现代通用计算机都使用存储程序数字计算机的原理。存储程序的概念源于1940年普林斯顿先进学科研究所,并在Baby计算机中得到实现。该计算机于1948年在英国的曼彻斯特大学首次运行。

50年的发展导致处理器性能的急剧增长,而其价格也同样大幅度下降。这段时期虽然计算机的性能价格比迅速提高,但操作原理只变化了一点点。多数改进都是由电子技术的进步造成的。由真空管到分立的晶体管,再到由几个双极晶体管组成的集成电路(IC, Integrated Circuit),然后经过几代IC技术的发展,到今天的超大规模集成电路(VLSI, Very Large Scale Integrated),把数百万个场效应晶体管集成到同一个芯片上。随着晶体管的变小,晶体管也越来越便宜、快速和省电。在过去30年间,这种相互促进的局面支撑着计算机工业的进步,并至少在今后几年内还会继续这个趋势。

但是在过去50年内,并非所有的发展都源于电子技术的进步。有时候,在技术应用方法上的一个新见解也会作出重大的贡献。这些见解以计算机体系结构和组织方式

来描述。下面介绍一些体系结构与组织方面的术语。

计算机体系结构

计算机体系结构描述从用户角度看到的计算机。指令集、可见寄存器、存储器管理表结构和异常处理模式都是体系结构的一部分。

计算机组织

计算机组织描述用户不能看到的体系结构的实现方式。流水线结构、透明的 Cache(高速缓存)、步行表(table-walking)硬件以及转换后备缓冲(TLB, Translation Look-aside Buffer)都是计算机组织的问题。

在计算机设计的这些进步中,20 世纪 60 年代早期引入的虚拟存储器、透明的 Cache 和流水线等都曾是计算机发展的里程碑。作为里程碑之一,RISC 思想大幅度提高了计算机性能价格比。

什么是处理器

通用处理器是一个执行存储器中指令的有限状态机。系统的状态是由存储器中的数据 and 计算机本身的某些寄存器的数据定义的(参见图 1.1。以 16 进制表示的存储器地址将在 6.2 节解释)。每一条指令都规定了总状态变化的特定方式,还指定随后该执行哪一条指令。

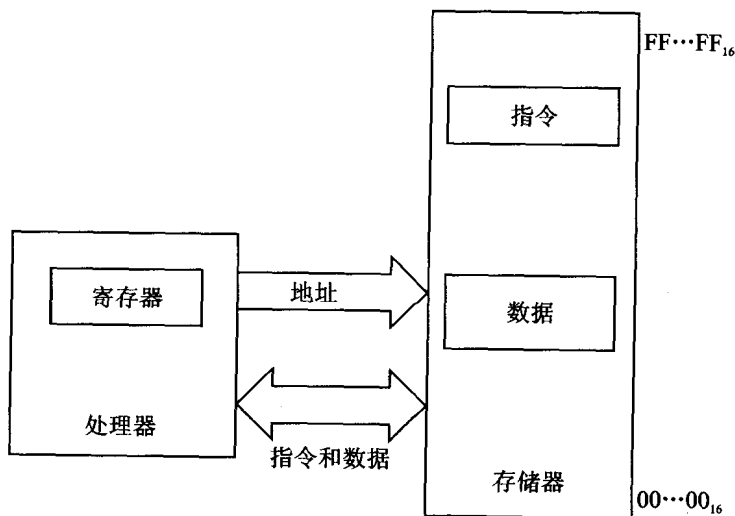


图 1.1 存储程序数字计算机中的状态

存储程序计算机

存储程序(stored-program)数字计算机把指令和数据存放在同一个存储器系统中,必要时可以将指令作为数据处理。这使处理器本身能够产生指令以供后面执行。在现在看来,在细粒度下(自修改代码)进行这种操作的程序是一种不良的形式,因为它

非常难于调试,但以粗粒度使用却是多数计算机操作的基础方式。只要计算机可以从磁盘装载新的程序(覆盖老的程序),然后执行,那么计算机就可以用这种方法改变自己的程序。

计算机应用

因为可编程,可以说存储程序数字计算机是**万能的**。也就是说,它可以承担任何能用适当算法描述的任务。这有时体现在,对于台式计算机,用户在不同时间运行不同的程序;而有时体现在,对于同一个处理器可用于不同的应用范围,而每种应用运行不同的固定程序。典型的应用是嵌入式产品,如移动电话、汽车发动机管理系统等。

1.2 硬件设计中的抽象

计算机是非常复杂的设备,它以非常高的速度工作。现在,一个微处理器可能由数百万个晶体管构成,每个晶体管每秒钟可以开关1亿次。看着一个文件在台式PC或工作站上卷屏,设想一下每一秒钟的动作是如何使用着数百亿次晶体管的开关动作。现在可以意识到,每一个这样的开关动作在某种意义上说都是有意设计的结果,其中没有随机或不可控的。实际上,在这些跳变中的单个错误就有可能使计算机崩溃而无法使用。那么,怎样才能设计这么复杂的系统,使它可以如此可靠地工作呢?

晶体管

答案的线索可能就在问题本身。我们用晶体管的观点描述了计算机的工作。但是什么是晶体管呢?晶体管是由仔细选择的、具有复杂电学性质的化学物质组成的古怪结构。这些电学性质只有参考量子力学理论才能搞清楚。然而晶体管的总体行为可以不用参考量子力学就可以描述为一系列方程式,它们给出了施加在晶体管端口上的电压与流经它的电流的关系。这些方程式是器件物理原理的本质行为的**抽象**。

逻辑门

描述晶体管行为的方程式还是相当复杂的。当一组晶体管连接到一起组成一个特殊结构时,例如图1.2所示的CMOS(互补金属氧化物半导体)与非门,对这一组晶体管的行为有一个特别简单的描述。

如果每一条输入线(A和B)保持接近于 V_{dd} 或 V_{ss} 的电压,那么输出也会依据下述规则接近 V_{dd} 或 V_{ss} ,即

- 如果A和B全都接近 V_{dd} ,则输出将会接近 V_{ss} 。
- 如果A或B(或两者都)接近 V_{ss} ,则输出将会接近 V_{dd} 。

再详细一点,我们可以规定这些规则中的“接近”是什么意思。把接近 V_{dd} 的值作为“真(true)”,而把接近 V_{ss} 的值作为“假(false)”。这样这个电路就实现了布尔逻辑的“与非”功能:

$$\text{output} = \overline{A \cdot B} \quad (1)$$