移动与嵌入式开发技术

Embedded Systems Design with Platform FPGAs, Principles and Practices

FPGA 嵌入式系统设计原理与实践

[美] Ron Sass
Andrew G. Schmidt 著
李 杨 译

MORGAN KAUFMANN
ELSEVIER
清华大学出版社

移动与嵌入式开发技术

FPGA 嵌入式系统设计原理与实践

[美] Ron Sass
Andrew G. Schmidt 著
李 杨 译

清华大学出版社
北 京

Ron Sass，Andrew G. Schmidt
Embedded Systems Design with Platform FPGAs, Principles and Practices
EISBN：978-0-12-374333-6

Authorized Simplified Chinese translation edition published by Elsevier (Singapore) Pte Ltd Press and Tsinghua University.
ISBN: 978-9-812-72891-3

北京市版权局著作权合同登记号 图字：01-2011-1166

图书在版编目(CIP)数据

FPGA 嵌入式系统设计原理与实践/(美) 萨斯(Sass, R.)，(美) 施密特(Schmidt, A.G.) 著；李杨 译.
—北京：清华大学出版社，2012.4

书名原文：Embedded Systems Design with Platform FPGAs, Principles and Practices
(移动与嵌入式开发技术)

ISBN 978-7-302-27969-3

Ⅰ.F… Ⅱ.①萨… ②施… ③李… Ⅲ.可编程逻辑器件—系统设计 Ⅳ.TP332.1

中国版本图书馆 CIP 数据核字(2012)第 011528 号

责任编辑：王　军　吴　乐
装帧设计：牛艳敏
责任校对：邱晓玉
责任印制：李红英

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
印 装 者：三河市金元印装有限公司
经　　销：全国新华书店
开　　本：185mm×260mm　　**印　　张**：20　　**字　　数**：487 千字
版　　次：2012 年 4 月第 1 版　　**印　　次**：2012 年 4 月第 1 次印刷
印　　数：1～3000
定　　价：48.00 元

产品编号：040677-01

前　言

Xilinx 公司在 1984 年推出了一款高级的可编程逻辑器件，现场可编程门阵列(Field-Programmable Gate Array，FPGA)。现在 FPGA 已经形成了数十亿的市场规模，开发的产品种类繁多，从数码相机、汽车，到驱动因特网的网络交换机；FPGA 甚至飞向了火星(Ratter, 2004)。

几乎从 FPGA 推出之始，人们就认识到了使用这些器件构建自定义的计算架构的潜力，但是迄今为止市场上的绝大多数器件是“胶合逻辑”和原型机。不过，技术的进步已经使得现代 FPGA 芯片具有非常大的容量以及各种各样的特性。这些特性的汇集——包括多个处理器、大量存储器、数百个乘法器以及高速 I/O——已经达到这样一个临界点：平台 FPGA 比以往任何时候都有望在计算系统中发挥更为突出的作用。

可在单个 FPGA 器件上部署复杂计算系统的能力，很有可能对嵌入式计算系统产生巨大的影响。虽然小型的(实际上是微型)8 位或 16 位计算系统仍然并且将会在嵌入式系统市场上占据一席之地，但过去几年的趋势表明嵌入式系统使用标准的现成的 32 位处理器也方兴未艾。这些更为高端的嵌入式系统在更高层次进行了集成，通常在(固定和制造)的芯片上已经纳入了嵌入式系统的相当一部分元件。该层次的集成益处众多，但是其中一个显著的缺点是有太多的系统架构被预先设定，这可能对于特定目标的应用程序来说并非是最优的。而替代方案，即在自定义的片上系统(System-on-a-Chip，SoC)中开发系统架构则过于昂贵，除非是用于大容量(百万单元级)的产品。当应用程序与所提供的资源相匹配时，那么一切安好。但事实往往并非如此：一些集成的资源被废置，而需要增加额外零散的硬件以弥补其不足来实现应用程序。有了平台 FPGA，工程师们可以获得集成所带来的全部好处，同时也能够根据每一个应用程序，灵活地开发一个均衡的系统架构。

平台 FPGA 配置了多种总线、各种直接通信链接、桥、I/O 元件以及琳琅满目的其他专用 IP 核，设计者可以随时自定义系统架构。例如，设计者可以使用数百个分布式块 RAM 来配置一个大型的可访问存储器，而将 RAM 作为独立缓存分布于系统各处；或者也可以设计一些两两组合的 RAM；还可以设计专用功能模块(即自定义硬核)。尽管专用集成电路(ASIC)的性能通常优于相同架构的 FPGA 实现，但是较之 ASIC 的制造，基于 FPGA 的解决方案避免了昂贵的资源、需要承担的潜在风险以及上市的时间等问题。FPGA 是一个虚拟的白板，使得工程师在对物理器件进行制造、测试以及验证之后，能够分配资源以最佳匹配应用程序。这样的灵活性提高了系统解决方案的效率，节省下来的每个离散元件都会降低成本，同时增加可靠性。

当然，这种硬件的强大灵活性也要付出代价。除了了解编译器、调试器以及其他用于基于处理器的嵌入式系统开发的传统软件工具以外，平台 FPGA 设计者还必须熟练掌握对硬件设计、综合和系统集成工具的使用。以前，架构的选择在很大程度上制约了设计者所能做出的决定——例如，如何在硬件和软件之间划分应用程序——而现在却有了充分的空间来提出解决方案。在理解特定处理器系统总线的特性之外，设计者还必须权衡多个通信机制之间的优势和劣势。对于熟悉在预先设定架构上工作的设计者来说，均衡片上元件组成的复杂网络是一个新的挑战。

这些挑战在很多方面体现了计算机工程的特点。然而直到平台 FPGA 的出现，这一挑战带来的实际问题(例如构建自定义芯片解决方案的代价问题)一直使得着手研究代价高昂。学生过去只是从教科书和概念模型中了解计算机系统架构，因此彻底地了解开发自定义计算机系统的实践环节则只局限于少数专业人士。知识和实践工具——例如，如何创建嵌入式系统的板级支持包——只是在用到的时候才开始接触。

本书的目的是向读者介绍平台 FPGA 的系统开发。它主要关注的是嵌入式系统，但是也可以作为构建自定义计算系统的通用指南。本书描述了指导平台 FPGA 系统开发的硬件、软件以及一系列设计原理的基本技术。其目标是表明如何系统地和有创造性地应用这些原理构建专用的嵌入式系统架构。同时也特别关注了免费开源软件，以提高生产率。

本书每一章节的组织结构都包括两部分：白色页面描述的是基本概念、原理以及常识；灰色页面则包括本章主要问题的技术实践，并且展示了应用于实践的概念，这包括对特定开发板和工具集的逐步详述，以便读者能够自己完成同样的步骤。本书并不试图在种类不同的开发板和工具链上示范这些概念，而是通篇只使用单个工具集(Xilinx Platform Studio、Linux 和 GNU)以及在示例中使用的开发板(Xilinx ML-510)。相信对于读者来说，完整地描述单个系统比部分地描述各种系统更有价值。

如何阅读本书

本书的编写是为了让不同的读者更快捷地找到所需要的信息。如果您是只有软件背景而没有参加过任何电子方面的课程的本科生，那么第 2 章白色页面的开始部分介绍了晶体管和 FPGA(一种固态器件)实现可编程硬件的基本知识。如果您是在嵌入式系统方面经验丰富而对 FPGA 缺乏了解的工程师，那么只需阅读灰色页面，可以跳过理论部分而直接进入实践层面，在 FPGA 上构建基于 Linux 的系统。如果您是参加过高级设计课程的学生，那么学习的重点可能是项目管理，您可能不会在此找到过多的资料，但是您会发现本书是完成项目的一个方便且实用的指南。另一方面，如果您有一个涉及尖端技术的非常具体的项目，那么本书可能没有以足够的深度涵盖这一特定的主题。不过如果您想入门，需要纵览一般性的概念，同时希望有足够的逐步指导来实现一个真实的工作系统，那么本书是您的最佳选择！

给教师的建议

本书在不同的计算机课程中可以扮演不同的角色。在很多院系中，嵌入式系统课程是高年级学生的技术选修课。该课程的主要内容是花费一个学期的时间讲述内容繁多的技术工程中的细节。本书是学习嵌入式系统内容的入门经典——因为它给学生提供了实现其项目所需的所有实践资料——教师可以根据情况在课堂上选择一些内容讲授。对于介绍嵌入式系统特定领域的课程(包括多种技术类的选修课程)来说，本书可以用于可重构计算的课程。该课程是对深入理解嵌入式系统的一个补充，其内容全面，包括小型系统以及实时问题的扩展描述。IEEE-CS/ACM 联合工作组计算机课程分组(Ironman 草案)将计算机工程基本元件作为“终极课程”。通常这在当今的课程中是作为高级设计或者毕业课程设计的。严格来说，该课程贯穿了上述课程计划中要求的独立的主题，有助于将学生的学术理论与职业生涯联系起来。FPGA 在该类课程中与其他技术相得益彰，并且经常被用到。其灵活地支持各种工程，同时这些工程中也必然包含软件和硬件元件：这是计算机工程的精髓。因此，在本书中，许多教师会为“终极课程”的学生找到非常优秀的资源。

在线资料

本书所参考的命令、脚本以及 URL 链接都可以在正文中找到。不过，为方便读者学习，出版商维护了一个网站来提供所有这些资料，包括因内容太长而无法在本书中插入的脚本。除了提供本书使用的开源软件的最新版本的链接之外，该网站还归档了其确切的(确知有效的)软件版本号。

参考文献

[1] Ratter,D. (2004). FPGAs onMars.*Xcell Journal*, Q3(50), 8–11。

致　谢

诸多人士都以各种方式对本书的编写做出了贡献。我们要感谢美国北卡罗来纳大学夏洛特分校 Reconfigurable Computing Systems 实验室的在读学生，他们阅读本书并且给出了反馈——有时反馈言简意赅。这包括 William V. Kritikos、Scott Buscemi、Bin Huang、Shanyuan Gao、Robin Jacob Pottathuparambil、Siddhartha Datta、Ashwin A. Mendon、Shweta Jain、Yamuna Rajasekhar 和 Rahul R. Sharma。其他同事和毕业学生也给予了支持；我们也要感谢 Brian Greskamp(D.E. Shaw)、SrinivasBeeravolu (Xilinx, Inc.)、ParagBeeraka (AMD Inc.)和 David Andrews(堪萨斯大学)，他们给予了本书很大的帮助。本书的很多问题和示例都来自于与他们的合作。一些外审也给出了很有价值的反馈。我们要感谢 RoyKravitz (Serveron, BPL Global 的一个分公司)、Duncan Buell(南卡罗来纳大学)、Cameron Patterson (Virginia Tech)和 Jim Turley (EmbeddedTechnology Journal)。还要感谢 Morgan-Kaufmann 的团队；特别是 Nate McFadden 和 Andre Cuello，他们协助我们按时完成手稿。另外要感谢 Xilinx 公司——他们鼓励我们开始编写本书，之后又赋予我们足够的自由来完成该书。最后，还要感谢家人——Jennie、Joseph 和 Hilary，是他们给我们时间来完成本书。你们是最棒的！

目　录

第 1 章

简　介

Em.bed——使 ••• 成为其一部分

——《韦氏在线字典》

从智能手机到医疗器械，从微波炉到汽车防抱死系统，现代嵌入式系统已经成为我们社会的一部分。为了研发产品，计算机工程师们在硬件和软件元件中普遍使用各种工具和技术来构建嵌入式系统。其中一种称为现场可编程门阵列(Field-Programmable Gate Array，FPGA)的技术变得日益重要起来。通俗地讲，可以将 FPGA 想象成一块空白板，在上面可以配置数字电路(如图 1-1 所示)。而且，在器件被制造出来并被组装成产品后，甚至在已经销售给消费者的情况下，还能够在该电路板上配置我们所需要的功能。这使得 FPGA 器件与其他的集成电路(Integrated Circuit，IC)器件有本质的区别。简而言之，一块 FPGA 器件给嵌入式系统开发人员提供了可编程的“硬件”。

近年来，FPGA 器件的角色也发生了变化。之前，这项技术最常见的应用是使用一块 FPGA 器件代替少量独立的小型或中型规模的 IC 器件，如无处不在的 7400 系列逻辑单元。随着半导体技术众所周知的发展和改进，每块 IC 芯片上晶体管的数目呈指数增加。这也显著增强了可编程逻辑的容量和功能，这对 FPGA 器件的开发大有裨益。这里所说的容量，是指可使用的等效逻辑门的个数；而功能则指器件所引入的一系列固定专用模块。本书后面还会再详尽地讨论它们。

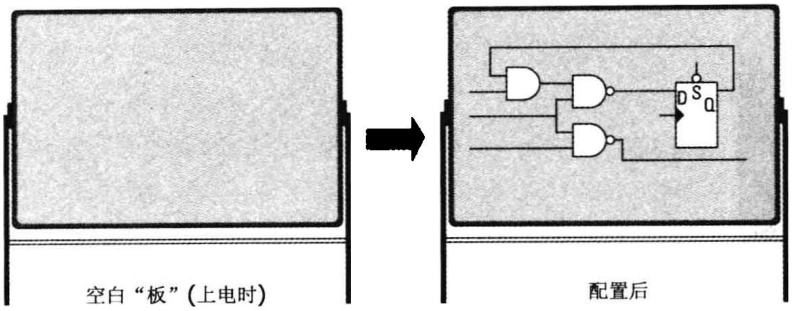

图 1-1　FPGA 宏观图；在芯片制造出来之后，可以对空白板进行配置实现数字电路

这种发展的结果就是现代 FPGA 在单一器件上能够支持处理器、总线、内存控制器、线路接口，以及数量持续增长的通用外设。虽然只是在一块 IC 芯片上，但是基于诸如 Linux 之类的操作系统的支持，这些 FPGA 器件在功能和容量上更像是一台 PC 机。此外，开源操作系统所催生的开源软件，也促进了 FPGA 设计的发展。

我们使用“平台 FPGA”这一术语来描述在单一器件上拥有足够的资源和功能以承载整个系统的 FPGA 器件。但是关于一个大型 FPGA 器件和一个平台 FPGA 器件的区别并不十分明显，因为它们没有实质的物理差异。差异只是概念性的，具体取决于开发人员如何使用 FPGA 器件。可以认为任何 FPGA 器件都能作为外设，并且在计算系统中起到支撑性作用。而相比之下，平台 FPGA 在计算系统中所起的作用则是核心性的。

总之，平台 FPGA 给嵌入式系统开发人员提供了很好的解决方案。除了能够降低芯片的使用个数外，平台 FPGA 也具有强大的灵活性。这对于系统开发来说是难能可贵的，特别是在对系统提出日益复杂而苛刻的要求的今天。然而，正是因为具有了灵活性，才使得 FPGA 技术较之传统的基于微处理器、结构化 ASIC(Application-Specific Integrated Circuits，专用集成电路)，或者其他片上系统(System-on-a-Chip，SoC)等的解决方案更有吸引力。

除了这些优势之外，FPGA 技术还带来了一系列新的挑战。之前嵌入式系统设计者所面临的最重要的问题是处理器的选择，这也制约了系统其余部分架构的设计(或者至少限制了设计选择的范围)。为了在充分利用其能力的同时节省成本，在系统开发时，更多的系统开发人员使用平台 FPGA，以便能够结合计算机工程、编程、系统分析和技术能力进行统筹设计。

本书将集中讲述这些内容，来为读者提供在平台 FPGA 上构建嵌入式系统所需的扎实完整的基础。这包括做出系统级决策所需的基本工程科学，以及部署集成的硬件和软件系统所需的实践技巧。

本书的组织有如下特点：每一章都由白色页面和灰色页面组成。白色页面更多的是强调理论概念和缓慢发展的科学。而灰色页面则是对当前流行技术的描述，以及强调常见问题的实际解决方案。

本章包含如下主要内容。

- 首先从抽象的角度了解计算设备，通过讲解与常用计算机系统的区别来定义嵌入式系统。因为要使用平台 FPGA 器件开发有特定应用的自定义计算设备，所以重新回顾硬件和软件的概念也很重要。同时，着重描述了两种各自使用的计算模型。
- 接着，介绍了嵌入式系统设计者当前所面对的特定挑战。包括产品较短的生命周期所导致的紧迫的项目开发进度、日益增加的复杂度、新的需求以及定义成功与否的常用的性能指标。这些挑战复杂交错，展现了下一代嵌入式系统将面临的重要问题。
- 最后，本章的白色页面对平台 FPGA 的特性进行了介绍并解释了为何这些器件能够满足现代嵌入式系统设计者的复杂要求。

在灰色页面展示了虚拟场景下的一个示例。我们简单描述了如何安装本书所需的软件工具，以及如何在平台 FPGA 开发板上创建常见的“Hello, World!”示例。总之，本章的目标是建立坚实的概念基础，以使用平台 FPGA 构建嵌入式系统以及使读者了解如何使用开发工具。

1.1 嵌入式系统

简而言之，一个嵌入式系统就是一台专用计算机。一台计算机器(或计算机)经常被建模为一个拥有输入、输出和计算单元的系统。它存在于某种类型的环境中，该环境提供能量来驱动机器。除了处理信息之外，计算机还会发热并且将热量散发到环境中。其结构如图 1-2 所示(该图比较抽象，其具体含义将在下一节进行阐述)。

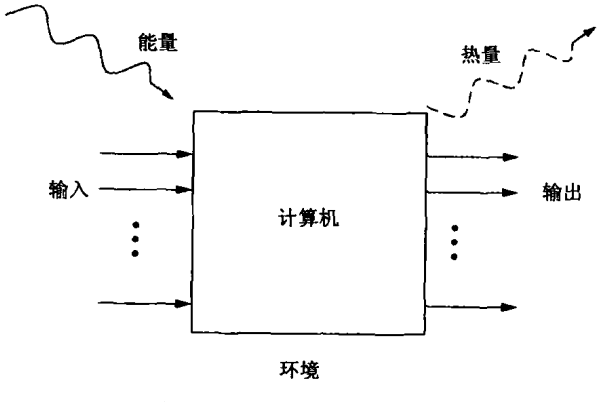

图 1-2 计算系统的抽象图

当机器被使用时，我们称之为执行，该时间段称为“运行时”；如果没有执行，则称为“离线”。输入来自运行环境，其确定了输出。输出由计算机传递给运行环境。输入和输出一般是物理信号，根据待处理问题的背景而被配置。这也是开发计算机的原因：解决问题。

一种众所周知的计算机就是个人电脑(或台式机)。除此之外，还有很多其他类型的计算机。计算尺和算盘属于老式的计算机，可以进行代数运算。之后它们被电子计算器所代替。这属于电子计算机。

1.1.1 嵌入式系统和通用计算机

根据计算机的使用方式，可以将其分为嵌入式计算机和通用计算机两类。嵌入式计算系统(或简而言之，嵌入式系统)作为一种计算机往往只是大型产品的一部分，其目的也仅仅是为了支持该产品。换言之，它是用于某一特定目的的计算机。根据图 1-2 的计算机抽象模型，嵌入式计算系统的环境就是它作为其一部分的产品。最终用户并不直接跟嵌入式系统交互，或者仅仅只是与其有限的接口交互，例如遥控器。即使计算机能够处理更多内容，嵌入式系统往往也只在封装它的产品中扮演有限的角色，比如控制产品的运转状态。

从我们日常用到的各种各样的电子产品(例如 DVD 播放器、MP3 播放器和游戏机)到节能电冰箱和宾馆的门卡激活系统，嵌入式系统无处不在，且与现代生活息息相关。甚至在更大型的重型推土设备上面也有数以百计的传感器和制动器与一个或多个嵌入式系统相连接。

相比之下，通用计算系统自身即为产品，最终用户直接与其进行交互。从另一个角度说，最终用户明确知道自己所购买的产品就是一台计算机。通用系统的输入和输出相对较少，规格标准却十分统一。这包括外设，例如键盘、鼠标、网络连接设备、视频监视器和打印机。嵌入式系统一般也有这些标准外设，但却有更多的专用设备。例如一个嵌入式系统可能从各种专用传感器(例如加速计、温度探测器、磁强计、按钮触点开关等)中接收数据，也可能以各种方式输出信号(例如通过灯具或 LED、制动器、TTL 电信号、液晶显示器等)。这些输入和输出设备在通用计算机中很难找到。此外，在嵌入式系统中输入和输出信号的编码方式也是跟设备相关的。而通用计算机使用的是标准编码。例如，鼠标的移动操作的信号一般通过低速串行通信传输，不同厂家生产的鼠标可以互用。另一个例子是，键盘上的每一个按键采用固定的标准 ASCII 编码。嵌入式系统也会采用这些标准，但它也很可能收到以脉冲频率形式编码的输入信号。从电信号的形式的角度来讲，通用计算机和嵌入式系统是相似的，但是它们的信号被赋予实际意义的方式却大不一样。

最后，嵌入式系统如此确切的定义可能与人们所认为的“较大的产品”很难达成一致。比如，一台掌上电脑是嵌入式系统吗？如果是的话，封装它的产品是什么？它并没有被用于控制任何东西。另外，计算系统也对用户可见：用户可以下载通用的应用程序。根据所描述的特征，这不是一个嵌入式系统。但掌上电脑却与嵌入式系统有着诸多共同之处：例如，跟嵌入式系统一样，掌上电脑对尺寸、重量和额定功率有严格的要求。因此，一些人倾向于称其为嵌入式系统。如果给系统加入一款移动电话应用程序，那么大部分人会同意这的确是一个嵌入式系统。通过增加一些特定的输入和输出设备，一个通用系统可以被看做是嵌入式系统。因此通用式和嵌入式系统之间的界限并非所认为的那样泾渭分明。

通用系统力求均衡：在一些地方性能会做出折中，以便更好地执行更广泛的任务。更为重要的是，通用计算机的设计是为了使其在常见情况下速度保持很快。嵌入式系统一般用于单一领域，因此设计者们还能够利用该领域更多的精确信息，在产品开发过程中这些信息可以被用来提高性能。这经常会实现性能实质上的提升。

例如，一台通用计算机可能处理家庭摄像机中的50 000帧录像，之后该同一台计算机还会在提交万维网内容时处理电子表格的计算。对于单个用户，这台计算机需要很好地处理这些函数(以及其他诸多任务)。至于其他问题，比如能耗，只要合情合理就可以了。然而嵌入式系统的设计者或许知道该系统永远也没必要处理电子表格的计算，因此工程师们可以据此更好地利用现有的可使用资源。

大部分的计算机课程针对的都是通用计算系统。操作系统、编程语言以及计算机架构都是基于通用计算机进行教授的。这是恰当的，因为其(不管是不是嵌入式系统)许多原理是相同的。可是，既然嵌入式计算系统在我们的日常生活中无所不在，因此有必要强调嵌入式系统独有的特性，并且学习这些特性。

1.1.2 硬件、软件和FPGA

通用计算系统和嵌入式计算系统的一个不同之处就是，嵌入式系统常常需要特定用途的硬件和软件。我们有必要事先定义这些术语。另外，平台FPGA模糊了硬件和软件之间的界限。在我们进一步了解FPGA技术之前，首先重新讨论一下硬件和软件的定义。

硬件是指计算机的物理实现，一般是一些电子电路的集合，也有可能是一些机械元件。简单地说，它是可感知的实物，例如一台由逻辑门(与、或、非门)组成的组合电路实现的集成在电路板上的计算机。硬件是可见的，其设计占据了有形的空间。硬件的另一特点就是所有元件同时工作。如果输入发生了变化，那么电路中的改变会以一种可以预知但并不一定同步的方式遍及整个系统。

然而软件却是信息，并且在物理世界中不显现自己。软件是描述机器的行为的一种规范，一般以一定的编程语言(例如C、MATLAB或者Java)进行编写。这一期望的机器行为的表示称为程序。虽然可以将程序打印到纸上，但这只是该程序存在于物理世界的一个表征。软件是打印内容所要表达的信息，其本质上是无形的(注意，编写软件的人称为程序员，而这一行为称为编程)。

这些传统的硬件和软件定义多年来无可厚非，但是随着FPGA和其他可编程逻辑器件的出现，这些定义以及何谓硬件、何谓软件的区分变得不太明确了。

1.1.3 执行模型

嵌入式系统设计最大的挑战就是在设计中既要兼顾硬件部分也要兼顾软件部分。这就需要从本质上强调其对应的不同的执行模型。

一些编程范式存在于高级语言中(如函数型或面向对象型程序设计语言等)。然而，从机器语言的层次来看，所有的商用处理器采用的都是同一种带有可寻址内存的命令式的、取-执行指令的模型。在软件中，该顺序执行模型将操作串行化，消除任何直接的并行操作。因而操作依次完成，这减轻了软件程序员执行程序时艰难的同步工作。相比之下，硬件理所当然地采用并行，计算机工程师们必须在其设计中显式包含同步。顺序、隐式顺序操作的概念与不加限制的计算模型在FPGA设计中尤为重要，也将贯穿全书始终进行讨论。

需要注意的是，平台 FPGA 系统中一些作为“硬件”的元件实际上是以软件的形式写出的。即该元件是用来规范器件是如何被配置的。实际的物理硬件是 FPGA 器件自身。因此，为了代替前面所给出的传统意义上的硬件的定义，我们将基于执行的模型来区分系统中的硬件和软件元件。之前我们说过软件的特点是顺序执行，而硬件执行模型是非顺序的。换言之，软件是指藉由一块处理器执行的规范，而硬件是指藉由 FPGA 架构执行的规范。并且有时，从更广泛的意义上讲，硬件是指计算系统的物理元件。为了使这一重要的区别更加具体，考虑如下的计算：

$$R0+R1+R2 \rightarrow Acc$$

假定这 4 个变量为寄存器。该计算以顺序模型的实现如图 1-3 所示。它表明了传统的取-执行指令的计算机。虚线框强调了该模型主要的机制。ALU 电路时分复用顺序执行每一步加法。电路的大部分用于译码和控制节拍。这通常被称为冯·诺依曼存储程序计算模型。我们也将继续将其作为顺序执行模型。相比之下，非顺序执行模型的形式更为广泛，但却与顺序执行有明显的差别。它没有通用的控制器来控制操作的顺序处理，也没有显式的已经命名且已固定的电路进行时分复用。该计算以并行模型的实现如图 1-4 所示。这通常被称为执行的“数据流模型”，因为它是数据流程图(在本书中我们广义使用该词汇；在 20 世纪 80 年代和 90 年代研究的数据流架构包含了比这更广泛的细节)的直接实现。

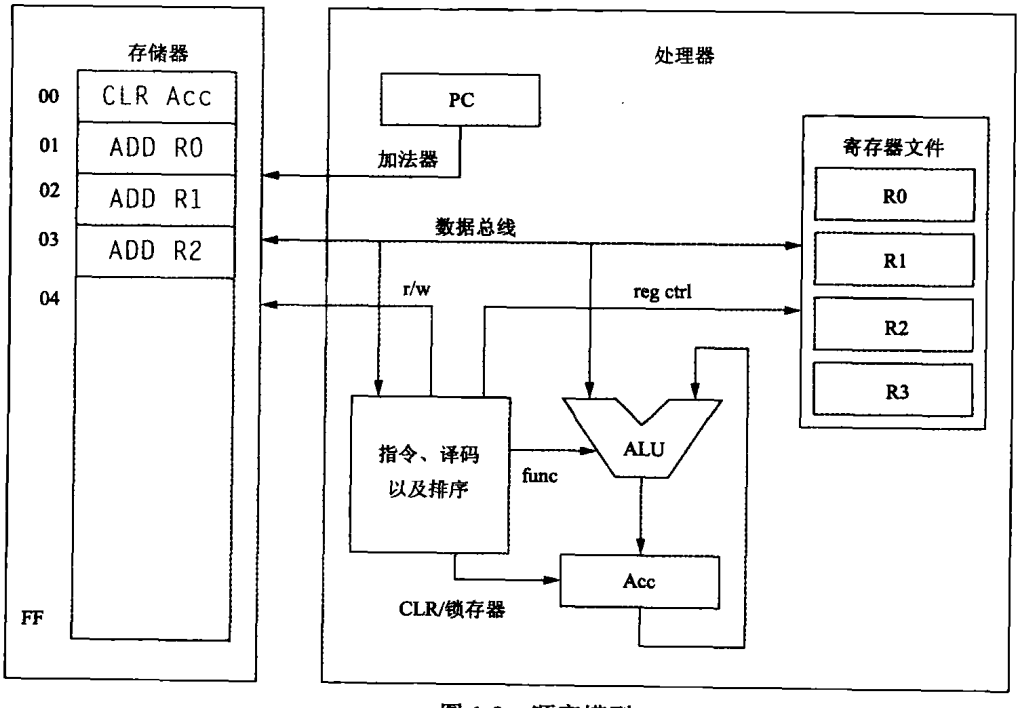

图 1-3　顺序模型

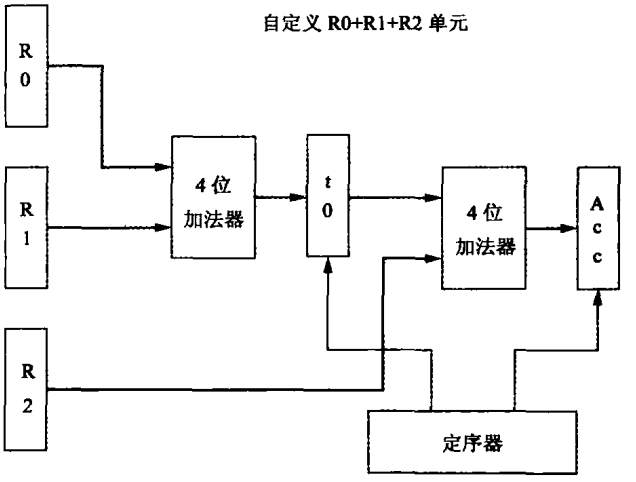

图 1-4 非顺序模型

为了说明这两个计算模型是如何工作的，考虑其随时间而进行的操作。在图 1-5 中，时间从顶至底推进。虚线描绘的每个“时隙”说明了单个时钟周期内的活动。需要注意的重要一点是并不能通过对比时钟周期数来确定这两种计算模型速度的快慢——它们各自的时钟频率是不一样的。一个典型 FPGA 的时钟周期要比使用相同技术实现的处理器的时钟周期长 5~10 倍。同时现代处理器通常同时操作多个指令，这对其性能来说也是至关重要的。

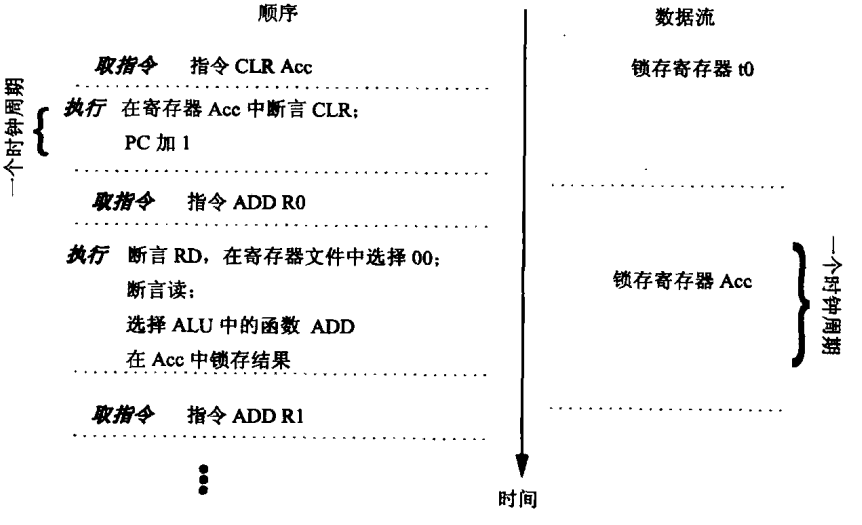

图 1-5 顺序模型和数据流模型的逐时钟周期操作

至此，我们可以说处理器是实现顺序执行模型的硬件，而软件运行于其上(注意，中央处理器(Central Processing Unit, CPU)和微处理器是处理器常见的同义词)。硬件是我们用于配置 FPGA 而不使用顺序执行模型的规范。