32位微处理机

上海交通大学出版社

32位微处理机

H·J·Mitchell 编著

金树福　等译

白英彩　　校

上海交通大学出版社

内 容 简 介

本书译自H.J.Mitchell所编著的《32-Bit Microprocessors》一书，书中主要阐述了当前国际上流行的32位微处理机的结构、组成、工作原理及其先进设计技术。

全书由RISC技术、WE32100、Inmos算元 Intel's 80386、Motorola 68020和Z80000等七章组成，内容新颖，数据资料翔实，理论和实际并重，对我国从事开发32位微机的广大读者将有所裨益。

32 位微处理机

上海交通大学出版社出版

（淮海中路1984弄19号）

上海崇明裕安晨光印刷厂印装

开本787×1092毫米 1/16 印张9.5 字数 237,000

1988年8月第 1 版 1988年8月第 1 次印刷

印数：1—2,000

ISBN7-313-00328-5/TP36

内部发行 **成本定价：4.80元**

序　　言

32位微处理器的品种在日益增多，本书旨在详细介绍几种典型的新型微处理器。

编写任何一书，例如本书，取决定作用的是编写的意图，有时不但要安排好预先的材料，而且还要努力编排好介绍这些内容的章节。由于一本书决不可能包罗万象，介绍所有新的器件，因此不仅对本书所收论文的数量必须有所限制，而且对所收论文的篇幅必须加以限制。尽管本书不得不忍痛割爱，省略某些内容，但是本书作了全面权衡提供一个较为客观的评价：目前的观点怎样，正在出现的观点怎样，而将来可能产生的观点又是什么。

本书旨在详细介绍几种典型的32位微处理器的体系结构和操作原理，所介绍的内容适合系统设计师、工程师以及研究生。当今人们不能孤立地来研究一种微处理器，因为它仅仅是系统中的一个部件。微处理器必须与有关的硬件、软件以及开发一个系统所必需的工具一起来进行分析研究。

第二章讨论简化指令系统(RISC)技术。目前，RISC 已从研究阶段转向工业生产，随着使用 RISC 技术的微处理器和系统的出现，对于这一重要学科领域进行综述将是适宜的。

以后几章主要阐述几种32位微处理机器件，除了介绍 Inmos Transputer 这一章外，其余的材料都是由微处理器厂家提供的。

H.M.

目　录

第一章 绪 论

微处理器问世以来，对于我们这些已从事微处理器工作多年的人来讲，32位微处理器已进入了一个较为稳定的发展阶段。这些新的微处理器都已经过深入的研究和精心设计。与七十年代初期的情况不同，那时各种微处理器象雨后春笋发展非常迅速，因此用户不得不冒很大的风险，万一选择不当，就会白白浪费大量的资金。

在七十年代，由于仪器设备市场激烈竞争，因而促进了微处理器的广泛应用。目前，从工程工作站一直到实时信号处理复杂科研领域，32位微处理器正在开拓新的市场，其中处理机阵列将会成为最常见的机器形式。计算机制造商也正在开发和出售他们自己的用于低档产品的32位微处理器，甚至把 Micro VAXII 也称作为低档产品。

大部分公认的微处理器制造商正在把他们的16位微处理器的体系结构移植到32位上，同时也增加了一些功能。采取这种做法的主要依据是保护用户的软件投资。这在某些应用场合是非常重要的。然而，对于许多新的应用来讲，这种考虑就没有必要了。我们已经用 Intel 8088 和 80286 作 IBM PC 的核心部件，而下一个合乎逻辑的步骤将是转向 Intel 80386；然而，最近发表的研究报告提议换一种方法试试。同样 Zilog 公司也已开始计划用 TA&T公司的 31000 而不是用本公司的 Z80000 来生产下一代的 Unix 系统。一些半导体制造商正在设法保护其在开发系统软件上的投资而不是用户的投资的做法将可能引起争议，而软件移植之所以重要是因为软件主要是在小型机市场。虽然DEC公司许多年前就认识到这一点，而且生产了PDP11机和现在的VAX机。但其他一些厂商(如著名的 Hewlett-Packard 公司)也正在急起直追，Hewlett-Packard(简称HP)公司正准备把他们的分散的产品，从工作站级到主机级都统一到一种通用体系结构上来。

在明确统一产品系列的概念之后，引起 HP 公司兴趣的是，他们企图采用一种简化指令系统计算机(RISC)作为他们未来产品的基本体系结构。HP 将是在他们新产品中采用这种方法来生产第一台主机，半导体制造商也试图将这个方法用于他们的新产品中。然而 IBM 和其他公司也在谨慎地采用 RISC 结构来研制他们的工作站。另一方面，正如我们现在所看到的，英国 Inmos 公司的 Transputer 在其提高计算机运算能力方面显示了 RISC 的潜力，另外，Fairchild、Acorn 和其他公司也生产出了具有 RISC 结构的器件。毫无疑问，通过应用 RISC和CISC 的争论，究竟孰是孰非不久就可见分晓。

本书中用得最多的术语也许是 Unix。在过去几年里，Unix 已作为操作系统应用在大部份大、中、小型计算机系统中。不管人们是喜欢还是反对，今后 Unix 还会存在多年。由于大多数半导体制造商认识到这一点，所以在其生产的器件中都设计了一些支持 Unix 的功能。正如所料，AT&T器件的设计的目的是能有效地支持C程序设计语言。这种器件包含了一些改进 Unix 性能的基本属性。目前市场上出现的少量32位机中，Unix 操作系统已经运用 Motorola 公司、国家半导体公司和 AT&T 公司机器上。

虽然 Unix 目前已成为通用计算机系统的标准操作系统之一，但是尚未成为嵌入式计算机系统的标准，尽管 Intel 公司的 RMX 已成为最广泛应用操作系统之一，但是我们也

许逐渐看到它正向使用 Ada 过渡。然而除非早期的编译系统的质量和性能得到改善，否则，许多用户会反对离开他们已熟悉的语言和操作系统。还有 Ada 语言及其支援的工具很复杂，这也可能阻止许多团体使用 Ada 从事开发工作。

在最复杂的系统中使用多处理器正变得越来越普遍。由于许多厂商都认识到了这一点，所以他们已在其产品中为多处理器应用设计了一些支持功能。特别是 Inmos 和 Zilog 两家公司已在他们的产品中为多处理器应用设计了丰富的支持功能，在这些支持功能中最新的均可在 Inmos Transputer 中找到，这种器件带有四个高速串行链路。不管是使用真正的多处理器还是使用协处理器，为了确保用户能方便使用，必须把那些基本的连接部分("hooks")设计进器件中，目前最普通的协处理器是用于执行算术或浮点运算，显然这些协处理器的运算速度还比不上小型计算机中的算术运算处理器，但是可以预料，某些性能将得到改进。

随着用户对每项产品的性能和存储器容量的要求不断增加，存储器系统的设计和管理已成为一门复杂的研究课题。虽然，所有的 32 位微处器都需要提供存储器管理支持，但是目前系统设计师必须考虑在他们系统中要不要使用外部高速缓存。随着大多数新型微处理器中都采用小型高速缓存，高性能系统或大容量存储系统，可能都得使用外部高速缓存。为使系统在任何地方都能全速运行，设计人员只有选用昂贵的高速静态存储器或者采用动态存储器对前端较慢的存储器实现高速缓存。可以设想，大部分 32 位微处理器将被应用在规模大而技术复杂的工程中，这些工程需要精心设计、仔细规划，提供技术物资保证，以便能达到预定的目标。任何大中型工程项目都需要丰富的综合支持工具，这些支持工具必须包括管理工具、办公室自动化工具、系统和软件设计辅助工具以及某些通用支持工具，例如代码管理系统。虽然这些支持工具正变得日益普遍，但是最需要做的将是软件设计和控制处理这两方面工作了。在军事工程项目中，软件的可靠性是至关重要的目标，因此整个项目设计过程必须精心管理。

最近至少有四种器件可供使用，还有一些器件可望不久即可应市。可以认为 32 位微处理器的时代已经来临。用户、微机市场和半导体制造商将决定微机工业的发展前景，然而下一步发展究竟是增加功能、增加字长，采用 RISC 技术还是并行技术，只有时间会告诉我们。

（黄德堃　译）

第二章　RISC 剖析

H. M. BRINKLEY SPRUNT, E. DOUGLAS JENSEN,
CHARLES Y. HITCHCOCK III, AND ROBERT P. COLWELL
Computer Science Dept., Dept. of Electrical and Computer Eng., Carnegie-Mellon University, Pittsburgh, Pa., USA

2.1 导论

一个高明的计算机设计师在建立一个新的计算机结构时，必须作出大量的设计决策。这些决策是以设计师的经验和系统需求等因素为基础的。目前，缩简指令系统计算机(RISC)已经引起人们的普遍关注，它们的倡导者确信RISC技术为计算机设计提供了一个最好的方法。与复杂指令系统计算机(CISC)相比，采用RISC技术更有利于设计高速机器，而且成本更低廉。尽管RISC思想很有价值，但RISC设计技术的实质尚未被人们很好地认识。RISC设计思想对长期以来主宰计算机设计领域的许多被视为理所当然的做法提出了挑战。目前，有关RISC的研究文章常常不能恰如其分地指出RISC技术的许多重要特征，并且会使读者产生误解。例如：RISC和CISC机器的比较结果就很难解释，因为它的性能报告实际上是以单片特性为基础作出的，而根本没有将这些特性增益投入到一定的机械结构中去测试。同样，对具有截然不同的设计目标的机器（如：VAX和RISC Ⅰ）进行比较，如果忽视它们的不同之处，就会使读者误入歧途。本章的目的是提供一个更有价值的RISC/CISC的比较分析方法。本报告是以Carnegie-Mellon大学(CMU)最近的实验结果为基础写成的。

不可能通过本章的论述来全面的纠正人们对RISC和CISC可能产生的所有误解或曲解，只是重点论述计算机设计的核心问题。例如：许多RISC的文章都致力于谈论计算机指令系统的大小及其复杂程度，这很有可能引起读者误解。指令系统的设计固然重要，但这些设计思想(RISC或者CISC)不能象宗教信仰那样墨守成规。我们应当着眼于系统结构中指定系统功能实现的一般性问题的讨论。这样，就不仅包括了指令系统的论述(通常，CISC机器在RISC机器更低的系统水平上设置系统功能)，同时也考虑到了其它的设计因素，如寄存器组、协处理器和缓存器等。

RISC技术的研究已经超出了指令系统的范围，即使在指令系统内部同样存在着模糊的限制。仔细阅读专门讨论RISC技术的文章可以得到一点启示，即RISC方法在何处可能失效。用简单机器获得最佳效果的说法的确非常诱人，但我们知道"每一个复杂的问题都有一个简单的解决办法"这种说法是错误的。RISC思想是正确的，但对它的肤浅认识却会导致错误。RISC思想包含许多不明确的涵义。RISC的研究工作有助于着眼于对计算机系统结构中的一些重要问题的研究。RISC的倡导者还常常忽视其应用、结构和实现问题的研究，而恰恰在这几个方面，它们的观点可能是需要纠正的。对RISC设计风格和

论点的仔细评价以及对 CISC 机器提出的质疑可以使人们深刻地理解硬件/软件之间的分工、计算机的性能、VLSI 对处理器设计的影响以及许多其它问题。

在这一章中，我们将简单回顾计算机设计的发展情况以及导致RISC设计思想的 RISC 研究工程，然后讨论 RISC 技术对计算机设计的主流所提出的一些重大挑战，并且介绍几台商业 RISC 机器。在讲述了背景资料和几台 RISC 样机之后，接着讨论我们认为伴随着 RISC 所产生的几个主要的混淆和曲解的概念。本章所讨论的内容基于 CMU 研究工程的总结报告，并指出了对 RISC/CISC 的一些争论的焦点及其分析。

2.2 RISC 的起源

要认识缩减指令系统计算机的机能，很重要的一点就是要把握指令系统设计的发展趋向。如果采用最早的数字电子计算机的设计思想，指令系统将趋于更加庞大和复杂。1984 年的 MARK-1 机器只有七条一般复杂程度的指令，而目前的象 VAX 那样的机器却拥有三百多条指令，而且这些指令可以是相当复杂的，如：将一个元素插入到双链表中或求任意阶浮点多项式的值等等。因此，VAX 的任何一个高性能的实现必须依赖于象流水线、预取指令和高速缓存[31]那样的复杂执行技术。

这种由小而简单的指令系统发展到大而复杂的指令系统的形式在单片处理器中最为突出。因为它们仅仅在过去的十年中才得到了发展。以Motorola公司的MC6800 和 MC68020 为例作一个比较，可以发现后者增加了七个寻址方式，指令系统也增加了两倍以上，而且还增加了一些新的功能，如：支持一个指令高速缓存和协处理器等。然而，指令和寻址方式的增加同时也增加了机器的复杂程度。

CISC 的这一发展趋势受很多因素的推动。包括以下几个方面：

- 计算机系列的新机型总是要和现有机型中的级别较高的机器兼容，其结果就是新机器的性能提升和递增。这就使得新机器以外来大型软件为基础而产生新的部件。
- 许多计算机的设计者都希望减少程序与计算机指令系统之间的“语义差异”。通过增加与程序所需语义相近的指令的方法，这些程序员希望以提供一个更便于使用的可编程机器[5]来减少软件成本。注意，这些指令由于其语义水准较高，因此也就比较复杂。(但是，通常这些多语义水准的指令还不能实际满足一个特定语言的要求[33])。
- 在开发快速机器时，设计者们不断地将一些功能从软件移植到微代码上或从微代码移植到硬件上。但是，通常这一做法是在忽略了增加系统结构特性对设计实现的反作用的情况下进行的。例如：增加一条指令就需要一个额外的译码逻辑，整个机器的执行速度也有可能因此而减慢(这叫做“n+1”现象[18])。
- 设计者们在处理大型系统结构的固有复杂性时增加了一些工具和手段。目前的 CAD 工具和微指令支持软件就是个例子。

微指令是助长复杂指令系统设计技术的一个极为诱人的例子，它以下面两种方式工作。第一，它提供了一个作为有效生成和修改用来控制执行计算机中数值计算指令和复杂指令的算法的结构化工具。第二，CISC 特性的快速增长是由微指令存储器的特性所推进的。在没有用尽微指令空间的机器中增加一种寻址方式或一条隐含指令还是比较容易的。

跟踪 CISC 机器的指令可以发现，大多数有效的指令在一定给定的计算环境中并没有被频繁地使用着。这一事实隐含了许多内容，本章的后面将对此加以讨论。正是在观察了系统/360的特性之后，IBM公司的 John Cocke 在七十年代初期便提出了摒弃传统计算机模

式的设想，从而开始了 RISC 的早期研究工程，创建性的以研究小组所在的办公楼编号(801)而命名的，801 工程就是建立在机关相关和协作设计原则的基础上的。有关该项工程的研究报告出版的很少，所发表的文章只是论述了原则性的和相关性的研究工作[27]。

801 系统设计的三个主要思想是：

- 提供一个硬件环境，以便在一个机器周期内执行它的一条指令，并且选择这些指令作为编译程序的一个良好目标。这表明频繁执行基本指令无需付出管理复杂指令(可能包括扩展译码和多周期操作)的执行时特性损失的代价，因为复杂指令的执行需要附加硬件。
- 设计一个分级存储系统，因此运算部件通常就不必等待存储器存取。如果运算部件在执行操作时要不断地停下来以等待指令或者数据，那么就失去了简单、快速的运算特性。
- 整个系统的设计以使用801编译程序为基础。程序员可以完全依靠编译程序进行良好的程序开发和管理，并且在系统的最佳特性固定时或者系统操作不能用源代码指定时只采用汇编语言编程。

这些思想产生了一个基于上述三个设计原则的指令系统结构。按照 Radin 的说法[27]，指令系统是运行时操作的那些指令。它们是：

- 不能转移到编译时使用；
- 不能被高级语言的编译程序所产生的目标码有效地执行；
- 在随机逻辑下执行比在等价的软件指令序列下执行更为有效。

801系统由于其硬件和软件的紧密结合产生了极高的性能。由简化愿望驱使硬件实现上以直接布线控制和单周期指令执行的特征。所有的访内操作都限于装载(Load)，存储(Store)指令，将指令高速缓存和数据缓存加以分开，以便允许指令和操作数并行存取。801系统的 CPU 具有 32 个32位通用寄存器，所有操作都在寄存器之间进行。编译程序使用了许多优化策略，其中包括一个功能强大的寄存器配置方案和全局优化方法。编译程序还承担了所有系统中的访问检测和保护的责任。

在七十年代中期 801 系统的一些基本思想传到了美国的西海岸。在加利福尼亚的伯克莱分校，基于这些思想开展了一系列 RISC 工程，并纳入研究生课程从而生产出了 RISC-Ⅰ，RISC-Ⅱ和 SOAR 以及大量的易于实现这些设计的 CAD 工具。这些课程为计算机的性能评价、CAD 和计算机实现的研究工作奠定了基础。

象 801 系统一样，RISC-Ⅰ处理器[24]是一台直接布线控制的 Load/Store机器(即：数据只能在寄存器中进行操作，而且只能由 Load 和 Store 指令访问内存)，它的大多数指令都是在一个机器周期内执行的。其31条指令的每一条都是一个简单的32位字，并且使用同样的译码格式。RISC-Ⅰ的一个特征就是它拥有庞大的寄存器数量(超过了一百个)，这些寄存器用于形成一系列重迭的寄存器组(MRS)。这个特征使得RISC-Ⅰ的处理器存储器总线上的数据流通量变得小。

希望MRS机器产生有益的性能的想法是可取的，因为面向过程的高级语言(HLL)使用寄存器来专门存放过程的指定信息。无论什么时候执行一个过程调用，都必须将该信息存放起来，通常是将它放在内存堆栈中，并且在返回过程时恢复这些信息。这些操作由于它们的内在数据传送要求而非常费时。RISC-Ⅰ使用它的多重寄存器组来减少对这些寄存器

的频繁存取，同时它也吸取了用于参数传递的寄存器组重迭的优点，甚至能够减少以内存作为传送介质(例如，通过堆栈)的称为参数传递方式所需的存储器读写[14]。

RISC-Ⅰ的寄存器堆具有八个重迭窗口，它们由 138 个 32 位寄存器组成(见图2.1)。在每一个窗口内，有六个寄存器与前一个窗口重迭(用以输入参数和输出结果)，另六个寄存器与后一个窗口重迭(用以输入结果和输出参数)。在任何过程中，这些窗口只有一个可以实际存取。一个过程调用使用一指针增量将当前的窗口变成下一个窗口，六个输出参数寄存器就提供了六个被调用过程的输入参数。同样，一个过程返回就转换到前一个窗口，输出结果寄存器就变成了被调用过程和输入结果寄存器。假设六个 32 位寄存器可以容纳全部参数，那么一个过程调用实际上不包含信息转移(只有窗口指针被修正过)。注意，由于寄存器窗口的上溢与下溢[24]芯片上的有限资源会限制实际节省。在一个寄存器组中有 10 个全局寄存器，它们总是可以存取的。

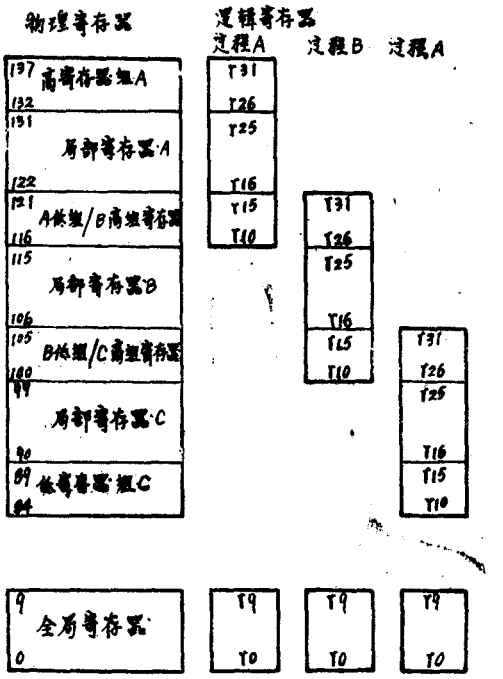

图 2.1 RISC-Ⅰ的重迭寄存器组

伯克莱在其描述 RISC-Ⅰ和 RISC-Ⅱ处理的文章中宣称他们的原始设计计划对 VAX 和 68000[24,26]那样的 CISC 机器作了大量的性能改进(二倍到四倍)。许多现象表明：这些改进结果以及获得这些结果的方法，其中最主要的是缩减指令系统性能影响和重迭寄存器性能影响有不可分隔性，而后者可以结合到任一通用寄存器的机器中，因为这些性能因素不是独立评价的，所以有关缩减性能主张是不能说明问题的，这一点将在本章的 2.6 小节中解释。

在伯克莱推出的第一台 RISC-Ⅰ后不久，一个叫做 MIPS[17] 的微处理器(无互锁的流水线处理器)在斯坦福大学研制成功。MIPS 微处理的主要目标是高性能执行编译代码。

为了达到该目标所采用的基本方法是用软件来控制 CPU 的内部并行处理。MIPS提供给用户的是释码量最少的微处理器，而不是一个直接指令系统的有效编码(RISC-Ⅰ的设计者

表 2.1　　MIPS指令系统

操　作	操 作 数	注　　释
Add	Src1, Src2, dest	整型数加
And	Src1, Src2, dest	逻辑与
Ic	Src1, Src2, dest	插入字节
Or	Src1, Src2, dest	逻辑或
Rlc	Src1, Src2, Src3, dest	循环组合
Rol	Src1, Src2, dest	循环
Sll	Src1, Src2, dest	逻辑左移
Sra	Src1, Src2, dest	算术右移
Srl	Src1, Src2, dest	逻辑右移
Sub	Src1, Src2, dest	整型数减
Subr	Src1, Src2, dest	反向整数减
Xc	Src1, Src2, dest	抽取字节
Xor	Src1, Src2, dest	逻辑异或
Ld	A[Src], dst	传送基值
Ld	[Src1+Src2], dst	传送变址基值
Ld	[Src1》Src2], dst	传送移位基值
Ld	A, dst	直接传送
Ld	I, dst	传送立即数
MOV	stc, dst	传送立即数
St	Src1, A[Src]	存储基值
St	Src1, [Src2+Src3]	存储变址基值
St	Src1, [Src2》Src3]	存储移位基值
St	Src, A	直接存储
Bra	dst	无条件相对转移
Bra	Cond, Src1, Src2, dst	条件转移
Jmp	dst	无条件直接转移
Jmp	A[src]	无条件基值转移
Jmp	(A[src])	无条件间接转移
Trap	Cond, Src1, Src2	陷阱指令
SavePC	A	陷阱或中断后，存储多级PC
Set	Cond, Src, dst	建立条件

是这样做的)。由此而产生的微处理器是一个直接布线的 Load/Store 机器，它的指令和微指令很接近(见表2.1)。

虽然 MIPS微处理器是流水线化的，但在指令系统上采用微处理器进行并行处理所造成的唯一结果是流水线中的设备管理和数据管理不采用流水线互锁硬件。该功能必须由软件来实现(即在编译时实现)。这样的方法与其他注重使用流水线技术的机器(如，IBM 360/91 型机器)所采用的方法截然不同。IBM 360/91是在运行时使用指定硬件来实现流水线隔行互锁功能的。MIPS的设计者打算将该功能搬到编译时来实现，以提供一个更有效的并具有最小执行周期的运算部件。

MIPS 微处理器的其它几个设计思想也都体现了以最小硬件复杂程度来实现最高的性能要求。MIPS提供了两个存储器接口：一个用于指令；另一个用于数据。这样，它的存储器带宽就是一个存储器接口允许的最大存储器带宽的一倍(这是 MIPS 的一个重要特征，因为如果没有足够的存储器带宽，MIPS 的快速微处理器就要停止工作)。MIPS 是一个字寻址机器，它减少了由字节寻址结构带来的硬件复杂程度。在MIPS中，指令的长短是一致的而且所有指令的执行都化费同样的时间。这个特征加上条件码的缺省特征简化了中断和页面故障的处理。任何可能产生页面故障的 MIPS 指令，在其故障没有得到排除之前，存储器的内容保证不会被修改，因此指令的重新启动无需后备的内部状态。

801，RISC- I 和 MIPS 这三台机器是 RISC 研究的核心，本文中所引用的文章详细地介绍了它们的原理，本文也对它们的某些特征进行了全面的论述。本文的重点是讨论上面三台机器，同时也考虑和讨论其它工业上和学术上的 RISC 机器。

对 RISC 技术的综合定义至今尚未找到，但这又是非常必要的。下面提出的六个协同因素可以说是 RISC 技术的根本之处。它们是：

1. 单周期操作：简化快速执行支配计算机指令流的简单功能的操作并有助于解释低开销。
2. Load/Store设计：由单周期操作的设想而获得。
3. 硬联控制：为可能最快单步操作而设计。微代码生成了较慢的控制通道并且增加了解释开销。
4. 相当少的指令和寻址方式：简化控制部件快速、简单解释操作。
5. 单一指令格式：一直使用单一指令格式简化了指令的同时加速控制通道译码。
6. 增强的编译时间功能：RISC 机器只执行编译后的代码，这就提供了一个将静态运行时刻的复杂操作搬到编译时刻去执行的机会。最好的一个例子是 MIPS 采用的软件流水线重新组织。

本章将用上述RISC特征来澄清对 RISC 技术所产生的曲解，并引出一个对 RISC 技术的争论焦点。虽然上述特征中的某些观点是具有争议性的，但它至少可以在本章中作为我们研究 RISC 技术的某些事件和含义的工作定义。

2.3 RISC的挑战

前面已经说过，RISC思想对多年来主宰计算机设计的一些观点提出了严肃的挑战。本小节将就这一点对前面所定义的每个 RISC 特征作一检验。其每一特征的分析报告都将说明它与传统设计思想的差别以及它对将来设计思想可能造成的影响。

Load/Store 结构并非是一个新概念，当然它也从来不是RISC技术存在的理由。Load/

Store 设计提供了一个比在一个单指令内进行多存储器存取的结构具有更低原子性的结构，这个较低原子性的结构需要更多的指令，但可以用简化了的指令实现来补偿。这一特征对于支持虚存的机器来说尤为正确，因为支持虚存的机器能从存储器存取失败中恢复。RISC机器中对建立一个能从失败中恢复并能协调地重新启动的机器的设想已作了大量的尝试，而如果需要在一个单指令中进行多存储器存取那就要困难得多。

对直接布线控制的要求的争论比较困难。从具有直接布线控制的机器中继承来的较快周期和较小控制区域的特征始终是很吸引人的。该特征已与能够形成一个非常简单的直接布线控制机器的 RISC 技术有效地结合了起来。对大机器而言，直接布线控制几乎不可能。RISC设计的一条经验就是"RISC指令必须以RISC速度运行"，即通常所使用的指令子集运行应该与直线布线控制下一样。当然，做到这点的唯一办法就是将它们用硬连线联接起来。这里有二个选择：一是要设计一个 RISC；另一个带来这样的问题"直接布线与微代码控制系统是否能有效地结合起来？"，虽然这样的商业机器可能已有样本，但我们还未听闻。

在一结构中具有相对少的指令和寻址方式，对其指令执行有两个好处：第一是简化快速设计步骤。虽然设计任何一台商业处理器都是一项复杂的工作，但在具备了所有文本资料和测试手段后，小而简易处理器的设计还是要比大而复杂机器的设计容易得多；第二个好处体现在处理资源上，如芯片区域、有效电源和面板空间等。一台简单处理器的指令执行比一台复杂处理器需要更少的资源，因此它就为一些非系统结构特征如缓存、地址译码或芯片(面板)I/O保留了空间，这就使它具有了很大的灵活性，而对于受约束程度和复杂程度较大的处理器的设计就不具备这一灵活性。

并非所有 RISC 定义特征只能用于小而简单的处理器。例如：RISC 不是唯一的能从简化指令中获益的系统结构类型；CISC 工具对必须支持复杂的、可变长度的、竖直编码指令有些力不从心：象 VAX 这样的结构在其三角寻址方式和多操作数指令中存在着性能局限。翻译这样的指令流肯定速度慢或者费资源，也可能既慢又费资源。将来在复杂、大型的机器结构中出现比目前更明确的指令格式是不为怪的。当今的一些大结构机器如：Ridge32/100 系列，就已经表现了这一趋势。

重点放在编译器技术上的RISC 设计对计算机设计的主流起到了正面的作用。虽然目前非常流行对 360 机器编写汇编代码，但对任何新的机器结构来说，有效地运行编译代码仍然是它们的正确目标。再则，将运行时刻的功能搬到编译时刻这个想法在 RISC 技术发展的后阶段才受到重视。这个原则应该在所有计算机中受到同样的重视。

采用单周期操作是 RISC 技术的一个特征，虽然许多方面已采纳了它，但尚未被论证过。不幸的是，计算机设计不是一门纯科学，没有一个理论可以论证单周期设计的优越性。人们至今还在探索单周期和多周期操作有效结合的混合方式。

2.4 商业 RISC 机器

到目前为止，我们对 RISC 机器的讨论还只停留在研究工程上。在这一节中，我们要讨论几台具有某些 RISC 特征的商业机器。我们还将特别讨论那些基本上依靠硬件执行来实现性能指标的机器。这是 RISC 设计思想的一个关键特征。对具有大量介于传统硬件/软件之间的有效设计思想的机器的讨论是很值得的。我们将在最后简单讨论几台商业机器，这几台机器由于种种因素被认为是RISC机器，但它们与上一节中所定义的RISC 特征还有很大的距离。