



普通高等教育“十一五”国家级规划教材配套参考书

计算机原理 与系统结构模拟实验

张晨曦 李江峰 刘 依 编著

高等教育出版社



普通高等教育“十一五”国家级规划教材

计算机原理与系统结构 模拟实验

Jisuanji Yuanli yu Xitong Jiegou Moni Shiyan

张晨曦 李江峰 刘依 编著

高等教育出版社·北京

内容提要

本书是普通高等教育“十一五”国家级规划教材,由高等教育出版社出版、张晨曦等编著的《计算机组成与结构》、《计算机系统结构》的配套实验教材。全书设计和编写了15个实验,包括计算机执行程序的过程,寻址方式,CPU数据通路的构建,微程序控制器的组成,4×4位阵列乘法器,存储器组成,中断的屏蔽、响应及处理,MIPS指令系统和MIPS体系结构,流水线及流水线中的冲突,指令调度和延迟分支,Cache性能分析,Tomasulo算法,再定序缓冲(ROB)工作原理,多Cache一致性——监听协议和多Cache一致性——目录协议。这些实验是基于MIPS指令集结构的,书中同时提供了实验所需相关知识的介绍。

本书覆盖面广,内容丰富,有利于加深学生对计算机原理与系统结构知识的理解。为便于学生学习,本书提供了作者专门为实验开发的一套运行于Windows平台的模拟器。这些模拟器界面友好,使用方便、直观,交互性很强。

本书可作为普通高等学校本科计算机与相关专业计算机组成原理、计算机系统结构、计算机组成与结构等课程的实验教材,也可作为自学者的参考书。

图书在版编目(CIP)数据

计算机原理与系统结构模拟实验 / 张晨曦,李江峰,
刘依编著. —北京:高等教育出版社,2015.6
ISBN 978-7-04-042996-1

I. ①计… II. ①张… ②李… ③刘… III. ①电子计
算机-基础理论-实验-高等学校-教材 ②计算机体系结
构-实验-高等学校-教材 IV. ①TP3-33

中国版本图书馆CIP数据核字(2015)第126430号

策划编辑 倪文慧
插图绘制 邓超

责任编辑 倪文慧
责任校对 李大鹏

封面设计 于文燕
责任印制 田甜

版式设计 王艳红

出版发行 高等教育出版社
社址 北京市西城区德外大街4号
邮政编码 100120
印刷 廊坊市科通印业有限公司
开本 787mm×1092mm 1/16
印张 15
字数 300千字
购书热线 010-58581118

咨询电话 400-810-0598
网址 <http://www.hep.edu.cn>
<http://www.hep.com.cn>
网上订购 <http://www.landraco.com>
<http://www.landraco.com.cn>
版次 2015年6月第1版
印次 2015年6月第1次印刷
定价 26.00元

本书如有缺页、倒页、脱页等质量问题,请到所购图书销售部门联系调换

版权所有 侵权必究

物料号 42996-00

前言

计算机组成原理和计算机系统结构课程内容比较抽象、单调,不少内容要通过实验才能得到更好的理解。而且,通过实验研究对系统结构进行量化分析是国际上流行的一种方法。经过多年教学实践,我们发现基于模拟器进行实验是一种很好的方式,有时效果甚至比实物实验更好。基于我们开发的具有自主知识产权的模拟器,本书设计和编写了以下 15 个实验:

- 计算机执行程序的过程
- 寻址方式
- CPU 数据通路的构建
- 微程序控制器的组成
- 4×4 位阵列乘法器
- 存储器组成
- 中断的屏蔽、响应和处理
- MIPS 指令系统和 MIPS 体系结构
- 流水线及流水线中的冲突
- 指令调度和延迟分支
- Cache 性能分析
- Tomasulo 算法
- 再定序缓冲(ROB)工作原理
- 多 Cache 一致性——监听协议
- 多 Cache 一致性——目录协议

每个实验的框架大体由实验目的、实验软件、实验内容和步骤、模拟器使用手册以及相关知识 5 部分构成。“相关知识”部分较为系统地论述了与实验相关的知识,供读者在实验前进行阅读和复习。

实验所需的模拟器和样例程序可从微信公众号 Arch365(张晨曦)获得。

也可扫描二维码并点击关注后,按提示操作。

本书可以与大多数计算机组成原理教材和系统结构教材配合使用,也可作为

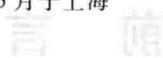


自学者的参考书。

由于作者水平有限,书中难免有错误和不妥之处,敬请读者批评指正。

张晨曦

2015 年 5 月于上海



目 录

实验 1 计算机执行程序的过程	1	4.4 相关知识:微程序控制器	50
1.1 实验目的	1	4.4.1 微程序控制的基本原理	50
1.2 实验软件	1	4.4.2 微程序控制器的组成与工作过程	51
1.3 实验内容和步骤	1	4.4.3 微程序的顺序控制	52
1.4 相关知识:计算机的硬件组成	3	实验 5 4×4 位阵列乘法器	57
1.4.1 计算机系统的硬件组成	4	5.1 实验目的	57
1.4.2 计算机执行程序的过程	7	5.2 实验软件	57
实验 2 寻址方式	25	5.3 实验内容和步骤	57
2.1 实验目的	25	5.4 相关知识:阵列乘法器	58
2.2 实验软件	25	实验 6 存储器组成	60
2.3 实验内容和步骤	25	6.1 实验目的	60
2.4 相关知识:寻址方式	26	6.2 实验软件	60
实验 3 CPU 数据通路的构建	30	6.3 实验内容和步骤	60
3.1 实验目的	30	6.4 相关知识:存储器的组成	61
3.2 实验软件	30	实验 7 中断的屏蔽、响应及处理	64
3.3 实验内容和步骤	30	7.1 实验目的	64
3.4 相关知识:CPU 数据通路	35	7.2 实验软件	64
3.4.1 关于模型机	35	7.3 实验内容和步骤	64
3.4.2 构建基本的数据通路	36	7.4 相关知识:中断的屏蔽、响应和处理	66
3.4.3 指令分步执行过程(按周期分步)	40	7.4.1 中断请求信号的建立与屏蔽	66
实验 4 微程序控制器的组成	48	7.4.2 中断响应与处理	67
4.1 实验目的	48	7.4.3 多重中断与中断屏蔽	69
4.2 实验软件	48	实验 8 MIPS 指令系统和 MIPS 体系结构	73
4.3 实验内容和步骤	48	8.1 实验目的	73

8.2 实验软件	73		
8.3 实验内容和步骤	73		
8.4 MIPSsim 使用手册	76		
8.4.1 启动模拟器	76		
8.4.2 MIPSsim 的窗口	76		
8.4.3 MIPSsim 的菜单	83		
8.5 相关知识:MIPS 指令系统	87		
实验 9 流水线及流水线中的冲突	94		
9.1 实验目的	94		
9.2 实验软件	94		
9.3 实验内容和步骤	94		
9.4 相关知识:流水线、相关与冲突	96		
9.4.1 一条经典的 5 段流水线	96		
9.4.2 相关与流水线冲突	98		
9.4.3 流水线的实现	102		
实验 10 指令调度和延迟分支	107		
10.1 实验目的	107		
10.2 实验软件	107		
10.3 实验内容和步骤	107		
10.4 相关知识:指令调度和延迟分支	108		
10.4.1 指令调度	108		
10.4.2 延迟分支	109		
实验 11 Cache 性能分析	112		
11.1 实验目的	112		
11.2 实验软件	112		
11.3 实验内容及步骤	112		
11.4 MyCache 模拟器使用方法	114		
11.5 相关知识:Cache 的基本原理	115		
11.5.1 Cache 的映像规则	115		
11.5.2 查找方法与替换算法	117		
11.5.3 写策略与改进 Cache 性能	118		
11.5.4 3 种类型的命中	119		
11.5.5 降低不命中率的方法	119		
11.5.6 分离 Cache 和混合 Cache	126		
实验 12 Tomasulo 算法	128		
12.1 实验目的	128		
12.2 实验软件	128		
12.3 实验内容及步骤	128		
12.4 Tomasulo 算法模拟器使用方法	129		
12.5 相关知识:Tomasulo 算法	132		
12.5.1 基本思想	132		
12.5.2 具体算法	135		
实验 13 再定序缓冲 (ROB) 工作原理	139		
13.1 实验目的	139		
13.2 实验软件	139		
13.3 实验内容及步骤	139		
13.4 ROB 模拟器的使用方法	140		
13.5 相关知识:再定序缓冲 ROB	143		
实验 14 多 Cache 一致性——监听协议	147		
14.1 实验目的	147		
14.2 实验软件	147		
14.3 实验内容及步骤	147		
14.4 监听协议模拟器使用方法	148		
14.5 相关知识:监听协议	149		
14.5.1 监听协议的基本思想	149		
14.5.2 监听协议的实现	150		
实验 15 多 Cache 一致性——目录协议	155		
15.1 实验目的	155		
15.2 实验软件	155		
15.3 实验内容及步骤	155		
15.4 目录协议模拟器使用方法	156		
15.5 相关知识:目录协议	157		
15.5.1 目录协议的基本思想	157		
15.5.2 目录协议实例	160		
附录 A MIPSsim 的指令列表 (MIPS 64 指令集的一个子集)	164		

附录 B MIPSsim 的指令格式及功能 (MIPS 64 指令集的一个子集)	175
---	-----

附录 C 模拟器 MIPSsim 的汇编语言	226
参考文献	229

实验 1 计算机执行程序的过程

1.1 实验目的

- 加深对计算机系统硬件组成的理解。
- 理解计算机是如何执行程序的。

1.2 实验软件

采用计算机执行程序模拟器 CompEXE。

下载地址：

微信公众号：Arch365。

请扫描以下二维码，点击关注后，按提示操作。



1.3 实验内容和步骤

- ① 双击 CompEXE。
- ② 观察及浏览每个部件及其名称，如图 1.1 所示。
- ③ 阅读存储器中的程序，即

Load R1,200(R0)

Load R2,#4

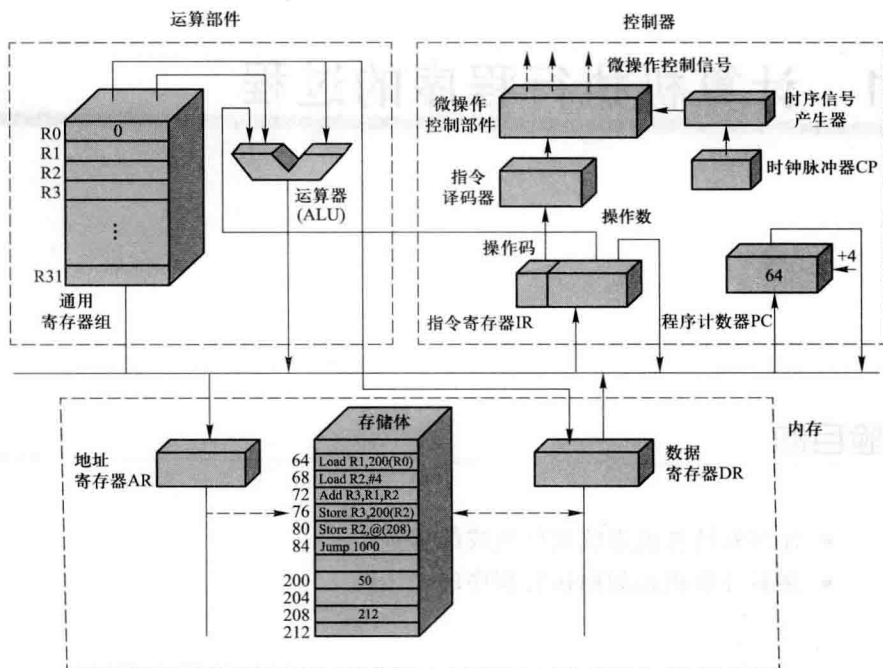


图 1.1 计算机的硬件组成

```

Add   R3, R1, R2
Store R3, 200(R2)
Store R2, @(208)
Jump  1000

```

④ 单击 Load R1, 200(R0) 指令, 观察该指令的执行过程, 观察各寄存器内容的变化。

⑤ 再次单击 Load R1, 200(R0) 指令, 观察操作数的来源, 观察访存有效地址是如何计算的。

⑥ 单击 Load R2, #4 指令, 观察该指令的执行过程, 观察各寄存器内容的变化。

⑦ 再次单击 Load R2, #4 指令, 观察立即数的使用。

⑧ 单击 Add R3, R1, R2 指令, 观察该指令的执行过程, 观察各寄存器内容的变化。

⑨ 再次单击 Add R3, R1, R2 指令, 观察两个操作数如何来自寄存器组。

⑩ 单击 Store R3, 200(R2) 指令, 观察该指令的执行过程, 观察各寄存器内容的变化。

⑪ 再次单击 Store R3, 200(R2) 指令, 观察数据存到了哪个存储单元, 观察是如何获得该单元的地址的。

⑫ 单击 Store R2, @(208) 指令, 观察该指令的执行过程, 观察各寄存器内容的

变化。

⑬ 单击 Jump 1000 指令,观察该指令的执行过程,观察程序计数器 PC 内容的变化。

1.4 相关知识:计算机的硬件组成

计算机在当今世界中已经是无处不在,而且几乎无所不能。人们的日常生活和工作都离不开计算机。当用手机打电话、用 mp3 听音乐、用 PSP 玩游戏、自驾出游、到银行取钱、乘飞机旅行时,计算机都在不知不觉中默默为人们工作。它让生活充满色彩,让工作多出成果,对于人类社会发展的影响是广泛而又极其深远的。

计算机系统由硬件和软件两大部分组成。这两部分密切配合,计算机才能正常工作和发挥作用,两者缺一不可。硬件是计算机系统的物质基础,少了它,再好的软件也无法运行;软件则像是计算机系统的灵魂,少了它,再好的硬件也毫无用途。它们只有并驾齐驱,才能充分发挥计算机的作用和效率。

从理论上讲,对于计算机的某一具体功能来说,既可以用硬件实现,也可以用软件实现,即硬件和软件在逻辑功能上是等效的。但其实现成本和速度则会有比较大的差别。在设计一个计算机系统时,必须根据设计要求、现有的技术和条件以及成本上的考虑,确定哪些功能由硬件实现,哪些功能由软件实现,即确定硬件和软件的功能分配。这个软件和硬件的交界面则称为计算机的系统结构,如图 1.2 所示。

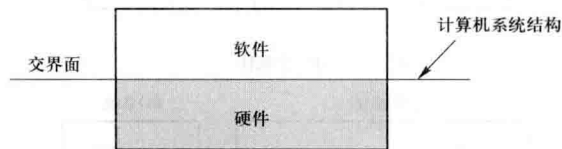
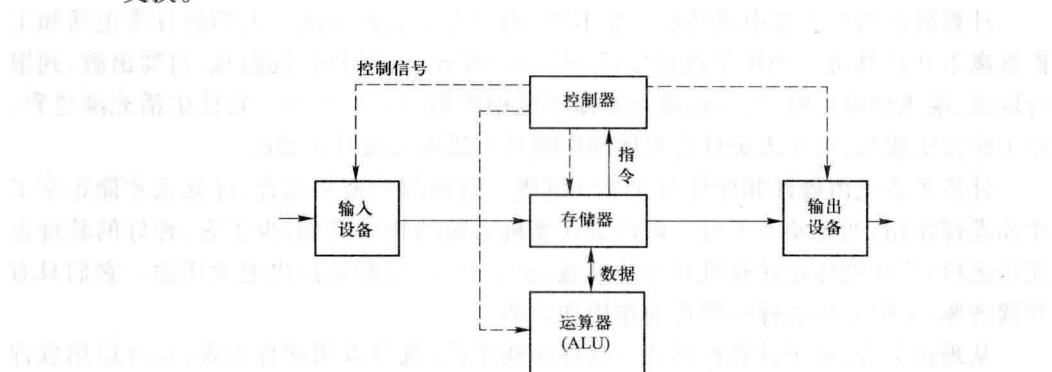


图 1.2 硬件、软件和计算机系统结构

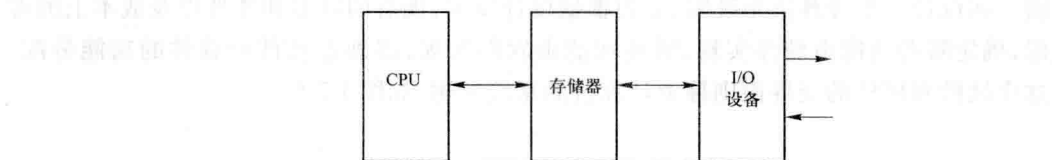
自第一台通用电子计算机诞生以来的 60 多年中,计算机技术得到了飞速发展,其速度之快,实在是令人赞叹。今天,用不到 5 000 元人民币购买的个人计算机,其性能、主存和硬盘容量都已经超过 20 多年前用 100 万美元购买的大型机。对于许多应用来说,现在的高性能微处理器的性能已经超过了 10 年前的超级计算机。在过去的 20 多年中,计算机的性能差不多每 18 个月就翻一番(尽管近几年有所放缓)。这种惊人的发展一方面得益于计算机硬件制造技术的发展,另一方面则是因为计算机系统结构的创新。

1.4.1 计算机系统的硬件组成

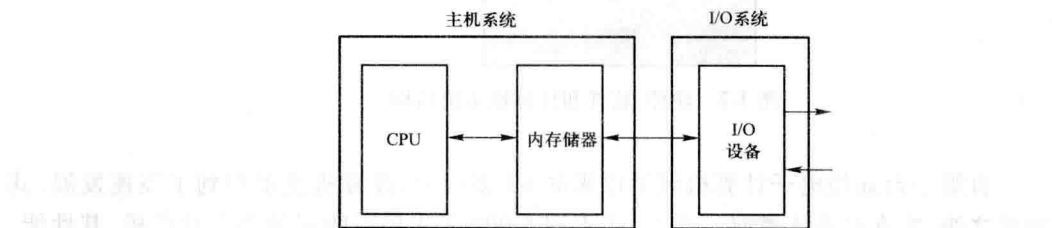
现代计算机系统的硬件结构如图 1.3(a) 所示, 它由 5 个部件构成: 运算器 (ALU)、存储器、控制器、输入设备和输出设备。运算器用于实现对数据的加工, 包括算术运算和逻辑运算; 存储器用于存储数据和程序; 控制器是计算机的指挥控制中心, 控制计算机各部件有序协调地工作; 输入设备和输出设备实现外部世界与计算机之间的数据交换。



(a) 5个部件



(b) 3个部件



(c) 两个子系统

图 1.3 现代计算机系统的硬件结构

由于运算器和控制器在逻辑关系上联系紧密, 而且往往又制作在同一块芯片上, 构成了众所周知的 CPU (Central Processing Unit, 中央处理器), 所以有时也称计算机系统是由 3 个部件构成的, 即 CPU、存储器、输入/输出设备 (也称 I/O 设备), 如图 1.3(b)

所示。

存储器主要由两部分构成:内存储器(简称内存)和外存储器(简称外存)。内存有时也称为主存(主存储器)。

有时,把“CPU+内存”称为主机系统,而把输入/输出设备及其相关的接口称为 I/O 系统,如图 1.3(c)所示。

早期的计算机采用的是如图 1.4 所示的结构。与图 1.3(a)相比,主要的区别是该结构以运算器为中心。这种结构是匈牙利裔数学家冯·诺依曼于 1946 年提出的,所以称为冯·诺依曼结构。虽然现代计算机在结构上已经有了很大的变化,但都可以看成是冯·诺依曼结构的改进,而且仍然是采用冯·诺依曼当时提出的存储程序原理。存储程序原理是指在计算机解题之前要事先编制好程序,并与所需要的数据一起预先存入主存当中。当程序开始执行后,由控制器按照该程序自动、连续地从存储器中取出指令并执行,直到获得所要求的结果。

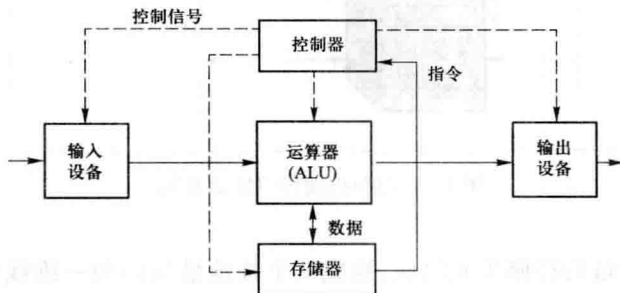


图 1.4 冯·诺依曼结构

图 1.5 所示是一个模型机硬件组织的示意图。这里,模型机是指为了讲解方便而假想的一台简化了的计算机。下面分别介绍其各个部件。

1. 运算部件

运算部件是计算机的执行部件,用于对数据进行加工处理。该部件由两部分构成:运算器和通用寄存器组。运算器即算术逻辑单元,简称 ALU (Arithmetic and Logical Unit)。用于完成算术运算和逻辑运算。算术运算包括加、减、乘、除以及它们的复合运算。逻辑运算包括与、或、非、异或、比较、移位等。通用寄存器组简称 GPR (General Purpose Register),它由若干个或几十个寄存器构成,用于暂时存放运算数据和中间结果。图 1.5 中共有 32 个通用寄存器 $R_0, R_1, \dots, R_{31}$ 。

2. 内存

存储器是计算机的存储部件,用于存储程序和数据。

现在的内存一般用半导体技术实现,外存则往往用磁记录方式实现,例如硬盘。

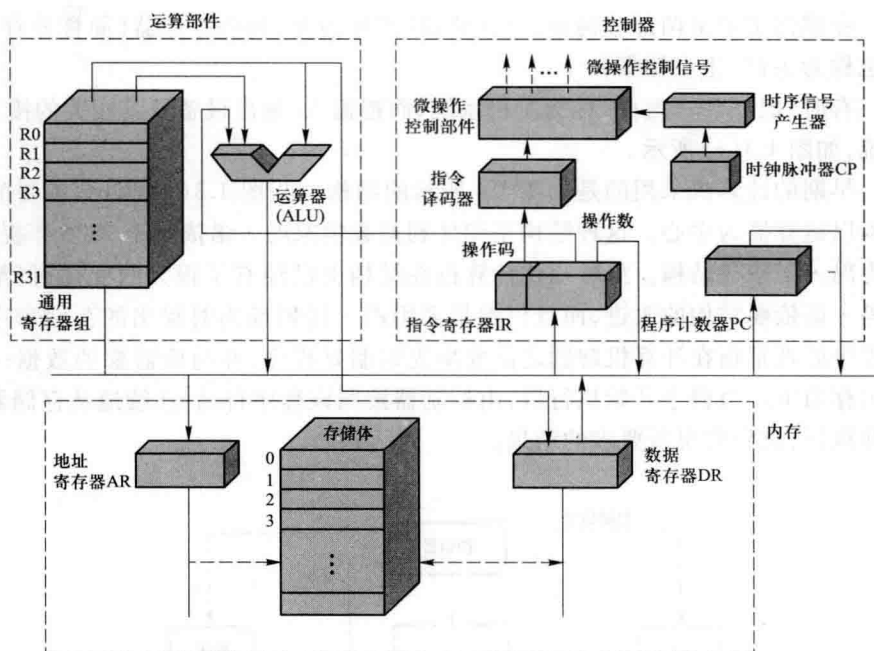


图 1.5 模型机硬件组织示意图

内存由大量的存储单元组成,构成一个按地址访问的一维线性空间。每个存储单元有一个唯一的编号(图 1.5 中存储体左边的编号),就像街道门牌号那样。这个编号称为该存储单元的地址,用这个地址可以唯一地访问到该单元。

每个存储单元可以存放多个二进制位,其位数一般与计算机的字长相同,一般是字节的整数倍。对存储器可以进行的操作包括“读”和“写”两种,有时称为访存。内存有两个重要的寄存器:地址寄存器 AR 和数据寄存器 DR。AR (Adress Register) 存放访存的地址,DR (Data Register) 存放从内存读出的数据,或者存放要写入内存的数据。

3. 控制器

控制器是计算机的管理机构和指挥中心,它协调计算机的各个部件自动地工作。具体来说,就是按照程序中事先设计好的解题步骤,控制计算机各个部件有条不紊地工作。

它由以下 6 部分构成:① 指令寄存器 IR (Instruction Register),用于存放当前正在执行的指令;② 程序计数器 PC (Program Counter),用于存放当前正在执行的指令的地址;③ 指令译码器,对指令进行译码,即识别当前指令是什么指令,以便形成相应的控制信号;④ 时钟脉冲 CP (Clock Pulse),协调计算机各部件操作的同步主时钟,其工作

频率称为计算机的主频;⑤ 时序信号产生器,按时间顺序周而复始地发出节拍信号;⑥ 微操作控制部件,微操作是指硬件电路中不可再细分的简单操作,如把一个寄存器中的数放到公用数据通路上、寄存器接收其数据入口的数据、计数器加 1 等。各种微操作都是在一个节拍内完成的。

任意一条指令的执行往往都要分解成许多微操作,这些微操作需按先后次序分配到各个节拍中去完成。微操作控制部件根据指令的译码结果以及时序部件给出的节拍和脉冲信号,产生该指令执行过程中各节拍所需要的微操作控制信号,并将它们发送给包括控制器本身在内的各部件,使之协调、分步骤地进行操作,实现指令的执行。

1.4.2 计算机执行程序的过程

为了深入理解计算机的组成和工作原理,需要对计算机的工作过程(即执行程序的过程)有比较系统和全面的认识。这对于建立计算机系统的整机概念是非常重要的。因此在讲解更多细节之前,本章以图 1.5 所示的模型机为例,分步讲解一个简单样例程序的执行过程。

1. 样例程序

程序是由语句(高级语言)或指令(机器语言或汇编语言)组成的序列,按从上到下的顺序执行。用高级语言(或汇编语言)编写的程序一般要编译(或汇编)并连接成可执行代码(程序)才可以执行,例如手机上的 APP 就是可执行程序。可执行程序在执行前要先调入内存。

为便于理解后面的论述,先做几点说明。

① Load 指令是装载寄存器指令,它把一个数据送入指定的寄存器,这个数据可以是立即数,也可以是来自存储器的某单元。

② Store 指令是存储指令,它把指定寄存器中的数据存入存储器的某个单元。

③ Add 指令是加法指令,它把两个寄存器中的数据相加,并把结果存到指定的寄存器。

④ Jump 指令是跳传指令,即跳转到新的地址去执行指令。

⑤ $[R_x]$ 表示寄存器 R_x 的内容。

⑥ $MEM[y]$ 表示存储器中地址为 y 的存储单元的内容,“ $\rightarrow$ ”表示传送。

考虑以下样例程序:

第 1 条: Load $R1, 200(R0)$ // $MEM[[R0]+200] \rightarrow R1$

第 2 条: Load $R2, \#4$ // $4 \rightarrow R2$, “4”这个值放在指令中,称为立即数

第 3 条: Add $R3, R1, R2$ // $[R1] + [R2] \rightarrow R3$

第 4 条: Store $R3, 200(R2)$ // $[R3] \rightarrow MEM[[R2]+200]$

第 5 条: Store $R2, @(208)$ // $[R2] \rightarrow MEM[MEM[208]]$, @ 表示间接寻址。

第6条: Jump 1000

// 1000→PC

2. 第1条指令的执行过程

(1) 说明

指令: Load R1, 200(R0)。

指令地址: 指令在内存中的地址为 64。

功能: $MEM[[R0]+200] \rightarrow R1$ 。即用 $[R0]+200$ 作为地址访问存储器, 将读出的内容送给 R1。

假设已经把程序计数器 PC 的内容设置为 64。

(2) 分步操作

第1步: 取指令。

① $[PC] \rightarrow AR$ 。即将 PC 的内容(64)传送到内存的地址寄存器 AR, 如图 1.6 中的粗线所示。

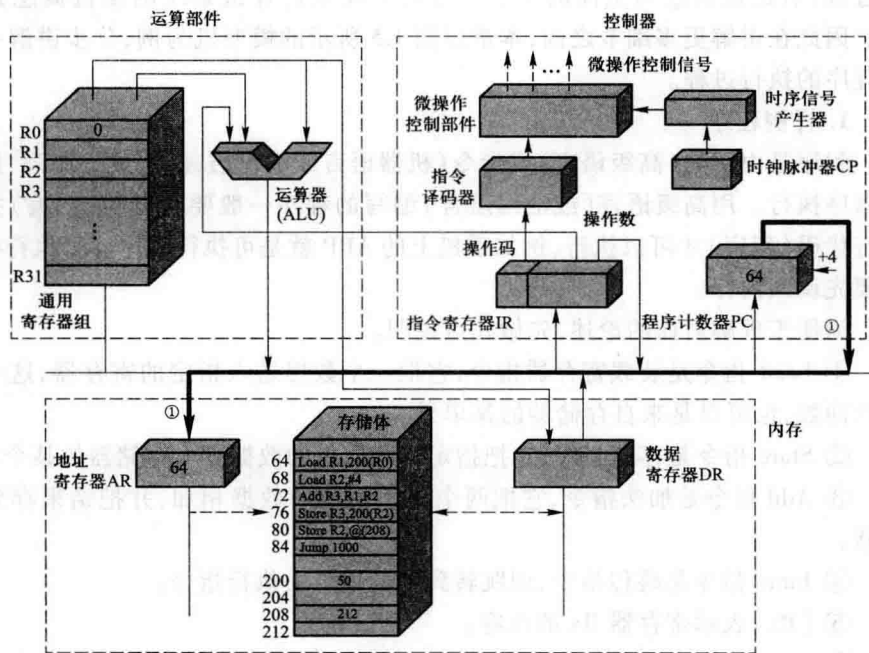


图 1.6 Load R1, 200(R0) 取指令第①步

② 从存储器读出第一条指令, 放到数据寄存器 DR, 如图 1.7 中的粗线所示。

③ 将该指令从 DR 送到指令寄存器 IR, 如图 1.8 中的粗线所示。

第2步: 指令译码器进行译码, 控制器结合时钟脉冲信号 CP 和时序信号产生器生成的节拍信号, 产生执行该指令所需要的微操作控制信号(注意, 这些信号被分配到

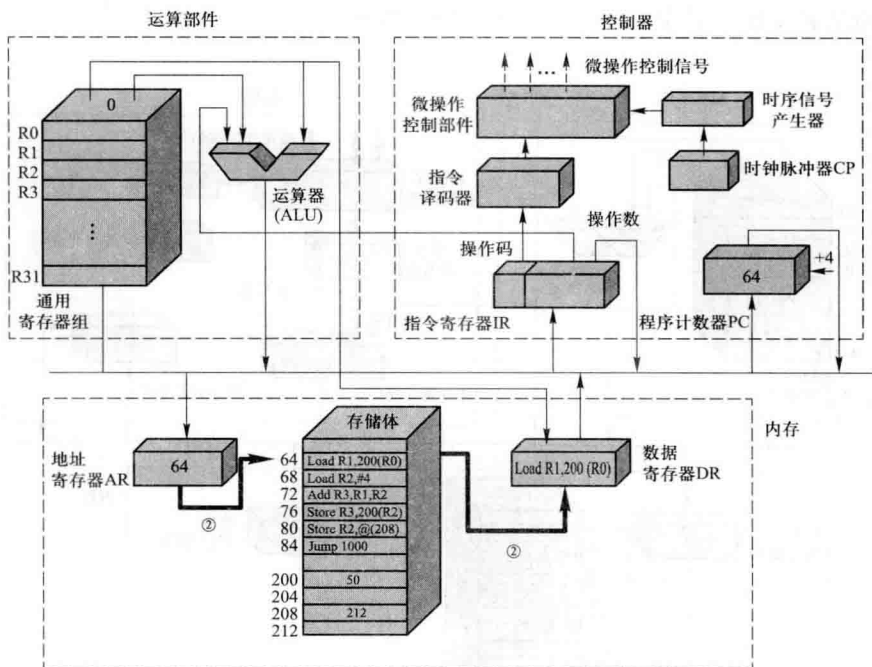


图 1.7 Load R1, 200(R0) 取指令第②步

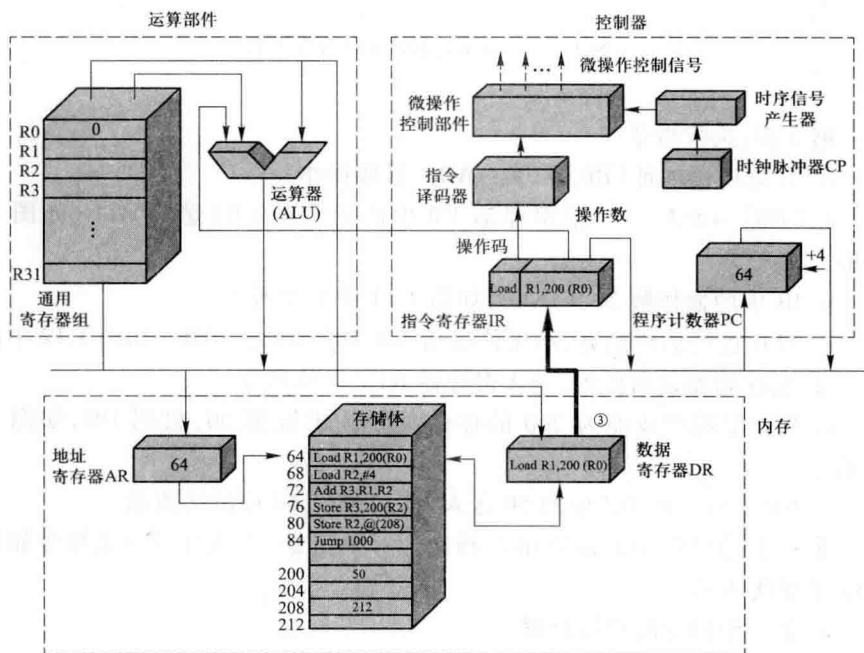


图 1.8 Load R1, 200(R0) 取指令第③步