

计 算 机 结 构 和 组 织

[美] 约 翰 P. 海 斯 编 著

上 海 科 学 技 术 文 献 出 版 社

计 算 机 结 构 和 组 织

〔美〕 约翰 P·海斯 著

上海科学技术文献出版社

计算机结构和组织

〔美〕约翰 P. 海斯 著

*

上海科学技术文献出版社出版
(上海市武康路2号)

新华书店上海发行所发行
吴江伟业印刷厂印刷

*

开本 787×1092 1/32 印张 19.25 字数 457,000

1982年1月第1版 1984年10月第3次印刷

印数: 12,801—32,400

书号: 15192·175 定价: 2.35 元

《科技新书目》78-209

译 者 的 话

近年来,由于计算机大量出现和计算机技术、工艺的进展,总体结构已形成了一门独立的学科,特别是大规模集成电路的发展,为计算机设计提供了各种极其有效而又廉价的元件,因此对于工程师和科学家来说,大量采用这些器件而研究其体系结构将变得越来越重要。计算机系统结构已经成为计算机领域本身的重要研究课题,已逐渐成为计算机科学和工程系的必修课目。

为向计算机工作者提供系统的设计理论,我们翻译了 John P. Hayes(约翰 P·海斯)编著的“Computer Architecture and Organization”译名为《计算机结构和组织》由美国 McGraw-Hill Book Company 1978 年出版。该书是作者多年在南加里福尼亚大学讲授计算机体系结构课程的教材,书中采用了许多实际的计算机,如 IAS、IBM 360/370, Intel 8080 和 8748, STAR-100, HP21-MX, ILLIACIV, ARPANET 计算机网络作为背景,从体系和系统的角度来阐述计算机各部分如处理机、控制器、存贮系统、系统组织的设计。该书内容充实、层次清晰、概念明确,对于计算机专业的总体设计人员、高等院校计算机专业的教师和研究生都可作为一本很好的参考书。

本书由复旦大学计算机科学系徐研人、曹邦伟、朱传琪分工翻译,翻译过程中曾得到舒仁本、郭明森同志大力协助。全书由宋百川、陈国尧、孙世球同志分别校对,在此谨表感谢。

译 者

一九八一年九月

原 序

计算机总体结构可以近似地定义为计算机的构造、作用以及设计的研究。近年来,主要由于计算机大量出现和计算机技术、工艺上的进展,总体结构已经形成了一门独立的学科。七十年代中、大规模集成电路的发展,为计算机设计提供了各种极其有效、而又便宜的元件。其中最突出的发展也许是微处理机和微型计算机的出现。其结果是在非常短的时间内,在过去采用专用逻辑线路的许多场合都可使用计算机系统。因此,计算机总体结构的概念对许多工程师和科学家来讲变得越来越重要。

本书主要打算作为电气工程和计算机科学系高年级和研究生的计算机总体结构和组织一课的教材。它着重在计算机硬件方面,但也论及有关软件方面的问题。本书假定读者对计算机程序设计已很了解,并至少已熟悉一种简单的计算机的运行。同时也假设读者对逻辑设计稍有熟悉。这里不要求有专门的数学知识,但偶尔也用到一些初等微积分和概率论。

本书的目的是要提供计算机总体结构的全面和完整的观点。题材的范围基本上与 IEEE 计算机学会特别工作组在计算机总体结构方面所推荐的内容相一致*。着重点是基本设计原理和性能评价。并且试图采用统一的术语和记号。本书中有将近一百个习题为学生提供计算机分析和设计方面有用的练习。所有问题的完整解答的题解手册,可以由课程教师从出版处买到。

本书共分六章。第一章从历史的观点追溯了计算机的演变

* G. E. Ross mann 等著“计算机硬件结构研究教程”见 Computer 第 8 卷第 12 期 1975 年 12 月 p. 44~63.

过程,并介绍全书采用的大部分的记号和术语。第二章论及设计方法学,并详细地讨论寄存器级和处理机级的设计。也介绍了利用初等排队论和模拟方法进行的性能评价。以下二章是关于指令系统处理机的设计。其中第三章包括指令系统、算术部件设计和并行处理。第四章的论题是控制部件设计,其中简单讨论了硬接线控制,同时也详尽地讨论了微程序设计。第五章讨论了包括虚拟存贮器、相联存贮器和高速缓冲存贮器在内的存贮技术和组织。最后一章讨论系统结构和通信。同时讨论了输入输出系统、多处理机、容错计算机以及计算机网络。本书采用了许多实际的计算机系统作为例子,它们有 Von Neumann 的 IAS 机器,IBM360-370 系列、Intel 8080 和 8748,控制数据公司(CDC)的 STAR-100, Hewlett-Packard 21 MX, ILLIAC IV, Bell No.1 ESS 和 ARPANET 计算机网络。

本书近几年来为南加利福尼亚大学计算机系统结构的研究生预备班所采用。本书所包含的论题稍为超过了一学期课程的内容,教师可有选择重点的余地。第一、二章的背景材料,例如,在 2.1 节中的逻辑设计概述,如学生已有一定的基础,有许多可以略去,或者安排为阅读材料。对于一些更专门的课题,例如,4.3.2 节“微指令规模的最小化”,5.3.3 节的“相联存贮器”或 6.3.2 节的“容错计算机”视情况而定也可以略去。这样并不会失去连贯性。

非常感谢我在南加利福尼亚的同事们和学生们,特别是 P. Vahdat 和 A. Zygielbaum 对本书提出的建议。也对 Berkeley 分校 G. Miller 所作的评论表示深切的感谢。特别感谢我的妻子 Terre,为本书的许多段作了审阅和校对。

约翰 P·海斯

目 录

第一章 计算机的演变	1
1.1 计算机的性质	1
1.1.1 计算的需要	2
1.1.2 抽象模型	4
1.1.3 计算机的局限性	7
1.2 机械时代: 1623~1945	13
1.2.1 第一台计算器	13
1.2.2 Babbage 的计算机	14
1.2.3 后期的进展	19
1.3 第一代计算机: 1946~1954	21
1.3.1 电子计算机	22
1.3.2 存贮程序计算机	23
1.3.3 第一代计算机	28
1.4 第二代计算机: 1955~1964	38
1.4.1 一般特性	38
1.4.2 第二代计算机	40
1.4.3 输入输出处理	43
1.4.4 软件及其它进展	46
1.5 近代进展	50
1.5.1 第三代计算机	50
1.5.2 第三代计算机族	58
1.5.3 微处理机和微型计算机	65
1.6 小结	74
习题	76
参考文献	78

第二章 设计方法学	81
2.1 引言	81
2.1.1 系统模型化	81
2.1.2 设计级	88
2.1.3 组合电路设计	91
2.1.4 时序电路设计	98
2.2 寄存器级	104
2.2.1 一般特性	104
2.2.2 描述语言	109
2.2.3 组合部件	115
2.2.4 时序部件	124
2.2.5 设计方法	131
2.3 处理机级	141
2.3.1 引言	141
2.3.2 组成部件	146
2.3.3 设计技术	152
2.3.4 排队论模型	158
2.3.5 模拟	163
2.4 小结	170
习题	171
参考文献	175
第三章 处理机设计	177
3.1 引言	177
3.1.1 处理机组织	177
3.1.2 信息表示	183
3.1.3 数据格式	189
3.2 指令系统	199
3.2.1 指令格式	199
3.2.2 指令类型	209
3.2.3 实现	213
3.3 算术运算	217

3.3.1	定点加减法	217
3.3.2	定点乘法	224
3.3.3	定点除法	237
3.3.4	ALU 设计	247
3.3.5	浮点运算	250
3.4	并行处理	259
3.4.1	引言	259
3.4.2	性能考虑	264
3.4.3	流水线处理机	274
3.4.4	多部件处理机	284
3.5	小结	291
	习题	293
	参考文献	299
第四章	控制部件设计	302
4.1	引言	302
4.1.1	指令时序	303
4.1.2	指令解释	308
4.2	硬接线控制	311
4.2.1	设计方法	311
4.2.2	乘法器控制部件	321
4.2.3	CPU 控制部件	329
4.3	微程序控制	334
4.3.1	基本概念	334
4.3.2	微指令长度最小化	347
4.3.3	乘法器控制部件	355
4.4	微程序计算机	365
4.4.1	CPU 控制部件	365
4.4.2	常规计算机	373
4.4.3	非常规计算机	381
4.5	小结	388
	习题	391

参考文献	393
第五章 存贮器组织	396
5.1 存贮技术	396
5.1.1 存贮器件特征	397
5.1.2 随机存取存贮器	405
5.1.3 顺序存取存贮器	420
5.2 虚拟存贮器	435
5.2.1 多级存贮器	435
5.2.2 主存分配	447
5.2.3 替换策略	452
5.2.4 段和页	460
5.2.5 文件组织	466
5.3 高速存贮器	474
5.3.1 交叉存贮器	475
5.3.2 高速缓冲存贮器(cache)	479
5.3.3 相联存贮器	483
5.4 小结	492
习题	495
参考文献	498
第六章 系统组织	500
6.1 通信	500
6.1.1 近程通信	500
6.1.2 远距离通信	502
6.1.3 互连结构	505
6.1.4 总线控制	510
6.2 输入输出系统	521
6.2.1 程序控制 IO	523
6.2.2 DMA 和中断	530
6.2.3 IO 处理机	540
6.2.4 CPU-IO 的交互作用	549
6.3 多 CPU 系统	558

6.3.1 多处理机	558
6.3.2 容错计算机	569
6.3.3 计算机网络	582
6.4 小结	591
习题.....	593
参考文献.....	596

第一章 计算机的演变

本章追溯了数字计算机的发展历史。提出了有关计算机结构的总的观点,并引出许多在以后几章中将深入探究的概念。同时也详细地介绍了许多代表性的计算机。

1.1 计算机的性质

几乎在整个历史发展中,人类主要都是依靠他们的大脑来进行计算的;换言之,他们本身就是一个计算机。人们发明了各种诸如算盘、计算尺等等的计算工具;这些工具能简化手工计算,但不能代替它。随着计算的规模和复杂程度不断地增长,手工计算中存在着的两种严重限制变得很明显。

1. 手工所能作的计算速度是有限的。象加法或乘法一类典型的基本运算就需要几秒钟甚至几分钟。目前需要上亿次这类运算的问题普遍使用计算机,并迅速地求解出来。要用人工在合情合理的时间内或用合情合理的代价来解决这一类问题是绝对办不到的。

2. 众所周知,手工的计算容易出错。所以用手工实现复杂的计算,一般不是很可靠的,除非采取精心的预防出差错的措施。由于机器不会出现人工通常会出现的错误(如思想不集中,粗心,疲劳等等引起的错误),因此,机器可以极大地限度地提供正确的结果。

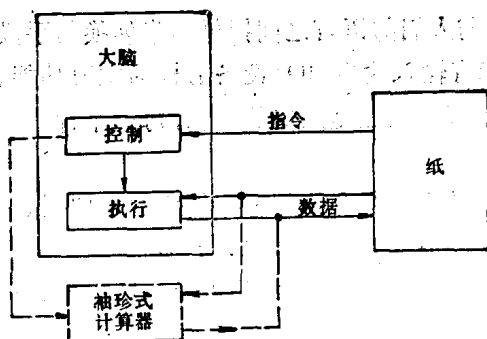
Charles Babbage 曾经多次引用下面的例子来说明他为什么去设计第一台计算机,即微分分析机^[20]。1794年,法国政府

在 G. F. Prony 的指导下, 开始了一次完全用手工来计算一套庞大的数学用表的计划。编造的表, 包括从 1 到 200,000 的自然数算到小数第 19 位的对数。另外, 还有正弦值与正切值及其对数, 以及正弦和正切与它们的弧度之比值的对数等类似的表。整个计划大约历时二年, 并且雇佣了 70~100 人才完成。其中大多数人的数学能力只限于作加法和减法, 由一个数学专家小组指导他们。为了使错误减到最少, 每一个数都由二个计算员独立地计算两遍, 并比较其结果。最后产生的表格, 按对开本装订了 17 卷(可是它从来没有出版), 仅仅自然数的对数表, 就包括了约 800 万个数字。

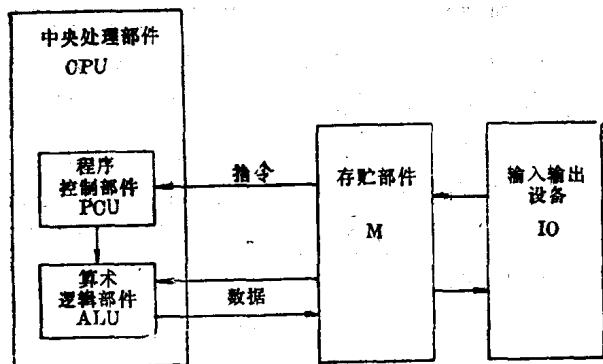
1.1.1 计算的需要

作为研究开始, 先分析一下采用纸和笔进行手工计算的过程是有益的。纸的基本目的是信息存贮。在纸上的信息, 可以包括进行计算时应当遵循的一系列指令——即算法或程序, 以及所使用的数据(数码)。计算过程的中间结果和最终结果都记录在纸上。必要的计算过程在大脑中进行, 大脑可以被称为处理器。大脑完成的两种不同的主要功能可分为: 控制功能(它解释指令并保证以正确的顺序执行), 和执行功能(它进行象加法、减法、乘法和除法等具体的计算)。现今在执行这后一功能时, 人类的大脑获得普遍存在的袖珍电子计算器的帮助。图 1.1a 显示了这种人类计算过程的概貌。

图 1.1(b) 指出了一个计算机器的主要部分, 它与人的计算是类似的。存贮部件相当于人工计算所用的纸; 其目的是存放指令和数据。程序控制部件解释指令并且按顺序排好。算术逻辑部件执行指令。它之所以这样称呼, 是因为指令可以包含数值运算指令(算术运算), 或象程序转移和符号处理等非数值运算指令, 后者亦称为逻辑运算, 这个名称不是很确切但是比较方



(a)



(b)

图 1.1 人计算(a)和机器计算(b)的主要部件

便的。程序控制和算术逻辑部件合在一起就形成中央处理部件(CPU)，它大致上与手工计算时的人的大脑相对应。人与机器之间最显著的差别，是信息(包括指令和数据)的表示方式不同。人类使用的是有着大量符号的正常的语言，同时通常又以十进制(以 10 为基)的形式表示数。但是，在现代计算机里，信息的存贮和处理均以二进制形式进行，也就是说，通常用 0 和 1 表示的两个符号。为了在机器及其用户之间提供通信，就需要在

机器的语言与人们的语言之间提供信息转换的手段。这就是图 1.1(b) 中标有输入输出(I/O)设备方框的主要功能。

因此,不管是人还是人造的机器,每一个计算机都必须具有下列部件。

1. 一个具有解释和执行程序能力的处理器。
2. 一个存放程序 and 数据的存储器。
3. 在存储器和处理器之间,以及在计算机和外界之间转换信息的工具。

1.1.2 抽象模型

计算可以理解为由某个函数 $f(X)$ 的值,这里 X 是给予的输入数据, $Z=f(X)$ 是所要求输出的数据。 X 、 Z 和 f 均可给予广泛的解释。 X 和 Z 可以代表数、词句、信息文件等等。 f 可以是数值计算、定理证明、文件更新规程等等。为了用一特定计算机求 $f(X)$,我们必须能将 f 表达成一串函数 $f_1, f_2, \dots, f_n$, 这些函数可以由计算机指令系统来确定。指令系统就是计算机能够执行基本功能的集合。 $f_1, f_2, \dots, f_n$ 就可以理解为对 $f(X)$ 求值的程序。基本操作的序列:

$$Y_1=f_1(X)$$

$$Y_2=f_2(Y_1)$$

.....

$$Y_{n-1}=f_{n-1}(Y_{n-2})$$

$$Z=f_n(Y_{n-1})$$

可以作为计算的一个形式定义。

在研究计算机设计之前,自然地会先提出这样的问题:是否存在着无法由“合情合理的”计算机进行计算的问题呢?如果存在,那末显然就需要认识到这一点,以免我们试图建造不可能完成任务的机器。下面这两个合理的说法,一般是可以接受的:

1. 机器不可能存放所有可能的问题的答案。

2. 机器处理信息(执行指令)的速度是有限的。

这样的一种机器,可以认为仅仅当它能以有限步骤,就是说,在有限时间内作出回答时,它才具有实现某项计算的能力。

1936年,英国数学家 Alan Turing (1912~1954) 提出了满足上述合理准则的计算机抽象模型^[26]。这个模型现在称之为 Turing 机器。图 1.2 指出了 Turing 机器的组成部分。正如前一节指出的,任何计算机的两个基本单元是存储器 and 处理器。Turing 机器的存储器是无限长度的一条带 M , 这条带在纵向上分成一个个方格。每个方格可以是空白,或者是有限个符号中的一个。处理机是有限个状态的数字机器,它具有一个读写头,能读出带上任一方格的内容,对该方格的内容加以改变,并能将带从读写头的现行位置上向左或向右移动一个方格。

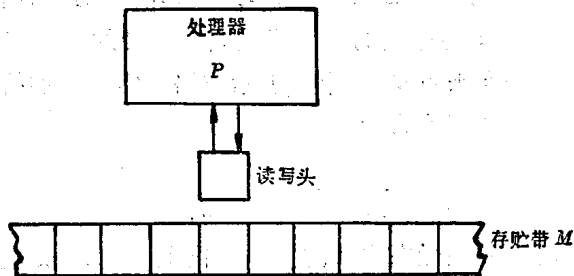


图 1.2 Turing 机器

Turing 机器可以看作有下列组成形式的有限指令集合:

$$s_h \quad t_i \quad o_j \quad s_k$$

其意义是当控制处理机处于 s_h 状态, 且符号 t_i 当时处于读写头之下, 那末执行操作 o_j , 而且将 P 的状态变到 s_k 。操作 o_j 可以是以下四种的任何一种:

1. $o_j = t_j$, 意即将符号 t_j 写到带上 (这即代替原来的符号 t_i)。

2. $o_j = R$, 意即将读写头的位置, 安置在当前方格的右面一个方格上, 就是说, 将带向左移一个方格。

3. $o_j = L$, 意即将读写头的位置, 安置在当前方格的左面一个方格上, 就是说, 将带向右移一个方格。

4. $o_j = H$, 意即停止计算。

指令集合基本上就是 P 的全部可能状态的变迁。在指令中的信息, 也可以从 P 的状态表得到。

用 Turing 机器按下面的方法, 可实现计算 $Z = f(X)$ 。首先, 将输入数据 X , 以适当的编码形式放在空白带上。然后启动 Turing 机器, 产生要完成的操作序列 $f_1, f_2, \dots, f_n$ 。在第 n 步操作结束时, 机器停止, 同时带里应包含了结果 Z 。

例 1.1: 可将两个自然数相加的 Turing 机器 任何自然数 n , 可以用 n 个 1 的序列的一元形式表示。带的符号字母, 只需二个符号 1 和 b 来组成, 其中 b 表示空格。需要相加的两数 n_1, n_2 先用下面形式写在带上:

$$b \underbrace{1 \ 1 \ \dots \ 1}_{n_1} \ b \underbrace{1 \ 1 \ \dots \ 1}_{n_2} \ b$$

同时将读写头放在最左面 1 的左面空格上。Turing 机计算 $n_1 + n_2$, 只要简单地将分割 n_1 与 n_2 的空格上用符号 1 代之, 然后删去最左边的 1, 就可得到模式

$$b \ b \underbrace{1 \ 1 \ \dots \ 1 \ 1 \ 1 \ \dots \ 1}_{n_1 + n_2} \ b$$

出现在带上的就是象输入数据同样的一元形式表示的所需答案。图 1.3 所示的就是进行这样计算的六状态 Turing 机器的程序。