计算机系统结构

西北电讯工程学院

苏东庄　主编

国防工业出版社

计算机系统结构

西北电讯工程学院

苏东庄 主编

国防工业出版社

内 容 简 介

本书讲述计算机系统结构的基本概念、基本原理、基本结构和分析方法。

共分八章。第一章讲述层次结构、程序的可移植性要求、应用与器件的影响。第二章讲述数据表示的确定、指令格式的优化、指令系统的分析、堆栈机器。第三章讲述重迭、流水和分布处理控制方式。第四章讲述总线、中断、通道、外围处理机。第五章讲述存贮层次、地址映象、替换算法、虚拟存贮器、Cache存贮器。第六章讲述故障检测与定位、纠错码、可靠性模型。第七章讲述并行结构、多机系统、并行处理机、多处理机。第八章讲述文本式和图形式硬件描述语言，计算机系统和系统结构的评价。

本书可作为有关专业的教材和科技人员的参考书。

计 算 机 系 统 结 构

西北电讯工程学院

苏东庄 主编

*

国防工业出版社出版

北京市书刊出版业营业许可证出字第074号

西北电讯工程学院印刷厂印刷

*

开本 787×1092 1/16 印张 27 1/8

印刷字数 681 千字 印数 1—11500册

1981年第一版 1981年7月第一次印刷

统一书号：N15034(教-80) 定价：2.78元

前　言

本书系高等院校工科电子类计算机专业统编教材之一。

本书按1978年全国工科计算机教材会议通过的大纲写出“讨论稿”后，除征求了有关专家的意见外，还按照1980年召开的、由来自全国五十二所兄弟院校的老师们参加的讨论班以及清华大学、上海交通大学、西安交通大学和国防科技大学代表参加的审稿会提出的意见进行了修改和补充。

本教材是按“研究软、硬件功能分配以及如何最佳、最合理地实现分配给硬件的功能”这个方向来编写的。目的是使读者对为什么要学习计算机系统结构以及如何学习和研究有一个明确的基本认识。本书力求着重于基本概念、基本原理、基本结构和分析方法；虽然在讲述时联系了实际机器的例子，但并不是围绕某种机型或是过多地从具体实现上的细节来讲述。此外，本书力求反映七十年代以来在系统结构上的重要进展以及八十年代的可能发展。

本课程应在“计算机（组成）原理”、“程序设计语言”和“数字逻辑”等课程之后开设。学生最好有“数据结构”方面的知识。本课程可以在“操作系统”、“编译原理”课程之后，或与它们同时开设。本书的第六、七、八章是按学生不选修“容错与诊断”、“纠错码”、“并行处理计算机结构”和“计算机系统性能评价”等课来写的。

本书是按计算机工程专业的需要来编写的，但也考虑了软件工程专业学生了解计算机系统结构的需要。在编写时还考虑了可供有关科技人员参考之用。

本书是三校合写的。清华大学金兰教授写了第七章；华东师范大学张东韩付教授、唐培艺同志写了第六章；西北电讯工程学院苏东庄付教授写了第一章，张志华同志写了第二、三章，李学干、张志华同志写了第四章，李学干同志写了第五章，马玉珍同志写了第八章。苏东庄付教授还对第二、三、四、五、八章进行了修改和补充，并对全书进行了统编。李学干同志承担了除第七章之外的全书文稿整理工作。

本书的主审单位是清华大学，主审人是房家国付教授和薛宏熙同志，他们对本书的编写始终给予了积极的支持和帮助；参加审稿会的所有代表都提出了很好的修改意见。我们对他们表示衷心的感谢。参加讨论班的老师们提出了很多宝贵意见，我们也对他们表示衷心的感谢。

本书的编写和出版工作得到了西北电讯工程学院教材处和图书馆的大力支持。有关领导部门、四机部教育局以及研究所和兄弟院校的很多领导和同志们对本书的编写给予了热情的鼓励，并提出了宝贵意见。我们在此表示热忱地感谢。

因时间紧迫，本书各章的习题这次来不及附上。

本教材除个别章外，还缺乏教学实践的检验，定会有不少缺点和错误，恳切希望读者予以指正。

编　者

1981年

目　录

第三章 控制方式

第四章 输入输出系统

第五章　存贮体系

主要参考文献

第六章　可靠性技术

第七章 多机系统

第八章 描述与评价

第一章　计算机系统结构的基本概念

本章首先讲述计算机系统的组成和本书所用的计算机系统结构的定义，同时叙述系统结构与组成、实现、分型等的关系；而后简述系统结构近三十年来的进展，并在此基础上分析应用与器件对系统结构的影响。以便在学习后面各章之前，能对计算机系统结构有一个基本的了解。

§1　计算机系统与计算机系统结构

本节先讨论计算机系统的组成，而后着重讲述什么是计算机系统结构及其在构成计算机系统中的作用。最后，分析计算机的分型（例如划分成巨型机、大型机、中型机、小型机、微型机等）与计算机系统结构的关系。

1.1　计算机系统的组成

1.1-1　由软件和硬件组成

大家知道，计算机系统是由硬件和软件组成的。硬件是计算机系统中的实际装置，从使用者来看，它主要指的是机器指令系统以及中央处理机、存贮器、外部设备和它们之间的信息联结。软件则是计算机系统中用户共同使用的一组程序（由机器语言构成）和有关的资料，它变得日益复杂和完善，是使用计算机系统所必须的。软件一般指的是汇编程序、编译程序、操作系统、调试程序、数据库管理系统以及各种应用程序包等。它的作用或是为了便于计算机系统的使用、或是用于提高计算机系统的效率、或是用于扩展硬件的功能等。

一方面，软件随着计算机系统使用等的需要而不断扩大。明显的例子是操作系统的日益庞大，因为使用者日益希望只需发出少数几条命令，就能使计算机系统执行复杂的动作。另一方面，软件和硬件的分界面又是模糊不清，而且是动态地在不断变化着。例如，早期的机器没有乘、除法指令，因此乘、除法运算是通过子程序（软件）实现；后来的机器则把这部分软件“硬化”，即用硬件来实现乘、除法运算。就是到最近，同一种运算或操作（例如浮点运算、字符行运算等）在微型机和一般小型机中是用软件来实现的；而在大、中型机中则是用硬件来实现。又如，一般机器只有整数（定点数）、实数（浮点数）和逻辑数的数据硬件表示，因而复杂的数据结构，如向量、数组等只能用软件实现；但目前已有一些机器把这些软件硬化，使机器具有向量或数组硬件表示及相应的向量与数组运算。其它，如中断处理、存贮管理等的软、硬功能分工也是随不同时期、不同机器而动态地在改变着，这个特点是学习本课程时要始终注意的。

由上可见，软件和硬件在逻辑功能上是等效的。由软件实现的操作（如编译操作、操作系统的基本操作等）在原理上是可以硬化成由硬件来实现；同样，由硬件实现的操作（如某条机器指令所完成的操作，条件码的置定等等）在原理上也是可以用软件的模拟来实现。由

自动机的基本理论可知，只要机器有“相减”及“转移”这两种指令就可用于算题，求解。这样，具有相同功能的计算机系统，其软、硬件功能分配是可以在很宽的范围内变化，如图1.1所示。选择什么样的分配比例，主要应取决于在现有硬件状况（主要是逻辑和存贮的器件状况）下的性能价格比。提高硬件功能的比例可以提高速度，减少所需的存贮容量，但会提高成本，以及降低硬件的利用率和计算机系统的灵活性和适应性；相反，提高软件功能的比例可以降低造价，提高灵活性和适应性，但速度要下降，所需的存贮器容量要增加。目

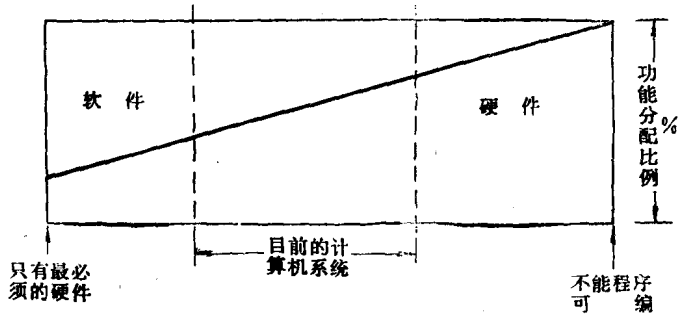

图 1.1 计算机系统的软、硬件功能分配

前，计算机系统的软、硬件功能分配对现有硬件状况、软件算法和解题算法是适应的。但这决不是一成不变的，尤其是在八十年代硬件价格随着超大规模集成电路技术的发展而日益下降的时候更是如此。

那么，对于由紧密相关的硬件和软件组成的计算机系统，我们可以用什么样的观点，从整体上去认识和分析它呢？一种观点是把计算机系统看成是按功能划分的多级层次结构。这和软件的发展过程是一致的。

1.1-2 计算机系统的多级层次结构

大家知道，早先的机器语言（即机器指令系统）由于受电子器件数量限制只能是最基本和简单的，而且使用者必须编写出用二进制表示的程序，这当然很不方便，很不适应于解题的需要及计算机使用范围的扩大。因此，在五十年代先是出现了符号式程序设计语言（汇编语言）。对它，程序员只需用诸如 ADD(加)、SUB、(减)、DIV（除）和MUL(乘)等符号来编制程序，而不必用加、减、乘、除指令所对应的二进制操作码来编制；而且程序员还可以用符号名称，而不必用二进制代码去指明指令或数据的地址。尽管汇编语言程序的每个语句和机器指令基本上仍是一一对应，但却方便多了。这样，对程序员来讲，可以把汇编语言看成是改进了的机器语言，只是没有实际的机器硬件与它直接对应，它的实现是经汇编程序翻译成真正存在的机器语言。我们可以把这想象成是在实际机器级（其机器语言由硬件实现）之上出现新的“虚拟”机器级，其机器语言为汇编语言，如图1.2所示。整个汇编语言（L2）程序（源程序）先经汇编程序变换成等效的机器语言（L1）程序（目标程序），而后再在实际机器级执行目标程序以获得结果。这样一种先把源程序变换成目标程序，而后再在机器上执行目标程序以获得结果的技术称为翻译。

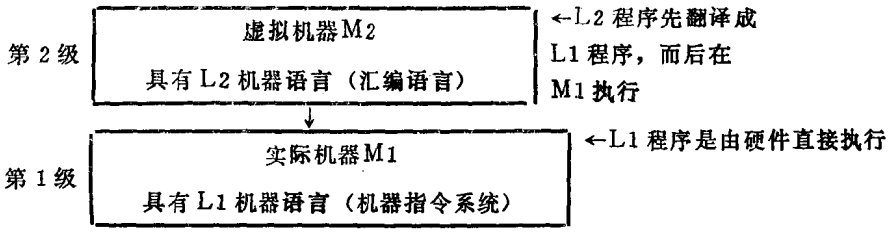

图 1.2　汇编虚拟机器的实现

由于汇编语言的语法、语义结构仍然和机器语言的基本一样，而且和解题所需的仍然差别较大，因此紧接着汇编语言，又出现了面向题目的高级语言，如 FORTRAN，ALGOL 等。同理，我们可以设想成在汇编语言级之上又出现了高级语言级，它的实现是先经编译程序把高级语言程序翻译成汇编语言程序（或是机器语言程序），而后再逐级的实现，如图 1.3 所示。程序员在用高级语言编制程序时就如同面对着其机器语言为高级语言的这样一种虚拟机器。

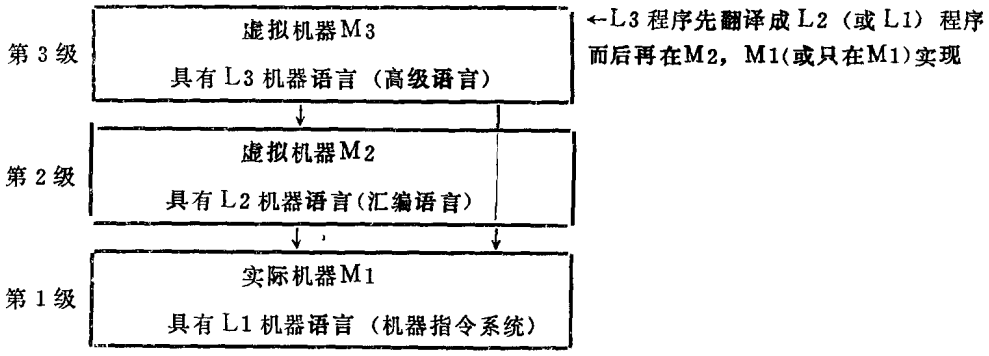

图 1.3　高级语言虚拟机器的实现

这种虚拟机器的垂直式层次概念对于理解各种语言的实质及其实现是有好处的。高级语言程序先翻译再执行的这样一种实现技术虽然已经持续了廿多年，但它是立足于当时的硬件条件（指硬件状况只适合提供简单的机器语言）。显然，随着我们对程序设计语言和编译技术认识的深入以及超大规模集成电路技术的发展和应用，我们应该探索并一定能够找到实现高级语言的更好办法。

这种层次概念还可引伸、应用于机器内部。对于采用微程序控制的机器，大家知道，每条机器指令对应一串微指令（一段微程序），而每条机器指令的实现是通过这一串微指令的执行。这样，我们又可以把图 1.2 的实际机器级分解成如图 1.4 所示的二级层次结构。微指令所执行的是最基本的操作，如寄存器间的数据传送、数据经过加法器或乘法器、主存与寄存器间的数据传送等等。但是，在这里并不是如同汇编语言或高级语言程序那样，先把高一

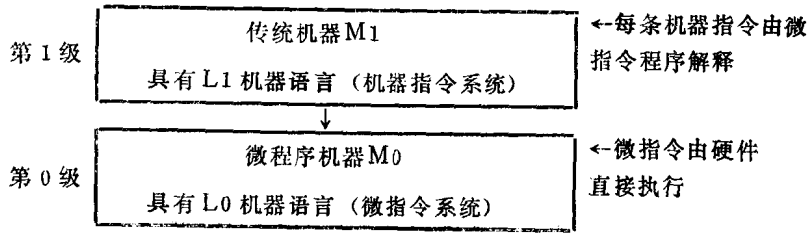

图 1.4　机器指令由微指令程序解释

级机器语言的程序翻译成低一级语言的等效程序，然后再执行它；而是每条机器指令由一串微指令实现，实现完某条机器指令后再由程序内的下条机器指令控制，进入实现它的另一串微指令。这种实现过程称为解释。如图1.5所示。这个过程也可理解为在Mi机器（这里是微程序机器MO）上，用一串Mi机器指令（这里是微指令）的执行来仿真比它高级的机器的一条语句（这里是比MO高一级的M1机器指令）。用微程序解释机器指令意味着把软件技术引入到传统机器的内部，这对提高硬件的规整性以适应大规模集成电路技术的需要是很有益处的。

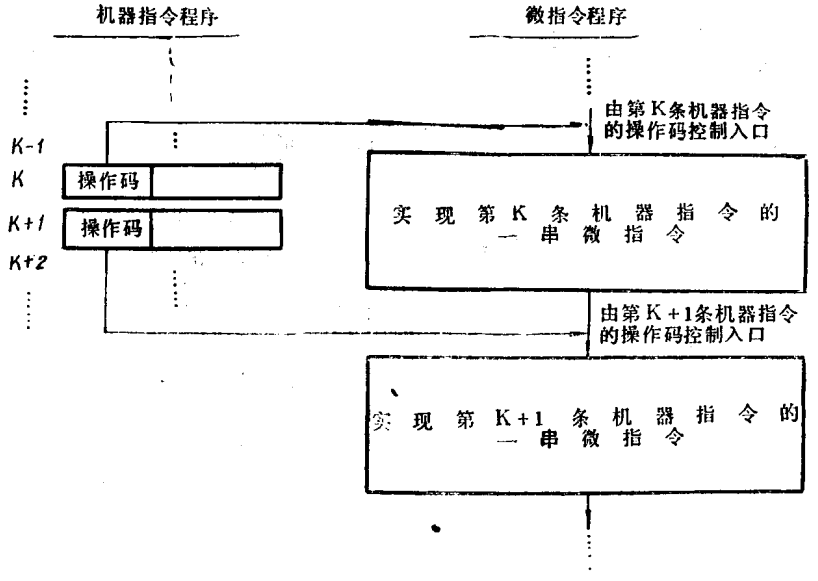

图 1.5　用微指令程序解释机器指令

翻译和解释是语言实现的二种基本技术，对于微程序控制的机器，其高级语言的实现，这二种技术都一定要用到，即先把高级语言程序经编译程序翻译成机器（传统的）语言程序，而后再经微程序对每条机器指令进行解释来实现。还要注意到，由高级语言程序变换成机器语言程序的过程中也不是全部只采用翻译技术，而往往是翻译和解释这二种技术并用。一般是先把高级语言源程序翻译成易于执行的某种中间型式，而后再用机器语言程序解释它（译码及执行）。至于机器语言程序的每条机器指令，那又需经微程序解释。

上面分析了可以把计算机系统看成由高级语言虚拟机器、汇编语言虚拟机器、传统机器语言机器和微程序语言机器所构成的多级层次结构。那么，操作系统在这个层次结构中应处于什么位置呢？这个问题比较复杂。

大家知道，操作系统实质上是传统机器的引伸，它提供了传统机器所没有的，但为汇编语言和高级语言的使用和实现所需的某些基本操作和数据结构，如文件结构与文件管理的基本操作、存贮体系以及多道程序和多重处理所用的某些操作等等。这些操作（可以看成是操作系统的指令系统或称为作业控制语言）和数据结构在已有的机器上大多是经机器语言程序的解释来实现。从上述分析看，操作系统级应是位于传统机器级之上，汇编语言机器级之下。当然，有些机器语言指令，如某些具体的I/O操作指令是要被操作系统“挡”住的，但大部分机器语言指令，如运算类指令等等，却是原样不动地穿过操作系统这级，而也包括在操

作系统级的指令系统之内。

但是，从另一方面看，操作系统还具有提高计算机系统的效率以及去控制汇编、高级等语言的实现和作业的运行等的功能；从这点看，似乎又应该把操作系统置于前述按语言结构划分的层次结构之外，并让它能作用于几乎所有各级。

另外，目前已有一些机器，尤其是今后的机器，它们的操作系统不是用汇编语言编写，而是用高级语言（不是用的面向解题的算法语言，而是用的面向系统软件的高级语言）编写。这样，操作系统级似乎又应该在高级语言机器级之上。

总之，操作系统级的位置是不能简单地指明的，不过从操作系统的基本功能来看，把它置于传统机器级与汇编语言之间基本上是能反映出其主要作用的，因此，用图 1.6 的多级层次结构来认识包括操作系统的计算机系统还是适宜的。图中每级对应一类机器（各有其自己的机器语言）。在这里，"机器"被定义为能存贮和执行程序的算法和数据结构的集合体。各

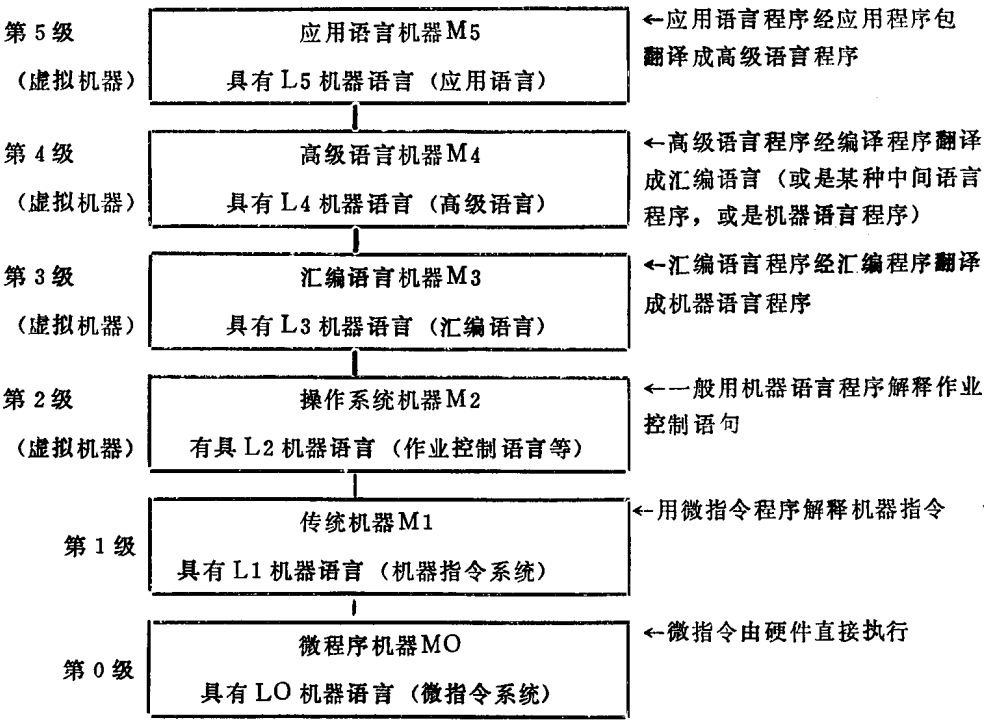

图 1.6　计算机系统的多级层次结构

级机器的算法和数据结构的实现方法不同：MO 是硬件实现；M1 是微程序（固件）实现；M2 到M5 是软件实现。我们称由软件实现的机器为虚拟机器以区别于由硬件或固件实现的实际机器。要注意，机器的实现和题目的得到解答不是一回事，后者是要从你用的那级开始逐级变换直至到达最低级，经硬件实现才能得到；而前者主要表现为如何把该级的程序或是翻译成比它低一级的语言的程序，或是由低一级的程序所解释。因此，相邻级的语言的语法结构差别不要过大，以便于翻译或解释。当然，有的级的程序可能翻译成更低级的程序，例如高级语言程序可能直接翻译成机器语言程序；有的级的语言可能被更低级的程序所解释，例如操作系统的某些命令就可能由比它低二级的微程序解释。操作系统机器的这种例子表明虚拟机器也不一定非得全由软件实现，其中有些操作也可能是固件或硬件实现。循此，为什么

就不能设想直接由微程序或硬件实现高级语言机器或是用固件实现操作系统机器呢？采用何种实现方式，要从整个计算机系统的效率、速度、造价、资源的使用状况等等方面全面考虑。而且要软件、硬件(包括固件)综合平衡，这点一定要认识到。对计算机系统的使用者，除了关心他直接接触的那级机器（或是应用语言机器、或是高级语言机器）的状况外，当然还得关心他所选用的计算机系统是否能提供更好的性能价格比。

对于不采用微程序控制的机器（从目前趋势看主要是大型机和巨型机，因为目前的微程序控制满足不了它们的速度要求），则只是没有第0级，而是直接由硬件实现机器指令系统。除此之外，上述关于计算机系统层次结构的分析对它仍然适用。

也可以考虑在第0级之下再分级。例如有的微程序机器采用二级微程序（称为微程序与毫微程序）去解释机器指令。每条微指令不是直接由硬件执行，而是由比它低一级的毫微程序解释。另外，也可考虑把第0级的硬件再往下分级，如分成部件级、线路级、器件级等，但对于大规模集成电路技术普遍采用的八十年代来说，这种分级的意义是不大的。

上面我们分析了计算机系统是如何由硬件和软件组成的。那么，本课程所要讲述的“计算机系统结构”在计算机系统的构成中又是起着什么样的作用呢？

1.2 计算机系统结构的定义

计算机系统结构（Computer Architecture）这个名词从七十年代以来已被广泛使用，但至今还没有统一的定义，或是说还没有能够确切地定义。这和前述软件和硬件的分界面模糊不清而且在不断变化着以及器件集成度的迅速发展都是有关的。

本书中，计算机系统结构（以下有时简称计算机结构）不是指的计算机系统的结构，而是指计算机的系统结构。

结合§1.1关于计算机系统多级层次结构的分析，可以看出，对计算机的结构，从不同的级看，从各个级的使用者看，其含义、概念和着重点是不同的。“从程序设计者看的计算机结构”和“从计算机设计者看的计算机结构”这二者在概念上差别最大。下面先分别从这二方面进行分析。

1.2-1 从程序设计者看

Amdahl等人于1964年在介绍IBM360时提出，计算机结构是从程序员所看到的计算机的属性，即其概念性结构与功能特性。

这个说法至今仍被不少人所接受，但它并不确切，还需进一步分析。因为按计算机系统的层次概念，从不同级的程序员看，计算机是具有不同的属性。

例如，从使用高级语言（如FORTRAN）的程序员看，NOVA机和PDP-11机，DJS-130机和DJS-180机就几乎没有什么差别，即具有几乎相同的属性；或是说这些机器之间本来存在的差别是高级语言程序员所“看不见”的，所不需要知道的。这种本来是存在的事物或属性，但从某种角度看好象不存在的概念称为透明性概念，它对于理解软、硬件之间的界面有好处。当然，对同一种高级语言，各个厂家实质上都是有些差异的，各有其“方言”及某些不同的规定，一般做不到使某个型号上运行的高级语言程序毫不修改地能在另一型号的机器上运行。而且，在编制高级语言程序时，往往还要考虑所用机器的字长及主存容量

等。但就这些机器的概念性结构及功能特性，却是高级语言程序员所看不见的，即对高级语言程序员是透明的。图 1.7 说明了这点。

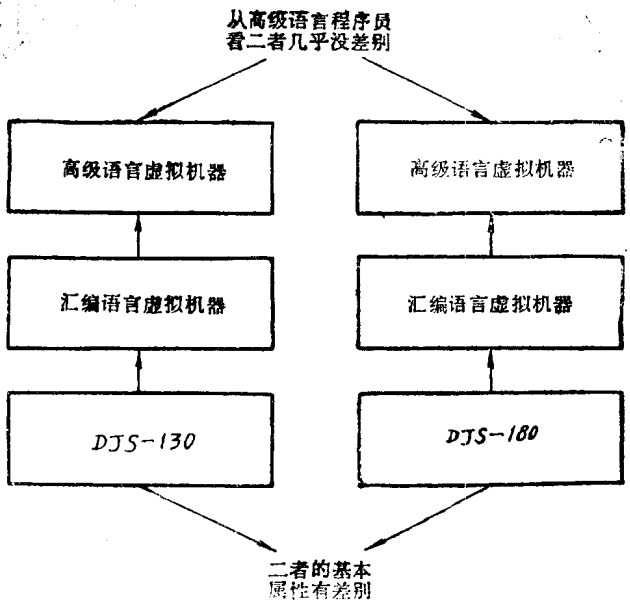

图 1.7　从高级语言程序员看的计算机属性

然而，从机器语言程序员或汇编语言程序员看，上述二种计算机的属性却是不同的，主要表现为指令系统（如操作类型和编址方式）和输入/输出设备连接方式的不同。Amdahl等人当时指的计算机结构就是从这个角度看的计算机的概念性结构和功能特性。

从机器（汇编）语言程序员（也可以包括编译程序的设计者）看，具有相同计算机结构的机器，其机器（汇编）语言程序（也可以包括编译程序）可以通用，而不必顾及各个机器是用什么方法实现其系统结构的。但它们的操作系统却应有差别，以利于发挥具体硬件的结构特点。

一般说，从程序设计者看的计算机结构指的是由机器语言程序员（包括汇编语言程序员和编译程序设计者）所看到的传统机器级(参看图 1.6)的属性。它是程序员为编制出能在机器上正确运行的机器语言程序所必须了解的计算机结构。按此定义的计算机结构实质上指的是软件与硬件的交界面(图1.8)。它的设计者要考虑如何使程序能被机器所识别，如何给程序所用数据编址以及如何表示数据等等；他要定义机器指令的操作类型和格式、数据的类型和格式；还要确定指令间的排序方式和机构，存贮器的最小编址单元、编址方式和保护的方法，以及输入、输出设备如何与机器接口等等。

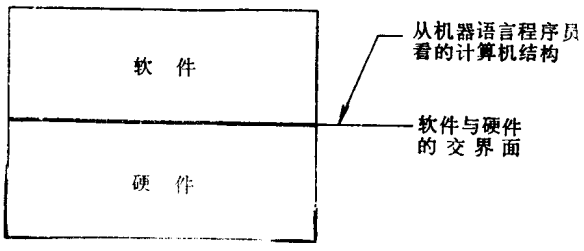

图 1.8　计算机结构是软/硬件交界面

按上述定义，DJS-130 机与 DJS-180 机，NOVA 机与 PDP-11 机当然是有不同的计算

机结构。

还要注意，“从程序设计者看”应该是即包括“从系统程序（如编译程序和操作系统等）设计者看”也包括“从应用程序设计者看”这二个方面。计算机系统结构的设计者一定要从系统程序和应用程序的要求来设计机器。当然，计算机结构的全面定义还应包括下面要讲到的，从计算机设计者看的计算机结构。

1.2-2 从计算机设计者看

从计算机设计者看，对计算机结构的研究应是在现有器件及设备的基础上，如何最佳、最合理地把诸如加法器、计数器、寄存器、堆栈、存贮器、磁盘机、磁带机、输入/输出设备等部件组成计算机，以实现程序设计者所提的设计要求。“最佳、最合理”的主要指标是速度、可靠性与价格。它并不着眼于如加法器、计数器、存贮器和外部设备控制器等的具体的逻辑设计；也不包括具体的工程实现，如信号传输、插件装配、机架结构与散热、通风等，虽然它们对机器的实现，尤其是巨型机的实现起着某种关键性的作用。

从计算机设计者看，计算机结构这三十年来的主要进展是围绕如何提高速度。例如，重迭结构（如输入、输出和运算的重迭、同时执行等）、流水线结构（如浮点运算中的求阶差、对阶、尾加和规格化的流水同时进行等）就是为适应于提高速度的要求而提出来的，这些在下面还要分析。

有些人为了要区分从程序员看和从计算机设计者看的计算机结构，而把前者称为计算机结构，后者称为计算机组成。即计算机结构指的是从程序员看的计算机的概念性结构和功能特性，而计算机组成指的则是这种计算机结构的实现。例如，指令系统属计算机结构；而指令的实现，如取指令、取操作数、运算、送结果等的实现则属计算机组成。这样，具有相同结构（如指令系统）的计算机，其组成却可由于诸如速度要求不同等各种因素而有差异。例如有的机器其指令实现中的取指令、取操作数、运算、送结果等是顺序地执行的，而有的机器为提高速度却使这四个操作重迭地执行等等。按此，从计算机设计者来看，可以认为计算机结构与计算机组成是同义的，在目前的文献资料中这二者有时就是混用的，另外有的人把计算机的组成称为计算机的实现，并进而划分为逻辑实现和物理实现二级。逻辑实现表现为大家所熟悉的逻辑框图，而物理实现则主要表现为所用的装配技术和器件工艺。

1.2-3 要结合起来全面看

本来，正确的途径应是既从程序员角度，也从计算机设计者角度，结合起来设计计算机系统结构，然而，这二者却往往是脱节的。这三十年来，计算机设计者对计算机结构的不少改进往往没给程序的设计带来直接的好处；而程序设计者所看到的计算机结构这二十年来却几乎没有什么变化。也就是说，软件设计者是立足于几乎二十年前的计算机结构来设计日趋复杂的软件。这就使得软件设计所需工时膨胀到了惊人的程度，设计费用昂贵，设计、调试周期很长，维护复杂。有的人甚至说，目前操作系统已膨大到了顶点。另外，软件的运行是要占机器时间的。目前真正用于解题的时间往往占不到机器运行时间的一半；在中央处理机，解题时间所占比例往往还要小。这种把复杂麻烦的负担尽量扔给软件的传统做法是有其历史根源的。因为早期的计算机所用电子器件既昂贵又不可靠，体积又大，当时只有尽可能

地使计算机结构简单到必不可少的地步。但是，到了逻辑器件体积已缩小到几百分之一，存贮器件已缩小到几千分之一，价格已下降几百倍的今天，仍然维持当时的软、硬件功能分配概念，那不仅不合理，而且已越来越成为计算机系统进一步发展的障碍。

下面我们结合计算机系统的设计方法来进一步分析。计算机系统的设计方法可以有“由上往下”和“由下往上”二种思路。

参看图1.6关于计算机系统的多级层次结构，所谓“由上往下”是指的先确定面对使用者那级虚拟机器的基本特性，如基本命令、指令或语言结构、数据的类型和格式等，而后再逐级往下设计，如图1.9所示。当然，每一步的设计还应包括各级的实现（例如，是采用翻译到下一级语言还是采用下一级语言解释等），而且每级的设计还要考虑如何使上一级能优化实现。这样设计出来的计算机系统，对于设计时所面向的这种应用其效能必然是很好的。这就是所谓“专用机”的概念，不过这不是指的过去那种只用（或主要用）硬件来实现某种特定函数的“专用机”。然而，当应用对象或应用范围改变时，这样设计出来的计算机系统，其软、硬件就会不适应，甚至会使整个计算机系统的效率大大下降。更何况软、硬件，尤其是软件的研制周期有时需要几年。因此，当软、硬件都配好时，其应用要求往往又会改变。所以，各级都按应用要求优化设计的情况是很少见的，尤其是硬件，它要求尽可能

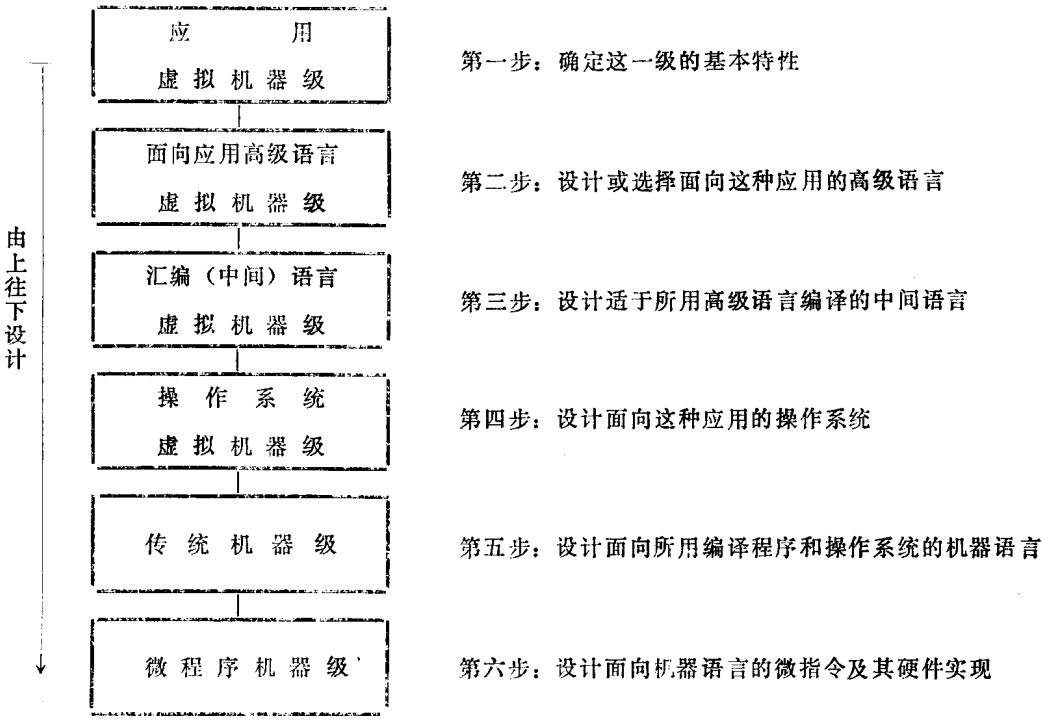

图1.9 计算机系统由上往下的设计方法

大批量生产，生产厂应避免生产批量少的、专门面向某种应用的专用系统。这样，就是采用“由上往下”的这种设计方法，其传统机器级或微程序机器级也一般不是专门设计，而是在可能买得到的各种机器中“选型”，这当然就难真正面向应用来优化实现。