内　部

78006

电子技术会议录

——中小型电子计算机设计技术专辑

第四机械工业部第一研究所

一九七八年五月

中小型电子計算机設計技术专輯

（內　部）

78006

一九七八年五月出版

编辑者：电子计算机科技情报网

出版者：第四机械工业部第一研究所

发行者：北京750信箱21分箱

前　言

四机部计算机专业科技情报网于一九七七年十月在山东淄博市召开了中、小型电子计算机设计技术交流会，参加会议的单位有117个厂、所、院、校，代表196名，提供的会议资料共有80余篇。为了进一步促进中、小型计算机设计技术的交流，更好地推广应用中、小型电子计算机，特将会议资料加以汇编成册，以供参考。由于篇幅所限，大部分资料未能印出，望未选入的单位和作者予以谅解，会议文献請参见附录。鉴于水平有限，选编有许多不妥之处，請批评指正。

目　　录

小型計算机的軟件

一九一五所　方家骅

一、計算机軟件，小型机軟件的特点

1.软件科学和软件工程

软件通俗地说就是程序，一般指一种比较固定，比较通用的程序，它往往用来改进、替代或扩大硬件的功能。广义的说，所有程序都可称为软件，不过实际上很少人把一个解决具体问题的程序称为软件，大家只把在功能上与计算机硬件类似的程序称为软件。

自从电子计算机诞生的第一天起自然已经有了程序，因为计算机离开了程序就变成了一堆“死东西”。但是真正形成软件的概念则是五十年代末期的事。软件的出现尽管晚一些，但是它的发展却很快。当前软件已成为现代科学的一个重要分支，和硬件一样，是当代计算机科学的重要组成部分。

软件科学从它的理论基础看属于近代应用数学的一个分支。它的内容有自动机原理、形式语言、排队论和组合分析、图论等，还有象并行程序（并行处理）和操作系统，数据结构和数据库（数据基地）等也正在形成自己的理论，此外象计算复杂性问题、程序正确性问题、图形处理和模式识别问题、人工智能问题都已成为软件科学的重要研究课题。总的说，软件科学和其他数学分支相比还是一门新兴的科学。但是不论从投入的研究力量，对生产和其他技术部门的影响，还是它的发展前途，都不亚于任何近代应用数学的分支。

软件除了是一门科学外它又是一种工程，即软件工程，这点并不是所有人都了解的。由于软件一般说是由数字和符号所组成的，所以人们很容易把它当作是一种数学。实际上软件和硬件一样是一种产品，而且是一种很复杂的，象手工艺品一样精致的产品。它和通常“硬”产品的区别只在于它主要是人类脑力劳动的结晶。事实上每个具体的软件（如一个编译程序或操作系统）都要经过设计、制作、调试、维护、推广使用这些过程。而对用户来说软件和硬件都是一种“设备”，有时他甚至不知道他得到的是一种软设备还是硬设备。从这个意义上说，把软件叫做软设备更确切一些。因为软件是一种产品，所以在评判一个软件的好坏时，要考虑它的性能/成本比，考虑它的可靠性、可用性和可维性。这就是说，在我们制作软件时应把重点放在工程问题上，而不要把它看成是解一个数学问题（那是软件科学的任务）。软件工程和软件科学的关系正象建筑工程和结构力学、材料力学的关系，我们造房子时当然要懂得和应用这些力学原理，但是研究这些力学原理并不能代替建筑物的设计图和具体的施工，而这正是建筑工程要解决的问题。

2.软件和硬件的关系

由于软件是一种程序，它必须以硬件作为它的基础。软件的功能和规模往往依赖于内存的容量，有无后备存贮器（外存）及它的容量，处理器的速度，指令系统和中断系统，外部设备的种类和数量等一系列硬件的指标。但是软件利用这些硬件基本功能却可完成很多硬件

根本不可能完成的任务。例如语言处理程序（编译程序或解释程序）使计算机可以执行用高级程序语言写出的程序，而目前世界上还没有一台计算机能不用软件直接执行这类语言写的程序。同样通过操作系统可以把一台硬件计算机变成若干台更适应用户使用的虚拟计算机。另外还可以用软件来实现本来用硬件完成的某些功能，例如用小型通用计算机加通讯软件构成通讯控制机（CCP）来代替硬件的通讯控制器（CCU），这样做往往显得更灵活，更经济。

自从计算机进入第三代，硬件不论从速度、容量还是可靠性上都提高了一个数量级，而成本则降低了一个数量级。近年来由于大规模集成电路的广泛使用，这个趋势还在继续发展。这种情况使软件在计算机系统中的地位变得越来越重要。一方面由于计算机硬件日趋庞大和复杂，计算机硬件设计越来越多采用微程序设计来代替原有的组合逻辑，某些复杂的，不常用的功能干脆用软件来代替，这就是所谓系统设计中的软硬结合。反过来，随着大规模集成电路的广泛使用和微程序设计方法的成熟，为了提高执行速度，不少原来用软件(程序)实现的功能改为用微程序来实现，即所谓软件固化，或称为固件。例如某些高级语言如APL语言的解释程序已有改用固件实现，从而大大提高了语言的执行速度。

另一方面，随着计算机性能的迅速提高和成本的急剧下降，计算机应用领域仍在日益扩大和趋于高级化，这些又促使软件向高级发展。近年来出现的计算机网络，计算机复合，高级人一机通讯，图象处理和模式识别，人工智能等都给软件（包括工程和科学两方面）提出新的、更高的要求。

今后的发展趋势看来是：传统的逻辑设计和电路设计逐渐被淘汰，代之以自动化的微程序设计和大规模集成电路设计，即所谓计算机辅助设计（CAD）。随着近年来微处理机的出现，计算机的设计方法还将进一步变革，绝大部分数字逻辑电路不仅用微程序代替，而且可用普通的程序来代替，即所谓面向设备的控制程序。原有硬件逻辑设计、微程序设计和普通程序设计成了计算机设计中的三种可供选择的手段。设计人员首先根据对系统性能/成本的分析决定采用何种方法或几种方法的混合，然后再进行具体设计，这就是以系统设计为中心的软硬一体化设计。

3.小型机软件的特点

小型机作为大型机的对立面，它的主要特点是规模较小，结构比较简单、价格比较低廉。它的应用领域主要是设备控制，如作为计算机终端、数据通讯、实验仪器、机床、生产过程等的控制部件，大部分属于所谓实时控制或实时数据处理，这些特点决定了小型机软件的特点。

小型机软件归纳起来有下述四个特点：

（1）简单、规模小、品种少

这个特点主要是由于它的资源比较少所决定的，特别是它的内存较小，很多情况只有4～8K内存，没有外存或外存较少，限制了软件的发展。

（2）要求软件的质量较高

由于小型机内存容量一般较小，所以要求它的软件尽量少占内存。另一方面，由于小型机主要用于面向I/O设备的控制，它的很多软件都是控制程序，要求它的可靠性很高，又不便于调试。

（3）手工劳动多，人力分散

由于第一个特点，决定了小型机缺少大型机上拥有的完备的编制程序的工具，编译程序、大型操作系统、程序库和数据库、各种实用程序等。在小型机上往往只有一个以汇编为中心

的软件系统，小型机的软件大部分都是用汇编语言编写的，即所谓手编程序的方法。这是一种十分耗费人力的工作。另外生产和使用小型机的单位大部分是小单位，缺少专业的程序员，进一步增加了这种困难。

（4）软件的规格和型号复杂

这点首先是由于小型机硬件的规格和品种繁多，更主要的是由于小型机的应用领域十分宽广，特别当它用于实时控制时，往往需要各自持特殊的应用软件。因此在制作小型机软件时，耗费的人力物力是十分巨大的，由于不能及时制作出合格的软件往往还影响到它的推广使用。

二、软件的分类及其功能

计算机软件通常分为系统软件和应用软件两大类，系统软件又分为操作系统和程序语言的处理（编译程序或解释程序）两部分。

所谓系统软件又称系统程序，它和硬件合起来构成计算机系统，从用户的角度看，它和硬件没有多大差别，可以认为只是实现的手段不同，正如某些自动化设备既可以用机械的方法，也可以用电子的方法来实现一样。系统软件通常又是软件的基本部分，它的功能面向于所有用户或很大一类用户。应用软件则是面向于某些特定的用户。这是一些介于用户程序（又称问题程序）与系统程序之间的程序。

下面我们按照软件的功能把软件分成三大类：

2.1 程序语言和其他编程序工具

直接使用机器的指令系统来编程序（即手编程序）是一件很繁锁的工作，这种程序既不容易编，也不容易改，其他人还很难看得懂。人们要学会编这种程序也较困难。因此最初软件的任务就是要提供一套便于编程序的工具。

分析一个程序设计（或生产）的过程，应包括以下几个阶段：方案的设计和论证，程序的编写，程序的调试（纠错），程序的维护（修改）。而所谓编程序的工具应对上述几个阶段都提供便利。第一阶段工作比较抽象，主要由人的思维活动来完成，因而现有的程序设计工具大都是针对后面几个阶段的。

在小型机上通常提供两类编程序工具：

纸带系统

纸带系统一般包括基本汇编程序、浮动汇编程序、宏汇编程序，输入/输出子程序，纠错程序、文字编辑程序等。这个软件系统最适合于只有4—8K内存、无盘的小型机，外部设备只限于控制台打字机、纸带读入、纸带穿孔三个设备，后两个设备可以用附加在控制台打字机上的慢速纸带读入和纸带穿孔代替。也就是它适合于硬件的最小配置。这个系统用纸带作为程序的主要载体，故称为纸带系统。

纸带系统的核心是汇编程序，它的主要功能是使程序员写出符号程序来代替原有绝对二进制程序，一定程度上方便了程序的写出、阅读和修改。所谓基本汇编就是把一一对应的符号指令翻译成二进制机器指令。浮动汇编则是在基本汇编基础上实现程序的分段编写和联结，这种具有一定独立性的程序段又称为程序模块，浮动汇编先把模块程序汇编成浮动二进制程序，又称目的模块，然后再通过联结程序把若干个可浮动的模块固定，联结成一个可执行的绝对二进制程序，称为装入模块（可加载程序）。这样分二步进行的好处在于它们不需

要在同一时刻连续进行。它允许把不同人、不同时刻编的程序联结成一个新的程序。由于各程序模块是相对独立和完整的，它可以单独进行调试，除了某些全程符号外，它所用的符号（往往对应于某些内存单元）亦与其他模块无关。这样在把一个大的程序分成几个人合作编写时可以减少不必要的冲突（如内存单元的冲突）。使用模块程序也可便于使用别人编写的、已经经过纠错的程序，包括所谓标准程序。因此浮动汇编为程序库的建立提供了良好的基础，当今大部分程序库都以目的模块形式提供。

对小型机来说，由于当前大部分程序仍以汇编形式编写，使用浮动汇编和程序库可以大大减少重复劳动，最适合于编写那些质量要求高、对设备依赖大的实时控制程序。

宏汇编通过把宏处理程序与汇编程序相结合的方法扩大了现有的指令系统。通过定义所谓宏指令的办法可以用一条新的指令来代替若干条原有的机器指令称为宏体，这条宏指令也允许带操作数（地址）通过宏展开，可把这些操作数替入宏体中某些机器指令中的操作数。

使用宏汇编，定义新的宏指令，不但可以把程序写得更简单明了，使用宏汇编，定义一组标准的宏指令，还可以扩大原有的指令系统，使它便于编写系统程序、图形处理程序或其他某个特定应用领域中的程序。使用宏汇编还可以实现不同指令系统之间汇编程序的兼容，或设计一种独立于机器指令系统的通用汇编语言。使用宏汇编，也便于用一个机器来汇编另一个机器的指令，即所谓交叉汇编。上述功能均可在纸带系统的范围内给程序编制提供较大的便利。当然宏汇编和基本汇编相比，需要较大的内存，这些内存主要用来存放宏定义。

纸带系统软件除了包括上述汇编程序外，一般还包括纠错程序和文字编辑程序。纠错程序主要用于调试手编程序，找出程序中的错误。它的主要功能可包括断点符合、逐条打印现场信息，追踪打印等。在小型机上一般采用联机纠错，使用者可以通过控制台打字机灵活地确定断点位置，逐条区间，追踪路线等，特别适合于调试那些逻辑上比较复杂的程序，如系统程序。另有一种所谓符号纠错程序，它的断点位置、逐条区间、追踪路线和单元均可用符号表示，特别适合于在沅程序一级调试汇编形式的程序。

文字编辑程序是用软件的办法来修改沅程序纸带，如增加、删去或替换一段沅程序（可以是一行或一个字符），这种修改一般采用联机形式，有的也允许采用脱机形式，即把修改内容予先穿在纸带上。文字编辑程序除可修改汇编形式的沅程序外，也可用来修改高级语言形式的沅程序。

高级程序语言

汇编语言和纸带系统一般适合于内存较小、没有外存的小型机，对于那些内存较大（如32K以上）、又有外存的小型机，用纸带系统来编程序就显得很不方便了。这时应采用高级程序语言加操作系统作为编程序的工具。

所谓高级语言是相对于汇编语言这类低级语言而言的，最常用的高级语言有FORTRAN、ALGOL、BASIC等，它们大都是一种以数学公式为基础的程序设计语言，特别适合于解决科学和工程计算问题。FORTRAN是使用最早也是最普及的一种语言，同样也是在小型机上广泛使用的一种语言。ALGOL语言又称算法语言，它的语言形式比较严密，功能也比FORTRAN强，但由于它的实现相对说比FORTRAN困难，目标程序（即由沅语言程序翻译出来的机器指令程序）不论在长度还是执行速度上都比较差，故在小型机上应用不广泛。BASIC语言是一种专为小型机设计的语言，它比较简单、易懂，又有ALGOL和FORTRAN的某些特点。另外它的翻译方法（又称语言处理方法）也有不同。FORTRAN和ALGOL一般采用所谓编译法，它通过一个编译程序把沅程序一次翻译成等价的机器指令程序（即目标程

序），以后再由机器执行目标程序。BASIC语言通常采用解释法，即由解释程序逐句解释执行源程序，直接得出结果。编译法和解释法正象外语翻译中的笔译和口译。与编译法相比，解释法的执行速度较低（一般低一个数量级）。其优点则是解释程序比较简单，并且有对话功能，可以采用一边编程序一边调程序的方法，便于及时发现程序中的错误。这对初学程序设计的人有很大好处，目前在小型机上已广泛使用BASIC语言，在解决一些比较简单的工程和科学计算问题时，BASIC语言显得更有效一些。

除了上述几种以科学和工程计算为主的语言外，当小型计用作事务处理（商业数据处理）时，还广泛使用一种产生报表的语言，称为RPG语言。这种语言的主要功能是对穿孔卡上的原始数据进行整理和汇总，根据要求从行打印机上印出报表。这种语言和前面所述几种语言在功能上有所不同，它只须要程序员说明要处理的数据形式以及必要的运算，就能自动生成处理和打印程序，而不要由程序员具体编写计算和打印程序。一般把这种针对某一类问题的语言称为面向问题的语言，而把前述语言称为面向过程的语言。

上述这些语言除BASIC外，原来都是在大型机上使用的，它们的主要应用领域是科学计算和事务处理。而小型机的应用往往以面向设备的实时控制为主，用上述语言编写这类实时控制程序时就显得很不灵便，甚至有些程序都写不出来。另一方面，由于这些语言一般是为大型机设计的，编译程序比较复杂，往往采用多趟扫描。这些都给小型机上的实现带来很大困难。例如基本的FORTRAN和BASIC需要8K以上的内存，ALGOL和扩充BASIC则需要12K以上的内存。由于采用多趟扫描，在程序编译过程中往往要产生若干个中间形式的程序式表格，若没有磁盘等外存时，就要把这些中间形式程序先穿在纸带上，然后再输进去，每进行一次编译都要多次输入输出纸带，给操作增加了很多麻烦。因此如果没有磁盘等外存，上述语言除BASIC外实际上很难使用。即使有了外存，采用多趟扫描，它们的目标程序质量仍不可能很高，和汇编语言相比，它们编出的目标程序往往又长又慢。上述原因造成了至今以实时控制为主的程序往往还要用汇编来编写的情况。

要解决上述矛盾，我们可以从两方面着手：一是设计一种比较适合于小型机实时控制应用的语言，其语言功能尽可能简单，但又要保证适合于实时控制。另一方面又希望它的编译程序也比较简单，但目标程序质量却要求较好。由英国皇家雷达研究院设计的CORAL66语言在相当程度上满足了上述要求。这个语言属于ALGOL系语言，采用分程序结构，定义严谨，结构清晰，同时它又采用了FORTRAN语言的分段形式，便于程序的分段编译和实现与汇编语言等其他语言混合使用。为了适应小型机的实时控制应用，它增加了定点数，表格部分字等数据类型和结构。对原来ALGOL中的数组和循环语句，过程说明和参数类型都加了一些必要的限制和修改，便于用比较简单的编译程序编出高质量的目标程序。据报道其目标程序长度比于编程序只增加25%。该语言还便于使用一趟扫描的编译法。从上述特点可以看出CORAL语言是一种比较适合于在小型机上推广使用的语言。

为了克服小型机资源较少的困难，可以采用大型机上的编译程序来产生小型机上目标程序的办法，即所谓交叉编译。如果加上必要的模拟程序，还可以在大型机上进行模拟调试。这可在一定程度上克服小型机上手工劳动多、编程序效率低的状况。近年来发展起来的高性能小型机，由于它的资源、性能（包括软件）都可与大型机相媲美，再加上它的指令系统等结构格式往往与同型号的小型机兼容，因此它可以作为一个理想的母机，来产生其他小型机上的程序。

2.2 操作系统

和大型机一样，小型机上另一大类系统软件就是操作系统。

操作系统按其功能可分成两大部分：

(1)面向设备的功能。这类功能包括I/O（输入/输出）控制，CPU的调度，MS(主存)的动态分配（虚存的调度），提供这些软件的目的一方面是为了便于用户使用这些设备，使用户不用知道中断系统、设备的控制状态字、处理机状态字（PSW）和处理机工作方式，程序的界或段页对照表等等与硬件密切相关的概念。通过这些软件使机器又回到了第一代计算机那种比较简单、朴实的概念。面向设备的操作系统的另一方面是为了提高机器的效率。计算机硬件从第一代发展到第二、第三代从结构上看主要是扩大了设备之间的并行性，使处理机、存贮器和外部设备的速度更加匹配，从而提高了系统的通过能力和周转时间。要充分发挥硬件设计中上述优点，其关键是增加程序的共行性（并行性）。上述面向设备的功能的主要目的是为了把一台实计算机变成若干台虚拟计算机。所谓虚拟机就是从用户角度看它就象一台真的计算机，即有一台处理机，一定容量的主存，一个控制台和若干个外部设备。而实际上往往是把一台实计算机通过按时间分配或按空间分配的方法变成若干个虚拟计算机或虚设备。这样做的主要优点是为减少各设备的打机等待（空调）。例如在第一代计算机每遇到外部设备进行传输和人在控制台上进行操作时，处理机（CPU）就处于等待状态，从而造成了资源的浪费。

（2）面向编程序和解题的功能。

操作系统的另一大类功能是为了便于用户编程序、调程序、算题，实际操作的灵便化和自动化，它和编译程序、实用程序等合在一起提供了比纸带系统方便得多的编程序工具。

属于这类功能的程序模块有命令解释程序，监督程序（批处理中的作业命令解释），文件库管理程序等等。通过命令解释程序可以便利人一机对话，用控制台打字机代替板键、按钮进行操作，减少操作错误，还可借助中断系统实现不打机操作，这对实现多道程序，即在一个机器内同时（轮流）运行几个程序是必不可少的。监督程序的功能与命令解释类似，前者用于直接控制（联机）方式，后者用于批处理（脱机）方式，在这种方式下用户除了把程序和数据交给机器外，还要把操作命令穿在纸带或卡片上交给机器。一个完整的问题（程序和数据）加上这些控制命令就称为一个作业。按某种次序依次运行这些作业就称为批处理方式。在批处理方式下，操作系统根据用户予先填写的控制命令，自动进行输入、编译、联结、修改、纠错、运算等工作，大大压缩了人的操作时间。既方便了用户，又提高了机器效率。此外，使用大容量存贮器，如磁盘，组成文件库、数据库，不但可以大大压缩费时的输入、输出操作，而且使计算机变成了一个信息（情报）处理中心。可以把成亿个信息贮存在计算机中，并且在很短的时间（例如不到1秒钟）内把它取出来。当然这种大的数据库一般只能建立在大型机上。

操作系统的第二类功能一般需要较多的资源（如大容量的外存）和较高的处理能力（尤其需要高的通道传输速率）。因此它主要是大型机操作系统的一个主要功能。对小型机往往不具备或具备很少一部分这方面的功能。

从层次的角度看，操作系统的第一类功能一般属于内层，构成系统的内核，这些功能不但用户要用，外层的操作系统模块也要用。和面向设备的功能相比操作系统的第二类功能属于外层，这些程序模块甚至可以看成和用户程序处于同等地位。

小型机操作系统按其功能和规模可以分成两类，

管理程序

管理程序又称为执行程序。NOVA机上的无盘操作系统（RTOS）也属于这一类。它的特点是以实现操作系统的面向设备的控制功能为主，着眼于提供程序的并行功能，即所谓多任务功能。当小型机用于面向设备的实时控制时，由于控制对象的并行性，必然要求控制程序也是并行的，所以这种多任务的管理程序最适合于实时控制应用。这类应用的另一个要求是希望操作系统尽可能简单、效率高，希望它占内存少（4K以下），花的处理机时间少。由于每个用户具体硬件配置差别很大，这就要求操作系统的伸缩性、灵活性尽可能大，为此一般采用系统生成的办法：即根据具体设备配置及用户的特殊要求生成一个最合适的操作系统（管理程序）。

管理程序除了实现面向设备的控制功能外，一般也提供了比纸带系统功能略强的编程工具，例如命令解释程序。不过这不是它的重点。

磁盘操作系统

磁盘操作系统是以磁盘为基础的一类操作系统。在这种操作系统中，磁盘一般起两个作用：一是利用磁盘高速随机存取的特点作为内存的扩充，不论是简单的程序复盖还是复杂的虚拟存贮器都是用磁盘作为内存的扩充。由于扩充了内存，首先可以扩大操作系统的规模，使它本身不必直接受内存容量的限制，一般操作系统占内存容量以其十分之一左右为宜，即内存为32K时，只能占3～4K，这就大大限制了操作系统本身的功能。利用磁盘的能力可以在8～10K的内存容量内，容下一个20～30K大小的操作系统。利用这种内存扩充功能还可实现多道程序、程序交换、程序复盖等功能，既扩充了用户程序的空间，又提高了内存的利用率，使机器性能有了大幅度改进。由于这种内外存之间交换的频率很高，一般以使用容量较小（数百K字）传输效率高的固定头磁盘为宜，也可以用大容量磁鼓代替。

磁盘的另一个用途是用作文件库的载体。这种文件库的容量达几兆至几百兆字节。有了这种文件库，既可以用来存放系统程序，又可存放用户程序，包括汇程序，模块程序和可装入程序。有了文件库，用户也可用来存放大量原始数据和中间结果。用于存放文件库的磁盘一般用活动头磁盘或磁盘组。

磁盘操作系统开始也是为大型机研制的，它的主要功能是为了研制程序和解题，提高机器的效率。目前在一些内存容量较大并有磁盘的小型机上也配置了不同规模的磁盘操作系统。在一些高性能小型机上则配置了性能比较完备的磁盘操作系统。配置这种操作系统的目的，一种是为了便于研制程序或进行科学计算，如PDP-11上的DOS操作系统属于这一类。也有是为了改进实时操作系统（管理程序）的功能，使其适合于大型复杂的实时环境。也有兼顾这两方面功能的。NOVA上的RDOS操作系统就属于后一种。不过从实用角度看，操作系统的功能不宜太广泛，这样会使操作系统太庞大，不但研制困难，而且增加了系统的开销，降低了可靠性。在小型机上以短小精悍的、专用性强的操作系统为宜。

除了上述两类操作系统外，在某些小型机上也有人配置了一些简单的分时操作系统。不过这类分时系统的功能比较有限，在各终端上只限于使用某一种会话语言，如BASIC语言，台式计算机语言。也有的只限于某种应用，如用于医院，饭店等部门。这点和大型机上的分时系统有本质上不同。大型机上的终端在某种意义上可以看作一个通用计算机—虚拟机，而小型机上的终端只能相当于一种限定功能的专用机。由于小型机的资源较少，企图在小型机上搞功能完善的分时系统是不合理的。在大型机上搞分时系统的目的是为了把一个大型机分成很多大大小小的机器来使用，小型机的出现则是从另一种途径——即在物理上变成很多计算机来实现这一目的。对小型机倒是应考虑另一条途径，把很多小型机组成一个阵列，达到

较大的处理能力，以代替大型机。

2.3应用软件

应用软件是属于面向某一方面应用的软件，对于这类应用来说，它是一种通用程序。对于所有计算机用户来说，它是一种专用程序。

应用软件的范围十分广泛。有些是为了便于用户使用计算机上的某些设备，如通讯设备和终端，绘图仪，显示器。pDp-11上面向通讯的多路终端管理COMPEX-11，用于显示的图形程序包等就属于这一类，它们可以看成操作系统的扩充。

另一种应用软件实现计算机的某些内部功能，如数控机床控制软件、实验分析软件，医院用程序包，这类软件可能是程序包（即一组完整的子程序），也可能是一个子系统。由于这类软件事实上已包含了大量的应用程序，即所谓功能子程序，例如数控机床控制软件已包含了很多标准零、部件的加工程序，用户只须指定参数或作某些局部修改。

最后，应用软件也可以是一种语言，称为面向应用的语言，也叫专用语言。用于数控机床的APT语言和面向测试操作的TOOL语言都是这类语言的例子。利用这类语言编写程序不但简单，而且直观。不论其形式还是内容都是针对这类应用的。事实上在这类语言中已嵌入了大量功能程序，因此必然可以节省应用程序的工作量。和前面提到的应用程序包相比，它又具有高级语言的特点，使用灵活、方便。

使用应用软件，可以大大节省用户编程序工作。和大型机不同，小型机的数量大，同类用户比较多，通过研制应用软件可以提高程序质量、减少重复劳动，缩短研制周期，是多快好省研制应用程序的捷径。

三、小型机软件的研制、生产和维护

小型计算机，由于它是一种普及产品，再加上它的软件相对来说，规模较小，结构比较简单，因此它的研制和生产容易受到人们的忽视。实际上软件在小型机上的地位并不亚于大型机。例如，用于设备控制的小型机，它的设备控制程序实际上就是硬件逻辑的延伸，它的执行速度直接影响到设备的响应时间，它的可靠性（包括程序的正确性、容错性能等）直接关系到设备的可靠性，它所占内存大小则关系到设备的成本，而它研制和生产的周期则在很大程度上决定了设备或系统的研制、生产周期。另一方面，小型机不论是生产还是应用大都由小单位、小工厂来进行，它们往往缺少有经验的程序员，再加上小型机的软件常常用于编程序编写，这种情况进一步加重了软件研制和生产中的矛盾。造成小型机软件品种不齐、质量低，很多小型机没有高质量的实时控制语言，缺少能用于不同場合的、积木式的实时操作系统。现有的软件也往往存在占内存大、执行时间长、可靠性低等缺点。这种情况不但直接影响到小型机的性能和成本，而且还大大妨碍了小型机的推广使用。由于缺少合适的高级语言等编程序工具，使编写应用程序变得很困难。

要解决上述矛盾，关鍵在于尽快地培养一大批合格的程序员，并从中挑出一部分优秀分子作为系统程序员，他们不光有丰富的系统程序和应程序方面的经验，并且受过计算机科学方面的基础训练，最好能达到大学研究生的水平。只有建立起一支宏大的软件队伍，其人数不是几千人，而是几万人，几十万人，几百万人，才能从根本上解决软件研制、生产中的矛盾，使同样数量的计算机发挥几倍、几十倍的作用，使小型机在生产自动化、管理自动化、武器自动化中起到越来越大的作用，为尽快实现四个现代化作出它应有的贡献。

要培养这样一批软件人员，一方面要加强高等院校计算机科学系和有关专业的教学工作。高等学校，特别是重点大学应以培养未来的系统程序员为主（当然还包括科研人员）。至于一般的程序员应以中等专业学校和七、二一大学为主进行培养。实际上程序员从工作性质上和工人相同，只是他以脑力劳动为主，和其他工种相比需要较多的文化、科学知识。此外有关的工厂、科研单位也可以开办临时的短训班来培养程序员。对程序员来说，主要的不是一定要有高深的基础知识，主要是要有钻研精神和实际工作经验。另外也不应该把系统程序员和一般程序员截然分开，一个好的系统程序员首先必须是一个好的程序员，而一个好的程序员只要加上计算机科学方面的基础训练也可以逐渐培养成一个系统程序员。

要培养这样一批软件人员，另一方面要加强现有软件人员的培养和考核。首先要改变人员使用一般化的做法，应该让那些具有系统程序员水平或者有条件培养成系统程序员的人去负责一个软件产品的设计、协调和联试。除了在实际工作中锻练培养外，也可通过进修、函授、自学等方法提高他们在计算机科学方面的基础知识。要知道计算机科学是一门年青的新兴科学，即使是十年前从大学本科或研究生毕业的人，如果在这期间他没有进行再学习，他对于今天的计算机科学，就会完全象一个新手。因此要定期对现有软件人员进行训练，让他们跟上当今世界上计算机科学的高速前进着的步伐。

要提高软件的产量和质量，还必须加强软件研制生产和维护的管理，建立起一套必要的规章制度。有关的工厂、研究所、车间领导必须懂得：软件和硬件一样，也是一种产品。因此在设计、生产以前必须经过认真的方案论证，因为软件产品很多是新产品，有关设计人员往往缺少这方面经验，这样工作就更加重要，最好能请有关科研单位、高等学校、使用单位中有经验的同志一起参加讨论、集思广益。有些软件产品或科研项目往往由于没有经过认真的总体设计和方案论证，造成“先天不足”。实际工作经验表明，一个正常的软件产品生产周期中，总体设计要占30％到40％的时间。由于忽视总体设计和方案论证，往往造成产品不合使用要求，结构有缺陷或错误、缺乏灵活性和可扩展性等毛病，造成软件生产阶段（编程序和调程序阶段）甚至到维护阶段的多次返工。由于方案不合理，也会影响到软件的“寿命”，要知道由于软件缺乏必要的兼容性和可扩展性，直接影响到应用程序的兼容和扩充，给用户造成大量人力物力的浪费。

软件研制、生产管理的另一个重要环节是成果或产品的鉴定。鉴定的内容一般可包括下述内容：

（1）正确性。主要检查实际的软件与产品的使用说明是否一致。这可通过编写一组实际例子进行检查，这些例子可包括鉴定小组专门编写的和软件制作者编写的两部分。

（2）效率。例如对编译程序，主要指目标程序的质量，包括程序执行速度和程序的长度两方面。对操作系统可测定其吞吐能力和响应时间等。

（3）功能和成本。一方面要看它和同类软件相比功能的大小，另一方面又要看为实现这些功能所花费的价价，如内存容量，研制或生产周期，所费人年数，如调试所费机器的时间等。

（4）可靠性、可用性和可维护性。软件的可靠性可从两种意义上来理解：一是看它已通过了多少题目或运转了多长时间，看它在这段期间出现了多少次软件故障。软件在调试完成后，一般要经过一段试运转（试算）期，考查这段期间的运行纪录可以看出该软件的可靠性。可靠性的另一个含义是软件抵御硬件出错或用户使用不当造成错误的能力。可靠的软件应该使错误不传播开去，并且尽可能准确地报告出错的原因。这个要求对于潜在的软件错误亦应如此。可用性一般指使用是否方便，包括方案和资料等方面。软件制作者应该有一分详

细而准确的使用说明书，还应提供完整的内部结构说明、框图、流程序等，既为了方便使用，也为了便于维护。此外还应提供必要的软件维护手段，例如一组检查的例子及详细说明。

（5）理论上、方法上是否有所创新。最后提一下软件的维护问题。软件不象一般的计算程序，由于其内部逻辑结构十分复杂，通过调试很难保证它一点不错。在使用过程中往往会发现一些新的错误，需要修改。另外，软件和其他产品一样，在设计阶段总不会是十全十美的，通过使用，往往发现一些不足之处。更主要的是随着使用范围的扩展以及要求的提高，用户希望改进和扩充原有功能。软件由于它的主要实现手段是程序，修改起来比较方便，这正是它比硬件优越之处。由于上述原因，软件产品应设专人维护。首先软件制作单位应设立维护小组，负责修改软件，他们可以兼任该软件的推广使用工作。其次，每个机房的维护班子或工厂的调机队亦应有软件维护人员参加。他们的任务是确诊和纪录软件的故障，向软件生产单位的维护小组报告，以便他们及时进行修改。为了便于这两部分维护人员的相互联系，可由生产单位印一些表格发给用户，请他们在发现软件出错时进行填写，并寄回给生产单位。另一方面生产单位的维护小组应定期或不定期的把软件修改内容（包括说明书的修改、程序的修改）印发资料寄给每个用户，必要时还可召开会议、交流软件维护经验。一般这种会可和推广使用会合起来开。只有通过这种经常性的维护，才能使软件的质量越来越高越来越可靠。而那些没有经过精心维护的软件是很难得到推广使用的。

四、小型软件的发展趋势和当前的重点

小型计算机自从60年代初期问世以来，开始配置的软件比较简单，一般以纸带系统为主。这是因为当时的小型计算机还是名符其实的小型机，不论是处理机速度还是内存容量都很低、很小。以后随着小型机的规模增大，软件系统也日益庞大、复杂，特别到七十年代初出现象PDP11/45这类高性能小型机以后，一般都配置了磁盘操作系统和高级语言的编译程序。近年来，随着计算机复合（紧密结合的多机系统）和计算机网络的出现，小型机的软件正在发生新的变化。75年国际信息处理学会（IFIP）召开了一次小型计算机的软件会议，从会上讨论的内容看，重点有二个：一是发展小型计算机上用的系统程序语言和相应的编译程序，以解决软件生产自动化问题。由于这类语言的编译程序比较复杂，广泛采用了交叉编译的方法。如美国DEC公司研制了一个供PDP11系列用的系统程序语言BLISS11，其编译程序是用大型机PDP10写的。实践经验表明，要实现功能比较完备的磁盘操作系统或目标质量较高、功能较强的高级语言编译程序用手编程序是十分困难的。这点对大型计算机是如此，对小型机也不例外。

会上讨论的另一个重点是为计算机网络或计算机复合配置的软件，包括操作系统和程序语言两部分。由于目前广泛使用的程序语言不论是机器语言（指令）还是高级语言，本质上都是一种顺序写出的线性语言，所以它只适合于顺序执行的单个计算机。而计算机网络，尤其是计算机复合，是一种高度平行处理系统，必须为它配置一种描述平行处理的二维语言，例如类似于框图（流程图）的那种语言。如果这个问题不介决，就不可能实现用多个小型计算机代替一个大型计算机的设想。在计算机网络上编程序还要解决计算机之间程序和数据的交流问题，否则就不可能实现网络内的资沅共享。这些都给操作系统和程序语言提出了很多新的要求。

从我们国内情况下，由于我们的发展比人家晚了一些，当前工作的重点应该有所不同。

除了配备一定力量去搞系统程序语言、计算机网络和计算机复合的软件外，还应该先把下面几方面工作抓起来：

（1）把现有的纸带系统配齐。除了汇编、联结、纠错、文字编辑、输入输出子程序外，还应包括宏汇编，（纸带）程序库等有用的内容。并且把这套系统广泛推广使用、维护好。

（2）研制几个比较适合小型机的高级语言，除了多用户BASIC语言外，应该把CORAL语言作为重点来搞。对那些规模较小的小型机，可以搞一些交叉编译程序。

（3）研制几个比较实用的实时管理程序或实时操作系统，它们规模不要很大，开销小一些，能适应不同实时控制用户的要求。为此应采用积木化设计和系统生成的方法。不要企图搞那种“万宝全”的大操作系统。

（4）为了尽快使现有的小型机真正发挥作用，应该重视应用软件和实时应用程序工作，选择几个典型，重点突破，然后逐步推广。应用的问题应该成为当前小型机软件工作的最大的重点。

由于本人在计算机软件工作方面经验不多、水平有限，文中难免有不少错误，欢迎大家批评指正。至于我提出的若干建议，仅供领导和同志们参考。谈出这些浅见陋识，是希望能引起大家进一步讨论，起一抛砖引玉之作用。

微程序工作的开展与通用微程序机器

西北电讯工程学院　苏东庄　李学干　任瑞英

一、对开展微程序工作的一些看法

微程序控制不是象常规机器那样经组合网络式硬联逻辑获得实现机器指令所需的控制信号，而是将存贮技术引入到CPU，将程序技术应用到机器的“构成级”。就是说它是按通常解题程序的概念把机器指令的每个节拍执行的微操作按排成微指令，用一串微指令组成微程序来实现对应的一条机器指令。把所有机器指令相应的微程序都存放在专门的存贮器，即控存中。逐条读出控存中的这些微指令就产生相应的控制信号去控制机器相应的操作，执行完一个微程序就完成了一条机器指令。用存放产生机器操作控制信号的控存来代替常规控制器的组合逻辑，把存贮逻辑引入到了控制环节。

采用微程序技术带来的主要好处如下：

（1）　用规整的存贮器结构代替不规整、复杂的硬联逻辑，使整体结构简化。各个机器指令的算法反映为不同结构的微程序，而这只表现为控制存贮器存放的内容不同，并不影响到控制存贮器的输入、输出接口和外部连接等的“硬”结构。这就便于修改与调整，设计和生产也规整方便，系统的可扩充性和可变性好，便于适应不同用途成为多目标系统，便于经济地实现复杂的指令系统。

（2）　有利于大规模集成电路的应用，器件厂可为不同机器的控制器生产统一的高集成度的通用片子，再由用户根据需要装入所需的微程序来实现不同的操作和指令系统，从而可能用较低价格实现复杂功能的体系结构。

（3）　能采用存贮器已有的各种校验技术来对控存进行诊断、校验、利于维修、调机，提高可靠性。

但是，微程序控制由于每拍都需访问控存，需要相应的微指令读出时间，因此较之硬联组合逻辑，在速度上会有损失。

要注意到，采用微程序技术并不是目的，而只是手段；衡量一个机器的好坏不能光看是否采用了微程序，而是要看通过采用微程序获得了什么，是否获得了上述好处。

这样，例如，用分离二极管控存矩阵构成的微程序控制器就并不一定比用组件构成的硬联组合逻辑控制器好。同样，对于控存所用器件的分析也如此，就可靠性和便于生产来讲，变压器型只读存贮器就优于分离二极管控存矩阵，只是考虑到要过渡到中、大规模集成电路及调机需要，后者才可采用。

目前，国内已经有了多种采用微程序的机器，积累了不少经验，应该在此基础上不断总结提高。下面就如何开展微程序工作，谈谈几点粗浅的想法。

1. 要设法解决好小型机“可变长操作码”给微程序带来的不利因素

小型机例如NOVA，PDP—11等，字长较短为16位，不能很好容纳指令系统所要求的操作码、地址位移量、地址模式、寄存器编号等诸部分，从而需采用可变长操作码。

这就影响到微程序的入口处理，即如何根据机器指令的要求和相应状态进入相应微程序的首条微指令地址。

对定长操作码来说微程序入口比较简单。可以直接把操作码作为入口地址，例如，操作码占据8位，就将这8位作为微程序入口地址，一次进入。当然，也有将8位分成4、2、2，多拍入口，然而这只是因为操作码各部分分别对应于地址形成方式、运算类型等等不同的功能，在微程序流程上本来就需分拍使用。

对可变长操作码情况就不同。若操作码为从8位可变到16位，则采用8、4、4的入口方法（如HP—21MX），对操作码为8位时只需一拍入口，而对操作码为16位时，就需经3拍才能进到首条微指令地址，从而使速度下降。

小型机功能又不完全反映在操作码上，如果考虑到地址模式入口、特殊功能入口，问题就更复杂了。

事实上，不止如此，在中间过程中也会不止一次要迁到按操作码转移。

对范围不大的可变长操作码（例如最大为8位）是可以直接把操作码最长的8位送入以ROM构成的编码器中形成首条微指令地址（这种方法这个ROM的利用率低）。而对16位可变长操作码，用ROM构成编码器就会使其容量大到2^{16}的极不合理的程度。

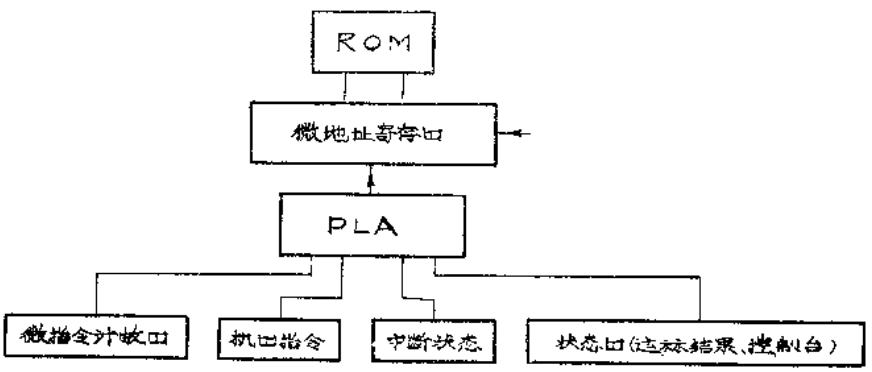
看出，不论用“多拍软入口”或是“编码ROM”方法都不能很好解决可变长操作码的微程序入口问题。

73年以来，可编程序逻辑阵列（PLA）的出现为这个问题的解决提供了硬件基础（器件进展是推动微程序技术的基础）。PLA具有存贮矩阵和地址译码矩阵都可编的能力。既能保持某些地址与内容，即输入与输出的一一对应，也可对某些地址根本没有对应的任何单元，或多个地址对应同个单元，或一个地址对应多个单元（可参看771所“微电子学与计算机”1977年第1、2期合刊上的“程序逻辑阵列PLA的基本概念及应用”一文）。

由于PLA的入口数可多达20～30位，就完全能处理好全字范围的可变长操作码的入口问题。而且，还可把“中断状态”“控制面板”“机器状态”等等也加到PLA的输入端，从而经PLA不只是形成入口地址，而且还可用于形成或控制流程中下字址的形成。

用PLA形成微程序入口，大致有以下三种方式：

（1）LSI—11方式（75年）



PLA作编码器用，其输入宽度超过指令最长操作码位数，而输出宽度只需等于控存地址位数。

（2）AM2911方式（75年）

这种方式，把上述PLA分成二个，一个专用于形成入口，另一个则主要起过程控制作