电子计算机情报
（内 部）

一九七三年八月出版

NOVA小型计算机软件介绍

NOVA小型计算机系列的总体和硬软件概况，本刊1973年第2期已有专文介绍。由于在同类小型计算机中，NOVA系列的软件比较完备，它的某些特点还可供大型计算机的分时终端参考，本文拟根据目前能看到的资料，再进一步介绍软件情况。全文分四个部份。第一节讲纸带格式，是一点必要的技术细节。第二节分类列出软件目录，附以简要说明。第三节介绍这些软件的若干共同特点和它们彼此间的关系。第四节选几种常用的软件，作更具体的叙述。全文都从使用者而不是软件工作者的角度讨论。

一、纸带格式

字长16位的NOVA系列计算机，允许从4096字到32768字内存容量，配备或多或少外部设备的不同组合。软件的齐全程度当然取决于系统构成。但是指令组和基本软件系统，则是全系列各型号计算机通用的。本文介绍的软件，适用于内存12288字，配有光电输入机、纸带穿孔输出机、电传打字机，没有外存的小型组合。

由于没有外存，全部软件都记录在纸带上。即使有磁盘操作系统，纸带副本也是产品的必要组成部份。因此我们就从纸带的格式讲起。NOVA系列采用八单位纸带，共分四种类型：

（1）ASCII（美国标准信息交换代码，包括26个字母、十个数码、标点符号、算术符号和一些控制信号等）纸带用户写出的全部源程序和厂家提供的一部份子程序是这类纸带。它们可以直接用电传打字机印出、供阅读和校对。

（2）绝对二进制纸带　按照给定的绝对地址输入内存的机器语言纸带，每两行代码拼成一个字。纸带上的信息分块穿孔，信息块有四种：数据块、重复数据块、作废块和结束块，各块前面有标志。数据块由不多于16个字的代码组成，前面加标志，字数，地址和检查和等信息。重复数据块使一段相连的内存单元置相同的内容。作废标志后面的整块信息不输入内存。结束块在纸带最后，块中可指明该程序入口地址，输入后即自动启动，故结束块又称起始块。

（3）浮动二进制纸带　在内存中的位置可以浮动的相对地址机器语言纸带。信息块共分九种：浮动数据块、标题块、入口名称块、外部名称块、外部位移量块、库头标志、库尾标志、局部符号块和结束块。它们的意义将在叙述过程中逐步明确。

（4）程序库纸带　这是前后加有库头、库尾标志的一段浮动二进制纸带，中间可有若干种独立的子程序，每个子程序带有自己的标题和入口信息。输入这种纸带时，只有已被前面输入的程序调用过的子程序，才按标题选择读入内存。

这四类纸带我们以后用字母A(ASCII)、B(Binary 二进制)、R(Relocatable 浮动) 和 L (Library 程序库) 表示。

顺便介绍一下输入机停机和代码和检查问题。由于纸带上可能出现任何符号，不允许指定一种符号作为输入机停机标志。输入机接受一切符号，只根据中央发来的信号停机。上述四类纸带都由特定的软件接受，A类纸带由各种汇编、解释或编译程序接受，B类由二进制引导程序接受，R和L类由浮动引导程序接受并分配内存。相应软件根据纸带上的结束信息发出停止输入信号。

A 类纸带输入时可进行偶校验。其它三类纸带的每种信息块中都有两行"检查和"，它的作用是把该块的代码和凑成零（$=0 \bmod(2^{16})$）。可见这些纸带不可能由人工穿孔产生，而须靠某种软件输出。这种代码和检查方式比数总代码和更有效，而且在快启停输入机上可以立即确定代码和出错的信息块。

二、软件概述

这一节里分类列出全部软件的名称。用"*"号标出的软件还要在第四节中具件介绍，其它软件只作简单说明。

（一）引导程序类

（1）初始引导程序。有两种方案：或是从面板搬键拨入 13 条指令，或是按"自动输入"键从两块 256 位的 MOS 只读存储器中把 32 条指令调入内存并自行启动。

（2）二进制引导程序。这就是普通的输入管理程序，可由面板第 0 位搬键指定由光电机或电传打字机输入。这种由初始引导程序接受的纸带格式特殊，不同于前面提到的四类。程序全长 120_8 字，通常保存在内存尾部，机器工作过程中不破坏它。

（3）* 浮动引导程序。这是功能复杂得多的软件，它要在输入过程中完成扩展汇编程序未作完的事情，实现最后代真、建立独立汇编的各个分程序之间的联系、为查错程序准备符号表等等。

（4）扩展的浮动引导程序。专用于输入 FORTRAN IV 和 ALGOL 60 目标程序。

（二）纸带处理程序类

（1）穿孔输出程序。将指定的内存段落穿成二进制引导程序可以接受的 B 类纸带。

（2）磁心比较程序。比较纸带和内存中已有信息，发现差异时将两者在电传打字机上印出。

（3）* 纸带编辑程序。实现各种源程序 A 类纸带的修改增删，把不同的纸带编排到一起，输出统一的一条 A 类纸带。

（4）宏编辑程序（Macroeditor）。具有纸带编辑程序的全部功能，使用者还可以定义宏寄存器中的字符串，在编辑过程中按寄存器名称引用。

（5）程序库文件编辑程序。在磁盘操作系统中配有此种软件，用以产生 L 类程序库纸带。

（三）符号语言类

（1）* 汇编程序。将符号语言的源程序汇编成绝对二进制目标程序（B 类纸带）。

（2）* 扩展汇编程序。比汇编程序多四种功能：浮点、浮动、程序间联系和条件汇编、产生 B 类或 R 类纸带。

（3）FORTRAN IV 和 ALGOL 60 专用扩展汇编程序，把编译程序产生的符号语言纸带汇编成 R 类目标纸带。

以上三种汇编程序都是两遍扫视的，每遍扫视要令源程序纸带从输入机上通过一次。内存中只保存符号表，不保留源程序，目标程序也随时穿孔输出。因此，可以用很小内存汇编出相当长的目标程序。

（4）* 浮点解释程序，共有三种。基本浮点解释程序只占用 1024_{10} 字和 64 个工作单元。扩展浮点解释程序增加了七种初等函数和定点输出格式，占用 1856 字和 110 个工作单元。浮动浮点解释程序与扩展浮点解释程序功能全同，但使用相对地址，可在内存中浮动。浮点数在内存中占两个字，即 32 位。

（5）算术程序库。为方便符号语言使用者，提供约三十种算术子程序，其中包括单双倍位乘除、双倍位加减、二——十进制转换、绝对值、随机数等等。大部分是 A 类纸带，使用时须借助纸带编辑程序编入用户的源程序，一部份于程序是浮动二进制的 L 类纸带，可根据调用情况选择输入。

（6）* 查错程序（Debugger）。小型计算机只有面板，没有控制台，特别是在分时工作制度下，更不允许一个用户停机检查程序状态。

符合停机、查看与修改各寄存器和内存单元等操作都交由软件实现。NOVA 软件中有大小不同的三种查错程序，其中简单的查错程序 I 只有 192 条指令，而查错程序 II 允许同时安排八个符合点，并可部份地使用程序中的符号（如语句标号）查看内存。这些查错程序都不具备动态的追踪示踪功能。

（四）对话语言 BASIC

对话型的算法语言 BASIC，简单易学，使用方便，功能虽不及 ALGOL 60 等编译语言，但也能处理相当复杂的数值计算问题。对于小型计算机和大型机的分时终端，这是一种很适用的软件。NOVA 软件系统中，根据整机组合情况，提供几种水平不同的 BASIC 解释程序。

（1）单用户 BASIC。只有 4096 字内存和一台电传打字机的最小组合，即可使用此程序。

（2）单用户扩展 BASIC。要求 12288 字内存。它增加了字符串操作、矩阵运算语句，通道处理能力，并能调用手编子程序和 FORTRAN IV程序库中的子程序。

（3）分时 BASIC。最多允许 16 个同时工作的用户，视内存大小（8192 字或 16384 字）、有无磁盘，提供几种不同方案。没有磁盘时，可根据用户要求，实行内存平均或不平均分配。有磁盘时，允许一个用户独占内存或多用户公用内存两种工作方式。用户可以在磁盘上建立自己的文件库，软件中考虑了保密措施。

（五）编译语言类

ALGOL 60 和 FORTRAN IV 两种基本算法语言的编译程序，几年来不断修改和扩充。早期版本的各种限制逐渐取消，最近的扩展编译程序功能与大型机的语言接近，目标程序的执行效率当然低于大型机。这种扩充功能很有利于用小型机为大型机编译和调试程序。

（1）FORTRAN IV 编译程序和程序库。这是与 1966 年美国 ANSI FORTRAN IV 标准一致的编译程序，在某些方面还有所扩充。允许定义双倍位（64 位）实数，复数和双倍位复数。允许在源程序中混入符号语言段。动态组织内存，标准子程序允许递归，同时很容易写出递归和重入的程序。数组维数最多达128，变量名称最长 31 个字符。允许使用不必单独编译的内部函数分程序和子程序。目标程序在一次扫视中编译并穿孔输出，但输出的是符号语言纸带，尚须再用 FORTRAN IV和 ALGOL 60 专用的扩展汇编程序汇编成机器语言的目标带。编译过程中检查约八十种语法错误，执行过程中检查 28 种错误。

FORTRAN IV 程序库包括一百九十余种子程序，其中部份是为用户提供的，部份由编译程序调用，组成目标程序的某些成份。

上述编译程序要求至少 12288 字内存。对于 8192 字内存的系统，另有一种加有限制的 FORTRAN IV 编译程序。

（2）ALGOL 60 编译程序及程序库。这个编译程序基本上实现了 1962 年 ALGOL 修改报告的全部规定，但编译过程要分成两步，由两个独占内存的编译程序分别实现，第一步输出一条中间纸带，第二步输出符号语言带，还要再汇编一次，才能得到最终的目标程序。复杂的编译手续，由于功能扩充而得到补偿。除了标准的 ALGOL 语言要求，还增加了若干重要功能：可以定义外部过程和变量，各类变量（整数、实数、复数、逻辑、字符串）都可以定义为简变、数组或过程，允许字符串或二进制按位操作，而且提供了一些这类操作的标准函数，语句标号可以是下标变量，可以编辑或删除文件，多倍位运算最多可拼 15 个字，保证十进制 60 位有效值。使用这种扩充了的算法语言，可以较容易地设计出符号运算程序、绘图或显示程序等非数值计算软件。

以上两种算法语言的编译和使用手续，在备有磁盘操作系统情况下，都可大为简化。

（六）操作系统类

（1）独立操作系统（Stand-alone Operating System 简称 SOS）。这就是普通的输出输

入管理程序，算法语言目标程序就要在 SOS 协助下执行。

(2) 实时操作系统(RTOS)。为备有实时钟和要求实时中断处理的外围设备的用户提供。

(3) 磁盘操作系统(DOS)。

(4) 实时磁盘操作系统(RDOS)。

所谓操作系统，是一种高水平的调度和管理程序。它把软件的输入、输出、编译、执行，外部设备的启停，通道间的信息交换，中断要求的处理等等，都看作某种"任务"。任务可能处于四种状态之一(见图 1)，由操作系统根据优先顺序、中断要求、执行情况和操作员的键盘命令来统一调度。操作员不直接干预机器的工作，他只是通过键盘命令来查看或改变各种任务的状态。在备配磁盘外存情形下，很容易在操作系统控制下实现多种任务分时工作。

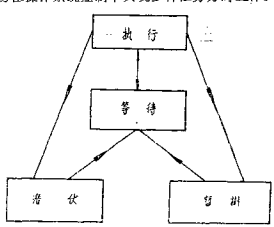


图 1 任务的四种状态

NOVA 系列的各种操作系统正在不断发展中，最近宣布为新的 NOVA 840 机配备了两道程序并行的磁盘操作系统，作为大型机远距终端的操作系统等。详细描述尚未见到。

(七) 诊断程序类

这不是本文重点，我们只点一下名：

(1) 逻辑检验：据称可以逐门检查运控逻辑工作情况。

(2) 算术检验。

(3) 内存检验。

(4) 指令执行时间检验：以电传控制电路中的时钟为标准，检查主振和各条指令的执行时间。

(5) 光电输入和穿孔输出检验。

(6) 电传检验。

(7) "练习"程序(Exerciser)。这是一个综合性的自检程序，技术说明书要求每天试通十五分钟。

诊断程序的设计要根据相应硬件的实际情况。因此多数诊断程序为一定的硬件专用，不是整个 NOVA 系列通用的。厂方为模数和数模转换、磁盘、磁带、数据通讯、笔绘仪、显示设备等都设计了诊断程序，随设备提供给用户。

三、软件特点与相互关系

NOVA 系列的软件考虑到在操作系统、分时、多级中断的实时处理等各种情况下工作，既照顾到 4096 字内存的最小组合，又允许发展扩充为更大的系统。因此，许多软件有一些共同特点。我们试图概括为以下几条：

(一) 浮动　许多软件提供绝对地址和浮动地址的两种方案，程序库都是浮动的，而且根据调用情况，选择输入。

(二) 重入(Reentrant)　同一个程序可以在多重中断情况下多次重入。为此要保证程序本身在执行过程中不变，而且每重用户指定自己专用的工作区。这样的程序可以放到只读存储器中，实现软件"硬化"。浮点解释程序是一个典型例子。

(三)递归　广泛采用动态组织内存，由软件实现各类后进先出区。标准子程序和标准过程大都允许递归，用户也很容易设计出递归子程序。

(四) 对话　许多软件通过电传打字机与操作员对话，规定输入输出设备，提醒操作员要作的事情，检查错误和确定自己的工作方式。我们在第四节中给一些实例。

(五) 自带必要的输出输入管理程序　除了二进制引导程序外，内存中没有"常驻"的程

序，每个软件都带有自己可能用到的输出输入程序，独占内存以进行工作。

（六）为 4096 字内存的最小组合准备了一套比较完备的基本软件，包括汇编、浮点解释、纸带编辑、查错、单用户 BASIC 语言等。

第二节中介绍的多数软件，都包括在磁盘操作系统中。没有磁盘时，它们以纸带形式提供，它们彼此间的关系这时反而更清楚一些。图 2 是主要软件相互关系的示意图。看了这幅图就可以更为突出地感到 BASIC 类型对话语言的方便之处；编辑、修改、查错、输入输出、标准子程序等等都包含在一套解释程序之中，用户可以在工作的任务阶段查看程序、工作区和中间结果，并进行修改。在电传打字机前工作一两个小时，就可以编写并动态调试出相当复杂的程序。总的工作效率高往往补偿了解释程序执行速度低的缺点。特别在多用户分时工作制度下，对话型的语言允许在不影响主机高速工作的同时，充分发挥人的能动作用。对于小型计算机和大型机的分时终端，这都是值得注意的发展方向。

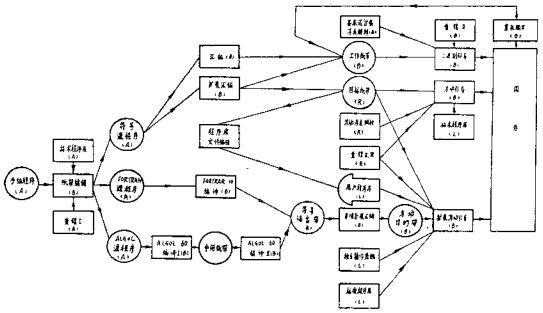


图 2　主要软件关系示意图

说明：方框中为软件，圆圈里是用户工作过程中产生的纸带。A，B，R，L 四类纸带见本文第一节

四　软件举例

在这一节里以几种常用的软件为例，具体说明它们的功能、用法，并给出键盘命令，错误标志等一览表，读者可借以形成较完整的概念。

（一）汇编程序

汇编程序把用符号语言写成的源程序翻译成机器语言程序，输出是绝对地址的 B 类纸带，同时可根据程序员要求印出清单和符号表。

符号语言中允许使用由整数、ASCII 符号拚写的名字、表达式、伪指令和符号表示的机器指令。

名字是由字母或句点开始包括字母、数码或句点的一串符号，且最长不超过五个字符。用户宜尽可能避免用句点开始的名字，以减少与各种软件保留的名字相重复的可能。名字可表示变量单元，语句标号或寻址用的位移量。

表达式由整数、名字和＋、－、＊、／和＆（逻辑乘）！（逻辑和）等符号组成。表达式一律从左向右算，算术运算无优先区别。表达

式常用以实现地址运算。

符号表示的机器指令（以下简称指令），共分三类22条。附以各种辅助操作符号，可形成两千多种有细致差别的指令。兹分别叙述之：

（1）访内指令类，共六条，见表一。

表一中m＝0，1，2，3，代表累加寄存器ACm。E是有效地址，它由位移量D和寻址方式X决定，见表二。

根据表二确定的是直接地址。如果指令中出现符号@（它在一行中的位置不限），则为间接地址，要以(E)作为有效地址。如果E所在行中又有@或最左位为1，则又是间接地址，以此类推可形成间接地址串。

表一 访内指令

操作	指令形式	说明
LDA 取数	LDA m,D,X	(E)→ACm
STA 存数	STA m,D,X	(ACm)→E
ISZ 增跳	ISZ D,X	(E)+1→E，如(E)=0则跳过下一条，否则顺序执行。
DSZ 减跳	DSZ D,X	(E)−1→E，同上
JMP 转移	JMP D,X	无条件转移到E
JSR 转子	JSR D,X	无条件转移到E，并将返回地址→AC3

表二 四种寻址方式

X	寻址方式	D	有效地址	备注
0	零页地址	000～377	E＝D	X＝0可不写明
1	相对地址	−200～177	E＝(指令计数器)＋D	可缩写为·±|D|
2	以AC2为变址器	同上	E＝(AC2)＋D	
3	以AC3为变址器	同上	E＝(AC3)＋D	

表三 算术逻辑指令

指令	意义	移位	进位	分枝	意义
COMn,m	$(\overline{ACn})\rightarrow ACm$	L左移一位	Z进位置0	SKP	无条件跳一条
NEGn,m	$-(ACn)\rightarrow ACm$			SZC	进位为0跳一条
MOVn,m	$(ACn)\rightarrow ACm$			SNC	进位非0跳一条
INCn,m	$(ACn)+1\rightarrow ACm$	R右移一位	O进位置1	SZR	结果为0跳一条
ADCn,m	$(\overline{ACn})+(ACm)\rightarrow ACm$			SNR	结果非0跳一条
SUBn,m	$(ACm)-(ACn)\rightarrow ACm$			SEZ	进位或结果为0跳一条
ADDn,m	$(ACn)+(ACm)\rightarrow ACm$	S左右八位互换	C进位求反	SBN	进位和结果皆非0跳一条
ANDn,m,	$(ACn)\wedge(ACm)\rightarrow ACm$				

（2）算术逻辑运算指令类，共八条，见表三。

每条算术逻辑运算指令都可以判断进位和结果情况，实现分枝。不用移位、进位、分枝等辅助操作时，可以不写。如果指令中出现符号井（它在一行中的位置不限），则运算结果并不送到ACm中去。例如指令

SUBZ井 1，0，SZC

在(AC 0)<(AC1)时跳过一条，但不改变两个累加寄存器的内容。

（3）外控指令类，共八条。

首先要定义外部设备。任何最多具备三个

字长 16 位的缓冲寄存器（简称A，B，C），两个工作状态触发器："忙碌"(B)和"完成"(D)，能由中央发出启(S)停(C)信号，必要时还可接收一个控制脉冲(P)的电路，可以看成NOVA计算机的一种外部设备。他们由相同的外控指令控制，区别仅在于指令中要注明该设备的两位八进制代码。对于通用的外部设备，代码已经标准化，只要给出该设备的英文缩写，汇编程序就代真成相应代码。下面是外控指令的典型例子：

NIOS TTO 无输出输入，启动电传输出；

NIOC TTI 无输出输入，停止电传输入；

SKPBN PTR 如光电输入机"忙碌"，跳一条；

SKPDZ PTP 如穿孔输出机未"完成"，跳一条；

DIA m, CDR 从卡片输入机的A寄存器取数到 ACm 中；

DOB m, LPT 从 ACm 送数到宽行打印机的B寄存器。

还有一组特殊指令，形式上是以中央处理机(CPU)作为设备代码的外控指令，一併列入表四。

伪指令是程序员用以将某些信息通知汇编程序的语句，它们不被汇编成机器指令。表五列举了汇编程序和扩展汇编程序的伪指令。有些伪指令要稍加说明：

(1) 字符串的首尾值得注意。串中不出现的任何字符（除回车、空格、制表、逗点、换行、分页、作废等符号）都可用作括号，因此字符串中可以有引号〃。作废等特殊符号其实也可以写入字符串，只须把它们的ASCII代码放在尖括号〈 〉中即可，尖括号本身因而不能直接出现在字符串中。如果程序员不记得某个符号的ASCII代码，则可用引号〃加上该符号本身表达，留待汇编程序去代真。例如要在某个

表四 特 殊 指 令

指令形式	缩写符号	意　义
DIA m,CPU	READS m	把面板开关内容读入ACm
DICC 0,CPU	IORST	全部外部设备复位
DOC 0,CPU	HALT	停　机
NIOS CPU	INTEN	允许中断
NIOC CPU	INTDS	禁止中断
DIB m,CPU	INTA m	响应中断，最优先设备代码取入 ACm
DOB m,CPU	MSKO m	送出 ACm 中的掩码，重排中断优先顺序
SKPBN CPU		中断触发器置位则跳一条
SKPBZ CPU		中断触发器复位则跳一条
SKPDN CPU		掉电触发器置位则跳一条
SKPDZ CPV		掉电触发器复位则跳一条

标号为 COMMA 的单元中存逗点的 ASCII 代码，只要写

COMMA: 〃, 汇编为 000054

借助这种手段，可以把字符串GOTO〈IN〉存放为

·TXT @GOTO 〈74〉IN〈76〉@

或 ·TXT #GOTO 〈〃〈 〉IN〈〃〉〉#

字符串通常从字的右半放起，但也可改为从左半存放。或再改回来。这就是伪指令·TXTM 的功能。

(2) 与符号表有关的伪指令是NOVA汇编程序的一个特点。对于经常出现的指令，可用等价关系(=)定义，如

G1=177

T=SUBZ#1,0,SZC

等等，程序中再遇到时可只写 G 1，T…。这些符号不进入符号表，只对包含它们的这一个程序有效。可以用伪指令把类似的关系纳入符号表，例如：

.DUSR G 1=177

.DALC T=SUBZ# 0,0,SZC

只要不遇到伪指令 .XPNG，不重新输入汇编程序，这些符号就留在符号表中（而且可以穿

表五 汇编程序伪指令表

	伪 指 令	说 明
基本	.LOC 表达式	表达式应能在第一次扫视算出(≤32767_{10})，指出内存开始地址。
	.BLK 表达式	空出一块内存。
	.RDX 表达式	表达式值在2到10之间，规定进制。八进制可不指明。
	.EOT	纸带结束，汇编程序停在00006单元，换纸带后按"继续"即可。
	.END 表达式	源程序结束，表达式值为核程序启动地址，可不写。
	.TXT 字符串	存字符串，串中不出现之某字符加在串前后作括号，奇偶位置零。
	.TXTF 字符串	同上，奇偶位置1。
	.TXTE 字符串	同上，偶校验。
	.TXTO 字符串	同上，奇校验。
	.TXTM 表达式	改变字符存放顺序。表达式不等于零时，从左向右存放。
	.DMR	定义一条访内指令。
	.DMRA	定义一条要用累加寄存器的访内指令。
	.DALC	定义一条算术逻辑运算指令。
	.DIO	定义一条输出输入指令。
	.DIOA	定义一条要用累加寄存器的输出输入指令。
	.DIAC	定义一条要用累加寄存器的指令（由机器指令第三、四位规定)。
	.DUSR	使用者定义一个符号
	.XPNG	取消符号表中除固定符号(如伪指令)外的全部符号，包括指令。
扩展	.ZREL	表示以下指令要求零页浮动内存。
	.NREL	表示以下指令要求非零页浮动内存。
	.TITL	定义程序的标题
	.ENT	说明入口名称
	.EXTN	说明外部名称
	.EXTD	说明外部位移量
	.IFE 表达式	条件汇编，当表达式为零时，.ENDC以前程序被汇编。
	.IFN	条件汇编，条件与上条相反。
	.ENDC	条件汇编程序段尾标志

孔输出一条带有扩充符号表的汇编程序纸带，表尾单元可从000004看出)，对于此后汇编的程序都是有效的。不仅如此，这样定义的新指令还可有自变量，例如写

T m, n

就表示(ΔCn)<(ΔCm)时跳一条的指令。

应当指出，汇编程序使用的全部指令缩写也是以这种方式定义在符号表中的，因此也可以被取消。使用伪指令.XPNG时要当心这一点。

汇编程序和扩展汇编程序的键盘命令和错误标志在表六和表七中给出。

(二) 扩展汇编程序

扩展汇编程序具有基本汇编程序的全部功能，并在以下四方面有所扩充：

(1) 浮动 可以使用零页浮动单元（伪指令.ZREL)，非零页浮动单元(伪指令.NREL)或绝对地址（伪指令.LOC)。其输出也可能是R类或B类纸带（第一次扫视后在电传机上通知程序员纸带类型)。

(2) 程序间的联系

扩展汇编的源程序可以引用在其它程序中定义的数据、地址或变量（用伪指令.EXTN

表六 汇编程序键盘操作表

软件印出	回答	说明	备注
IN: (输入设备)	1 2 3 4 5	电传机纸带输入，无奇偶校验。 同上，有奇偶校验。 光电输入机，无奇偶校验。 同上，有奇偶校验。 电传机键盘输入，无奇偶校验。	有奇偶校验时用↑代替出错字符，并在该行前印出错误标志I。
LIST: (打印源程序和目标程序清单的设备)	1 2 3 4 5	ASR 33 型电传机（软件模拟分页，制表信号）。 ASR 35 型电传机 宽行打印机 穿孔输出机，然后在 ASR 33 上脱机打印。 同上，在 ASR 35 上脱机打印。	
BIN: (输出二进制目标程序的设备)	1 2	电传穿孔，无局部符号。 穿孔输出机，无局部符号。	回答前应将纸带装妥
	3 4	电传穿孔，带局部符号。 穿孔输出机，带局部符号。	仅扩展汇编程序有此功能
MODE: (工作方式)	1 2 3	第一遍扫视 第二遍扫视，输出目标程序纸带 第二遍扫视，输出程序清单(包括符号表)	回答前应将纸带重新装妥
	4	第二遍扫视，输出目标纸带和程序清单	BIN 和 LIST 用同一设备时禁用
	5	输出符号表	算法语言专用汇编程序无此功能

说明占全字长的外部名字，用 .EXTD 说明占半字长的外部位移量或外部名字)，也可以把本程序中的名字宣布为将被其它程序引用的入口（伪指令.ENT)。一个名字必须在某一个程序中说明为 .ENT，才能在其它若干个程序中说明是.EXTN 或.EXTD。由于符号查错程序和程序库编辑程序要引用程序的标题，可以用伪指令.TITL 为一段程序加标题。

标题和各种名字都以压缩形式放在符号表中。压缩办法是引入包括 26 个字母、十个数码和句点等符号的四十进制，每个名字就成为一个五位的四十进制数，只须二进制 27 位即可存放。每个名字占两个字，还剩下五位存放其它信息。标题、入口和外部名字，用户定义的各种局部符号，都可以信息块形式输出。浮动引导程序要借助这些信息，在输入各个独立汇编的程序时实现最后代真。

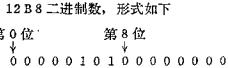
(3) 数的定义　可以直接写十进制数（不必用.RDX 事先说明）浮点数、双倍位整数和二进制数。下面的几个例子足以说明问题：

31　八进制整数；

31．0或.31E2浮点数，汇编后占两个字；

−1D双倍位整数，汇编为两个全1单位；

12B8 二进制数，形式如下



(4) 条件汇编

源程序的一段，可用伪指令.IFE（或.IFN）和.ENDC分出，如果第一个伪指令后

表七　汇编程序输出的错误标志表[注]

标　志	错误类型	举例或说明
A	地址错误	LDA 0,400 或 ISZ .+317
(A)		表达式计算结果不是绝对地址，NREL 或 ZREL；ZREL 超出零页，NREL 超出寻址范围。
B	非法字符	LA$L；LDA 1,23；标号中不能用 $
C	冒号错误	A+2:　　；冒号前不能有表达式
D	进制错误	.RDX 12　；只允许十以内进制
E	等式错误	REG=3+B　；其中 B 未定义时
F	格式错误	ADD 2　　；算术逻辑指令要求两个操作数
(G)		内部或外部符号说明有错
I	输入错误	奇偶错或某字符不对
(K)	条件汇编错误	.IFE 或 .IFN 的参数在第一次扫视中算不出来或两者发生嵌套
L	输入地址错	.LOC　-1
M	重复定义	某符号在多处定义
N	数字错	C77；7A
(N)		数字太大或太小，不能用浮点表示
O	溢　出	LDA 4,LOC；没有 4 累加寄存器
P	扫视错误	某符号在第一、二次扫视中数值不同
Q	本行中有问题	.+.END
(R)	表达式错	表达式计算结果与寻址规则不符，或表达式中出现 NREL 和 ZREL 符号的不正确组合
S	表　溢	符号表溢出
T	符号表伪指令错	14+.XPNG
U	符号未定义	. . . . .
X	符号串错	LET："CB;"后只能有一个字符 3+.TXT　；.TXT前不能有表达式
(Z)	表达式错	表达式中有非法符号，如外部符号、伪指令、双倍位数或浮点数等

[注]　括号中是扩展汇编程序增加的标志。

的表达式在第一次扫视后算出的数值为零（或不为零），则这一段程序被汇编，否则被跳过。

(三)浮动引导程序

浮动引导程序在输入过程中完成扩展汇编程序未作完的事情，分配零页和非零页内存，对外部名称和外部位移量代真，形成入口表，对程序库纸带作选择输入并检查若干错误。这个程序较长，输入完毕后自行清除，不继续占用内存。从以下操作顺序可以看出它的大致功能。

浮动引导程序本身由二进制引导程序输入后立即在电传机上印出。

SAVE=

操作员用一个八进制数（如400）回答并回车，浮动引导程序就在内存最后保护400个单元，然后从后向前存放符号表。如果只回车不给数字，则只保护二进制引导程序不被破坏（约200_8个单元）。接着印出"号，等候操作员规定工作方式，操作员以一位数码回答，其意义如下：

1——从电传打字机输入一条 R 类或 L 类纸带；

2——从光电输入机输入一条 R 类或 L 类纸带；

3——为非零页浮动指定一个开始地址，拨在面板开关中（不用工作方式 3 时，从 400或 440 单元开始）；

4——通知浮动引导程序输入全部局部符

号，这是查错程序Ⅲ所要求的，引导程序应立即回答S；

5——印出当前内存状态，包括零页第一个空单元(ZMAX)，非零页第一个空单元（NMAX)，符号表头（SST）和符号表尾（EST)，算法语言专用浮动引导程序这时还印出FORTRAN IV所用COMMON变量所占单元个数；

6——印出入口表，其中未定义的名称前有标志U，重复定义的名字前有标志M；

7——浮动引导程序复原，当输入过程出错，不能继续时，可用此准备重输入；

8——输入完毕，浮动引导程序自行清除，可根据入口表开始正式计算；

9——只印出入口表中未定义的名字（只有算法语言专用的浮动引导程序允许此种回答方式)。

浮动引导程序的查错功能可以从表八中看到。

表八　浮动引导程序错误标志表

错误标志	错误性质	说　　明
CS	严　重	检查和错
DO		位移量溢出（实际用000代真）
DN	严　重	外部位移量与已定义的外部名字冲突
IB		非法信息块（标志不对）
ME		重复定义的入口（只有第一次定义有效）
MO	严　重	内存溢出
NA	严　重	出现负地址
ND		外部名字与已定义的外部位移量冲突
OW	严　重	企图重写已输入内容的内存单元
TO		输入时间过限，通常发生在没有结束块或库尾标志时
ZO	严　重	零页溢出

〔注〕　出现严重错误后，输入过程不能继续，必须排除错误，重新开始。

（四）纸带编辑程序

用以修改各种源程序纸带，并把不同的纸带编辑成一条统一的纸带。纸带上的信息按分页符号分页，页内按回车符号分行。软件定义一个编辑缓冲器（实际是全部可用的内存)，每次输入一页文件，由一个想象中的指向字符间隙的“箭头”指示增补删除的位置。有屏幕显示设备时，这个箭头当然可以具体实现。操作员使用各种键盘命令（见表九）实现输入输出和修改编辑。若干键盘命令可组成一个命令串一起执行。如

*YP$$ 输入一页并立即穿孔输出；

*B3L1K1T$$ 将箭头移到页首，跳过两行，删除第四行，印出下一行。这个命令等价于

*2J1K1T$$

编辑程序整个采用人机对话方式工作。输入后即询问输出输入设备，要奇偶校验否。得到回答后就印出*号，等候操作员发出键盘命令，如果命令有错，就印出？？*，等候新命令。出现内存溢出，键盘命令寄存器满，字符串找不到等情形，就印出完整的文字信息，因此不必记忆错误标志表。

这里应介绍一下在编辑程序控制下穿孔。只要机器时间允许，在编辑程序控制下为源程序穿孔是很方便的。实际只须用键盘命令I将源程序作为字符串送入内存，每页都可随时印出查看、修改无误后再穿孔输出。编辑程序还可以模拟大型电传打字机的某些机械动作，使穿孔更为方便。例如大型打字机的“制表”键(TAB）使字头每次空走八格，小型电传打字机（如75行的ASR 33）只能发出TAB电信号，没有相应的机械动作。纸带编辑程序接到信号后，用指令模拟制表动作，收到与大型打

表九　纸带编辑程序键盘命令表

类别	命　令	说　　明
输入	Y	输一页到编辑缓冲器中，原有内容被清除，箭头指向第一字符前。
	A	输一页到编辑缓冲器中，接在原有内容后，取消原有分页符号，箭头指向第一字符前。
移动箭头	B	令箭头移至页首。
	Z	令箭头移至页尾。
	nM	令箭头越过几个字符，n>0 时向页尾方向，但最多到页尾，n<0 时向页首方向，但最多到页首。
	nL	令箭头越过 n 行，n>0 时向页尾方向越过 n 个"回车"符，n<0 时向页首方向越过 |n|+1 个"回车"符，箭头停在新到行首。只打 L 时，箭头移到本行首。
	nJ	令箭头从页首向页尾方向跳几行（与命令串 Bn-1L 等价）。
删除	nD	箭头移动方式同 nM，越过的字符被删除。
	nK	箭头移动方式同 nL，越过的各行被删除。
增补	nI	取 n 右端 7 位，将相应 ASCII 符号增到箭头前面。
	I 字符串 $	将字符串补入箭头所在处，箭头移到新增字符串后面。
字符串操作	S 字符串 $	从箭头所在处向页尾寻找字符串，找到后箭头停在该字符串后，找不到时印出信息，箭头重新回到页首。
	N 字符串 $	从箭头所在处向页尾寻字符串，找不到时将本页穿孔输出，输入下一页继续寻找……。
	Q 字符串 $	同上，但不穿孔输出。
	C	改字符串。C（第一串）$（第二串）$$，将第一串换成第二串。C 字符串 $$，则将该字符串取消。找不到时印出信息。
输出	T	在电传打字机上印出编辑缓冲器全部内容。
	nT	从箭头所在处打印出几行。
	P	将全部缓冲器内容穿孔输出，并在最后加分页符号。
	nP	从箭头所在处穿孔输出几行，并加分页符号。
	PW, nPW	与 P，nP 同，但不加分页符号。箭头位置不变。
	F	穿一个分页符号。
	nF	穿 n（n≤100）时中导孔。
	nR	执行命令串 PY 几次。
其它	:	印出编辑缓冲器中行数。
	=	印出编辑缓冲器中字符数。
	.	印出箭头所在行号。
	CRTL C	取消键盘命令，停止执行键盘命令。
	CRTL T	停止输入，清除内存。
	Rubout	取消最近打入的一个字符（被取消字符将再印出一次），可多次连用。
	ESC	使用一次，表示字符串结束，连用两次，表示命令结束，每次印出 $。

字机相同的效果。

（五）浮点解释程序

浮点指令是符号汇编语言的一部份，形式上很接近指令，也由汇编程序或扩展汇编程序代真为 16 位字长的信息，由浮点解释程序取去解释。

使用浮点解释程序（简称"浮解"）时，应指定工作区（基本浮解用 64_{10} 个单元，扩展浮解用 110_{10} 个单元），把工作区首地址送入

007单元。此外还应提供输入一个字符和输出一个字符的子程序，把入口分别置入040和041单元。这时就可以使用指令FETR进入浮解，然后用指令FEXT退出浮解，这两条指令之间应全是浮点指令。一个程序之内可以多次进入和退出浮解，但第一次进入之前应用指令FINI准备好工作区，它的功能是清除工作区并为浮点子程序的嵌套建立后进先出区。

与机器指令相似，浮点指令也分三类。它使用由软件定义的四个浮点累加寄存器FAC0～FAC3。下面的表十、十一应和表一、三对照阅读。

表十 浮点访内指令

操作	指令形式	说明
取浮点数	FLDA m, A	(A,A+1)→FACm
存浮点数	FSTA m, A	(FACm)→A,A+1
浮点增跳	FISZ A	(A)+1→A，如(A)=0则跳过下条浮点指令，否则顺序执行。A中为定点数。
浮点减跳	FDSZ A	(A)−1→A，余同上。
浮点转移	FJMP A	无条件转移到A处的浮点指令
浮点转子	FJSR A	同上，并将返回地址→AC3
取数到AC3	FLD3 A	(A)→AC3 用于在浮点指令段内形成循环等
存AC3	FST3 A	(AC3)→A 在浮点指令段内存返回地址等
浮点变定点	FFIX A	将(A,A+1)处浮点数变为双倍位定点数
定点变浮点	FFLO A	将(A,A+1)处双倍位定点数变为浮点数

表十一 浮点算术逻辑指令

指令	意义	备注	分枝	意义
FMOV n, m	(FACn)→FACm	可加字母U取消规格化	FSLT	结果<0.时跳过一条浮点指令
FPOS n, m	|(FACn)|→FACm		FSLE	〃 ≤0. 〃
FMNS n, m	−|(FACn)|→FACm		FSGT	〃 >0. 〃
FNEG n, m	−(FACn)→FACm		FSGE	〃 ≥0. 〃
FRND n, m	(FACn)舍入→FACm		FSNR	〃 ≠0. 〃
FADD n, m	(m)+(n)→m		FSZR	〃 =0. 〃
FSUB n, m	(m)−(n)→m		FSKP	无条件跳过一条浮点指令
FMPY n, m	(m)*(n)→m	结果一定规格化	注意浮点运算舍入误差，慎重使用FSNR和FSZR	
FDIV n, m	(m)/(n)→m			
FHLV n, m	(n)/2.→m			

浮点算术逻辑指令（表十一）运算结果通常要规格化。但除了最后三种指令外，可在操作缩写后加字母U取消规格化。这类指令也可以用符号#，使运算结果不送回FACm中。

下面是浮点算术逻辑指令的两个例子：

```
FSUB#    1, 0,      FSLT
FMOVU    3, 0
```

前面已经提到，用户应提供两个输出、输入一个字符的子程序。浮点输出输入指令就套用这两个子程序来和具体的输入输出设备发生关系。这类指令只有三条：

FDFC n 浮点十翻二，从输入设备送入FACn；

FDFCI n 同上，每次先印出标志F；

FFDC n 浮点二翻十输出，从(FACn)送到输出设备。

此外还有三条特殊指令：

FIC 2　　(AC 2)＋2→AC 2；

FIC 3　　(AC 3)＋2→AC 3；

FHLT　　停机。

以上介绍了基本浮点解释程序的全部指令。扩展浮点解释程序增加了七种初等函数(FALG, FATN, FCOS, FSIN, FTAN, FEXP 和 FSQR)和一种输出格式。初等函数指令的用法很简单：

FALGn, m；ln(FACn)→FACm

新增加的定点数输出格式FFDCF n要求在使用前将总字符数W(包括空格)和小数点后的位数D送入工作区的指定单元，其位移量分别为W＝136，D＝137。

基本浮解占用内存5600～7577单元，扩展浮解占用4100～7577单元。此外还提供一个使用相对地址的浮动浮点解释程序。使用时应先将工作区、输入子程序和输出子程序入口分别以名字WSA，GETC，PUTC送入零页浮动区，同时应在有关程序中说明为入口(.ENT)。浮解准备和进入浮解改用指令FINT和FENT，它们应说明为外部名称：

.EXTN　FINT，FENT

(六) 查错程序

我们以最简单的查错Ⅰ为例。这个只有192条指令的软件，允许用户以键盘命令指定一个符合点。发生符合后用户可以查看累加寄存器和内存单元的内容。办法是用键盘命令“打开”某个寄存器，它的内容立即用八进制印出，如要修改，只需在电传打字机上打入新内容再“回车”即可。如不需修改，可在查看后用“回车”关闭之。

表十二　查错1键盘命令表

命　令	意　义
地址 B	在该地址安排符合点
nA	打开累加寄存器(n＝0-3，n＝4为允许中断触发器和进位)
P	从符合点继续执行
地址 R	从该地址启动
地址/	打开该地址单元
换行	关闭被查看的寄存器，顺序打开下一个寄存器
回车	关闭被查看的寄存器

查错Ⅱ允许安排四个符合点，查错Ⅲ允许安排八个符合点。被查看的内容可按各种形式(指令符号、八进制数、ASCII代码)印出和修改，可以规定两次符合之间通过某个点的次数，可按指定的掩码顺序取出内存某些位与规定的标准比较。这两个程序还包含了穿孔输出程序的全部功能，以便把经过修改的程序以纸带形式输出。它们的键盘命令比表十二复杂得多，此处从略。

(中国科学院物理所二室207组编)