第4卷 第1期 电 子 学 通 讯 Vol. 4, No. 1
1982 年 1 月 JOURNAL OF ELECTRONICS Jan., 1982

关于阿尔法微型计算机系统初始化命令文件的讨论和改进*

江 钧 基
(中国科学院电子学研究所)

提 要

本文通过对阿尔法微型计算机的系统初始化命令文件的探讨，提出缩小操作系统的方法，指出原文件中存在的一些问题并提出了改进的措施．改进后的文件可使该机在带有5个终端和5个作业的情况下，操作系统的大小小于16K字节．提出了两种消除因用户使用 MEMORY 0命令而引起的机器工作不正常的方法．给出了一个新文件的实例及其实验结果．此外，还从实用的角度出发，论讨了这种计算机最大可带的终端数和作业数．

一、引 言

美国的阿尔法微型系统公司（Alpha Microsystems）生产的阿尔法微型计算机系统（Alpha microcomputer system）是一种多用户、多终端、分时工作的计算机系统．该机适用于企业管理和教育试验．在采用了分组转换（bank switching）的工作方式后，虽说内存量可扩展到1024K字节（1K为1024，下同），但每个用户仍只能取得64K字节，还包括共用的操作系统在内．当它用于科学计算的场合，往往感到内存量太小，因此，在不影响计算机性能的前提下，应尽量设法缩小其操作系统．

操作系统是在计算机启动时，按照系统初始化命令文件（system initialization command file）的要求生成的．在这个文件中要定义作业、终端、打印机和磁盘的名称，分配给它们内存量，规定内存板的地址，设置比特图（bit map）和放入公用程序等等．它不仅直接决定了操作系统的大小，而且还对计算机工作的质量有影响．因此，如何编写好这一文件，是值得重视的一个问题．

二、系统初始化命令文件中的一些问题

图1是某一阿尔法机系统内存的分配情况．这个系统包括2个 5M 字节的硬盘、2个软盘、1个打印机、5个终端、5个作业以及3块64K字节的内存板．BANK0—BANK4

* 1981年5月7日收到．

是5个可转换的内存组；固定的内存区由操作系统M和可共用的内存 S 所组成．根据文献[1]所述，区域 S 只能为 BANK0 内的用户所利用，但实际上我们发现通过某种手段，其它各组内的用户也可接触到这一部分．计算机在工作时，各组在不同的时间内和固定的内存区相接，共用一个操作系统，这就是所谓分组转换的工作方式．

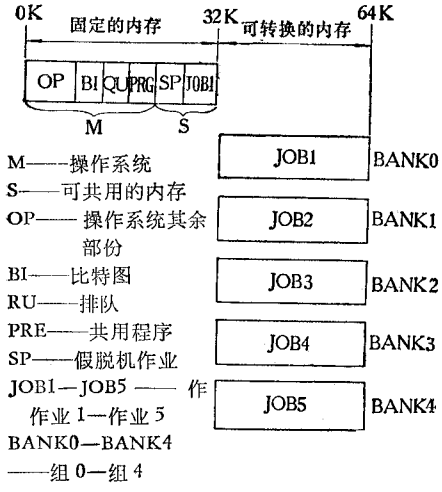

图1 按阿尔法机原有的系统初始化命令文件设置的某一系统的内存分配图

Fig. 1 Memory map of a certain system set up by the original system command file of ALPHA MICRO Computer

因为硬件结构的限制，内存的转换点只可在16K、32K或48K处．在图1的系统中，内存的转换点是在32K处．

附录1是形成这一分配的系统初始化命令文件．它是根据系统盘上所带的原始文件（版本4.4b）修改而成的．

这个文件执行后建立起的操作系统的大小为18254字节，其中包括：终端的驱动程序 TRM·DVR 和软盘的驱动程序 DDA·DVR 以及软盘和硬盘的比特图 BI．为使打印机工作而设置的假脱机作业 SP 占据4000字节，它紧接着 BI，完全位于可共用的内存区 S 内．因为操作系统M的大小已超过16K字节，所以内存的转换点只可设置在32K字节处．作业 JOB1—JOB5 分别安排在组 BANK0—BANK4 内，并与终端 TRM1—TRM5 相联．JOB2—JOB5 各占32K字节（实际上还要减去高地址端供I/O接口使用的256字节）．JOB1 约占43K字节，有一部分位于 S 内．

该系统在使用时发现有下列问题：

（1）操作系统大于16K字节，但是按图1的分配方案，却很难使操作系统小于16K字节．

（2）在用户输入 MEMORY 0 命令后会使计算机工作不正常．

MEMORY 命令的用法是：任一用户在自己的终端上打入 MEMORY X（X是一个数字）后，即可把自己的内存量变为 X 字节，但只能由大变小．在打入 MEMORY 0 命令后，内存中的内容全被清洗掉，同时该用户所占内存量变为零．如果以后再打入某些命令或其它字符，则可取得该用户或作业所能取得的最大内存量．

表1是用 SYSTAT 命令从终端上观察到的系统工作状态的一部分．JOB2—JOB4 的内存量本来都是像 JOB5 那样为 32512 字节，其起始地址也都是 100000（8进制）．但 JOB4 在打入 MEMORY 0 后，内存量变为零；JOB2 先打入 MEMORY 0，再打入命令 DIR，JOB3 先打入 MEMORY 0，再打入一条不存在的命令 ABCD 后，它们的内存地址都向低端扩展到43516处，允许取得47026字节的内存量．假定这时 JOB2 和 JOB3 中的任一个用命令 LOAD 把程序从盘上调入内存，则另一个作业的用户也能接触到这些程序，或用命令 DEL 把它们清洗掉．这时如果在 JOB4 内输入命令或其它内容，也会立即把 JOB2 或 JOB3 内存里的内容清洗掉，此外，除 JOB1 外，只要任一个作业的内存地

表 1　用附录 1 的系统初始化命令文件形成的系统在执行某些命令后的情况

Tab. 1　Status of system set up by the file in appendix 1 after executing some commands

作　业	终　端	刚执行过的命令	内 存 量	内存所在组号及起始地址
JOB1	TRM1	SYSTAT	43026 BYTES	AT 0: 53356
JOB2	TRM2	DIR	47026 BYTES	AT 1: 43516
JOB3	TRM3	ABCD	47026 BYTES	AT 2: 43516
JOB4	TRM4	MEMORY	NO MEMORY ALLOCATED	
JOB5	TRM5	SYSTEM	32512 BYTES	AT 4: 100000
SPOOL	PRNTR	LPTSPL	4000 BYTES	AT 0: 43516

址扩展到可共用的内存区，假脱机作业 SP(SPOOL) 就会停止工作．这时，如果有作业申请打印，则不但打不出来，还会引起系统发生死锁 (dead lock) 现象，必须重新启动．

文献[2,3]中介绍了使用 MEMORY 命令的方法，并认为只有 BANK0 内的用户可以使用这命令．以 MEMORY *X* 来改变内存量，试验所编的程序要多大的内存量才能执行，然后再用 MEMORY 0 来恢复原状，的确是一种简便有效的手段．它不像命令 JOBMEM 需要计算和输入八进位地址，麻烦而不直观．因此它的被禁止使用，会使用户有时感到不便．即使这样，也还不能避免用户无意中使用它所带来的问题．

三、减小操作系统的途径

通过适当地编写初始化命令文件和分配内存是有可能把操作系统进一步缩小的．下面仍以附录 1 中的文件为例来说明所用的方法．

比特图是唯一可从操作系统中移到可转换的内存区中的部分[1]．要做到这一点，可先用命令 SYSMEM 在可转换的内存区中定义一个可供操作系统用的内存区，然后在命令 BITMAP 的末端加上符号/*S* 把比特图置入这一区域．在本例中它是被移入 BANK 4 的始端地址 (8 进制) 为 100000—104000 的区域 (参阅附录 2 中的第 18、20 和 21 条命令)．这样做后可使操作系统在固定内存区的部分减少约 1.5K 字节．但这还不够，我们还把软盘的驱动程序 DDA·DVR 从操作系统中除去．因为硬盘的存取速率要比软盘快得多，而且可靠，因此在硬盘和软盘都有的系统中，当然是把硬盘的固定盘作为系统盘．每个用户把自己软盘上要执行的内容先复制到可拆卸的硬盘上，再进行操作或算题，在结束后再把可拆卸硬盘上的结果复制到软盘上，最后从硬盘上洗去这一部分内容．这样，软盘机主要只用作输入输出设备．从操作系统中除去了软盘的驱动程序后，虽然在每次执行有关软盘的操作时，程序 DDA·DVR 会被自动地调入内存区，占据一定的用户内存量．但因为这些操作都是在算题的开始前或结束后，用户内存区是很空的．程序 DDA·DVR 的 848 字节是不会容纳不下的．经实验证明，这样做后，一切工作正常．

采取上述措施后，在 5 个作业，5 个终端的情况下，操作系统所占固定内存区的部分可以缩小到 15838 字节．这样，内存区的转移点就能设置在 16K 处．每个作业(除 JOB5 外)都可得到 48K 字节的内存量．当然内存板的数量要从 3 块增加到 4 块，板上的跳线要改动，附录 1 中第 13 条至 17 条命令也需作相应的修改．

把排队(QUEUE)数和终端的输入字宽，输入、输出缓冲区的大小稍加缩减，有可能容纳下 6 个作业和 6 个终端．但这样做后，系统的性能在某些场合下会下降，例如文件打印排队容易客满；没有足够多的空的排队区来供其它用途[4]；以及不能一次从键盘上输入太长的命令等等．所以要根据使用情况，有限度地减小这些数字．

四、消除作业间相互干扰的措施

为了解决假脱机作业受其它用户区作业干扰的问题，用 JOBMEM 命令把它移至 BANK4 内，紧接在比特图的后面(见图 2 和图 3)．这样做后，打印机的工作是正常了，但各作业还有可能进入可共用的内存区，互相干扰．

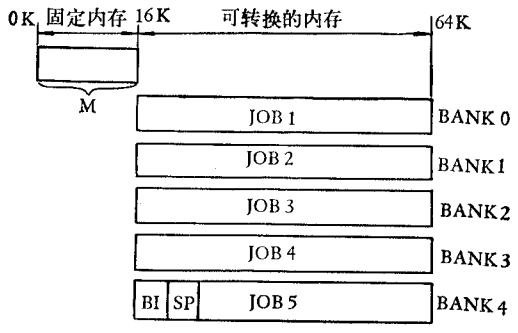

图 2 操作系统正好为16K字节，用户就无法进入固定内存区

Fig. 2 Exact 16K bytes operating system for preventing users from touching the fixed area of memory

仔细地分析一下可发现：作业之间的相互干扰是由于 1 个以上的作业占用了可共用的内存区而引起的．如果能设法使可共用的内存区缩小到零，即操作系统的末端正好达到内存转换点，这个问题即可解决．这可用修改文件中任一终端的输入字宽，输入缓冲区或输出缓冲区的大小或排队 (QUEUE) 所占的内存量来达到．输出缓冲区一个数代表 2 个字节；命令 QUEUE 后面一个数代表 16 个字节．

具体的做法是：先把新的系统初始化命令文件用命令 MONTST 启动，再用命令 SYSTEM 来显示出新操作系统所占固定内存区的大小，再计算出它和固定内存量之差．在上例中，新操作系统的大小为 15838 字节，它与 16384 字节之差为 546 字节．我们把排队作为粗调，输出缓冲区作为细调．所以把附录 1 中的第 21 条命令 QUEUE 15 改为 QUEUE 49；把第 3 条命令最后的数字 100 改为 101．文件中的每一项都是按偶数字节来执行的，如果出现奇数字节也会自动加 1 变为偶数字节．

图 2 是经过这种修改后系统的内存分配情况．操作系统恰好为 16K 字节；BANK0—BANK4 各占 48K 字节．没有可共用的内存区． JOB1—JOB4 也各为 48K 字节．比特图和假脱机作业位于 BANK4 的始端，占 6K 字节，所以 JOB5 占 42K 字节．

上述方法适用于可共用内存区本来就很小的情况，因为这样修改后这一部分内存实际上是没有用了．如果仍然使用 3 块 64K 字节的内存板，BANK0—BANK4 各占 32K 字节，操作系统的大小为 16K 字节左右，内存区的转移点设置在 32K 处，这样可共用的内存区约为 16K 字节．为了使得这一部分的内存量可被某一作业所利用，我们建议采用另一种改进的方法．

在图 2 的情况下，JOB5 的前面由于有假脱机作业 SP 的阻挡，内存地址无法向更低端推进． 因此我们可在 JOB2，JOB3 和 JOB4 的始端相应地设置 3 个作业 BAR1，BAR 2 和 BAR3． 它们不连接任何终端，所占内存量也极小，各为 4 字节．其作用纯粹是为了

阻挡 JOB2，JOB3 和 JOB4 的内存地址向低端扩展而进入可共用的内存区．图 3 示出了用这种方法的内存分配情况．JOB1 是可以利用到可共用内存区的．

这种方法会使操作系统稍稍扩大，因为每增加 1 个新的作业就会多占 292 字节的内存量．但也带来一些其它的好处，即在必要时可从键盘上利用 JOBMEM 命令来扩大 BAR 的内存量，使之真正成为一个可供使用的作业．这样，用户在 1 个终端上可同时利用 1 个以上的作业来解不同的算题或进行不同的操作．

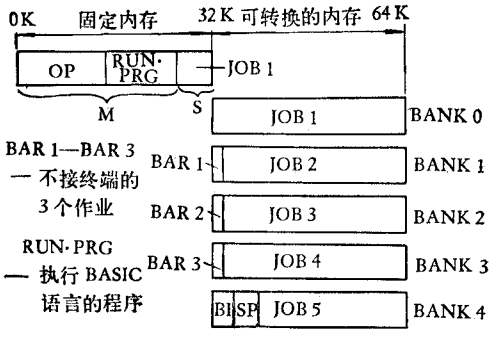

图 3　加入作业 BAR1—BAR3 以阻止作业 JOB2—JOB4 进入可共用的内存区 s

Fig. 3 BAR1—BAR3 added to provent users of JOB2—JOB4 from touching the sharable memory

用命令 SYSMEM 在某组内定义的供操作系统用的内存区是无法阻止用户在使用 MEMORY 0 后内存地址向低端的扩展，所以在 BANK4 内的比特图后面必须跟有假脱机作业或其他作业来与 JOB5 隔离．

附录 2 是形成图 3 所示的内存分配的一个初始化命令文件的实例．与附录 1 内原有的文件相比可以看出在第 1 条命令内增加了 BAR1，BAR2 和 BAR3；第 18 条命令在 BANK 4 内定义出供操作系统用的内存区；第 20 和 21 条命令把硬盘和软盘的比特图放

表 2　附录 2 的系统初始化命令文件在执行后系统的工作状态

Tab. 2　Status of system just set up by the file in appendix 2

```
.SYSTAT
STATUS OF AMOS VERSION 4.4B
JOB1    TRM1    DSKO:1,4          023344 RN    SYSTAT    37062   BYTES   AT 0: 67072
JOB2    TRM2    DSKO:1,4          024010 ^C    SYSTEM    32508   BYTES   AT 1:100004
JOB3    TRM3    DSKO:1,4          024454 ^C    SYSTEM    32508   BYTES   AT 2:100004
JOB4    TRM4    DSKO:1,4          025120 ^C    SYSTEM    32508   BYTES   AT 3:100004
JOB5    TRM5    DSKO:1,4          025564 ^C    SYSTEM    26366   BYTES   AT 4:114002
BAR1            NOT LOGGED IN     026230 ^C                4    BYTES   AT 1:100000
BAR2            NOT LOGGED IN     026674 ^C                4    BYTES   AT 2:100000
BAR3            NOT LOGGED IN     027340 ^C                4    BYTES   AT 3:100000
SPOOL   PRNTR   DSKO:1,2          030004 EW    LPTSPL    4096    BYTES   AT 4:104002
9 JOBS ON SYSTEM
DSKO      7650   BLOCKS FREE      DSK1      9618   BLOCKS FREE
DDAO      NOT MOUNTED             DDA1      NOT MOUNTED
7    DEVICES ON SYSTEM
.SYSTEM
THE FOLLOWING PROGRAMS ARE ALLOCATED IN SYSTEM MEMORY:
        TRM     DVR
        RUN     PRG
TOTAL RESIDENT MONITOR SIZE IS 28218 BYTES
MONITOR VERSION IS 4.4B
```

入这一区内;第 24 条命令把 BASIC 语言用的 RUN·PRG 放入操作系统内;第 29、37 和 45 条命令分别把作业 BAR1、BAR2 和 BAR3 的地址定义在 BANK1、BANK2 和 BANK3 的始端;第 60 条命令把假脱机作业定义在 BANK4 内,原有的命令 MEMORY 4K 被删去.

阿尔法机在用上述系统初始化命令文件启动后,用命令 SYSTAT 和 SYSTEM 在终端荧光屏上观察到的系统工作状态如表 2 所示. 从表 2 可看出内存的分配情况是与图 3 相符合的. 假脱机作业工作于 EW (external waiting) 状态,说明打印机工作正常. 2 个硬盘 DSK0 和 DSK1 以及 2 个软盘 DDA0 和 DDA1 的运行情况也显示出来了. 这说明比特图工作正常. 因为 RUN·PRG 包括在操作系统内,所以操作系统约占 28K 字节,JOB1 大约可有 37K 字节的内存量供使用. 如果不包括这一程序在内,JOB1 的内存量可达 48.5K 字节左右.

可共用的内存区也可为 JOB2—JOB4 中的任一个用户所利用. 例如在 JOB1 的终端上打入命令 MEMORY 0,再在其它任一终端上打入命令 JOBMEM BAR1 0:100000—100002,最后在 JOB2 的终端上打入命令 MEMORY 0,JOB2 就可取得 37K 字节的内存量.

如果有更多的内存板,在图 2 和图 3 所示的系统内还可加添 BANK5. 把比特图和假脱机作业都放在其中. 这样,JOB5 的用户即可拥有和 JOB2—JOB4 用户同样多的内存量.

五、最大可带终端数和作业数

阿尔法机的资料[5]上曾指出这类计算机最多可带 24 个终端,但这只是纯粹从硬件的观点出发. 终端增加时,作业也应相应地增加. 两者都使操作系统扩大. 同时,每个用户所能分享的时间减少,解题或操作所需的时间变长. 这些都是不利的,发展到一定程度,就会变成不能容忍的缺点. 现在仅从内存容量的角度来探讨一下究竟带多少终端和作业是切实可行的.

表 3 是从实验得出的阿尔法机在执行一些主要功能时所需最小的内存量,其中不包括文件或源程序或目标程序占用的内存量. 从表 3 中可以看出当操作系统内包括入程序 RUN·PRG 或 PRUN·PRG 时,执行 BASIC 和 COMPIL 或 PRUN CMPILR 和 PRUN PLINK 所需的内存量都会大大地减少.

假定操作系统的大小超过 32K 字节,则每个组只能有 16K 字节的内存量. 如果操作系统内不包括任何有关的程序,则除了能执行命令 RUN 和 PRUN 外,表 3 内的其它命令都将无法执行. 尤其是 VUE,各种语言的源程序均需用它来进行编辑. 但执行 VUE 需 18000 字节内存量,而用户内存量已小于此数,所以执行 VUE 功能所需的程序 VUE·PRG (14314 字节)必须包括入操作系统内. 这样它最多还只能再放入程序 PRUN·PRG (9228 字节), RUN·PRG (11494 字节)和 LISP·PRG (16710 字节)中的一个. 即所有用户都只能利用 PASCAL, BASIC 和 LISP 程序语言中的一种,而且所编的程序只能是较小的. 所以操作系统大于 32K 字节是不合适的,它将使计算机丧失某些主要的

表 3　操作系统内包含或不包含有关程序时执行命令所需的内存量

Tab. 3　Memory requirement for executing commands in the case of the operating system including Concerned programs or not

命　令	功　能	包含在操作系统内的有关程序	执行命令时所需最小内存量(字节)
VUE	在终端的屏幕上编辑文件	无	18000
RUN	执行 BASIC 语言的目标程序	无	12000
BASIC	进入解释 BASIC 语言	无	25000
		RUN · PRG	12000
COMPIL	把 BASIC 语言的源程序编译为目标程序	无	23000
		RUN · PRG	11000
PRUN	执行 PASCAL 语言的目标程序	无	15000
PRUN CMPILR	把 PASCAL 语言的源程序编译为中间文件	无	23000
		PRUN · PRG	11000
PRUN PLINK	联接不同的中间文件并变换成一个目标程序	无	20000
		PRUN · PRG	11000
LISP	进入 LISP 语言	无	21000
		LSP · PRG	4000

功能.

假定操作系统的大小在 16K—32K 字节之间,则根据增加 1 个终端的同时,还增加 1 个作业和文件打印排队(共 810 字节) 来计算,的确可容纳下 24 个终端和 24 个作业. 但是,为了可使计算机解决较大一点的题目,建议应根据用户的需要把 RUN · PRG 或 PRUN · PRG 包括入操作系统内. 根据 RUN · PRG 的大小来计算,此时计算机可带 11 个终端和 11 个作业.

经常修改系统初始化命令文件是很麻烦的,可预先编制成几种包括不同共用程序的文件,把它们存放在系统盘中,有不同的文件名,然后,根据用户题目的类型和大小,将其中的 1 个,以命令 MONTST 来重新启动计算机,使之在新操作系统的管理下运行.

六、结　论

通过适当地编写系统初始化命令文件,可使阿尔法机的操作系统在不超过 16K 字节的情况下带有 5 个终端和 5 个作业,并有可能扩大到 6 个终端和 6 个作业. 如果愿意付出增添内存板的代价,这种方案将使每一个用户都可取得 48K 字节的内存量.

当阿尔法机经常使用各种程序设计语言时,操作系统不应大于 32K 字节,否则用户将只能使用 BASIC, PASCAL 和 LISP 三种语言中的一种,而且只能解很小的题目.

当操作系统在 16K 字节至 32K 字节之间时,为了能解较大一点的题目,建议终端数

与作业数各不超过11个，使得操作系统有足够的容量来存放一些必要的共用程序．利用预先存放在系统盘内的不同的系统初始化命令文件，可很方便地调换共用程序．

把假脱机作业设置在可共用的内存区内不但减小了用户可取得的最大内存量，而且有被干扰而停止工作的危险．应该把它移到一个可转换的内存组内．

由于一个以上的用户进入可共用的内存区而引起的相互干扰是可以避免的．应用本文中提出的两种改进方法的任一种后，用户可在任何时间随意地使用命令 MEMORY 0，而不会造成计算机的不正常工作状态．

参 考 文 献

[1] *Alpha Microsystems, System Operator's. Information Section of AM-100 Documentation,* May, 1980.

[2] *Alpha Microsystems, AMOS System Commands Reference Manual,* May, 1980.

[3] Leo J. Rotenberg, *Administrator's Guide to the Alpha Microsystem AM-100 and AMOS, Alpha Microsystems, Irvine, California,* 1979.

[4] *Alpha Microsystems, AMOS Monitor Calls,* p. 5—2, 1979.

[5] *Micro Datasystems, AM-100/T*电子计算机系列(样本)，1980.

附录1：原始的系统初始化命令文件(版本4.4b)

Appendix 1: Original system initialization command file (version 4.4b)

编 号	文 件 内 容
1	:T
2	JOBS JOB1,JOB2,JOB3,JOB4,JOB5,SPOOL
3	TRMDEF TRM1,AM300=1:16,ACTIV,100,100,100
4	XY=0
5	TRMDEF TRM2,AM300=3:16,ACTIV,100,100,100
6	TRMDEF TRM3,AM300=4:16,ACTIV,100,100,100
7	TRMDEF TRM4,AM300=5:16,ACTIV,100,100,100
8	TRMDEF TRM5,AM300=6:16,ACTIV,100,100,100
9	TRMDEF TI810,AM300=2:16,TELTYP,2,2,5
10	TRMDEF PRNTR,PSEUDO,NULL,25,25,2
11	DEVTBL DSK1,TRM,RES,MEM
12	DEVTBL DDA0,DDA1
13	MEMDEF 100,0,14
14	MEMDEF 101,3,0
15	MEMDEF 101,14,0
16	MEMDEF 102,3,0
17	MEMDEF 102,14,0
18	MEMERR 250
19	BITMAP DSK,606,0,1
20	BITMAP DDA,154,0,1
21	QUEUE 15

续上表

编　号	文　　件　　内　　容	编　号	文　　件　　内　　容
22	SYSTEM DDA. DVR[1,6]	44	KILL JOB4
23	SYSTEM TRM.DVR[1,6]	45	FORCE JOB4
24	SYSTEM	46	LOG 1,4
25	CLKFRQ 50	47	SYSTAT
26	SET GUARD	48	SYSTEM
27	SET DSKERR		
		49	JOBMEM JOB5 4:100000—177376
28	JOBMEM JOB2 1:100000—177376	50	ATTACH TRM5,JOB5
29	ATTACH TRM2, JOB2	51	KILL JOB5
30	KILL JOB2	52	FORCE JOB5
31	FORCE JOB2	53	LOG 1,4
32	LOG 1,4	54	SYSTAT
33	SYSTAT	55	SYSTEM
34	SYSTEM		
		56	ATTACH PRNTR,SPOOL
35	JOBMEM JOB3 2:100000—177376	57	KILL SPOOL
36	ATTACH TRM3,JOB3	58	FORCE SPOOL
37	KILL JOB3	59	MEMORY 4K
38	FORCE JOB3	60	LOG 1,4
39	LOG 1,4	61	LPTINI PRNTR. INI
40	SYSTAT		
41	SYSTEM	62	WAIT SPOOL
		63	MOUNT DSK1:
42	JOBMEM JOB4 3:100000—177376	64	MEMORY 0
43	ATTACH TRM4,JOB4		

附录 2：改进后的系统初始化命令文件

Appendix 2: System initialization command file improved

编　号	文　　件　　内　　容
1	:T
2	JOBS JOB1,JOB2,JOB3,JOB4,JOB5,BAR1,BAR2,BAR3,SPOOL
3	TRMDEF TRM1,AM300=1:16,ACTIV,100,100,100
4	XY=0
5	TRMDEF TRM2,AM300=3:16,ACTIV,100,100,100
6	TRMDEF TRM3,AM300=4:16,ACTIV,100,100,100
7	TRMDEF TRM4,AM300=5:16,ACTIV,100,100,100
8	TRMDEF TRM5,AM300=6:16,ACTIV,100,100,100
9	TRMDEF TI810,AM300=2:16,TELTYP,50,50,25
10	TRMDEF PRNTR, PSEUDO,NULL,25,25,2
11	DEVTBL DSK1,TRM,RES,MEM
12	DEVTBL DDAO,DDA1
13	MEMDEF 100,0,14

续上表

编 号	文 件 内 容
14	MEMDEF 101,3,0
15	MEMDEF 101,14,0
16	MEMDEF 102,3,0
17	MEMDEF 102,14,0
18	SYSMEM 4:100000—104000
19	MEMERR 250
20	BITMAP DSK,606,0,1/S
21	BITMAP DDA,154,0,1/S
22	QUEUE 15
23	SYSTEM TRM.DVR[1,6]
24	SYSTEM RUN.PRG[1,4]
25	SYSTEM
26	CLKFRQ 50
27	SET GUARD
28	SET DSKERR
29	JOBMEM BAR1 1:100000—100002
30	JOBMEM JOB2 1:100004—177376
31	ATTACH TRM2,JOB2
32	KILL JOB2
33	FORCE JOB2
34	LOG 1,4
35	SYSTAT
36	SYSTEM
37	JOBMEM BAR2 2:100000—100002
38	JOBMEM JOB3 2:100004—177376
39	ATTACH TRM3,JOB3
40	KILL JOB3
41	FORCE JOB3
42	LOG 1,4
43	SYSTAT
44	SYSTEM
45	JOBMEM BAR3 3:100000—100002
46	JOBMEM JOB4 3:100004—177376
47	ATTACH TRM4,JOB4
48	KILL JOB4
49	FORCE JOB4
50	LOG 1,4
51	SYSTAT
52	SYSTEM
53	JOBMEM JOB5 4:114002—177376
54	ATTACH TRM5,JOB5
55	KILL JOB5

续上表

编　号	文　　件　　内　　容
56	FORCE JOB5
57	LOG 1,4
58	SYSTAT
59	SYSTEM
60	JOBMEM SPOOL 4:104002—114000
61	ATTACH PRNTR,SPOOL
62	KILL SPOOL
63	FORCE SPOOL
64	LOG 1,4
65	LPTINI PRINTR.INI
66	WAIT SPOOL
67	MOUNT DSK1:
68	MEMORY 0

DISCUSSION AND IMPROVEMENTS ABOUT SYSTEM INITIALIZATION COMMAND FILE OF ALPHA MICRO COMPUTER

Jiang Jun-ji

(Institute of Electronics, Academia Sinica)

An investigation for the system initialization command file of ALPHA MICRO computer is described. Methods for reducing the size of the operating system have been studied. Some problems of the original file are pointed out and improvements have been made. The file improved contains five terminals and five jobs while the size of the operating system is less than 16 K bytes. Two methods are presented to avoid the abnormal running of the computer caused by users using the MEMORY 0 command simultaneously. A sample of the new file and its experimental result are given. The maximum number of terminals and jobs is also evaluated in view of actual effect.