

操作系统原理

DJS-200 系列机联合设计组

引言

早期的计算机运算速度低，外围设备少，软设备简单，计算机的用户可从对计算机有较详细的了解，它可直接地利用计算机的硬，软设备来编制它的程序，并且自己上机解题。解题的过程基本上是“手工操作”，尽管这样做浪费很多机时时间，但是，在当时的条件下，这并没有多大问题。

随着计算机的发展，运算速度提高，外围设备增加，软设备丰富，用户很难直接利用计算机的硬，软设备来编制他的程序了，同时解题的“手工操作”大大地降低了计算机的效率，此外，这样一个复杂的系统要直接由人工管理也是很难设想的了，因此，随着计算机的发展，这种用于系统进行控制和管理的程序系统也在不断发展，早期的计算机，有的就配制了控制输入/出的子程序以方便使用。当计算机具有分时操作和程序中断的功能后，就有了管理程序。它可以说是初级的“操作系统”，随着计算机的进一步发展，这种用于系统进行控制和管理的子程序不断丰富而形成一个程序系统，操作系统。

目前，操作系统主要从下一些功能：

1. 控制硬设备操作，主要是控制输入/出设备，处理机时故障，以及计时等。

2. 管理数据资料，主要是对各种软设备（如编译程序，汇编程序，编辑程序，程序库等）和用户的专用程序，数据，以及其它各种信息进行编目，存档，在需要时可取出。

3. 监督程序运行。主要是控制程序输入和结果输出，安排程序和时间以及按照用户写的“工作说明书”对程序的运行进行控制和监督。

4. 分时系统的内部控制，所谓分时系统是指这样的计算机系统：它有若干供用户解题使用的“终端设备”（例如，现有的程序控制打字机，就可以作为一种终端设备），用户可通过各自的终端设备同时使用计算机来解决它的问题。实现这样的系统，操作系统是不可缺少的。它通过用户的终端

设备和用户“对话”。一方面，按照用户要求控制计算机系统工作，另一方面将运行情况或出现事故通知用户。

操作系统还有一些功能，例如在计算机网络方面的内容并未列入，随着计算机的发展操作系统的功能将不断扩大，内容将不断丰富。

操作系统的主要作用是：提高效率，方便使用，减少出错，操作系统对整个计算机系统进行全面控制，因此大大减少了人工干预，从而提高了效率。操作系统为用户提供了利用计算机系统的各种手段，因此，用户无需对计算机系统有详细的了解，就可以利用系统的各种设备来解决各种问题。从用户角度说，计算机系统更复杂了，功能更强了，然而他必须了解的却减少了，操作系统还提供了各种保护和检查手段，从而可以避免许多可能发生的错误，同时一旦发生错误也能较及时发现并报告。

实现一个操作系统，对硬件设备的主要要求是：稳定程度高，以及有一定容量的内存和外存贮，和具有程序中断的能力，操作系统的功能大小将根据计算机的规模 and 具体要求而定。

目 录

引言	
第一章 多道程序	1
第一节 分时操作	1
第二节 中断装置	1
第三节 多道程序	6
第二章 输入/出设备管理	11
第一节 分配	19
第二节 启动	19
第三节 中断处理	24
第三章 文件系统	27
第一节 概念	31
第二节 种类和结构	31
第三节 管理	32
第四节 例子	35
第四章 操作者和机器的联系	41
第一节 操作命令	46
第二节 报告设备	47
第五章 非会话型任务的管理	50
第一节 成批处理	53
第二节 输入/出设备的使用	55
第三节 工作控制语言	59
第六章 会话型任务的管理	68
第七章 调动	74

第一章 多道程序

第一节 分时操作

早期的计算机没有分时的操作功能，输入/出控制过程大致是这样的：一条输入（或输出）指令，它指出要启动的外围设备从哪个内存区。当执行输入（或输出）指令时，首先启动指令所指定的外围设备，然后逐个进行输入至指定的内存区（或从指定的内存区输出）。直到全部输入（或输出）完毕时，这条输入（或输出）指令才认为是做完了。

由于外围设备的速度比中央处理机的速度低得多。因此，用上述方式进行输入（或输出）时，中央处理机经常处于等待状态，因而效率不高。例如，从纸带输入机上输入一个字符要花 $1\text{m}\cdot\text{S}$ 的时间。而中央处理机将读出的一个字符存放好可能只要 $10\mu\cdot\text{S}$ 或更少的时间就够了。因此，在一个字符读出后至下一个字符读出前这 $1\text{m}\cdot\text{S}$ 的时间里，中央处理机只工作了 $10\mu\cdot\text{S}$ 的时间，占整个时间的 1% 。也就是说，在执行从纸带输入机输入的指令时，中央处理机 99% 的时间都浪费了。

随着计算机的发展，中央处理机和外围设备的速度差距愈来愈大，外围设备的数量，种类也不断增加，这种原始的控制输入（或输出）方法就很不适应了。怎样解决这个问题呢？分时操作就是一个方法。

所谓分时操作就是在上述中央处理机等待的时间里，不是空等，而是“抓紧时间”做别的工作。在上述的例子中，当输入机输入一个字符后，中央处理机花了 $10\mu\cdot\text{S}$ 将它存放好。余下的 $990\mu\cdot\text{S}$ 时间，中央处理机可以进行别的工作。直到下一个字符输入后，中央处理机才“回过头来”处理一下，也就是说：中央处理机和外围设备“分享”了那剩余的 $990\mu\cdot\text{S}$ 时间，或者说，实现了中央处理和外围设备的同时工作。

具体实现分时操作的方法是多种的，但基本原理是一样

的。现将实现的基本原理叙述如下：

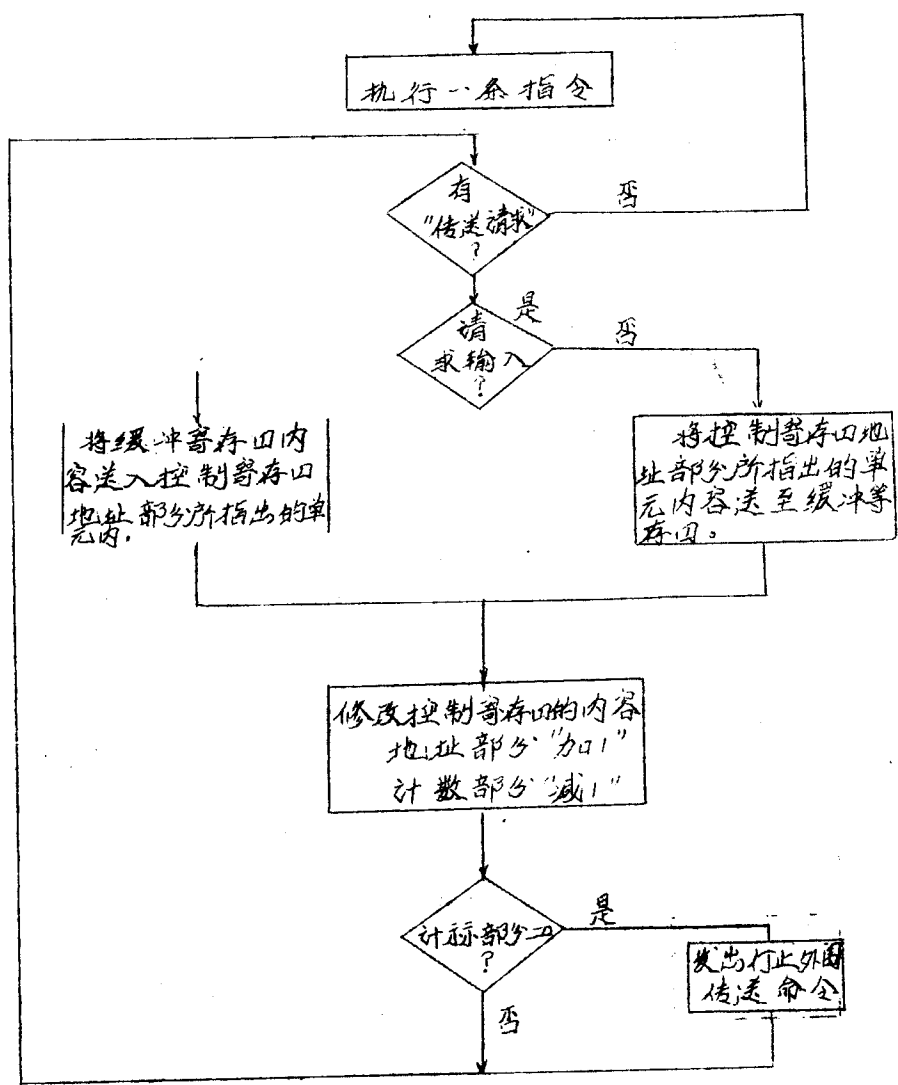
假定每台外围设备都有一个缓冲寄存器和一个控制寄存器。在启动外围设备进行输入（或输出）之前，先将要求传送的IP数和内存的起始地址置于控制寄存器内，然后启动外围设备输入（或输出）。当一个字符输入至缓冲寄存器（或一个字符从缓冲寄存器中输出）后，外围设备向中央处理机发出传送请求。中央处理机在每执行完一条指令后，都询问一下是否有传送请求，如果有，则从请求传送的外围设备的缓冲寄存器中传送一字符至该设备的控制寄存器指定的内存单元（或从指定的内存单元取出一字符送入缓冲寄存器）。完成传送后，中央处理机修改控制寄存器的内容（地址增加1，计数减少1）以备下次传送控制用。当控制寄存器的计数部分减为0时，传送就此结束。

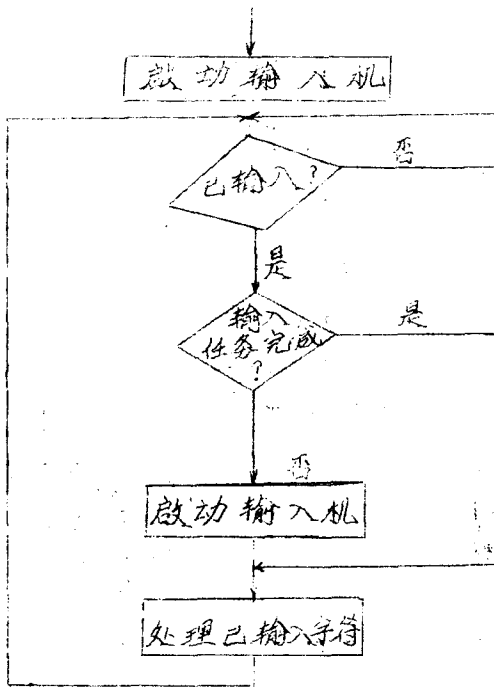
上述过程可以用以下框图来描述（图见下页）

不同类型的计算机、对于不同的外围设备，实现分时操作的方法各不相同，归结起来，大致有以下几种方法：

1. 程序实现、即上述处理内存和外围设备之间的传送是通过若干条指令的专用程序来实现的。

例如，计算机有：“启动外围设备”指令，“测试外围设备”指令（它用以判定是否已输入或输出一个字符）和缓冲寄存器与累加器之间的传送指令。那么启动外围设备（例如输入机）进行输入的过程，可以用下述程序来实现。





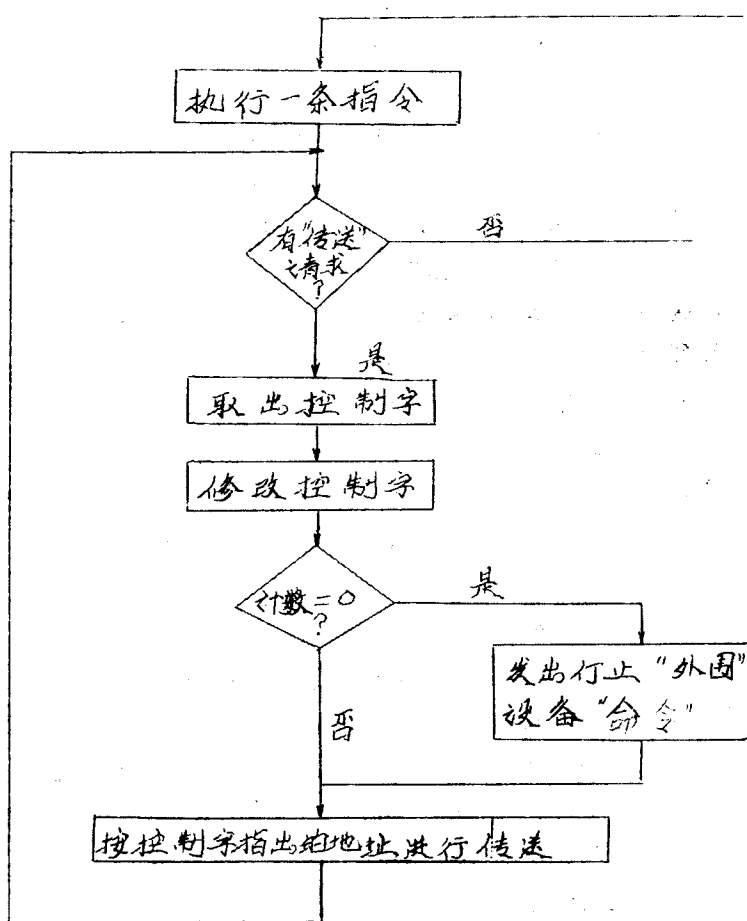
显然，处理已输入的一个字符（包括传送和其它处理）时，输入机就在输入下一个字符，因而实现了分时操作。

用这种方法实现分时操作是很简单的，但效率是不高的，一般小型计算机，对慢速外围设备，可采用这种方法。（可以参阅NOVA计算机的有关资料）。

2. 简单硬设备实现。也就是叙述原理时所介绍的方法。不过，一般“控制寄存器”不是真正的寄存器，而是利用某个特定的内存单元。对于每个外围设备，都是一个（或两个）特定的内存单元，用以存放控制输入（或输出）的“计数和地址”（简称输入/出控制字）。

这种实现方法，可以用下图来描述：

（图见下页）



用这种方法，除了增加一些控制线路外，几乎不增加什么设备，只是访问内存较多一些，一般中小型机对于慢速外围设备常用这种方法。

3. 专门硬件设备实现（或称交换方式）即计算机有一个专门部分（称为交换器）它用来控制外围设备输入或输出。而当输入（或输出）了若干个字符（例如一个字或两个字）以后，它才向存储器取出（或写入）若干个字符。而外围设备和交换器之间的传递方式和2中叙述相同。

这种方法：设备化费多一些，但内存访问次数大大减少了。对于大型计算机常采用这种方法。

第二节 中断装置

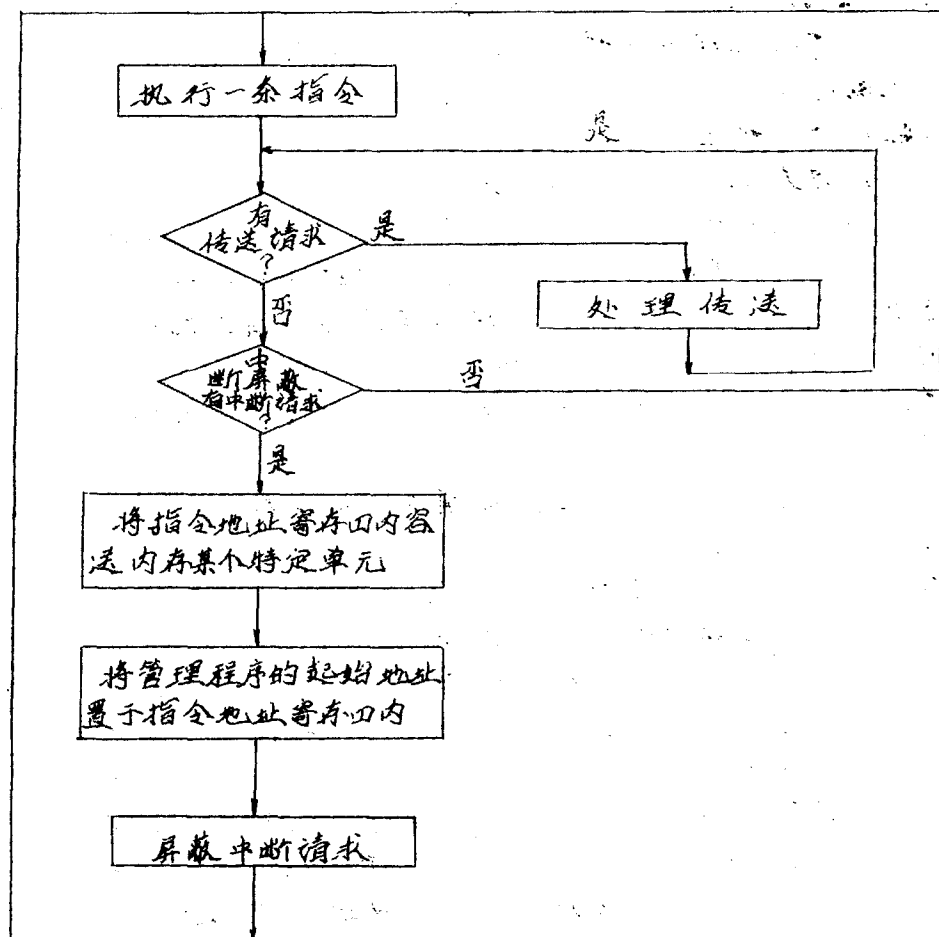
早期的计算机毫无处理事故的能力，当事故发生时，只好停机等待人的干预，这就大大地降低了计算机的效率。例如，从磁鼓中读入一组信息。如果传送中发生偶错。这完全没有必要停机。较好的办法是启动它重读。显然，后一方法就比前一方法效率高得多。又如程序执行过程中发生溢出，一种方法是立即停机等待人的干预，另一种方法是将发表溢出的指令及其地址以及有关的一些信息，通过打字机通知操作员，在操作员干预之前，发生溢出的这个程序停止运行，但机口并不停机还可以进行别的工作，显然第一种方法是不方便的，效率很低的。从这两个简单的例子，我们不难想象处理事故更有效的方法。

如果事先编好处理各种事故的程序并将它存放好，当事故发生时，计算机不但不将处理事故的程序引出来进行处理，这样做，显然比简单地用停机来对付一切事故要好得多。我们将“停止运动程序而引出处理事故的程序”称为“程序中断”产生“程序中断”的事故，称为中断源，处理中断的程序称为管理程序。发现中断源而产生程序中断的硬设备装置就是“中断装置”。

中断装置，一般说，是很简单的：它主要有以下功能。

1. 发现中断源，提出中断请求。
2. 保护现场（指存放指令地址寄存器内容等）以便处理中断结束后，继续执行被中断的程序。
3. 引出管理程序（即将处理被中断的子程序的起始地址置于指令地址寄存器内）。

具有中断装置的中央处理机，响应中断的过程可用下图表示：



图中“屏蔽中断请求”的作用是：在管理程序执行过程中不响应中断请求。有时候并不加此限制。例如进行实时控制时，往往允许在管理程序执行时，接受实时中断请求；又如管理程序执行时发生“中央处理机出错中断”，往往也是响应的，为了做到这点，中断装置和管理程序的编制往往要复杂一些。即将中断流分成若干级。当管理程序在处理中断时，较低级的中断请求不响应，而较高级的中断请求是可以

响应的。

现代计算机的中断源种类较多。大大超出早期计算机引起行机的原因。一般说来中断源大致有以下四类：

1. 中央处理机故障性中断。例如存储的奇偶错，电流变化过限等。对于这类中断的一般处理办法是：保护现场，通知操作员，机内暂停工作。

2. 程序性中断，又可分为两大类：

(1) 出错性的，例如运行程序出现了未有定义的操作码的指令，地址越界，溢出等，这些错误，一般说，管理程序是无法纠正的。因此，处理办法是给出有关的信息（例如发生未定义操作码的指令地址，产生越界的指令地址，发生溢出的指令地址和指令本身等），然后，产生错误的程序被行止运行。

(2) 需求性的。例如广义指令，即该指令对硬设备没有定义，但对软设备有定义，所引起的中断，处理这种中断的办法是转致相应的子程序。

3. 外围设备中断，又可分为两种：

(1)、出错性的，例如传输奇偶错，输入（有出）没有纸带或纸带断了等。又如纸带即将用完，纸带即将用完等警告性的。对于这类中断的处理办法，一般是行止传输立即向操作员报告于磁带，磁鼓等读（写）奇偶错则采取重读（写）的办法，直连续重读（写）若干次仍错才行止读（写）进行报告。

(2)、报告性的，例如外围设备传输任务完成，人工置联机脱机，或是外围完成了某个命令（如磁带反绕到头）等。

4. 外界中断，例如时钟中断，控制请求线中断等。

下面以 IBM 360 为例，来对中断装置作一些叙述。

一、中断源：

1. 机内检查，当硬设备出错时产生中断请求。
2. 外界中断、时钟或控制请求线的中断请求。
3. 转管理中断，执行指令：“转管理”产生的中断。
4. 程序性中断，产生的原因有以下几种：

(1)、未定义操作码

(2)、机因在问题程序状态而出现不允许的指令。(有些指令,如启动输入/出,只有被管理程序使用才是“合法”)

(3)、执行指令错误(例如执行指令所指定的指令(是执行指令或执行指令指定的地址非偶数)。

(4)、保护链不符

(5)、地址超越

(6)、说明不当

(7)、数据形式不合

(8)、定长溢出

(9)、定长除法出错(包括除数=0)

(A)、十进运算溢出

(B)、十进除法商数超过指定的数据区。

(C)、阶码溢出

(D)、阶码消失

(E)、浮点结果全“0”

(F)、浮点除数为“0”

5、输入/出中断,由输入/出结束,或人工干预引起。

各类中断优先级次序是。机因检查,输入/出、外部,程序中断或转管理。

当中断请求被响应后,中央处理机将程序状态字(PSW)(它包括指令地址和对中断的屏蔽要求等),存放在某个特定单元,然后根据中断的种类:从不同的特定单元取出新的程序状态字,从而就转入处理中断的程序。下表列出了存放PSW和取出新的PSW的各特定单元。

中断种类	当前PSW存放单元	新PSW取出单元
外界	24	88
转管理	32	96
程序性	40	104
机因检查	48	112
输入/出	56	120

程序状态字(PSW)的第0~7位36~39位给出对中断的屏蔽,它的具体定义如下

位	意义
0	屏蔽复合通道中断 (即此位为0时, 对来自复合通道的输入/出中断请求无响应)
1	屏蔽选择通道1的中断,
2	" " " " " " " " 2 " " " ,
3	" " " " " " " " 3 " " " ,
4	" " " " " " " " 4 " " " ,
5	" " " " " " " " 5 " " " ,
6	" " " " " " " " 6 " " " ,
7	屏蔽外界中断
13	" " 机内检查中断
36	" " 定长溢出 " "
37	" " 十进 " " " "
38	" " 阶码 " " " "
39	" " 意义丧失 " "

程序状态字的第16 ~ 31位用以存放“中断代码”指出产生中断请求的原因, 以供管理程序处理中断时用。具体意义如下:

中断种类	PSW位	意义
输入/出	21—23	提出中断请求的通道地址
	24 ~ 31	提出中断请求的设备号
转管理	16 ~ 23	置0
	24 ~ 31	记录该指令的8 ~ 15位
外界	24	置1若为时钟中断
	25	置1若为控制键中断
	26 ~ 31	其它设备中断
程序中	28 ~ 31	给出一个16位数, 1—9, A—F 分别对应16种程序性中断。

当引出处理该类中断的子程序后, 它从保护存放的PSW字的21 ~ 31位中, 很快就知道产生中断的具体原因, 从而进行处理。

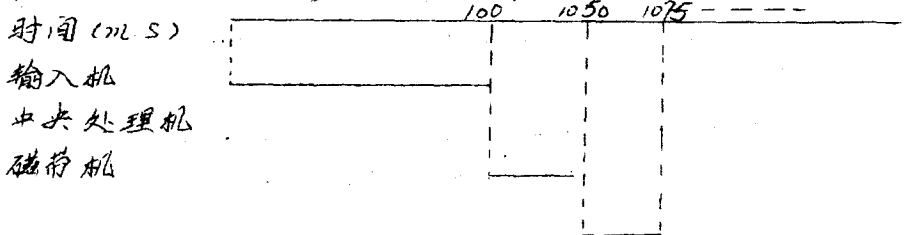
第三节 多道程序

前边已经叙述了，采用分时操作可以提高计算机的效率。但是，对于中型，大型计算机，外围设备较多，如果采用分时操作，效率并不是很高的。为了阐明这一点，我们分析以下几个例子。

例1. 某个数据处理问题的计算过程如下：

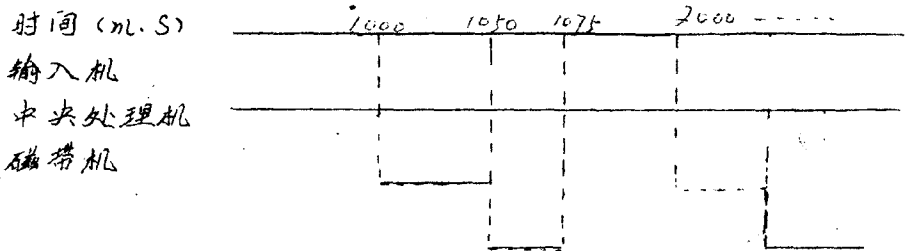
从输入机（速度为1000字符/秒）输入1000个字符经处理（耗时50m.S）后，将处理过的1000个字符存入磁带上（磁带机速度为40KC/S）。

对于没有分时操作功能的计算机，整个计算过程如下：



中央处理机的效率为 $50 / (1000 + 50 + 25) \approx 4.6\%$

对于有分时操作功能的计算机，整个计算过程如下：



中央处理机的效率为 $50 / 1000 = 5\%$

例2. 某个数据处理问题计算过程如下：

从磁带上（速度为40KC/S）输入1000个字符经50m.S的处理后，从打印机上打出（假定一次处理需时1秒打印）。

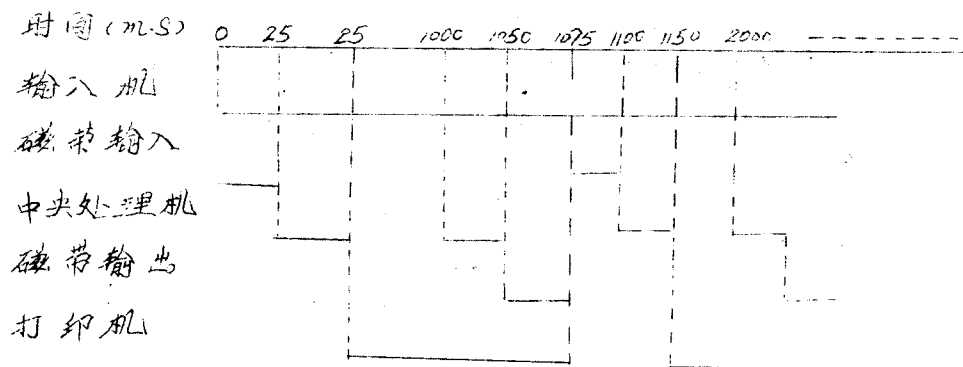
这个问题的整个过程和例1相仿。对有分时操作的计算机中央处理机的效率仍为5%。

从上述两个简单的例子中可以看到，仅有分时操作功能的计算机效率并不高。因为分时操作提供了中央处理机和外

图设备同时工作的能力和各外围设备同时工作的能力，但是程序很难想象编得如此巧妙，以致使中央处理机和各外围设备都能同时工作起来，从例1看，在时间1050以后，中央处理机有能力去做别的工作，但是由于下一字符组未全部输入，因此没有工作可做。另一方面，在例1的问题进行过程中，打印机是处于空闲状态，从第50时，操作允许打印机，输入机等外围设备同时工作。

有了分时操作功能以后，怎样进一步提高计算机的效率呢？多道程序就是解决的办法。所谓多道程序就是在内存中存放两道或两道以上的程序，当一道程序由于某种原因（例如等待输入或出现程序性错误等待人工干预，而不能继续运行时，另一道程序就被启动。显然，这样一来计算机的效率可以提高。

如果计算机不仅有分时操作功能而且有多道程序功能。那么对上述例1、例2，两个计算机是可以同时进行的。其工作过程如下：



中央处理机效率为 $(50 + 50) / 1000 = 10\%$

如果还有一道程序同时进行，那么效率又可增加若干，特别如果搭配得好，例如上例中再增加一道计算机较多而访问外围设备较少的程序，那么效率将有许多提高。

从上述简单例子可以看到，除了技术上提供稳定可靠的硬设备外，分时操作和中断装置是实现多道程序的必要条件。此外，要实现多道程序还必须妥善地解决从以下几个问题：

1. 地址保护。由于几道程序共存于一个存储器内，就必须采取措施防止程序间的相互干扰。否则一道程序的错误会扰乱其它的程序以致不可收拾。所谓地址保护是指这样一些硬设备：它只允许每一道程序的任一条指令访问该道程序所占据的区域或一些可以允许访问的公共区域。

实现地址保护的方法有若干种。最简单的方法是：“界地址”的办法。

假定程序都存放于内存中单元 Q 至单元 $Q + n - 1$ 的几个连续区域内。在执行程序 P 时，每一条指令访问内存之前，都将访问内存的真地址 D 和存放区域的下界“ Q ”和“上界 $Q + n$ ”相比较，若满足

$$Q \leq D < Q + n$$

那么内存可以访问，否则不允许访问而产生“越界中断”。

因为编制程序时，并不知道它执行时是存放于内存何处。因此它总是考虑从单元 0 开始的。如果它执行时是存放于单元 Q 至 $Q + n$ 内，那么只要在访问内存时，地址加上“下界”就得到访问内存的真地址了。例如程序中的某条指令访问内存 x ，那么若它存放于 Q 至 $Q + n$ 区域时，真正访问的内存单元是 $Q + x$ 。

由此可见，用界地址办法实现地址保护，只要在中央处理机中，增加两个寄存器，下界地址的寄存器。当执行某个程序之前，先将该程序的上、下界地址存放好。在执行中，每一条访问内存的指令，其地址是由指令所指出的地址和下界相加而得。在访问内存之前，必须将其地址和上下界比较看是否发生“越界”。为了节省元件，常常可以让界地址的右若干位总是为 0 ，即每道程序总是存放在 $Q \cdot 2^k$ 至 $(Q + n) \cdot 2^k$ 这 $n \cdot 2^k$ 个连续区域内。这样每个界地址寄存器的长度不是计算机的内存地址长度 L ，而是 $L - k$ 。例如，某计算机的最大内存容量为 2^{10} ，假定界地址的右 10 位总是为 0 ，即每道程序总是占有整数个 $1024 (=2^{10})$ 单元，那么每个界地址寄存器只要 10 位就够了。

2. 外围分配。如果计算机内只有一道程序，那么计算机