MS—DOS 4.0

磁盘操作系统

郁少文　张津申　赵　煌　赵建英　编著

MS—DOS4.0

磁盘操作系统

郁少文　张津申
赵　煌　赵建英　编译

陕西电子编辑部

编 译 者 序

为了满足当前计算机领域内日益增长的需要，我们主要根据美国MICROSOFT公司的MS—DOS V4·0(1986年12月出版)编译出本手册。第一章到第五章由郁少文编译，张建英校阅，第六章到第十章由张津申编译，赵煌校阅，第十一章到第十六章由赵煌编译，张津申校阅，第十七章和附录由张建英编译，郁少文校阅。全书由西安冶金建筑学院魏文郁付教授总校阅。在此手册的编译过程中，得到陕西电子编辑部主任张忠智高级工程师和副主任孙彩贤工程师的大力帮助，在此深表谢意。

由于我们的水平有限，时间短促，工作中定会有缺点，错误和不当之处，欢迎读者给予批评和指正。

编 译 组

1988年7月

目 录

第一章　编程环境

1.1　前言

MS—DOS4.0是计算机8086系列的多任务磁盘操作系统。它能运行一个前台应用程序和多个后台应用程序。

随着个人计算机和办公室自动化软件的日趋完善，多任务处理功能已成为一种需求。多任务功能改进了系统的性能，因为它允许建立像假脱机打印这样的后台任务。而且，可以后台任务运行网络文件盘及邮件系统进程，使计算机进入网络环境。

1.2　必要的条件

在MS—DOS3.2上设计的程序在MS—DOS4.0上也能运行。此外，MS—DOS4.0的较强的应用程序接口（API）功能使新的用户能够享用它的多任务处理特性，从而为用户提供了新的有效的功能。

在MS—DOS4.0的支持下，应用程序即可在前台区执行，也可在后台区执行。在前台区运行的应用程序，像MS—DOS3.2的程序一样，可以直接同计算机及用户相互作用。但后台区的使用只限于那些专为MS—DOS4.0后台环境而开发的应用程序。

MS—DOS4.0运行的机型完全相同于MS—DOS3.2。不过为了最大限度地发挥MS—DOS4.0的性能，这些计算机应当提供中断——驱动I／O方式。

1.3　8086结构上的局限

MS—DOS4.0力图在8086系列微机上运行MS—DOS3.2的各种应用程序。8086CPU和286实地址方式都没有保护和硬件再定位功能，这两种功能在多任务环境中是十分有用的。8086系列微机没有对处理机芯片增加外部硬件存储保护，所以它们不能防备故障，也不能防止带故障程序对其它程序及DOS本身的破坏。

286具有保护及硬件再定位功能，而且是以与8086二进制不兼容的方式提供的。因为不可能将286这些新的特性传送到老的应用程序，所以，在实地址方式下运行程序时，MS—DOS4.0在8086及286微机上均没有保护措施。也不能防止或恢复被实地址方式运行的程序所破坏的任务。

1.4　定义

本书将用到下列定义：

任务（Task）——一个独立程序的执行。多个任务可以同时存在且通过任务数据结构

有各自的DOS考贝。

并发性（Concurrency）——多个应用程序同时运行的能力。

设备驱动程序（Device Driver）——专为硬件编制的例行程序。通过它将DOS和被DOS系统调用的硬件相连。每个设备驱动程序都有一个开始操作关键入口点，同时还有一个中断入口点，在操作结束时被调用。

中断（Interrupt）——需要立即服务的外部事件，使8086暂时从它的正常执行中中断。在某些情况下，中断服务程序将不受时间片的约束，但典型的中断服务程序都是被优化为能使8086尽可能快地回到正常执行中去。

被分离的任务（Detached Task）——即是那些在后台环境运行的任务。

被锁定的任务（Blocked Task）——只有当DOS请求或某一设备条件满足后才能继续执行的任务。

可运行任务（Runnable Task）——与被锁定任务相反，不需经DOS请求或某一设备满足，调度程序即可选择“可运行任务”进行运行。

任务转换（Task Switch）——DOS系统挂起某一任务而恢复另一个任务的执行。为了任务转换，DOS系统必须转换内部数据结构，每秒钟内这种转换可进行多次。

执行（Exec）——一个任务(母任务)调用另一个任务(子任务)的能力。所有由命令解释程序所调用的应用程序都是它的子程序。母任务即可被中止执行，直到子任务执行完为止，也可以不受子任务的影响继续执行（至少在初始化时是如此）。

外壳程序（Shell）——控制一系列程序运行的一个命令解释程序。

前台区（Foreground Partition）——前台区运行那些直接与计算机和用户相互作用的应用程序。任何时候前台都只能有一个程序运行。

后台区（Background Partition）——后台区运行那些专在MS—DOS4.0上开发的程序。这些程序利用MS—DOS所提供的服务功能与计算机用户相互作用。在后台内几个不同程序可并发运行。

1.5 DOS接口

MS—DOS功能是通过INT 21H进入的。DOS调用控制着计算机的每一部分。一个程序唯有通过DOS调用才能和用户、其它程序或设备打交道。

对程序设计者来说，DOS是可重新进入的。因为设计后台任务，设计者可以不考虑其它任务所带来的限制。然而，从一个给定的任务看，DOS系统一直是与之同步的，如果一个任务还在DOS系统中，而且受到DOS的控制（如直接受一次硬件中断）时，则此任务便不能再呼叫DOS。同样，这种限制使设备驱动程序仅在初始化时或借助辅助功能时才能呼叫DOS。

1.6 设备驱动程序

1.6.1 多任务设备驱动程序

虽然大多数现有的MS—DOS3.2设备驱动程序均可在MS—DOS4.0上运行，但这些驱

动程序降低了系统的性能。因此，所有的设备驱动程序最好都按MS--DOS4.0所提供的特性来重写，这些性能包括：

·锁定任务：当等待一个I／O操作完成时，MS—DOS4.0可从运行队列中移去一个任务。

·重叠I／O：当这种磁盘操作发生时，其它可运行的任务也被启动以使CPU处于忙状态。由于多个任务在文件系统中能够同时激活，所以这些任务能够产生磁盘请求。

·排队请求：由于在文件系统中多个任务可以被激活，设备驱动程序可将多个请求进行排队。设备驱动程序能随意地按最有效的方法对请求进行重新排列。

1.6.2 控制台驱动程序

控制台设备驱动程序分为两个部分：屏幕显示和键盘。由于这些驱动程序日益复杂，较为理想的方法是分别替换它们。

1.7 键盘中止

键盘中止通过键盘操作（如CTRL—C）来实现。在通常的操作中，当前命令解释程序（称为外壳“Shell”，一般为Command.com程序）启动SIGINTR信号。如果用户键入CTRL—C，外壳程序便接收SIGINTR信号并通过系统调用发出一个SIGTERM信号给当前运行的程序。在该进程已产生多个子进程的情况下，它便将SIGTERM信号发给当前运行进程的命令子树，直到所有的子进程结束后，外壳程序才会提示用户。

过程能从以下三种方式来阻止键盘中止的实现：

(1) 进程可能不希望随意被删除，也许因为它们需要清除某些锁定的文件或者需要恢复某个数据库。在这种情况下，当信号被接收后，程序中止SIGTERM信号，服务码被清除然后退出进程。

(2) 程序可通过将键盘设备转入原始方式来影响SIGINTR（即CTRL—C）的处理。这是由屏幕编辑等程序完成的，它们将CTRL—C作为一个数据字符来看待，在退出之前，程序将键盘恢复到设定状态。

(3) 运行程序本身也可以是一个外壳程序。例如，编辑程序、多设定（Multiplan）程序等等。这类程序确实可用CTRL—C报警，但却不会因此被删除，它们会发出Set—Sigial—Handler系统调用来中止SIGINTR信号。这就自动地挂起了母外壳程序对SIGINTR信号的响应。当新的外壳程序退出母外壳程序时，母外壳程序又能自动地再响应这些信号。

1.8 内部结构

每一个任务都包括了一个Per Task Data Area(PTDA)和一个程序段，这种设计结构被称为多逻辑内核触发。这种技术(也用于XENIX)大大地简化了内核结构，且有助于与以前的版本保持兼容。为了保证其简单、完整，整个MS—DOS的内核都被认为是不能中断的临界代码区。如果每次只可能有一个任务在内核中活化，这本身就造成了一个非优化的环

境。将临界区退入到设备驱动程序中（这就要求在调用设备驱动程序之前将所有DOS数据结构适当的修改），如果一个任务留在设备驱动程序外面，那么多个任务即可同时在内核中活化。由于从进入内核到检索设备驱动程序的时间很短，所以为实现程序设计简单明了，这点代价还是值得的。

如上所述，任务程序段与其它MS—DOS3.2下的任何程序具有同样的格式（256个节字标题，加上实际程序），而PTDA是为DOS而设的数区，据它包含所有任务专用信息。该区域只受DOS控制，任务本身却无法使用它，每一个任务都有一个单独的PTDA。

唯有前台应用程序才能使用8087协处理器。

1.9 硬件条件

能支持MS—DOS的最小系统是带有定时器的8086／8088计算机（即处理器、存储器、视频I／O等），事实上，这个最小系统中还包括中断—驱动式I／O和对磁盘的DMA（即直接存储器存取）功能。

MS—DOS4.0的设计是为了充分支持并行I／O。并行I／O就是说系统能同时响应 n 个不同的I／O请求。虽然每次只能处理一个主要的I／O请求，但每个任务都有其各自的待处理请求。并行I／O对磁盘设备来说是十分有利的。磁盘设备的待处理请求将按其在磁盘上的位置进行排序，这样就减少了查找的时间，同时增加了总的I／O处理能力。为了最大限度地利用这一功能，硬件系统应保证每台设备都能使用并行I／O。这包括给每台设备配备合理的中断线路，对每个设备单独的DMA通道和并行磁盘查找。所有的硬件中断都是由相应的设备驱动程序来处理的，当I／O操作完成后，也是由这个设备驱动程序来通知DOS系统。

第二章　MS—DOS4.0新的系统调用

2.1　进程控制

新的系统调用

在MS—DOS4.0中有三个新的或者说增强了的系统调用，用它们来启动和控制任务或进程的执行。

Exec　启动新的任务。任务可以是同步的也可以是异步的。Exec也可为前台进程装入复盖程序。

Aexec　启动新的异步任务。这个调用不能用窗口程序将其中止。

Wait／Cwait　母进程(发送端)等待子进程结束，同时检索从子进程中返回的代码。

Kill　终止一个进程。

新的特点

MS—DOS4.0的一个主要功能是多任务处理。MS—DOS4.0最多能处理32个任务。一个被操作系统处理，其余31个为可用任务。

在MS—DOS3.2中，程序建立一个子任务，但是，在母任务能重新开始运行之前子任务必须结束运行。而在MS—DOS4.0中，母任务即可等待子任务完成，又可以继续它本身的运行。子任务能享有其母任务的环境——包括打开的文件描述和共享存储区指针。这种功能与管道功能一起，使程序可以调用其它程序来完成它的一部分工作。这种设计方式被称为软件工具（Softwars Tools）结构。比如说，一个对文件进行排序的程序并不需要包括一个排序软件包，它可以将系统中的排序实用程序作为一个子进程来调用。

这种结构中的一个非常关键的问题就是影响进程的很多操作同样也会影响由指定进程产生的进程。例如，如果一个当前运行程序X调用了排序程序作为一个子任务，并假定X被冻结，则排序任务也被冻结，尽管建立X的任务或外壳程序对排序程序一无所知。事实上，外壳程序或是X都不知道排序程序或许又调用了另一个子进程。这条规律的一个例外就是前台进程的终止。因为在任何时刻仅有一个前台进程执行，所以只有正在运行的前台进程被终止，它的原始进程才不会受影响。

每一个进程及其它的所有分支被称为一个命令子树。

2.2　并行支持

“并行”是指操作系统可以一次进行几个任务中的任意一个，并可随意在任务间转换。人就是并行处理器的一个例子。一个人一次只能干一件事情，但他可以在几个进行的工作中任意挑选，并可随意转换。

新的系统调用

Rreeze　停止一个程序的执行，一直到它明显地重新启动为止。

Fesume　恢复执行一个“冻结”的程序。

新的特点

当前台程序运行时，为了使后台程序能够在屏幕上显示信息并接收用户的输入，MS—DOS4.0为后台程序提供了一种功能，即在允许前台程序继续运行之前，冻结当前前台程序的执行，且使后台使用屏幕和键盘。这一功能是借助POPUP包来实现的。

为了避免与前台相互干扰，后台进程在运行中有以下限制：

·除非后台进程事先保存了屏幕内容并用POPUP的功能打开了一个上托屏幕，否则它们便不能在CON设备上进行读、写。

·除了INT 9H和2FH以外，后台进程不能设置任何中断向量，如果进程确实需要查找某个关键字，可以设置INT 9H向量。

·后台进程不能使用8087协处理器和仿真浮点，只能用调用接口。

·后台进程可以在急需时分配存储区，但不提倡这样做。最好在开始时即分配所需的存储区，以避免存储区碎化，也避免和前台任务发生冲突。

·许多程序采用两个步骤分配存储区，首先发出一个Alloc系统调用，查找出自由存储区内最大的存储块。然后，再发出一个Alloc系统调用，分配这个存储块。如果在这两个调用间出现一个上下文转换开关，且如果后台任务分配了存储区，那么，前台任务的第二个Alloc系统调用可能失效。

·后台进程必须应用Criterr系统调用使所有INT 24H硬件错误失效。使用缺省时DOS自动地将后台进程产生的硬件错误失效。

·欲在MS—DOS4.0后台环境下并行运行的程序必须用Link4进行连接。Link4以一种新的可执行文件格式来产生程序，通常，这种格式在以前的MS—DOS版本不能运行。

·后台应用程序必须由Exec子功能4系统命令或Aexec命令来调用。Aexec调用避免了用窗口程序来对Exec进行仿真。

·后台进程必须规格化为·exe文件。Exec及Aexec系统调用能认识·xce文件（新规格·Exec文件也可在前台运行）。

·后台进程只能发出异步的Exec调用，为模拟同步Exec调用，后台进程必须先执行一个异步Exec调用，再调用一个Cwait。

·后台进程比前台进程有更高的优先级。前台进程有可能被后台进程排挤。但与CPU相关的那些后台进程不可能被排挤，它们会暂时让位于CPU，以保证其它进程继续被调度。

·INT 11H、12H、21H、INT 2AN和INT 2FH可由后台进程在任何时间发出。INT 10H和INT 16H却只能在进程已打开一个上托屏幕后才能发出。除此以外，后台进程不能发出其它的中断请求。

·后台进程不能使用逻辑驱动器B，因为这将使BIOS为进程产生控制台I／O。

·后台进程不能复盖其代码段。

2.3 内部进程并行性

新的库程序

新的内部进程库程序有：

CritEnter/FCritEnter	封锁一个存储信号
CritLeave/FCritLeave	释放一个存储信号
PBlock	封锁存储单元上的一个进程
PRUN	释放存储单元上的一个进程

新的特点

下列新的库程序是用于联锁在进程间共享的关键数据结构存取，这些程序将存储在Microsoft提供的库内：

CritEnter	Critleave
FCritEnter	FCritleave

用RAM中的一个标志字来操纵这些程序。这些标志字还能同步各种操作且保护共享数据区。这可称为RAM信号机制，它们可从C语言中调用。

对共享存储区的紧密偶合的各进程间，这些程序十分有效，它们可用来控制同一代码块的并行访问。

下述命令则用于封锁与释放特定存储单元上的进程：

PBlock	PRun

PBlock和PRun是用在某一进程需被挂起直到某个特定事件发生为止的情况下。这须由协同操作程序完成：一个程序和DOS系统或一个程序和BIOS系统。这是因为，调用 PRun 程序必须要知道封锁的内存地址。

2.4 进程间的通讯

新的系统调用

Pipe	用于建立一个管道。
Creat Mem	用于建立一个共享的存储区。
Get Mem	获得对已存在的共享存储区的存取权。
Release Mem	释放对共享存储区的存取权。

新的特点

通过管道结构和共享存储区，MS—DOS4.0为用户提供了进程间通讯的功能。

管道

一个管道即是连接两个或两个以上进程的串行的通讯通道，它是一条信息管道。写任务将数据信息送入管内，而读任务则从管内取走信息。管道的输出／输入是通过标准的系统Read／Write调用来实现的，它与通常的读／写文件相同。这样做是经过考虑的，因为它允许一个任务与控制台、磁盘文件，其它任务以及网络上的任务进行相互通讯。管道是由Pipe系统调用来建立的。Pipe调用在管道的两端都可返回。仅当某一进程或某些进程将它打开时，管道才存在。任何其它任务都不能将管道打开，因为没有名子。当所有的进程关闭它们的操作以后，管道被删除。被管道占用的存储区便被重新分配。管道不在磁盘上进行缓冲。

由当前任务产生的任何子任务都将继承其母任务选定的管端，这就使母任务可以运行子任务且从子任务中读／写数据。母任务为管道通道的典型安排为 STDIN和STDOUT 通道。

因此，子任务便不知它正被某一任务调用。用这种方式，任何程序（例如排序程序）都可被调用去为其它程序服务。

共享存储区

MS—DOS4.0支持一个共享存储区。CreatMem系统调用按所要求的长度分配一个存储段，并记录下它的名称。一旦建立以后，这个共享的存储区便可由其它进程通过GetMem系统调用进行访问。共享存储区通常允许两个或两个以上的密切协同的进程访问共用的表格及缓冲区。

共享存储区的名称占用文件系统名称的空间，但目前限制为用／SHAREMEM／开头。这一限制确保了同将来的增强结构在共享存储区时的兼容性。

2.5 上托包

新的系统调用

Checkpu	检查上托包是否装入。
PostPu	打开／关闭上托会话。
SzvePu	保存上托屏幕。
RestorePU	恢复上托屏幕。

新的特点

上托包是一个终止保持常驻的程序，它在前台区运行却同时允许后台区应用程序得到对屏幕和键盘的控制。上托屏幕使后台的应用程序与用户之间进行通讯，用户也可以和后台程序进行通讯。这是通过后台应用程序截取键盘中断并调用上托包来实现的。

MS—DOS4.0在运行后台应用程序时，对这些程序有两条限制。

①后台应用程序能通过上托机制访问屏幕和键盘。

②后台应用程序被置于屏幕和键盘的控制以后，也只能在屏幕上进行写操作，在键盘上执行读操作。

为了使用上托包，后台应用程序调用下列功能：

- CheckPU　用于检查POPUP包是否装入。
- PostPU　冻结和恢复前台应用程序，打开或关闭上托屏幕。
- SavePU　保存当前屏幕内容、键盘状态及送入上托包的约定区域或由应用程序所指定的区域的内容。
- RestorePU　恢复屏幕内容、键盘状态以保存在上托包的约定区域或应用程序所指定区域的信息。

在向PostPU要求的开启或关闭上托屏幕过程中可多次调用SavePU和RestorePU。

2.6 调度程序

新的系统调用

Sleep	将某一任务的执行延迟一定的时间。

SetPri　　得到或设置一个任务的优先级。

GetPID　　回送一个进程的ID码。

新的特点

• 只有当系统调用停止且设备驱动程序在特定环境下退出时，后台进程才会被排挤。后台进程也只有在它们被冻结、封锁时才让出CPU。

• 进程利用一个零超时来发出Sleep系统调用，从而让出CPU。

• 设备驱动程序可以通过调用DevHlP功能：Yield，来让出CPU。

• 只有前台进程才会在中断时被排挤。

• 编写MS—DOS4.0调度程序是有利于后台进程的执行。如果一个后台进程执行时间过长，它就将暂时让出CPU以便其它进程得以调度。

调度算法

每一个任务的调度参数是按下列公式计算出来的。通常，每秒钟进行一次。所有计算的参数和用于计算的变量都是由任务而定的。

Slice: = (19 − Bmode) / 4

Runpri: = Priforce? Runpri:(CPUuse/2 + Bmode + InBackGround? 32:0 + Scrnio? 32:0 + Upri * 5)

• BPUmode表示任务的I/O的局限性。它用来减小给I/O限制任务的时间片的大小并提高它们的优先级别。当一个任务建立时，Bmode被置初值0，它可在0—15间变化。当任务在I/O中封锁一次，Bmode就加1，而当任务每用完一次它的时间片，Bmode就减1。

• CPUuse是表述一个任务使用CPU历史的变量，它的初值为20，可在0～26间变化。任务每用去一个时钟它就减1，且每过1秒它又加20。CPUuse可告诉调度程序任务已使用了多少个CPU时间。占用CPU时间越多，CPUuse值则越小。如果一个任务现在还没有运行，CPUuse就将增强加其优先级。

• InBackground是某个任务正在后台运行的标志，使后台任务的优先级增高。因为这样前台进程便有一个在空循环中等待的趋势，而不是象在正常的多任务系统中那样阻碍事件发生。此外，后台进程或许有紧急需求以便得到CPU。

• priforce也是一个标志。一但建立这个标志，就告诉调度程序当用前的Runpri值来代替重算的Runpri值。若4秒或更长时间，某个任务都被CPU拒绝接受，Runpri就被置为255，且priforce即被设定以确保下一次运行这个任务。不管什么时候，只要是睡眠的任务被唤醒，它的Runpri就加20且priforce被设定。当任务用完它的时间片时priforce即被清0。这种机制使任务拥有一个暂时的优先级增加，应用时不影响任务正常的优先级别。

• Runpri是任务的优先级。拥有最高优先级的任务总能被执行。上述的公式即是用来计算任务的优先级，通常Runpri在0～247之间变化。这条规则的例外情况参看Priforce的有关说明。

• Scrnio是一个标志符号，当一个任务在最后两个时钟时控制台I/O操作即置“1”该标志。强调这种进行控制台I/O操作的任务是为了使与用户的交流以一种定时的方式进行。

• Slice是一个任务被允许运行的定时器时钟的数量。它在1～4之间变化。一个任务的I/O约束条件决定了它的Slice值，I/O约束越多，Slice越短。

·Upri是任务的用户自定义优先级，其初值为16，在0～31之间变化。由Setpri系统命令设置。请注意，在优先级计算公式中用户自定义优先级的乘因子很大，使它的影响大于公式中其它几项。

MS—DOS4.0和中断向量管理

DOS系统不能有效地对时钟中断向量进行管理，因此DOS必须被置为处理时钟中断的最后一个处理程序。

MS—DOS4.0只进行少数中断向量管理，这种管理与BASIC以及一些通用的即终止又保持常驻的程序相干扰。因此MS—DOS4.0唯一参与管理的向量是INT21H。

无论何时当DOS上下转换开关从前台进程转向后台进程，DOS都保存当前的INT21H向量。用约定的INT21H向量来代替。当DOS返回到前台时，被保存的INT21H即被恢复，这就使后台进程可方便地访问DOS，而且同时又使前台进程仍然掌握着INT21H。

DOS不管理INT8H向量（时钟中断），调度程序便成为INT8H通路的最后一个时钟中断处理程序(而不是第一个)。将调度程序作为INT8H通路的最后一个处理程序有以下两个附带的影响：

·调度程序将不关心那些改变时钟中断频率的程序。改变时钟频率的程序仍然会掌握住时钟中断并以正常的频率将它们送至通路中的其它处理器。(BASIC和窗口都是这样做的)作为INT8H通路上最后一个处理器的调度程序将以正确的频率接收时钟中断信号。这点是很重要的，因为它无法判断时钟中断频率是否改变。如果调度程序以一个不正常的频率来接收时钟中断，那么调度程序的特性和内务操作以及其它进程的响应性都会大受影响。

·DOS系统不能排挤后台进程。仅当后台进程处于下列情况时，它们才能放弃CPU：

——被冻结或封锁

——明显地让出CPU

——发出系统调用

通过调度程序响应时钟中断，在调度程序响应时钟中断时，一个额外的中断结构的任意号码或许会先于调度程序由被调用的INT8H处理程序放在堆栈内。假若调度程序将要排挤一个以上的进程，这些中断结构的考贝就可能会被放入几个进程的堆栈之中，这就将导致与这些结构有关的处理程序重复地、且在不需要的时候多次被调入。假如这些处理程序不是可重新进入的，这种重复进入将会带来不希望的结果。

2.7 存储管理

新的系统调用

Partion 设置前台及后台存储区的大小。

新的特点

Microsoft的存储管理提供了如下功能：

·前台和后台的存储分区

·纯代码段的自动共享

存储区