MS-DOS内部结构与RAM盘程序开发

王玺玉　编译

科海培训中心

八九．四

MS-DOS 内部结构与RAM 盘程序开发

王玺玉 编译

科海培训中心

八九·四

序　言

MS-DOS (Microsoft Disk Operating System) 自1981年开始在IBM-PC上采用以来，为众多16位计算机利用作为操作系统软件，几乎占领了整个微机市场。当初发行的MS-DOS 1.25版型是继承CP/M（美国Digital Research公司的产品）的谱系，比较集中地反映了其功能的16位CPU的DOS，而后来的2.0版型是受到UNIX（美国AT&T的贝尔研究所开发）强烈影响，功能得到大幅度扩充的型号。其后，2.10、3.00、3.10……等相继问世，功能不断扩充，流行范围越来越广，而其源皆出于2.0版型。本书所述MS-DOS即以此2.0版型为标准，并顾及到以后系统的发展。

MS-DOS之所以能在16位计算机上得到普及，是由于被IBM-PC采用、价格低廉、并由Microsoft公司提供优良的应用等。但是，更重要的原因是DOS本身功能的卓越。本书所述MS-DOS内部结构为MS-DOS的核心，具有丰富的技术信息，是了解DOS、分析问题、扩充系统、开发软件必须熟悉的内容。

另外，为便于读者有效地利用MS-DOS的内部信息，掌握设备驱动程序的设计和安装方法，本书最后一章以实际RAM盘程序设计为例，详细阐述了块型设备驱动程序的开发过程。

对于本书的了解需要具有关于计算机系统结构和汇编语言等方面的基本知识，请读者根据需要参照相应文献，本书不再赘述。

本书参考文献列于最后。主要参照《MS-DOS HANDBOOK》(1988年4月第1版第16次印刷）和《MS－DOS PROGRAMMER'S HADBOOK》（1986年9月第1版第7次印刷）二书，在日本广泛流行。本书内容深入浅出，有各种概念图表和上机操作的实例清单，大量的图示解说便于初学者理解和模仿操作，适于自学。而对一般编程人员也可从图表中简明地查找技术数据，兼有指南和手册的功效，希望本书为读者在有效地利用MS-DOS上能有所裨益。由于编译者水平所限，书中错误之处欢迎批评指正。

随本书发行备有51/4吋MS-DOS系统宏汇编工具软盘片，以便于用户编制机器语言程序。软盘片上附有开发的实用RAM盘程序，可利用本书提供的方法安装运行，使用户不增加任何设备投资而获得一个快速盘（虚盘）设备的使用效益。

感谢科海培训中心对本书出版给予的支持。

编译者　1988.12

目　录

第一章　MS-DOS文件的软件体系

随着构成MS-DOS各种软件功能和其相互关系的叙述，本章顺序讲解MS-DOS的所有控制模块怎样启动和存贮器如何分配。

第二章以后分别阐述各模块的详细内容。

1.1　MS-DOS的基本构成

只在命令级利用MS-DOS时，不会感到MS-DOS系统的存在。而真正使你感觉不到它存在的，才能说是优秀的DOS。MS-DOS也正是考虑到这点，把称为"MSDOS.SYS"，"IO.SYS"的控制模块作为隐含文件，使它成为用户眼中看不见的。

然而，用汇编开发软件和要扩充外围装置的时候，需要理解构成MS-DOS各模块的功能和关系。

作为了解MS-DOS内部构造方法的开始，本节先简单地说明关于"MS-DOS"、"IO.SYS"和"COMMAND.COM"三个模块的概况和相互关系。

1.1.1　COMMAND.COM（命令执行部分）

"COMMAND.COM"具有所谓解释、执行用户输入命令的功能。若输入命令为内部命令，则转移到"COMMAND.COM"的各命令处理例程；若为外部命令，则从盘进行装载。

1.1.2　MSDOS.SYS（文件管理部分）

"MSDOS.SYS"是构成MS-DOS的核心部分，进行文件、输入输出和系统调用的管理。

在MS-DOS上支持的输入输出装置有键盘、显示、打印机、辅助输入输出、盘驱动器（5吋/8吋软盘、硬盘）、鼠标器、实时时钟等。"MS-DOS.SYS"把这些外围装置分成几类，以便可类似使用的尽可能由同一的程序处理（如屏幕显示和打印机等）。此外，"MS-DOS.SYS"还具有对盘上的文件和设备文件等的文件处理、存贮器管理以及进程管理（程序的装载、结束代码的处理）等功能。

1.1.3　IO.SYS（输入输出控制部分）

"IO.SYS"又称BIOS（基本输入输出系统），响应"MSDOS.SYS"的输入输出要求，进行实际的输入输出操作。

给"IO.SYS"发出输入输出调用（BIOS调用）由"MSDOS.SYS"进行，一般用户不调用"IO.SYS"。BIOS调用的形式，譬如说，把某个驱动器中盘的某扇区传送给指定的地址。"IO.SYS"接受此命令，向FDC和DMAC等发出命令，而后把结果（读出/写入正常否，出错及种类等）返回给"DOS.SYS"。

"IO.SYS"使用标准设备为，盘、控制台、打印机、辅助输入输出和实时时钟等5种（图1.1）。

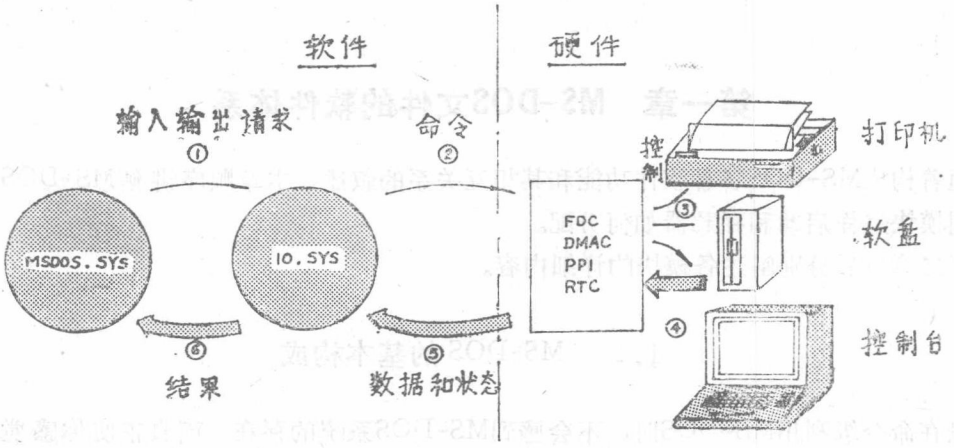

图 1.1 MSDOS.SYS和IO.SYS

1.1.4 各模块的关系

至今叙述过的三个模块，常驻在存贮器上，辅助执行外部命令等。这里执行外部命令，以受理来自控制台的1个字符输入为例来考虑各模块的关系。

首先，外部命令向"MSDOS.SYS"请求1字符输入。接着，"MSDOS.SYS"接受此请求，向"IO.SYS请求1字符输入。而后，"IO.SYS"从键盘输入1字符，并把该字符送往"MSDOS.SYS"。最后，把所要的字符送往发出最初请求的外部命令，该程序应该连续处理。图1.2汇总了这些模块的相互关系。

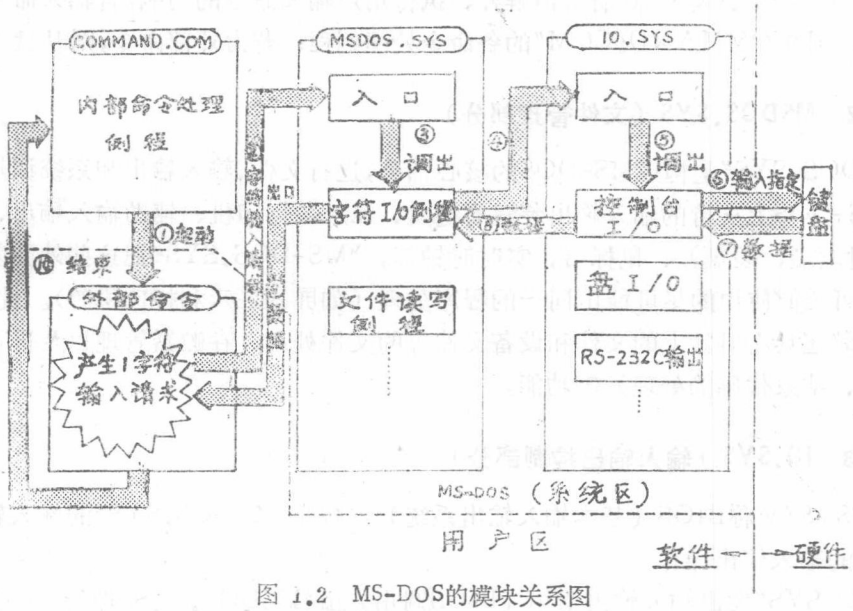

图 1.2 MS-DOS的模块关系图

1.2 MS-DOS的存贮映象和引导程序（起动机构）

如前节所述，MS-DOS的全部操作包含在其软件体系的程序里。作为"MSDOS.SYS"、"IO.SYS"、"COMMAND.COM"三个模块，被控制、分配在存贮器指定的位置

上。因此本节对控制MS-DOS的各程序，从引导MS-DOS时开始依次去看它们在存贮器中是怎样分配的。

1.2.1 引导时的存贮映象

首先，把系统启电或刚复位（RESET）后的存贮器状态示于图1.3。

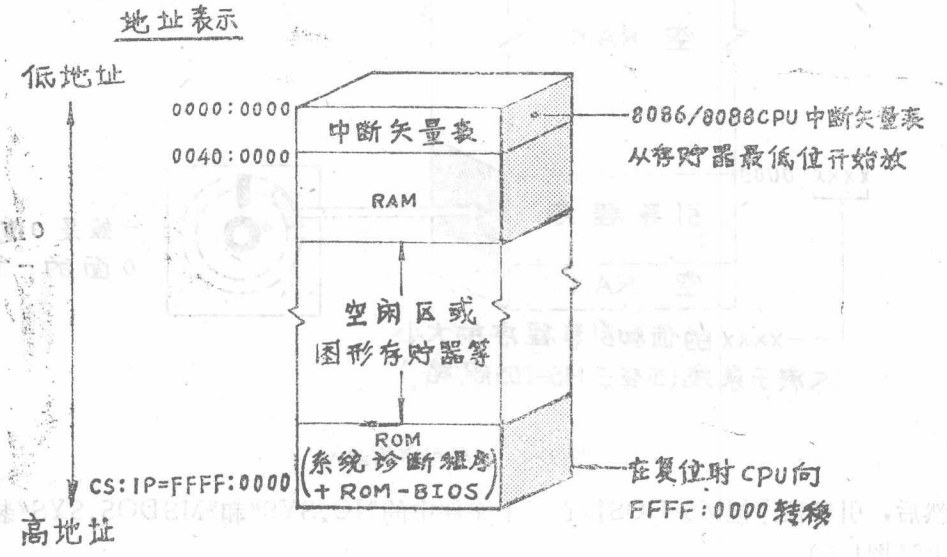

图 1.3 复位后的存贮映象

8086 CPU的结构，由于中断矢量表从存贮器地址的最低位开始放，故不论那个系统这种RAM大都从系统的低位地址（段＝0000H）开始占用，ROM放在存贮器的最高位。

RAM最低位的1K字节存放着8086 CPU中断矢量表，用户能自由使用的区域从段地址0040H开始。

假如从段地址0000H开始没有RAM，那就不能使用DOS。另外，在中间既或一部分存贮器上有不连续区域（例如，从段地址1000H开始仅没有64K字节的时候），则其不连续区域以后就不属MS-DOS管理。

1.2.2 引导程序

在ROM中记录有进行通常系统检查的例程（系统诊断程序）和ROM-BIOS。由于制作的BIOS不仅能对应MS-DOS，也能对应CP/M和其它的OS，所以"IO.SYS"实际上不需要直接操作FDC.DMAC和时钟设备等，而是通过这些ROM-BIOS操作盘、键盘和打印机等。这样做的优点是"IO.SYS"和CP/M中的BIOS等容易制作，而且DOS的尺寸也会变小。

且说，刚复位后ROM中被启动的程序，首先建立ROM-BIOS（开始准备使用）和检查硬件（外围设备连接正确否？RAM运行正常否？等等）。

其后，ROM-BIOS从系统盘把进行引导的程序装载到存贮器上（图1-4）。

MS-DOS的引导程序写在各盘的0道、1扇、0面。因为其内容随厂家不同而有可能不同，所以用不同厂家的MS-DOS系统盘时引导操作不能正确进行。

当该引导程序被正确装载之后，控制就从ROM-BIOS转移（JMP）到该程序。

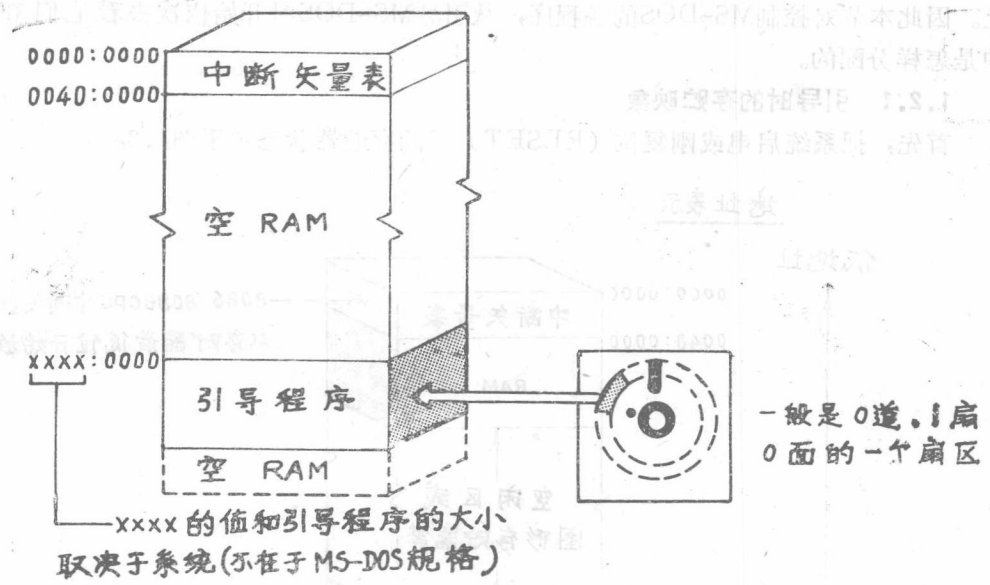

图 1-4 引导程序的读入

然后，引导程序把MS-DOS核心三个文件中的"IO.SYS"和"MSDOS.SYS"装载到存贮器上（图1.5）。

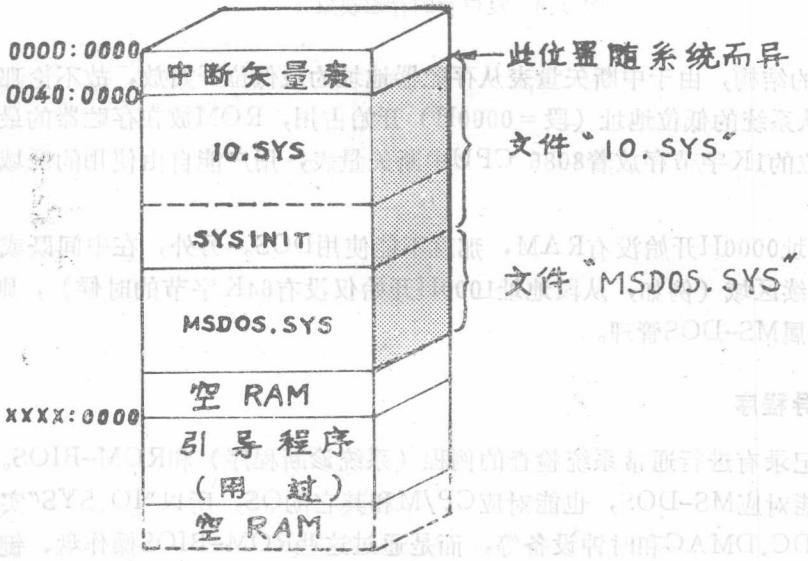

图 1-5 引导

其中"IO.SYS"由系统常驻部分（IO.SYS）和仅初始化使用的例程（SYSINIT）二者组成。

1.2.3 IO.SYS例程的初始化动作

引导程序的处理在图1.5所示的存贮映象状态下结束，之后控制转移给"IO.SYS"例程。

IO.SYS例程掌握连接着的外围装置的状态，而且同时进行各驱动例程的初始化。例如，掌握系统上连接的盘驱动器的种类（是8吋还是5吋？是单面还是双面？是硬盘还是软盘？等）和驱动器的数量，或是进行键盘通信用板（16位机器在键盘控制上大多使用专用CPU）、打印机和RS-232C控制器的初始设定等。

还有，检查系统能使用的RAM容量也是这部分的工作。这些检查结束，控制就转移到SYSINIT例程。

SYSINIT例程把自己本身重新装载到存贮器的最高位，然后把"MSDOS.SYS"移到由SYSINIT例程移动后空下的部分（确切地说，在"IO.SYS"初始化部分所指定的量，通常正是SYSINIT例程的大小），看图1.6。

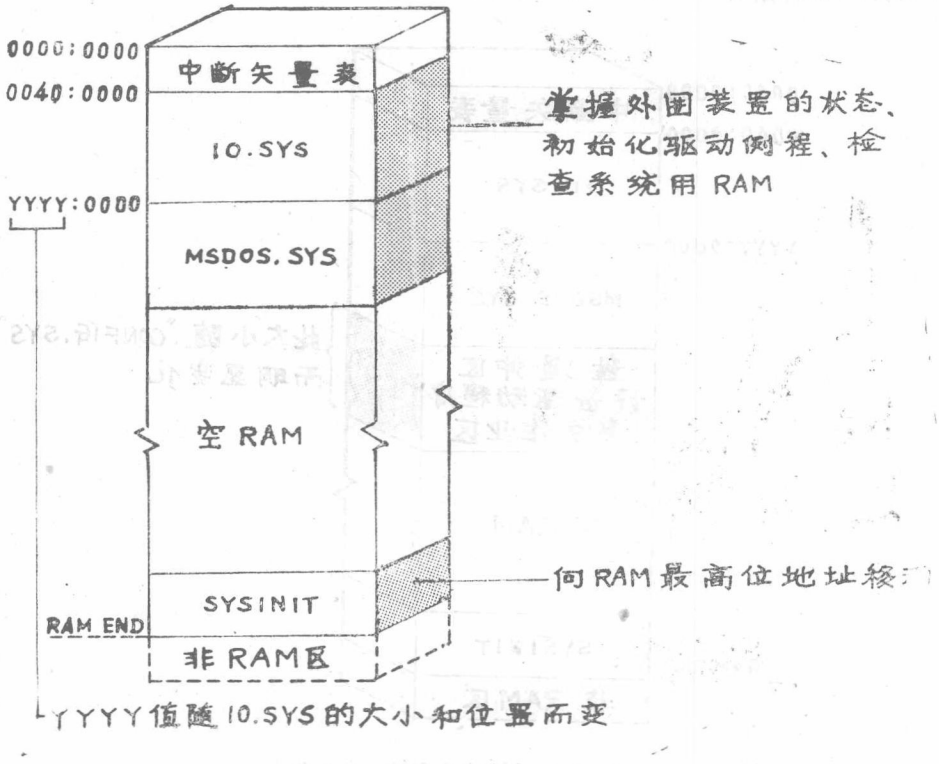

图 1.6 "IO.SYS"的初始化

至此，由于"MSDOS.SYS"在存贮器上的位置已定，故SYSINIT例程进行"MSDOS.SYS"的初始化。具体上，应该调用"MSDOS.SYS"的初始化例程。

1.2.4 MSDOS.SYS的初始化

"MSDOS.SYS"的初始化例程确保用于管理所连接的标准外围设备（控制台、打印机、盘驱动器、时钟设备、通信电路）的作业区。另外，也确保处理文件用的管理表（系统表）和向盘传送用的缓冲区等。除此之外，为支持系统调用，在进行各种初始化之后，再次把控制返回到刚才的SYSINIT例程。

由于以上的一系列初始化动作，现在MS-DOS功能中的系统调用（除存贮器管理调用）就全可使用了。

1.2.5 CONFIG.SYS的设定

SYSINIT例程，在这里要读入文件"CONEIG.SYS"。该文件（Configuration File）是任选，没有也无关。但是当用户给系统附加新设备驱动程序，或增加能用DOS同时处理的文件时是需要的。

读、解释、执行"CONFIG.SYS"文件由上述的SYSINIT例程完成。新加给系统的设备驱动程序在存贮变换时，跟在"MSDOS.SYS"后面装载。另外，在"CONFIG.SYS"中能指定盘缓冲区（为临时放置向盘送取数据的区域）数量，而这些缓冲区仍然选在"MS-DOS.SYS"后面的存贮区内。

SYSINIT例程解释"CONFIG.SYS"之后的存贮器状态示于图1.7上。

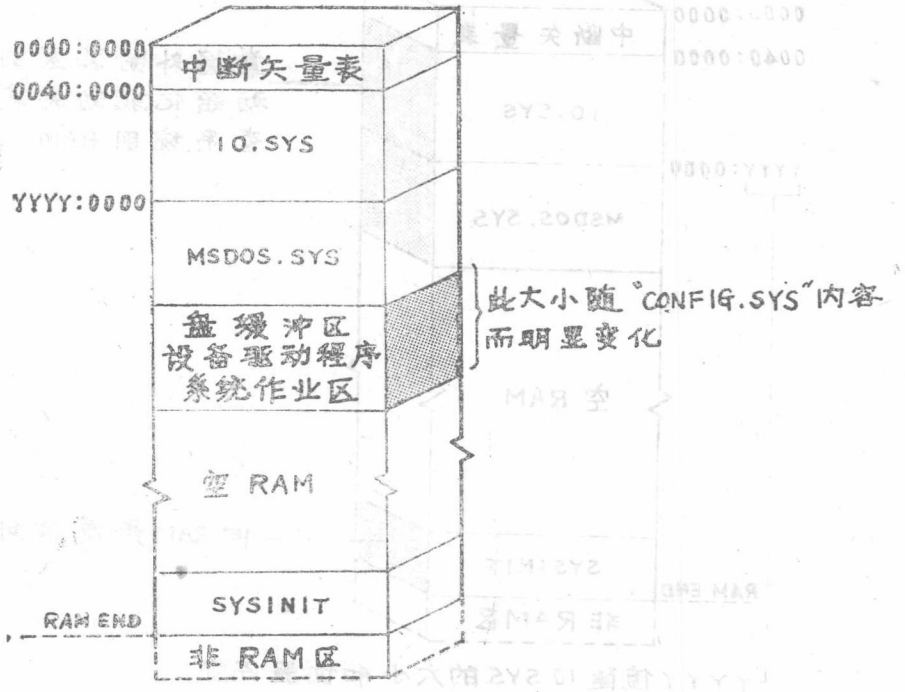

图 1.7 MSDOS.SYS的初始化

1.2.6 COMMAND.COM的装载

"IO.SYS"，"MSDOS.SYS"的初始化动作在图1.7状态下全部结束。而后把"COM-MAND.COM"装载到存贮器上。若转移控制，则初始化处理应该全部结束。

"COMMAND.COM"也和"IO.SYS"同样由二部分组成，但由于初始化例程和所谓本体没分开，故以常驻部分和非常驻部分区别之。因此由SYSINIT例程装载、控制转移向"COMMAND.COM"后，首先开始把非常驻部分从盘传送到如图1.8所示的可用存贮器的最高位。

"COMMAND.COM"的非常驻部分解释、执行用户的输入命令。DIR和COPY等内部命令全由该"COMMAND.COM"处理。

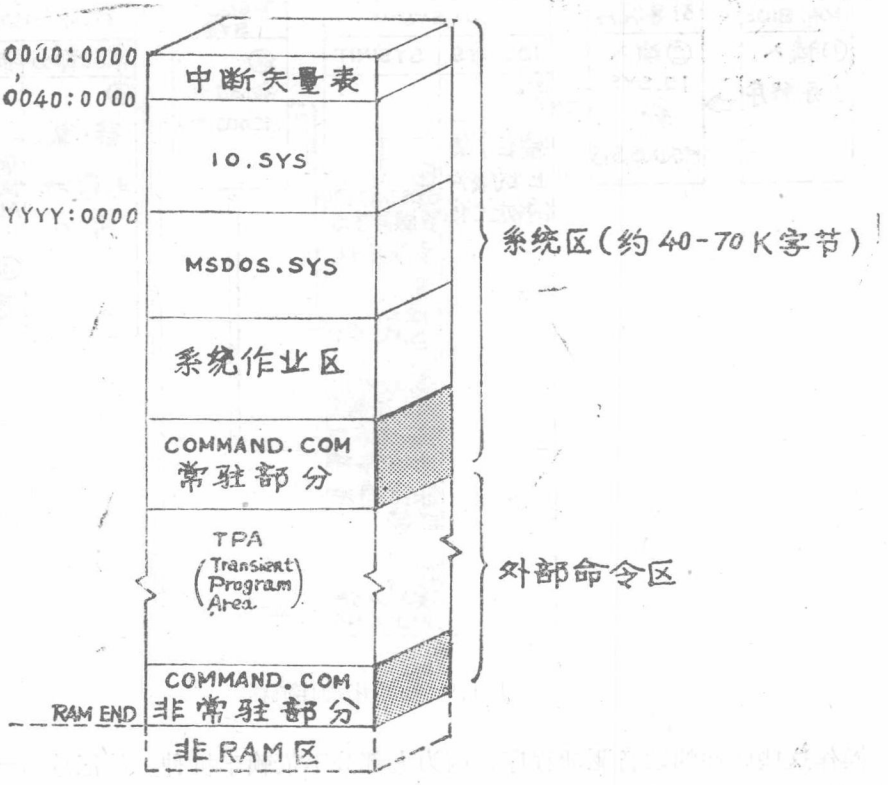

图 1.8 COMMAND.COM的装载

在CHKDSK和DEBUG等外部命令的场合，这些程序装载到图1.8的TPA（Transient Program Area）处，控制转移到该外部命令。这时该程序能使用的RAM从被装载的位置到RAM的最高位。即使"COMMAND.COM"的非常驻部分被破坏（使用）也无关紧要。

外部命令执行完了的时候，由于非常驻部分有可能被破坏，故控制转向常驻部分。常驻部分利用检查和办法检测非常驻部分是否受到外部命令破坏，如果必要的话再装载非常驻部分。当然，若知道没被破坏就不进行再装载，所以和每次执行外部命令装载相比，还是"COMMAND.COM"的反应为快。

1.2.7 初始化动作的意义

前面解释说明的初始化动作看起来相当复杂，汇总起来如图1.9所示。

这样，MS-DOS分别由几个文件构成，虽然各文件能分担功能，但也造成更加复杂的相互关系。这样做的理由如下：

首先考虑为什么分成三部分（COMMAND.COM/MSDOS.SYS/IO.SYS）。

某种DOS为了普及，提出的第一个条件就是便于使用。此外，能在多种系统上工作也是重要因素。为此，需要尽可能少的依赖硬件。DOS支持的硬件就是受到相当限制的。MS-DOS上，控制台（屏幕+键盘）、盘、打印机、另外也许连接了支持的通信回路和时钟设备等五种成为基本设备。

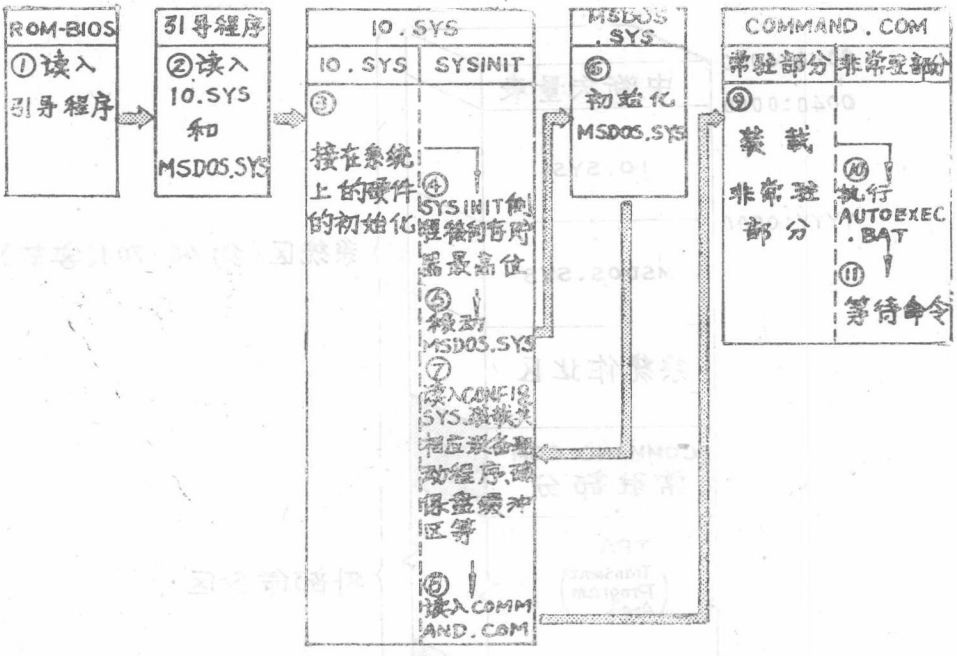

图1.9 初始化动作流程

操作这些硬件的设备驱动程序，因为大部分不依赖于机种，故汇总为一个文件，就是"IO.SYS"。"IO.SYS"因为只支持上述的五种设备，而不能支持用户新附加的设备，为此，MS-DOS上让用户能编制成设备驱动例程。然而因为把新编制的设备驱动程序每次装入"IO.SYS"都需要用户或支持设备的厂家花费工夫，所以备有称为"CONFIG.SYS"的文件，只要在其中预先登录好设备名(用编辑程序能简单地登录)，就会使系统自动装载好。

其次，考虑关于不依赖机种的部分

该部分有处理系统调用的例程（设备、文件和存贮器等的管理）和解释、执行用户输入命令的例程。其中前者在"MSDOS.SYS"上，后者在"COMMAND.COM"上执行。"COMMAND.COM"另外成为文件，这主要起因于其功能(MSDOS.SYS和COMMAND.COM的作用完全不同)，而为了能根据需要和新的命令处理程序置换也是理由。

在MS-DOS的最新版型上有个叫作VISUAL SHELL的命令处理程序，在功能上和"COMMAND.COM"大致相同，但其大部分操作能使用鼠标器（Mouse）进行人机对话(INTERACTIVE)。

为了置换使用像这样的命令处理程序也必须把"COMMAND.COM"和"MSDOS.SYS"分开。

如上所述，若综合考虑前面的引导过程，就能理解这样做是妥当的。

首先开始给存贮器装载"IO.SYS"和"MSDOS，SYS"，顺序进行初始化动作。其次，采用"MSDOS.SYS"的系统调用打开文件"CONFIG.SYS"，根据其内容装载设备驱动程序。

然后，SYSINIT例程消失，而控制转移到"COMMAND.COM"。"COMMAND.COM"的非常驻部分移往存贮器的最高位，确保尽可能大的TPA，其结果常驻部分以后应该是用户（的程序）全部可用的TPA。

第二章　COMMAND.COM

MS-DOS的命令处理程序"COMMAND.COM"采用叫作I/O再定向(REDIRECT)和管道（PIPE）的UNIX外壳（SHELL）的概念。本章以"COMMAND.COM"的特征为中心说明其有关构造。

2.1　命令的输入和执行

"COMMAND.COM"上有所谓实行用户输入命令的功能。例如，在控制台输出提示符，用户若打入"DIR"，则在屏幕上显示出当前驱动器上存放的盘的文件名清单，再等待下面命令输入，这就是实行输入命令的功能。另外，输入的命令名若不是DOS的内部命令，则从盘装载其外部命令使之执行。下面讲述用户在命令行输入命令之后，"COMMAND.COM"怎样解释和执行它的过程。

2.1.1　文件的执行顺序

省去I/O再定向和管道功能的场合，在命令行输入命令的一般形式如下：

〈命令名〉［〈参数〉…］

其中对"COMMAND.COM"有意义的只是〈命令名〉，而参数串则仅交给被启动的命令。

作为命令名有内部命令和外部命令，后者再分为三种：

（1）COM文件

（2）EXE文件

（3）BAT文件

三种类型的文件分别以文件名的扩充部分".COM"，".EXE"，".BAT"区别之。

因为内部命令和外部命令总共有四种，所以预料到有时存在相同名字的命令，即相对内部命令的COPY，可能有"COPY.COM"、"COPY.EXE"和"COPY.BAT"的命令文件。这时四种命令以如下顺序检索。

内部命令—COM文件—EXE文件—BAT文件

这个检索顺序中，通常最优先检索内部命令。这是因为判定其它命令伴随着盘的存取，需要花费时间。而"COMMAD.COM"常驻在存贮器中，所以判定是否内部命令很快就完。

EXE文件和COM文件都是用8086 CPU机器语言叙述的命令，它们在这点上是一致的，执行时存贮器的使用方法稍有差异。详情请参阅第三章。

"COMMAND.COM"基本上只是对应接受的用户命令执行内部命令或让外部命令执行，动作概要如图2.1。

此图是相当大略的流程图，在本节以后的解释说明中将严格化。

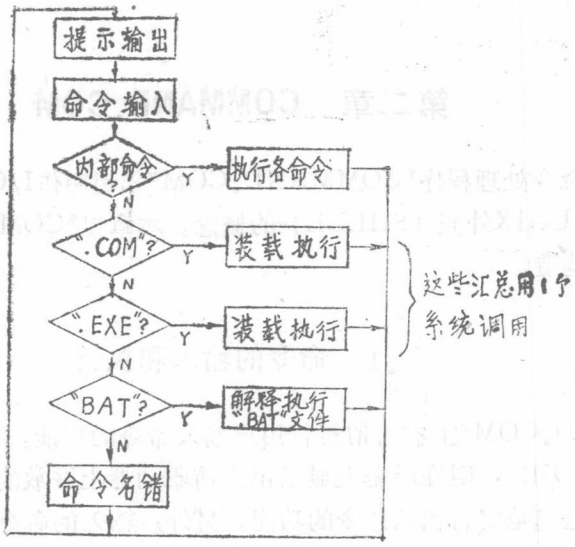

图 2.1 COMMAND动作概要流程

2.1.2 路径的检索处理

MS-DOS能指定对命令检索的路径名。例如，若设置如下的命令搜索路径（¥=＼）

PATH A:¥ BIN; B:¥BIN

则指定的命令（文件）在当前目录上没有时，就搜索"A:¥BIN"（目录名）。若在这个目录上也没有，则搜索"B:¥BIN"里面，若此处也没有时，就输出表示该命令不存在的出错信息。

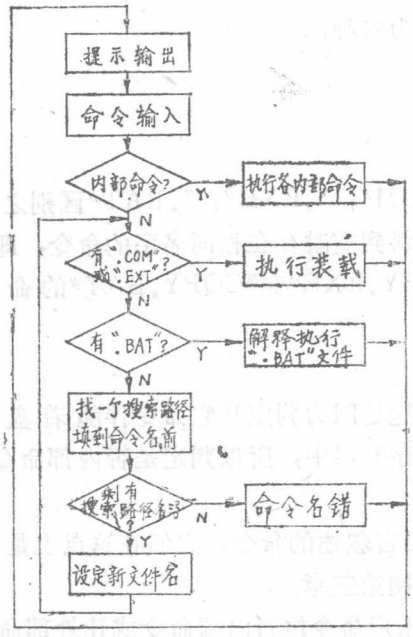

图 2.2 命令动作概要流程之二

为使前面的流程图（图2.1）符合这里的情况，并不需太大改变。即当检索的文件在指定目录中没有时，每次给出设定的路径附加到命令名前面，编制新路径名，再次检索文件就行了（给路径名，检测其文件是否存在，可使用MS-DOS的系统调用）。例如命令名为XYZ的时候，搜索下面文件（图2.2）。

XYZ.*
A: ¥BIN ¥XYZ.*
B: ¥BIN ¥XYZ.* ("*"相当COM, EXE, BAT三种)

2.1.3 批命令的处理

虽然叫批处理，但不需要考虑什么特别的事。批处理与普通的命令输入不同之处，只是输入方法不同而已。若考虑前面的流程图（图2.2）仅"输入命令"部分不同，不是来自

键盘而是由文件进行，后面流程完全相同。

为了表示命令输入是由哪儿进行的，假定备有叫CIN（当前输入文件）的区域。在其中，当前输入若是控制台就放入"CON"；若在批文件的处理中，像"TEST.BAT"那样，放入当前处理中的批文件名。

该CIN区当"COMMAND.COM"起动时为"CON"，而在批文件处理中分配其处理中的文件。因此，如叫"SAMPLE.BAT"的文件在处理中，把叫"CIN←SAMPLE.BAT"的文件名代入到CIN上的话，则由于命令输入基本上只从文件和CIN取出来一行就完了，所以对于当前由哪儿进行输入即或不知道也没关系。

但是，因为用批命令必须处理参数和环境变数，所以在命令输入部分进行这些置换。这种处理，从文件取出来一行命令，检测其内容，若为"%〈数字〉"则用"parameter[数

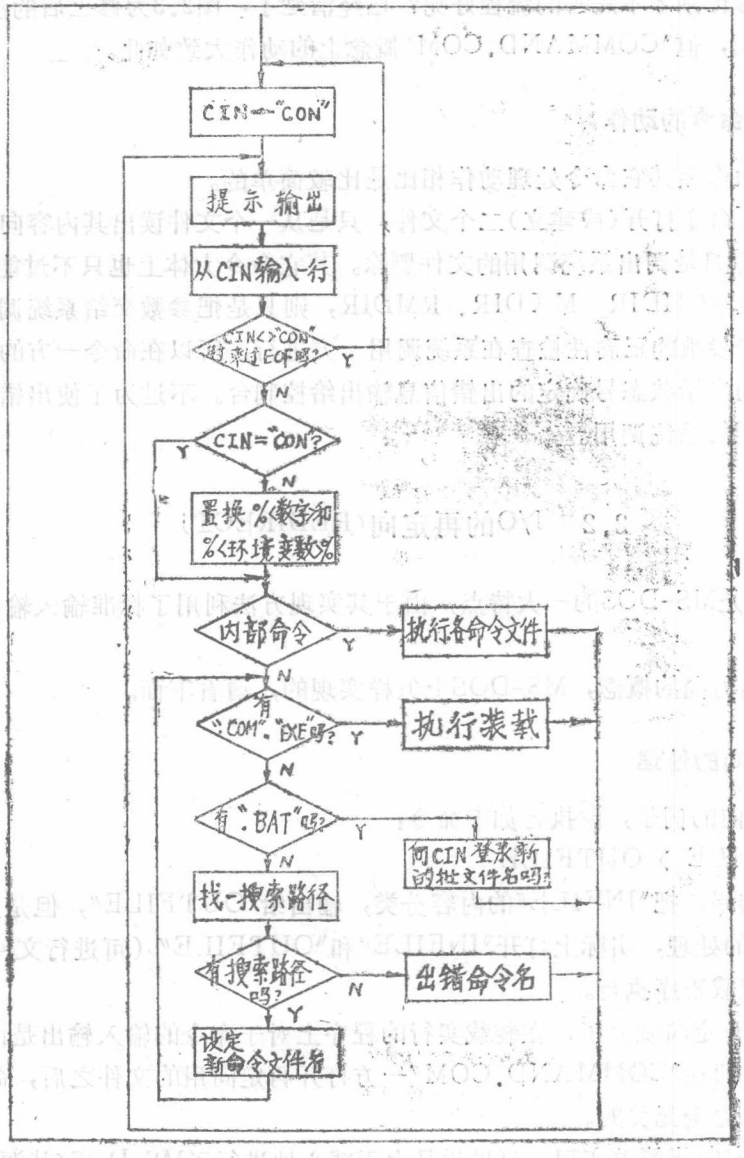

图 2.3 COMMAND动作流程

字]〞置换，为〝%〈环境变数〉%〞则把它用〝environment [环境变数]〞置换。而后若把它送给命令解析例程就可以了。但是，〝parameter [n]〞，意味对应批文件第几号的参数（的字符串）；〝environ ment [环境变数]〞，指在〈环境变数〉上设置的字符串内容（环境由SET命令设置）。

因此，批文件中的1行字符串如

```
SAMPLE %1 %TTY% ABC
```

被置换成下列形式后送给命令解析例程。

```
SAMPLE ARG1 VT 100 ABC
```

此外，有关环境的环境变数和字符串，以及〝COMMAND.COM〞保存在另外的存贮区，而批文件的参数必须先用〝COMMAND.COM〞本身正确管理。

至此，怎样修正刚才不完善的流程好呢？已经清楚了，图2.3为修正后的流程图。细节与前面稍有不同，但〝COMMAND.COM〞概念上的动作大致如此。

2.1.4 内部命令的动作

内部命令的动作和其它命令处理动作相比是比较简单的。

例如，COPY命令打开(或建立)二个文件，只是从一个文件读出其内容向另一个文件写入；而DEL命令只是调出系统调用的文件删除。其它命令大体上也只不过能实现几个系统调用的组合。至于CHDIR、MKDIR、RMDIR，则只是把参数交给系统调用后动作便结束。此时，由于参数的正确性检查在系统调用一方进行，所以在命令一方的处理只是把根据返回寄存器的出错状态号确定的出错信息输出给控制台。不过为了使出错管理简单，也要考虑用功能强的系统调用。

2.2 I/O的再定向(REDIRECT)

I/O的再定向是MS-DOS的一大特点，由于其实现方法利用了标准输入输出，所以处理很简单。

转换输入输出方向的概念，MS-DOS上怎样实现的，请看下面。

2.2.1 再定向的处理

作为I/O再定向的例子，设执行如下命令：

```
SORT <INFILE > OUTFILE
```

命令的执行结果，把〝INFILE〞的内容分类，输出给〝OUTFILE〞，但是这时〝COMMAND.COM〞的处理，实际上打开〝INFILE〞和〝OUTFILE〞（可进行文件的输入输出），而后只是装载程序执行。

仅如此I/O的再定向就完了，在装载实行的程序上对于命令的输入输出是由何处进行的完全勿需干预。即在〝COMMAND.COM〞一方打开再定向用的文件之后，命令和输入输出方法相互间是完全无关的。

这样I/O的再定向能简单实现，可以说是由于精心地进行了MS-DOS(其源为UNIX)的标准输入输出概念设计。

2.2.2 标准输入输出和再定向

在MS-DOS上，一次打开的文件叫文件句柄(File Handle)相当BASIC上使用的文件号。表2.1的0～4是对应各种标准输入输出文件句柄的值。该值不是来自机器语言级，不可就这样使用，但是此处为说明"COMMAND.COM"而相应地予以引用。

表 2.1 标准输入输出和编号

0	标准输入
1	标准输出
2	标准出错输出
3	标准辅助装置
4	标准打印机

例如 SORT命令称为由标准输入输入数据向标准输出输出的操作，就是由文件句柄0的文件输入向文件句柄1的文件输出。即SORT命令与现在由哪儿进行输入输出完全无关，只是从文件0读字符串给文件1写入而已。换言之，在前面SORT命令例上，对应文件0和文件1不是控制台而是分配好"INFILE"和"OUTFILE"就行了。用BASIC表示时，把标准状态下

```
OPEN "CON" FOR INPUT    AS #0
OPEN "CON" FOR OUTPUT AS #1
```

再定向标准输入输出时，如下设置即可。

```
OPEN "INFILE" FOR INPUT    AS #0
OPEN "OUTFILE" FOR OUTPUT AS #1
```

这样设置之后，使系统调用程序装载和执行（功能4BH），并令SORT命令执行。此时在装载一方（该情况下为COMMAND.COM）被打开的文件和环境全被SORT命令一方继承。因此，SORT命令一方完全不办理文件的打开手续就可直接开始文件内容的读写。另外，关闭的手续也不需要。命令执行后，文件在"COMMAND.COM"一方被关闭。

2.3 管道

本节是再定向概念的延伸，叙述有关管道功能的处理。

首先，为确认管道的基本动作输入如图2.4所示命令，看其结果。

此时第二个DIR不接受第一个DIR的执行结果。即完全不管第一个DIR的结果。这样的使用方法，所谓使用管道，只不过让其连续执行两个命令。

且看图2.4的结果输出，其上存有文件"% PIPE1.$$$"及"% PIPE2.$$$"。然而，没感到编制过这种文件。再一次用DIR命令看看目录（图2.5）

不明确，因此再一次输入和前面相同的命令看（图2.6）。