

IBM 370/148 OS/VS1 系统

使 用 简 介

上海市计算技术研究所

第一部分 OS/VSI 系统的简单介绍

一、OS/VSI

它是 Operating-System/Virtual Storage 的缩写。它对 IBM-370 计算机系统提供中间一级的支持。这一支持比得上在不可浮动环境里的 IBM-360 操作系统 MFT。VSI 是在 MFT 基础上作了一些改进，使之成为一个更有效和可广泛使用的操作系统。这些改进里很有意义的两项是虚存和作业输入子系统。

二、两项有意义的改进

1. 虚存

① 虚存向用户提供的最大存贮量是 16,777,216 字节，在它上面用户可以运行自己的计算系统。虚存设在辅助（外）存贮器 DASD 上，2048 字节为一单元，称为页面。

② 虚存的优点

对用户来说，虚存的最大优点是，用户可以在有较小实存的机器上调试大部分有大的设计要求的程序。就是，一般来说，用户不必担心存贮量不够。

2. 作业输入子系统 JES

JES 是一些系统功能集中后的名称。

① JES 执行三个基本功能：

I. 它从输入设备上输入所有初始输入流，并把它们存贮在 DASD 上，其存贮方式将方便于系统或用户程序而后的处理。

II. 它又把系统和用户的输出，存贮到 DASD 上，当方便的时候，再真正把它们打印出或卡片穿孔出（输出）。

III. 如果系统资源是竞争的对象（如，分派缓冲区），JES 就以保证系统的最高效率为目标，来解决这一问题。

头两个功能称为 Spooling（假脱机）。

② 假脱机的优点：

I. 非公用设备, 一般是单元记录设备, 如果有足够的缓冲区可用, 就被充分使用。

II. 非公用设备, 只有在需要输入数据, 打印数据或穿孔出数据时, 才使用上。

三、几个术语

1. JOB, 作业, 是指任一外部特指的工作单元 (Unit of work)。一个用户程序为 JOB 卡所指出。

Task, 任务, 指由 CPU 执行的工作的任一单元, 且在任务控制块控制之下。

2. MFT, 任务数固定的多道程序设计

多道程序设计, 指任务的几个单元同时执行。

多道程序设计的意义是, 它提供增加吞吐量和资源的较好的利用。

任务数固定, 指系统能同时执行的任务的最大个数, 在系统生成时已决定。

3. 同时运行

在一多道程序设计系统里, 任务是同时执行的。执行, 不是一齐执行, 或重迭执行, 或固定部分交替执行。决定哪个任务获得 CPU 的控制权, 是基于“等待”和“信件”。等待一事件发生, 把任务从竞争移入控制。任务的信件, 标志着等待的事件已完成了, 使任务处于“就绪”状态。任务成为活动, 就是最高优先级的任务已就绪了。该任务一直继续进行到发生另一事件, 使它放弃控制权。

4. 任务转移

一个任务放弃控制权, 另一任务获得控制权, 称为任务转移。

5. 分区, partition

① 虚存被分成页式区和非页式区, 再把页式区的一部分进一步分成一些不连续的区, 称为分区。定义的分区的数目, 决定了能同时执行的用户任务的数目。

② 用户程序的分区最多 15 个，余下的是系统任务的分区，37 个，共 52 个。

③ 除非系统里有分时间片或动态调度的功能，否则每个分区在系统里有固定的优先级。分区 0 优先级最高，分区 51 优先级最低。

④ 当一等待情况发生时，优先级决定哪个分区首先获得 CPU 控制权。每个作业的优先级由它所驻留的分区所决定。作业通过作业卡的 Class 参数，被指派到某一分区。

四、兼容性

VS1 向上兼容到 VS2。这些兼容包括：源程序编码，目标程序编码，作业控制语言，常规和标准。

五、VS1 控制程序

它有四个主要功能：

作业管理，是程序员、操作员与控制程序间通讯的处理。

任务管理，监督和控制整个操作系统。

数据管理，控制所有与 I/O 设备有关的操作。

复执管理，是 VS1 提供的一种服务，它试图克服因处理机、通道或 I/O 设备的故障造成的事故的影响。

第二部分 JCL 作业控制语言

一、简介

1. 你可以用任何语言写程序，操作系统把它翻译成机器语言并执行之，来完成你的工作。你必须用JCL说明你的作业(JOB)，才能把它们送入系统工作。

一个作业可以包含一个或多个作业步。每个作业步是一个处理程序或一个编目过程，及其有关的数据组成的工作组合体。用EXEC语句确定每一个作业步，每一个作业步用到的数据用DD语句定义，作业本身用JOB语句定义。这三句再加上其他六句：

/* 定界符语句

// 空语句

PROC 过程语句

PEND 过程结束语句

/* 注解语句

Command 命令语句

共九句，总称JCL。JOB语句，EXEC语句，DD语句和//语句是必须的。

2. 定义控制语句参数用的记号

① 大写字母和字——在控制语句里书写它们时，按照它们在格式里描述的形式，照实书写。下述字符也按实书写：

&

"

,

=

()

.

② 小写字母、字和记号（除上述“1”中外）——表示变量，当写上这个参数时，必须填入指定的信息。

③ { }——是一特殊记号，不要写在控制语句里。它用来把有关项集中在一起，并指出你必须书写其中的一项。

④ []——是一特殊记号，不要写在控制语句里。它指出，包括在里面的项是任选的，你可以写其中一项或不写。

⑤ ... ——是一特殊记号，不要写在控制语句里。它指出，前面的项可连续写多次。

3. JCL 所采用的字符集。

数字 0—9；

字母 A—Z；

固定符号：@，\$，#

特殊符号：，

：

／

‘

(

)

*

&

+

—

=

≡

4. 控制语句的四个域

① 名字域——标识语句本身，使其他语句和系统控制块可引证它。它由 1—8 个字母数字和固定字符组成，第一个字符必须是字母或固定字符，必须从第 3 列开始写。

② 操作符域——指出控制语句的类型，当是 Command 语句时，就是命令。操作符域必须跟在名字域后，前后各至少有一个空格。

③ 操作数域——包含用 ‘，’ 分开的参数。它必须紧跟在操作

符域之后，前后至少各有一个空格。

操作数域里有两种类型的参数：

定位参数：由它在操作数域里的位置和与其它参数的关系来刻划。

关键字参数：由一关键字和等号及变量信息来刻划。它必须跟在定位参数后面。

④ 注解域——任何信息，是由编写这控制语句的人认为有用的任何信息。它必须紧跟在操作数域后面（但在 Comment 语句里例外），前后至少各有一个空格。

5. 如何写控制语句的续行

JCL 控制语句太长，超过第 71 列时，你可以写到下一行去。

命令语句、注解语句、定界符语句、空语句不能续行。

继续操作数域：

① 在第 72 列之前，且在一完整的参数或子参数后，包括一个“，”，中断它。

② 在下一行的第 1 和 2 列上写上//，第 3 列空格，从第 4 列到第 16 列中任一列开始写续行内容，但不能从第 17 列开始。

6. 返回参照

你能使用 JCL 中的返回参照，使系统去参照作业中较前的 DD 语句里包含的有关信息。形式如下：

① parameter = * . ddname

② parameter = * . Stepname . ddname

③ parameter = * . Stepname . procstepname
.ddname

三、JOB 语句

标志着作业的开始，当作业被堆积在输入流中时，它标志前面作业的控制语句的结束。

1. 语句形式：

//Jobname JOB Operands Comments

2. 书写规则：

① 对每一个作业必须书写JOB语句，对每一JOB语句，必须写唯一的JOB名。jobname必须包含1到8个字母数字和固定字符；第一个字符必须是字母或固定字符。

② 下面的名字或字符不能作为jobname，除非你用括号括起来：A, CONSOLES, N, P, PFK, Q, R, RT, SQA, T, TP, U, USER.

③ 定位参数的次序是：

accounting information	财务信息
programmer'	程序员名

是否必要，由系统生成而定。

④ 关键字参数有13个：ADDRSPC, CLASS, COND, MPROFILE, MSGLEVEL, MSGCLASS, PROFILE, PRTY, PD, REGION, RESTART, TIME, TYPRUN.

可任意次序。

3. 下面介绍几个关键字参数：

① CLASS=jobclass

- 给每个作业指定一个作业级。
- 可以写任意字母A—Z或数字0—9，依系统生成时而定。
- 如果不写class的值，内定值一律为A，或依系统生成时而定。

② MSGCLASS=Output Class

- 指定系统信息输出的级别。
- 可写A—Z或0—9中任一字符，如果不写，或者是A级，或者依系统生成时而定。

· 系统信息和输出数据集的级都可指定为A级。即JOB语句中MSGCLASS=A，及DD语句中SYSOUT=A。

③ MSGLEVEL=(statements, messages)=(I, J).

指出输出清单中哪部分是作业的哪种输出。

“I”，指出输出何种作业控制语句。

$$I = \begin{cases} 0, & \text{只输出 JOB 语句。} \\ 1, & \text{输出全部输入的作业控制语句, 编目过程语句, 和过程语句参数的符号参数被代入以后的内部表示。} \\ 2, & \text{只输出输入的作业控制语句。} \end{cases}$$

“J”，指出输出何种存贮分配/终止信息。

$$J = \begin{cases} 0, & \text{不输出任何存贮分配/终止信息, 除非作业非正常结束。} \\ 1, & \text{输出所有存贮分配/终止信息。} \end{cases}$$

$$(4) \text{ TIME} = \begin{cases} (\text{minutes}), (\text{Seconds}) \\ 1440 \end{cases}$$

· 指出该作业使用 CPU 的最大时间值。该作业使用的 CPU 时间将列出在输出清单中。

· minutes 指出作业能使用 CPU 的最大分值，必须小于 1440。

· seconds 指出除 minutes 以外，作业能使用的 CPU 的最大秒值，必须小于 60。

· 1440 表示没有时间限制。

· 如果不写时间参数 TIME，一般系统自动提供 30 分钟的时间限制值，达到该值，作业被终止。

三、EXEC 语句

它是每个作业步和编目过程步的第一个语句，它标识执行的程序名或调用的编目过程名。

1. 语句形式

//Stepname EXEC Operands Comments

2. 书写规则

① 每个作业步必须有一个 EXEC 语句。

② 一般一个作业的作业步或过程步不能多于 255 个。

③ Stepname 是任选的，它是由 1 到 8 个字母数字和固定符

组成，第一个字符必须是字母或固定字符。

④ 定位参数的次序是：PGM

PROC.

⑤ 关键字参数是：ACCT, ADDRSPC, COND, PARM, RD, REGION, TIME.

3. 下面介绍几个参数：

① $PGM = \begin{cases} \text{Program name} \\ *. \text{Stepname. ddname} \\ *. \text{Stepname. procstepname. ddname} \end{cases}$

· 指出要执行的一个程序，这个程序必须是一个临时库，系统库或一专用库的一段 (member)。

· program name 是由 1—8 个字母、数字、固定字符组成，第一个字符必须是字母或固定字符。

② PROC 参数：

$\begin{cases} PROC = \text{procedure name} \\ \text{Procedure name} \end{cases}$

· 指出要调用并执行一个编目过程或随流过程。

· procedure name 必须由 1—8 个字母、数字或固定字符组成，第一个字符必须是字母或固定字符。

· 如果你在 EXEC 语句里，PROC 作为第一个参数，那么，PROC 可省写。

③ PARM 参数

PARM = Value

传送变量信息给执行时的程序。Value 可以有多至 100 个字符的信息。

④ TIME 参数

$TIME = \begin{cases} [\text{minutes}] [, \text{seconds}] \\ 1440 \end{cases}$

它的规则 and 限制与 JOB 语句中差不多，只是这里只限于本作业

步的 CPU 执行时间。

四、DD 语句

它描述了在作业步中用到的数据集和指出它在输入输出方面的要求。

1. 语句形式

//ddname DD Operands Comments

2. 书写规则

- ① 一个作业步里每一数据集都要有一个 DD 语句。
- ② ddname 从第 3 列开始写，允许有 1—8 个字母，数字或固定字符组成，第一个字符必须是字母或固定字符。
- ③ 一个作业步里的 ddname 必须唯一。如果一步里有重名，所有的参照（引证）都指向第一个有该 ddname 的语句。
- ④ 下面七个专用 ddname 请不要使用：JOB CAT（只对 VSAM），JOB LIB，STEP CAT（只对 VSAM），STEP LIB，SYS ABEND，SYS DUMP，SYS CHK。
- ⑤ 每一个作业步允许的 DD 语句最多是 255 个。
- ⑥ 定位参数是：*

DATA

DUMMY

- ⑦ 关键字参数是：AFF，AMP（只对 VSAM），BURST，CHARS，CHKPT，COMPACT，COPIES，DCB，DDNAME，DEST，DISP，DLM，DSID，DSNAME，FCB，FLASH，HOLD，LABEL，MODIFY，MSVGP，OUTLIM，QNAME，SEP，SPACE，SPLIP，SUBALLOC，SUBSYS，SYSOUT，TERM，UCS，UNIT，VOLUME。

3. 下面介绍几个参数：

① JOBLIB Facility

JOBLIB DD 语句由系统定义一个专用库，使对整个作业有

效。

- 必须紧跟在JOB 语句之后，作为第一个DD 语句。
- 当你写上JOBLIB DD 语句后，每当作业要使用一程序时，系统先搜索专用库，如果没有，它再搜索系统库。

② STEPLIB Facility

· STEPLIB DD 语句由系统定义一专用库，使对整个作业步有效。

- 它可以在作业步的DD 语句中的任意位置出现。

③ “*” 参数 //ddname DD *

· 指出数据跟在这DD 语句后面，这时，系统检验读入设备上的输入结束符（/* 或// 或EOF）。

- 每一个作业步可以有一个以上的DD * 语句。

· 在DD * 语句里，允许的關鍵字是DLM, DCB=BLKSIZE, DCB=BUFNO, DCB=LRECL, VOLUME, DSIID.

· 当你的数据中包含的语句以“//”开始时，用DATA 代替*，即DD DATA。

④ DATA 参数, //ddname DD DATA

参阅前面“③”。

⑤ DUMMY 参数 //ddname DD DUMY

指出：

- 对数据集没有分配外存空间。
- 对数据集不执行任何配置处理（DISP 参数）。
- 当 SAM 和 QSAM 时，对数据集不执行任何输入输出操作。

(6) DCB 参数。

$$DeB = \left\{ \begin{array}{l} \text{(list of attributes)} \\ \left\{ \begin{array}{l} dsname \\ *.ddname \\ *.Stepname.ddname \\ *.Stepname.procstepname. \\ \quad \quad \quad ddname \end{array} \right\} \text{(,list of attributes)} \end{array} \right\}$$

· 在执行一作业时，用来完善一数据集的 DCB 的信息。

· “list of attributes” 就是 DCB 关键字描述数据集的子参数。它们按存取方法列出。因为存取方法不同，子参数及其内容都不同。现在讲几个经常碰到的：

1 BLKSIZE 子参数，BLKSIZE = 字节数。

它指定记录块的最大长度，以字节计。对定长不成块的记录，这子参数也指定记录长度。最大长度允许 32,760 字节。对磁带而言，最小长度是 18 字节。

它根据记录格式而变化（由 RECFM 子参数指定）。

11 DSORG 子参数，DSORG = 数据集的结构

它指出数据集的结构，例：

PS 物理上顺序型结构的数据集。

PO 分块型结构数据集。

111 RECFM 子参数

指出数据集里记录的格式和特征，它们必须完整地描述。

IV. LRECL 子参数，LRECL = 字节数

为固定长记录指出真正的或最大长度，为可变长记录指出最大长度。为一逻辑记录指出真正的或最大长度。以字节为单位，且根据记录格式（由 RECFM 子参数给出）而变化。

⑦ DISP 参数

DISP (	$\left[\begin{array}{l} \text{NEW} \\ \text{OLD} \\ \text{SHR} \\ \text{SHARE} \\ \text{MOD} \end{array} \right]$	$\left[\begin{array}{l} , \text{DELETE} \\ , \text{KEEP} \\ , \text{PASS} \\ , \text{CATLG} \\ , \text{UNCATLG} \\ , \end{array} \right]$	$\left[\begin{array}{l} , \text{DELETE} \\ , \text{KEEP} \\ , \text{CATLG} \\ , \text{UNCATLG} \end{array} \right]$	)
--------	--	--	--	---

它描述数据集对系统而言的状态。也指出，当处理它的作业或作业步终止以后，对数据集做些什么。你可以指出 DISP 参数的一种配

置，即当作业执行后正常终止和不正常终止时的配置。

(8) DSNNAME 参数

$$\left\{ \begin{array}{l} \text{DSNAME} \\ \text{DSN} \end{array} \right\} = \left\{ \begin{array}{l} \text{dsname} \\ \text{dsname(member name)} \\ \text{dsname(generation number)} \\ \text{dsname(area name)} \\ \&\& \text{ dsname} \\ \&\& \text{ dsname(member name)} \\ \&\& \text{ dsname(area name)} \\ *. \text{ ddname} \\ *. \text{ stepname.ddname} \\ *. \text{ stepname.procstepname.ddname} \end{array} \right\}$$

它为数据集指定一个名字。系统使用这个DSN把数据集在一台设备上定位。

&&是指临时数据集。

(9) LABEL 参数

$$\text{LABEL} = (\text{data set sequence number}) \left(\begin{array}{l} . \text{SL} \\ . \text{SUL} \\ . \text{AL} \\ . \text{AUL} \\ . \text{NSL} \\ . \text{NL} \\ . \text{BLP} \\ . \text{LTM} \end{array} \right) \left(\begin{array}{l} . \text{PASSWORD} \\ . \text{NOPWREAD} \\ . \end{array} \right)$$

$$\left(\begin{array}{l} . \text{IN} \\ . \text{OUT} \end{array} \right) \left(\begin{array}{l} . \text{EXPDT} = \text{yyddd} \\ . \text{RETFD} = \text{nnnn} \end{array} \right)$$

它：

- 描述一数据集的标记。

- 描述不是第一个数据集它的序号。
- 对数据集指定一个保留周期。
- 对数据集指定一个保护口令。
- 对用 BSAM 方法的数据集，废弃 OPEN 宏指令。

⑩ SPACE 参数

$$\left\{ \begin{array}{l} \text{SPACE} = \left(\begin{array}{l} \text{TRK} \\ \text{CYL} \\ \text{or klength} \end{array} \right) \left(\begin{array}{l} \text{primary quantity} \\ \text{secondary quantity} \end{array} \right) \\ \text{SPACE} = (\text{ABSTR}, (\text{primary quantity}, \text{address} \\ \quad \quad \quad \left(\begin{array}{l} \text{, directory} \\ \text{, index} \end{array} \right))) \\ \left(\begin{array}{l} \text{, directory} \\ \text{, index} \end{array} \right) \left(\begin{array}{l} \text{, RLSE} \\ \text{,} \end{array} \right) \left(\begin{array}{l} \text{, CONTIG} \\ \text{, MXIG} \\ \text{, ALX} \end{array} \right) \left(\begin{array}{l} \text{, ROUND} \end{array} \right) \end{array} \right\}$$

指出，对一新的数据集，在直接存取设备上要分配多少空间。

⑪ UNIT 参数

$$\left\{ \begin{array}{l} \text{UNIT} = \left(\begin{array}{l} \text{unit address} \\ \text{device type} \\ \text{group name} \end{array} \right) \left(\begin{array}{l} \text{, unit count} \\ \text{, P} \\ \text{,} \end{array} \right) \left(\begin{array}{l} \text{, DEFER} \end{array} \right) \\ \text{UNIT} = \text{AFF} = \text{ddname} \\ \left(\begin{array}{l} \text{, SEP} = (\text{ddname}, \dots) \end{array} \right) \end{array} \right\}$$

指出，你为一数据集指定何利类型设备和多少该种类型的设备。

(12) VOLUME 参数

$$\left\{ \begin{array}{l} \text{VOLUME} \\ \text{VOL} \end{array} \right\} = ((\text{PRIVATE}) \left[\begin{array}{l} , \text{RETAIN} \\ , \\ , \text{volume sequence number} \\ , \\ , \text{volume count} \end{array} \right] \left[\begin{array}{l} , \text{SER} = (\text{serialnumber}, \dots) \\ , \text{REF} = \text{dsname} \\ , \text{REF} = * . \text{ddname} \\ , \text{REF} = * . \text{stepname} . \text{ddname} \\ , \text{REF} = * . \text{stepname} \\ \quad . \text{procstepname} . \text{ddname} \end{array} \right] \right)$$

验证数据集驻留的或将要驻留的设备的卷号。

第三部分 连接、编辑程序简介

一、目标模块和装入模块

任何课题的源程序输入后经编或编译，下面必要的一步是连接编辑处理。连接编辑是一个处理程序和服务程序，与语言翻译程序一起使用。

每一个课题的源程序是为满足某种目的而设计的。为了达到这目的，这程序一般可划分成几个逻辑单元，每单元执行某特定的功能，一个编码的逻辑单元，执行一个或几个相关功能，称为模块 (module) 或功能块。一般讲，不同的功能应该编成不同的模块，称为模块化程序设计。每个模块可用最适合于那种功能的符号语言写。这些符号语言有汇编，ALGOL，COBOL，FORTRAN，PL/I，RPG 等。

每个模块各自被一种语言翻译程序所汇编或编译。对语言翻译程序的输入称为源模块；从语言翻译程序的输出称为目标模块。在目标模块能执行以前，必须经过连接编辑程序的处理。连接编辑的输出称为装入模块。

目标模块是浮动格式，不可执行的机器代码。装入模块也是浮动的，但是是可执行的机器代码。

目标模块和装入模块有相同的逻辑结构。

二、连接编辑处理程序

1. 连接编辑程序的输入

(1) 初级输入或原始输入，它里面只有目标模块和连接编辑的控制语句。

(2) 外加的用户指定的输入，它里面可以有目标模块和控制语句，或者就是装入模块，这种输入可以由用户指定，也可由连接编辑程序自动去调用一个库后，再合并在一起而成。在处理过程中，连接编辑程序产生中间数据。