GC27-0052-0
File No. S370-36

Systems

OS/VS2
HASP II Version 4
User's Guide

Program Number 370H-TX-001

VS2 SVS Release 1.7

IBM

First Edition (September 1976)

This is the first edition of a new publication that applies to HASP II Version 4.1 in support of OS/VS2 SVS Release 1.7.

Information in this publication is subject to significant change. Any such changes will be published in new editions or technical newsletters. Before using the publication, consult the *IBM System/370 Bibliography*, GC-20-0001, and the technical newsletters that amend the bibliography, to learn which editions and technical newsletters are applicable and current.

Requests for copies of IBM publications should be made to the IBM branch office that serves you.

Forms for reader's comments are provided at the back of the publication. If the forms have been removed, comments may be addressed to IBM Corporation, P. O. Box 50020, Programming Publishing, San Jose, California 95150. All comments and suggestions become the property of IBM.

PREFACE

HASP is an optional program not required for the operation of an OS/VS2 SVS system. This book is intended for programmers at installations that have chosen to install HASP.

This book is divided into four sections:

- Routing a job through the system with HASP
- Obtaining output with HASP
- Coding JCL in the HASP environment
- Coding HASP control statements

Note: The external characteristics of HASP can be greatly influenced by the parameters used at HASPGEN. This book does not describe all of those options. Also, HASP is often modified by an installation to provide different external characteristics. Thus, the descriptions herein should not be regarded as typical. If your installation's system programming staff has made changes to HASP, be sure you understand the effects of those changes.

Related Publications

The following publications are referred to in this book:

- *OS/VS JCL Reference,* GC28-0618 (obtained by ordering GT28-0618)
- *OS/VS2 HASP II Version 4 Operator's Guide,* GC27-6993
- *OS/VS2 HASP II Version 4 System Programmer's Guide,* GC27-6992

In addition, HASP users that print output on an IBM 3800 should be familiar with the contents of the *IBM 3800 Printing Subsystem Programmer's Guide for OS/VS2 SVS,* GC26-3859.

CONTENTS

ROUTING A JOB THROUGH THE SYSTEM WITH HASP

The operating system interprets JCL statements to determine the resource requirements of jobs and job steps. HASP reads a job into the system, satisfies the requirements requested on JCL and HASP statements, schedules the job, and selects it for execution. HASP automatically performs most of these services for you, but you can code JCL and HASP parameters to influence how these services are performed. For example, HASP schedules a job for execution, but you can influence when the job is selected by coding the HASP PRIORITY control statement and the CLASS parameter of the JOB statement.

This section contains four topics:

- Job Scheduling
- Passing Information to the Job in Execution
- Delaying Job Initiation
- Direct submission of jobs to HASP from a program

Job Scheduling

HASP controls the selection of jobs for processing. As HASP reads a job into the system, JCL statements and any input stream data are placed on respective logical data sets. The HASP statements are checked for syntax errors and appropriate error messages are issued. If the HASP statements are syntactically correct, the job is placed on an execution queue. The execution queue is divided into job class queues and, within each job class queue, jobs are placed according to their priority. Jobs in the same class with the same priority are placed on the execution queue in the order they were read into the system. A HASP initiator is assigned job classes to process. It selects jobs from the first class assigned to it according to the priority of the jobs until no more jobs exist in that class, and then selects jobs from the next class assigned. You can influence how a job is placed in the execution queue—therefore, when it is selected for execution—by assigning a job class and priority to the job. Do this by coding the CLASS parameter on the JOB statement and the HASP PRIORITY control statement.

To insure that one job is selected before another or that the desired volumes are mounted before a job is executed, delay the job's selection by coding TYPRUN=HOLD on the JOB statement, by coding a job class that will force TYPRUN=HOLD, or by coding a SETUP control statement.

Assigning a Job to a Job Class

Job classes are established by an installation to group jobs. By assigning jobs to job classes, the installation tries to avoid contention between jobs that require the same resources by preventing them from running concurrently and tries to provide a better mix of jobs for more efficient system use. The installation determines which characteristics are most important in achieving a good balance of jobs in the computing system. Assign a job to a job class by coding the CLASS parameter on the JOB statement.

Assigning a Priority to a Job

Within a job class, jobs are selected for execution from the execution queue according to job priority. Jobs with the same class and priority are placed in the execution queue in a first in/first out order. In most cases, HASP will calculate the job's priority. However, for certain jobs, you or the operator can be instructed to assign different priorities. Specify job priority by coding a HASP PRIORITY statement.

Priority is explicitly stated on a PRIORITY statement and used by HASP. The estimated number of cards, lines of output, and the time for job execution are used according to installation algorithms to calculate the priority, and are also used by HASP to monitor job execution time and output. If these estimates are not stated, installation defaults are assumed. If any of these estimates are exceeded, the operator is notified. In some cases, the installation can specify that the job be canceled. For example, an installation might specify that the lower the estimated execution time and output, the higher the priority. This can enforce correct amounts to be specified or the job will be canceled.

Assigning a Dispatching Priority to Job Steps

In most jobs, you will want the job's dispatching priority to default to an automatic priority group (APG) instead of assigning your own dispatching priority. The automatic priority group function is an algorithm that the system will use to attempt to increase system throughput by dynamically adjusting the dispatching priority of associated tasks.

If you do assign a dispatching priority, code the DPRTY parameter on the EXEC statement. In the DPRTY parameter, you can code two values. The system substitutes these values in the following formula to form the dispatching priority:

(value1 x 16) + value2 = step's dispatching priority

If value1 is not coded, the system assumes the APG value for the default. (Value2 has a default of 11.) If you omit the DPRTY parameter completely or if DPRTY is equal to the APG value, the step has the same dispatching priority as the APG value.

Passing Information to the Job in Execution

You can include data in the input stream for a job using the HASP support of SYSIN data sets. When the job enters HASP, the data is preceded by a DD statement containing the "*" or "DATA" parameter. If no SYSIN-type DD statement is present, HASP generates a DD * statement with a ddname of SYSIN in accordance with the rules for inclusion of data in a VS JCL stream (see the *OS/VS JCL Reference*). The HASP restrictions on the use of the SYSIN JCL facilities are in the section "Coding JCL in the HASP Environment."

The HASP mechanisms for providing SYSIN data to programs in execution differ considerably from those employed by the operating system without HASP. When the job is read into the system, HASP spools the data onto its own private direct-access space. There is no VS direct-access data set representing the input data, as would be the case with the operating system support of SYSIN data. This technique represents a considerable reduction of overhead in the processing of SYSIN data. The input data is automatically blocked on the spooling devices without any explicit statement, such as BLKSIZE, on the DD statement.

To pass the data to the executing job, HASP employs the pseudo-device construct. While the job is being scheduled for execution, the DD statements representing the SYSIN data are modified to request the allocation of specific pseudo 2540 card readers. As the name "pseudo device" implies, these card readers do not actually exist outside the control block structure of the operating system. HASP intercepts the Read requests for SYSIN data via an exit from the I/O supervisor. The HASP program does actual I/O to its spooling devices when it is required to satisfy a job's input request. When the I/O is complete, the data is moved directly from the HASP buffers to the user's input areas. There is nothing in the JCL that causes the programmer to be aware of the pseudo-device construct.

The job in execution can be aware of the device types of the pseudo devices. In addition to the standard support through BSAM and QSAM, the job can read SYSIN data by invoking the I/O supervisor directly using execute channel program (EXCP). This is not the recommended way to communicate to a HASP pseudo-device. It is recommended that BSAM or QSAM be used.

To the program reading the SYSIN data, the data set appears as a directly-allocated card reader. The VS SYSIN data set appears as a direct-access data set. This difference of appearance leads to some restrictions to HASP data sets which do not apply to VS SYSIN data sets. Backspacing a card reader is traditionally a more difficult problem than backspacing within a direct-access data set. The program cannot do anything with this pseudo card reader that it could not do to a directly-allocated real card reader. (In fact, there are some things it could do to a real card reader that HASP does not support. See the *OS/VS2 HASP II Version 4 System Programmer's Guide* for a summary of these restrictions.)

Because the facilities of VS SYSIN data sets offer some different features, HASP allows the selection (via HASPGEN) of an option that makes VS SYSIN data sets available to jobs scheduled with HASP. If the "OS input option" is selected, any DD statement that contains the "*" or "DATA" parameter and the DCB parameter, is passed with its associated data to the VS reader interpreter. The DD statement is not modified by HASP. It results in a VS SYSIN data set just as it would if HASP were not in the system. Consult your system programmer to see if this option has been selected by your installation.

Delaying Job Initiation

Although you can influence a job's selection by assigning a job class and priority to the job, you cannot predict whether a job in one job class queue will be selected for execution before another job in a different job class queue. When jobs exist in the same job class queue, you cannot be certain that one job will complete execution before the other job is selected, even if you assign a higher priority to the first job. In some cases, when submitting two jobs, JOBA and JOBB, JOBA must complete execution before JOBB is initiated–JOBA might create records that JOBB will use. JOBB's initiation will have to be delayed until JOBA completes execution. It is also possible that resources a job requires will not be available–in this case, you will want to delay the job's initiation until required resources are available. Delay a job's initiation by coding TYPRUN=HOLD on the JOB statement or by specifying a job class defined by the installation to force a TYPRUN=HOLD default. You can also delay a job's initiation and have specific volumes mounted before the job executes, by coding the SETUP control statement to notify the operator which volumes are required.

To delay a job's initiation, code SETUP, TYPRUN=HOLD, or job class; the job remains on the execution queue but cannot be selected for processing until the operator releases the job. When the operator releases the job, it is again eligible for selection according to class and priority.

If you code a SETUP control statement, you are able to notify the operator what volumes are to be retrieved from the library. The operator will mount the requested volumes and then should release the job.

Direct Submission of Jobs to HASP From a Program

A procedure known as the internal reader allows direct introduction of jobs to the HASP job queue from batch programs running under SVS. Passing jobs is accomplished by writing the JCL to a pseudo-2520 card punch. Standard SVS BSAM WRITE or QSAM PUT can be used.

The internal reader devices are allocated to programs with JCL specifying UNIT=INTRDR.

The internal reader facility must be generated by an installation at HASPGEN time, and only a limited number are available. Consult your installation staff before using this facility.

End-of-File Cards

The final job submitted in the internal reader must be followed by a statement containing /*EOF in columns 1-5.

Deleting the Current Job in the Internal Reader

The current job in the internal reader can be deleted by inclusion of a statement containing /*DEL in columns 1-5.

OBTAINING OUTPUT WITH HASP

By coding JCL statements, you can request output data sets, listings of JCL statements, system messages, and abnormal termination dumps. By coding JCL and HASP statements, you can request special forms processing, routing of output to specific devices, and multiple original printing by data sets within a job. The HASP statements have the same options as JCL with some additional features such as multiple destination and data set grouping.

This section includes these topics:

- Requesting Listings of JCL Statements and System Messages
- Writing Output Data Sets
- Controlling Output Destination

Requesting Listings of JCL Statements and System Messages

The system produces messages about a job concerning allocation of units and volumes, disposition of data sets, and termination of job steps and the job. You can request that these messages and/or the JCL statements from the job and from cataloged procedures called by the job be included on an output listing.

By coding the MSGLEVEL parameter on the JOB statement, you inform the system of what statements and messages you want included on the output listing.

By coding the MSGCLASS parameter on the JOB statement, you assign messages and JCL statements to an output class. A default is assigned if MSGCLASS is not coded.

Writing Output Data Sets

There are three ways to write output data sets:

- Assign the data set to a HASP output class.
- Assign the data set to an OS output class.
- Specify the device on which the output should be written.

When you assign a data set to a HASP output class, the data set is first written to the HASP spool device and then written to the final output device by HASP. When you assign a data set to an OS SYSOUT output class, it is first written on a temporary direct-access data set, and then written to the final output device by the SVS output writer. When you specify the device on the UNIT parameter, if the device is available, it is assigned to your job and under the control of your program.

Assigning Output Data Sets to Output Classes

Output classes include output with similar characteristics that are written to the same device. There are 36 possible output classes that can be coded on either the SYSOUT or MSGCLASS parameters. The letter and number names have no inherent meaning--each installation defines its own output classes. For example, output class W might contain low priority output; class X might be reserved for high-volume output. If you want the output data set and messages from the job to be printed on the same output listing, specify SYSOUT=$ or the same output class in the SYSOUT parameter as specified for messages in the MSGCLASS parameter.

The installation can designate certain classes to be processed by OS rather than by HASP. If one of these classes is selected, then the data set is processed as though HASP were not in the system. Also, certain classes can be defined as requeue classes. HASP processes the class by copying any system messages to be written by HASP, and then requeuing the data set to the class specified by the installation. This allows processing by an output writer or the TSO Output command.

Specifying the Device

To write an output data set without using the HASP SYSOUT service, code the UNIT parameter on the DD statement defining the unit record device on which the data set is written. The system will allocate the device exclusively to the job if the device is available: no other job can write output to that device until it is released. Jobs cannot share an output device as they can when output is assigned to output classes.

Data management routines write the output from the program to the device specified in the UNIT parameter. Specifying a particular output device in the UNIT parameter generally is not the most efficient method for obtaining system output.

Processing Output Classes

Using HASP is an efficient way to write output. HASP supports the use of local and remote printers and punches as devices on which output classes are written. For OS output classes, the output writer supports local printers, punches, and tape devices and user-written routines.

Output will be printed on the same listing if such parameters as CLASS, FORMS, FCB, UCS, FLASH, BURST, and DEST have similar characteristics for all data sets and a user-written writer is not specified. The installation may choose to put all data sets that specify the same output class as the MSGCLASS parameter out on the same listing, even though FORMS, UCS, FCB, and DEST are different. (This facility is not available if the 3800 printer is installed.)

Suppressing the Writing of an Output Data Set

Whether writing an output data set by coding the SYSOUT parameter or the UNIT parameter, you can suppress the writing of the data set by defining it as a dummy data set. This is useful when testing a program and you do not want data sets printed until you are sure they will contain meaningful output. Suppressing the writing of a data set saves processing time.

If you are routing an output data set by coding the SYSOUT parameter, code the DUMMY parameter to define the data set as a dummy data set. When DUMMY is coded, the SYSOUT parameter is ignored and the data set is not written.

Limiting Output Records

HASP ignores the OUTLIM parameter on SYSOUT DD statements for HASP output classes. HASP provides output limiting at the job level. The action taken when a job exceeds its estimated output lines is dependent on an option chosen at HASPGEN. Depending on the choice, either the operator is informed that the output limit has been exceeded, or HASP cancels the job. Consult your system programming staff to determine which option is used at your installation.

Requesting Page Overflow Processing

HASP will limit the number of lines printed per page (for example, to prevent printing over the edge of the form) if requested either by the installation during HASP generation or by the programmer coding LINECT on the JOBPARM statement. The installation-specified number of lines per page can be overridden by the JOBPARM LINECT parameter. Line limiting can be turned off by coding LINECT=0. Set the line count sufficiently large to prevent unwanted page ejects for output from programs that provide page eject carriage control parameters.
Note: The 3800 printer does not print in the ½-inch margin at the top and bottom of each page, regardless of the value of LINECT.

HASP Support of the 3211 Indexing Feature

You can specify that output that is printed by the HASP writer onto a 3211 Printer be indexed to the right by coding the INDEX parameter on the OUTPUT statement. This parameter is ignored if the output is sent to a device other than a 3211. Determine whether an output class has been set aside to designate output to be processed by a HASP writer onto a 3211 Printer by asking the installation's system programming staff.

Requesting Multiple Copies of an Output Data Set

You can control the number of hard copies produced by the SYSOUT data sets. As many as 255 copies of an output data set are obtained by coding the COPIES parameter on the SYSOUT DD statement defining the data set or on the HASP OUTPUT control statement. As many as 255 copies of the entire job-related output are obtained by coding the COPIES parameter on the HASP JOBPARM control statement.

If you request multiple copies of job-related output by coding the OUTPUT or SYSOUT DD statements and the JOBPARM control statement, HASP output processing gives the multiple of the requested amount for each SYSOUT data set. For example, if you request two copies of the entire job output (code COPIES=2 on the JOBPARM statement) and three copies of a certain output data set (code COPIES=3 on a SYSOUT DD statement or OUTPUT control statement), you will receive two copies of the entire job output but will receive a total of six copies of the SYSOUT data set.

With the 3800 printer, you can request grouping of copies by using the COPIES parameter (with *group values* specified) on the SYSOUT DD statement or the COPYG parameter on the OUTPUT statement.

Requesting a Special Output Form

Special forms are requested for printing output data sets by including the form name in the SYSOUT parameter on the DD statement defining the data set or on the OUTPUT control statement. For example, to assign a data set to an output class and specify the data set be printed on a special form, code SYSOUT=(A,,FMS2). Alternatively, you can code a /*OUTPUT F=FMS2 control statement. HASP ensures that the proper form is mounted.

The entire job can be printed on a special form by coding the FORMS parameter on the JOBPARM statement. If you code a forms name on either the SYSOUT or the OUTPUT statements, it overrides the forms name in the JOBPARM statement. The forms name coded on the OUTPUT statement overrides a form named using the SYSOUT parameter.

Requesting a Specific Character Set on UCS Printers

Universal character set (UCS) features are requested by coding the UCS parameter on a DD statement defining an output or SYSOUT data set or by coding UCS on the OUTPUT control statement for SYSOUT data sets. You can request UCS features for different sets of characters to be printed for various applications.

To request a specific character set for a 3211 or 1403 Printer, specify the name or HASP abbreviation identifying the character set in the UCS parameter or the OUTPUT statement. The names and HASP abbreviations for the IBM standard character sets are as follows:

1403	3211	HASP Abbreviation	Characteristics
AN	A11	A	Arrangement A, standard EBCDIC character set, 48 characters
HN	H11	H	Arrangement H, EBCDIC character set for FORTRAN and COBOL, 48 characters
PN	P11	P	PL/I alphameric character set
QN		Q	PL/I preferred alphameric character set for scientific applications
RN		R	Preferred character set for commercial applications of FORTRAN and COBOL

Where two values exist (for example, AN for the 1403 and A11 for the 3211 Printer), either can be coded and HASP will select the set corresponding to the device onto which the data set is printed.

Not all of these character sets may be available at your installation. In addition, the installation can design character sets to meet special needs and assign a unique name to them. See the system programming staff for a complete list of available character sets for the installation.

If the data set is routed to a 3800 printer and UCS is specified but CHARS is not, then the UCS id is used as the first character arrangement table name. If it is possible for the data set to be routed to a 3800, then the HASP abbreviations should not be used.

Requesting Character Arrangement Tables for the 3800

You can request up to four character arrangement tables for the 3800 Printing Subsystem by specifying their names with the CHARS parameter on your SYSOUT DD statement or the OUTPUT statement. If CHARS is specified and UCS is not specified and the data set is rerouted to a 1403 or 3211 with the UCS feature, then the first character arrangement table name specified is used as the UCS id.

Requesting a Specific FCB Image

Specific forms control buffer images (controlling the number of lines per page) for a 3211 or 3800 printer are requested by coding an image identifier in the FCB parameter in JCL or by coding FCB on the OUTPUT control statement. The FCB parameter can also specify a specific carriage control tape for the 1403 Printer for HASP output processing.

HASP provides two standard FCB images: 6 and 8. FCB 6 specifies that 6 lines per inch are to be printed on an 11-inch form. FCB 8 specifies that 8 lines per inch are to be printed on an 8½ inch form. Additional FCB images can be specified by the installation.

With the 3800 Printing Subsystem, three additional standard FCB images are supplied by IBM. They are STD1, STD2, and STD3. STD1 specifies that 6 lines per inch are to be printed on an 11-inch form; STD2 specifies that 8 lines per inch are to be printed on an 8½-inch form; and STD3 specifies that 8 lines per inch are to be printed on an 11-inch form. Do not use these FCB images unless instructed to do so by your system programming staff.

For FCB images used with the 3211, all images must be defined at HASPGEN. For the 3800, additional FCB images can be defined by using the IEBIMAGE utility program. If there is a possibility that your data set may be routed to a 3211 rather than a 3800, you should ensure that the 3211 has an FCB image equivalent to the one that you are requesting on the 3800.

Requesting Flashing of Forms on the 3800

A specific forms overlay for the 3800 printer can be requested by specifying the FLASH parameter on either the SYSOUT DD statement or the OUTPUT statement. HASP writes a message to the operator asking that the requested forms overlay be inserted into the 3800 prior to processing the data set. You can obtain a group of data sets with some flashed with the overlay and some not flashed (provided all other group factors are the same) by specifying the forms overlay name for each data set and explicitly coding a flash count of zero for those which are not to have the overlay.

Requesting Burst Output on the 3800

If the optional Burster-Trimmer-Stacker is installed on your 3800, you can request that your output be burst by coding BURST=Y on the SYSOUT DD statement or the OUTPUT statement. If no value is coded, then the installation default is used. If BURST is coded and the output is rerouted to a printer other than a 3800, the burst specification is ignored. If bursting is requested and no Burster-Trimmer-Stacker is installed on the 3800, the operator must explicitly set the printer to process output that requires bursting and must hold that output aside so that it can be burst in some other manner.

Requesting Copy Modification on the 3800

You can request copy modification on the 3800 by specifying the name of your copy modification module with the MODIFY parameter on either the SYSOUT DD statement or the OUTPUT statement. If MODIFY is coded and the data set is rerouted to a printer other than the 3800, then MODIFY is ignored.

Controlling Output Destination

HASP allows you to submit jobs to a central computing center from a work station and to route output to work stations.

The default output location is the submitting location, either a remote work station or the central site (destination of LOCAL). To receive the output at the submitting location, simply assign output data sets to any output class (with the SYSOUT parameter) and messages from your job to an output class (with the MSGCLASS parameter). HASP at remote stations offers most of the same options for writing data sets that are requested when submitting the job at the central computing center. You can request:

- That a data set be held until the operator requests that it be printed.
- That a special output form be used by specifying a form name in the SYSOUT parameter.
- That multiple copies of the data set be used.

Whether at a remote station or at the central computing center, you can also request that a data set be routed to another destination. To route an output data set to another destination, code the identification of that destination in the DEST parameter on the DD statement defining the data set or code DEST on the OUTPUT statement. If you code a destination on either the SYSOUT or the OUTPUT statements, it will override the destination in the ROUTE statement. Work stations are identified by a destination identification that has been established by the system programmer. The destination parameter will cause output to be routed to local printers or punches or to any remote station.

Example of Obtaining Output With HASP

This example shows the use of HASP and JCL statements that can be used to obtain output.

```
/*PRIORITY 5
//OUTJOB   JOB      BAKER,MSGCLASS=J
/*JOBPARM  COPIES=2,LINECT=20,ROOM=233,FORMS=GRN1
/*OUTPUT   PSET DEST=PRINTER8,FCB=STD3,FORMS=2PRT,UCS=AN,COPYG=2
/*SETUP    253194
//STEP1    EXEC     PGM=TESTSYSO
//DD1      DD       DSN=DATA,UNIT=2314,VOL=SER=SCHLIB,
//                  DISP=(OLD,KEEP),SPACE=(TRK,(5,2))
//DD2      DD       DSN=&TEMP,UNIT=2314,DISP=(NEW,DELETE),
//                  SPACE=(TRK,(10,5))
//DD3      DD       SYSOUT=(A,,PSET)
//DD4      DD       SYSOUT=(A,,GRPH)
//DD5      DD       SYSOUT=L
```

1. The job will be selected at priority level 5.
2. All system messages are to be written to output class J.
3. The JOBPARM statement indicates that:
   a. Two copies of the entire job-related output will be printed.
   b. No more than 20 lines per page will be printed (LINECT=20).
   c. The programmer's office number is 233. This appears on the separator page and is used for distributing output.
   d. Forms name GRN1 is the name of the form to be used by all data sets unless a specific form is defined on a DD statement.
4. The OUTPUT statement indicates that:
   a. PSET is the code that, when indicated on a DD statement, causes all parameters on the OUTPUT statement to override default parameters.
   b. The destination for the output is a printer and is number 8 if it is local print/punch routing; otherwise, PRINTER8 is equivalent to LOCAL.
   c. If the printer has the forms control buffer feature, STD3 must be the name of a member of SYS1.IMAGELIB. STD3 defines the special FCB image to be used for processing the job.
   d. Forms name 2PRT is the name of the forms for data sets coding PSET in the SYSOUT parameter.
   e. UCS=AN specifies printing with the AN character set on a 1403 printer, or with the equivalent A11 set on a 3211. If this job is rerouted to a 3800, since the CHARS parameter is not specified, UCS=AN is recognized as a request for an AN character arrangement table.
   f. If data sets with PSET coded in the SYSOUT parameter are printed on a 3800, COPYG=2 specifies that two copies be made. Each page of one copy is adjacent to the corresponding page of the other copy. This approximates the specification of 2-part paper for an impact printer.
5. The SETUP statement indicates that volume 253194 should be mounted before this job begins processing.
6. SYSOUT data sets and message class are printed on green (GRN1) paper except DD3 and DD4. The DD4 SYSOUT data set is printed on graph (GRPH) paper; the DD3 SYSOUT data set is on 2-part paper (not valid for a 3800 printer).

CODING JCL IN THE HASP ENVIRONMENT

HASP accomplishes many of its functions by appropriately modifying the JCL submitted by the user prior to passing the job to the operating system. These modifications occur either directly under HASP or in exit routines in the reader/interpreter. Because of these changes, it is possible that certain errors will be undetected or that there will be a delay in the determination of an error. (HASP does not give JCL to the SVS Reader/interpreter until the job is ready to execute.) Many of the features relating to HASP can be requested either through JCL or HASP control cards. Errors that can be detected in HASP control cards are determined when the job is read into the system with a minimum of delay.

The JOB Card

The JOB card is a variable-field control card that defines the beginning of a job (and, of course, the end of the previous job) within an input stream. The format of the JOB card is as described in the *OS/VS JCL Reference* manual. In addition to recognizing certain keyword entries, HASP may be required to extract information from the accounting field of the JOB card (refer to the *HASP System Programmer's Guide* section describing the &RJOBOPT HASPGEN parameter).

When HASP utilizes the accounting field, subfields of the accounting field are treated as positional parameters. Each subfield, in accordance with OS/VS JCL standards, can be enclosed in single quotes. To omit a specific subfield, the comma normally following the subfield should be punched in the subfield's place. To omit the remainder of the subfields, the closing right parenthesis should follow the last subfield entered.

HASP assumes the accounting field to be in the following format:

(pano,room,time,lines,cards,forms,copies,log,linect)

where:

pano = programmer's accounting number. This subfield should consist of one to four alphameric characters (example: "4301").

room = programmer's room number. This subfield should consist of from one to four alphameric characters (example: ",E305").

time = estimated execution time in minutes. This subfield should consist of up to four numeric digits (example: ",30" for 30 minutes). If omitted, a standard value is assumed.

lines = estimated line count in thousands of lines. This subfield should consist of up to four numeric digits (example: ",5" for 5000 lines). If omitted, a standard number of lines is assumed.

cards = estimated number of cards to be punched. This subfield should consist of up to four numeric digits (example: ",200" for 200 cards to be punched). If omitted, a standard number of cards is assumed.

forms = special forms for printing entire job. This subfield should consist of from one to four alphameric characters (example: ",5" for 5-part forms). If omitted, a standard form is assumed.

copies = number of times output is to be printed or punched. This subfield should consist of up to three numeric digits and should not exceed an installation-specified limit (maximum 255) (example: ",2" for two copies). If omitted, one copy is assumed.

log = HASP Job Log option. This subfield should consist of one character. If this character is an "N", the HASP Job Log will not be produced. If any other character, or if omitted, the log is produced.

linect = lines to be printed per page. This subfield should consist of up to three numeric digits (example: ",34" for 34 lines per page) and should not exceed the value 255. If coded as zero, no automatic overflow will be produced. If omitted, a standard value is assumed.

Note: If HASP has been generated to terminate a job having an accounting field subparameter that is illegal by HASP standards, then the first two subfields (that is, pano and room) are required.

The following would be a typical JOB card:

```
//ORBIT     JOB (7808,E305,,2,200),                    CONTINUED
//          'J. JACKSON',MSGLEVEL=1,CLASS=B
```

In this case:

pano = 7808

room = E305

time = 2 minutes (assumed value)

lines = 2000 lines

cards = 200 cards

forms = standard forms (assumed)

copies = 1 copy (assumed)

log = YES (assumed value)

linect = standard value (assumed)

The other fields on the JOB card are also interpreted for accounting purposes and job control. The JOB card may be continued in accordance with OS/VS JCL specifications.

SYSIN Data

SYSIN data can be included in the job according to the SVS JCL rules for DD * and DD DATA statements. Prior to execution of the program requiring the data, HASP modifies the JCL to specify a pseudo card reader. The program reads from this card reader exactly as though the SYSIN data were placed in a card reader directly allocated to the job.

The following restrictions apply, which do not apply to OS SYSIN data:

- The SYSIN DD statement cannot be continued, under any circumstances.
- The program cannot do anything that it could not do to a directly-allocated card reader. Additionally, HASP may not accurately simulate a card reader in the areas of timing, data chaining, and Input/Output appendages. (See *OS/VS2 HASP II Version 4 System Programmer's Guide* for additional information.)
- HASP support of the DLM parameter is compatible with SVS, with the following exceptions:
  - The DLM parameter must be the last parameter to appear on the DD statement.
  - The apostrophe (') cannot be used as a delimiter.

- HASP control cards (for example, /*MESSAGE, /*SETUP, /*ROUTE) are not recognized if the DLM specification is other than "/*".
- The DLM parameter does not override Internal Reader control cards (/*EOF or /*DEL) or remote device control cards (for example, /*SIGNON). The control card is *not* processed as in-stream data.

SYSOUT Data

The SYSOUT parameter is specified on the SYSOUT DD statement in the following format (see the section "Obtaining Output with HASP" for additional information on specifying output):

SYSOUT = (*class name* [, *program name*] [{, *code id* / , *form name*}])

class name is an alphameric character (A-Z, 0-9, or $). If class=$ then the message class is used. Certain classes can be defined by an installation as "OS classes." If one of these classes is specified, then the data set is treated as an SVS SYSOUT data set and HASP does no processing of that JCL statement. Consult your installation staff to find out if these classes exist at your installation.

program name specifies a user-written writer to process the data set. HASP does not support user-written writers directly. If a *program name* is coded, then HASP does not process the SYSOUT data; it is assumed to be an OS SYSOUT data set. Do not specify this field if you want HASP to process your output.

code id is a 1- to 4-alphameric character string that specifies a corresponding *id* on an OUTPUT control statement from which the data set characteristics are obtained.

form name If no corresponding OUTPUT control *id* is found, then this positional parameter is assumed to be the form name.

Other Parameters on the SYSOUT DD Statement

HASP recognizes and processes the following DD statement keyword parameters requesting HASP SYSOUT services (other parameters are ignored):

BURST	**FCB**
CHARS	**FLASH**
COPIES	**MODIFY**
DCB	**UCS**

HASP is in general more forgiving of JCL errors than the operating system. Hence in many cases where OS would cause the job to fail with a JCL error, HASP assumes some meaning for the parameter in error and proceeds. In general, if a parameter is too long, HASP uses the first n characters (where n is the maximum number of meaningful characters for the parameter). If excessive subparameters are specified for a keyword parameter, the excess is ignored.

HASP converts the SYSOUT DD statements to requests for HASP pseudo units during interpretation of the JCL. Most parameters are processed by HASP directly and are not available to either the system or the problem program during job execution. Certain information is passed to the system on the request for the pseudo device. This information includes CHARS, FCB, and most DCB subparameters. If you code a JFCBE exit routine (for a 3800) the CHARS and FCB information is available, but any modifications to the JFCBE are ignored by HASP.

Other differences between the JCL with and without HASP are noted below:

BURST = {Y|N} If neither **Y** nor **N** is specified, then the installation default burst setting is chosen.

CHARS HASP stops processing **CHARS** at the first null character arrangement table name; if more are specified, they are ignored.

COPIES If any subparameter of **COPIES** is greater than 255, then it is replaced by zero. If a non-numeric is encountered in a subparameter, it is replaced by zero. If the total of copy groups is greater than 255, then the copy group causing the value to exceed 255 is reduced such that the total is 255. Processing of the **COPIES** parameter is terminated when the first zero copy group is encountered.

FCB For 3211 Printers, the **FCB** specification refers to FCBs that are defined to the HASP program at HASPGEN time. For the 3800 Printing Subsystem, the **FCB** can be either one defined at HASPGEN and included in the HASP program or an **FCB** in SYS1.IMAGELIB. HASP looks for included 3800 FCBs before searching SYS1.IMAGELIB. The **ALIGN** and **VERIFY** subparameters are ignored.

FLASH If the flash count is greater than 255 or contains a non-numeric character, then it is replaced by zero. A flash count specified as zero still requires the specified forms overlay to be inserted in the printer, but no flashing is done. This allows you to have all your data sets print together when some require flashing and some do not. If the flash count is unspecified, then all copies are flashed. If a flash count is specified without an overlay name, it is ignored.

MODIFY Specification of a table reference character (*trc*) that is non-numeric or greater than 3 causes unpredictable results. If a *trc* is specified without a copy modification module name, it is ignored.

UCS For 1403 and 3211 Printers with the UCS feature, the **UCS** *id* refers to **UCS** *id's* internally generated during HASPGEN.

CODING HASP CONTROL STATEMENTS

The HASP control statements are coded with JCL statements to control the input and output processing of jobs. Rules for coding JCL, including syntax, apply to the HASP control statements. However, there are additional rules for coding HASP statements. They are:

- When the description of a control statement specifies the column in which a field must appear, it means *exactly* that column. Most errors in coding control statements occur because a field begins in the wrong column.
- Columns 1 and 2 always contain the characters /*.
- HASP statements cannot be continued. You can use multiple control statements if more than one statement is needed.
- Do not place HASP control statements in a cataloged procedure; they are ignored.
- If you code more than one statement with the same parameters, the last statement coded overrides any other statements.
- If you code more than one of the same parameters on the same statement, the last parameter coded overrides any other parameters.
- You can code the HASP control statements in any order. However, the COMMAND and the PRIORITY statements must be placed in front of the JOB statement and all other HASP statements should follow the JOB statement.
- The JOBPARM statement overrides the installation default but can itself be overridden by a specific output statement.

The Command Statement

The command statement specifies HASP operator commands that can be entered through the card reader or the system console. Examples in this book illustrate the format for commands entered through the card reader. Commands entered through the system console should omit the /* from the message.

For a detailed description of the command statement and the names of the correct HASP verbs and operands, *see OS/VS2 HASP II Version 4 Operator's Guide.* The command statement consists of the characters/* in columns 1 and 2. Column 3 contains a character either established at HASP generation by the installation or defaults to '$'. There are two fields–a HASP command verb starting in column 4 followed by one or more operands. An "N" may be coded in column 72. Columns 73-80 are ignored.

```
/*$command verb operand[,operand...] [N]
```

command verb
an operand indicating which HASP operator command is to be performed.

operand
one or more variable-length operands.

The following HASP commands can be entered through the input stream.

$A	$E	$L	$R	$Z
$B	$F	$N	$S	
$C	$H	$O	$T	
$D	$I	$P	$VS	

N
indicates that the command will not be repeated on the operator's console.

Rules for Coding

- Code as many command statements as are needed, but do not continue them from one statement to the next.
- Command statements must be placed before jobs being entered through the input stream. Any command statements within a job are ignored.
- Commands that are entered on the command statement are executed immediately. They cannot be linked with any execution process of a job.
- HASP commands entered through the input stream are of the form /*$command. The $ is a HASPGEN option.

Example of the Command Statement

```
/*$SI3-5
```

This command starts initiators three through five. The command is $S and the operand is I3-5.

The JOBPARM Statement

The JOBPARM statement specifies job-related parameters for HASP.

The JOBPARM statement consists of the characters /* in columns 1 and 2, the word JOBPARM in columns 3-9, a blank in column 10, and parameters in columns 11-71. Columns 72-80 are ignored.

For further information, see "Obtaining Output with HASP."

/*JOBPARM parameters

Code one or more of the following parameters in the longer form (full word) or the shorter form (one letter abbreviation).

{CARDS=*nnn* / C=*nnn*} {,COPIES=*nnn* / ,N=*nnn*} {,FORMS=*xxx* / ,F=*xxx*}

{,LINECT=*nnn* / ,K=*nnn*} {,LINES=*nnn* / ,L=*nnn*} {,NOLOG / ,J}

{,ROOM=*xxx* / ,R=*xxx*} {,TIME=*nnn* / ,T=*nnn*}

CARDS=*nnn*
a value estimating the number of output cards from this job (from 0 to 9999999 cards).

COPIES=*nnn*
a value indicating the number of printed output copies of job-related output that is to be produced (from 1 to 255 copies).

FORMS=*xxx*
an alphameric value indicating the print and punch forms for this job's output that are not further defined in this job (from 1 to 4 characters).

LINECT=*nnn*
a value showing the number of lines to put on each output page for HASP page overflow processing (from 0 to 255 lines).

LINES=*nnn*
a value estimating the number of output lines from this job—in thousands of lines (from 0 to 9,999).

NOLOG
a parameter meaning that you do not want the HASP job log as output. (The job log contains a list of job-related console messages and operator replies produced during processing of the job.)

ROOM=*nnn*
an alphameric value indicating a programmer's room number to be placed on the job's separator page for routing back to the programmer (from 1 to 4 characters).

TIME=*nnn*
a value estimating the job execution time in minutes of real time (from 0 to 279,620 minutes).

Rules for Coding

- Any JOBPARM statement parameter value supersedes the equivalent parameter value from the accounting field (in HASP format) of the JOB statement or from any preceding JOBPARM statement in this job's JCL. All of these statements override the default established by the installation.
- Any number of the above parameters can be placed on a single JOBPARM statement and as many JOBPARM statements as desired can be placed together with a given input stream. A JOBPARM statement cannot be continued.
- Place the JOBPARM statement after the JOB statement.
- If you code LINECT=0, HASP will not eject to a new page when the number of lines has exceeded the page limit that was established at HASP generation. (The 3800 printer, however, does not print in the ½-inch margin at the top and bottom of each page, regardless of the value of LINECT.)

Example of the JOBPARM Statement

```
/*JOBPARM L=60,R=4222,T=50
```

The three specifications mean the following:

L=60	The job's estimated printed output will be 60,000 lines.
R=4222	The programmer's room is 4222. This information will be placed in the separators for both printed and punched data sets.
T=50	The job's estimated execution time is 50 minutes.

The MESSAGE Statement

The MESSAGE statement permits you to send messages to the operator (via the operator console) at HASP job input time.

The MESSAGE statement consists of the characters /* in columns 1 and 2, the word MESSAGE in columns 3-9, a blank in columns 10 and 11, and the message in columns 12-71. Columns 72-80 are ignored.

```
/*MESSAGE  message to be written
```

Rules for Placement

- Place the MESSAGE statement after the JOB statement. This allows the job number to be appended to the beginning of the message.
- If the MESSAGE statement is not included within the boundaries of a job, the input device name is appended to the beginning of the message.

Example of the MESSAGE Statement

```
/*MESSAGE  CALL DEPT58 WHEN PAYROLL JOB IS FINISHED--EX.1946
```

This message requests that department 58 be called when the payroll job is complete.

The OUTPUT Statement

The OUTPUT statement specifies characteristics and/or options of a specific SYSOUT data set or groups of SYSOUT data sets.

For further information on the OUTPUT statement, see "Obtaining Output with HASP."

The OUTPUT statement consists of characters /* in columns 1 and 2, the word OUTPUT in columns 3-8, and a code beginning in column 10 followed by the keyword parameters. Columns 72-80 are ignored.

```
/*OUTPUT code parameters
```

Code one or more of the following parameters in the longer form (full word) or the shorter form (one letter abbreviation).

{BURST=*x* / B=*x*} {,CHARS=*xxxx* / ,X=*xxxx*} {,COPIES=*nnn* / ,N=*nnn*} {,COPYG=*nnn* / ,G=*nnn*}

{,DEST=*nnn* / ,D=*nnn*} {,FCB=*xxxx* / ,C=*xxxx*} {,FLASH=*xxxx* / ,O=*xxxx*} {,FLASHC=*nnn* / ,Q=*nnn*}

{,FORMS=*xxxx* / ,F=*xxxx*} {,INDEX=*nn* / ,I=*nn*} {,MODIFY=*xxxx* / ,Y=*xxxx*} {,MODTRC=*n* / ,M=*n*}

{,UCS=*xxxx* / ,T=*xxxx*}

code
alphameric characters referencing all SYSOUT data sets within your job whose code in the form number subparameter of the SYSOUT parameter matches the "code" specified on the OUTPUT statement (from 1 to 4 characters).

BURST=*x*
one character, either Y or N. Y specifies that the output from the 3800 printer is to be burst, and N specifies that the output is to be continuous fanfold.

CHARS=*xxxx*
alphameric values specifying 1 to 4 character arrangement table names for the 3800 printer. In order to specify more than one table name, code **CHARS=(*xxxx,xxxx*...)**. Each table name can be from 1 to 4 characters.

COPIES=*nnn*
a value indicating the number of copies of printed job-related output that is to be produced (from 1 to 255 copies).

COPYG=*nnn*
from 1 to 8 numbers specifying the grouping of copies on the 3800 printer. More than one copy group is obtained by coding **COPYG=(*nnn,nnn* . . .)**.

DEST=*nnn*
1 to 4 different destinations can be specified for each output data set. In order to specify more than one destination, code: DEST=(nnn,nnn,nnn,nnn). DEST is an alphameric value indicating one of the following devices:

- LOCAL–any local device.
- RMTn–remote terminal, "n" indicating a 1- to 3-digit numeric value, left justified with no leading zero. (For HASP compatibility, REMOTEn (n is one character) is also an acceptable way of defining a valid device specification.)
- PRINTERn–printer, "n" indicating a 1-digit numeric value, defining to which printer the output is to be sent.
- PRINTRnn–printer, "nn" indicating a 2-digit numeric value, defining to which printer the output is to be sent.
- PUNCHn–card punch, "n" indicating a 1- or 2-digit numeric value, left justified with no leading zero, defining to which card punch the output is to be sent.

FCB=*xxxx*
an alphameric value indicating the data set forms control or carriage specifications (from 1 to 4 characters).

FLASH=*xxxx*
an alphameric value (from 1 to 4 characters) specifying the name of the forms overlay to be used on output from the 3800 printer.

FLASHC=*nnn*
a value specifying the number of copies to be flashed with the specified forms overlay for the 3800 printer (not to exceed 255 copies).

FORMS=*xxxx*
an alphameric value indicating the print and punch forms (from 1 to 4 characters).

INDEX=*nn*
a value indicating the data set indexing print position offset (to the right) for the 3211 Printer (from 1 to 31).

MODIFY=*xxxx*
an alphameric value (from 1 to 4 characters) specifying the copy modification module name for the 3800 printer.

MODTRC=*n*
a number (0, 1, 2, or 3) that is the table reference character (specifying which character arrangement table is to be used) for the copy modification on the 3800 printer.

UCS=*xxxx*
an alphameric value indicating the universal character set specification (from 1 to 4 characters).

Rules for Coding

- PRINTERn, PRINTRnn, and PUNCHn are the same as LOCAL unless the specified printer or punch is subject to local print/punch routing.

- Parameters specified on the OUTPUT statement will replace any equivalent parameters specified on the referenced DD statement.
- Code as many OUTPUT statements as you need. If more than one OUTPUT statement has the same "code" starting in column 10, the first OUTPUT statement, parameters will be used. If there are duplicate parameters on the same OUTPUT statement, the last parameter will override any preceding duplicate parameter.
- Place the OUTPUT statement after the JOB statement and before any SYSOUT DD statements that refer to the OUTPUT statement.

Coding the CHARS Parameter

- If any character arrangement table specification is null, then all following must be null.
- If CHARS is coded and UCS is not coded on the OUTPUT statement or the DD statement, and if the data set is reassigned from a 3800 to a printer with the UCS feature, then the first character arrangement table name is used as the UCS id.

Coding the COPYG Parameter

- Each copy group must not exceed 255, and the total of all copy groups must not exceed 255.
- The specification of COPYG=0 is ignored.
- The specification of copy groups must not have imbedded zero groups. Once a zero is encountered, all succeeding groups must be zero or unspecified.

Coding the DEST Parameter

If more than one destination is coded, the destinations must be in parentheses. If only one destination is coded, the parentheses are optional.

Coding the FCB Parameter

- If the printer on which the data set is to be printed does not have the forms control buffer feature, the operator is sent a message to mount the proper carriage control tape.
- Do not specify any value other than 6 or 8 unless the installation indicates that you should.

Coding the FLASH and FLASHC Parameters

- If FLASHC is coded and FLASH is not, the FLASHC specification is ignored.
- If FLASH is coded and FLASHC is not, all copies are flashed with the overlay.
- FLASHC cannot be used to increase the number of copies printed. If FLASHC exceeds the number of copies specified with the COPIES parameter or the total of the copy groups, then only that specified number of copies is printed and all copies are flashed.

- If FLASHC=0, then the data set is grouped with any that require that forms overlay name, provided other grouping factors (FORMS, FCB, BURST, and UCS) are equal, and no copies are flashed.

Coding the INDEX Parameter

If the 3211 Printer has the INDEX feature, it will offset the first physical print position to the right by the number of print positions specified to cause the total print line width to be reduced by the number of print positions specified. (That is, a specification of 30 means that the maximum line width is now 30 positions less than the original value.) This parameter is ignored on other than 3211 Printers.

Coding the MODIFY and MODTRC Parameters

- The MODIFY and MODTRC parameters are ignored if the data set is written on other than a 3800.
- If MODTRC is not specified, a value of 0 is assumed. Also, if a value other than 0, 1, 2, or 3 is specified, MODTRC defaults to 0.
- If the value specified for MODTRC exceeds one less than the number of character arrangement tables requested, then MODTRC defaults to 0 (that is, if two tables are specified with the CHARS parameter, then a specification of MODTRC=2 defaults and the first table specified is used).

Coding the UCS Parameter

If UCS is coded and CHARS is not coded on the OUTPUT or DD statement, then if the data set is printed on a 3800 the UCS id is used as the character arrangement table name.

Example of the OUTPUT Statement

```
/*OUTPUT ABCD COPIES=4,DEST=RMT23
```

This statement refers to all SYSOUT data sets within the job whose DD statement specified SYSOUT=(c,,ABCD).

The PRIORITY Statement

The PRIORITY statement assigns a job selection priority to a job.

For further information on the use of PRIORITY, see "Routing a Job Through the System with HASP."

The PRIORITY statement consists of the characters /* in columns 1 and 2, the word PRIORITY in columns 3-10, and the code "p" in columns 16-17. Columns 18-80 are ignored.

```
/*PRIORITY      p
```

p
either a number between 0 and 15 or the character "*" indicating the priority of the job.

Default: If PRIORITY is not present, priority will be derived using information from (1) the JOBPARM statement, (2) the accounting information (in HASP format) on the JOB statement, or (3) an installation-defined default.

Rules for Coding

- If "p" is a number, the value of "p" is assigned as the priority of your job.
- If "p" is the character "*", the installation-defined default is used.
- The PRIORITY statement must immediately precede the JOB statement. If it does not, the PRIORITY statement is ignored and the input stream is flushed until a JOB statement or another PRIORITY statement is found.

Example of the PRIORITY Statement

```
/*PRIORITY     7
```

The job has a selection priority of 7. This value only has meaning in relation to other jobs in the system.

The ROUTE Statement

The ROUTE statement specifies the destination of the output which is not specifically routed using the DEST parameter.

The ROUTE statement consists of the characters /* in columns 1 and 2, the word ROUTE in columns 3-7, PRINT or PUNCH in columns 10-14, and one of the device specifications in columns 16-23. Columns 24-80 are ignored.

```
/*ROUTE  { PRINT }  { LOCAL   }
        { PUNCH }  { RMTn    }
                   { PRINTERn }
                   { PRINTRnn }
                   { PUNCHn   }
```

PRINT
specifies that the job's printed output is to be routed.

PUNCH
specifies that the job's punched output is to be routed.

LOCAL
any local device.

RMT*n*
remote terminal, "n" indicating a 1- to 3-digit numeric value, left-justified with no leading zero, defining to which remote terminal the output is to be sent. REMOTEn is also an acceptable way of defining a valid device specification.

PRINTER*n*
printer, "n" indicating a 1-digit numeric value, defining to which printer the output is to be sent.

PRINTR*nn*
printer, "nn" indicating a 2-digit numeric value, defining to which printer the output is to be sent.

PUNCH*n*
card punch, "n" indicating a 1- or 2-digit numeric value, left justified with no leading zero, defining to which card punch the output is to be sent.

Rules for Coding

- A ROUTE statement can be used to direct either print or punch routing of output, but not both. If both print and punch are to be routed, two cards must be used.
- Place the ROUTE statement after the JOB statement.
- PRINTERn, PRINTRnn, and PUNCHn are the same as LOCAL, unless the specified printer or punch is subject to local print/punch routing.

Examples of the ROUTE Statement

```
/*ROUTE    PRINT RMT6
```

This statement routes printed output to remote terminal 6.

```
/*ROUTE    PUNCH PUNCH2
```

This statement routes punched output to card punch 2.

The SETUP Statement

SETUP is a control statement that is used to indicate volumes needed for executing a phase of the job.

For further information on the use of the SETUP statement, see "Routing a Job Through the System with HASP."

The SETUP statement consists of the characters /* in columns 1 and 2, the word SETUP in columns 3-7, and volume serial numbers that begin in column 16. Columns 72-80 are ignored.

```
/*SETUP        volume serial number[,volume serial number...]
```

volume serial number
identifies the volume or volumes required for execution of the job.

Rules for Coding

- All SETUP statements should be placed after the JOB statement.
- As many SETUP statements as necessary can be used.

Example of the SETUP Statement

```
/*SETUP        666321,149658
```

The two volumes requested will be listed on the console when the job enters the system. The job is then placed in hold status awaiting release by the operator when the required volumes are available. The message informs the operator that the volumes should be mounted before the job is run.

INDEX

123

OS/VS2
HASP II Version 4
User's Guide
GC27-0052-0

Reader's Comment Form

Your comments about this publication will help us to improve it for you. Comment in the space below, giving specific page and paragraph references whenever possible. All comments become the property of IBM.

Please do not use this form to ask technical questions about IBM systems and programs or to request copies of publications. Rather, direct such questions or requests to your local IBM representative.

If you would like a reply, please provide your name and address (including ZIP code).

Fold on two lines, staple, and mail. No postage necessary if mailed in the U.S.A. (Elsewhere, any IBM representative will be happy to forward your comments.) Thank you for your cooperation.

Fold and Staple

First Class Permit
Number 6090
San Jose, California

Business Reply Mail

No postage necessary if mailed in the U.S.A.

Postage will be paid by:

IBM Corporation
P.O. Box 50020
Programming Publishing
San Jose, California 95150

Fold and Staple

OS/VS2 HASP II Version 4 User's Guide (File No. S370-36) Printed in U.S.A. GC27-0052-0

IBM®

International Business Machines Corporation
Data Processing Division
1133 Westchester Avenue, White Plains, New York 10604
(U.S.A. only)

IBM World Trade Corporation
821 United Nations Plaza, New York, New York 10017
(International)