GC27-6992-0
File No. S370-36

Systems

OS/VS2
HASP II Version 4
System Programmer's Guide

Program Number 370H—TX—001

VS2 SVS Release 1.7

IBM

PREFACE

This publication consists of self-contained chapters, each of which provides information necessary to generate, install, and implement capabilities of the HASP program. It is designed primarily for system programmers responsible for generating, maintaining, and extending HASP features.

Topics:

OS SYSGEN Requirements
Generating a HASP System (HASPGEN)
HASPGEN Parameters
Installing HASP in the System
Generating HASP Remote Terminal Programs (RMTGEN)
Remote Generation for Non-HASP Users
RMTGEN Parameters
Storage Requirements
Reference Listing of HASPJCL
Internal Reader
HASP-TSO Interface
Execution Batch Scheduling
Generating More Than Fifteen Logical Partitions
Multiple Devices on MULTI-LEAVING Remotes
HASP 2770 and 3780 RJE Support
3211 Forms Control Buffer Additional Loads
HASP-SMF Interface
General HASP Restrictions
HASP Overlay Programming Rules

RELATED PUBLICATIONS

- IBM 3800 Printing Subsystem Programmer's Guide for OS/VS2 SVS, GC26-3859, which contains information on the 3800 for both system programmers and application programmers.
- OS/VS2 HASP II Version 4 User's Guide, GC27-0052, which contains information for the programmer who uses HASP.

First Edition (March, 1973)

This edition, as amended by technical newsletters GN25-0121 and GN27-1553, applies to HASP II Version 4.1 in support of OS/VS2 Release 1.7 and to any subsequent versions of HASP and releases of SVS unless otherwise indicated in new editions or technical newsletters.

Information in this publication is subject to significant change. Before using this publication, consult the latest IBM System/370 Bibliography, GC20-0001, and the technical newsletters that amend the bibliography, for the editions that are applicable and current.

Requests for copies of IBM publications should be made to your IBM representative or to the IBM branch office serving your locality.

Forms for readers' comments are provided at the back of this publication. If the forms have been removed, comments may be addressed to IBM Corporation, P. O. Box 50020, Programming Publishing, San Jose, California 95150.

Contents

Illustrations

OS SYSGEN REQUIREMENTS

The only OS SYSGEN requirements for HASP are pseudo device generation and UNITNAME assignments. These mandatory SYSGEN requirements and other SYSGEN options are discussed in the following subsections.

PSEUDO DEVICES

Pseudo readers, printers and punches should be generated according to the following formulas:

Number of pseudo 2540 readers = INDD*&MAXXEQS+1

Number of pseudo 1403 printers = PRDD*&MAXXEQS+1

Number of pseudo 2540 punches = PUDD*&MAXXEQS

Number of pseudo 2520 punches = &NUMINRS

where:

INDD = maximum number of DD * (or DD DATA) cards per job step

PRDD = maximum number of print data sets per job step

PUDD = maximum number of punch data sets per job step

&MAXXEQS = maximum number of simultaneous job executions

&NUMINRS = number of Internal Reader interfaces.

Note that the term "pseudo device" implies a physically nonexistent device. An address chosen for a pseudo device may be any device address acceptable to OS, but it should not match the address of any existent device or other pseudo device. Since the allocation of pseudo devices will affect the allocation of other devices on the same channel, it is recommended that all pseudo devices be placed on a nonexistent channel.

ADDITIONAL SYMBOLIC UNIT NAMES

The symbolic unit name "A" should be assigned to all pseudo 1403 printers, except the one identified by the HASPGEN parameter &WTR. The symbolic unit name "B" should be assigned to all pseudo 2540 punches. The symbolic name "R" should be assigned to all pseudo 2540 readers, except the one identified by the HASPGEN parameter &RDR.

The pseudo device and symbolic unit name requirements are satisfied by using the SYSGEN macros IODEVICE and UNITNAME. The following examples give a simple method of generating the required devices and names for OS/VS2:

1. Pseudo 2540 Reader (except &RDR)
```
IODEVICE  UNIT=HASP-2540R,ADDRESS=xxx
UNITNAME  NAME=R,UNIT=xxx
```

2. Pseudo 2540 Reader (for &RDR)
```
IODEVICE  UNIT=HASP-2540R,ADDRESS=xxx
```

3. Pseudo 1403 Printer (except &WTR)
```
IODEVICE  UNIT=HASP-1403,ADDRESS=xxx
UNITNAME  NAME=A,UNIT=xxx
```

4. Pseudo 1403 Printer (for &WTR)
```
IODEVICE  UNIT=HASP-1403,ADDRESS=xxx
```

5. Pseudo 2540 Punch
```
IODEVICE  UNIT=HASP-2540P,ADDRESS=xxx
UNITNAME  NAME=B,UNIT=xxx
```

6. Pseudo 2520 Punch
```
IODEVICE  UNIT=HASP-2520,ADDRESS=xxx
```

The pseudo 2520 punches may be given a descriptive symbolic unit name, as in the following example:

```
UNITNAME  UNIT=(301,302,...),NAME=INTRDR
```

This will make allocation easier for programmers using the Internal Reader feature of HASP.

INSTALLATION OF THE HASP SVC

Installation of the HASP SVC is necessary upon completion of OS system generation. The installation procedure is described later in this manual under the topic "Installing HASP in the System."

GENERATING A HASP SYSTEM (HASPGEN)

To generate a HASP System which conforms to the needs of a particular installation, it is necessary to allocate and catalog several data sets, build a tailored version of the HASP source coding in one of the data sets, assemble several of the HASP source modules, and perform a few miscellaneous utility functions.

DATA SET REQUIREMENTS FOR HASPGEN

Figure 1 lists the data sets required for HASPGEN and their contents at the end of the full HASPGEN process. Figure 2 shows a sample job which will allocate and catalog the required data sets on two 2314 disk volumes. UNIT and SPACE parameters should be changed as appropriate, if other direct-access devices are used. The VOLUME parameter may be changed as desired. Data sets SYS1.UT1 and/or SYS1.UT2 may be assigned to labeled tape(s), if desired.

Data Set Name	Member Names	Description
SYS1.HASPSRC (HASP Source Modules)	$#ADD thru $XXC	96 HASP Macros
	CVT	OS CVT Macro
	HASPACCT	Accounting Routine
	HASPBR1	Return Module
	HASPCOMM	Command Processor
	HASPCON	Console Support
	HASPINIT	Initialization Routine
	HASPJCL	Sample Install Jobs
	HASPMISC	Miscellaneous Routines
	HASPNUC	HASP Nucleus
	HASPOBLD	Overlay Build Utility
	HASPPRPU	Print/Punch Processor
	HASPRDR	Input Processor
	HASPRTAM	Remote Support
	HASPSVC	SVC Routine
	HASPWTR	SMB Writer
	HASPXEQ	Execution Processors
	HRTPB360	360 and M20 BSC Remote Program
	HRTPLOAD	1130 Loader Program
	HRTPOPTS	RMTGEN Standard Option Lists
	HRTPSYS3	System/3 Remote Program
	HRTP1130	1130 Remote Program
	IEFUCBOB	OS UCB Macro
	IHASPP	VS SETPRT Parmlist
	NULL	HASP Macro
SYS1.HASPOBJ (HASP Object Modules)	HASPBR1	Same as SYS1.HASPSRC
	HASPNUC	
	HASPRDR	
	HASPXEQ	
	HASPWTR	
	HASPPRPU	
	HASPACCT	
	HASPMISC	
	HASPCON	
	HASPRTAM	
	HASPCOMM	
	HASPINIT	
	HASPSVC	
	HASPOBLD	

Figure 1. HASPGEN Data Set Description (Sheet 1 of 2)

Data Set Name	Member Names	Description
SYS1.HASPMOD (HASP Load Modules)	HASPGEN	HASPGEN Program
	EXRMTGEN	Initial RMTGEN Program
	RMTGEN	RMTGEN Control Program
	GENRMT	RMTGEN Effector Program
	LETRRIP	1130 RMTGEN Post-Processor
	SYS3CNVT	System/3 RMTGEN Post-Processor
	HASPOBLD	Overlay Build Utility
SYS1.UT1		Sequential Scratch Data Set
SYS1.UT2		Sequential Scratch Data Set
SYS1.UT3		Sequential Scratch Data Set

Figure 1. HASPGEN Data Set Description (Sheet 2 of 2)

```
//CATALOG  JOB (0000,0000),'HASP DATA SETS',MSGLEVEL=1
//SCRATCH EXEC PGM=IEHPROGM
//TWOSPACK  DD UNIT=2314,VOLUME=SER=222222,DISP=OLD
//HASP      DD UNIT=2314,VOLUME=SER=HASP,DISP=OLD
//SYSPRINT  DD SYSOUT=A
//SYSIN     DD *
       UNCATLG DSNAME=SYS1.HASPSRC
       UNCATLG DSNAME=SYS1.HASPOBJ
       UNCATLG DSNAME=SYS1.HASPMOD
       UNCATLG DSNAME=SYS1.UT1
       UNCATLG DSNAME=SYS1.UT2
       UNCATLG DSNAME=SYS1.UT3
       SCRATCH VTOC,VOL=2314=222222,PURGE
       SCRATCH VTOC,VOL=2314=HASP,PURGE
/*
//ALLOCAT EXEC PGM=IEHPROGM
//SYSIN     DD DUMMY
//SYSPRINT  DD DUMMY
//HASPSRC   DD DSNAME=SYS1.HASPSRC,UNIT=2314,VOLUME=SER=HASP,
//             DISP=(,CATLG),SPACE=(CYL,(40,5,10)),
//             DCB=(RECFM=FB,LRECL=80,BLKSIZE=3360)
//HASPOBJ   DD DSNAME=SYS1.HASPOBJ,UNIT=2314,VOLUME=SER=HASP,
//             DISP=(,CATLG),SPACE=(CYL,(5,5,5)),
//             DCB=(RECFM=FB,LRECL=80,BLKSIZE=400)
//HASPMOD   DD DSNAME=SYS1.HASPMOD,UNIT=2314,VOLUME=SER=HASP,
//             DISP=(,CATLG),SPACE=(CYL,(5,5,5))
//UT1       DD DSNAME=SYS1.UT1,UNIT=2314,VOLUME=SER=222222,
//             DISP=(,CATLG),SPACE=(CYL,(20,5))
//UT2       DD DSNAME=SYS1.UT2,UNIT=2314,VOLUME=SER=HASP,
//             DISP=(,CATLG),SPACE=(CYL,(20,5))
//UT3       DD DSNAME=SYS1.UT3,UNIT=2314,VOLUME=SER=222222,
//             DISP=(,CATLG),SPACE=(CYL,(20,5))
```

Figure 2. Sample Job to Catalog Data Sets for HASPGEN

HASPGEN PARAMETER CARDS

All HASPGEN parameters and their default values are discussed under HASPGEN Parameters. After the desired value for each parameter has been determined, the values of those that are to be changed from the default values are usually punched into cards, to be read by the HASPGEN utility program.

Each parameter should be punched in the format "option=value", beginning in column 1 of a card. "Option" represents a HASPGEN parameter and "value" represents a permissible value for that parameter (see HASPGEN Parameters). The format must not contain embedded blanks. The first blank terminates the "value" field, and the rest of the card may contain comments.

HASPGEN parameter cards may occur in a deck in any order. If the same parameter occurs more than once, the last occurrence determines the parameter's value. A deck of one or more HASPGEN parameter cards is usually terminated by a card with "END" punched in columns 1-3. If symbolic updates (PTFs or user modifications) are to be applied, the "END" card should be replaced by an "UPDATE" card. Alternate methods of entering HASPGEN parameters are discussed later in this chapter.

HASPGEN UPDATE CARDS

Source coding of any member in SYS1.HASPSRC (see Figure 1) may be updated by cards punched according to formats acceptable to the IEBUPDTE OS utility program. This is the method used to apply official HASP maintenance changes (PTFs, etc.) and user modifications to HASP, if any. Updates are placed after the HASPGEN parameter deck, immediately following a card with "UPDATE" punched in columns 1-6.

All IEBUPDTE control cards, except the ./ ALIAS ... detail statement, are defined for use with the HASPGEN update. The ./ NUMBER ... detail statement will be accepted but will be ignored. Only the NAME, SEQ1, and SEQ2 keywords will be interpreted for meaning. Other keyword parameters are ignored and may be omitted.

A card without "./" in columns 1 and 2 replaces an existing source card (if columns 73-80 match an existing card in the member) or is inserted between existing source cards, according to ascending collating sequence based on columns 73-80. Cards that are blank in columns 73-80 are inserted immediately following the last modification card that was in ascending collating sequence. Update cards that do not maintain an ascending collating sequence in columns 73-80, but are not blank, will terminate the HASPGEN with an update error.

All PTFs (and user modifications, if any) that apply to one source module must be integrated in ascending sequence number order into a single deck, beginning with a CHANGE card naming that module. If more

than one module is updated, the decks must be placed together so that the module names on CHANGE cards are in ascending collating sequence, as listed in Figure 1 under SYS1.HASPSRC.

The last source update card must be followed by a ./ ENDUP control card, a /* delimiter card, or a hardware end-of-file. Figure 3 shows a composite deck of HASPGEN parameters and source updates in correct order.

```
Columns
0                                                                    7      8
1                                                                    3      0
&NUMLNES=1
&BSCCPU=YES
&LINE01=02011
RMT01=01010100153643
UPDATE                    (END if no source updates follow)
./       CHANGE  NAME=HASPMISC
          .
          .
          (modifications to module HASPMISC)                         nnnnnnnn
          .
          .
          .
./        CHANGE  NAME=HASPWTR
          .
          .
          .
          (modifications to module HASPWTR)                          mmmmmmmmm
          .
          .
          .
./        ENDUP
/*
```

Figure 3. Sample HASPGEN Parameter and Update Deck

STANDARD COMPLETE HASPGEN PROCESS

For most installations, a complete standard HASPGEN may be performed (if the required data sets are allocated and cataloged) simply by using the first file of the distributed HASPGEN tape as an OS input stream and executing, in order, all the jobs it contains. Figure 4 lists the jobs, steps, and functions of each, in the order they occur in the first file of the tape.

The first file of the tape may be executed directly, under HASP with VS2, by using the procedure for tape input to HASP as outlined in the section on Internal Reader.

If VS2 without HASP is used, the first file of the tape must be punched and the first job (HASPGEN) must be run to completion before the other jobs are read by the OS Reader/Interpreter. During subsequent generations with the same OS, the first file may be processed directly by the OS RDR.

During the first job (HASPGEN), the HASPGEN utility program will write the following WTOR message on the console:

```
nn ENTER HASPGEN OPTION CHANGES (option=value), CARDS, UPDATE, OR END
```

The composite HASPGEN parameter and update deck (example Figure 3) should be placed in the 2540 Card Reader and the following reply should be entered:

```
R nn,cards
```

The listing output of the HASPGEN job includes:

1. All HASPGEN parameters with their default values
2. User changes to HASPGEN parameters
3. Source changes to modules by HASPGEN Update.

The jobs (listed in Figure 4) must execute in sequential order under a single initiator.

The load module name ASMBLR, used for the assemblies, must be an assembler that supports the S/370 Advanced Function instructions, e.g., the VS2 system assembler.

If the HASPGEN parameter &BSCCPU is set to include programmable RJE support, job HRMTGEN will issue another WTOR console message, which allows optional generation of Remote Terminal programs as part of the full HASPGEN process. Refer to Generating HASP Remote Terminal Programs for further details.

If all jobs in the first file of the HASPGEN tape are executed successfully, all data sets and members (listed in Figure 1) will be completed and the punched card output will contain:

1. Any Remote Terminal Programs created by HRMTGEN (optional; see Generating HASP Remote Terminal Programs).

2. HASPJCL, the deck of sample jobs to install HASP (described under Installing HASP in the System).

HASPGEN COMPLETION CODES

HASPGEN determines the success of the overall source generation process and returns a completion code as follows:

1. Completion Code 0 - No errors were detected (or all detected errors were corrected, and all members of the SYS1.HASPSRC data set were successfully constructed).

2. Completion Code 24 - An unrecoverable error, which prohibited the successful construction of the SYS1.HASPSRC data set, was detected. A further indication of the error will be displayed on the operator's console as follows:

   a. xxxxxxxx -- INVALID HASPGEN PARAMETER - xxxxxxxx is not one of the defined HASPGEN parameter , and a batch HASPGEN is being performed. If a batch HASPGEN is not being performed, the program is not terminated, and control is returned to the console for operator correction.

   b. INVALID SYNTAX - The HASPGEN parameter specification violates the requirements stated in the previous sections, and a batch HASPGEN is being performed. If a batch HASPGEN is not being performed, the program is not terminated, and control is returned to the console for operator correction.

   c. HASPGEN UPDATE ERROR -- HASPGEN TERMINATED - An error was discovered in the source update process. This error may result from an error on an update control card or a sequence error in the update deck. The last card processed or current card being processed will be the last card listed on the SYSPRINT data set.

   d. INSUFFICIENT DIRECTORY SPACE IN HASPSRC DATA SET - The directory space in the SYS1.HASPSRC data set is not large enough to contain all the members in the HASP source data set. The SYS1.HASPSRC data set should be scratched and reallocated with a larger directory size. The HASPGEN should then be repeated.

   e. DIRECTORY I/O ERROR ON HASPSRC DATA SET - An I/O error occurred during update of the directory for the SYS1.HASPSRC data set.

Job	Step (If Multi-Step)	Function
HASPGEN	LNK	Link edits object decks for HASPGEN, EXRMTGEN, RMTGEN, GENRMT, LETRRIP, and SYS3CNVT into SYS1.HASPMOD
	HASPGEN	Executes HASPGEN program which reads all source code from second file of tape, applies user HASPGEN Parameter modifications and (optionally) source code modifications, and builds each source member in SYS1.HASPSRC
	PROCS	Adds procedures ASMHASP, HASPGEN, and RMTGEN to SYS1.PROCLIB, if not already there
HASMBR1		Assembles source module HASPBR1
HASMNUC		Assembles source module HASPNUC
HASMRDR		Assembles source module HASPRDR
HASMXEQ		Assembles source module HASPXEQ
HASMWTR		Assembles source module HASPWTR
HASMPRPU		Assembles source module HASPPRPU
HASMACCT		Assembles source module HASPACCT
HASMMISC		Assembles source module HASPMISC
HASMCON		Assembles source module HASPCON
HASMRTAM		Assembles source module HASPRTAM
HASMCOMM		Assembles source module HASPCOMM
HASMINIT		Assembles source module HASPINIT
HASMSVC		Assembles source module HASPSVC
HASMOBLD	OBLD	Assembles source module HASPOBLD
	LNKOBLD	Link edits object module HASPOBLD into SYS1.HASPMOD
HRMTGEN		Performs optional initial RMTGEN for one or more HASP Remote Terminal Programs
HASPJCL	PRINT	Prints source member HASPJCL
	PUNCH	Punches source member HASPJCL

Figure 4. HASPGEN Tape First File Job Description

SOME HASPGEN VARIATIONS

An installation may find it necessary or desirable to vary some of the standard HASPGEN process described previously. A few possible variations are discussed in the following paragraphs.

The necessity of punching or copying the first file of the HASPGEN tape to generate under a system without HASP is discussed under Standard Complete HASPGEN Process. The installation's requirements for particular job card accounting fields or classes, or the absence of a 2540 Card Reader, may also require that the first file be punched, listed, and used as an input stream after appropriate modifications to the JCL.

During execution of the HASPGEN utility, responses to the WTOR message other than CARDS may be used. Individual HASPGEN parameters may be entered with a reply text of 'option=value'. (Option represents a HASPGEN parameter, and value represents a permissible value for that parameter.) Lower case may be used, but no blanks or comments are allowed. Each HASPGEN parameter entered from the console is acknowledged by a message if correct or by a diagnostic, with opportunity to reenter a correct form. The same parameter may be entered repeatedly; only the last value entered will be used. The 'CARDS' reply may be entered at any time to enable further parameter reading from the 2540 Card Reader. If all parameters are entered from the console, a reply text of 'UPDATE' may be entered to enable reading of an update deck only (all cards after UPDATE in Figure 3) from the 2540 Card Reader. If all parameters are entered from the console, and if there are no updates, a reply text of 'END' may be used to terminate all entry to HASPGEN.

If all the actions of the HASPGEN job (Figure 4) are performed once and the three partitioned data sets (SYS1.HASPSRC, SYS1.HASPOBJ, and SYS1.HASPMOD) are preserved on a disk pack, full or partial HASPGENs may be performed under a production batch system at a later time (see Figure 5 for sample jobs to be used). Execution of the HASPGEN procedure invokes only the HASPGEN utility; a PARM field causes the WTOR and reply to be omitted so that parameters and updates are read directly from the input stream. The data set SYS1.HASPSRC would normally be scratched and reallocated, prior to running this job. When all 14 assembly jobs are to be run (Figure 4), SYS1.HASPOBJ should also be scratched and reallocated. Figure 5 shows how to use the ASMHASP proc for assemblies. If HASPOBLD is assembled, a step should be added to link edit HASPOBLD from SYS1.HASPOBJ into SYS1.HASPMOD.

```
//HASPGEN  JOB  ...
//GEN      EXEC  HASPGEN
//HASPGEN.OPTIONS DD *
  (deck as in Figure 3)
/*
//HASMNUC  JOB  ...
//NUC      EXEC  ASMHASP,MODULE=HASPNUC
//HASPINIT JOB  ...
//INIT     EXEC  ASMHASP,MODULE=HASPINIT
```

Figure 5. Sample Batch HASPGEN Jobs

A partial HASPGEN may be used to save processing time, if only minor changes are made to HASPGEN parameters or only a small number of modules are changed by updates. The recommended process is to scratch and reallocate SYS1.HASPSRC only and then use the HASPGEN proc and full parameter/update deck to recreate SYS1.HASPSRC. Only required assemblies are performed, using the ASMHASP proc and producing decks that replace those of the same name in SYS1.HASPOBJ.

A module must be reassembled if a HASPGEN parameter(s) is changed from the previous HASPGEN and if the module depends upon the parameter(s). Figure 6 indicates module dependencies. If an altered parameter is used as a default value for another parameter, that parameter is changed also, and all modules depending on it must be reassembled. For example, if &NUMTPPR is allowed to default, then changing &NUMLNES will cause &NUMTPPR to change. A change (from the previous HASPGEN) in the update portion of the deck for a module also requires that the module be reassembled. When reassembly requirements are doubtful (e.g., changes in update deck for any member of SYS1.HASPSRC other than one of the 14 assembly modules), all 14 modules must be reassembled.

The module HASPBR1 does not actually depend on any generation parameter. However, it contains the most completely commented documentation of all HASP control blocks. Therefore, HASPBR1 should be reassembled periodically to provide listing documentation current with the operational HASP.

Figure 6 refers to each assembly module by a single alphabetic character as follows:

H = HASPNUC

R = HASPRDR

X = HASPXEQ

T = HASPWTR

P = HASPPRPU

A = HASPACCT

V = HASPMISC

W = HASPCON

M = HASPRTAM

C = HASPCOMM

N = HASPINIT

S = HASPSVC

O = HASPOBLD

&APGPRTY-X
&AUTORDR-CN
&BSCCPU -PM
&BSC2770-M
&BSC2780-M
&BSC3780-M
&BSHPRES-M
&BSHPRSU-M
&BSHTAB -M
$BSPACE -W
&BSVBOPT-M
&BUFSIZE-HRXPMN
$CKPTIME-V
&CLS(n) -X
&DEBUG -HXVCNO
$DELAYTM-M
&DMNDSET-P
$ESTIME -R
$ESTLNCT-R
$ESTPUN -R
&FCBV -PC
&HDWCHAR-NP
&HDWFCB -NP
&JBURST -NXP
&JCOPYLM-P
&JFLASH -NP
&JMARK -NP
&JXLATn -NP
$LINECT -R
LINEmm -N
&LOGOPT -X
&LONGCOM-C
&MAXCLAS-XCN
&MAXJOBS-VN
&MAXPART-X
&MAXXEQS-HXVWCN
&MINBUF -N
&MINJOES-P
&MLBFSIZ-M
&NOPRCCW-HPC
&NOPUCCW-HPC
&NUMBUF -N
&NUMCLAS-NWRCPXHM
&NUMDA -HRXPAVWMCN
&NUMDDT -X
&NUMINRS-HRXN
&NUMJOES-PVN
&NUMLNES-HRPWMN
&NUMOACE-N
&NUMOSC -C
&NUMPRTS-HPVCN
&NUMPUNS-HPVN
&NUMRDRS-HN
&NUMRJE -MCN
&NUMSMFB-HPAVMCN
&NUMTGV -HRXPAVWCN
&NUMTPBF-N
&NUMTPPR-HPVN
&NUMTPPU-HPVMN
&NUMTPRD-H
&NUMWTOQ-HN
&NUM3800-HRXTPAVWMCN
&OLAYLEV-RXPAVMCN
&OREPSIZ-HN
&OSC(n) -X
&OSINOPT-R
&OUTPOPT-X
$OUTXS -X
&PID(n) -X
&PRI(n) -X
$PRIDCT -P
&PRIHIGH-V
&PRILOW -V
&PRIRATE-HV
$PRTBOPT-P
&PRTFCB -NP
&PRTRANS-PM
&PRTUCS -N
$PUNBOPT-P
&RDR -XN
$REPRDR -N
$REPWTR -N
&RESCORE-N
&RJOBOPT-R
RMTnn -N
$RPRBOPT-P
&RPRI(n)-R
&RPRT(n)-R
&RPS -HNT
$RPUBOPT-P
&SMFRSIZ-N
&SPOLMSG-PMN
&SPOOL -N
&STDFORM-NP
&TIMEOPT-X
$TIMEXS -X
&TPBFSIZ-MN
$TPIDCT -P
&TRACE -HXVWN
&TSOSTCN-HW
&USASCII-M
$WAITIME-M
&WCLSREQ-T
&WTR -XN
&WTRCLAS-XTR
&XBATCHC-RXWCN
&XBATCHN-RXWC
&XLIN(n)-RX
&XPRI(n)-RX
$$x -X

Figure 6. Module Dependencies on HASPGEN Parameters

COMPATIBLE HASP SPOOL VOLUMES

There are two levels of SPOOL volume compatibility between two different HASP generations: (1) the compatibility level that allows one HASP System to use another SPOOL volume without requiring complete reformatting of the SPOOL data sets and (2) the compatibility level that allows one HASP System to continue another system's job processing with job data contained on the SPOOL data set checkpoint record(s).

If a new release of HASP or an official HASP modification requires a reformat or prohibits a "warm start" the release documentation will alert the installation. Alteration of certain HASPGEN parameters between HASP generations of the same release may also produce incompatible SPOOL volumes. The only HASPGEN parameter whose alteration requires reformatting of the SPOOL volumes between systems is:

&BUFSIZE

Alteration of the following HASPGEN parameters prohibits "warm start" between systems:

&NUMDA	&NUMPRTS	&NUMTPPU
&NUMTGV	&NUMTPPR	&NUM3800
&MAXJOBS	&SPOLMSG	
&NUMJOES	&NUMPUNS	

Many of these parameters are automatically assigned values by HASPGEN, based on the assignment of other parameters. These parameters must be overridden to provide compatibility. For example: &SPOLMSG depends upon the number of remote terminals capable of communicating with HASP (&NUMRJE), which in turn depends on the number of lines (&NUMLNES), unless overridden by a user specification.

HASPGEN PARAMETERS

Generation of a HASP system involves specification of certain parameters, called HASPGEN parameters. With these parameters, the system programmer specifies the characteristics of the System/370 to be used with HASP and the optional features to be included in the generated HASP System.

The following subsections describe each HASPGEN parameter, giving an explanation, the default value, and, frequently, notes which expand on the explanation and refer to related HASPGEN parameters. The HASPGEN parameters are given in alphabetical order (ignoring the first character if it is & or $), except for parameter $$x, which appears last, and the IBM 3800-related HASPGEN parameters, which are grouped in a section at the end of this chapter.

&APGPRTY

Explanation: Variable symbol &APGPRTY specifies the priority level of the Automatic Priority Group in VS2. &APGPRTY should be set equal to the priority specified in the APG parameter at OS SYSGEN or NIP time.

Default: &APGPRTY=7

Notes: This parameter is used in conjunction with related parameters &PRI(n). If a job is initiated under a HASP logical initiator n, whose &PRI(n) is equal to &APGPRTY, then HASP will remove DPRTY= parameters from EXEC statements of that job, so each step will be initiated at the dispatching priority of the VS2 APG.

&AUTORDR

Explanation: Variable symbol &AUTORDR specifies the inclusion or exclusion of code in HASP to recognize automatically when a physical card reader available to HASP becomes ready. The specification must be either YES or NO.

Default: &AUTORDR=YES

Notes: If &AUTORDR=NO, HASP's physical card readers remain in the INACTIVE state when they become ready; the operator must issue a $SRDRn command to make HASP begin reading cards from READERn.

&BSCCPU

Explanation: Variable symbol &BSCCPU specifies inclusion or exclusion, in the HASP Remote Terminal Access Method, of support for HASP MULTI-LEAVING Remote Job Entry.

Default: &BSCCPU=NO

&BSC2770

Explanation: Variable symbol &BSC2770 specifies inclusion or exclusion, in the HASP Remote Terminal Access Method, of Remote Job Entry support for the 2770 Data Communication System. The specification must be either YES or NO.

Default: &BSC2770=NO

&BSC2780

Explanation: Variable symbol &BSC2780 specifies inclusion or exclusion, in the HASP Remote Terminal Access Method, of Remote Job Entry support for the 2780 Data Transmission Terminal. The specification must be either YES or NO.

Default: &BSC2780=NO

&BSC3780

Explanation: Variable symbol &BSC3780 specifies inclusion or exclusion, in the HASP Remote Terminal Access Method, of Remote Job Entry support for the 3780 Data Communication Terminal. The specification must be either YES or NO.

Default: &BSC3780=NO

&BSHPRES

Explanation: Variable symbol &BSHPRES specifies inclusion or exclusion, in the HASP Remote Terminal Access Method, of support for the Space Compression/Expansion feature of 2770 and 3780 terminals. The specification must be either YES or NO.

Default: &BSHPRES=NO

Notes:

1. This support must be included if any 2770 or 3780 terminal will transmit to HASP using the Space Compression/Expansion feature. (See appropriate component description manuals for details.)

2. Use of this support for output to any terminal is controlled by specification in the RMTnn parameter for that terminal. (See RMTnn parameter.)

&BSHPRSU

Explanation: Variable symbol &BSHPRSU specifies inclusion or exclusion of the HASP Remote Job Entry Printer Interrupt feature for binary synchronous hardware terminals. If this feature is included, the remote terminal operator may interrupt printing to transmit jobs or HASP commands to HASP. The specification must be either YES or NO.

Default: &BSHPRSU=YES

Notes: If &BSHPRSU=YES, HASP will recognize certain control characters (from the binary synchronous hardware terminal), which indicate that the printer has stopped. Appendixes to the HASP Operator's Guide contain additional information concerning the use of the feature.

&BSHTAB

Explanation: Variable symbol &BSHTAB specifies inclusion or exclusion, in the HASP Remote Terminal Access Method, of support for the Printer Horizontal Format Control feature of 2770, 2780, and 3780 terminals. The specification must be either YES or NO.

Default: &BSHTAB=YES

Notes: Use of this support for output to any 2770, 2780, or 3780 terminal is controlled by specification in the RMTnn parameter for that terminal. (See RMTnn parameter.)

$BSPACE

Explanation: Ordinary symbol $BSPACE specifies the character that will be interpreted as the System/370 hardware-defined backspace character X'16'. When the $BSPACE character is entered on any operator console, it will be removed from the command text along with the previously entered character (if any). Characters following the $BSPACE character will be shifted left to replace the removed characters. The $BSPACE edit is performed on all commands entered via operator command input sources, regardless of their position in the text of the data entered. $BSPACE does not apply to HASP card reader, commands entered from any sources other than operator consoles, or remote work station sources. $BSPACE is specified using the two hexadecimal digit representation of the EBCDIC character.

Default: $BSPACE=5F

Notes:

1. The default specification indicates that the EBCDIC character "¬" is to be used to backspace command entry on operator consoles.

2. The character selected for the backspace function must be chosen with extreme caution since it eliminates the use of that character (except as a backspace operation) in all commands and replies to WTORs.

&BSVBOPT

Explanation: Variable symbol &BSVBOPT specifies inclusion or exclusion, in the HASP Remote Terminal Access Method, of code to recognize an EM (end of media) punch in card images transmitted nontransparently by the 2780 Data Transmission Terminal. The specification must be either YES or NO.

Default: &BSVBOPT=NO

&BUFSIZE

Explanation: Variable symbol &BUFSIZE specifies the size in bytes of each HASP buffer. If the value specified is not a multiple of eight, it will be adjusted upward to a multiple of eight. The specification must be an integer not larger than 4008 and not smaller than:

```
400+&NUMDA*&NUMTGV/8
```

Default: &BUFSIZE=1960

Notes:

1. The default &BUFSIZE of 1960 is the maximum size that will allow 2 buffers per page of virtual storage and good utilization of both 2314/2319 and 3330 track capacities, i.e., 3 or 6 records per track, respectively. The maximum permissible value (4008) allows 3 records per track on the 3330.

2. &BUFSIZE must be 536 or greater if 3211 Printers are used by HASP.

3. Each HASP buffer is allocated to virtual storage, so the IOB (88 bytes) and the data area (&BUFSIZE bytes) are always contained in a single 4K page.

4. The maximum number of /*OUTPUT control cards allowed per job is:

```
(&BUFSIZE-4)/32
```

$CKPTIME

Explanation: Ordinary symbol $CKPTIME specifies the interval, in seconds, at which certain HASP information will be checkpointed for warm start.

Default: $CKPTIME=60

Notes: The time interval specified is a maximum. Checkpoints are also taken when a job changes its status in the HASP job queue.

&CLS(n)

Explanation: Subscripted variable symbols &CLS(n) specify HASP job classes. The nth HASP logical partition may select, for OS execution, a job from the HASP job queue, only if the job's class (specified by the user in the CLASS parameter of the JOB card, or defaulted to A) is one of the characters specified in the &CLS(n) parameter or specified by the operator in the set command, $T Imm,list (where mm=&PID(n)). Each specification must be a 1- to 53-character string of valid HASP job classes. The same HASP job class may be specified in two or more specifications.

Default:
```
&CLS(1)=A
&CLS(2)=BA
&CLS(3)=CBA
&CLS(4)=DCBA
&CLS(5)=EDCBA
&CLS(6)=FEDCBA
&CLS(7)=GFEDCBA
&CLS(8)=HGFEDCBA
&CLS(9)=IHGFEDCB
&CLS(10)=JIHGFEDC
&CLS(11)=KJIHGFED
&CLS(12)=LKJIHGFE
&CLS(13)=MLKJIHGF
&CLS(14)=NMLKJIHG
&CLS(15)=ONMLKJIH
```

Notes:

1. Only the first &MAXPART specifications, &CLS(1) through &CLS(&MAXPART), will be used.

2. If &MAXCLAS is specified less than 53, a maximum of &MAXCLAS characters for each value should be specified.

3. If &MAXPART is specified greater than 15, see Generating More Than Fifteen Logical Partitions.

&DEBUG

Explanation: Variable symbol &DEBUG specifies inclusion or exclusion of debugging code in the generated HASP System. The specification must be either YES or NO.

Default: &DEBUG=NO

Notes: The &DEBUG option is independent of the &TRACE option.

$DELAYTM

Explanation: Ordinary symbol $DELAYTM specifies a delay time in micro-seconds to be applied by the HASP Remote Terminal Access Method when transmitting to either a MULTI-LEAVING System/360 Model 20, Submodel 2, 4, or 6, or to a 2922 remote terminal over a high-speed (19,200 baud or greater) teleprocessing line. This delay time will avoid the possibility of certain line errors.

Default: $DELAYTM=100

Notes: If data overrun line errors occur at the work station with the default value, the value should be increased to minimize such errors.

&DMNDSET

Explanation: Variable symbol &DMNDSET specifies whether inline printer setup will be allowed for data sets whose SYSOUT class matches the job message class.

Default: &DMNDSET=YES

Notes:

1. If &DMNDSET=YES, all SYSOUT data sets whose class matches message class will be printed on one printer, with appropriate setup messages to the operator as the data sets are printed.

2. If &DMNDSET=NO or if SYSOUT class does not match message class separate work elements will be created for each unique setup required. Thus, data sets can be printed simultaneously on all printers available.

$ESTIME

Explanation: Ordinary symbol $ESTIME specifies the default estimated execution time, in minutes, for a job. The specification must be an integer greater than zero.

Default: $ESTIME=2

Notes: If a user does not specify, in the accounting field of his JOB card, or on a /*JOBPARM card, a value for estimated execution time, the value $ESTIME is used.

$ESTLNCT

Explanation: Ordinary symbol $ESTLNCT specifies the default estimated print line count, in thousands of lines, for a job. The specifications must be an integer greater than zero.

Default: $ESTLNCT=2

Notes: If a user does not specify, in the accounting field of his JOB card, or on a /*JOBPARM card, a value for estimated print line count, the value $ESTLNCT is used.

$ESTPUN

Explanation: Ordinary symbol $ESTPUN specifies the default estimated punched card count, in cards, for a job. The specification must be an integer greater than zero.

Default: $ESTPUN=100

Notes: If a user does not specify, in the accounting field of his JOB card, or on a /*JOBPARM card, a value for estimated card count, the value $ESTPUN is used.

&FCBV

Explanation: Variable symbol &FCBV specifies inclusion or exclusion of the 3211 Variable Forms Control Buffer loading capability. If set to YES, the "V" specified FCB image is generated and the code to support the $TF... command is included. The specification must be either YES or NO.

Default: &FCBV=NO

&JCOPYLM

Explanation: Variable symbol &JCOPYLM specifies the maximum allowable value for job output copies requested in the JOB card accounting field or on the /*JOBPARM card. The specification must be an integer between 1 and 255, inclusive.

Default: &JCOPYLM=3

Notes:

1. If the number of job copies requested is greater than the value of &JCOPYLM, the request will be reduced to the value of &JCOPYLM. No error message will be produced.

2. The setting of this parameter does not affect requests for multiple copies of data sets via the /*OUTPUT card.

$LINECT

Explanation: Ordinary symbol $LINECT specifies the default maximum number of lines to be printed per page of a job's printed output.

Default: $LINECT=61

Notes:

1. If a user does not specify, in the accounting field of his JOB card, or on a /*JOBPARM card, a value for line count, the value $LINECT is used.

2. Setting $LINECT=0 will cause automatic page overflow, normally provided by HASP, to be suppressed unless overridden by the JOB card accounting parameter or /*JOBPARM card specification.

3. If a print data set is generated without any ejects (no skips to any channel in the carriage tape), and if $LINECT=0 or the LINECT parameter on the JOB card or /*JOBPARM card of the job producing the data set is zero, then that data set will be treated as one page, when forward-spaced, backspaced, interrupted, or warm started while printing.

LINEmm

Explanation: Ordinary symbols LINEmm specify the characteristics of teleprocessing lines to be used by HASP Remote Job Entry.

Lines must be defined consecutively, starting with LINE01. Each specification must be a 5-character string of the form:

LINEmm=aaalc

where the letters represent the following:

Code Letters	Range	Description
mm	01-99	Line number
aaa	000-FFF	BSC adapter address (see Note 2)
l	0-5	Line descriptions as follows: 0 = Interface A - half-duplex (1200-9600 baud) 1 = Interface A - full-duplex (1200-9600 baud) 2 = Interface A - full-duplex (19.2-230.4 k-baud) 3 = Interface B - half-duplex (1200-9600 baud) 4 = Interface B - full-duplex (1200-9600 baud) 5 = Interface B - full-duplex (19.2-230.4 k-baud)
c	0-7	Code as follows: 0 = Code A - EBCDIC - no transparency 1 = Code A - EBCDIC - transparency 2 = Code A - USASCII - no transparency 3 = Code A - USASCII - transparency 4 = Code B - EBCDIC - no transparency 5 = Code B - EBCDIC - transparency 6 = Code B - USASCII - no transparency 7 = Code B - USASCII - transparency

Default: LINEmm=***01

Notes:

1. Parameter &NUMLNES must specify the number of LINEmm specifications to be included in the generated HASP system.

2. The unit address aaa may be specified as ***. HASP initialization will assign unit addresses to lines whose unit addresses are specified as *** by scanning the OS UCBs. A teleprocessing UCB whose device type field specifies a 2701 BSC Adapter or a 2703 BSC Adapter will be recognized as a UCB defining a line. If the unit address of such a UCB is not specified explicitly in any of the first &NUMLNES line definitions LINEmm, HASP initialization will assign the UCB to the first line number whose unit address is specified ***

and will change the *** to the EBCDIC address specified in the UCB. However, if no line definition remains whose unit address is ***, HASP will not use the line.

3. If a line specification LINEmm designates USASCII, that line cannot be used with any but 2770, 2780, or 3780 USASCII terminals. HASP will translate each record it receives into EBCDIC, and each record it transmits into USASCII before transmission. See also HASPGEN parameter &USASCII.

4. Interface B and Code B refer to the second code or interface in a BSC Adapter with the Dual Communications Interface and/or Dual Code special feature(s).

5. The same unit address aaa may be specified in more than one LINEmm definition to allow use of different interfaces or codes available in a single BSC Adapter. HASP will allow only one such LINEmm to be started by the operator at any one time.

&LOGOPT

Explanation: Variable symbol &LOGOPT specifies inclusion or exclusion of code to support the HASP Job Log feature. The specification should be either YES or NO.

Default: &LOGOPT=YES

Notes:

1. The HASP Job Log is output with the user's console messages produced during processing of the job and replies to WTORs issued during processing of the job.

2. If &LOGOPT=YES, the HASP Job Log may be suppressed on an individual job basis, via a parameter in the accounting field of the JOB card or via a parameter on a /*JOBPARM card.

3. If the HASP Job Log is suppressed, the HASP statistics information, normally printed with the job, is also suppressed.

&LONGCOM

Explanation: Variable symbol &LONGCOM specifies whether the alternate form of HASP commands is available to the operator. A specification of NO indicates that the long form of commands is not available, and a specification of YES indicates that the long form is available.

Default: &LONGCOM=NO

&MAXCLAS

Explanation: Variable symbol &MAXCLAS specifies the maximum number of job classes which may be specified via the HASP command $TIn,list for a HASP logical partition. The specification must be an integer from 1 to 64, inclusive.

Default: &MAXCLAS=8

Notes: No more than &MAXCLAS characters may be specified for each of the parameters &CLS(n).

&MAXJOBS

Explanation: Variable symbol &MAXJOBS specifies the maximum number of jobs that can be in the HASP System at any given time. The specification must be an integer greater than zero.

Default: &MAXJOBS=100

Notes:

1. This variable does not affect the range of HASP job numbers, 1 to 9999.

2. This variable strongly influences the size of the first and second HASP checkpoint records. The size of the first checkpoint record is:

```
16*(&MAXJOBS)+28*(&NUMPRTS+&NUMPUNS+&NUMTPPR+
&NUMTPPU)+32
```

The size of the second checkpoint record is:

```
&MAXJOBS*8
```

A third checkpoint record is described in the section explaining the &NUMJOES generation parameter.

If any checkpoint record is longer than the track size of the device on which the primary SPOOL volume is mounted, HASP will not operate correctly.

&MAXPART

Explanation: Variable symbol &MAXPART specifies the number of HASP logical partitions to be defined. The specification must be an integer between 1 and 63, inclusive.

Default: &MAXPART=&MAXXEQS

Notes:

1. The nth logical partition is further defined by the specifications &PID(n), &PRI(n), &OSC(n), and &CLS(n).

2. If &MAXPART is specified greater than 15, see Generating More Than Fifteen Logical Partitions.

&MAXXEQS

Explanation: Variable symbol &MAXXEQS specifies the maximum number of jobs which may concurrently be active in the HASP execution phase. The specification must be an integer greater than zero.

Default: &MAXXEQS=3

Notes: See also &MAXPART, the variable which determines the number of HASP logical partitions.

&MINBUF

Explanation: This variable allows installations, which depend on the Dynamic Buffer Construction feature of HASP, to detect the condition where sufficient buffers for proper operation can not be obtained. The specification should be an integer value, representing the minimum number of buffers determined necessary for the installation (see &NUMBUF).

Default:
```
&MINBUF=4*&MAXXEQS+3*&NUMRDRS
       +2*&NUMINRS+2*&NUMPRTS+&NUMPUNS
       +2*&NUMTPBF
```

Notes:

1. HASP will automatically attempt to utilize, via variable GETMAIN, any free space in its region as additional buffers. If the number of buffers added to the variable &NUMBUF is less than the value of &MINBUF the warning message;

```
n BUFFERS AVAILABLE
```

will be issued during HASP initialization, and processing will continue.

2. Since changes in HASPGEN options, local modifications and/or OS can affect the number of HASP buffers, proper setting of this variable can prevent a possible undetected performance degradation.

3. See the description of HASPGEN parameters &NUMBUF and &RESCORE for related information.

&MINJOES

Explanation: Variable symbol &MINJOES specifies the lower bound for the number of free Job Output Elements. When the free count drops below &MINJOES, no new work will be added to the in-storage queues, until the termination of a print or punch activity raises the free count.

Default: &MINJOES=&NUMJOES/5

Notes: If the Job Output Element free count is allowed to go to zero, there will be no way to support the $I and $N operator commands for printers or punches.

&MLBFSIZ

Explanation: Variable symbol &MLBFSIZ specifies the size (in bytes) of each HASP MULTI-LEAVING buffer. The specification for &MLBFSIZ must be a positive integer no greater than &TPBFSIZ.

Default: &MLBFSIZ=400

Notes:

1. The value specified for &MLBFSIZ automatically becomes the MULTI-LEAVING buffer size in each HASP MULTI-LEAVING Remote Terminal program.

2. Satisfactory support of one device of each type (reader, printer, punch, console) on 8K terminal CPUs is based on the assumption that &MLBFSIZ is 400 or less. If the supported terminals include any 8K CPUs, it is recommended that &MLBFSIZ not be increased above 400, even if support of a nonprogrammable terminal requires increasing &TPBFSIZ to 516.

&NOPRCCW

Explanation: Variable symbol &NOPRCCW specifies the maximum number of channel command words per channel program for local printers.

Default: &NOPRCCW=30

Notes: The recommended value for this parameter is approximately &BUFSIZE/(average line length + 2) where the installation's average line length should be estimated after allowing for truncation of trailing blanks by HASP.

&NOPUCCW

Explanation: Variable symbol &NOPUCCW specifies the maximum number of channel command words per channel program for local punches.

Default: &NOPUCCW=30

Notes: The recommended value for this parameter is approximately &BUFSIZE/(average card length + 2), where the installation's average card length should be estimated after allowing for truncation of trailing blanks by HASP.

&NUMBUF

Explanation: Variable symbol &NUMBUF indicates the number of input/output buffers to be included in the HASP load module and should normally be set by each installation (according to the formulas below), to reflect the total number of buffers required for proper operation of HASP. However, since HASP will automatically utilize free space in its region to dynamically construct additional buffers, there are circumstances when &NUMBUF may be set to a value less than the actual number of buffers required for proper HASP operation. In this case, it is assumed that sufficient additional buffers will be dynamically obtained from free storage in the HASP region to provide an adequate total number of buffers (see &MINBUF and &RESCORE). This facility could be used, for example, to provide a HASP whose size (and performance and function) can be controlled by the setting of the region size.

Default: &NUMBUF=15

Notes:

1. Since all HASP buffers are maintained in a dynamic pool until required by an active function, the installation should determine the appropriate number of buffers for HASP, based on predicted simultaneity of the various functions required at the installation. The following list indicates the number of buffers required for each logical function. A defined function which is inactive requires no buffers.

   a. 3 for each local input function

   b. 2 for each Internal Reader

   c. 2 for each remote input function

   d. 2 for each local print function (1 if $PRTBOPT=1)

   e. 1 for each remote print function (2 if $RPRBOPT=2)

   f. 1 for each punch function (2 if $PUNBOPT=2)

   g. 1 for each remote punch function (2 if $RPUBOPT=2)

   h. 3 for each OS job execution (minimum value)

   For performance reasons, additional buffers must be available to sustain periods of high SYSIN/SYSOUT activity by jobs being processed by OS. It is recommended that additional buffers (beyond the requirements indicated above) be included corresponding to the value:

   ```
   1+&MAXXEQS
   ```

   Severe performance and/or device degradation can occur in a system containing insufficient buffers to perform the required functions.

3. To avoid a complete system failure caused by a buffer "lockout" condition, the number of available buffers must never be less than the value:

   ```
   2*&MAXXEQS+2*&NUMRDRS+&NUMINRS+&NUMTPRD+1
   +&NUMPRTS*($PRTBOPT-1)+&NUMPUNS*($PUNBOPT-1)
   +&NUMTPPR*($RPRBOPT-1)+&NUMTPPU*($RPUBOPT-1).
   ```

&NUMCLAS

Explanation: Variable symbol &NUMCLAS specifies the maximum number of classes for which a printer or a punch may be simultaneously started. Since there are 36 unique SYSOUT classes, the maximum allowable value of &NUMCLAS is 36.

Default: &NUMCLAS=8

&NUMDA

Explanation: Variable symbol &NUMDA specifies the maximum number of direct-access volumes that can be mounted concurrently as SPOOL volumes. The specification must be an integer greater than zero.

Default: &NUMDA=2

Notes:

1. All direct-access devices supported by OS/VS2 are eligible for use as SPOOL devices.

2. Specifying &NUMDA greater than the default may require increasing the value of &BUFSIZE.

3. During HASP initialization, if more than &NUMDA direct-access volumes whose volume serials begin with the characters SPOOL (see HASPGEN parameter &SPOOL) are mounted, the message:

   MAXIMUM OF &NUMDA SPOOL VOLUME(S) EXCEEDED

   will be written to the operator, and HASP will quiesce.

4. An associated variable is &NUMTGV.

5. Mixtures of different supported SPOOL device types are permitted.

&NUMDDT

Explanation: Variable symbol &NUMDDT specifies the number of Data Definition Tables (DDTs) to be assembled into HASP. The specification should be an integer between 3 and 256, and equal to:

2 + &MAXXEQS + A + B + R

where:

A = number of pseudo 1403 printers defined at SYSGEN time

B = number of pseudo 2540 punches defined at SYSGEN time

R = &MAXXEQS * (maximum number of DD * or DD DATA cards per job)

Default: &NUMDDT=40

Notes:

1. The pseudo-unit for &WTR should not be counted in A above.
2. If not enough DDTs are specified, a permanent lockout condition can occur.

&NUMINRS

Explanation: Variable symbol &NUMINRS specifies the number of 2520 pseudo-punches to be used by the generated HASP System as Internal Readers.

Default: &NUMINRS=0

Notes:

1. If &NUMINRS is specified as or defaulted to zero, code to support the HASP Internal Reader feature (see section on Internal Reader) will be deleted from the generated system.
2. If more than &NUMINRS 2520 pseudo-punches have been specified at SYSGEN time, only the first &NUMINRS 2520 pseudo-punches can be used. It is permissible to specify &NUMINRS greater than the number of 2520 pseudo-punches specified at SYSGEN time.
3. The count of 2520 pseudo-punches is not included in HASPGEN variable &NUMDDT.

4. Internal Readers are the only means of entering jobs from sources other than HASP supported devices. User procedures that read tape or disk data sets and punch into the Internal Reader provide an effective means of job stream entry.

&NUMJOES

Explanation: Variable symbol &NUMJOES specifies the number of Job Output Elements to be generated for queueing of output work for printers and punches.

Default: &NUMJOES=10*(&NUMPRTS+&NUMPUNS+&NUMTPPR+&NUMTPPU)

Notes:

1. This variable strongly influences the third HASP checkpoint record, whose size is:

   84+36*&NUMJOES

   If any checkpoint record is longer than the track size of the device on which the primary SPOOL volume is mounted, HASP will not operate correctly.

2. One Job Output Element is required for:

   a. Each unique class of SYSOUT that appears in a job queued for output

   b. Each active printer or punch

   c. Each interrupted or restarted job that is not currently active on a printer or punch

   d. Each unique combination of Forms ID, UCSID, FCBID, FLASH ID (only if &NUM3800>0), and BURST (only if &NUM3800>0) for all jobs currently queued for output.

   e. Each job that was interrupted by a system failure (while printing or punching) and has not yet been warm started on an output device.

3. Too small a value results in jobs waiting for in-storage queueing for completion of active print or punch work.

&NUMLNES

Explanation: Variable symbol &NUMLNES specifies the largest teleprocessing line identification number (mm in LINEmm) and thus the number of line definitions to be used by the generated HASP System. The specification must be an integer between 0 and 99, inclusive. The specification for &NUMLNES automatically becomes the specification for &NUMRJE, unless &NUMRJE is specified explicitly.

Default: &NUMLNES=0

Notes: See also the HASPGEN variable LINEmm.

&NUMOACE

Explanation: Variable symbol &NUMOACE specifies the number of overlay areas to be provided for the standard HASP Overlay feature. The specification must be an integer greater than zero.

Default: &NUMOACE=2

Notes:

1. More than two overlay areas will benefit only a system with high performance orientation (a very fast CPU or a work load consisting of a large number of short jobs).

2. See also parameter &OLAYLEV.

&NUMOSC

Explanation: Variable symbol &NUMOSC specifies the number of OS consoles HASP will support for redirection of HASP command responses. The specification must be an integer between 1 and 32, inclusive, and should reflect the number of OS consoles generated at SYSGEN time.

Default: &NUMOSC=3

&NUMPRTS

Explanation: Variable symbol &NUMPRTS specifies the maximum number of physical printers HASP may use to print job output. HASP supports 1403, 3211, and 3800 Printers. The specification must be an integer greater than zero but less than or equal to 99.

Default: &NUMPRTS=2

Notes:

1. In the case of more printers than specified in &NUMPRTS, the message:

   MAXIMUM OF &NUMPRTS PRINTER(S) EXCEEDED

   is written to the operator and HASP initialization continues normally, using only the &NUMPRTS printers with lowest unit addresses.

2. Although HASP will logically connect to supported printers (up to &NUMPRTS) during initialization, those printers that are offline (as shown by a TIO) will not be used until they are physically switched online and activated by the operator.

3. Handling of special forms by printer, the optional 1403 UCS buffer, and the 3211 UCS and forms control buffers are explained under System Output in the HASP Operator's Guide.

4. This variable influences the size of the HASP checkpoint record; see Note 2 of variable &MAXJOBS.

5. &BUFSIZE must be 536 or greater if 3211 Printers are used by HASP.

&NUMPUNS

Explanation: Variable symbol &NUMPUNS specifies the maximum number of physical punches used by HASP to punch job output. HASP supports 3525, 2540, and 2520 Card Punches. The specification must be an integer greater than or equal to zero and less than or equal to 99.

Default: &NUMPUNS=1

Notes:

1. In the case of more than &NUMPUNS 3525, 2540, and 2520 Punches, the message:

   MAXIMUM OF &NUMPUNS PUNCH(S) EXCEEDED

   is written to the operator, and HASP initialization continues normally, using only the &NUMPUNS punches with lowest unit addresses.

2. Although HASP will logically connect to supported punches (up to &NUMPUNS) during initialization, those punches that are offline (as shown by a TIO) will not be used until they are physically switched online and activated by the operator.

3. This variable influences the size of the HASP checkpoint record; see Note 2 of variable &MAXJOBS.

&NUMRDRS

Explanation: Variable symbol &NUMRDRS specifies the maximum number of physical card readers HASP may use to read OS job streams. HASP supports 3505, 2540 and 2501 Card Readers. The specification must be an integer greater than zero and less than or equal to 99.

Default: &NUMRDRS=1

Notes:

1. When more than &NUMRDRS 3505, 2540 and 2501 Card Readers are initialized, the message:

   MAXIMUM OF &NUMRDRS READER(S) EXCEEDED

   is written to the operator, and HASP initialization continues normally, using only the &NUMRDRS readers with lowest unit addresses.

2. Although HASP will logically connect to supported readers (up to &NUMRDRS) during initialization, those readers that are offline (as shown by a TIO) will not be used until they are physically switched online and activated by the operator.

&NUMRJE

Explanation: Variable symbol &NUMRJE specifies the largest remote terminal identification number (nn in RMTnn) and thus the number of remote terminal definitions to be used by the generated HASP System. The specification must be an integer between 0 and 99, inclusive.

Default: &NUMRJE=&NUMLNES

Notes:

1. See also the HASPGEN variable RMTnn.

2. If this variable is not specified, and if &NUMLNES is specified as an integer greater than zero, the first &NUMLNES remote terminal definitions (RMTnn) are used by the generated HASP System, whether they are specified explicitly or by default.

&NUMSMFB

Explanation: Variable symbol &NUMSMFB specifies the number of HASP SMF (System Management Facilities) buffers to be assembled into the generated HASP System. The specification must be an integer greater than or equal to zero.

Default: &NUMSMFB=5

Notes:

1. If &NUMSMFB is less than 2, HASP will neither produce any SMF records nor take the IEFUJP exit.

2. See also parameter &SMFRSIZ.

&NUMTGV

Explanation: Variable symbol &NUMTGV specifies the number of units (track groups) into which each SPOOL volume will be divided for HASP allocation purposes. The specification must be a positive integer no greater than the number of tracks on the SPOOL device with the fewest tracks.

Default: &NUMTGV=400

Notes:

1. The user should decide upon the number of tracks he requires in a track group and then divide by that number the total number of tracks (except alternate tracks) on a typical SPOOL device type at the installation. For example, to obtain a track group size of ten tracks on a 2314, the division would specify &NUMTGV=400. If the same installation occasionally used a 3330 as a SPOOL device, the track group size for the 3330 would automatically become 19 tracks.

2. Specifying a large &NUMTGV may require increasing the value of &BUFSIZE.

3. For each SPOOL volume it finds, HASP initialization calculates number of tracks per group by dividing the total number of tracks on the volume by &NUMTGV. It then marks as unavailable all track groups that lie partially or wholly outside the first extent of data set SYS1.HASPACE on that volume.

&NUMTPBF

Explanation: Variable symbol &NUMTPBF specifies the number of HASP teleprocessing buffers (for HASP RJE) to be assembled into the generated HASP System. The specification must be an integer greater than or equal to zero.

Default: &NUMTPBF=&NUMLNES

Notes:

1. Each signed-on HASP MULTI-LEAVING terminal requires at least two HASP teleprocessing buffers; all other signed-on terminals require at least one buffer each. If &NUMTPBF is specified too small, HASP RJE may not work correctly. (If a MULTI-LEAVING terminal has more than one output function running concurrently, additional buffers will be used to increase performance.)

2. See also parameters &TPBFSIZ and &MLBFSIZ.

&NUMTPPR

Explanation: Variable symbol &NUMTPPR specifies the maximum number of HASP remote terminal (including MULTI-LEAVING) printed output streams that can simultaneously be active. The specification must be an integer greater than or equal to zero.

Default: &NUMTPPR=&NUMLNES

Notes:

1. If any remote terminal is to receive printed output, &NUMTPPR must not be zero.

2. This variable influences the size of the HASP checkpoint record; see Note 2 of variable &MAXJOBS.

3. See also the section Multiple Devices on MULTI-LEAVING Remotes.

&NUMTPPU

Explanation: Variable symbol &NUMTPPU specifies the maximum number of HASP remote terminal (includ MULTI-LEAVING) punched output streams that can simultaneously be active. The specification must be an integer greater than or equal to zero.

Default: &NUMTPPU=&NUMLNES

Notes:

1. If any remote terminal is to receive punched output, &NUMTPPU must not be zero.

2. This variable influences the size of the HASP checkpoint record; see Note 2 of variable &MAXJOBS.

3. See also the section Multiple Devices on MULTI-LEAVING Remotes.

&NUMTPRD

Explanation: Variable symbol &NUMTPRD specifies the maximum number of HASP remote terminal (including MULTI-LEAVING) input streams that can simultaneously be active. The specification must be an integer greater than or equal to zero.

Default: &NUMTPRD=&NUMLNES

Notes:

1. If any remote terminal is to read cards, &NUMTPRD must not be zero.

2. See also the section Multiple Devices on MULTI-LEAVING Remotes.

&NUMWTOQ

Explanation: Variable symbol &NUMWTOQ specifies the number of message buffers to be provided in HASP. The specification must be an integer greater than two.

Default: &NUMWTOQ=15

Notes:

1. If Remote Job Entry is used, more message buffers are needed. This is especially true with console support for MULTI-LEAVING terminals.

2. Serious system degradation can be caused by specifying too few message buffers.

3. During periods of high console activity, when no message buffers are available, certain noncritical HASP messages will be discarded to avoid delaying the associated process. These noncritical messages include certain RJE oriented messages (such as communication line error messages), execution time/line/card excession messages (continued excession will be noted when a message buffer becomes available), and certain I/O error messages on HASP-controlled devices.

4. The number of buffers specified should be sufficient to accommodate all outstanding operator requests and still allow message processing to continue (i.e., each active local or remote output device that is waiting for forms, UCS, or carriage setup will have an outstanding request).

&OLAYLEV

Explanation: Variable symbol &OLAYLEV specifies a HASP overlay level to be used for assembly of the various HASP control sections. Any potential overlay code defined (by the $OVERLAY macro) with a residence factor greater than &OLAYLEV will be assembled as resident code rather than overlay code. The specification for &OLAYLEV must be an integer between 0 and 15, inclusive.

Default: &OLAYLEV=15

Notes:

1. HASP uses only residence factors 4, 8, 12, and (for HASP initialization only) 0.

2. If &OLAYLEV=15, all potential overlay code will be assembled as overlay code.

3. If &OLAYLEV=0, all potential overlay code, except that in HASP initialization, will be assembled as resident code. HASP main storage requirements will be increased by approximately 24K (over the case &OLAYLEV=15).

4. After the HASPGEN process is complete, regardless of &OLAYLEV, the installation system programmer may use control cards for the HASP Overlay Build Program to specify that a particular section of potential overlay code be made either resident code or overlay code. See Installing HASP In The System for more information.

&OREPSIZ

Explanation: Variable symbol &OREPSIZ specifies the size (in bytes) of a table used to hold REP data for true overlay code. The REPs associated with a particular section of true overlay code will be applied to that code every time it is brought into main storage from the HASP overlay library. The specification for &OREPSIZ must be either 0 or an integer not less than 10.

Default: &OREPSIZ=50

Notes:

1. Each entry in the HASP Overlay REP Table consists of 8+n bytes ($2 \leq n \leq 256$), where n is the number of contiguous bytes to be changed in a section of overlay code.

2. The table is used only if the operator specifies to HASP initialization that REPs are to be used and if some of the REPs are for sections of true overlay code.

3. If the HASP Overlay REP Table is too small to handle all true overlay REPs, HASP initialization writes to operator the message:

   OVERLAY REPPING ERROR

   and HASP quiesces.

&OSC(n)

Explanation: Subscripted variable symbols &OSC(n) specify OS job classes. A job selected by HASP logical partition n will be submitted to OS with the job class &OSC(n). Each specification must be a single letter between A and O, inclusive.

Default:

```
&OSC(1)=A
&OSC(2)=B
&OSC(3)=C
&OSC(4)=D
&OSC(5)=E
&OSC(6)=F
&OSC(7)=G
&OSC(8)=H
&OSC(9)=I
&OSC(10)=J
&OSC(11)=K
&OSC(12)=L
&OSC(13)=M
&OSC(14)=N
&OSC(15)=O
```

Notes:

1. Only the first &MAXPART specifications, &OSC(1) through &OSC(&MAXPART) will be used.

2. HASP initialization issues &MAXPART number of commands as follows:

```
S INIT.HOSINIT&OSC(1),,,&OSC(1)
                .
                .
                .
S INIT.HOSINIT&OSC(&MAXPART),,,&OSC(&MAXPART).
```

3. If &MAXPART is specified greater than 15, see Generating More Than Fifteen Logical Partitions.

&OSINOPT

Explanation: Variable symbol &OSINOPT specifies inclusion or exclusion of the HASP OS Input SPOOLing option. The specification must be either YES or NO. If &OSINOPT=YES and a DD * (or DD DATA) statement specifies the DCB keyword, HASP will pass the DD statement and the data following it to the OS Reader/Interpreter; OS will perform input SPOOLing. If &OSINOPT=NO, or if &OSINOPT=YES and no DCB parameter is specified on the DD * (or DD DATA) statement, HASP will SPOOL the input data as usual.

Default: &OSINOPT=NO

Notes:

1. The use of this feature may require changes in the HOSRDR procedure. Specifically, the region size, PARM field, and IEFDATA statement may need modification.

2. This feature allows the input data set to be defined as a direct-access data set, rather than as a unit record data set. This permits the data set to be concatenated more freely to other disk data sets, permits the data set to be reread (closed with a reread option), permits the data set to meet specific blocksize or record format requirements, and permits more flexible use of the checkpoint/restart facility.

&OUTPOPT

Explanation: Variable symbol &OUTPOPT specifies the action to be taken when a job exceeds its estimated print lines or punched cards of output. The specification must be one of the integers 0, 1, or 2. For &OUTPOPT=2, output excession causes the job to be cancelled with a dump. For &OUTPOPT=1, output excession causes the job to be cancelled without a dump. For &OUTPOPT=0, output excession does not cause the job to be cancelled.

Default: &OUTPOPT=0

Notes:

1. Regardless of the specification for &OUTPOPT, output excession causes messages to be written to the operator. See also Notes 1 and 2 of $OUTXS.

2. If &OUTPOPT=2 is specified, users should use SYSUDUMP or SYSABEND DD cards if a storage dump is desired on output excession.

3. For &OUTPOPT=1 or 2, the job will not be cancelled if, at the time of output excession, the job step task is normally or abnormally terminating.

$OUTXS

Explanation: Ordinary symbol $OUTXS specifies the interval, in print lines/punched cards, at which messages will be written to the operator, informing him that a job's print line count or punch card count has been exceeded. The specification must be an integer greater than zero.

Default: $OUTXS=2000

Notes:

1. The first print line excession message will be written to the operator when the job's estimated print line count has been exceeded.

2. The first punched card excession message will be written to the operator when the job's estimated punched card count has been exceeded.

&PID(n)

Explanation: Subscripted variable symbols &PID(n) specify the identifiers to be used with the HASP logical partitions. Each specification must be a unique 1- or 2-character string.

Default:
```
&PID(1)=1
&PID(2)=2
&PID(3)=3
&PID(4)=4
&PID(5)=5
&PID(6)=6
&PID(7)=7
&PID(8)=8
&PID(9)=9
&PID(10)=10
&PID(11)=11
&PID(12)=12
&PID(13)=13
&PID(14)=14
&PID(15)=15
```

Notes:

1. Only the first &MAXPART specifications, &PID(1) through &PID(&MAXPART), will be used.

2. The identifiers &PID(n) are used in messages to and commands from the operator. For example, when an operator uses the set command $T Imm,list, he is referring not to logical partition mm but to logical partition n, where &PID(n)=mm.

3. If &MAXPART is specified greater than 15, see Generating More Than Fifteen Logical Partitions.

&PRI(n)

Explanation: Subscripted variable symbols &PRI(n) specify OS job priorities. A job selected by HASP logical partition n will be submitted to OS with the job priority &PRI(n). Each specification must be an integer between 0 and 13, inclusive.

Default:
```
&PRI(1)=7
&PRI(2)=7
&PRI(3)=7
&PRI(4)=7
&PRI(5)=7
&PRI(6)=7
&PRI(7)=7
&PRI(8)=7
&PRI(9)=7
&PRI(10)=7
&PRI(11)=7
&PRI(12)=7
&PRI(13)=7
&PRI(14)=7
&PRI(15)=7
```

Notes:

1. The priorities defined by &PRI(n) affect only OS execution. The priority of a job in the HASP job queue is determined by parameters &RPRT(m), &RPRI(m), &XLIN(m), and &XPRI(m).

2. Only the first &MAXPART specifications, &PRI(1) through &PRI(&MAXPART), will be used.

3. For efficient processing of normal batch jobs, it is recommended that these values all be set equal to the priority level of the VS2 Automatic Priority Group, as specified by the APG parameter at OS SYSGEN or NIP time. However, certain HASP logical partitions may be established (with &PRI(n) values above or below that of the APG) for special processing, such as TP or low priority background work.

4. See also related parameter &APGPRTY.

5. If &MAXPART is specified greater than 15, see "Generating More Than Fifteen Logical Partitions."

$PRIDCT

Explanation: Ordinary symbol $PRIDCT specifies the number of print lines to appear on each HASP job separator page for local printers. The specification must be an integer greater than or equal to zero. If the specification is zero, no separator page will be produced on local printers.

Default: $PRIDCT=61

Notes:

1. The equivalent HASPGEN parameter for remote terminal printers is $TPIDCT.

2. If the specification is 30 or larger, the first 29 lines will be used to produce block-lettered job name, job number, and SYSOUT class.

&PRIHIGH

Explanation: Variable symbol &PRIHIGH specifies a HASP priority to be associated with the HASP Priority Aging feature. A job will not be priority-aged if its HASP priority is (or becomes) greater than or equal to &PRIHIGH. The specification must be an integer between 0 and 15, inclusive.

Default: &PRIHIGH=10

Notes:

1. If &PRIRATE=0, parameter &PRIHIGH is not used.

2. See also parameters &PRIRATE and &PRILOW.

&PRILOW

Explanation: Variable symbol &PRILOW specifies a HASP priority to be associated with the HASP Priority Aging feature. A job will not be priority-aged by HASP unless its HASP priority is initially at least &PRILOW. The specification must be an integer between 0 and 15, inclusive.

Default: &PRILOW=5

Notes:

1. If &PRIRATE=0, parameter &PRILOW is not used.

2. See also parameters &PRIRATE and &PRIHIGH.

&PRIRATE

Explanation: Variable symbol &PRIRATE specifies the amount by which a job's HASP priority will be incremented in 24 hours by the HASP Priority Aging feature. For example, if &PRIRATE=3, a job's priority will be incremented by one for every eight hours it remains in the system. But a job's priority will not be incremented unless it is at least &PRILOW; nor will a job's priority be incremented above &PRIHIGH. The specification must be an integer greater than or equal to zero. If zero is specified, Priority Aging is excluded from the generated HASP System.

Default: &PRIRATE=0

Notes:

1. If &PRIRATE=0, parameters &PRILOW and &PRIHIGH are not used.

2. See also parameters &RPRT(n), &RPRI(n), &XLIN(n), and &XPRI(n).

3. If a job's priority is specified on the /*PRIORITY control card, the job will be priority-aged if its priority is eligible.

$PRTBOPT

Explanation: Ordinary symbol $PRTBOPT specifies the printer buffering option to be used for local HASP printers. The specification must be either 1 (for single buffering) or 2 (for double buffering).

Default: $PRTBOPT=2

&PRTFCB

Explanation: Variable symbol &PRTFCB specifies the 1- to 4-character name of the forms buffer image or of the carriage control tape, which HASP initially assumes is mounted on every local printer. The FCB identifier can be modified by the operator for each individual printer.

Default: &PRTFCB=6

&PRTRANS

Explanation: Variable symbol &PRTRANS specifies translation for lines of print not directed to 3211 Printers. The specification must be either YES or NO.

Default: &PRTRANS=YES

Notes:

1. If &PRTRANS is specified as YES, each line to be printed by a local 1403 or any remote printer is first translated. Translation changes lower-case letters to upper-case letters and characters that are invalid on a PN train to blanks.

2. If any print train is to be used on a HASP-controlled local 1403 or remote printer that has characters not equivalent to those on a PN train, &PRTRANS must be specified as NO.

3. If all printers are 3211s (no 1403s or remotes), &PRTRANS should be specified as NO.

&PRTUCS

Explanation: Variable symbol &PRTUCS specifies the name of the print chain or print train which HASP initially assumes is mounted on every local 1403 Printer SYSGENed with the UCS feature and on every local 3211 Printer. The UCS identifier can be modified by the operator individually by printer. The specification should be either AN, HN, PN, QN, RN, UN, A11, H11, P11, U11 or 0.

Default: &PRTUCS=0

Notes:

1. A specification of zero causes HASP to bypass the UCS loading procedure on all local printers until the UCS type of each printer is specified by the operator.

2. If a UCS specification that is not valid for the type of printer being addressed is encountered, the USC loading procedure will be bypassed (with an operator warning message).

3. The UN and U11 specifications are provided for installation use to support other types of print chains.

$PUNBOPT

Explanation: Ordinary symbol $PUNBOPT specifies the punch buffering option to be used for local HASP punches. The specification must be either 1 (for single buffering) or 2 (for double buffering).

Default: $PUNBOPT=1

&RDR

Explanation: Variable symbol &RDR specifies the unit address of a pseudo 2540 reader to be used with HOSRDR to supply jobs to OS. The specification must be a valid unit address, specified at SYSGEN time as a pseudo 2540 reader.

Default: &RDR=0FC

Notes: The unit address assigned to this parameter must not be assigned a symbolic unit name at SYSGEN time, as described for other pseudo 2540 readers.

$REPRDR

Explanation: Ordinary symbol $REPRDR specifies the unit address of a physical 3505, 2540 or 2501 Card Reader, from which HASP initialization will read REP cards, if requested by the operator. The specification must be a valid unit address.

Default: $REPRDR=00C

$REPWTR

Explanation: Ordinary symbol $REPWTR specifies the unit address of a physical 1403 or 3211 on which each REP card read is to be printed, if printing of REP cards is requested by the operator. The specification must be a valid unit address.

Default: $REPWTR=00E

&RESCORE

Explanation: Variable symbol &RESCORE specifies storage size, in multiples of 1024 bytes. HASP will always issue a GETMAIN for additional storage; all additional storage (but &RESCORE*1024 bytes) will be used for HASP buffers.

The specification must be an integer greater than or equal to zero.

Default: &RESCORE=0

&RJOBOPT

Explanation: Variable symbol &RJOBOPT specifies the type of scan that should be performed on job cards which are processed by the HASP Input Processor and specifies whether an illegal JOB card will prevent execution of the associated job.

The specification must be an integer between 0 and 5, inclusive, where the specified values have the following meanings:

Value	Scan HASP Parameters	Terminate On HASP Parameter Error	Terminate On OS Format Error
0	YES	YES	YES
1	YES	YES	NO
2	YES	NO	YES
3	YES	NO	NO
4	NO	--	YES
5	NO	--	NO

Default: &RJOBOPT=2

Notes:

1. The scan of JOB card parameters CLASS, MSGCLASS, and TYPRUN=HOLD is always performed.

2. The only HASP JOB card parameters scanned are those included in the JOB Card Accounting Field as defined in the HASP Operator's Guide.

3. A HASP parameter error is any violation of the requirements of the HASP JOB card parameters as defined in the HASP Operator's Guide.

4. An OS format error is any error which prevents HASP from continuing the scan of the JOB statement as defined in the OS/VS Job Control Language Reference manual (e.g., illegal continuation, illegal characters, missing parentheses).

RMTnn

Explanation: Ordinary symbols RMTnn specify the characteristics of remote terminals to be used with HASP Remote Job Entry. Terminals must be defined consecutively, starting with RMT01. Each specification must be a 14-character string of the form:

RMTnn=mmooppiillwtdf

where the letters represent the following:

Code Letters	Range	Description
nn	01-99	Remote number
mm	01-99	Line number (** indicates /*SIGNON assignment)
oo	00-99	Print routing (remote number)
pp	00-99	Punch routing (remote number)
ii	00-15	Priority increment for this remote
ll	00-15	Priority limit for this remote
w	0-6	Printer width as follows: 0 = 80 characters 1 = 100 characters 2 = 120 characters 3 = 132 characters 4 = 144 characters 5 = 150 characters 6 = 96 characters
t	0-7	Terminal type as follows: 0 = 2770 1 = 2780 2 = 2922, System 360/20 Submodel 2, 4, 6 3 = System 360/20 Submodel 5 4 = System 360/22, 25, 30, 40, etc. 5 = 1130 6 = System/3 7 = 3780

d 0-4 Data Format as follows:

0 = Unblocked fixed length
1 = Blocked fixed length
2 = Unblocked variable length (Note - use this for basic 2770 terminals.)
3 = Blocked variable length (Note - use this for all 3780 and 2780 terminals and for 2770 terminals with Buffer Expansion.)
4 = Programmable interface (Note - use this for all BSC MULTI-LEAVING interfaces.)

f Terminal features as follows:

0-9 3780 Terminal features

f	Compress Expand	Horizontal Format Control	Trans-parency
0	No	No	No
1	No	No	Yes
4	No	Yes	No
5	No	Yes	Yes
8	Yes	No	No
9	Yes	No	Yes

0-a 2770 Terminal features

f	Compress Expand	Horizontal Format Control	Additional Buffer Expansion	Trans-parency
0	No	No	No	No
1	No	No	No	Yes
2	No	No	Yes	No
3	No	No	Yes	Yes
4	No	Yes	No	No
5	No	Yes	No	Yes
6	No	Yes	Yes	No
7	No	Yes	Yes	Yes
8	Yes	No	No	No
9	Yes	No	No	Yes
#	Yes	No	Yes	No
a	Yes	No	Yes	Yes

0-7 2780 Terminal features

f	Horizontal Format Control	Multiple Record Feature	Transparency
0	No	No	No
1	No	No	Yes
2	No	Yes	No
3	No	Yes	Yes
4	Yes	No	No
5	Yes	No	Yes
6	Yes	Yes	No
7	Yes	Yes	Yes

0-3 MULTI-LEAVING Terminal features

f	Console Support	Transparency
0	No	No
1	No	Yes
2	Yes	No
3	Yes	Yes

Default: RMTnn=**nn0000153131

Notes:

1. Parameter &NUMRJE must specify the number of specifications (RMTnn) to be included in the generated HASP System.

2. No two specifications (RMTnn) may specify the same line number (mm). If ** is specified instead of a line number for mm, the associated remote terminal may connect to HASP via any suitable line. HASP will logically connect the terminal with the line when it recognizes the /*SIGNON control card. If line number is specified explicitly, the associated terminal need not use a /*SIGNON card.

3. The line number specification mm refers to line specification LINEmm, which in turn specifies the unit address of the line.

4. For print and punch routing, a specification of 00 causes output from jobs submitted at the remote terminal to be printed/punched locally, unless rerouted.

5. Priority increment is the value to be added to the priority of a job submitted from the remote terminal.

6. Priority limit is the maximum value of priority for any job submitted from the remote terminal.

7. If any MULTI-LEAVING work station is to use more than one reader, printer, or punch, see Multiple Devices on MULTI-LEAVING Remotes for additional information.

8. For a basic 2770 (128-byte buffers), printer width must be specified as 120 characters or less.

9. Printed and nontransparent punched output to a basic 2770 will be variable-blocked up to the limit of the 128-byte buffers, even though the data format must be specified as variable-unblocked.

10. The following table gives, for &TPBFSIZ, minimum values required to support various nonprogrammable terminals:

Minimum &TPBFSIZ	Terminal Type, Features
128	2770 basic
260	2770 with Buffer Expansion
400	2780
516	2770 with Buffer Expansion and Additional Buffer Expansion
516	3780

See also parameters &TPBFSIZ and &MLBFSIZ.

11. The 3780 plus the 3781 punch must be generated as a 2770 with appropriate features in order to get output device component selection. For example, the last three characters of the RMTnn parameter would be 739 for a 3780 alone with transparency and 03@ for a 3780 plus 3781 punch with transparency.

$RPRBOPT

Explanation: Ordinary symbol $RPRBOPT specifies the printer buffering option to be used for all printers at HASP remote terminals. The specification must be either 1 (for single buffering) or 2 (for double buffering).

Default: $RPRBOPT=1

Notes: The specification refers to HASP regular buffers, not to HASP teleprocessing buffers.

&RPRI(n)

Explanation: Subscripted variable symbols &RPRI(n) specify job priorities corresponding to intervals defined by subscripted variable symbols &RPRT(n). If a user does not supply a /*PRIORITY control card with his job, the queueing priority is computed as defined in Note 1. Each RPRI(n) specification must be an integer between 0 and 15, inclusive.

Default:
```
&RPRI(1)=9
&RPRI(2)=8
&RPRI(3)=7
&RPRI(4)=6
&RPRI(5)=5
&RPRI(6)=4
&RPRI(7)=3
&RPRI(8)=2
&RPRI(9)=1
```

Notes:

1. The queueing priority is computed as:

   priority=[&RPRI(n)+&XPRI(m)]/2

   The subscript n is the smallest number for which:

   t≤&RPRT(n)

   The subscript m is the smallest number for which:

   o≤&XLIN(m)

   where t is the estimated execution time from the accounting field of the JOB card or from the /*JOBPARM control card, and o is the sum of the estimated output lines and cards from the accounting field of the JOB card or from the /*JOBPARM control card.

2. See the description of &RPRT(n).

3. See also the description of &XPRI(m).

&RPRT(n)

Explanation: Subscripted variable symbols &RPRT(n) specify estimated execution times, which are associated with the priorities &RPRI(n). Each specification must be an integer between 1 and X'FFFFFF'/60, inclusive.

Default:
```
&RPRT(1)=2
&RPRT(2)=5
&RPRT(3)=15
&RPRT(4)=X'FFFFFF'/60
&RPRT(5)=X'FFFFFF'/60
&RPRT(6)=X'FFFFFF'/60
&RPRT(7)=X'FFFFFF'/60
&RPRT(8)=X'FFFFFF'/60
&RPRT(9)=X'FFFFFF'/60
```

Notes:

1. See the description of &RPRI(n).

2. See also the description of &XPRI(m).

3. These values are not used if the job uses a /*PRIORITY HASP control card.

&RPS

Explanation: Variable symbol &RPS specifies inclusion or exclusion of Rotational Position Sensing for all HASP channel programs directed to direct-access devices with the Rotational Position Sensing feature. The specification must be either YES or NO.

Default: &RPS=NO

Notes: Regardless of the setting of &RPS, HASP will correctly operate with any supported direct-access device or combination of devices.

$RPUBOPT

Explanation: Ordinary symbol $RPUBOPT specifies the punch buffering option to be used for all punches at HASP remote terminals. The specification must be either 1 (for single buffering) or 2 (for double buffering).

Default: $RPUBOPT=1

Notes: The specification refers to HASP regular buffers, not to HASP teleprocessing buffers.

&SMFRSIZ

Explanation: Variable symbol &SMFRSIZ specifies the size of the largest SMF record to be written by HASP or the size of one SMF common exit parameter area, whichever is greater. If SMF records are to be written by HASP, the specification must be an integer equal to or greater than 228.

Default: &SMFRSIZ=228

Notes:

1. If &NUMSMFB is less than 2, &SMFRSIZ will be ignored. Otherwise, HASP will generate SMF records and &SMFRSIZ will be the maximum size SMF record written by HASP.

2. See also parameter &NUMSMFB.

&SPOLMSG

Explanation: Variable symbol &SPOLMSG specifies the number of physical records (in the first extent of SYS1.HASPACE on the primary SPOOL volume) to be reserved for operator messages and HASP messages for HASP remote terminals. Each physical record is capable of holding one or more messages for a single remote terminal. Messages are held if they are directed to:

1. Any terminal not signed on
2. Any signed-on hardware terminal that is currently processing an input or output stream
3. Any signed-on computer terminal that is not a MULTI-LEAVING terminal with a console.

If a message is to be held but no space is available to hold it, the message is thrown away without operator notification.

The specification for &SPOLMSG must be an integer greater than or equal to zero. If &SPOLMSG is specified as zero, no messages will be sent to hardware terminals.

Default: &SPOLMSG=10*&NUMRJE

Notes:

1. Only the $DM command can generate messages to a terminal that is not signed on.
2. For signed-on terminals, messages are generated for job-on-reader, by $DM, and as responses to commands from the terminal.
3. Each message to a terminal (except to a MULTI-LEAVING remote defined with a console) is held until it can be printed, or until HASP is restarted.

&SPOOL

Explanation: Variable symbol &SPOOL specifies the first five characters of the volume serial of each direct-access volume mounted for SPOOLing by HASP. The specification must be exactly five characters and must be valid as a volume serial.

Default: &SPOOL=SPOOL

Notes:

1. If this variable is changed from its default, certain HASP messages will vary from their documentation in the HASP Operator's Guide.

2. With the default, HASP requires that at least the primary SPOOL volume (normally SPOOL1) be mounted. If, for example, &SPOOL=$-#-@, HASP would require at least $-#-@1 be mounted.

&STDFORM

Explanation: Variable symbol &STDFORM specifies a 4-character alphameric value, which will be used as a Forms ID when no Forms ID is given. Also, &STDFORM specifies the initial setup of all printers and punches at HASP startup.

Default: &STDFORM=STD.

&TIMEOPT

Explanation: Variable symbol &TIMEOPT specifies the action to be taken when a job's estimated execution time is exceeded. The specification must be one of the integers 0, 1, 2, or 4. For &TIMEOPT=4, the job's time limits will not be monitored. For &TIMEOPT=2, time excession causes the job to be cancelled with a dump. For &TIMEOPT=1, time excession causes the job to be cancelled without a dump. For &TIMEOPT=2, &TIMEOPT=1, or &TIMEOPT=0, time excession causes messages to be written to the operator.

Default: &TIMEOPT=4

Notes:

1. See also HASPGEN parameter $ESTIME, which applies for &TIMEOPT=0, &TIMEOPT=1, and &TIMEOPT=2.

2. For &TIMEOPT=1 or 2, the job will not be cancelled if, when time is exceeded, the job step task is normally or abnormally terminating.

$TIMEXS

Explanation: Ordinary symbol $TIMEXS specifies the interval, in minutes, at which messages will be written to the operator informing him that a job's execution time is exceeded. The specification must be an integer greater than zero.

Default: $TIMEXS=1

Notes:

1. The first time excession message is written to the operator when the job's estimated execution time has been exceeded.

2. If &TIMEOPT is specified greater than 2, $TIMEXS is not used.

&TPBFSIZ

Explanation: Variable symbol &TPBFSIZ specifies the size in bytes of each HASP teleprocessing buffer. The specification must be a positive integer less than or equal to 32k.

Default: &TPBFSIZ=400

Notes:

1. The value of &TPBFSIZ is the maximum size of any HASP teleprocessing buffer. See also parameter &MLBFSIZ, which may never be specified larger than &TPBFSIZ.

2. HASP remote terminal programs are MULTI-LEAVING programs and use &MLBFSIZ.

3. The parameter &TPBFSIZ is specified only once, at HASPGEN time; it is conveyed automatically to the requisite remote terminal programs by HASPGEN.

4. See RMTnn for minimum &TPBFSIZ required when HASP supports nonprogrammable terminals.

$TPIDCT

Explanation: Ordinary symbol $TPIDCT specifies the number of print lines to appear on each HASP job separator page for printed output directed to any HASP remote terminal. The specification must be an integer greater than or equal to zero. If the specification is zero, no separator page will be produced on remote printers.

Default: $TPIDCT=6

Notes:

1. The equivalent HASPGEN parameter for local printers is $PRIDCT.

2. If the specification is 30 or larger, the first 29 lines will be used to produce block-lettered job name, job number, and SYSOUT class.

&TRACE

Explanation: Variable symbol &TRACE specifies inclusion or exclusion of a facility for event-tracing in the generated HASP System. It also specifies the number of entries to be generated in the HASP trace table. The specification must be an integer greater than or equal to zero.

Default: &TRACE=0

Notes:

1. The HASP trace facility is a development tool only which causes system degradation and causes the OS program interrupt exit (SPIE) mechanism to work incorrectly. For these reasons, the HASP Trace should not be included in any generated HASP System designed for normal production.

2. The &TRACE option is independent of the &DEBUG option.

3. See the HASP Logic manual for a description of trace table entries generated by this feature.

&TSOSTCN

Explanation: Variable symbol &TSOSTCN specifies inclusion or exclusion of support for STATUS and CANCEL commands related to jobs submitted by TSO users through the HASP Internal Reader. The specification must be YES or NO.

Default: &TSOSTCN=NO

Note: If this variable is set to NO, any attempt to find a job queued by HASP with a STATUS or CANCEL command will receive a JOB NOT FOUND response.

&USASCII

Explanation: Variable symbol &USASCII specifies inclusion or exclusion in the HASP Remote Terminal Access Method, of the capability to use USASCII line-control characters, as well as EBCDIC line-control characters. If any line specification LINEmm for a BSC line has value c set to 2, 3, 6, or 7, &USASCII should be set to YES; otherwise, &USASCII should be set to NO.

Default: &USASCII=NO

Notes: See Note 3 for LINEmm.

$WAITIME

Explanation: Ordinary symbol $WAITIME specifies a time interval, in seconds. For hardware terminals, the HASP Remote Terminal Access Method will wait $WAITIME seconds, at the completion of processing of any input stream, printed output stream, or punched output stream, to allow the operator to alter the normal sequence of Remote Job Entry operations. For example, the operator may wish to transmit another job to HASP after a previous job has finished printing, rather than wait until all queued output has finished processing.

The specification for $WAITIME must be an integer greater than zero.

Default: $WAITIME=1

&WCLSREQ

Explanation: Variable symbol &WCLSREQ specifies optional requeueing for OS output classes specified by &WTRCLAS.

The HASP writer subtask (load module HASPWTR) processes jobs queued in the OS output queues defined by &WTRCLAS. After processing a job whose output class is the nth character of &WTRCLAS, HASPWTR examines the nth character of &WCLSREQ. If the nth character of &WCLSREQ is *, HASPWTR deletes the job from the OS job queue. But if the nth character of &WCLSREQ is an OS output class, HASPWTR requeues the job in the OS output queue specified by the nth character of &WCLSREQ (which must be different from any class specified in &WTRCLAS).

The specification must be a string of one to eight characters each of which is either * or a unique valid OS output class different from any specified in &WTRCLAS. If more characters are specified than were specified for &WTRCLAS, the excess characters are unused.

Default: &WCLSREQ=********

Notes:

1. The output requeueing option is useful for providing an extra copy of a job's system messages to, for example, a conversational programming terminal.

2. A requeued job is not referenced by HASP but must be accessed by a standard OS output writer or other suitable means.

3. A requeued job may contain a mixture of system messages and SYSOUT data sets of the same class, if the SYSOUT data sets were SPOOLed by OS (see HASPGEN parameter $$x). The module HASPWTR does not process the SYSOUT data sets but requeues the entire job containing them in the new class specified by &WCLSREQ. The system messages and SYSOUT data sets are then available to a standard OS output writer, which is processing the new class.

4. Any DD statements in the system messages of a requeued job, which are originally coded as DD * or DD DATA and are not subject to OS SPOOLing (see HASPGEN parameter &OSINOPT), are available to a writer processing a &WCLSREQ class as DD $ and DD CATA, respectively. They are printed as DD $ and DD CATA, unless the writer is programmed to change them to their original form.

&WTR

Explanation: Variable symbol &WTR specifies the unit address of a pseudo 1403 printer to be used by a writer to retrieve, from the OS job queue, System Message Blocks (SMBs) for jobs controlled by HASP. The specification must be a valid unit address, which has been specified at SYSGEN time as a pseudo 1403 printer.

Default: &WTR=0FE

Notes: The unit address assigned to this parameter must not be assigned a symbolic unit name at SYSGEN time, as described for other pseudo 1403 printers.

&WTRCLAS

Explanation: Variable symbol &WTRCLAS specifies the OS system output classes to be processed by HASP. The HASPWTR is intended to process only those System Message Blocks (SMBs) created by OS jobs submitted to and controlled by HASP. If other OS writers are to be used concurrently with the HASP writer, none of them may process any of the output classes specified in &WTRCLAS.

The specification for &WTRCLAS must be one to eight unique characters, each of which is a valid OS output class different from any specified in &WCLSREQ.

Default: &WTRCLAS=A

Notes:

1. HASP examines the MSGCLASS parameter of every JOB card it sends to OS. If MSGCLASS is not specified or is not one of the classes specified by &WTRCLAS, HASP adds a MSGCLASS parameter (the leftmost character of &WTRCLAS) to the JOB card.

2. If a job submitted to OS by HASP has certain errors on the JOB card, OS will fail the job and change its MSGCLASS to a default class specified in the parameter field of the HOSRDR procedure. Therefore, it is required that this default class be specified in &WTRCLAS. If this class is not specified and such an error happens, HASP will not operate correctly.

3. See also HASPGEN parameter $$x.

&XBATCHC

Explanation: Variable symbol &XBATCHC specifies a list of job classes to be used with the Execution Batch Scheduling feature. The specified classes are excluded from running jobs outside Execution Batch Scheduling. The specification for &XBATCHC is a string of one to eight characters (letters and numbers) that specify valid unique HASP job classes. If &XBATCHC is left at its default, the generated HASP System will not include Execution Batch Scheduling.

Default: &XBATCHC=[null string]

Notes:

1. For further information, see Execution Batch Scheduling.

2. If &XBATCHC is not specified, then &XBATCHN is not used.

&XBATCHN

Explanation: Variable symbol &XBATCHN specifies the first five characters of the name of each OS job to be started internally by HASP, when required for the execution of a user job under the Execution Batch Scheduling feature. The specification must be a five-character string, of which the first character is alphabetic or national and the remaining four are alphameric or national.

Default: &XBATCHN=$$$$$

Notes:

1. For further information, see Execution Batch Scheduling.

2. If &XBATCHC is specified, HASP will reject all user submitted jobs whose jobnames start with the five characters in &XBATCHN.

&XLIN(m)

Explanation: Subscripted variable symbols &XLIN(m) specify output record counts, which are associated with the priorities &XPRI(m). Each specification must be an integer between 1 and 16,777,215.

Default:
```
&XLIN(1)=2000
&XLIN(2)=5000
&XLIN(3)=15000
&XLIN(4)=X'FFFFFF'
&XLIN(5)=X'FFFFFF'
&XLIN(6)=X'FFFFFF'
&XLIN(7)=X'FFFFFF'
&XLIN(8)=X'FFFFFF'
&XLIN(9)=X'FFFFFF'
```

Notes:

1. See the description of &XPRI(m).

2. See also the description of &RPRT(n).

3. These values are not used if the job uses a /*PRIORITY HASP control card.

&XPRI(m)

Explanation: Subscripted variable symbols &XPRI(m) specify output priorities that correspond to intervals defined by subscripted variable symbols &XLIN(m). If a user does not supply a /*PRIORITY control card with his job, the job's priority is recomputed after execution, based on the actual number of print and punch records it produced. If the job produced p print lines and c punched cards, its output priority will become &XPRI(m), where m is the smallest number for which:

$$p+c \leq \&XLIN(m)$$

Each specification must be an integer between 0 and 15.

Default:
&XPRI(1)=9
&XPRI(2)=8
&XPRI(3)=7
&XPRI(4)=6
&XPRI(5)=5
&XPRI(6)=4
&XPRI(7)=3
&XPRI(8)=2
&XPRI(9)=1

Notes:

1. See the description of &XLIN(m).

2. See also the description of &RPRT(n).

$$x

Explanation: Ordinary symbol $$x specifies the destination for an output data set designated in the user's JCL as SYSOUT=x. The specification for each of these ordinary symbols must be one of the characters A, B, or *. These characters indicate:

1. $$x=A - associated SYSOUT data sets will normally be printed.

2. $$x=B - associated SYSOUT data sets will normally be punched.

3. $$x=* - associated SYSOUT data sets will be processed entirely by OS. In this case, HASP will add the specification UNIT=SYSDA to the JCL, unless the user has himself specified UNIT= information.

Default:

$$A=A	$$S=A
$$B=B	$$T=A
$$C=A	$$U=A
$$D=A	$$V=A
$$E=A	$$W=A
$$F=A	$$X=A
$$G=A	$$Y=A
$$H=A	$$Z=A
$$I=A	$$0=A
$$J=A	$$1=A
$$K=B	$$2=A
$$L=A	$$3=A
$$M=A	$$4=A
$$N=A	$$5=A
$$O=A	$$6=A
$$P=A	$$7=A
$$Q=A	$$8=A
$$R=A	$$9=A

Notes:

1. For any output class x, regardless of the value specified for $$x, a four-character special forms designation can be coded as the third positional parameter of the SYSOUT= keyword. See /*OUTPUT control card under System Input in the HASP Operator's Guide.

2. A user SYSOUT specification that includes the second positional parameter (program name) will be processed entirely by OS, regardless of whether the associated $$x parameter was specified as *.

3. If a given output class x is one of the classes assigned to &WTRCLAS, it must not be used in a SYSOUT specification to be processed by OS (caused if $$x=* or if the second parameter of SYSOUT is used), unless that class is subject to requeuing, as described under the parameter &WCLSREQ.

IBM 3800 HASPGEN PARAMETERS

Support for the IBM 3800 Printing Subsystem is controlled by the HASPGEN parameter &NUM3800. (For other planning and system programming information on the 3800, see the IBM 3800 Printing Subsystem Programmer's Guide for OS/VS2 SVS.) If &NUM3800 is specified with a non-zero value, you can specify several other HASPGEN parameters that are specifically for the 3800. There are also things you should consider when specifying other HASPGEN parameters. The parameters not specifically for the 3800 that you should consider are:

- &BUFSIZE - This value should be as large as possible (for example, 4008 bytes on a 3330). Choice of a small value for &BUFSIZE could cause degraded performance of a 3800.

- &DMNDSET - This option should not be used with the 3800.

- $LINECT - The 3800 does not allow printing within 1/2 inch of either end of the form. For example, the maximum number of lines that can be printed on an 11-inch long form at 6 lines per inch is 60. $LINECT should be chosen with this in mind.

- &NOPRCCW - This value should take into account the larger buffer size chosen for the 3800.

- &NUMPRTS - The number of printers should include the 3800.

- $PRIDCT - The comment that applies to the choice of $LINECT also applies to $PRIDCT.

FCBs for the 3800 Printing Subsystem

FCBs for the 3800 can be generated in HASP like FCBs for the 3211 Printer. Additionally, HASP supports 3800 FCBs located in SYS1.IMAGELIB. When an FCB is requested, the local FCB table is checked first, then SYS1.IMAGELIB is searched. The variable FCB is not supported for the 3800.

FCBs for the 3800 and the 3211 printers are not compatible. If output is to be redirected from one of these printers to the other, you must ensure that corresponding FCBs exist.

In addition to the IBM-supplied FCBs in SYS1.IMAGELIB, HASP supplies local FCBs named "6" and "8". These correspond to FCBs "6" and "8" for the 3211, except for the position of channels 9 and 12.

The HASPGEN parameters that are specifically 3800-related are the following:

&NUM3800

Explanation: Variable symbol &NUM3800 specifies the number of 3800 printers to be attached to the HASP system.

Default: &NUM3800=0

&HDWCHAR

Explanation: Variable symbol &HDWCHAR specifies the name of the character arrangement table to be assumed as the hardware default when only one character arrangement table is specified.

Default: &HDWCHAR='GF10'

Notes:
1. The 3800's hardware default and GF10 are not exactly the same. The hardware default includes the lozenge as a valid character, while GF10 does not.

2. At least one character arrangement table is always specified by the &JXLATn parameter, which defaults to &HDWCHAR if it is not specified.

&HDWFCB

Explanation: Variable symbol &HDWFCB specifies the FCB name to be used for the hardware default FCB on the 3800.

Default: &HDWFCB=&PRTFCB

Notes: If the standard defaults are taken, &HDWFCB defaults to the FCB named "6". While the hardware default FCB does print at 6 lines per inch, only channel 1 is defined.

&JBURST

Explanation: Variable symbol &JBURST specifies the default Burster-Trimmer-Stacker paper threading assumed for each 3800 printer. The specification must be either Y (for bursting) or N (for no bursting).

Default: &JBURST=N

Notes: HASP never checks the actual Burster-Trimmer-Stacker threading, but assumes the threading to be that specified by &JBURST or reset by the operator using the $T PRT command.

&JFLASH

Explanation: Variable symbol &JFLASH specifies the name of the default forms overlay negative assumed to be loaded in the 3800 and assumed to be requested by output data sets that do not specify FLASH.

Default: &JFLASH='   ' (four blanks, indicating that there is to be no forms overlay)

&JMARK

Explanation: Variable symbol &JMARK specifies that the perforations between separator pages at the end of each output data set should or should not be marked by the 3800 printer. The specification must be either Y (for marking) or N (for no marking).

Default: &JMARK=N

Notes:

1. The 3800 does not allow printing on the perforations between pages, other than the marks obtained using the Mark Form channel command.

2. This parameter can be overridden at execution time by the operator's use of the $T PRTn,M=Y|N command.

3. Forms marking only takes place when separator pages are being used. The end separator is marked; the start separator is not.

4. The 3800 produces three copies of the marked separator page, or, if either of the two shortest forms lengths is being used, five copies.

&JXLATn

Explanation: The variable symbols &JXLAT1 through &JXLAT4 specify the character arrangement tables to be loaded into the 3800 for jobs that do not specify the CHARS parameter on the SYSOUT DD card or the /* OUTPUT card. Each specification must be the name of a character arrangement table stored in SYS1.IMAGELIB (for the names of the IBM-supplied tables, see Appendix B of the IBM 3800 Printing Subsystem Programmer's Guide for OS/VS2 SVS).

Default: &JXLAT1-'&HDWCHAR'
&JXLAT2='   ' Note: Four blanks, for the 2nd,
&JXLAT3='   ' 3rd, and 4th tables.
&JXLAT4='   '

INSTALLING HASP IN THE SYSTEM

To install HASP, it is necessary to perform some or all of the following four processes, after HASPGEN has been completed. Four sample jobs, one for each process, are printed and punched from the source member HASPJCL when HASPGEN is performed as described previously. These jobs are also listed in Reference Listing of HASPJCL.

It must be emphasized that the sample jobs are just samples. If run exactly as punched, they will probably produce incorrect results. Each process is discussed below with comments about what modifications to the sample job may be necessary.

INSTALLING HASP SVC

Installation of the HASP SVC is necessary upon completion of the OS system generation. The "Sample Job HASPSVC" (shown in the "Reference Listing of HASPJCL" section of this manual) may be used.

The HASPSVC job link edits the standard Nucleus with the HASP SVC and replaces the Nucleus in the standard SYS1.NUCLEUS data set.

Alternative procedures may be used to install the HASP SVC, including use of alternate members within the single data set SYS1.NUCLEUS, if space permits. Naming of these members and IPL procedures are described in appropriate OS documentation.

INSTALLING HASP PROCS

The sample job HASPROCS should be used to add necessary cataloged procedures (members) to the system's SYS1.PROCLIB data set. The members are described in the following subsections.

Member HASP

HASP is invoked when the operator types the OS START command, as described in the HASP Operator's Guide.

The load module name must be HASP, as created by the HASPHASP job. The OLAYLIB DD statement must refer to the library created by the HASPHASP job (see Installing HASP Program).

The region size required is determined partially by the address space requirement discussed under Storage Requirements and partially by

rounding caused by LINKEDIT (see HASPHASP job under Installing HASP Program), VS2 4K multiple subpools, and VS2 64K multiple regions.

HASP may be executed in a V=R region, but this is neither required nor recommended.

Member HOSRDR

The member HOSRDR is used by HASP to invoke the single OS Reader/Interpreter necessary to send jobs to OS for execution.

The DCB field of the IEFRDER DD statement must not be modified. The IEFDATA statement may be modified to fit installation requirements, but this will have effect only if the HASPGEN parameter &OSINOPT=YES and a DD * or DD DATA card with DCB parameters is encountered in an input stream read by HASP.

INSTALLING HASP PROGRAM

The HASP program consists of one load module, made up of resident CSECTs from each of 11 object modules, and several overlay CSECTs taken from some of these object modules. During HASP operation, each overlay CSECT exists as a single record in a sequential data set on a direct-access device.

The three step sample job HASPHASP shows how the above components of the HASP Program are constructed from the object decks produced by HASPGEN. The first step simply scratches the overlay data set to be later allocated and built.

The second step executes a utility called HASPOBLD whose primary input is 11 object modules from SYS1.HASPOBJ as shown. The overlay CSECTs are written to SYS1.HASPOLIB and all references to them (in other overlays or in resident CSECTs) are resolved. Resident CSECTs are written to the SYSLIN DD temporary data set as input to the third step.

The third step uses the OS Linkage Editor to resolve all external references between resident CSECTs and produce the load module, HASP. Note that the load module and the overlay data set produced by this job belong together and should be invoked as a single entity by the proclib member HASP. Load modules must not be used with overlay data sets produced by different executions of this job.

All uses of ZZZZZZ in the sample job as a volume label should be changed to the volume of the overlay data set, which may be any direct-access volume including the SPOOL volumes. The data set should be considered a high activity system data set, just like SYS1.LINKLIB, and should be placed accordingly for optimum performance. Space allocation must be a single extent and must be less than 128 tracks. The example shows space

for seventy 1280 byte records, sufficient for an unmodified HASP System with HASPGEN parameter &OLAYLEV set for maximum overlay. After the overlay data set is originally written by HASPOBLD, it may be moved, but only to another volume of the same device type.

Any CSECT which is programmed for overlay (third character of name is a "$") may be changed from resident to overlay or vice versa during execution of HASPOBLD, by reading control cards from the SYSIN DD file (shown as empty in the sample). The CSECT name is punched in column 1 of a control card, beginning with "HA$". The fourth character is punched "O" to make the CSECT overlay, or "P" to make it resident. Fifth and following characters are taken from the CSECT name as given in the appropriate assembly external symbol listing. If a CSECT is being made overlay, a priority number in the range 0-15 may be punched, beginning in column 16, to change the priority.

An information listing is produced by HASPOBLD. Any control cards are listed first. Then each "HA$" CSECT name is listed, with its length and OCON (or relative position), in the HASP Overlay Table (OTB). For actual overlay CSECTs the relative and absolute record address is given, as well as the priority for use of overlay resources. The CCHHR is especially useful when using IMASPZAP to inspect or change a particular overlay CSECT on a direct-access device.

Self-explanatory error messages, for example, "TOO LONG," "DUPLICATE," or "UNDEFINED," may be produced with any listed CSECT name. They should not occur unless erroneous user modifications to HASP have been made. CSECTs of greater than 1280 bytes may be made resident by use of a control card described previously; otherwise, the CSECT is truncated to 1280 bytes. In the case of duplicate CSECTs, the first copy encountered in HASPOBLD input is used.

Following a normal HASPOBLD, completion code 0 is returned. If duplicate CSECTs were encountered, the completion code will be 4, and if any overlays were too long or undefined, a completion code of 8 will be returned. Sample job HASPHASP prevents the link edit step if a completion code greater than 4 is returned.

If object module input to HASPOBLD causes overflow of any internal tables, the program will terminate with a U0101 ABEND after printing the last card read.

ALLOCATING SPOOL DIRECT-ACCESS SPACE

For direct-access space, HASP requires one or more volumes whose volume serial numbers begin with the characters "SPOOL". (It is assumed that HASPGEN parameter &SPOOL has been left at the value "SPOOL".) One and only one of these volumes is designated as the primary SPOOL volume; it must be labelled SPOOL1. Each SPOOL volume must have a data set named SYS1.HASPACE; HASP will use the first extent of this data set for

SPOOLing space. SPOOL volumes may reside on any combination of direct-access device types. HASP sets up an individual parameter list for each SPOOL volume, ensuring full use of all allocated space.

It is strongly recommended that each SPOOL volume be entirely devoted to HASP usage. To allocate other frequently-referenced data sets on a SPOOL volume would degrade the efficiency of HASP's direct-access allocation algorithm. The sample job HASPOOLS shows full-volume allocation; it assumes 1-track VTOCs on cylinder 0, track 1. If full-volume allocation is used, the remainder of this section may be ignored.

If the installation requires that other data sets be allocated on a SPOOL volume, a simple example will show how to allocate the SYS1.HASPACE data set so that it contains no dead space. HASP's unit of direct-access allocation is the track group; the number of tracks in a track group is obtained by dividing the total number of tracks on a volume by &NUMTGV (number of track groups per volume). For example, the number of tracks for a 2314 volume is 4000 (regardless of the size of the SYS1.HASPACE allocation); if &NUMTGV was set to 500 at HASPGEN time, the number of tracks per track group is 4000/500 or 8. HASP will use only those track groups that fall completely within the SYS1.HASPACE allocation; therefore, an improperly allocated SYS1.HASPACE could have dead space at both ends of the extent.

For allocation, use the JCL specification:

```
SPACE=(ABSTR,(quantity,address))
```

To allocate any SPOOL volume but the primary SPOOL volume, use both "quantity" and "address" as integral multiples of number of tracks per track group. For example, specify SPACE=(ABSTR,(1000,16)), if number of tracks per group is 8.

To allocate the primary SPOOL volume (normally SPOOL1), follow the above procedure, but add 3 to "quantity" and subtract 3 from "address." HASP uses the first three tracks of allocation on this volume for checkpoint information. For example, specify SPACE=(ABSTR,(1003,13). This would allocate the 1003 tracks beginning with track 13 and ending with track 1015. HASP would use tracks 13 through 15 for checkpoint information. The 125 track groups beginning with track group 3 (which starts on track 16 and extends through track 23) and ending with track group 127 (which starts on track 1008 and extends through track 1015) would be available for HASP SPOOL data sets. The other 375 track groups on the 2314 would be permanently unavailable for HASP SPOOL data sets.

GENERATING HASP REMOTE TERMINAL PROGRAMS (RMTGEN)

This section describes the process of generating the HASP remote terminal programs described in the appendixes to the HASP Operator's Guide.

HASPGEN PREPARATIONS FOR RMTGEN

HASPGEN inserts the RMTGEN procedure into SYS1.PROCLIB and builds appropriate members of the HASP libraries SYS1.HASPMOD and SYS1.HASPSRC. These data sets, along with the procedure required for RMTGEN, should be retained in the system for (1) the initial HASP remote terminal program generation run, and (2) subsequent batch HASP remote terminal program generation runs. Figure 7 lists the data sets and members required for these generation runs.

Each new HASPGEN will recreate the HASP libraries and will require that remote terminal programs be regenerated when any of the following conditions exist:

1. Official HASP modifications are used in updating the remote terminal program source decks on SYS1.HASPSRC.

2. Installation HASPGEN parameters are changed which affect the HASP remote terminal interface (see HASPGEN Parameters).

3. Local modifications are made to HASP and/or the remote source programs that affect the remote terminals.

All named data sets must be cataloged in the System Catalog. The initial RMTGEN run will use data sets SYS1.UT1, SYS1.UT2, and SYS1.UT3 allocated for HASPGEN.

Installations should create and maintain RMTGEN option decks for use in recreating the revised remote terminal programs after each new HASPGEN (when necessary). (Note that RMTGEN runs may be required, even though no changes to the RMTGEN option decks are required.)

DSNAME	DSORG	MEMBERS	DESCRIPTIONS
SYS1.PROCLIB	PO		System Procedure Library
		RMTGEN	RMTGEN procedure
SYS1.HASPMOD	PO		HASP Load Module Library
		RMTGEN	RMTGEN main module
		GENRMT	RMTGEN source deck preparation and update module
		EXRMTGEN	HASPGEN RMTGEN executor module
		LETRRIP	Post-processor for 1130 remote Terminal Programs
		SYS3CNVT	Post-processor for System/3 Remote Terminal Programs
SYS1.HASPSRC	PO		HASP System Source Library
		HRTPOPTS	HASP Remote Terminal Standard Options
		HRTPB360	Source deck for HASP 360 and M20 BSC Remote Terminal Programs
		HRTPLOAD	Source deck for HASP 1130 BSC loader
		HRTP1130	Source deck for HASP 1130 BSC Remote Terminal Programs
		HRTPSYS3	Source deck for HASP System/3 BSC Remote Terminal Programs
SYS1.LINKLIB	PO		System Load Module Library
		ASMBLR	VS Assembler to be used for work station assemblies

Figure 7. RMTGEN Data Sets

INITIAL HASP RMTGEN RUN (PERFORMED AS PART OF HASPGEN)

If CPU remote terminals are indicated in the HASPGEN parameters, HRMTGEN will type the message "PLACE RMTGEN OPTIONS IN UNIT XXX AND REPLY 'GO', OR REPLY 'CANCEL' ". XXX is the address of the OS allocated 2540 card reader attached to the system. The operator should make sure the named 2540 card reader is not being used for any other function, e.g., HASP reader. He should clear any cards remaining in the reader; load the reader with RMTGEN options for all desired remotes; and reply GO. If no remote generations are desired initially the operator should reply CANCEL.

BATCH HASP REMOTE TERMINAL PROGRAM GENERATION RUN

RMTGEN runs may be made as a normal batch stream job. Figure 8 shows a sample job stream for a batch RMTGEN. The user options and control cards are the same as for an initial RMTGEN run.

Normally RMTGEN modules write messages to DD name SYSPRINT using format FBM with LRECL=121. The DD name may be changed to SYSLIST by including a SYSLIST DD card in the RMTGEN step. This will cause RMTGEN and assembler output to be placed on separate data sets. For example:

```
//GENJOB   JOB (.....
//STEP     EXEC RMTGEN
//RMTGEN.SYSLIST DD SYSOUT=A
//RMTGEN.OPTIONS DD *
     .   .   .   .
     .   .   .   .
     .   .   .   .
/*
```

An assembler other than the OS/VS2 ASMBLR may be used for RMTGEN processing. The PARM parameter on the RMTGEN EXEC card will override the assembler name and will pass control parameters to the assembler. The format of the EXEC card is:

```
//stepname EXEC RMTGEN,PARM='pgmname=parm1,parm2,...,parmN',...
```

Any parameters acceptable to the assembler may be placed in the parameter list (except for the DECK, NODECK, OBJECT, NOOBJECT options which are set by the RMTGEN program).

The following code exemplifies the use of an alternate assembler:

```
//GENJOB   JOB (.....
//STEPA    EXEC RMTGEN,PARM='IEV90=MSGLEVEL=5',REGION=210K
//RMTGEN.SYSLIN DD UNIT=SYSDA,SPACE=(CYL,(5,5))
//RMTGEN.OPTIONS DD *
     .   .   .   .   .
     .   .   .   .   .
     .   .   .   .   .
/*
```

Note that the SYSLIN DD statement should never be used if the assembler places its object deck output on the SYSGO data set.

```
//RMTGENJB JOB (0000,0000),'GEN REMOTE PROGRAMS',MSGLEVEL=1
//RMTGEN  EXEC RMTGEN
//RMTGEN.OPTIONS DD *
$.RMTM20,2
&RDEV(1)=2560
&RADR(1)=2
&UDEV(1)=2560
&UADR(1)=2
&WDEV(1)=2152
&NUMTANK=5
$.RMTEND
$.RMT360,3
&CMPTYPE=3
&PDEV(2)=1403
&ADAPT=030
&WADR=009
&NUMTANK=7
&CORESIZ=16
$.RMTEND
```

Figure 8. Example of Batch RMTGEN Run

RMTGEN PROGRAM EXECUTION

RMTGEN expects its input stream to contain one or more remote terminal program descriptions. Each terminal program is described by card entries in the following order:

1. HASP remote terminal program identification card
2. User RMTGEN option cards
3. $.UPDATE control card (optional)
4. Update cards if $.UPDATE card is used
5. $.RMTEND end of remote description.

This description format is repeated for each terminal to be generated. Descriptions do not affect successive descriptions in the current run or in succeeding runs.

The RMTGEN program module, acting as a monitor, LINKs to the various modules that generate the HASP remote work station load decks as follows:

1. GENRMT is invoked to read the card input stream for the remote terminal program identification, to select the appropriate standard options list for the desired remote terminal program, and to print the default values on the SYSOUT=A device.
2. GENRMT reads the overriding options from the card input stream and changes the current values. Overriding options are printed on the SYSOUT=A device as they are encountered. (See RMTGEN Parameters for RMTGEN option specifications.)
3. When $.UPDATE or $.RMTEND is encountered, the remote terminal program source is copied to a scratch data set by GENRMT (ddname=SYSIN) for the assembler. During the transfer, the final options (as specified) are used to update the source. If update is specified, data from the card input stream will be used to modify the source.
4. After the update, the assembler is invoked (by RMTGEN) to assemble the remote terminal program and, except for 1130 and System/3 programs, punch self loading object decks on the SYSOUT=B data set. 1130 or System/3 assembly places the object deck on a scratch data set.
5. On return from the assembler, if the program is for the 1130 or System/3, RMTGEN invokes a post-processor (LETRRIP or SYS3CNVT) which creates a load deck image on the SYSPUNCH data set. The resultant cards are:

a. For 1130, the RTPLOAD or RTP1130 deck

b. For System/3 without 5424, a complete load deck

c. For System/3 with 5424, a deck to be further processed (described later in this chapter).

6. If more cards are in the card input stream, the above procedures are repeated.

All listings produced by GENRMT and the assembler will have the remote terminal SIGN-ON identification number at the top of each page. With the exception of loader bootstrap cards, all object deck cards will have the identification number punched in columns 75-76.

RMTGEN INPUT CARD SPECIFICATIONS

RMTGEN accepts four basic input card groups: RMTGEN control cards, user options, update control cards, and update cards.

RMTGEN Control Cards

The general RMTGEN control card format is:

Columns	Field	Description
1-2	$.	Control card identification
3-71	operands	Variable length, separated by comma with no blanks allowed (last operand must be followed by blank)
73-80	ignored	

The first card of a remote terminal program description is the HASP remote terminal program identification card. It serves two functions:

1. Selects the appropriate standard options group and source member from the library.

2. Sets the remote terminal identification number.

The card format is:

```
$.name,n
```

where name is the name specified in Figure 9 for the remote terminal program to be generated, and n is a 1- or 2-digit terminal number, followed by a blank. (The first digit of n must not be zero.)

RMTGEN has two additional control cards:

1. $.UPDATE, which sets the update mode and causes following cards to be used to modify the remote program source deck for the current generation description

2. $.RMTEND, which signals the end of the remote generation description.

User Options

The general user options card format is:

Col. 1-n name=value

where name is a legal option specified in the appropriate remote terminal program options section (see RMTGEN Parameters), and value is a character string of up to 17 characters, ending in blank. Blanks must not appear anywhere on the card except after the value.

User options may appear in any order after the remote terminal program identification card. Each option may occur more than once. The last value for each option overrides previous values and is used in generating the remote terminal program. See default option values under RMTGEN Parameters.

Update Control Cards

The update control card format is:

Columns	Field	Description
1-2	./	Control identification
3-n	Blank	0 to 44 blanks
(n+1)-(n+6)	DELETE	Verb for delete source cards indicated
(n+7)-m	Blank	1 to (45-number of previous blanks)
(m+1)-(m+13)	SEQ1=serial1	Starting card serial number
(m+14)		
(m+15)-(m+27)	SEQ2=serial2	Ending card serial number

Update control cards may be used only during an update run, i.e., after a $.UPDATE card. The DELETE card is used to delete one or more source cards from the source deck for the described remote terminal program, as the deck is being prepared for the assembler. The DELETE card may be mixed with insertion and replacement cards containing new source statements for the assembler. All library source cards, starting with serial1 through and including serial2, will be omitted from the assembler input source. ENDUP terminates the remote terminal program description. It may be replaced by $.RMTEND, which also serves this function.

Update Cards

Update cards are assembly language source cards in the format described in the OS/VS assembler manuals. Each card may be serialized in columns 73-80 or may have all blanks in columns 73-80. Cards with blank serials will be inserted in the source deck immediately after the last serialized input card or, if following a DELETE control card, in place of the deleted source cards. Serialized cards will replace current source program cards if the serial numbers are equal to those on existing source cards, or they will be inserted in the source deck in the appropriate location, based on the serial number.

All serialized input (including update DELETE cards) must indicate ascending order serial numbers.

SYSTEM/3 96-COLUMN CARD RMTGEN OUTPUT

As described previously, RMTGEN for System/3 invokes the post-processor SYS3CVNT to produce the System/3 load deck image on the SYSPUNCH data set. The cards created are 80-column cards which, if routed (by use of a /*ROUTE card or the $R operator command) to a System/3 remote terminal utilizing the System/3 Starter System, will be punched as full 96-column System/3 load mode cards. They may also be punched locally or remotely as 80-column cards (with the punched outputs of other RMTGENs) and later be separated and routed to a System/3 Starter System, as the punched output of an 80/80 card-to-punch job. The data set utilities IEBPTPCH or IEBGENER, for example, might be used. See the HASP Operator's Guide for a System/3 Starter System description.

System/3 96-column load mode cards must be punched, as described previously, in order to use the output of a RMTGEN on a System/3. 80-column cards are not loadable on a System/3, even if the supported RPQ 1442 card reader is attached. (This section does not apply to System/3 with 1442s.)

Instead of the System/3 Starter System, any HASP System/3 Remote Terminal Processor program generated with &S396COL=1 may be used to punch RMTGEN output routed to a System/3 as described above.

HASP Remote Terminal Processor Program For	Terminal Program Identification Card (First Card Of Each Remote Description)
360/20, 2922	$.RMTM20,n
360/22, 25, 30, 40, etc.	$.RMT360,n
1130 Loader	$.RTPLOAD,n
1130	$.RTP1130,n
System/3	$.RMTSYS3,n

n= remote SIGNON number

Figure 9. RMTGEN Terminal Program Identification Cards

RMTGEN COMPLETION CODES

RMTGEN determines the highest completion codes returned by any of the HASP supplied generation modules or the assembler used for object deck creation and returns that code to the system. HASP supplied RMTGEN modules detect and return completion codes as follows:

1. ABEND 20 (RMTGEN) - A module read to end of data on the CARDIN data set without setting RMTGEN module's EODAD exit. The load modules on the Job or Step library are not correct and should be restored.

2. Completion Code 24 (EXRMTGEN) - The operator replied CANCEL to the request to place cards in the CARDIN data set, or the generation of remotes was suppressed, based on HASPGEN parameters (no HASP work station decks are generated).

3. Completion Code 24 (GENRMT) -An error was detected by GENRMT module and one of the following messages was displayed on the SYSPRINT data set:

   a. ****INVALID SELECTION CARD****

      (1) The program identification card named an unsupported remote.

      (2) The format of the identification card was incorrect.

      (3) The numeric field was not numeric.

      The card in error is displayed preceding the error message and the generation of the requested remote is suppressed.

   b. ****OPTION SPECIFICATION ERROR****

      (1) The specified RMTGEN parameter was misspelled.

      (2) The format of the card was incorrect.

      (3) Card sequence numbers were not in ascending order.

      (4) An invalid ./ card was encountered.

      (5) A $. card, other than $.RMTEND, was encountered during the update process.

      (6) A /* card was encountered within a remote description deck.

      The card in error is displayed preceding the error message and the generation of the requested remote is suppressed.

c. ****UNEXPECTED END OF CARD INPUT****

The last card of the CARDIN data set was not $.RMTEND, or in case of updating operations ./ ENDUP. The generation of the requested remote is suppressed.

d. ****HASP SOURCE LIBRARY ERROR****

(1) An internal control card on the GENPDS data set member HRTPOPTS was incorrect or missing.

(2) An overflow of the GENRMT standard options table has occurred.

The generation of the requested remote is suppressed. The user should check the spelling on his selection card and, if spelling is correct, recreate the HASP generation modules and source.

4. Completion Code 24 (LETRRIP) - An error was detected by the 1130 post-processor and an appropriate message was displayed as follows:

a. ****REMOTE id DECK INCOMPLETE****

b. **** REMOTE id EXCEEDS AVAILABLE 1130 STORAGE***.

5. Completion Code 12 (SYS3CNVT) - Unexpected or missing end-of-file indication on the input data set. The message displayed on the operator console and in the HASP Job Log is:

S3CNVT - UNEXPECTED OR MISSING END-OF-DATA

6. Completion Code 16 (SYS3CNVT) - Unable to open one or both data sets. The message displayed on the operator console and in the HASP Job Log is:

S3CNVT - UNABLE TO OPEN ONE OR BOTH DATASETS

REMOTE GENERATION FOR NON-HASP USERS

This section outlines procedures required to generate HASP remote workstation programs, without installing the complete HASP System.

PREPARATION

The remote generation (RMTGEN) process requires creation of appropriate data sets as discussed under Generating Remote Terminal Programs. The requirement may be satisfied using the following procedures:

1. Allocate and catalog the data sets:

   a. SYS1.HASPMOD - for HASPGEN and RMTGEN load modules

   b. SYS1.HASPSRC - for HASP and work station source

   c. SYS1.UT3 - for Linkage Editor utility data set.

   Refer to Figure 2.

2. Mount the HASP distribution tape on an appropriate drive and start a reader to the tape. Do not allow the jobs to begin executing.

3. Cancel all jobs read in from tape, except the first job (jobname HASPGEN).

4. Allow the HASPGEN job to execute. This will cause the required work station source decks, RMTGEN object modules, and RMTGEN procedures to be added to the system.

5. The HASPGEN job will request that the operator enter modifications to the default options (see Standard Complete HASPGEN Process). The remote work station programs are dependent on the following HASPGEN options, described under HASPGEN Parameters:

   a. &TPBFSIZ

   b. &MLBFSIZ

   The value of &MLBFSIZ is the maximum size record that can be transmitted over the communication line. This parameter must be set to the size specified at the central CPU.

If official modifications are required for the remote work station programs, these modifications should be inserted into the 2540 card reader behind the option modification cards and the UPDATE card as described under Generating a HASP System.

When the HASPGEN job completes successfully, the data sets required are ready for the RMTGEN process.

EXECUTING RMTGEN

Upon completion of the HASPGEN job one or more RMTGEN jobs may be submitted in accordance with the discussion, Batch HASP Remote Terminal Program Generation Run.

The SYSLIST DD statement that is not required when running under HASP is required when running under the Operating System without HASP. This data set may be blocked at multiples of 121 bytes. The SYSPRINT and SYSPUNCH data sets may also be blocked if permitted by the assembler. The SYSPUNCH data set used by RMTGEN 1130, and SYSTEM/3 post-processors should be a multiple of 80 bytes. Examples of SYSLIST, SYSPRINT, and SYSPUNCH DD cards follow:

```
//RMTGEN.SYSLIST  DD SYSOUT=A,DCB=BLKSIZE=1210
//RMTGEN.SYSPRINT DD SYSOUT=A,DCB=BLKSIZE=1210
//RMRGEN.SYSPUNCH DD SYSOUT=B,DCB=BLKSIZE=800
```

RMTGEN PARAMETERS

Remote generation requires specification of certain RMTGEN parameters. The following subsections describe the parameters for each of six different types of remote generation (System/360 Model 20 BSC, System/360 BSC (models other than Model 20), 1130, 1130 Loader, System/3, and 2922). The parameters are given in alphabetical order. Each parameter is described by an explanation, the default value, and notes that expand on the explanation and refer to related RMTGEN parameters.

RMTGEN PARAMETERS FOR SYSTEM/360 MODEL 20 BSC

This section describes the parameters used in assembly of the System/360 Model 20 BSC Remote Terminal Processor (RTP) program for HASP MULTI-LEAVING Remote Job Entry. The parameters are used during RMTGEN to specify hardware configuration and software options.

&CCT

Explanation: Variable symbol &CCT specifies, for all text compression (except trailing blank compression), the minimum number of characters to be compressed. A duplicate character string of fewer than &CCT characters will be treated as a string of nonduplicate characters for compression purposes. The specification must be an integer between 3 and 31, inclusive.

Default: &CCT=4

Notes:

1. See also &CMPTYPE. The value of &CCT is not used if &CMPTYPE=1.
2. A smaller value of &CCT increases efficiency of communication line usage at the expense of compute time required for compression.

&CMPTYPE

Explanation: Variable symbol &CMPTYPE specifies the type of compression to be applied to all text transmitted from the Model 20 to the central computer. The specification must be either 1, 2, or 3. The value 1 specifies trailing blank compression; 2 specifies compression of leading, embedded, and trailing blanks; 3 specifies compression of all duplicate character strings.

Default: &CMPTYPE=2

Notes: See also &CCT.

&CORESIZ

Explanation: Variable symbol &CORESIZ specifies the size of Model 20 main storage in Kbytes (1 Kbyte = 1024 bytes). The specification must be an integer between 8 and 32, inclusive.

Default: &CORESIZ=8

&ERRMSGN

Explanation: Variable symbol &ERRMSGN specifies the number of 4-byte entries to be assembled in the Model 20 RTP program as an error message log table. The specification must be an integer not less than 8.

Default: &ERRMSGN=10

&LINESPD

Explanation: Variable symbol &LINESPD specifies the speed, in baud, of the communication line to be used between the Model 20 and the central computer. The specification must be a positive integer.

Default: &LINESPD=2000

&NUMBUFS

Explanation: Variable symbol &NUMBUFS specifies number of teleprocessing buffers to be constructed by the Model 20 RTP program. The specification must be an integer no less than:

2*X+1

where:

X = 1, if either a 2520 or a 2560 is to be used as both a reader and a punch

X = 0, otherwise.

Default: &NUMBUFS=8

Notes:

1. The length of each buffer is &MLBFSIZ+5 bytes (rounded up to the next full word); the value of HASPGEN parameter &MLBFSIZ is automatically propagated to RMTGEN.

2. If &NUMBUFS specifies more buffers than can be built in available storage, the RTP program will build as many buffers as it can.

3. It is recommended that at least two buffers be provided for each output device and for the communication adapter.

&NUMTANK

Explanation: Variable symbol &NUMTANK specifies the number of decompression buffers ("decompression tanks") to be assembled in the Model 20 RTP program. The specification should be an integer not less than 2.

Default: &NUMTANK=8

Notes:

1. The length of each decompression tank is &PRTSIZE+6.

2. It is recommended that at least two tanks be provided for each printer and punch.

3. For an 8K Model 20, specification of &NUMTANK greater than 8 may cause the RTP program to assemble more than X'1F00' bytes (8K-256); the resultant program will fail to load.

&PDEV(1)

Explanation: Subscripted variable symbol &PDEV(1) specifies device type for the Model 20 printer. The specification must be either 1403 or 2203.

Default: &PDEV(1)=2203

&PRTCONS

Explanation: Variable symbol &PRTCONS specifies the degree of use of the printer as an output console. &PRTCONS is dependent on the specifications used in the generation of the HASP System pertaining to the handling of messages for the remote as follows:

1. If HASP is informed, via the RMTnn HASPGEN parameter that the remote has a console, &PRTCONS has the following meanings:

   a. &PRTCONS=0 - Error logging and display will be suppressed, and operator messages created while the remote is online to HASP will be discarded.

   b. &PRTCONS=1 - The printer will be used as an output console, when sufficient operator messages from HASP have been queued for output at the remote. If the printer is busy with job stream output, that output will be interrupted for the printing of operator messages from HASP and error messages from the remote log. When the console queue is empty, job stream output will continue.

   c. &PRTCONS=2 - The printer will be used as an output console but will not interrupt the printing of jobs. Operator messages received from HASP while jobs are being printed will be discarded.

2. If HASP is informed via the the RMTnn HASPGEN parameter that the remote does not have a console, and if HASP does not have message SPOOLing capability (as determined by the &SPOLMSG HASPGEN parameter) &PRTCONS has the following meanings:

   a. &PRTCONS=0 - Error logging and display will be suppressed, and no operator messages will be displayed.

   b. &PRTCONS=1 - Error log messages will be displayed when the printer is free to print them (no job's printed output will be interrupted).

c. &PRTCONS=2 - Same as &PRTCONS=1.

3. If HASP is informed via the RMTnn HASPGEN parameter that the remote does not have a console and if HASP has message SPOOLing capability (as determined by &SPOLMSG parameter) &PRTCONS takes on the same meanings as 2 above with an additional capability. Operator messages queued for the remote by HASP and transmitted to the remote when the printer is free and set to receive messages (via the $TRMr.PR1 command) are printed.

Settings for &PRTCONS must be 0, 1, or 2.

Default: &PRTCONS=0

Notes:

1. If &WDEV(1) is not zero, &PRTCONS should be set to zero.

2. See HASPGEN parameters RMTnn and &SPOLMSG.

3. Regardless of the settings of &WDEV(1) and &PRTCONS, error messages resulting from loggable errors detected by the remote will be discarded when the errors occur at a rate faster than the output device can display them.

&PRTSIZE

Explanation: Variable symbol &PRTSIZE specifies the length in bytes of the text portion of each decompression tank. Each tank must be long enough to hold a maximum-length output record for either the printer, the punch, or the operator console. The specification must be an integer that is the largest of 80 (if &UDEV(1) is not zero), 120 (if &WDEV(1) is not zero), or the line width of the printer.

Default: &PRTSIZE=120

&RADR(1)

Explanation: Subscripted variable symbol &RADR(1) specifies the unit address of the Model 20 card reader. The specification must correspond to the specification for &RDEV(1) as follows:

&RDEV(1)	&RADR(1)
2501	1
2520	2
2560	2

Default: &RADR(1)=1

Notes: This parameter should not be altered when generating a 2922 work station program.

&RDEV(1)

Explanation: Subscripted variable symbol &RDEV(1) specifies device type for the Model 20 card reader. The specification must be either 2501, 2520, or 2560.

Default: &RDEV(1)=2501

Notes: This parameter should not be altered when generating a 2922 work station program. See also &RADR(1).

&SUBMOD

Explanation: Variable symbol &SUBMOD specifies the submodel number of the System/360 Model 20 for the specified remote terminal. The specification must be a valid System/360 Model 20 submodel number.

Default: &SUBMOD=2

Notes: This parameter should not be altered when generating a 2922 work station program.

&UADR(1)

Explanation: Subscripted variable symbol &UADR(1) specifies the unit address of the Model 20 card punch. The specification must correspond to the specification for &UDEV(1) as follows:

&UDEV(1)	&UADR(1)
1442	3
2520	2
2560	2
0	not present

Default: &UADR(1)=3

&UDEV(1)

Explanation: Subscripted variable symbol &UDEV(1) specifies device type for the Model 20 card punch. The specification must be either 1442, 2520, 2560, or 0. Specification 0 is used when the Model 20 does not include a card punch.

Default: &UDEV(1)=1442

Notes: &UDEV(1)=0 for 2922, unless RPQ punch is included (in which case &UDEV should not be altered). See also &UADR(1), unless &UDEV(1)=0.

&WDEV(1)

Explanation: Subscripted variable symbol &WDEV(1) specifies device type for the Model 20 console. The specification must be either 2152 (if a console is present) or 0 (if no console is present).

Default: &WDEV(1)=0

Notes: If &WDEV(1)=2152, console support must be indicated for this remote terminal at HASPGEN time. See HASPGEN parameter RMTnn.

&WTOSIZE

Explanation: Variable symbol &WTOSIZE specifies the maximum length in bytes of a HASP operator command to be transmitted from the Model 20 to the central computer. The specification must be a positive integer not greater than 120.

Default: &WTOSIZE=120

Notes: If &WDEV(1)=0, this parameter is not used.

&XPARENT

Explanation: Variable symbol &XPARENT specifies presence or absence of the Text Transparency feature. If the Binary Synchronous Communication Adapters at both the Model 20 and the central computer have the Text Transparency feature, YES should be specified; otherwise, NO should be specified.

Default: &XPARENT=YES

RMTGEN PARAMETERS FOR SYSTEM/360 (EXCEPT MODEL 20) BSC

This section describes the parameters used in assembly of the System/360 BSC Remote Terminal Processor (RTP) program for HASP MULTI-LEAVING Remote Job Entry. The parameters are used during RMTGEN to specify hardware configuration and software options.

&ADAPT

Explanation: Variable symbol &ADAPT specifies the unit address of the Binary Synchronous Communication Adapter used by the System/360 remote terminal to communicate with HASP at the central computer. The specification must be a valid unit address.

Default: &ADAPT=020

&CCT

Explanation: Variable symbol &CCT specifies, for all text compression except trailing blank compression, the minimum number of characters to be compressed. A duplicate character string of fewer than &CCT characters will be treated as a string of non-duplicate characters for compression purposes. The specification must be an integer between 3 and 31, inclusive.

Default: &CCT=4

Notes:

1. See also &CMPTYPE. The value of &CCT is not used if &CMPTYPE=1.
2. A smaller value of &CCT increases efficiency of communication line usage at the expense of compute time required for compression.

&CMPTYPE

Explanation: Variable symbol &CMPTYPE specifies type of compression to be applied to all text transmitted from the System/360 remote terminal to the central computer. The specification must be either 1, 2, or 3. The value 1 specifies trailing blank compression; 2 specifies compression of leading, embedded, and trailing blanks; 3 specifies compression of all duplicate character strings.

Default: &CMPTYPE=2

Notes: See also &CCT.

&CORESIZ

Explanation: Variable symbol &CORESIZ specifies the size of main storage for the System/360 remote terminal in Kbytes (1 Kbyte = 1024 bytes). The specification must be an integer between 8 and 32, inclusive. If the System/360 is larger than 32K bytes, &CORESIZ must be specified as 32.

Default: &CORESIZ=8

&ERRMSGN

Explanation: Variable symbol &ERRMSGN specifies the number of 4-byte entries to be assembled in the System/360 remote terminal as an error message log table. The specification must be an integer not less than 8.

Default: &ERRMSGN=10

&LINESPD

Explanation: Variable symbol &LINESPD specifies the speed, in baud, of the communication line to be used between the System/360 remote terminal and the central computer. The specification must be a positive integer.

Default: &LINESPD=2000

&MACHINE

Explanation: Variable symbol &MACHINE specifies the model number of the System/360 to be used as a HASP remote terminal. The specification must be a valid number for a System/360 that includes the standard instruction set and the decimal instruction set.

Default: &MACHINE=30

&NUMBUFS

Explanation: Variable symbol &NUMBUFS specifies the number of teleprocessing buffers to be constructed by the System/360 RTP program. The specification must be an integer no less than:

2*X+1

where:

X = the number of 2520 or 1442 units used as both a reader and a punch

Default: &NUMBUFS=8

Notes:

1. The length of each buffer is &MLBFSIZ+5 bytes (rounded up to a multiple of 4); the value of HASPGEN parameter &MLBFSIZ is automatically propogated to RMTGEN.

2. If &NUMBUFS specifies more buffers than can be built in available storage, the RTP program will build as many buffers as it can.

3. It is recommended that at least two buffers be furnished for each output device and for the communication adapter.

&NUMTANK

Explanation: Variable symbol &NUMTANK specifies the number of decompression buffers ("decompression tanks") to be assembled in the System/360 RTP program. The specification must be an integer not less than 2*(number of 2540 punches attached).

Default: &NUMTANK=5

Notes:

1. The length of each decompression tank is &PRTSIZE+6.

2. It is recommended that at least two tanks be provided for each printer and each punch (3 for a 2540 Punch).

&PADR(n)

Explanation: Subscripted variable symbols &PADR(n) specify unit addresses for the printers defined by &PDEV(n). For each &PDEV(n) not specified as zero, the corresponding symbol &PADR(n) must specify the device's 3-character hexadecimal unit address.

Default:
```
&PADR(1)=00E
&PADR(2)=00F
&PADR(3)=FFF
&PADR(4)=FFF
&PADR(5)=FFF
&PADR(6)=FFF
&PADR(7)=FFF
```

&PDEV(n)

Explanation: Subscripted variable symbols &PDEV(n) specify the existence and device types of the remote terminal printers. Each specification must be either 1403, 1443, or 0. A specification of 0 indicates that the associated printer does not exist.

Default:
```
&PDEV(1)=1403
&PDEV(2)=0
&PDEV(3)=0
&PDEV(4)=0
&PDEV(5)=0
&PDEV(6)=0
&PDEV(7)=0
```

Notes:

1. If &PDEV(n) is specified as a device type, &UDEV(8-n) must be specified as zero.

2. If &PDEV(n+1) is specified as a device type, &PDEV(n) must be specified as a device type.

3. If more than one printer is specified, a Device Control Table (DCT) for each additional printer must be added to the HASP System.

&PRTSIZE

Explanation: Variable symbol &PRTSIZE specifies the length in bytes of the text portion of each decompression tank. Each tank must be long enough to hold a maximum-length output record for either a printer, a punch, or the operator console. The specification must be an integer that is the larger of 120 or the line width of the widest printer.

Default: &PRTSIZE=132

&RADR(n)

Explanation: Subscripted variable symbols &RADR(n) specify unit addresses for the readers defined by &RDEV(n). For each &RDEV(n) not specified as zero, the corresponding symbol &RADR(n) must specify the device's 3-character hexadecimal unit address.

Default:
```
&RADR(1)=00C
&RADR(2)=FFF
&RADR(3)=FFF
&RADR(4)=FFF
&RADR(5)=FFF
&RADR(6)=FFF
&RADR(7)=FFF
```

&RDEV(n)

Explanation: Suscripted variable symbols &RDEV(n) specify the existence and device types of the remote terminal readers. Each specification must be either 2540, 2501, 1442, 2520, or 0. A specification of 0 indicates that the associated reader does not exist.

Default:
```
&RDEV(1)=2540
&RDEV(2)=0
&RDEV(3)=0
&RDEV(4)=0
&RDEV(5)=0
&RDEV(6)=0
&RDEV(7)=0
```

Notes:

1. If &RDEV(n+1) is specified as a device type, &RDEV(n) must be specified as a device type.

2. If more than one reader is specified, a Device Control Table (DCT) for each additional reader must be added to the HASP System.

&UADR(n)

Explanation: Subscripted variable symbols &UADR(n) specify unit addresses for the punches defined by &UDEV(n). For each &UDEV(n) not specified as zero, the corresponding symbol &UADR(n) must specify the device's 3-character hexadecimal unit address.

Default:
```
&UADR(1)=00D
&UADR(2)=FFF
&RADR(3)=FFF
&RADR(4)=FFF
&RADR(5)=FFF
&RADR(6)=FFF
&RADR(7)=FFF
```

&UDEV(n)

Explanation: Subscripted variable symbols &UDEV(n) specify the existence and device types of the remote terminal punches. Each specification must be either 2540, 2520, 1442, or 0. A specification of 0 indicates that the associated punch does not exist.

Default:
```
&UDEV(1)=2540
&UDEV(2)=0
&UDEV(3)=0
&UDEV(4)=0
&UDEV(5)=0
&UDEV(6)=0
&UDEV(7)=0
```

Notes:

1. If &UDEV(n) is specified as a device type, &PDEV(8-n) must be specified as zero.

2. If &UDEV(n+1) is specified as a device type, &UDEV(n) must be specified as a device type.

3. If more than one punch is specified, a Device Control Table (DCT) for each additional punch must be added to the HASP System.

&WADR(1)

Explanation: Subscripted variable symbol &WADR(1) specifies the unit address of the 1052 Operator Console on the System/360 remote terminal. The specification must be a 3-character hexadecimal unit address.

Default: &WADR(1)=01F

&WTOSIZE

Explanation: Variable symbol &WTOSIZE specifies the maximum length in bytes of a HASP operator command to be transmitted from the System/360 remote terminal to the central computer. The specification must be a positive integer not greater than 120.

Default: &WTOSIZE=120

&XPARENT

Explanation: Variable symbol &XPARENT specifies presence or absence of the Text Transparency feature. If the Binary Synchronous Communication Adapters at both the System/360 remote terminal and the central computer have the Text Transparency feature, YES should be specified; otherwise, NO should be specified.

Default: &XPARENT=YES

RMTGEN PARAMETERS FOR 1130

This section describes the parameters used in assembly of the 1130 Remote Terminal Processor (RTP) program for HASP MULTI-LEAVING Remote Job Entry. The parameters are used during RMTGEN to specify hardware configuration and software options.

&CLOCK

Explanation: Variable symbol &CLOCK specifies the type of communication adapter clocking available on the 1130. A specification of &CLOCK=0 means that data set clocking is being used. The value &CLOCK=1 specifies internal (1130) clocking.

Default: &CLOCK=0

Notes:

1. The rate of insertion of the synchronous idle sequence in the transmitted data is determined by variables &CLOCK, &LINESPD and &TRANPRN. The relationship of these variables to the insertion rate is:

&CLOCK	&TRANPRN	Insertion Rate
0	0	Every &LINESPD/8 characters
0	1	Every &LINESPD/8 characters
1	0	Every 70 characters
1	1	Every &LINESPD/8 characters

2. The equation used for the insertion rate is:

   (&LINESPD/8)*T

   where T is 1.00 second, which is the nominal 2701 timer value.

&CMPTYPE

Explanation: Variable symbol &CMPTYPE specifies the compression technique to be applied to the data transmitted to the central HASP System. Possible values of &CMPTYPE are:

1. &CMPTYPE=0 - no compression of duplicate characters and no truncation of trailing blanks.

2. &CMPTYPE=1 - trailing blank truncation only.

3. &CMPTYPE=2 - full compression, trailing blank truncation, and encoding of duplicate characters.

Default: &CMPTYPE=2

Notes: The process of compressing input data offers optimum performance with respect to efficient line utilization. However, the factors of line speed, CPU availability, buffer size, line turnaround time, nature of the data to be compressed, etc., contribute to the overall operation of the work station program. Since compression and truncation require considerable CPU time, the user may decide, on the basis of the other variables, to respecify the compression technique.

&DELAY

Explanation: Variable symbol &DELAY defines the number of time intervals that RTP1130 will delay in transmitting a "handshaking" sequence (DLE-ACK0) to the central HASP site. The hardware program timer clock is used to measure the delay and is assumed to be set to a nominal value of .35 seconds.

Default: &DELAY=3

Notes:

1. &DELAY=3 results in a delay of 1.05 seconds, assuming a timer interval of .35 seconds.

2. The purpose of the delay when "handshaking" is to minimize CPU processing at the central HASP computer when no data is being transmitted.

3. The value of &DELAY must not be set to such a large increment that the delay will be greater than the timeout period of the central site 2701/2703.

&FULLIST

Explanation: Variable symbol &FULLIST specifies the type of assembly listing produced by the OS assembler during the RMTGEN process. If &FULLIST=0, the assembly listing will be produced according to the PRINT NOGEN stipulation of the assembler. If &FULLIST=1, the listing will be produced according to the PRINT GEN stipulation.

Default: &FULLIST=1

Notes: Since most of the code in RTP1130 and RTPLOAD is created by macro instructions, the specification of &FULLIST=0 will essentially produce a source listing (cross-referenced) without the 1130 assembled instructions. Error messages will not appear on the listing.

&LINESPD

Explanation: Variable symbol &LINESPD specifies the baud rate for the communication line interface to the work station program. The value should correspond to the selected setting of the baud rate switch on the 1130 SCA control panel: 1200,2000, ..., etc.

Default: &LINESPD=2000

Notes: The rate of insertion of the synchronous idle sequence (DLE-SYN or SYN-SYN) in the transmitted data is determined by variables &CLOCK, &LINESPD, and &TRANPRN. See Note 1 of &CLOCK description.

&MACHSIZ

Explanation: Variable symbol &MACHSIZ specifies the amount of 1130 core to be used by RTP1130. The value of &MACHSIZ is in units of 1130 words.

Default: &MACHSIZ=8192

Notes:

1. The value of &MACHSIZ indicates that &MACHSIZ number of words, starting at location 0, are available for the work station program (RTPBOOT, RTPLOAD, and RTP1130).

2. The same variable symbol must be defined for RTPLOAD and should have the same value.

3. The value of &MACHSIZ may be less than the actual available storage but must not be greater.

&PN1442

Explanation: Variable symbol &PN1442 defines a 1442 Punch. If the variable is set to 1, RTP1130 will include support for punched card output produced by jobs at the central HASP site. If the variable is set to 0, no support for the 1442 Punch will be provided. See &RD1442 for the definition of a reader function on the 1442.

Default: &PN1442=1

&PRFOTLW

Explanation: Variable symbol &PRFOTLW defines the line width of the 1403 Printer. Line widths can be 120 or 132 characters.

Default: &PRFOTLW=120

Notes: The definition of the line width for all printers on a particular remote is a HASPGEN requirement. See HASPGEN parameter RMTnn.

&PR1132

Explanation: Variable symbol &PR1132 defines an 1132 Printer. If the variable is set to 1, RTP1130 will include 1132 support for printing job output. If the variable is set to 0, no support for the 1132 will be included in RTP1130.

Default: &PR1132=0

&PR1403

Explanation: Variable symbol &PR1403 defines a 1403 Printer for use as an output device. If &PR1403=1, the 1403 function will be included in RTP1130. If &PR1403=0, the function is deleted from RTP1130.

Default: &PR1403=1

Notes: See &PRFOTLW 1403 line width specification.

&RD1442

Explanation: Variable symbol &RD1442 defines a 1442 as a card reader. If the variable is set to 1, RTP1130 will be assembled with all necessary control blocks and support routines to provide job input from the 1442. If the variable is set to 0, no support for the 1442 Card Reader will be provided in RTP1130. See &PN1442 for a definition of the 1442 Punch function.

Default: &RD1442=1

Notes: If &RD1442 is set to 1, and the 1442 Card Reader does not exist, operation of the work station program may be unpredictable.

&RD2501

Explanation: Variable symbol &RD2501 defines a 2501 Card Reader. If the variable is set to 1, RTP1130 will be assembled with all necessary control blocks and subroutines to support the 2501 as a job input device. If the variable is set to 0, no support for the 2501 will be included in RTP1130.

Default: &RD2501=0

Notes: If the variable &RD2501 is set to 1, and a 2501 does not exist, operation of the work station program will be unpredictable and usually unproductive.

&RTPLORG

Explanation: Variable symbol &RTPLORG defines the origin in 1130 storage of RTPLOAD, which is used to load RTP1130.

Default: &RTPLORG=2*(&MACHSIZ-1024)

Notes:

1. Assuming &MACHSIZ=8192, the default value is 14336 (which is twice the actual 1130 storage address because the value is used in an ORG operation and must be in terms of bytes, not 1130 words).

2. The RTPLOAD program must origin in storage available between the beginning of the buffer pool and the end of defined (&MACHSIZ) storage, minus the length of RTPLOAD. The default value of &RTPLORG allows an RTPLOAD of 1024 words.

&TRANPRN

Explanation: Variable symbol &TRANPRN defines the simulation of the Binary Synchronous Transparency feature. If &TRANPRN is set to 1, RTP1130 will simulate the Transparency feature in the same manner as the 2701 SDA-II Adapter, equipped with the Transparency feature. If the variable is set to 0, no simulation will occur, and data containing transparent characters cannot be properly processed by RTP1130.

Default: &TRANPRN=1

Notes:

1. If &TRANPRN=0, the conversion of card code data is monitored, and all BSC control characters are converted to hexadecimal 0. This prevents mispunched data from causing an infinite error retry if the central site does not have transparency.

2. See &LINESPD and &CLOCK for additional influence of &TRANPRN.

3. If &TRANPRN=1, the generated RTP program will communicate only with a 2701 or 2703 Adapter that has the Text Transparency feature.

RMTGEN PARAMETERS FOR 1130 LOADER

This section describes the parameters used in assembly of RTPLOAD, the 1130 Loader Program. RTPLOAD loads the 1130 Remote Terminal Processor (RTP) program. RTPLOAD's three parameters specify machine size, loader origin, and an assembler list option.

The RMTGEN processes produce the object decks for RTPLOAD and RTP1130. The bootstrap loader (RTPBOOT) cannot be produced on System/370 and must be keypunched as indicated in the RTP section of the HASP Logic manual.

&FULLIST

Explanation: Variable symbol &FULLIST specifies the type of assembly listing produced by the OS assembler during the RMTGEN process. If &FULLIST is set to 0, the assembly listing will be produced according to the PRINT NOGEN stipulation of the assembler. If &FULLIST is set to 1, the listing will be produced according to the PRINT GEN stipulation.

Default: &FULLIST=1

Notes: Since most of the code in RTP1130 and RTPLOAD is created by macro instructions, the specification of &FULLIST=0 will essentially produce a source listing (cross-referenced) without the 1130 assembled instructions. Error messages will not appear on the listing.

&MACHSIZ

Explanation: Variable symbol &MACHSIZ specifies the amount of 1130 core to be used by RTPLOAD. The value of &MACHSIZ is in units of 1130 words.

Default: &MACHSIZ=8192

Notes:

1. The value of &MACHSIZ indicates that &MACHSIZ number of words, starting at location 0, are available for the work station program (RTPBOOT, RTPLOAD and RTP1130).

2. The same variable symbol must be defined for RTP1130 and should have the same value.

3. The value of &MACHSIZ may be less than the actual available storage but must not be greater.

&RTPLORG

Explanation: Variable symbol &RTPLORG defines the origin in 1130 storage of the RTPLOAD program, which is used to load RTP1130.

Default: &RTPLORG=2*(&MACHSIZ-1024)

Notes:

1. Assuming &MACHSIZ=8192, the default value is 14336 (which is twice the actual 1130 storage address because the value is used in an ORG operation and must be in terms of bytes, not 1130 words).

2. RTPLOAD must origin in storage available between the beginning of the buffer pool and the end of defined (&MACHSIZ) storage minus the length of RTPLOAD. The default value of &RTPLORG allows an RTPLOAD of 1024 words.

RMTGEN PARAMETERS FOR SYSTEM/3

This section describes the parameters used in assembly of the System/3 Remote Terminal Processor (RTP) for HASP MULTI-LEAVING Remote Job Entry. The parameters are used during RMTGEN to specify hardware configuration and software options.

&COMP

Explanation: Variable symbol &COMP specifies degree of text compression to be provided for all text transmitted from the System/3 to HASP. The specification must be either 0, 1, or 2, indicating:

1. &COMP=0 - neither compression nor truncation is performed.
2. &COMP=1 - trailing blanks are truncated from each logical record before it is transmitted.
3. &COMP=2 - compression takes place after truncation. Strings of from two to 31 blanks are compressed to a single byte; strings of from three to 31 duplicate characters are compressed to two bytes.

Default: &COMP=2

&DEBUG

Explanation: Variable symbol &DEBUG specifies inclusion or exclusion of certain validity tests and a core dump program in the System/3 Remote Terminal Processor (RTP) program. The specification must be either 0 or 1.

Default: &DEBUG=0

&DIAL, &DIAL1

Explanation: Variable symbols &DIAL and &DIAL1 specify the telephone number to be used during the initialization process. The values will be included on the default /*SIGNON card assembled into the System/3 RTP program and will be preceded by the keyword DIAL (unless the parameters are left at their defaults). Each specification is a string of from one to eight decimal digits. If the telephone number is eight or fewer digits long, it should be specified by &DIAL. If the telephone number is longer than eight digits, its leftmost eight digits should be specified by &DIAL and the remaining digits by &DIAL1.

Default: &DIAL=(null string)
&DIAL1=(null string)

&MACHSIZ

Explanation: Variable symbol &MACHSIZ specifies the size of System/3 core storage. The specification should be either 8192, 12288, 16384, 24576, or 32768 for core storage sizes of 8K, 12K, 16K, 24K, or 32K, respectively.

Default: &MACHSIZ=8192

&PASSWD

Explanation: Variable symbol &PASSWD specifies a password to be used during the SIGNON process. The value will be included on the default /*SIGNON card assembled into the System/3 RTP program. The specification must be a character string of from one to eight characters. If blanks are desired, no specification may be made.

Default: &PASSWD=(null string)

&PC(n)

Explanation: Subscripted variable symbols &PC(n) specify skip information for the 5203 or 1403 Printer. &PC(n) is set to the print line number to which paper will be skipped when the System/3 RTP program simulates the 1403 command "Skip to Channel n". Each specification must be an integer between 0 and &S3FORML, inclusive. A specification of 0 causes no forms movement.

Default:

```
&PC(1)=1
&PC(2)=0
&PC(3)=0
&PC(4)=0
&PC(5)=0
&PC(6)=0
&PC(7)=0
&PC(8)=0
&PC(9)=0
&PC(10)=0
&PC(11)=0
&PC(12)=&S3FORML-5
```

&PRTCONS

Explanation: Variable symbol &PRTCONS specifies utilization of the 5203 or 1403 Printer as an operator's output console. The specification must be 0, 1, or 2, indicating:

1. &PRTCONS=0 - the printer will never be used as an operator's output console.

2. &PRTCONS=1 - the System/3 RTP program will attempt to hold operator messages from HASP until a job has completed printing. However, if two or more MULTI-LEAVING buffers containing HASP operator messages are received, the printer will eject a page (skip to channel 1), print the HASP operator messages, eject another page, and resume printing its job.

3. &PRTCONS=2 - the System/3 RTP program will throw away all operator messages while the printer is printing a job. While the printer is dormant, it will print any received messages.

Default: &PRTCONS=2

Notes:

1. If &S35471=1, the value of &PRTCONS is ignored and assumed to be zero.

2. Regardless of the setting of &PRTCONS, messages temporarily saved on disk for a remote terminal will be printed to the terminal as a job. Thus, they will always appear on the printer, even if another console exists. See also HASPGEN parameter &SPOLMSG.

3. If &PRTCONS is specified greater than zero, MULTI-LEAVING console support should be specified in HASPGEN parameter RMTnn.

&S3CMDS

Explanation: Variable symbol &S3CMDS specifies inclusion or exclusion, local to the System/3, of a command facility and commands to assist the System/3 operator. The specification must be either 0 or 1.

Default: &S3CMDS=0

Notes: Commands available with this facility are explained in the System/3 appendix to the HASP Operator's Guide.

&S3FORML

Explanation: Variable symbol &S3FORML specifies the number of print lines on a page of the continuous forms used on the 5203 or 1403 Printer. The specification must be an integer not less than 6.

Default: &S3FORML=66

&S3NPUNS

Explanation: Variable symbol &S3NPUNS specifies the maximum number of jobs that can be punching simultaneously at the System/3 remote terminal. The specification must be 1, 2, or 3. (A value of 3 allows simultaneous operation of both 5424 hoppers and the 1442 hopper as punches.)

Default: &S3NPUNS=1

Notes: If &S3NPUNS is set to 2 or 3, extra DCTs for the appropriate remote must be added to the HASP System at HASPGEN time.

&S3NRDRS

Explanation: Variable symbol &S3NRDRS specifies the maximum number of job streams that can be reading simultaneously from the System/3 remote terminal. The specification must be 1, 2, or 3. (A value of 3 allows simultaneous operation of both 5424 hoppers and the 1442 hopper as readers.)

Default: &S3NRDRS=1

Notes: If &S3NRDRS is set to 2 or 3, extra DCTs for the appropriate remote must be added to the HASP System at HASPGEN time.

&S3OBJDK

Explanation: Variable symbol &S3OBJDK specifies inclusion of a facility to punch OS object decks. Text Transparency should be present. The specification should be 0 or 1.

If &S3OBJDK=1, each card of an OS object deck will be expanded and punched into two 96-column cards. These cards will be recognized when read by a System/3 RTP program for which &S3OBJDK=1. For every two 96-column cards read, an OS object deck card image will be transmitted.

Default: &S3OBJDK=0

&S3SIP

Explanation: Variable symbol &S3SIP specifies usage of those bytes of System/3 core storage between X'100' and X'1FF', inclusive. The specification must be either 0 or 1. For &S3SIP=1, the System/3 RTP program will not use the bytes; their values will be saved for use by the System/3 Card System Initialization program.

Default: &S3SIP=0

&S3TRACE

Explanation: Variable symbol &S3TRACE specifies the number of 4-byte entries in the System/3 RTP program's internal error message table. The specification must be an integer greater than 1.

Default: &S3TRACE=10

&S3XPAR

Explanation: Variable symbol &S3XPAR specifies presence or absence of the EBCDIC Text Transparency feature. The specification should be 1 if both the central computer's communications adapter and the System/3 BSCA have the EBCDIC Text Transparency feature; otherwise, the specification should be 0.

Default: &S3XPAR=0

&S31442

Explanation: Variable symbol &S31442 specifies inclusion or exclusion of support for the 1442 Card Reader/Punch. The specification must be 1 for inclusion and 0 for exclusion of 1442 support.

Default: &S31442=0

Notes: If &S31442=1, the resultant System/3 RTP program requires that a 1442 be present on the System/3.

&S35424

Explanation: Variable symbol &S35424 specifies inclusion or exclusion of support for the 5424 Multi-Function Card Unit. The specification must be 1 for inclusion or 0 for exclusion of 5424 support.

Default: &S35424=1

Notes:

1. If &S35424 is specified as 0, &S31442 must be specified as 1.

2. See Generating HASP Remote Terminal Programs for RMTGEN considerations for &S35424=0.

3. See the System/3 appendix to the HASP Operator's Guide for program loading considerations for &S35424=0.

&S35471

Explanation: Variable symbol &S35471 specifies presence or absence of a 5471 Printer-Keyboard on the System/3. The 5471 will be used as an operator's input/output console. The specification must be 1 if a 5471 is present; otherwise it must be 0.

Default: &S35471=0

Notes:

1. If console support is desired, HASPGEN parameter RMTnn must specify MULTI-LEAVING console support.

2. Regardless of the setting of &S35471, messages from HASP can print on the printer. See RMTGEN parameter &PRTCONS, Note 2, and HASPGEN parameter &SPOLMSG.

&S35475

Explanation: Variable symbol &S35475 specifies presence or absence of a 5475 Data Entry Keyboard on the System/3. The 5475 will be used as an operator's console. The specification must be 1 if a 5475 is present; otherwise, it must be 0.

Default: &S35475=0

Notes:

1. If &S35471=1, this parameter is ignored.

2. If console support is desired, HASPGEN parameter RMTnn must specify MULTI-LEAVING console support.

3. For output console specification, see RMTGEN parameter &PRTCONS.

&S396COL

Explanation: Variable symbol &S396COL specifies inclusion or exclusion of the System/3 load-mode punch option. The specification must be either 0 or 1. If &S396COL is specified, the resultant System/3 RTP program will be capable of receiving correctly the punched output of a System/3 RMTGEN.

Default: &S396COL=0

RMTGEN PARAMETERS FOR 2922

To generate a 2922 Remote Terminal Processor (RTP) program for HASP MULTI-LEAVING RJE, the parameters and procedures for the System/360 Model 20 BSC should be used, subject to the following discussion.

Some parameters should be specifically set. They are:

```
&PDEV(1)=1403
&PRTSIZE=132
&UDEV(1)=0
&WDEV(1)=2152, if the optional typewriter console is installed
&XPARENT=NO, if optional transparency is not installed
&LINESPD=xxxx (the actual line speed used)
```

Some parameters should not be altered from their default values. They are:

```
&CORESIZ     &RADR(1)     &RDEV(1)
&SUBMOD      &UADR(1)
```

All other Model 20 BSC parameters may be allowed to default or may be altered as desired, according to the description under RMTGEN Parameters.

STORAGE REQUIREMENTS

This section is provided to allow installations to compute the size of a HASP System, based on the HASPGEN options selected. In all computations, the maximum degree of HASP overlay (&OLAYLEV=15) is assumed.

ADDITIONAL NUCLEUS STORAGE REQUIREMENTS

In addition to storage required as a region, HASP also requires certain fixed space in the Nucleus of the Operating System as follows:

1. Space required for the HASP pseudo device Unit Control Blocks
2. Space for the HASP initialization SVC (200 bytes).

ADDITIONAL LSQA STORAGE REQUIREMENTS

OS/VS2 fixes a minimum of two 4K pages in the LSQA segment of an active job. There is approximately 3K unused in this fixed 8K after HASP has been initiated. HASP uses this space (subpool 255) to dynamically build all DCBs, all DEBs, and certain work areas for RPS devices. Only an extremely large configuration (more than 50 unit record devices and HASP RJE lines) may cause another 4K page of LSQA to be fixed.

HASP MODULE STORAGE REQUIREMENT

Storage requirement of the primary HASP module is expressed by the following formula:

```
SHASP = 26,000 + S1 + S2 + S3 + S4 + S5 + S7 + S8 + S9 +
        S10 + S11 + S12 + S13 + S14 + S15 + S16 + S17 + S18 +
        S19 + S20 + S21 + S22 + S23 + S24 + S25 + S26 + S27 +
        S28 + S29 + S30 + S31 + S32 bytes
```

where the values of Sn are defined in the following paragraphs.

To facilitate ease in computation and simplicity of equations, the following value should be computed first:

```
DAMAP = &NUMDA * ((&NUMTGV + 7) / 8)
```

```
   DCT  = ((&NUMCLAS + 53) / 4) * 4
```

The values of Sn can then be computed as follows:

```
   S1  = &NUMRDRS * (280 + DCT)

   S2  = &NUMOSC * 18

         {0 . . . . . . . . . . . . . . . . . . . . if &NUMINRS = 0
   S3  =
         {400 + &NUMINRS * (280 + DCT)  . . . . if &NUMINRS ≠ 0

   S4  = &NUMPRTS * (552 + DCT + 8 * &NOPRCCW)

   S5  = &NUMPUNS * (740 + DCT + 8 * &NOPUCCW)

   S7  = &NUMDA * 42 + DAMAP - 16

   S8  = &NUMBUF * (88 + &BUFSIZE)

   S9  = &NUMOACE * 1376

   S10 = &NUMWTOQ * 140

   S11 = &MAXJOBS * 24

   S12 = &NUMJOES * 36

   S13 = &MAXXEQS * 208

   S14 = &MAXPART * 4 * ((12 + &MAXCLAS) / 4)

   S15 = &NUMDDT * 37

         {0 . . . . . . . . . . . . . . . . . . . if &NUMSMFB < 2
   S16 =
         {730 + &NUMSMFB * (8 + &SMFRSIZ) . . . if &NUMSMFB ≥ 2

         {0 . . . . . . . . . . . . . . . . . . . if &PRIRATE = 0
   S17 =
         {108 . . . . . . . . . . . . . . . . . . if &PRIRATE ≠ 0

         {0 . . . . . . . . . . . . . . . if &WCLSREQ = ********
   S18 =
         {96  . . . . . . . . . . . . . . if &WCLSREQ ≠ ********

         {0 . . . . . . . . . . . . . . . . . . if &OSINOPT =  NO
   S19 =
         {26  . . . . . . . . . . . . . . . . . if &OSINOPT = YES
```

$$S20 = \begin{cases} 0 & \text{if \&XBATCHC = null} \\ 567 + 14 * \&MAXPART & \text{if \&XBATCHC} \neq \text{null} \end{cases}$$

$$S21 = \begin{cases} 0 & \text{if \&TIMEOPT = 4} \\ 134 & \text{if \&TIMEOPT} \neq 4 \end{cases}$$

$$S22 = \begin{cases} 0 & \text{if \&PRTRANS = NO} \\ 306 & \text{if \&PRTRANS = YES} \end{cases}$$

$$S23 = \begin{cases} 0 & \text{if \&TSOSTCN = NO} \\ 262 & \text{if \&TSOSTCN = YES} \end{cases}$$

$$S24 = \begin{cases} 0 & \text{if \&DEBUG = NO} \\ 226 + DAMAP & \text{if \&DEBUG = YES} \end{cases}$$

$$S25 = \begin{cases} 0 & \text{if \&TRACE = 0} \\ 534 + 64 * \&TRACE & \text{if \&TRACE} \neq 0 \end{cases}$$

$$S26 = \begin{cases} 0 & \text{if \&OREPSIZ = 0} \\ 72 + \&OREPSIZ & \text{if \&OREPSIZ} \neq 0 \end{cases}$$

$$S27 = \begin{cases} 0 & \text{if \&DMNDSET = NO} \\ 60 & \text{if \&DMNDSET = YES} \end{cases}$$

$$S28 = \begin{cases} 0 & \text{if \&FCBV = NO} \\ 222 & \text{if \&FCBV = YES} \end{cases}$$

$$S29 = \begin{cases} 0 & \text{if \&RPS = NO} \\ 148 + 4 * \&NUMDA & \text{if \&RPS = YES} \end{cases}$$

$$S30 = \begin{cases} 0 & \text{if \&NUMLNES = 0} \\ \left.\begin{matrix} R1 + R2 + R3 + R4 + R5 + R6 + R7 + \\ R8 + R9 + R10 + R11 \end{matrix}\right\} & \text{if \&NUMLNES} \neq 0 \end{cases}$$

where:

R1 = &NUMTPBF * (144 + &TPBFSIZ)

R2 = &NUMLNES * (196 + DCT)

R3 = { 0 . . . . . if &DEBUG = NO
     { 36 . . . . . if &DEBUG = YES

R4 = &NUMTPRD * (280 + DCT)

R5 = &NUMTPPR * (368 + DCT)

R6 = &NUMTPPU * (368 + DCT)

R7 = { 0 . . . . . if &SPOLMSG = 0
     { (&SPOLMSG + 7) / 8 + &NUMRJE * 8 + 38 . . if &SPOLMSG ≠ 0

R8 = { 0 . . . . . if &BSHTAB = NO
     { 194 . . . . . if &BSHTAB = YES

R9 = { 0 . . . . . if &BSHPRES = NO
     { 196 . . . . . if &BSHPRES = YES

R10 = { 0 . . . . . if &USASCII = NO
      { 548 . . . . . if &USASCII = YES

R11 = a value selected from the following table:

&BSC2770 or &BSC2780 or &BSC3780	BSCCPU	R11
YES	NO	5,840
NO	YES	6,202
YES	YES	8,460

S31 = { 0 . . . . . if &NUM3800 = 0
      { 123 * &NUM3800 + 13000. . . . . if &NUM3800 > 0

S32 = { 0 . . . . . if &NUM3800 = 0
      { 846 . . . . . if &NUM3800 > 0 and &OLAYLEV = 0
      { 1766. . . . . if &NUM3800 > 0 and &OLAYLEV > 0

STORAGE REQUIREMENT FOR A TYPICAL HASP

Consider a HASP package which has been HASPGENed to be used on a machine with Remote Job Entry capabilities. The HASPGEN parameters might be set as follows:

```
&NUMOSC  = 2          &NUMJOES = 70
&TSOSTCN = NO         &DMNDSET = NO
&NUMCLAS = 8          &SMFRSIZ = 228
&NUMDA   = 2          &RPS     = NO
&NUMTGV  = 400        &OSINOPT = YES
&NUMRDRS = 2          &XBATCHC = W
&NUMINRS = 1          &PRTRANS = YES
&NUMPRTS = 2          &DEBUG   = NO
&NOPRCCW = 30         &TRACE   = 0
&NUMPUNS = 1          &OREPSIZ = 0
&NOPUCCW = 30         &FCBV    = YES
&AUTORDR = YES        &NUMLNES = 2
&NUMBUF  = 15         &NUMTPBF = 2
&BUFSIZE = 1960       &TPBFSIZ = 400
&NUMOACE = 2          &NUMRJE  = 2
&NUMWTOQ = 10         &NUMTPRD = 2
&MAXJOBS = 200        &NUMTPPR = 2
&MAXXEQS = 3          &SPOLMSG = 20
&MAXPART = 3          &BSHTAB  = YES
&MAXCLAS = 8          &BSHPRES = NO
&NUMDDT  = 30         &USASCII = NO
&PRIRATE = 3          &BSC2770 = YES
&WTRCLAS = HAQ        &BSC2780 = YES
&WCLSREQ = **R        &BSCCPU  = NO
&TIMEOPT = 4          &NUMTPPU = 2
```

The storage requirement would be computed as follows:

```
DAMAP = 2 * 50 = 100

DCT   = 4 * ((8 + 49) / 4) = 56

SHASP = 26,000 + 544 + 36 + 672 + 1376 + 776 + 168

        + 30,720 + 2752 + 1400 + 4800 + 1820 + 600 + 60

        + 1110 + 1850 + 108 + 96 + 26 + 609 + 0 + 306

        + 0 + 0 + 0 + 0 + 0 + 222 + 0

        + (1088 + 504 + 0 + 672 + 848 + 848 + 57 + 194 + 0 + 0 + 5840)

      = 86,214 bytes.
```

HASP FIXED STORAGE REQUIREMENT

All of the HASP module (size described previously) is pageable except the HASPNUC CSECT, which is long-term fixed by HASP during HASP initialization. The size of this CSECT is approximately given by the following formula:

```
4500
+ 374 if &NUMLNES ≠ 0
+ 112 if &NUMSMFB ≥ 2
+ (140 + DAMAP) if &DEBUG = YES
+ 48 if &RPS = YES
+ 88 if &PRIRATE ≠ 0
+ 88 if &TSOSTCN = YES
+ (468 + 64 * &TRACE) if &TRACE ≠ 0
+ 192 * (&NUMRDRS + &NUMINRS)
+ 272 * &NUMTPRD
+ 176 * &MAXXEQS
+ (312 + 8 * &NOPRCCW) * &NUMPRTS
+ (320 + 8 * &NOPUCCW) * &NUMPUNS
+ 304 * (&NUMTPPR + &NUMTPPU)
```

A typical configuration with RJE, SMF, debug, priority aging, TSO, RPS, 1 reader, 1 Internal Reader, 3 initiators, 2 printers, 1 punch, and 4 active TP lines each with 3 active functions would require 11,546 bytes or 3 fixed pages.

HASP REGION REQUIREMENT

The previous formula for the size of the HASP module (SHASP) is imprecise because of 4K boundary alignments in the buffer pool which are difficult to express in a simple formula. If the HASP module size is precisely known (the LINKEDIT of HASP will give such a value), the minimum region requirement is determined by rounding this value up to a multiple of 4K, adding 8K (minimum subpools 0 and 252 required by VS2), and rounding the total up to a multiple of 64K (VS2 region allocation).

Additional space may be required for dynamic construction of additional HASP buffers (see HASPGEN parameter &NUMBUF). Any of the rounding space mentioned previously will be used for this purpose.

REFERENCE LISTING OF HASPJCL

This section contains a reference listing of the source module HASPJCL which is printed and punched during a complete HASPGEN, as described under Standard Complete HASPGEN Process. The module contains four sample jobs for use when installing HASP, as described under Installing HASP in the System.

SAMPLE JOB HASPSVC

```
//HASPSVC  JOB (0000,0000),'INSTALL HASP SVC',MSGLEVEL=1                00020000
//LKED    EXEC PGM=IEWL,PARM='XREF,LET,LIST,NCAL,REGION=192K            00200000
//HASPOBJ   DD DSNAME=SYS1.HASPOBJ,DISP=SHR                             00220000
//SYSUT1    DD UNIT=SYSDA,SPACE=(CYL,(10,5))                            00260000
//SYSLMOD   DD DSNAME=SYS1.NUCLEUS,DISP=OLD                             00280000
//SYSPRINT  DD SYSOUT=A                                                 00340000
//SYSLIN    DD *                                                        00360000
         INSERT IEAANIP0                                              00380000
         INSERT IEAQFX00                                              00420000
        INCLUDE HASPOBJ(HASPSVC)                                      00440000
        INCLUDE SYSLMOD(IEANUC01)                                     00460000
           NAME IEANUC01(R)                                           00480000
/*                                                                    00500000
```

SAMPLE JOB HASPROCS

```
//HASPROCS JOB (0000,0000),'INSTALL HASP PROCS',MSGLEVEL=1            00660000
//PROCS    EXEC PGM=IEBUPDTE,PARM=NEW                                 00680000
//SYSPRINT  DD SYSOUT=A                                               00700000
//SYSUT2    DD DSNAME=SYS1.PROCLIB,DISP=SHR,DCB=LRECL=80              00720000
//SYSIN     DD DATA                                                   00740000
./          ADD NAME=HASP,LIST=ALL                                    00760000
./       NUMBER NEW1=20000,INCR=20000                                 00780000
//IEFPROC EXEC PGM=HASP,REGION=128K                                   01020000
//OLAYLIB   DD DSNAME=SYS1.HASPOLIB,DISP=SHR                          01040000
./          ADD NAME=HOSRDR,LIST=ALL                                  01320000
./       NUMBER NEW1=20000,INCR=20000                                 01340000
//IEFPROC EXEC PGM=IEFIRC,REGION=52K,                                C01360000
//             PARM='00103000100125205011SPOOL                       C01380000
//                   BPPTTTOOOMMMIIICCCRLSSSSSSSS                     01400000
//IEFRDER   DD UNIT=00C,DISP=OLD,                                    C01420000
//             DCB=(RECFM=F,LRECL=80,BLKSIZE=80,BUFNO=1)             01440000
//IEFPDSI   DD DSNAME=SYS1.PROCLIB,DISP=SHR                          01460000
//IEFDATA   DD UNIT=SYSDA,VOLUME=REF=SYS1.LINKLIB,                   C01480000
//             SPACE=(80,(200,200),RLSE,CONTIG),DISP=OLD,            C01500000
//             DCB=(DSORG=PS,RECFM=FB,LRECL=80,BUFL=80,BLKSIZE=80)   01520000
./       ENDUP                                                      01640000
/*                                                                  01660000
```

SAMPLE JOB HASPHASP

```
//HASPHASP JOB (0000,0000),'INSTALL HASP PROGRAM',MSGLEVEL=1          01680000
//SCRATCH EXEC PGM=IEHPROGM                                          01700000
//SYSPRINT  DD SYSOUT=A                                              01720000
//OLAYLIB   DD UNIT=SYSDA,VOLUME=SER=ZZZZZZ,DISP=OLD                 01740000
//SYSIN     DD *                                                     01760000
       UNCATLG DSNAME=SYS1.HASPOLIB                                  01780000
       SCRATCH DSNAME=SYS1.HASPOLIB,VOL=SYSDA=ZZZZZZ,PURGE           01800000
/*                                                                   01820000
//OBLD    EXEC PGM=HASPOBLD                                          01840000
//STEPLIB   DD DSNAME=SYS1.HASPMOD,DISP=SHR                          01860000
//SYSIN     DD *,DCB=BLKSIZE=80                                      01880000
/*                                                                   01900000
//SYSOBJ    DD DSNAME=SYS1.HASPOBJ(HASPNUC),DISP=SHR                 01920000
//          DD DSNAME=SYS1.HASPOBJ(HASPRDR),DISP=SHR                 01940000
//          DD DSNAME=SYS1.HASPOBJ(HASPXEQ),DISP=SHR                 01960000
//          DD DSNAME=SYS1.HASPOBJ(HASPWTR),DISP=SHR                 01970000
//          DD DSNAME=SYS1.HASPOBJ(HASPPRPU),DISP=SHR                01980000
//          DD DSNAME=SYS1.HASPOBJ(HASPACCT),DISP=SHR                02000000
//          DD DSNAME=SYS1.HASPOBJ(HASPMISC),DISP=SHR                02020000
//          DD DSNAME=SYS1.HASPOBJ(HASPCON),DISP=SHR                 02040000
//          DD DSNAME=SYS1.HASPOBJ(HASPRTAM),DISP=SHR                02060000
//          DD DSNAME=SYS1.HASPOBJ(HASPCOMM),DISP=SHR                02080000
//          DD DSNAME=SYS1.HASPOBJ(HASPINIT),DISP=SHR                02100000
//SYSLIN    DD DSNAME=&&TEMP,UNIT=SYSSQ,DISP=(NEW,PASS),             C02120000
//             SPACE=(400,(400,50)),DCB=BLKSIZE=400                  02140000
//OLAYLIB   DD DSNAME=SYS1.HASPOLIB,UNIT=SYSDA,VOLUME=SER=ZZZZZZ,   C02160000
//             DISP=(NEW,CATLG),LABEL=EXPDT=99366,                  C02180000
//             SPACE=(1280,70,,CONTIG)                              02200000
//SYSPRINT  DD SYSOUT=A,DCB=BLKSIZE=121                              02220000
//LKED    EXEC PGM=IEWL,PARM='LIST,XREF',REGION=192K,COND=(4,LT,OBLD)02240000
//SYSUT1    DD DSNAME=SYS1.UT3,DISP=OLD                              02280000
//SYSLMOD   DD DSNAME=SYS1.LINKLIB,DISP=OLD                          02300000
//SYSPRINT  DD SYSOUT=A                                              02320000
//SYSLIN    DD DSNAME=&&TEMP,DISP=(SHR,PASS)                         02340000
//          DD *                                                     02360000
         ORDER HASPNUC(P),HASPOTAB                                   02380000
          PAGE HASPMISC,HASPINIT                                     02400000
          NAME HASP(R)                                               02420000
/*                                                                   02480000
```

SAMPLE JOB HASPOOLS

```
//HASPOOLS JOB (0000,0000),'ALLOCATE SPOOL SPACE',MSGLEVEL=1          02500000
//SCRATCH EXEC PGM=IEHPROGM                                          02520000
//SYSPRINT  DD SYSOUT=A                                              02540000
//SPOOL1    DD UNIT=SYSDA,VOLUME=SER=SPOOL1,DISP=OLD                 02560000
//SPOOL2    DD UNIT=SYSDA,VOLUME=SER=SPOOL2,DISP=OLD                 02580000
//SYSIN     DD *                                                     02600000
       SCRATCH VTOC,VOL=SYSDA=SPOOL1,PURGE                           02620000
       SCRATCH VTOC,VOL=SYSDA=SPOOL2,PURGE                           02640000
/*                                                                   02660000
//ALLOCAT EXEC PGM=IEFBR14                                           02680000
//SPOOL1    DD DSNAME=SYS1.HASPACE,VOLUME=SER=SPOOL1,               C02700000
//             DISP=(NEW,KEEP),LABEL=EXPDT=99366,                   C02720000
//             UNIT=2314,SPACE=(ABSTR,(3998,2))                      02740000
//SPOOL2    DD DSNAME=SYS1.HASPACE,VOLUME=SER=SPOOL2,               C02760000
//             DISP=(NEW,KEEP),LABEL=EXPDT=99366,                   C02780000
//             UNIT=3330,SPACE=(ABSTR,(7674,2))                      02800000
```

INTERNAL READER

A procedure exists in HASP to allow the introduction of jobs directly into the HASP job stream from any other program operating in the system. The following subsections describe techniques to accomplish this.

PROCEDURE FOR USING THE HASP INTERNAL READER

Passing jobs to HASP through the Internal Reader is accomplished by writing cards to a pseudo 2520 Card Punch device. Standard OS QSAM PUT or BSAM WRITE macros may be used to write the cards. The information, instead of being physically punched into a real 2520 Card Punch, will be passed to the normal HASP reader for insertion into the HASP job queue. The last job must be followed by a card with an end-of-file indicator (/*EOF in columns 1-5). The end-of-file card is used to free the last job, allowing it to be scheduled for execution.

JCL CONSIDERATIONS

Since any resident (nonswappable) system or user task may utilize the HASP Internal Reader, the method of allocating and controlling the use of the device is via the OS Job Control Language UNIT= parameter. The number of buffers used for the Internal Reader data set should be set to 1. See the following discussion on tape input for an example of JCL to use the Internal Reader.

OS SYSGEN CONSIDERATIONS

Pseudo 2520 punch units must be specified at OS SYSGEN time. The device addresses selected as pseudo punches must be legal System/370 addresses but must not be recognized by the physical devices or control units attached to the System/370. One device should be generated for each Internal Reader, and the number of devices generated should correspond to the value of HASPGEN parameter &NUMINRS. The following card generates appropriate unit control blocks:

```
IODEVICE        UNIT=HASP-2520,ADDRESS=(301,3)
```

The devices should be descriptively named for ease of allocation. The following card names three Internal Readers:

```
UNITNAME        UNIT=(301,302,303),NAME=INTRDR
```

DELETION OF CURRENT JOB ON READER

If the submitting task determines that the JCL and/or data of the job currently being punched into the Internal Reader is incorrect, a deletion card (/*DEL in columns 1-5) may be punched. This will cause the job currently on the device to be deleted, as though cancelled by the operator.

TAPE INPUT TO HASP USING THE INTERNAL READER

Figure 10 shows an example of how to support tape input to HASP, using the Internal Reader. Two members (TPE and EOF) are added to the system PROCLIB. Input is initiated by the operator command S TPE (or S TPE,cuu if a specific tape address is to be used). The operator may change or add any JCL parameters on the IEFRDER statement. For example, S TPE,,LABEL=(1,SL), VOL=SER=ABC123, DSN=FILE1 could be used to read input from a labelled tape.

The two steps are necessary since most input tapes don't have the required /*EOF card as the last record. A specific device address is coded for the Internal Reader in both steps to ensure that the two steps use the same device. It is recommended that this be the highest address assigned to the name INTRDR to minimize allocation conflict with tasks that use this name.

This technique can be adapted to input from other device types, e.g., disk or 96-column card reader. Also, a facility to start jobs by operator command can be developed by placing the jobs in PROCLIB (each ended by /*EOF) and reading them with a single step IEBGENER (whose input member is a JCL symbolic parameter that can be changed by the operator).

```
//HASPTPE  JOB (0000,0000),'INSTALL TAPE SUPPORT',MSGLEVEL=1
//PROCS    EXEC PGM=IEBUPDTE,PARM=NEW
//SYSPRINT  DD SYSOUT=A
//SYSUT2    DD DSNAME=SYS1.PROCLIB,DISP=SHR
//SYSIN     DD DATA,DLM='/$'
./         ADD NAME=TPE,LIST=ALL
./       NUMBER NEW1=20000,INCR=20000
//IEFPROC EXEC PGM=IEBGENER
//SYSIN     DD DUMMY
//SYSPRINT  DD DUMMY
//SYSUT2    DD UNIT=303,DCB=(RECFM=U,BLKSIZE=80,BUFNO=1)
//SYSUT1    DD DDNAME=IEFRDER
//IEFRDER   DD UNIT=2400,VOL=SER=TPE,LABEL=(,NL),DISP=OLD,
//             DCB=(RECFM=FB,LRECL=80,BLKSIZE=1600)
//EOF     EXEC PGM=IEBGENER
//SYSIN     DD DUMMY
//SYSPRINT  DD DUMMY
//SYSUT2    DD UNIT=303,DCB=(RECFM=U,BLKSIZE=80,BUFNO=1)
//SYSUT1    DD DSN=SYS1.PROCLIB(EOF),DISP=SHR
./         ADD NAME=EOF,LIST=ALL
./       NUMBER NEW1=20000,INCR=20000
/*EOF
/$
```

Figure 10. Sample Use of HASP Internal Reader For Tape Input

HASP-TSO INTERFACE

TSO may be interfaced with HASP by using the TSO background reader, HASP Internal Reader, HASP requeueing writer, and TSO command procedures. The included command procedure PRT (see Figure 11) will serve to illustrate the output technique, but it is only an example. Most installations will probably want to tailor/improve/expand the output technique to suit their specific needs.

INSTALLATION CONSIDERATIONS

HASP installation should include:

1. HASPGENing features as indicated in the following example parameters:

   ```
   &NUMINRS=1                    &TSOSTCN=YES

   &WTRCLAS=AQ                   &WCLSREQ=*R

   $$Q=*
   ```

   Note that:

   a. The Internal Reader (&NUMINRS parameter) requires that a 2520 pseudo device be generated in the Operating System for each reader.

   b. JCL for jobs with message class "Q" and SYSOUT=Q data sets will be requeued by HASP to class R after HASP has copied the JCL for output to HASP print devices.

2. Installing the command procedure PRT (see Figure 11) in an appropriate library that is accessible to TSO terminal users.

3. Adding the following card to the normal BRDR procedure:

   ```
   //HASPRDR DD UNIT=INTRDR,DCB=BUFNO=1
   ```

   BRDR must be started to activate the SUBMIT interface.

INTERFACE USAGE

In using the HASP-TSO interface:

1. Prepare jobs for background submission just as required for non-HASP systems by using the EDIT command with the CNTL descriptive qualifier. Job names must begin with the userid. Code NOTIFY=userid on the job card if desired. Code MSGCLASS=Q and SYSOUT=Q for output to be retrieved at the TSO terminal.

2. To submit a data set containing one or more jobs, use the command:

   SUBMIT dataset-name

3. Use STATUS and CANCEL as described for non-HASP systems. Replies will indicate the HASP or OS status of the job(s). (See appropriate TSO documentation for complete information on replies.)

4. To retrieve class Q output, use the command:

   OUTPUT (jobname) CLASS(R) PAUSE

5. To print a class Q SYSOUT at HASP REMOTE7 after inspection at the terminal, use:

   SAVE dataset-name (a subcommand of OUTPUT)

   and later after the OUTPUT command is terminated:

   PRT userid.dataset-name.OUTLIST AT(REMOTE7)

```
./          ADD NAME=PRT,LIST=ALL
./       NUMBER NEW1=20000,INCR=20000
PROC 1 DSN AT(LOCAL) STAT(OLD) DISP(DELETE)
EDIT 'SYS1.PROCLIB (DUMMY)' CNTL
10 //BPRT JOB MSGLEVEL=1
20 /*ROUTE  PRINT &AT
30 // EXEC PGM=IEBPTPCH
40 //SYSPRINT DD SYSOUT=A
50 //SYSUT1 DD DSN=&DSN,DISP=(&STAT,&DISP.)
60 //SYSUT2 DD SYSOUT=A
70  PRINT PREFORM=A
SAVE BPRT
END
SUBMIT BPRT
DELETE BPRT.CNTL
END
./          ADD NAME=DUMMY,LIST=ALL
(Blank card)                                                    00000010
```

Figure 11. Sample TSO Command Procedure For Output To HASP

EXECUTION BATCH SCHEDULING

This feature is a modification of normal HASP scheduling of jobs into logical partitions for execution by OS. Execution Batch Scheduling allows the system to realize performance improvement by avoiding unnecessary OS job management overhead between jobs or transactions processed by an appropriate batch processing program; however, the feature maintains the flexibility of having these jobs or transactions submitted to HASP independently: coming from possibly different input sources, having different printed and punched output routing, and with separate accounting.

BATCH PROCESSING PROGRAM CHARACTERISTICS

The processing programs to be used with the HASP Batch Scheduling feature may cover a wide variety of application areas such as:

1. Compile and go debugging compilers
2. File inquiry programs
3. Hardware or software system emulators.

However, a particular program used in the batch scheduling mode must have certain characteristics:

1. It must read all user input from a single sequential data set.
2. It must recognize a standard OS JOB card, or its own control card, to determine the beginning of a job.
3. It must recognize a standard OS null JCL card (// followed by 78 blanks), or its own control card, to determine the end of a job.

The batch processing program will receive an end-of-file condition when a card with $$ in columns 1 and 2 is read while processing a job. The program may continue to the next logical subfile by simply resetting appropriate bits in OS I/O control blocks and continuing reading or by CLOSEing the data set. The data set may then be reOPENed to continue reading at the card following the $$ card.

It is desirable that the program process jobs or transactions of relatively short duration. If not, the saving in OS job management overhead between successive jobs may not be a large enough percentage of total job execution time to justify use of this feature.

SUBMISSION OF BATCH JOBS

To use a batch processing program under the Batch Scheduling feature of HASP, the user simply constructs jobs as described in the following paragraphs.

The first card of each job should be a standard HASP/OS JOB card, which includes a CLASS=x parameter, where x is the class (installation defined) indicating which batch program is to process the job. The accounting field is interpreted by HASP, just as for nonbatch jobs.

All HASP control cards (e.g., /*JOBPARM, /*ROUTE) are effective with batching jobs except /*OUTPUT, which is ignored.

No other JCL is used. All other cards should be control cards, source cards, data cards, etc., as required by the batch program. These will be read by the batch program, just as if they had been placed in a DD * data set and the batch program had been invoked by standard JCL. If the batch program requires it, each logical subfile should be terminated by a card with $$ in columns 1 and 2.

BATCH SCHEDULING PROCESS

Special actions take place when HASP recognizes that a batch job has been selected for execution.

If the batch program is not already active in the logical partition for which the job was selected, HASP generates and sends to the OS reader/interpreter an internal job which uses JCL from SYS1.PROCLIB to invoke the program. The entire user job as submitted (JOB card, all other user input), followed by two null JCL cards added by HASP, is allocated as an input data set to the batch program.

If the batch program is already active and simply waiting for another job, HASP makes the input data set allocation as above, and processing begins immediately, without any use of OS job management.

Job termination is detected by the batch program, when it reads its own ending control card or one of the null JCL cards added by HASP. After writing any remaining SYSOUT data for the completed job, the batch program attempts to read ahead in its input file for another job. HASP detects this condition, temporarily forces the batch program into a wait state, and accomplishes job termination actions for the job (flushes output buffers, releases input SPOOL space, queues job for printing, etc.). The batch program remains in the logical partition.

When a batch program is waiting in a logical partition, HASP job selection is altered. Instead of scanning for all classes eligible to execute in that partition, HASP first tries to start a job with the same

class as the batch program still in the partition. If successful, processing can begin immediately as described previously.

If no jobs of the same class are available to execute, all other job classes of the partition are scanned in order. If a job is found, HASP internally cancels the batch program, and normal scheduling using OS job management takes place. If no jobs of the other classes are found, the partition remains idle, awaiting availability of a job in any of its classes. If a job becomes available in the class of the batch program still in the partition, processing begins immediately.

If a batch program ends (ABEND or normal return to OS), HASP detects this as a nonbatch termination in the partition. OS job management will be used to reinvoke the batch program when another job for its class is selected.

Use of the operator commands $PI or $PIn will cause HASP to cancel an idle batch program when the partition(s) become drained.

INSTALLING BATCH SCHEDULING

The batching feature is included in HASP by setting the &XBATCHC HASPGEN parameter equal to a list of job classes to be processed by the rules described previously. The &XBATCHN parameter should also be set (see descriptions of these two parameters in HASPGEN Parameters).

Each batch class should be used to represent one batch processing program. Each batch class should be made eligible to execute in one or more logical partitions, by setting the &CLS(n) HASPGEN parameters or by use of the $T operator command.

The batch processing program for each class must be available in loadable form somewhere in the system.

For each combination of batch class and logical partition in which it may execute, there must be a procedure in SYS1.PROCLIB whose name is "nnnnncid"; where nnnnn are the five characters assigned to &XBATCHN, c is the particular batch job class (one of the list assigned to &XBATCHC), and id is the 1- or 2-character logical partition identification, set by the parameters &PID(n). These procedures actually call the batch processing programs for each class and define all data sets, other than the user input data set.

The procedures may either be single step, or they may have preliminary steps before the single step that processes the user jobs (stepname GO). The processing program invoked by this step must read its input from a ddname SYSIN, or the procedure must refer to DDNAME=SYSIN on a DD card whose name is used for input by the processing program. The DCB parameter BUFNO=1 should be included on any SYSOUT data sets in a procedure. This will help to ensure that HASP has actually received all

output produced by the batch program for a job or transaction, when the program is suspended while trying to read ahead to the next job.

All SYSOUT classes that indicate SPOOLing by HASP will be changed to either class A or class B, as specified by the HASPGEN parameter $$x. Forms, FCB=, UCS= on SYSOUTs SPOOLed by HASP will be ignored. If OS output SPOOLing is used with any SYSOUT (see HASPGEN parameter $$x), the output is not queued for the OS writer until the batch processing program terminates (not necessarily when any batch job terminates).

If a given batch class is eligible to execute in more than one logical partition, the requirement for a separate procedure name for each class-partition combination may be satisfied by alias names of a single procedure or by actual separate procedures, which may specify different region sizes, work files, etc.

The following example shows the internal job that HASP would generate to initially load a program to process batch class X jobs, in a partition whose &PID(n)=3, assuming the default setting for &XBATCHN.

```
//$$$$$X3  JOB 1,SYS,MSGLEVEL=1
//FAKE EXEC $$$$$X3
//GO.SYSIN DD DATA,DCB=BUFNO=1
//
```

This job calls a procedure. The following is an example of a procedure that an installation might use for a simple file inquiry program, which reads inquiry input from SYSIN, interrogates a file, and prints responses to SYSPRINT.

```
//GO EXEC PGM=FINDPART
//SYSPRINT DD SYSOUT=A,DCB=(BLKSIZE=121,BUFNO=1)
//PARTFILE DD DSN=PARTFILE.MASTER,DISP=SHR
//SYSUDUMP DD SYSOUT=A
```

This procedure would be placed in SYS1.PROCLIB with the name $$$$$X3.

GENERATING MORE THAN FIFTEEN LOGICAL PARTITIONS

When &MAXPART is specified greater than 15's, values for the HASPGEN parameters &PID(n), &PRI(n), &OSC(n), and $CLS(n) for those logical partitions above 15 must be specified as updates to source module HASPXEQ as follows:

Name	Op	Operand	Serial
&PID(nn)	SETC	'id'	X0272nn0
&PRI(nn)	SETA	p	X0312nn0
&OSC(nn)	SETC	'o'	X0352nn0
&CLS(nn)	SETC	'c...'	X0392nn0

where nn is the number of the logical partition (16-63), and the contents of the operands are chosen as described previously for those HASPGEN parameters.

All update cards must be arranged in ascending order by serial number and applied using the HASPGEN update facility described under Generating A HASP System. Figure 12 shows an example of the update, assuming &MAXPART=18.

```
Columns
         1       1                                                  7       8
1        90      6                                                  3       0
./       CHANGE  NAME=HASPXEQ
&PID(16) SETC    '16'                                               X0272160
&PID(17) SETC    '17'                                               X0272170
&PID(18) SETC    '18'                                               X0272180
&PRI(16) SETA    7                                                  X0312160
&PRI(17) SETA    7                                                  X0312170
&PRI(18) SETA    7                                                  X0312180
&OSC(16) SETC    'A'                                                X0352160
&OSC(17) SETC    'B'                                                X0352170
&OSC(18) SETC    'C'                                                X0352180
&CLS(16) SETC    '98765432'                                         X0392160
&CLS(17) SETC    'PQRSWXYZ'                                         X0392170
&CLS(18) SETC    '76543210                                          X0392180
```

Figure 12. Sample Update For Eighteen Logical Partitions

MULTIPLE DEVICES ON MULTI-LEAVING REMOTES

If a HASP System includes MULTI-LEAVING RJE support (&NUMLNES > 0 and &BSCCPU=YES), and if any remote terminal to be supported has multiple devices (i.e., more than one reader, printer or punch), then the following considerations should be reviewed before performing HASPGEN and RMTGEN for that configuration.

RMTGEN CONSIDERATIONS

The discussion of RMTGEN parameters describes how to specify support for a second (or third, etc.) reader, printer, or punch, when performing RMTGEN for the various types of MULTI-LEAVING remote work station programs.

HASP PROCESSOR CONSIDERATIONS

It may be necessary to increase the value(s) of HASPGEN parameters &NUMTPPR, &NUMTPPU, and &NUMTPRD to allow concurrent operation of all remote devices in the system.

For example, if &NUMLNES=3 and the default value &NUMTPPR=&NUMLNES is taken, then the HASP System can only support three concurrent remote print operations. If all three lines are active and one of the three active remotes has two printers, then unless &NUMTPPR is increased to four, one of the four possible concurrent remote print operations may be delayed until a print operation on another remote comes to the end of a job.

The decision to increase these parameters, and by how much, depends on the total remote configuration and an estimate of the number of active remotes that will usually be at the same stage of job processing.

HASP REMOTE DEVICE CONSIDERATIONS

HASP generates a Device Control Table (DCT) for each type of device (reader, printer, and punch) on each remote terminal known to HASP (RMT01 through RMTnn where &NUMRJE=nn). If a remote terminal has more than one of each type of device, a DCT for each such additional device must be generated. Each additional DCT must be specified on a card of the following format:

```
$RMTDCT type,device                          -serial-
```

Values for this card should be chosen from the following list:

	type	device	serial
readers	RJR	RMnn.RDm	N0730nnm
printers	RPR	RMnn.PRm	N0732nnm
punches	RPU	RMnn.PUm	N0736nnm

where nn is the remote number (same as in the RMTnn HASPGEN parameters but with a leading zero omitted in device) and m is the device number (must be 2 or greater, up to a maximum of 7). All cards describing additional devices for remotes must be placed in ascending order by serial number and must be added to the source module HASPINIT, using the HASPGEN Update facility described under Generating a HASP System. Figure 12 shows how to generate a second printer DCT for remote 2 and a second reader DCT for remote 5.

```
Columns
         1     1                                        7     8
1       90     6                                        3     0
./       CHANGE NAME=HASPINIT
          $RMTDCT RJR,RM5.RD2                          N0730052
          $RMTDCT RPR,RM2.PR2                          N0732022
```

Figure 13. Sample Update For Generating Second Printer and Reader

HASP 2770 AND 3780 RJE SUPPORT

2770 CONFIGURATION

The basic 2770 with standard keyboard and either EBCDIC or USASCII code is supported.

Optional supported devices are the 2502 Card Reader, 2213 Printer, 2203 Printer, and 545 Output Punch. The Printer must be attached to OUTPUT PRINTER. The card punch must be attached to OUTPUT 2.

EBCDIC Transparency, Printer Horizontal Format Control, Space Compression/Expansion, and all three buffer sizes (128, 256, 512) are supported by HASP programming. The Multipoint Data Link Control feature must not be present. All other devices and features may be attached but are either unaffected by programming or are not supported.

I/O FORMATS

Although HASP formally supports only the keyboard, card, and printer I/O devices listed previously, the basic design of the IBM 2770 (i.e., media formats independent of transmission format) may make it possible for individual installations to use other I/O devices. This must be done only after careful analysis, design, and testing (by the customer and local IBM Representative) to establish the feasibility of the proposed device usage in the customer's environment. Refer to SRL GA27-3013, especially pages 2772-9, CU-2,3, and appropriate device sections. The following descriptions of input and output transmission block handling will also aid in analysis of other device usage possibilities.

Input

Input blocks to HASP from any device on the 2770 are transformed into 80-character records of an OS job stream, according to one of the following two rules:

1. If the block is nontransparent, it is interpreted as one or more records of 80 or fewer data characters, each ended by an IRS character, which does not become part of the record processed by OS. Compressed blanks, indicated by the IGS character, are detected and expanded prior to processing by OS, if the HASPGEN parameter &BSHPRES=YES.

2. If the block is transparent, it is interpreted as one or more records of exactly 80 data characters. No record ending characters are recognized.

Transparent and nontransparent input blocks may be mixed, in any order, in any job or series of jobs transmitted to HASP. Proper handling of compressed blanks in nontransparent input blocks and proper handling of transparent input blocks is not dependent on the setting of the RMTnn HASPGEN parameter describing the particular terminal.

Therefore, to use other input devices, the input medium and device must conform as pointed out in the previous discussion. The device may be connected to any INPUT position, as long as that position is switched on before transmission is initiated. Input which does not conform to these rules will cause unpredictable deblocking when received by HASP and probable error messages or incorrect results when processed by OS or the user's program.

If the input medium/device cannot produce an IRS record ending character, or if control characters are used as data, the transmission must be unblocked and/or possibly transparent. The processing program must handle as data any record ending character, (other than IRS) the medium/device may produce.

An input medium other than cards may not be suitable for the preparation and transmission of OS JCL cards (e.g., //ANY JOB ... up to //SYSIN DD *), which are required to precede data in an OS input job stream. The keyboard may be used to transmit such cards, followed by data from the other device, using an operational procedure similar to that described for the keyboard and card reader in the 2770 appendix to the HASP Operator's Guide.

Output

Output from HASP to the 2770 is in two forms: one intended for printing, the other for punching cards. These outputs are produced during OS execution, via the disposition SYSOUT= on DD cards. The decision to produce printed or punched output from a given SYSOUT class is controlled for the entire system by the HASPGEN parameters $$x, described under HASPGEN parameters.

Output block maximum length is 128, 256, or 512 bytes, as indicated by the RMTnn HASPGEN parameter. Output records do not span transmission block boundaries. Each printed or punched output job is ended by an EOT transmission.

Printed Output

Printed output is always sent as nontransparent blocks. All data characters less than X'40' are translated to X'00', or if the &PRTRANS parameter is set to YES, all nonprinting characters are translated to X'40'. The first block of a job contains the component selection character DC1. One or more variable length records are transmitted in each block. Each record begins with the 2-character ESC x carriage control sequence, has data characters up to the maximum specified for printer width in the RMTnn parameter, and ends with the IRS character.

If indicated by parameter RMTnn (and supporting settings of &BSHTAB and &BSHPRES), blanks are compressed and encoded using either the HT or IGS characters. Encoding by HT sets electronic tabs, every 10 columns beginning with column 11; this can be changed by altering internal assembly variable &HTDIST.

The listing content for each job is the same as for all jobs printed by HASP: beginning and ending separator pages (number of separator lines controlled for all remotes in the system by the $TPIDCT parameter), HASP Job Log, OS system messages (JCL, etc.), and any printed SYSOUT data sets.

It is probably not very practical to direct printed output to another device for output data purposes, due to the inclusion of separator pages, messages, etc. The material could be directed to another medium (e.g., paper tape) for later listing offline or on another machine; however, because only the printer can be attached to the OUTPUT PRINTER position, HASP would have to be modified to use other than DC1 for print component selection. This would be a trivial 1-card modification if all 2770s in the system were configured and used the same way but would be more difficult if not.

Punched Output

Punched output is sent as transparent blocks if the RMTnn parameter indicates that the Transparency feature is present. In this case, the component selection character DC2 is transmitted alone in a nontransparent block, at the beginning of the job. All other blocks are transparent and contain one or more records of exactly 80 data characters, without any record ending characters.

If Transparency is not indicated by the RMTnn parameter, all punched output data characters less than X'40' are translated to X'00'. Only nontransparent blocks are transmitted, with the DC2 in the first block. Each block contains one or more variable length records. Blanks are compressed and encoded using the IGS character, if indicated by RMTnn (and supporting &BSHPRES). Each record contains 80 or fewer data characters and ends with the IRS character.

Punch job content is: separator card, punched SYSOUT data sets, and one blank card at the end of the job. Blank cards may be produced at the

end of each SYSOUT data set by some OS access methods, but these are simply transmitted as data by HASP. A second blank card at the end of each job is produced at the 545 Output Punch by a mechanical eject when EOT is received.

Punched output, except for separator and terminal blank cards, is pure data output whose content is controlled completely by the application program execution. Therefore, it may be practical to direct punched output to another device connected to the OUTPUT 2 position or to other positions (if HASP is appropriately modified to use other than DC2 for punch component selection). If the nontransparency, variable length record, form of punched output described previously is considered more desirable for the output device in question, HASP may be forced to produce it, by omitting Transparency in the RMTnn parameter (even if the 2770 has the Transparency feature). This will not prevent the 2770 from transmitting transparent input blocks to HASP.

3780 SUPPORT

The previous description of 2770 support applies to the 3780 also, with minor exceptions. The 3780 is assumed to have standard 512-byte buffers, card reader, printer, but no keyboard or card punch. Component selection characters are not sent to the 3780. Although Transparency, Horizontal Format, and Compression features are standard, their use for output is controlled by the RMTnn parameter, as with the 2770.

3211 FORMS CONTROL BUFFER ADDITIONAL LOADS

Installations using HASP with 3211 printers may want to add carriage tape images to the standard images provided. Each image is named by a 1- to 4-character alphanumeric field comprised of letters A-Z and numbers 0-9. The alphameric '1bbbb' is reserved to allow the operator to force single-spacing and the 'Vbbbb' is reserved for the operator-variable FCB load. HASP supplies images for '6bbbb', '8bbbb', and 'Ubbbb'.

ADDING AND CHANGING FCB LOADS

The mechanism for defining Forms Control Buffer loads in HASP is the $FCB macro. To add or change an FCB image, the installation system programmer:

1. Selects an FCB image whose label will be 'FCB' followed by a 1- to 4-character ID (e.g., FCBLCL6 for local at 6 lines per inch)
2. Codes a $FCB macro with the image label and an operand field that describes the FCB image
3. Assigns it a card sequence number from the range of numbers given later in this section
4. Includes the card in his HASPGEN update deck following a CHANGE card for module HASPPRPU as described previously
5. Does a HASPGEN to create new source for HASPPRPU
6. Reassembles HASPPRPU
7. Executes HASPOBLD to create a new HASP load module and overlay library.

The format of an FCB macro is described in the HASP macro appendix to the HASP Logic manual.

FCB LOADS PROVIDED BY HASP

HASP provides three FCB loads that can be called by the characters '6', '8', and 'U'.

The '6' image is designed for 11-inch forms. Channel 1 is punched at line 1, channel 2 at line 7, channel 3 at line 13, and so forth through

channel 8. Channel 10 is at line 49, channel 11 at line 55, channel 12 at line 61, and channel 9 at line 63. The $FCB macro is:

```
FCB6 $FCB 6,66,1-1,2-7,3-13,4-19,5-25,6-31,7-37,8-43,10-49,11-55,
     12-61,9-63
```

The '8' image is designed for 8-1/2 inch forms, at 8 lines per inch. The punches are the same as for the '6' image, except that channel 9 is at line 64. The $FCB macro is:

```
FCB8 $FCB 8,68,1-1,2-7,3-13,4-19,5-25,6-31,7-31,8-43,10-49,11-55,
    12-61,9-64
```

The 'U' image specifies only carriage channel 1 at line 1; other carriage channels are filled in by the $FCB macro to prevent forms runaway. The $FCB macro is:

```
FCBU $FCB 6,66,1-1
```

NOTE: In order to prevent misaligned forms, carriage channel 1 must always be defined at FCB address 1.

RECOMMENDED CARD SEQUENCE NUMBERS

It is recommended that additional FCB images be assembled using card sequence numbers P7028001-P7031999.

HASP-SMF INTERFACE

If a HASP System is generated with &NUMSMFB greater than or equal to 2, System Management Facilities (SMF) records will be written, and the IEFUJP user exit will be invoked if EXT=YES was specified. For a further explanation of SMF refer to the SRL GC35-0004, OS/VS System Management Facilities (SMF).

If &NUMSMFB is equal to or greater than 2, HASP will write two accounting record types, 6 and 26. The HASP Output Processor will create a type 6 record for each unit of work processed. Purge processing (HASPMISC) will create a type 26 record for each job purged. If SMF user exits are to be taken (EXT=YES), the HASP Purge Processor will also save the common exit parameter area for the IEFUJP exit. Subroutines within HASPNUC will obtain HASP SMF buffers and will POST the HASPACCT subtask.

If &NUMSMFB is equal to or greater than 2, HASP will write four subsystem record types: 43, 45, 47, and 48. HASPINIT will create a type 43 record when HASP is started, and HASPCOMM will create a type 45 record when HASP is stopped. HASPRTAM will create types 47 and 48 records when lines are started or stopped or when remotes SIGNON or SIGNOFF.

The HASP SMF Writer routine, HASPACCT, is a subtask of HASP. It is ATTACHed by HASPINIT and is DETACHed for $PHASP in HASPCOMM. The HASP SMF Writer routine will call the IEFUJP exit, when necessary, and will call the OS SMF writer to write all HASP SMF records. For a further description of HASP SMF records, see the Data Areas section of the HASP Logic manual.

HASPGEN CONSIDERATIONS

Two HASPGEN parameters pertain to SMF processing within HASP, &NUMSMFB and &SMFRSIZ. &NUMSMFB determines the number of buffers generated to hold HASP SMF records and copies of the common exit parameter area (for the IEFUJP exit). &SMFRSIZ must be as large as the maximum HASP SMF record size or must be large enough to hold one copy of the common exit parameter area, whichever is greater.

There are several ways to include an IEFUJP exit routine into the HASP load module. An update to IEFUJP CSECT in HASPACCT may be created and included in HASPGEN. Before running the HASPHASP job, the user can create an object module of the IEFUJP exit routine and can add a DD card to the SYSOBJ concatenation (before the card for the HASPACCT routine).

After HASPGEN, HASP can be re-LINKEDITed, and the dummy IEFUJP CSECT can be replaced with the users IEFUJP. Refer to the sample job HASPHASP in this manual for an example of the SYSLIN input cards.

SMF CONSIDERATIONS

After the first IEFUJV user exit, HASP will change the reader start time and date in the common exit parameter area to reflect when HASP read the job from an input device. If the user wrote any user SMF record(s) at the first IEFUJV exit, the time stamp reflects interpreter start time and can not be correlated to other SMF records for the same job.

The reader device class and unit type fields in the type 5 SMF record will reflect the physical reader UCB as opposed to a pseudo device UCB. HASP will place the time and date HASP stopped physically reading a job and the job class from the job card into the type 5 SMF record. If the SMFDEFLT parameter EXT was specified as NO, HASP will not change the reader stop time and date or the job class in the type 5 record. In this case, reader stop time and date will reflect interpreter stop time and date, and job class will be an initiator identifier specifying the initiator that executed the job (as determined from &OSC(n) HASPGEN parameter).

In the HASP IEFUJP user exit, the common exit parameter area will not reflect any modifications made during the last IEFACTRT exit. For example, if the user changed the user communication field in the IEFACTRT exit at job termination, this change will not be reflected in the user communication field at the IEFUJP exit. If HASP receives a cancel command before a job is sent to VS2 for execution, the step number field in the common exit parameter area will be binary zero in the IEFUJP exit, indicating that the job was never executed by VS2.

The following SMF fields have altered interpretation when jobs are executed under HASP control with VS2:

1. In types 4 and 5 SMF records, the SYSIN count fields will be zero when HASP does the input SPOOLing. The type 26 record provides a total job input card count.

2. In the type 4 record, the EXCP count field for pseudo devices is equal to the number of logical records transferred.

3. In types 4 and 5 records, the priority fields will contain the dispatching priority as set by the HASPGEN parameter &PRI(n) for the initiator that executed the job. Step priority may be set lower by DPRTY on the EXEC card if the initiator's &PRI(n) value is not subject to APG. HASP job purge record, type 26, provides the execution selection priority.

4. In the type 5 record the SYSOUT class indicator will be zero for output data sets SPOOLed by HASP.

GENERAL HASP RESTRICTIONS

Because of the techniques used in HASP implementation, certain features and/or functions of the Operating System may not be available or may differ in operation in a system utilizing HASP. Additionally, certain features and functions implemented by HASP may not perform in the same manner as similar functions replaced in OS or may be affected by various environmental or operating characteristics of a particular installation. The following sections indicate a partial list of these restrictions, excluding those restrictions made obvious by the general interface technique utilized by HASP.

UNSUPPORTED OS FEATURES

OS features that are not supported include:

1. All I/O operations for SYSIN/SYSOUT data SPOOLed by HASP will appear to the user as direct use of unit record devices (which do not actually exist). Therefore, a program that depends on the physical characteristics of a particular device for processing SYSIN/SYSOUT data may not function properly in a HASP environment.

2. All I/O requests for SYSIN/SYSOUT data files controlled by HASP must be made through the standard use of the EXCP macro instruction.

3. SYSIN/SYSOUT operations, which appear to programs as the direct use of unit record devices, are actually performed by simulating the function of the unit record device. In simulating the operation of these devices, certain functions of the actual device may not be accurately simulated by HASP. These include:

   a. Timing - I/O operations to the pseudo devices will not have the same timing characteristics as I/O operations to an actual device.

   b. Data Chaining - HASP does not support the Channel Command Word Data Chaining feature of System/370 when simulating unit record devices. The command chaining feature is, however, fully supported.

   c. Input/output appendages - In responding to requests for I/O operations, HASP will not enter any I/O appendage. Because of the instantaneous nature of HASP I/O operations, the use of appendages is not applicable and will be ignored, if specified.

4. The use of the Checkpoint/Restart feature of OS is, in general, inconsistent with the SPOOLing techniques utilized by HASP. In

many cases, Checkpoint/Restart will not function properly in a HASP environment. The user is responsible for verifying the compatibility of the various Checkpoint/Restart features to be used. Jobs requiring the use of unsupported features of Checkpoint/Restart may be run in a HASP environment, directly under OS, outside the control of HASP.

5. HASP does not support the continuation of the DD * or DD DATA JCL statements under any conditions.

6. HASP support of the DLM parameter on DD * and DD DATA JCL statements is compatible with OS support of this parameter, with the following exceptions:

   a. If DCB parameters are specified, they must be specified physically before the DLM specification (i.e., the DLM specification must be the last parameter on the DD statement).

   b. The apostrophe (') cannot be used as a delimiter character.

   c. HASP control cards (/*MESSAGE, /*SETUP, /*ROUTE, etc.) will not be recognized if the DLM specification is other than "/*".

   d. The DLM parameter will not override Internal Reader control cards (e.g., /*EOF) or remote device control cards (e.g., /*SIGNON). The control card will NOT be processed as in-stream data.

HASP FUNCTION/FEATURE RESTRICTIONS

Function and feature restrictions are:

1. The capability to dynamically withdraw HASP from the system and continue operation is intended, primarily, as a programming aid for the system programmers and is highly dependent on the individual operational environment. For these reasons, this function is not designed to (and may not) effect a complete withdrawal so that the previous presence of HASP is completely transparent to the host Operating System. Each installation utilizing this feature should individually verify the accuracy and completeness of the withdraw operation.

2. HASP will not operate correctly if two or more jobs being processed simultaneously by OS have identical job names. While HASP will protect against this circumstance for jobs under its control, it is the responsibility of the user to ensure that no job submitted outside of HASP control has the same job name as any job being controlled by HASP.

3. Because of the HASP/OS interface techniques and the total system control status of HASP, no provision has been made to allow processing to continue after a HASP failure. Any abnormal

termination of HASP is considered a system failure and requires a re-IPL.

4. If any SYSOUT data set contains more than 65,535 pages (or skips to any channel in the carriage tape), print positioning (either forward/backward spacing or warm start spacing) will not function after the 65,535 page (or skip) is reached.

5. The 'OUTLIM' parameter of a SYSOUT DD card is not supported by either SVS or HASP. The 'LINES' subfield of the HASP Job card accounting field should be used in conjunction with the &OUTPOPT HASPGEN parameter to limit the number of lines of output.

6. While HASP makes an attempt to enter every WTO, WTOR, and short form reply to a WTOR in the HASP Job Log of the job associated with the message, there are certain messages (e.g., DDR messages) which are not readily associated with any given job and may not be logged.

7. While HASP is programmed to recover from most catastrophic input/output errors in such a way that the impact on the installation will be minimal, it is conceivable that multiple unusual errors might occur in a time relationship so that loss of data is inevitable and complete recovery by HASP is impossible.

8. Password protection as provided by OS/VS for the message IEC301A is not a function of HASP. If reply is entered in the HASP numeric or short format, that reply will appear in the HASP Job Log. If it is necessary to enter the reply to an OS/VS request in HASP format, and undesirable to have that reply appear in the HASP Job Log, it is suggested that 'NOLOG' be specified in the HASP JOBPARM card.

HASP OVERLAY PROGRAMMING RULES

The following comments summarize the rules for coding and using "overlayable code" in HASP. All rules apply to use of any control sections created by use of the $OVERLAY macro, even if the code so produced is optionally made permanently resident as part of the overlay build process.

CREATING OVERLAY CONTROL SECTIONS

The beginning of a portion of HASP executable coding or tables to be made overlayable is indicated by the $OVERLAY macro. By convention, the name field begins with "HASP" and continues with up to four more characters. The fifth character (first after "HASP") usually indicates the Processor of which the overlayable code is a part; e.g., R for read, X for execution, P for print/punch, etc. A specific example is "HASPXJI1", the name of the first of two overlays used by the HASP Execution Processor for job initiation actions. The name coded with $OVERLAY will be defined at the first location coded by the programmer after the $OVERLAY and will be used to derive a name for the control section created.

The operands of $OVERLAY specify the priority for use of overlay resources and, in conjunction with the HASPGEN parameter &OLAYLEV, whether the code created is to be actually disk or main memory resident during HASP operation.

The $OVERLAY macro is a functional replacement for CSECT, USING, and BALR or L when creating a HASP overlayable control section. $OVERLAY creates an actual assembly control section and indicates local addressability in register BASE3. Overlay Service and Roll functions ensure that the proper base value is loaded into BASE3 when an overlay section is being used.

An overlay control section's coding may be terminated and all effects of a previous $OVERLAY canceled in one of two ways. Another overlay may be begun by a new $OVERLAY macro. Non-overlay coding may be resumed by DROPing register BASE3 and reestablishing an appropriate CSECT.

If it is desired to add more coding to a previously terminated overlay section, the actions in the following example must be performed. &xyz is a properly declared variable symbol. HASPabcd is the overlay name chosen by the programmer. Other symbols are defined in standard HASP assemblies. The second statement must be placed after the $OVERLAY defining the overlay section to be resumed, before another $OVERLAY is used.

```
HASPabcd    $OVERLAY  12,0        (original definition)
&xyz        SETC '&OSECT'
              .
              .
              .
&xyz        CSECT                 (later additional code)
            USING  HASPabcd-OACEPROG+BUFDSECT,BASE3
```

CALLING OVERLAY ROUTINES

The three executable macros $LINK, $XCTL, and $LOAD cause an overlay routine to be made available for use in addressable memory. The single operand of each of these macros gives the name of the overlay to be used, either directly or by providing (in register form) the address of a $OCON macro which gives the name. The name referenced is that used with a $OVERLAY macro to create the overlay routine. The overlay control section ($OVERLAY and following code) may be in the same or a different HASP assembly as a macro which calls it.

The $LINK and $LOAD macros must be physically placed in non-overlay CSECTs and executed only when no other overlay routine is being used, i.e., nested calling of overlays is not defined. With $LINK, program control is eventually passed to the first instruction after $OVERLAY of the called routine. The address of the caller's next instruction is saved for later return. $LOAD returns control to the next instruction after $LOAD when the routine is available in memory.

$XCTL relinquishes use of an overlay routine, previously called by $LINK or $XCTL, and calls a new overlay routine which is entered as if called by $LINK. Return address saved by the original $LINK is not altered. $XCTL must always be executed when an overlay is in use, but may physically be in an overlay routine or in non-overlay coding, subject to the requirements stated in "Coding While Using Overlay Routines."

$RETURN and $DELETE both relinquish use of an overlay routine, which must be in use when they are executed. These macros have no operands; the routine released is the only one in use at the time. $RETURN causes control to pass to the next instruction after the $LINK previously executed by the Processor from non-overlay code. $RETURN, like $XCTL, may physically reside anywhere. $DELETE must physically reside in non-overlay code and is valid only after a routine was previously called by $LOAD. Control continues following $DELETE, after use of the overlay routine has been released.

Overlay routines may be called only by HASP Processors operating under the primary HASP TCB, HASP Dispatcher, and PCE control. Overlay routines may not be called in exits from the Asynchronous Post Processor.

CODING WHILE USING OVERLAY ROUTINES

On entry to an executable overlay by $LINK or $XCTL or after loading an overlay with $LOAD, the caller's registers R0-R7 and R9-R13 are preserved. However, registers BASE3 (same as R8 or WG in unmodified HASP), LINK, R15, and the condition code are destroyed and are not later restored. While an overlay routine is being used (after the execution of $LINK or $LOAD but before the execution of $RETURN or $DELETE), the program must not alter the value of register BASE3.

Coding in an overlay routine is "covered" by local addressability provided by $OVERLAY. Coding physically outside an overlay but referring to it (usual case after a $LOAD) must be "covered" by a USING like that in the example in the section "Creating Overlay Control Sections." Other addressability (e.g., BASE1, BASE2) remains in effect if not dropped and may be used.

Program control may be transferred out of or into an overlay routine and its storage may be retrieved, as long as overlay control of that routine is in effect (has not been released by $RETURN, $DELETE, or $XCTL to a new routine) and proper addressability is maintained. References to locations in an overlay routine from physically outside the overlay at any other time are illegal.

Relocatable valued A or V type constants must not be physically coded in overlay routines. Such constants may be coded in non-overlay CSECTs and referenced from overlay routines. Relocatable A or V type literals may be coded if the literal pool containing them is not physically in an overlay routine. An A or V constant or literal containing an "un-paired" (see OS/VS - DOS/VS - VM/370 Assembler Language, GC33-4010) reference to a symbol defined in an overlay routine is always illegal, regardless of location.

When use of an overlay routine is released by $RETURN or $DELETE, only the LINK and BASE3 registers are destroyed. All other registers and the condition code are preserved as set prior to the execution of these macros.

Total size of all coding in an overlay routine must not exceed the value of the internal assembly variable &OLAYSIZ, currently set at 1024 bytes in unmodified HASP. An error message will be produced during the Overlay Build process for each routine that violates this restriction.

Index

```
&NUMDA    19,20,37,131,132,133,135
&NUMDDT    19,38,132,135
&NUMINRS    5,19,36,38,132,135,136,141,144
&NUMJOES    19,39,132,135
&NUMLNES    12,19,20,30,40,43,45,46,133,135,136,153
&NUMOACE    19,40,132,135
&NUMOSC    19,40,132,135
&NUMPRTS    19,32,34,39,41,80.1,132,135,136
&NUMPUNS    19,32,33,34,39,42,132,135,136
&NUMRDRS    19,43,132,135,136
&NUMRJE    19,20,40,43,64,134,135,153
&NUMSMFB    19,44,68,132,136,161
&NUMTGV    19,20,37,44,84,131,135
&NUMTPBF    19,45,133,135
&NUMTPPR    17,19,20,32,34,39,45,134,135,136,153
&NUMTPPU    19,20,32,34,39,46,134,135,136,153
&NUMTPRD    19,36,46,134,135,136,153
&NUMWTOQ    19,47,132,135
&NUM3800    19,20,80,80.1
&OLAYLEV    19,40,48,83,131
&OREPSIZ    19,49,133,135
&OSC(n)    19,33,50,151,152,162
&OSINOPT    19,51,75,82,132,135
&OUTPOPT    19,52
$OUTXS    19,52
&PID(n)    19,26,33,53,149,150,151,152
&PRI(n)    19,21,33,54,151,152,162
$PRIDCT    19,55,72,80.1
&PRIHIGH    19,56,57
&PRILOW    19,56,57
&PRIRATE    19,56,57,132,135,136
$PRTBOPT    19,36,57
&PRTFCB    19,57
&PRTRANS    19,58,133,157
&PRTUCS    19,58
$PUNBOPT    19,36,59
&RDR    5,6,19,59
$REPRDR    19,59
$REPWTR    19,59
&RESCORE    19,32,35,60
&RJOBOPT    19,61
RMTnn    19,23,43,62-65,72,101,102,104,117,125,128,129,153,154,
  156,157,158
$RPRBOPT    19,36,65
&RPRI(n)    19,54,57,66,67
&RPRT(n)    19,54,57,66,67,78,79
&RPS    19,67,133,135,136
$RPUBOPT    19,36,68
&SMFRSIZ    19,44,68,132,135,161
&SPOLMSG    19,20,69,101,102,125,128,134,135
&SPOOL    19,37,70,83
&STDFORM    19,70
&TIMEOPT    19,71,133,135
$TIMEXS    19,71
&TPBFSIZ    19,34,45,65,72,96,133,135
$TPIDCT    19,55,72,157
&TRACE    19,26,73,133,135,136
```

IBM Technical Newsletter

This Newsletter No. GN27-1553
Date September 15, 1976

Base Publication No. GC27-6992-0
File No. S370-36

Previous Newsletters GN25-0121

OS/VS2
HASP II Version 4
System Programmer's Guide

This technical newsletter, a part of Release 1.7 of OS/VS2 SVS, provides replacement pages for the subject publication. These replacement pages remain in effect for subsequent SVS releases unless specifically altered. Pages to be inserted and removed are:

cover – 8
19-22
37-42
53, 54
65, 66
79-82 (80.1-80.3 added)
131-134
165-166.4 (166.1-166.4 added)
167-176

Each technical change is marked by a vertical line to the left of the change.

Summary of Amendments

Support for the IBM 3800 Printing Subsystem under HASP is added. In addition, miscellaneous technical and editorial changes have been made throughout the book.

Note: Please file this cover letter at the back of the manual to provide a record of changes.

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

Printed in U.S.A.

IBM

Technical Newsletter

This Newsletter No. GN25-0121
Date May 30, 1975

Base Publication No. GC27-6992-0

Previous Newsletters None

OS/VS2
HASP II Version 4
System Programmer's Guide

This Technical Newsletter, a part of release 1 of the IBM System/370 Operating System/Virtual Storage 1, provides replacement pages for the subject publication. These pages remain in effect for subsequent OS/VS2 releases unless specifically altered. Pages to be removed/inserted are:

Cover	111,112
19, 20	137, 138
31, 32	159, 160
69, 70	165, 166

A change to the text or to an illustration is indicated by a vertical line to the left of the change.

Summary of Amendments

Miscellaneous technical and editorial changes have been made throughout the book.

Note: Please file this cover letter at the back of the manual to provide a record of changes.

IBM Corporation, Department 63T
Neighborhood Road, Kingston, New York 12401

Printed in U.S.A.

OS/VS2
HASP II Version 4
System Programmers Guide

GC27-6992-0

READER'S COMMENT FORM

Your views about this publication may help improve its usefulness; this form will be sent to the author's department for appropriate action. Using this form to request system assistance or additional publications will delay response, however. *For more direct handling of such requests, please contact your IBM representative or the IBM Branch Office serving your locality.*

How did you use this publication?

- ☐ As an introduction
- ☐ As a reference manual
- ☐ For another purpose (explain)_____
- ☐ As a text (student)
- ☐ As a text (instructor)

Please comment on the general usefulness of the book; suggest additions, deletions, and clarifications; list specific errors and omissions (give page numbers):

What is your occupation?_____

Number of latest Technical Newsletter (if any) concerning this publication:_____

Please include your name and address in the space below if you wish a reply.

Thank you for your cooperation. No postage stamp necessary if mailed in the U.S.A. (Elsewhere, an IBM office or representative will be happy to forward your comments.)

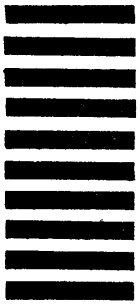
Cut or Fold Along Line

Your comments, please . . .

This manual is part of a library that serves as a reference source for systems analysts, programmers, and operators of IBM systems. Your comments on the other side of this form will be carefully reviewed by the persons responsible for writing and publishing this material. All comments and suggestions become the property of IBM.

Fold Fold

First Class
Permit 40
Armonk
New York



Business Reply Mail
No postage stamp necessary if mailed in the U.S.A.

Postage will be paid by:

International Business Machines Corporation
Department 636
Neighborhood Road
Kingston, New York 12401

Fold Fold

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)

Cut or Fold Along Line