

## **Systems**

## **OS/VS Dynamic Support System**

**VS1 Release 3**  
**VS2 Release 2**

### **Where to Find**

Data Fields 19  
DSS Functions 30  
Commands 56  
Command Summary 89  
User's Guide 101  
Examples Collection 127  
Messages 149

### **Other Information**

Contents 5  
Figures 7  
Index 201

The IBM logo is displayed in a large, bold, outlined font at the bottom center of the page.

### **Second Edition (November, 1973)**

This is a major revision of, and obsoletes, GC28-0640-0 and Technical Newsletter GN28-2563. See the Summary of Amendments following the Contents. Changes or additions to the text and illustrations are indicated by a vertical line to the left of the change.

This edition applies to Release 3 of OS/VS1 and Release 2 of OS/VS2 and to all subsequent releases until otherwise indicated in new editions or Technical Newsletters. Any references to OS/VS2 appear for planning purposes only, until the availability of Release 2 of OS/VS2.

Changes are continually made to the information herein; before using this publication in connection with the operation of IBM systems, consult the latest **IBM System/360 and System/370 Bibliography**, GA22-6822, 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.

A form for readers' comments is provided at the back of this publication. If the form has been removed, comments may be addressed to IBM Corporation, Publications Development, Department D58, Building 706-2, PO Box 390, Poughkeepsie, N.Y. 12602. Comments become the property of IBM.

# Preface

This book is a combined language reference manual and user's guide for the Dynamic Support System (DSS).

DSS is a debugging tool that IBM Program Systems Representatives and user-authorized personnel can use to identify and provide temporary corrections to software errors in the IBM Operating System with Virtual Storage (OS/VS).

This book has six sections and three appendixes:

**Section 1: Introduction** — describes the functions and capabilities of DSS.

**Section 2: Command Language Reference** — describes the commands and operands that make up the DSS command language.

**Section 3: Command Summary** — a collection of tables that summarize the information presented in Section 2.

**Section 4: User's Guide** — tells how to use DSS.

**Section 5: Output Formats** — describes the formatted output (dumps and displays) produced by DSS.

**Section 6: Messages** — describes the messages issued by DSS.

**Appendix A: Restrictions** — lists general restrictions to DSS use.

**Appendix B: Command-Syntax Diagrams** — displays DSS commands in a graphic metalanguage.

**Appendix C: Glossary** — defines special terms used in this publication.

## Prerequisite Knowledge

The reader must be familiar with IBM System/370 operator procedures, and virtual storage concepts as implemented in OS/VS.

Before actually using DSS, the reader must know the internal logic of OS/VS. In addition, the following reference material will be helpful:

**IBM System/370 Principles of Operation, GA22-7000**

**OS/VS1 Debugging Guide, GC24-5093**

**OS/VS2 System Programming Library: Debugging Handbook, GC28-0632**

**OS/VS1 System Data Areas, SY28-0605**

**OS/VS2 Data Areas, SYB8-0606 (microfiche)**

**OS/VS Data Management Macro Instructions, GC26-3793**

**OS/VS1 Supervisor Services and Macro Instructions, GC24-5103**

**OS/VS2 Supervisor Services and Macro Instructions, GC28-0683**

**OS/VS Message Library:**

**VS1 System Codes, GC38-1003**

**VS2 System Codes, GC38-1008**

**VS1 System Messages, GC38-1001**

**VS2 System Messages, GC38-1002**

**Service Aids and OLTEP Messages, GC38-1006**

**Routing and Descriptor Codes, GC38-1004**

**Operator's Library: System/370 Model (your model) Operating Procedures**

**OS/VS DSS Command Language Reference Summary, GX28-0690**

Because the manual is arranged mainly for reference purposes, certain words and phrases appear, of necessity, earlier in the manual than the principal discussions explaining them. The reader who encounters a problem of this sort should refer to the index, which will direct him to the key description.

Information that applies only to OS/VS2 is shaded like this. This information is for planning purposes until DSS is available in OS/VS2 Release 2.



# Contents

<b>Section 1: Introduction</b> . . . . .	11
DSS Command Language . . . . .	11
Commands . . . . .	11
Operands . . . . .	12
Functions . . . . .	12
Expressions . . . . .	12
DSS Monitoring . . . . .	12
DSS I/O . . . . .	13
Display Consoles . . . . .	13
System Requirements . . . . .	13
Performance Considerations . . . . .	14
<b>Section 2: Command Language Reference</b> . . . . .	15
Elements of the Language . . . . .	15
The Character Set . . . . .	15
Using the Character Set . . . . .	16
Identifiers . . . . .	16
Operators . . . . .	16
Other Special Characters . . . . .	17
Contents of a Command . . . . .	18
Data Fields . . . . .	19
Data Field Attributes . . . . .	20
Immediate Attribute Designation . . . . .	21
Immediate Data Fields . . . . .	22
Mapped Data Fields . . . . .	25
Subscripts and Arrays . . . . .	26
Range Specifications . . . . .	28
Syntax Notation . . . . .	29
DSS Functions . . . . .	30
Machine Register Functions . . . . .	30
Program Status Functions . . . . .	31
Machine Status Functions . . . . .	33
Command-Related Functions . . . . .	36
Attribute Functions . . . . .	39
Location Functions . . . . .	40
Map Functions . . . . .	46
Event Functions . . . . .	47
Expressions . . . . .	48
Arithmetic Operators . . . . .	49
Comparison Operators . . . . .	50
Boolean Operators . . . . .	50
Examples of Expressions . . . . .	51
Input/Output . . . . .	51
The SNAP Dump . . . . .	54
The Hardcopy Log . . . . .	55
Commands . . . . .	56
The Comment Command . . . . .	56
ASSIGN . . . . .	57
AT . . . . .	58
COLLECT . . . . .	60
DEFINE . . . . .	63
DISABLE . . . . .	65
DISCONNECT . . . . .	65
DISPLAY . . . . .	66
DIVERT . . . . .	67
DUMP . . . . .	69
ENABLE . . . . .	70
END . . . . .	71
EQUATE . . . . .	71
GO . . . . .	73
GOTO . . . . .	73
IF . . . . .	74
INVOKE . . . . .	75
ON . . . . .	76
PATCH . . . . .	78

PROCEDURE . . . . .	79
RELEASE . . . . .	81
REMOVE . . . . .	82
RETURN . . . . .	83
REVERT . . . . .	84
SET . . . . .	85
<b>Section 3: Command Summary . . . . .</b>	<b>89</b>
<b>Section 4: User's Guide . . . . .</b>	<b>101</b>
How to Gain Control . . . . .	101
When DSS is Dormant . . . . .	101
When DSS is Executing Commands . . . . .	102
When DSS is Monitoring . . . . .	103
Multiprocessing Considerations . . . . .	104
RESTART Summary . . . . .	104
How to Enter Commands . . . . .	105
DSS Command Processing . . . . .	105
How to Correct an Invalid Command . . . . .	107
How to Display Data . . . . .	110
How to Alter Data . . . . .	111
Command Procedures and Command Streams . . . . .	111
How to Do Simple Branching . . . . .	114
How to Return Control to OS/VS . . . . .	115
Control Levels . . . . .	116
System Errors . . . . .	124
How to Obtain a Dump . . . . .	125
Resumption of Processing . . . . .	125
Examples . . . . .	127
<b>Section 5: Output Formats . . . . .</b>	<b>133</b>
DISPLAY Output . . . . .	133
DUMP Output . . . . .	135
SNAP Output . . . . .	136
Type 1 — ATs, ONs, (&B, &I, &RA, &SA) . . . . .	136
Type 2 — ONs (&LD, &UNLD) . . . . .	137
Output Produced for DSS Functions . . . . .	137
&AT . . . . .	138
&ON . . . . .	139
&PATCH . . . . .	139
&PROC . . . . .	140
&SYM . . . . .	141
&PREFIX, &EPSW, &EPSWN, &IPSW, &IPSWN, &PPSW, &PPSWN, &SPSW, &SPSWN, &MPSW, &MPSWN, &RPSW, &RPSWN, &PSW, &CAW, &CID, &CSW, &IOEL, &MC, &PRM, &RANGE, &TEA, &ASID, &ASID(job-name), &CPUID, &AC, &ADDR, &M . . . . .	142
&DOUT, &QT, &SOUT, &TOD . . . . .	142
&HDR . . . . .	142
&ID . . . . .	142
&P, &S, &B, &I, &RA, &SA, &LD, &UNLD . . . . .	143
Map Output Formats . . . . .	143
&O, &L, &SZ, &T . . . . .	146
&G, &C, &F . . . . .	146
&SVM, &JOB, &TCB, &LM, &PRB, &Q . . . . .	147
&RM . . . . .	147
Error Dump . . . . .	147
<b>Section 6: Messages . . . . .</b>	<b>149</b>
Problem Determination (Table I) . . . . .	150
<b>Appendix A: Restrictions . . . . .</b>	<b>191</b>
<b>Appendix B: Command-Syntax Diagrams . . . . .</b>	<b>193</b>
How to Use a Syntax Diagram . . . . .	193
DSS Commands . . . . .	194
<b>Appendix C: Glossary . . . . .</b>	<b>199</b>
<b>Index . . . . .</b>	<b>201</b>

# Figures

Figure 1.	A DSS Session . . . . .	10
Figure 2.	DSS Special Characters . . . . .	16
Figure 3.	Operators (Part 1 of 2) . . . . .	16
Figure 4.	Special Characters That Are Not Operators . . . . .	17
Figure 5.	A Sample Array . . . . .	27
Figure 6.	Use of &Q, &QT, and &LM to Specify Symbolic Addresses . . . . .	45
Figure 7.	Operator Priorities . . . . .	49
Figure 8.	I/O Names and Related Commands . . . . .	53
Figure 9.	Magnetic Tape Data Attributes . . . . .	53
Figure 10.	SNAP Output Information . . . . .	54
Figure 11.	Collection Area . . . . .	62
Figure 12.	Truncation and Padding by SET and DEFINE . . . . .	86
Figure 13.	DSS Commands (Part 1 of 3) . . . . .	90
Figure 14.	Command Operands (Part 1 of 3) . . . . .	93
Figure 15.	DSS Functions (Part 1 of 2) . . . . .	96
Figure 16.	Using DSS Functions (Part 1 of 2) . . . . .	98
Figure 17.	I/O Names and Their Uses . . . . .	100
Figure 18.	The RESTART Key . . . . .	104
Figure 19.	Command Processing . . . . .	106
Figure 20.	DSS Response to an Invalid Command . . . . .	108
Figure 21.	How to Respond to the Canceling of a Command or Command Operand . . . . .	109
Figure 22.	The Definition and Invocation of a Command Procedure . . . . .	112
Figure 23.	The Definition and Starting of a Command Stream . . . . .	113
Figure 24.	GOTO (Within a Command Procedure) . . . . .	114
Figure 25.	DIVERT . . . . .	114
Figure 26.	REVERT . . . . .	115
Figure 27.	GOTO (Command Stream to Command Procedure) . . . . .	115
Figure 28.	After DSS Initiation . . . . .	116
Figure 29.	A Succession of Control Levels . . . . .	117
Figure 30.	After a Monitoring Interrupt . . . . .	117
Figure 31.	Skipping Control Levels on Return . . . . .	118
Figure 32.	Command Inputs Within a DSS Control Level . . . . .	119
Figure 33.	Command Inputs Within the Primary Control Level . . . . .	120
Figure 34.	DSS Control Levels . . . . .	121
Figure 35.	Return of Control to OS/VS . . . . .	122
Figure 36.	Example of a DSS Session . . . . .	123
Figure 37.	DISPLAY Line Prefixes . . . . .	133
Figure 38.	Problem Determination (Table I) . . . . .	150
Figure 39.	Syntax Diagram . . . . .	193



## Summary of Amendments for GC28-0640-1 VS1 Release 3 and VS2 Release 2

Information in this manual applies to both OS/VS1 Release 3 and OS/VS2 Release 2. With this edition, most amendments to the publication are in support of OS/VS2 Release 2.

### New DSS Functions

- &PRDMP[RM], a new operand in the DUMP command, allows dumps to be saved on magnetic tape for formatting and printing by the AMDPRDMP service aid program.
- &ASID, a new DSS function, allows reference to specific address spaces in OS/VS2 storage.
- &CPUID, a new DSS function, allows reference to specific CPUs in a multiprocessing system.
- Other new DSS functions, &CID, &IOEL, &PREFIX, and &TEA, represent additional information for OS/VS2 debugging.

### DSS Shares the RESTART Resource with RTM

- The PSADSSGO byte at OS/VS2 location 279 (hexadecimal) must be set to a nonzero value before the RESTART key can activate DSS.
- RECOVER, a new operand of the GO command, gives control to recovery termination management (RTM) as if the RESTART key was pressed when DSS did not own the RESTART resource.

## Summary of Amendments for GC28-0640-0 as Updated by GN28-2563

Information in this manual applies to both OS/VS2 Release 1.6 and the component release of DSS for OS/VS1 Release 2. To support both of those releases, page 51 was changed to show an improvement in the COLLECT command's maintenance of a collection area's offset. Several other pages received editorial changes and technical corrections.

### The DSS Error Dump

- DSS-error dumps are written directly to a printer, not to a direct access device. Therefore, there is no DSS error print utility in OS/VS2 Release 2.

### Dependence on the NUCMAP System Parameter

- The validity of the DSS maps of the OS/VS2 nucleus depends on the inclusion of the NUCMAP system parameter in system initialization whenever the nucleus is changed without changing its name.

### New Messages

- Many new messages have been added to "Section 6: Messages."

### Rewrite of Section 4

Section 4 has been rewritten to give a more step-by-step description of DSS use.

Information that applies to DSS only in OS/VS2 Release 2 is shaded as above. This information is for planning purposes until DSS is available in OS/VS2 Release 2.

Along with the incorporation of descriptions of new facilities and their impact, the entire manual has been reviewed and, where necessary, changed to correct technical and stylistic discrepancies and to improve the presentation.

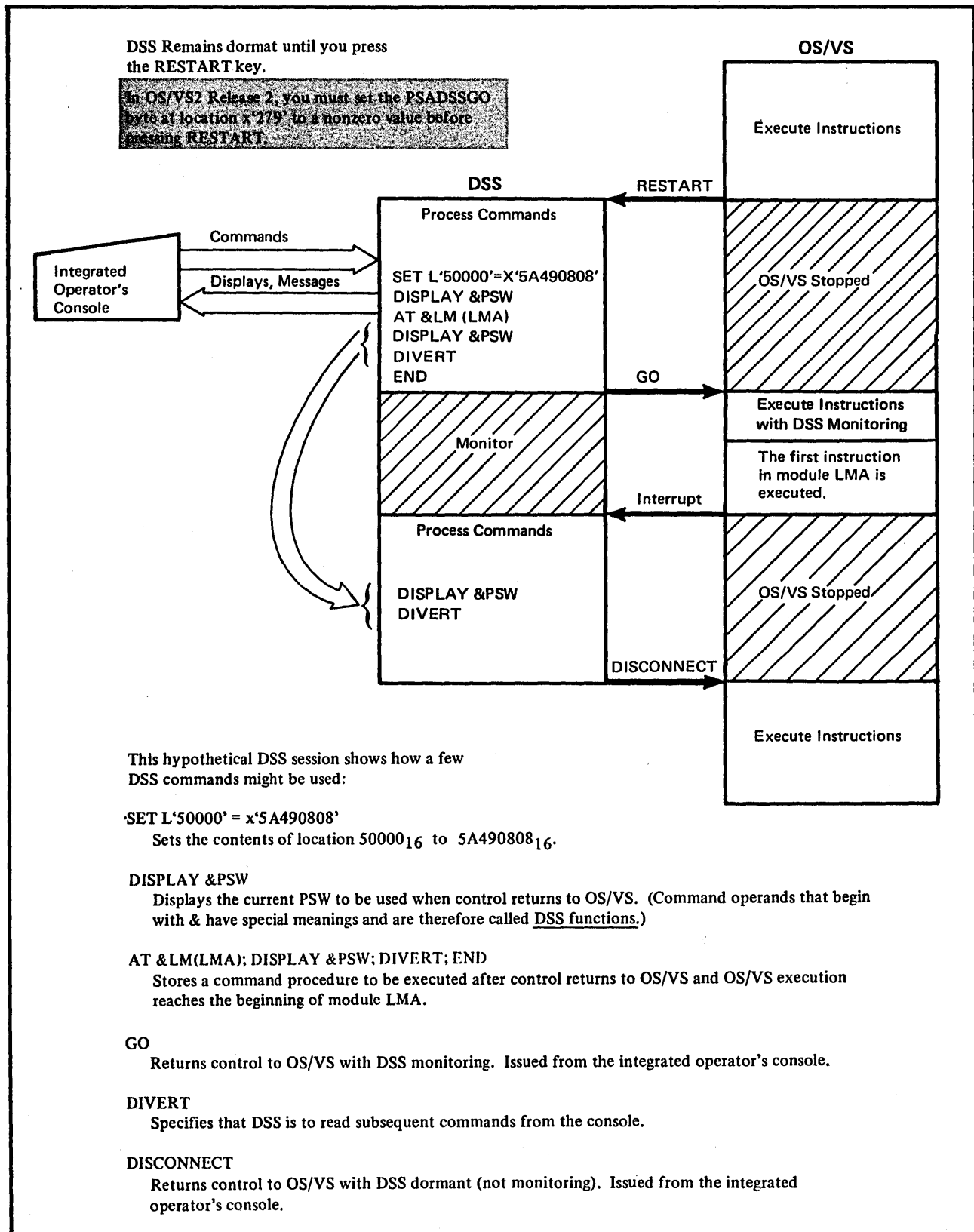


Figure 1. A DSS Session

## Section 1: Introduction

The Dynamic Support System (DSS) is a debugging tool that helps authorized personnel, such as IBM Program Systems Representatives, to identify and provide temporary corrections to software errors in OS/VS.

DSS allows you to:

- Display and alter real storage, virtual storage, and registers.
- Stop the operation of OS/VS at any instruction, or when a selected event occurs, and perform maintenance functions automatically or with your personal intervention.
- Store debugging information for later use, or write the information on tape or on a printer.
- Resume OS/VS operations at the point of interruption or at almost any point of your choice.
- Use tape or card readers for secondary command input devices.
- Write command procedures to be used as reiterative command sequences.

You begin a DSS session by pressing the RESTART key. After DSS initialization is complete, DSS responds by typing a message to inform you that it is ready; you can then enter any DSS commands.

During the session, DSS operates independently of OS/VS by performing its own I/O, storage management, and interrupt handling.

Figure 1 shows an example of a DSS session. Figure 1 shows just some of the things that DSS might do; there are many more commands available to you.

### The DSS Command Language

The DSS command language is a versatile, high-level programming language.

#### Commands

DSS commands can address real or virtual storage with either numeric or symbolic addresses. There are system-defined symbols that refer to those OS/VS modules that are associated with jobs or subtasks or other mapped portions of OS/VS such as the **link pack area** or nucleus. In addition, you can define your own symbols to represent any areas of storage.

DSS commands can be stored as reiterative command procedures to be executed later. Commands in these procedures can be prefixed by labels that allow branching. The IF command can be used to perform comparisons in a command procedure and make decisions based on those comparisons.

DSS commands can be used to identify program errors in the parts of DSS that are mapped into OS/VS virtual storage. However, normal program error identification and repair methods must be used for the majority of errors in DSS itself.

## Operands

Operands, which most commands must include, indicate what a command is to work on or with. In the command `DISPLAY AREA`, the `AREA` operand is the name of the data field to be displayed. In `DIVERT CARD`, `CARD` indicates the card reader as the device from which subsequent commands are to be read.

## Functions

Operands that begin with `&` are called DSS functions; they perform specialized functions such as referring to a data field that could not be referred to in any other way.

### Examples:

```
REMOVE &PATCH(1)
```

removes the first patch that you put in OS/VS during the current DSS session.

```
DISPLAY &G(3)
```

displays the fourth general register as it was when OS/VS was interrupted.

## Expressions

When a mathematical operation (arithmetic, comparison, or boolean) is included in a DSS command, it is called an expression. An expression consists of operands that are related by operators, such as `A+B>C|D`, which means "A plus B is greater than C OR D."

## DSS Monitoring

DSS can gain control when OS/VS:

- Executes a branch instruction.
- Loads into or alters the contents of a designated general register.
- Fetches an instruction from a designated storage location.
- Stores into a designated storage location.
- Loads or unloads a module.

Whenever you specify an event to be monitored (by means of an `AT` or `ON` command followed by a `GO` command), subsequent occurrence of the program event will result in a program interrupt that passes control to DSS and records the cause of the interrupt. You can monitor all of OS/VS except for:

- DSS.
- NIP.
- Prefixed storage areas (PSAs), except that PSAs can be monitored by the `ON` command.
- Part of the restart interrupt handler.
- The part of the machine check handler that owns the new PSWs.
- Part of the stop/restart subroutine.
- Wake-up code in vary-online processing.
- The first instruction of each OS/VS1 first-level interrupt handler.



## DSS I/O

DSS accepts command input from the integrated operator's console, card reader, or a tape drive; the DUMP command enables you to save commands on cards or magnetic tape for later input.

DSS messages are printed at the integrated operator's console.

The DISPLAY and DUMP commands enable you to display data at the integrated operator's console, at a high-speed printer, or on magnetic tape.

I/O devices are specified by DSS-defined symbols called I/O names; in DIVERT CARD, CARD is the I/O name for the card reader.

## Display Consoles

DSS can be used if the integrated operator's console is a display console. Typing conventions are slightly different from the typing conventions described in this manual: Instead of an END key, there is an ENTER key; instead of the backspace equivalent (?) you can use the cursor to make corrections.

On some models, RESTART is an operator function instead of a key. Except for the RESTART operator function, DSS does not support the light pen. Console programming support that allows message deletion, message numbering, and system status displays is not active when DSS has control. Messages and commands are displayed beginning at the fifth line from the bottom and bumped up and eventually rolled off the top of the screen as the screen is filled.

When control is returned to OS/VS, the screen is restored as it was when OS/VS was interrupted.

With display consoles, you should keep a hardcopy log; see "The Hardcopy Log," under "Input/Output," in Section 2.

## System Requirements

**CPU:** DSS runs on the same CPUs as OS/VS. All CPUs must have relocation hardware, program event recording (PER), and the RESTART function. In OS/VS1, the RESTART function must be dedicated to DSS use.

**System Generation:** System generation as described in *OS/VS1 System Generation Reference, GC26-3791* or *OS/VS2 System Generation Reference, GC26-3792*, together with the startup procedures performed by the operators, prepares OS/VS for DSS.

**System Initialization:** The programmers who perform system initialization must include the NUCMAP system parameter in system initialization whenever the OS/VS2 nucleus is changed without changing its name. See *OS/VS2 System Programming Library: Initialization and Tuning, GC28-0681*.

**Paging Devices:** DSS uses the same paging devices as OS/VS.

**I/O Devices:** The integrated operator's console and the standard card readers, printers, tape drives, and card punches supported by OS/VS are supported by DSS.

**Real-Time Conflicts:** DSS cannot be used concurrently with a teleprocessing, time sharing, or other real-time task. It may be used if the real-time task is quiesced or if processing is expected to be terminated by the problem being debugged. When DSS is active, RES (Remote Entry Services) cannot be used because of teleprocessing timing considerations. That is, within one IPL, either DSS or RES can be used, but not both.

**Main Storage Estimates:** DSS requires approximately 1K of permanently resident OS/VS storage. DSS dynamically acquires approximately 130K or 34K of fixed storage (paged in from the primary paging device—the LPA or the RRMA — and freed by DISCONNECT) and 600K or 384K on a permanent DSS paging data set.

## Performance Considerations

When DSS has not been initiated, there should be less than 1% decrease of OS/VS performance due to code added to system modules and permanent DSS storage requirements. When DSS is initiated, DSS requires 130K or 34K of page-fixed storage; this reduces the amount of storage available for other use and therefore decreases system throughput. Further performance degradation depends on the use made of DSS and may range from minor to severe.

When OS/VS runs with DSS monitoring, normal multiprogramming occurs, with performance decreased by the monitoring function of DSS. Each use of a monitoring function decreases OS/VS performance. For example, an ON command slows down OS/VS within the range of monitored instructions.

When DSS is fully activated, no other OS/VS processing takes place. Full activation is required to initiate DSS, process commands, write a SNAP buffer, or delete the DSS control blocks associated with a load module that is being unloaded from the system.

The multiprocessing configuration that gives the best DSS performance is where all devices that DSS is to use (paging device, integrated operator's console, printers, tapes, card reader, and card punch) are addressable from the same CPU.

## Section 2: Command Language Reference

This section describes the DSS command language. It contains two topics:

### Elements Of the Language

A general description of the elements of the DSS command language. This includes:

- The Character Set
- Using the Character Set
- Contents Of A Command—label, command verb, operands, and end-of-record
- Comments—the inclusion of comments with stored commands
- Data Fields—the use of operands to represent data
- Syntax Notation—used throughout this publication to describe syntax
- DSS Functions—command operands with specialized functions
- Expressions—arithmetic, comparison, and boolean operations
- I/O—operands and suboperands that control I/O

### Commands

A detailed description of the format and function of each DSS command, in alphabetical order.

For a summary of the DSS command language, refer to “Section 3: Command Summary.”

Examples of how to use DSS are at the end of “Section 4: User’s Guide.”

**Note:** Shaded information applies only to OS/VS2.

## Elements Of the Language

### The Character Set

The character set used to construct DSS commands consists of:

- The 26 letters: A through Z (Although you can enter letters in lowercase, DSS translates them to uppercase.)
- The six alphabet-extender characters  $\epsilon$ , !, \$, #, @, "
- The digits 0 through 9
- The 21 special characters; see Figure 2

Name	Character
Blank	(A blank, or space, represented in this publication by "b")
Period	.
"Less than" symbol	<
Left parenthesis	(
Plus sign	+
"Or" symbol	
Ampersand	&
Asterisk	*
Right parenthesis	)
Semicolon	;
"Not" symbol	~
Minus sign	-
Slash	/
Comma	,
Percent sign	%
Underscore	_
"Greater than" symbol	>
Backspace equivalent	?
Colon	:
Apostrophe	'
Equal sign	=

Figure 2. DSS Special Characters

## Using the Character Set

### Identifiers

An identifier is used in a command as a name of a data field, a name of a command procedure, a name of records stored by a command, or a label for a command. An identifier must begin with one of the 32 alphabetic or alphabet-extender characters (A-Z, *ç*, !, \$, #, @, " ), must contain only alphameric characters, and must not contain blanks. No identifier can exceed eight characters in length. These are identifiers:

```
A
DIVERT
WORKAREA
LOOP3
$495
```

### Operators

Some special characters function as operators. See Figure 3.

Type of Operation	Operator	Operation
ARITHMETIC	+ - * / //	addition or prefix plus subtraction or prefix minus multiplication division (quotient) division (remainder)

Figure 3. Operators (Part 1 of 2)

Type of Operation	Operator	Operation
COMPARISON	>	greater than
	¬>	not greater than
	>=	greater than or equal to
	=>	equal to or greater than
	=	equal to
	¬=	not equal to
	=<	equal to or less than
	<=	less than or equal to
	<	less than
	¬<	not less than
BOOLEAN	¬	not
		or
	&&	and

Figure 3. Operators (Part 2 of 2)

### Other Special Characters

Figure 4 shows the functions of other special characters and special-character combinations.

Name	Character	Use
blank	␣	separate command elements (see "Blanks," later in this section)
Period	.	connect elements of a qualified name
parentheses	()	delimit immediate attribute designation
ampersand	&	enclose lists; enclose subscripts; enclose parts of expressions
		if typed by you, indicates a DSS function (see "DSS Functions," later in this section)
		if typed by DSS, prompts you to enter a command
semicolon	;	separate commands in a line that contains two or more commands
equal sign	=	denotes a correspondence pair, i.e., A = B means that B corresponds to A
		also used as an operator
comma	,	separate operands or elements of a list
percent sign	%	indirect addressing (see "Indirect Address," later in this section)
underscore	—	if typed at beginning of line, indicates escape from a definitional command, i.e., the following DIVERT or REVERT command is to be executed immediately, rather than stored as part of the command procedure.
		if typed at end of line, indicates that command is continued on next line
backspace equivalent or question mark	?	if typed by you, indicates backspace
		if typed by DSS, indicates that you can type an error-correcting command, if necessary
colon	:	terminate command label (see "Contents of A Command," later in this section)
		indicate a range (see "Range Specifications," later in this section)
number sign	#	typed by DSS to prompt during command-procedure entry or after a continuation underscore
		also an alphabet-extender character
apostrophe	'	enclose literals (see "Immediate Data Field," later in this section)

Figure 4. Special Characters That Are Not Operators

## Contents of a Command

A command has:

- An optional label (specified as an identifier followed by a colon)
- A command verb, which indicates the function to be performed
- Operands, if required, which indicate what is to be worked on or with
- A semicolon, or a press of the END key with no continuation character preceding, to terminate the command

Example of a command:

```

LBL:      DISPLAY      AREA1,AREA2,AREA3
  label    command-    operands
           verb
           semicolon or
           end of record
  
```

**Note 1:** A null command consists of only a semicolon and indicates no operation.

**Note 2:** Any command verb that satisfies the rules for identifiers can also be used as a command label, data-field name, or command-procedure name. Thus,

```
INVOKE:  INVOKE INVOKE, INVOKE
```

is a syntactically valid command. Such practices, however, are not recommended.

**Note 3:** A carrier return is ignored.

**Note 4:** A press of the END key indicates the end of an input line if there is a preceding command; it is ignored if there is no preceding command.

A command can be either **simple** or **definitional**. A simple command specifies actions to be performed unconditionally. A definitional command specifies actions to be performed if a certain condition or event occurs; those actions may be defined by commands that accompany the definitional command. There are four definitional commands: AT, ON, IF, and PROCEDURE.

The commands with a definitional command are **text commands**. The last text command with a definitional command must be END.

Example of definitional command:

```

LBL:      AT ENTRY1,&P(SNAP); DISPLAY AREA1,AREA2;LBL1:INVOKE P;END
  label    command-verb  text      semicolon
           and operands  commands  or end of
           record
  
```

**Note 5:** Commands need not be separated by semicolons if they are entered on separate lines. Example:

```

LBL:      AT ENTRY1, &P(,SNAP)
          DISPLAY AREA1,AREA2
          LBL1: INVOKE P
          END
  
```

**Note 6:** When you type commands at the integrated operator's console, DSS displays prompting characters between the input lines. For example, if the above commands were typed at the console, they would look like this:

```
&
LBL:   AT ENTRY1, &P( ,SNAP )
#
      DISPLAY AREA1,AREA2
#
      LBL1: INVOKE P
#
      END
```

Figure 4, above, explains the meanings of the three prompting characters (&, #, and ?) that can be typed by DSS.

**Note 7:** A text command can be either definitional or nondefinitional. The maximum depth of definitional-command nesting is 256.

**Note 8:** In a definitional command, the END command is required even if no other text commands are specified.

**Note 9:** The maximum size of a simple command or text command, excluding continuation underscores, is 256 characters.

### Blanks

Blanks may be used before a command, within a command, and after a command. Within a command, blanks may be placed only:

- Before and after special characters; exceptions to this rule:
  1. A blank must not precede a continuation underscore, unless the line ends at a point where a blank is permitted.
  2. Composite symbols (for example, >=, &&) must not contain blanks.
- Before and after the command verb.
- Within a character literal.

Any number of blanks may appear wherever one blank is allowed.

The only place one or more blanks are **required** is between the command verb and the first operand.

### Comments

To include comments in text commands or a command stream, use the comment command (\*). The comment command is the first command described in this section, under "Commands."

## Data Fields

A data field is any storage area that is represented by a DSS command operand. A data field can be in either DSS storage or OS/VS storage, and it can be either an **immediate** or a **stored** data field.

An immediate data field is defined by a literal, by indirect addressing, or by an expression. It exists only during the execution of the command that defines it.

A stored data field is either predefined by DSS or defined by you. The data fields that are predefined by DSS are represented by some of the DSS functions; these data fields are described later in this section, under "DSS Functions." The data fields defined by you are represented by identifiers that you assign in the DEFINE command, the EQUATE command, or the parameter operand of the PROCEDURE command. (See the descriptions of these commands under "Commands," later in this section.)

### Data Field Attributes

The following attributes are associated with every data field:

- Base
- Offset (o)
- Length (l)
- Type (t)
- Size (sz)

**Base:** The address of the leftmost byte of the data field. If the data field is in DSS storage, the base is represented by the data field's identifier and cannot be represented numerically. If the data field is in OS/VS storage, the base can be represented by use of the &ADDR DSS function.

**Offset:** The location, as a displacement from the base, of the first byte that participates in any operation involving the data field. COLLECT and SET are the only commands that can change this attribute.

**Length:** The number of bytes that participate in any operation involving the data field.

**Type:** How the contents of the data field are to be interpreted. There are three codes that you can use to specify type attributes:

- I a signed, binary, two's-complement integer in the range  $-2^{8n-1}$  to  $+2^{8n-1}-1$ , where  $n$  is the length of the data field in bytes.
- X a hexadecimal byte string in the range 0 to  $2^{8n}-1$ , where  $n$  is the length of the data field.
- C EBCDIC characters.

**Size:** The total number of bytes in the data field.

**Summary:** When you refer to a data field, the base and size attributes define the data field, the type attribute specifies how the data field is to be interpreted, and the length and offset attributes specify what part of the data field you are referring to.

Base points to the first byte of the data field.

Base + offset points to the first byte of the referenced string.

Base + offset + length - 1 points to the last byte of the referenced string.

Base + size - 1 points to the last byte of the data field.

**Note:** Offset plus length must not exceed size.



Example: The command DEFINE ITEM would create a stored data field named ITEM.

```
ITEM
┌───────────┐
|←4 bytes→|
```

The attributes of ITEM have not been initialized by the DEFINE command and have not been changed by a SET command; therefore, they default to:

```
base = an address represented by the symbol ITEM
offset = 0
length = 4 bytes
type = X (hexadecimal)
size = 4 bytes
```

ITEM will contain zeros until it is filled in by a SET command.

### Immediate Attribute Designation

Immediate attribute designation allows you to override any of a data field's attributes except base. The data field is redefined, only for the operation in which the immediate attribute designation is used, by the immediate attributes specified. In addition, when you define a data field for the first time, using a DEFINE or EQUATE command, immediate attribute designation can be used to establish the data field's attributes.

The format of the immediate attribute designation is:

```
data-field.(o,l,t,sz)
```

**data-field**

The data field to which the immediate attributes are to be assigned.

**Specified as:** Any valid representation of a data field (its name, its address literal, or its indirect address).

**o,l,t,sz**

The offset, length, type, and size attributes. You can omit any of these attributes and accept original values, but you must retain all commas to the left of the last attribute specified.

**Specified as:** Offset, length, and size are normally specified as decimal literals or hexadecimal literals (descriptions of these literals follow), although any symbol or expression that returns an appropriate value can be used. Type is code I, X, or C. The size attribute is valid only when you define a DSS-storage data field with the DEFINE command. If size is specified with any other immediate attribute designation, it is ignored or responded to by an error message.

### Examples:

```
DISPLAY DATA.(,8,X)
```

displays the first eight bytes of DATA in hexadecimal format.

```
DISPLAY DATA.(8,8,I)
```

displays the second eight bytes of DATA in decimal format.

### Immediate Data Fields

An immediate data field is defined by a literal, by indirect addressing, or by an expression. It remains defined only during the execution of a single command.

**Decimal Literal:** Written as a string of decimal digits. Example: 12980. The range of maximum values is  $+2^{31}-1$  to  $-2^{31}$ . During execution, the decimal literal produces an immediate data field in DSS storage with the attributes:

offset = 0  
length = 4  
type = I  
size = 4

**Hexadecimal Literal:** Written as a hexadecimal integer enclosed in apostrophes and preceded by X. Example: X'C4E2E2'. There is no maximum value. If an odd number of digits is specified, the literal is padded on the left with zero. During execution, the hexadecimal literal produces an immediate data field in DSS storage with the attributes:

offset = 0  
length = one-half the number of hexadecimal digits  
type = X  
size = length

**Character Literal:** Written as a string of characters enclosed in apostrophes. If you want an apostrophe in the character string, include two apostrophes; one will be edited out. Example: '2" S COMPLEMENT'. The content of the data field is: 2" S COMPLEMENT.

The &S DSS function and the backspace equivalent ('?') control input editing even if they are in a literal. (Example: 'COMPLEN?MENT'. The content of the data field is: COMPLEMENT). All input editing functions are performed before the character literal is resolved.

During execution, the character literal produces an immediate data field in DSS storage with the attributes:

offset = 0  
length = the number of characters in the string (after input editing)  
type = C  
size = length

**Address Literal:** There are two forms of address literal: simple and range. The simple address literal is written as one to eight hexadecimal digits enclosed in apostrophes and preceded by L. Example: L'C40000'. If less than eight digits are specified, the address is padded on the left with zeros to produce eight digits.

A simple address literal specifies an immediate data field in OS/VS storage with the attributes:

base = the address specified in the literal  
offset = 0  
length = the smaller of size or 4  
type = X  
size = in OS/VS1, if the data field is in an area mapped by DSS, the size of the CSECT or load module. Otherwise in OS/VS1, or in OS/VS2, if the literal is not qualified by &RM, the highest OS/VS virtual address, minus base plus one; if the literal is qualified by &RM, the highest real address, minus base, plus one

The address range literal is written as an address of one to eight hexadecimal digits, a colon (:), and another address of one to eight hexadecimal digits, all enclosed in apostrophes and preceded by L. Example: L'2000:3000'. If less than eight digits are specified in either address, the address is padded on the left with zeros to produce eight digits. The second address must be greater than or equal to the first address.

The address range literal specifies an immediate data field in OS/VS storage with the attributes:

base = the first address  
offset = 0  
length = the second address, minus base, plus one  
type = X  
size = in OS/VS1, if the data field is in an area mapped by DSS, the size of the CSECT or load module. Otherwise in OS/VS1, or in OS/VS2, if the literal is not qualified by &RM, the highest OS/VS virtual address, minus base plus one; if the literal is qualified by &RM, the highest real address, minus base, plus one

For either the simple address literal or the address range literal, the immediate attribute designation may be used to modify the offset, length, and type attributes. The size attribute is ignored if specified.

**Note 1:** The use of an address literal to access a program-status indicator (PSW, interrupt code, etc.) or a machine-status indicator (CSW, CAW, etc.) will not indicate OS/VS status, since these areas are also used by DSS. To refer to OS/VS status, use the appropriate DSS function.

**Note 2:** Address literals normally refer to virtual-storage locations. You can refer to a real-storage location by typing &RM.address-literal.

In multiprocessing, for PSA (prefixed storage area) addresses, specify which PSA by typing &CPUID (number).address-literal or omit '&CPUID (number).' and accept the default to the CPU in which the DSS interrupt occurred.

**Note 3:** By typing &ASID(number).address-literal, you can specify which address space the address literal applies to. Otherwise, DSS assumes that the address is in the currently active OS/VS2 address space.

#### Examples:

L'13579B' specifies an immediate data field in OS/VS storage with the attributes:

base = X'13579B'  
offset = 0  
length = 4  
type = X  
size = the highest OS/VS virtual address, minus X'13579B', plus one

L'8FA4'.(16,8,C) specifies an immediate data field in OS/VS storage with the attributes:

base = X'8FA4'  
offset = 16  
length = 8  
type = C  
size = the highest OS/VS virtual address, minus X'8FA4', plus one

**Note:** The actual data referred to is the eight bytes that are sixteen bytes past X'8FA4'

L'3FC:40F' specifies an immediate data field in OS/VS storage with the attributes:

base = X'3FC'

offset = 0

length = 20

type = X

size = the highest OS/VS virtual address, minus X'3FC', plus one

L'123A2:123B5'(.,,C) specifies an immediate data field in OS/VS storage with the attributes:

base = X'123A2'

offset = 0

length = 20

type = C

size = the highest OS/VS virtual address, minus X'123A2', plus one

**Indirect Address:** Indirect addressing is used to address a data field in OS/VS storage by employing the contents of another data field as a pointer. The format is: data-field%, where "data-field" is the name, the address literal, or the indirect address (see the note preceding the examples of indirect addresses) of the pointer.

The addressed data field has the attributes:

base = the location pointed to

offset = 0

length = the smaller of: size or 4

type = X

size = in OS/VS1, the size of the CSECT or load module

in OS/VS2, the highest OS/VS2 virtual address, minus base, plus one

Immediate attribute designation can modify the offset, length, and type attributes. The size is ignored if specified.

The pointer is assumed to have a type attribute of x. In addition, DSS forces the pointer's length attribute to be four: In OS/VS1, DSS moves the pointer to a four-byte area and zeros the high-order byte.

In OS/VS2, if the pointer's length attribute is:

- less than four, DSS pads with zeros on the left;
- four, DSS uses the pointer directly;
- more than four, DSS truncates on the right.

If the value of the four-byte pointer points above the highest OS/VS2 virtual address, DSS ignores just enough of the lefthand bits to bring it within the virtual address range.

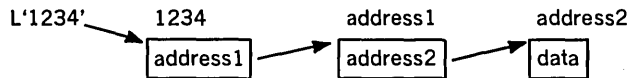
**Note:** Since indirect addressing results in a valid representation of a data field, you can specify multiple levels of indirect addressing. The second example below shows two levels of indirect addressing.

**Examples:**

ABLE%. (8,4,I) points to a data field in OS/VS storage with the attributes:

- base = the location pointed to by the contents of ABLE
- offset = 8
- length = 4
- type = I
- size = the highest OS/VS virtual address, minus base, plus one

L'1234'%% has this effect:



The result is the definition of an immediate data field in OS/VS storage with the attributes:

- base = address2
- offset = 0
- length = 4
- type = X
- size = the highest OS/VS virtual address, minus base, plus one

**Expression:** Any expression results in the definition of an immediate data field; this data field can reside in DSS storage or in OS/VS storage. The operations that can be performed in an expression, and the resulting data fields, are described later in this section, under "Expressions."

**Mapped Data Fields**

DSS maintains maps of programs loaded into OS/VS storage. You can refer to these mapped data fields by including a location function as a DSS command operand. The general form of the operand is:

```
[&ASID(number).][map-id][.load-module-name][. { CSECT-name }
{ entry-point-name } ]
```

For details on &ASID, the address-space identifier, refer to "Location Functions," under "DSS Functions."

The map-id, which must always be qualified by &ASID and/or load-module-name, specifies one of four types of external-symbol maps:

- &SVM — supervisor virtual storage
- &JOB — job step task
- &TCB — task
- &PRB — program request block

**Example:**

```
DISPLAY &SVM.IEANUC01.IEAQFX00.(X'1D0',2)
```

Here, &SVM is the map-id.

For a description of each type of external-symbol map, refer to "Location Functions," under "DSS Functions."

(To display or dump the maps themselves, use the map functions: &SVM MAP, &JOB MAP, and &TCB MAP. See "Map Functions," under "DSS Functions.")

In each external-symbol map are names of up to three different forms of data fields:

- Load modules
- The CSECTS within a load module
- The entry points within a CSECT, or if the CSECTS are not mapped by DSS, the entry points within a load module.

The attributes of a load module are:

base = the address of the beginning of the load module  
offset = 0  
length = size  
type = X  
size = the size of the load module

The attributes of a CSECT are:

base = the address of the beginning of the CSECT  
offset = 0  
length = size  
type = X  
size = the size of the CSECT

The attributes of an entry point are:

base = the address of the entry point  
offset = 0  
length = size  
type = X  
size = the size of the CSECT from the entry point to the end of the CSECT; or, if DSS does not have a map of the CSECTS in the module, the size of the load module from the entry point to the end of the load module

**Note 1:** For a load module, CSECT, or entry point, you can override any of the above attributes except base by designating immediate attributes. If you designate offset but not length, the length is size minus offset.

**Note 2:** To ensure that DSS will always give an accurate map of the OS/VS2 nucleus, the programmers who perform system initialization must include the NUCMAP system parameter in system initialization whenever the nucleus data set is changed without changing its name. See the *OS/VS2 System Initialization and Tuning Guide*, GC28-0681. A programming error in this area could not be detected by the system, and would not be apparent until you felt the consequences of &SVM.IEANUCxx or &SVMMAP (NUC) malfunction. However, you could still refer to OS/VS2 locations via address literals and indirect addressing.

### Subscripts and Arrays

A subscript is used to refer to an element in an array. There are two formats for subscripting:

data-field(n), or data-field(n:m).

data-field

the array.

**Specified as:** any valid representation of an array (its name, address literal, or indirect address).

**n**  
a index value for locating an element of an array.  
Specified as: a decimal literal, a hexadecimal literal, or an expression.

**n and m**  
a pair of index values that refer to a range of elements in the array.  
Specified as: Decimal literals, hexadecimal literals, or expressions; n and m can have different type attributes, but n must be smaller than m.

To define an array, issue a DEFINE command that sets the size attribute to a multiple of the length attribute. That is, DEFINE array-name.(o,l,t,sz), where sz is a multiple of l. (See "Immediate Attribute Designation," earlier in this section.) Size divided by length is the number of array elements; size is the size of the array; length is the length of an array element. DSS locates an array element by:

$$\text{base} + \text{offset} + (\text{subscript} * \text{length})$$

When offset is zero, data-field(0) is the first element in the array. If offset is greater than zero, however, some negative subscripts may be specified, as long as they are within the bounds of the array.

Example: DEFINE XYZ.(4,X,8\*4) defines an array, named XYZ, as shown in Figure 5.

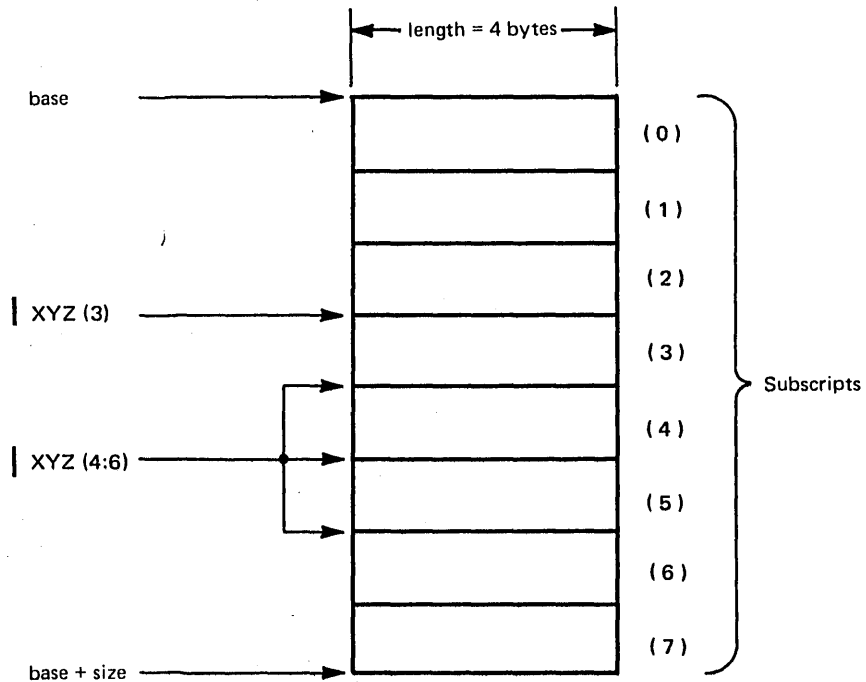


Figure 5. A Sample Array

## Range Specifications

The format for range specifications is indicator1:indicator2. Indicator 1 specifies the beginning of the range; indicator 2 specifies the end of the range.

There are three types of range specification:

1. A range of indexes. This is used in subscript notation and in several of the DSS functions. Examples:

```
ARRAY(0:63)
&G(3:6)
```

Indicators 1 and 2 must be a pair of index values. Refer to the subject "Subscripts and Arrays," immediately preceding this subject.

2. An address range literal. Example:

```
L'30000:3FFFF'
```

Indicators 1 and 2 must be a pair of hexadecimal addresses. See "Address Literal," earlier in this section.

3. A range of locations bounded by two separately-defined data fields. Indicators 1 and 2 must be either data fields in OS/VS storage — example: L'3000':L'3FFF' — or areas within one data field in DSS storage — example: A(1):A(5). (A data field in DSS storage is created by the DEFINE command or by the "parameter" operand of the PROCEDURE command.) DSS resolves the range specification by creating an immediate data field with the attributes identified by 3 in the following list; 1 and 2 identify attributes of indicators 1 and 2:

```
base3 = base1 + offset1
offset3 = 0
length3 = base2 + offset2 + length2 - base3
size3 = (the highest OS/VS virtual address) - base3 + 1
base2 + offset2 + length2 must be greater than base1 + offset1.
```

Examples of ranges bounded by data fields:

```
L'3000':L'FFFF'
&G(5)%: &G(6)%      (See "Machine Register Functions,"
                    later in this section.)
USERSYM1(1):USERSYM1(7)
USERSYM2.(8,4):USERSYM2.(16,4)
&JOB(JOBA).MODA.CSECTA: &Q.CSECTB
L'885'%:L'889'%
USERSYM3%:USERSYM3%+2048
L'4096': &G(4)%      (See "Machine Register Functions,"
                    later in this section.)
```

**Note:** A range bounded by address literals is not exactly the same as an address range literal. Compare these examples:



L'0:L'1F' represents a data field with the attributes

base = 0  
offset = 0  
length = 35 (that is, 1F + 0 + 4 - 0)  
type = X  
size = the highest OS/VS virtual address plus one

L'0:1F' represents a data field with the attributes

base = 0  
offset = 0  
length = 32 (that is, 1F - 0 + 1)  
See the formula for the length of an address range literal, given under  
"Address Literal," earlier in this section.  
type = X  
size = the highest OS/VS virtual address plus one

## Syntax Notation

The following paragraphs describe the conventions that are used in representations of the syntax of DSS functions and commands.

Within each syntax illustration, any term in capital letters must be typed exactly as shown, except that you can also use lowercase letters. Terms shown in lowercase letters are terms for which you must substitute your own values.

If a syntax illustration shows a term in parentheses, you must enclose the term in parentheses if you specify it; conversely, empty parentheses are not allowed.

DSS commands have no keyword operands of the type used in OS/VS JCL and operator commands (that is, KEYWORD=parameter). DSS operands are all positional, in the sense that they must be specified in the order shown in the format representations.

These notational symbols are used in format representations; they are not punctuation marks and must not be typed in actual commands:

- Square brackets ([ ]) enclose terms that may be omitted. Example:

```
[ name ]  
[ number ]
```

The syntactical unit in square brackets is optional.

- Braces ({} ) denote groupings. If items are stacked within the braces only one item may be used. Example:

```
{ name }  
{ number }
```

Choose either name or number.

- Three dots (...) denote that the preceding syntactical unit may be repeated one or more times in succession. Examples:

```
[,parameter]...
```

Any number of parameters may be specified; each parameter must be preceded by a comma.

digit[,...]

At least one digit must be specified; multiple digits must be separated by commas.

{name<sub>1</sub>=name<sub>2</sub>}[,...]

At least one name<sub>1</sub>=name<sub>2</sub> unit must be specified; multiple units must be separated by commas.

## DSS Functions

DSS functions represent data fields or perform functions when used as command operands. Each DSS function is represented by a unique name that has & as its first character.

In the following descriptions of DSS functions, "Access:" describes the access (read-only or read-write) permitted to the data field that a DSS function represents. "Access: N/A." means that the DSS function does not represent a data field. Unless otherwise noted, DSS functions that represent data fields can be used with immediate attribute designation.

### Machine Register Functions

These functions represent data fields that contain the values held in the registers when DSS received control; the registers are loaded from these data fields when DSS returns control to OS/VS. You can alter these data fields (except &PREFIX) using the SET command, or display them using the DISPLAY or DUMP command. The contents of the registers are typed as hexadecimal byte strings. Immediate attribute designation usually results in an error message and is not recommended.

In multiprocessing, the machine register functions can be qualified by &CPUID.

The register numbers are specified as decimal integers. If a range of registers is specified, the second must have a higher value than the first.

The machine register functions are:

&G( {register } )  
      { range }

Data-field attributes: offset=0, length=4, type=X, size=64. Treated as an array of sixteen elements.

Examples:

```
SET &G(0)=PARM
```

sets general register 0 to the contents of data field PARM.

```
DISPLAY &G(3:6)
```

displays general registers 3, 4, 5, and 6.

Access: read-write.

**&C**( { register }  
          { range } )

Refers to the four-byte contents of each specified control register. **Note:** The contents of OS/VS1 control registers 9, 10, and 11 should not be altered.  
Data-field attributes: offset=0, length=4, type=X, size=64. Treated as an array of sixteen elements.  
Access: read-write.

**Example:**

```
DUMP &C(0:15)
```

Dumps all of the control registers.

**&F**( { register }  
          { range } )

Refers to the eight-byte contents of the specified floating-point registers.  
Data field attributes: offset=0, length=8, type=X, size=32. Treated as an array of four elements.  
Access: read-write.

**Example:**

```
DISPLAY &F(0:4)
```

displays floating-point registers 0, 2, and 4.

**&PREFIX**

Refers to the contents of the prefix register.<sup>1</sup>  
This function, for multiprocessing systems only, can be qualified by &CPUID.  
Data field attributes: offset=0, length=4, type=X, size=4.  
Access: read-only.

**Program Status Functions**

These functions represent data fields that contain the program status words (PSWs) in effect when DSS received control.

A reference to an old PSW is to the basic (EC) PSW as well as to noncontiguous status information that is treated as a contiguous byte string. Examples of status information: interrupt code, I/O address, instruction-length code (ILC).

**In multiprocessing, the program status functions can be qualified by &CPUID.**

Data field attributes: offset=0, length=total of lengths shown in diagram, type=hexadecimal, size=length.

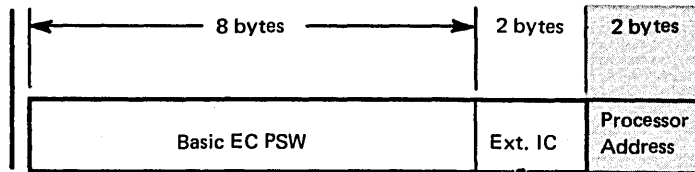
Access: read-write.

---

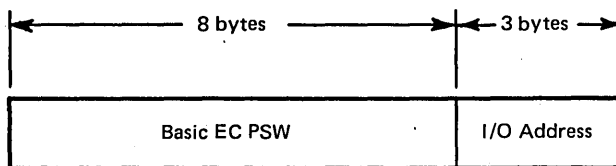
<sup>1</sup> See "Prefixing," in IBM System/370 Principles of Operation, GA22-7000.

The program status functions are:

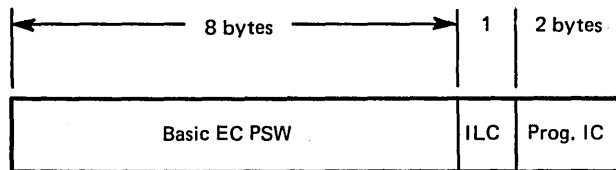
**&EPSW** external old PSW



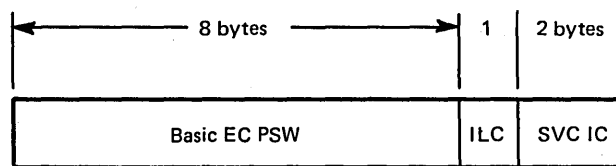
**&IPSW** I/O old PSW



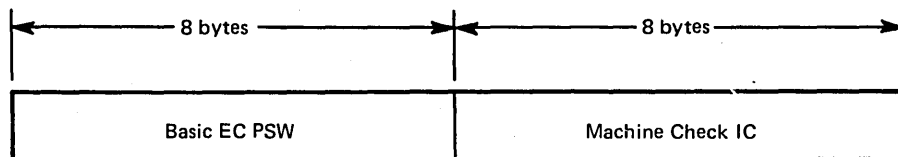
**&PPSW** program old PSW



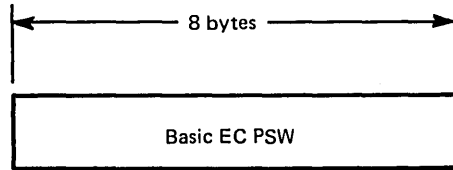
**&SPSW** SVC old PSW



**&MPSW** machine-check old PSW



&PSW            current PSW  
 &EPSWN        external new PSW  
 &IPSWN        I/O new PSW  
 &PPSWN        program new PSW  
 &SPSWN        SVC new PSW  
 &MPSWN        machine check new PSW  
 &RPSWN        restart new PSW  
 &RPSW         restart old PSW



refers to the PSW to be used when DSS returns control to OS/VS.

**Examples:**

```
IF &PPSW.(5)=X'FF'; DISPLAY &PPSW; END
```

If the sixth byte of the PPSW equals X'FF', DSS will display the PPSW. The next example is self-explanatory.

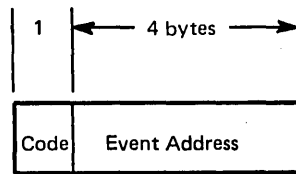
```
DISPLAY &PSW, &EPSW, &EPSWN, &PPSWN
```

**Machine Status Functions**

These functions represent data fields that contain miscellaneous machine status indicators that were current when DSS received control.

The machine status functions are:

&AC            activation code



The code part of the activation code indicates the type of event that activated DSS.

Code (Hexadecimal)	Event	
80	Module loaded (ON &LD)	} System Events
40	Module unloaded (ON &UNLD)	
20	Instruction fetch (ON &I)	} Program Events
10	Storage alteration (ON &SA)	
08	Successful branch instruction (ON &B)	
04	General register alteration (ON &RA)	
02	AT encountered	
01	Asynchronous interrupt (RESTART key)	

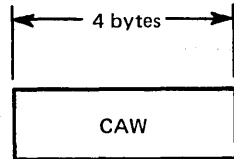
Multiple events may be indicated in the code. (When multiple events occur, the priority for processing their associated command procedures is the same as the order of the above list. That is, module loaded has highest priority; a module unloaded has second priority; etc.)

The event address is the address of the instruction that caused the breakpoint, for codes 20, 10, 08, 04, and 02. (That instruction has been executed.) The event address is meaningless for codes 80, 40, and 01.

Data field attributes: offset=0, length=5, type=X, size=5.

Access: read-only

**&CAW** channel address word



In multiprocessing, &CAW can be qualified by &CPUID.

Data field attributes: offset=0, length=4, type=X, size=4.

Access: read-write.

Example:

```

DISPLAY &CPUID(1).&CAW      input
&CAW 001DB0A8              output

```

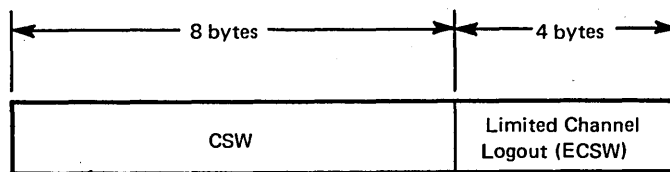
**&CID** channel identification<sup>1</sup>

In multiprocessing, &CID may be qualified by &CPUID.

Data field attributes: offset=0, length=4, type=X, size=4.

Access: read-write.

**&CSW** channel status word



In multiprocessing, &CSW can be qualified by &CPUID.

Data field attributes: offset=0, length=12, type=X, size=12.

Access: read-write.

<sup>1</sup> See IBM System/370 Principles of Operation, GA22-7000.

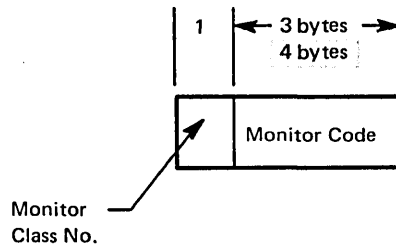
**&IOEL** I/O extended logout address<sup>1</sup>

In multiprocessing, &IOEL may be qualified by &CPUID.

Data field attributes: offset=0, length=4, type=X, size=4.

Access: read-write.

**&MC** monitor call



Data field attributes: offset=0, length=4 or 5, type=X, size=4 or 5.

Access: read-only.

**&PRM**

The register alteration mask. There is a one-to-one correspondence between the sixteen bits of this data field, reading from left to right, and the sixteen general registers 0-15. The initial setting of this data field is X'FFFF' (all registers set for monitoring).

Data field attributes: offset=0, length=2, type=X, size=2.

Access: read-write.

**Example:**

```
SET &PRM=X'2F40'
```

DSS will monitor general registers 2, 4, 5, 6, 7, and 9.

**Note:** &PRM cannot be qualified by &CPUID. Register monitoring is consistently the same for all CPUs.

**&RANGE**

Refers to the range of addresses to be monitored for all program events. Only one range of addresses can be in effect at one time. The initial setting of &RANGE is to all of OS/VIS storage, that is, &ASID(0).L'0:FFFFFF'.

Access: read-write.

**Note:** You can use immediate attributes to specify a range of addresses for &RANGE. For example,

```
SET &RANGE=L'03000:03099'
```

is equivalent to

```
SET &RANGE=L'03000'.(0,X'9A')
```

<sup>1</sup> See IBM System/370 Principles of Operation, GA22-7000.

**Examples:**

```
SET &RANGE=&LM( IGC2903D )
```

allows DSS to monitor for program events in OS/Vs load module IGC2903D.

```
SET &RANGE=&ASID( 2 ).L' 03000:03099'
```

changes the monitoring range for program events from the previous locations to locations X'3000-3099' in address space number 2. If &ASID(2) were omitted, DSS would supply the &ASID from &QT.

```
DISPLAY &RANGE          input
&RANGE 0000          00000000  00001000  output
```

Shows that the program-event monitoring range is locations X'0-1000' in all the address spaces.

**&TEA** translation exception address<sup>1</sup>  
In multiprocessing, &TEA may be qualified by &CPUID.  
Data field attributes: offset=0, length=4, type=X, size=4.  
Access: read-write.

**&TOD** time-of-day clock  
Indicates the time when DSS was activated; written as hours.minutes.seconds.  
Example: 12.30.30  
Data field attributes: offset=0, length=8, type=C, size=8.  
Access: read-only.

**Example:**

```
DISPLAY &TOD          input
&TOD    12.30.30      output
```

**Command-Related Functions**

The command-related DSS functions are:

**&AT**[( { name } | { number } [,...] )]

Refers to ATs by name or by number. An AT, created by the AT command, is control information that tells DSS where to interrupt OS/Vs and what to do after the interrupt. (See the description of the AT command later in this section.) If no name or number is specified, &AT refers to all ATs. &AT may be used as an operand of a DISPLAY, DUMP, REMOVE, ENABLE, or DISABLE command.

Access: read-only.

<sup>1</sup> See IBM System/370 Principles of Operation, GA22-7000.



**&ON**( $\left\{ \begin{array}{l} \text{name} \\ \text{number} \end{array} \right\}$  [,...])

Refers to ONs by name or by number. An ON, created by the ON command, is control information that tells DSS when to interrupt OS/VS and what to do after the interrupt. (See the description of the ON command, later in this section.) If no name or number is specified, &ON refers to all ONs. &ON may be used as an operand of a DISPLAY, DUMP, REMOVE, ENABLE, or DISABLE command.

Access: read-only.

**&PATCH**( $\left\{ \begin{array}{l} \text{name} \\ \text{number} \end{array} \right\}$  [,...])

Refers to patches by name or by number. (See the description of the PATCH command, later in this section.) If no name or number is specified, &PATCH refers to all patches. &PATCH may be used as an operand of a DISPLAY, DUMP, or REMOVE command.

Access: read-only.

**&PROC**([procedure-name[,...]])

Refers to command procedures stored by PROCEDURE commands. (See the description of the PROCEDURE command, later in this section.) If no name is specified, &PROC refers to all command procedures stored by PROCEDURE commands. &PROC may be used as an operand of a DISPLAY, DUMP, or REMOVE command.

Access: read-only.

**&SYM**([dss-data-field[,...]])

May be used as an operand of a DISPLAY, DUMP, or REMOVE command.

- In a DISPLAY or DUMP command, &SYM refers to data fields that you have defined by issuing a DEFINE command, EQUATE command, or “parameter” operand of a PROCEDURE command. If no “dss-data-field” is specified, &SYM refers to all data fields that you have defined.
- In a REMOVE command, &SYM refers to data fields that you have defined by issuing a DEFINE or EQUATE command. If no “dss-data-field” is specified, &SYM refers to all data fields that you have defined in DEFINE and EQUATE commands.

Access: read-only.

**Example:**

```
EQUATE MYNAME=IGC116
DISPLAY &SYM(MYNAME)
```

DSS replies:

```
&SYM NAME=MYNAME REF=EQUATED OFFSET=00000000 LNG=000001F0 TYPE=X
SIZE=000001F0 SCOPE=EXTERNAL QUAL=&SYM.IEANUC01.
```

**&P**( $\left\{ \begin{array}{l} \text{name} \\ \text{[name][,SNAP][,DISABLE]} \end{array} \right\}$ )

An optional operand in the AT, ON, and PATCH commands. (See the descriptions of these commands.)

Access: N/A.

### **&PRDMP[RM]**

An optional operand in the DUMP command; indicates a dump of virtual storage (&PRDMP) or real storage (&PRDMPRM) that is to be usable as input to the AMDPRDMP service aid program.

Access: N/A.

### **&S(dss-data-field[,...])**

Substitutes a predefined character string into the operand portion of a command statement.

#### **Example:**

```
DEFINE A=' &P( ,SNAP,DISABLE) '  
ON &RA, &B, &S(A);END  
AT LOC1, &S(A)  
DISPLAY LOC1.( 100,200 );END
```

The character string defined as A replaces &S(A), and the ON and AT commands appear to DSS as:

```
ON &RA, &B, &P( ,SNAP,DISABLE );END  
AT LOC1, &P( ,SNAP,DISABLE )
```

**Note 1:** You must have defined “dss-data-field” using DEFINE, EQUATE to a defined field, or the parameter operand of PROCEDURE. In addition, “dss-data-field” must have a type attribute of C.

**Note 2:** &Ss must not be nested. Example: &S(&S(A)) results in an error message.

**Note 3:** &S(A,B) is equivalent to &S(A),&S(B).

Access: N/A.

### **&DOUT**

Contains the I/O name for DUMP output.

#### **Example:**

```
SET &DOUT='PRINT1'
```

For DUMP output to be written out, &DOUT must be set to an I/O name and the I/O name must be assigned a device address. See “Input/Output,” later in this section.

Access: read-write

### **&SOUT**

Contains the I/O name for SNAP output. Example:

```
SET &SOUT='TOUT1'
```

SNAP output is stored in a wraparound buffer that is written to a device when (1) the buffer is full, (2) &SOUT is set to an I/O name, and (3) the I/O name is assigned a device address. See “Input/Output,” later in this section.

Access: read-write

## &HDR

Contains the subheading to be printed during execution of a DUMP command.

### Example:

```
SET &HDR='SAMPLE OF DSS DUMP'
```

establishes a subheading for all subsequent dumps, until the contents of &HDR are changed or the current DSS session ends.

Data-field attributes: offset=0, length=120, type=C, size=120.

Access: read-write

&M( {register } [...])  
      {range }

The mask function, which is provided to help set the register alteration mask. See the description of &PRM, earlier in this section.

### Example:

```
SET &PRM=&M( 2,4:7,9 )
```

has the same effect as

```
SET &PRM=X'2F40'
```

In each case, DSS will monitor general registers 2, 4, 5, 6, 7, and 9.

The digits specified by “register” or “range” correspond to those bit positions that are set to one in &PRM (or the first two bytes of any data field on the left of the equal sign). The remaining bits in those two bytes are set to zero.

Data-field attributes: each use of &M creates an immediate data field in DSS storage with offset=0, length=2, type=X, size=2.

Access: read-only.

## Attribute Functions

Attribute functions refer to the offset, length, type, and size of any data field that can be represented symbolically (that is, by a DSS function or an identifier).

Note: SET {&O }  
          {&L } (any DSS function) is an invalid command.  
          {&T }  
          {&SZ }

## &O(symbol)

The offset of the data field represented by “symbol”; a four-byte hexadecimal string.

### Example:

```
DISPLAY &O(ANSWER)      input  
&O ANSWER 000000A8     output
```

Access: read-write if “symbol” is an identifier, read-only if “symbol” is a DSS function or an identifier qualified by a DSS function.

### &L(symbol)

The length of the data field represented by "symbol"; a four-byte hexadecimal string.

#### Example:

```
SET &L(TABLE)=6
```

Access: read-write if "symbol" is an identifier, read-only if "symbol" is a DSS function or an identifier qualified by a DSS function.

### &T(symbol)

The type of the data field represented by "symbol"; a one-byte character string. Allowable values are:

- X = hexadecimal
- C = character
- I = decimal integer

#### Example:

```
DISPLAY &T(MESSAGE)      input
&T MESSAGE C             output
```

Access: read-write if "symbol" is an identifier, read-only if "symbol" is a DSS function or an identifier qualified by a DSS function.

### &SZ(symbol)

The size of the data field represented by "symbol"; a four-byte hexadecimal string.

#### Example:

```
DISPLAY &SZ(AREA)      input
&SZ AREA 0000014E     output
```

Access: read-only.

#### Example:

```
EQUATE MYNAME=IGC116
DISPLAY &O(MYNAME), &L(MYNAME), _ }      input
      &T(MYNAME), &SZ(MYNAME)
&O MYNAME 00000000 }
&L MYNAME 000001F0 }
&T MYNAME X      }      output
&SZ MYNAME 000001F0 }
```

### Location Qualifier Functions

The location qualifier functions are used to refer to OS/VS data fields. There are two types of location qualifier functions:

1. Map-ids — &SVM, &JOB, &TCB, and &PRB.
2. Miscellaneous location qualifiers — &ADDR, &ASID, &CPUID, &ID, &RM, &LM, &QT, and &Q.

These DSS functions are described below, in the same order.

### **&SVM**

A map-id that lets you access modules in supervisor virtual storage, that is, the nucleus and either the **link pack area (LPA)** for VS2 or resident reenterable module area (RRMA) for VS1. The SVM map list describes all load modules loaded into OS/VS virtual storage at IPL. All CSECTs and ENTRYs for the nucleus load module are defined. The LPA or RRMA map list contains only the descriptions of the load module names, aliases that are objects of loads, and entry points that are objects of IDENTIFY macros.

#### **Example:**

```
DISPLAY &SVM.IEANUC01.IEAQFX00.(X'1D0',2)
```

displays two bytes at offset X'1D0' from entry point IEAQFX00 in supervisor load module IEANUC01.

Access: N/A.

### **&JOB**( $\left. \begin{array}{l} \text{simple-address-literal} \\ \text{job-name} \end{array} \right\}$ )

A map-id that lets you access the modules in the currently executing step of the named job or addressed job-step TCB. CSECT and ENTRYs are included for modules that were loaded rather than link edited.

#### **Example:**

```
DISPLAY &JOB(JOB1).MOD1
```

displays all of module MOD1 in job JOB1.

Access: N/A.

### **&TCB**(simple-address-literal)

A map-id that lets you access a nonsharable subtask module associated with the addressed TCB.

#### **Example:**

```
DISPLAY &TCB(L'029B0').MOD2.CSECT1
```

displays all of CSECT1 in the nonsharable module MOD2 that is associated with the TCB at X'029B0'.

Access: N/A.

### **&PRB**(simple-address-literal)

A map-id that lets you refer to a specific copy of a load module; &PRB is needed only when duplicate copies of the same module are loaded for the same task. The PRB map list describes only one module.

#### **Example:**

```
DISPLAY &PRB(L'9810'%).MOD3.ENTRY1
```

displays the MOD3 associated with the PRB at the location pointed to by the four bytes at L'9810', from ENTRY1 to the end of module MOD3.

Access: N/A.





### **&ID(simple-address-literal)**

Represents an immediate data field with the following format:

address ... fully-qualified-symbolic-address

where “address” is the base address of the load module or the address of the nearest lower mapped symbol in the area of the address given with &ID and the fully qualified symbolic address is in the form:

**&ASID(number).map-id.module-name[. { csect-name  
entry-point name } ]**

See “Mapped Data Fields,” earlier in this section.

When the address that you specify with &ID does not reside in a mapped data field, “address” in the output is the specified address and is followed by

\*NOT IN LOAD MODULE\*

Data-field attributes: offset=0, length=variable, type=C, size=48.

Access: read-only.

### **&RM.address-literal**

Specifies a real address. Example:

```
DISPLAY &RM.L'1000'
```

All address literals are treated as virtual addresses unless qualified by &RM. An external symbol may also be qualified by &RM if its address is within the virtual=real range.

**Note:** The prefixed storage area (PSA) is in L'0:FFF', not &RM.L'0:FFF'.

Access: N/A.

### **&LM(module-name)**

Indicates that the name in parentheses is a module name, not a CSECT name.

Example:

```
DISPLAY &LM(IGC0503D)
```

displays load module IGC0503D.

**Note 1:** &LM must not be preceded by a map-id.

**Note 2:** &LM is needed only for unqualified module names that differ from the module name in &QT. See “Establishing Default Qualification,” below.

Access: N/A.

### **&QT**

Sets the qualify table so that the map-id and load module need not be repeatedly specified. See “Establishing Default Qualification,” below. Although &QT is not a data field, it can be altered by the SET command, displayed, or dumped.



**Note 1:** The initial default value of &QT is &CPUID(a).&ASID(b).&SVM.IEANUC0c, where a is the number of the CPU in which the RESTART interrupt occurred, b is the number of the OS/VS address space that was active in that CPU, and c is the number of the currently-used nucleus. After DSS monitors OS/VS, the default value of &QT changes to show what module was interrupted.

**Note 2:** &QT allows you to restrict your symbolic addressing to a single area. This could be useful when you use DSS to debug a problem program.

**Note 3:** The &QT setting must include at least a map-id or load module name.

Access: N/A.

**&Q**

Specifies that the contents of the qualify table are to be used for symbolic address resolution. See "Establishing Default Qualification," which follows.

Examples:

```
SET &Q.IEAQFX00.(X'1D0',2)=X'4700'
DUMP &Q
```

Access: N/A.

**Establishing Default Qualification:** If the same maps or load modules are frequently referred to, you can avoid retyping the explicit qualification by use of the qualify-table function (&QT), the qualify function (&Q), and the load-module function (&LM). Figure 6 illustrates the optional use of these functions.

Long Form of Symbol	Short Form After SET &QT=map-id1	Alternate Forms Of Symbol After SET &QT=map-id1.module1
&ASID(x).map-id1.module1	&LM (module1)	&Q
&ASID(x).map-id1.module1.csect1	module1.csect1	csect1
&ASID(x).map-id1.module1.entry-point1	module1.entry-point1	entry-point1
&ASID(x).map-id1.module2	&LM (module2)	&LM (module2)
&ASID(x).map-id1.module2.csect2	module2.csect2	module2.csect2
&ASID(x).map-id1.module2.entry-point2	module2.entry-point2	module2.entry-point2
<p><b>Note:</b> &amp;CPUID(y) is also in &amp;QT, and can be set along with the other qualifiers by attaching it to them as the leftmost qualifier.  <b>Example:</b> SET &amp;QT=&amp;CPUID(1).&amp;ASID(3).&amp;SVM.IEANUC01            This attachment of &amp;CPUID to &amp;ASID is allowed only when it follows SET &amp;QT=.</p>		

Figure 6. Use of &Q, &QT, and &LM to Specify Symbolic Addresses

Examples for Figure 6:

1. SET &QT=&ASID(1).&JOB(JOBA) would allow you to use short form 1 for references to locations in JOBA of address space 1. For example, you could then refer to module XYZ in JOBA of address space 1 as simply &LM(XYZ) (first line under short form 1), rather than &ASID(1).&JOB(JOBA).XYZ (first line under long form).
2. SET &QT=&ASID(1).&JOB(JOBA).XYZ would allow you to use short form 2 for references to locations in JOBA of address space 1.

For example, you:

- Can refer to CSECT IEADEF module XYZ as IEADEF (second line under short form 2), rather than &ASID(1).&JOB(JOBA).XYZ.IEADEF (second line under long form).
- Can refer to module XYZ as &Q (first line under short form 2), rather than &ASID(1).&JOB(JOBA).XYZ (first line under long form).
- Can refer to module ABC in JOBA as &LM(ABC) (fourth line under short form 2), rather than &ASID(1).&JOB(JOBA).ABC (fourth line under long form).
- Must refer to CSECT IEAQFX in supervisor virtual storage for address space 2 as &ASID(2).&SVM.IEANUC0d.IEAQFX (second line of long form), because &ASID(2) and &SVM are not in &QT.

Map Functions

The map functions let you display the various maps maintained by DSS, using a DISPLAY or DUMP command. These functions are valid only as operands of the DISPLAY and DUMP commands.

The map functions are:

$$\&SVM\text{MAP}(\left\{ \begin{array}{l} \text{NUC} \\ \text{LPA} \\ \text{ALPA} \\ \text{RRMA} \end{array} \right\})$$

A map of the supervisor virtual storage. The optional suboperands are used to select portions of the SVM map:

NUC — a map, ordered by address, of the CSECTS and ENTRYs in the OS/VS nucleus.

LPA — (OS/VS2 only) an unordered map of the modules listed in the link-pack-area directory.

ALPA — (OS/VS2 only) an unordered map of the modules on the active link-pack-area queue.

RRMA — (OS/VS1 only) an unordered map of the modules in the resident reenterable module area.

If no suboperand is specified, in OS/VS1 NUC is displayed; in OS/VS2, all maps are displayed.

Access: read-only.

`&JOBMAP( { job-name  
simple-address-literal } )`

An unordered map of the sharable and nonsharable modules associated with the currently executing step of the named job or the addressed job-step TCB. `&JOBMAP` cannot be qualified by `&ASID`; DSS uses the `&ASID` in `&QT`.  
Access: read-only.

`&TCBMAP (simple-address-literal)`

An unordered map of the nonsharable modules associated with the addressed TCB. `&TCBMAP` cannot be qualified by `&ASID`; DSS uses the `&ASID` in `&QT`.  
Access: read-only.

**Example:**

```
DUMP  &SVMMAP( NUC )
```

umps the DSS map of the OS/VS nucleus.

**Event Functions**

While an OS/VS program is executing, it can be monitored for the occurrence of certain program events and system events. The occurrence of a monitored-for event causes an interrupt, followed by the execution of DSS commands that you specified as subcommands of the `ON` command.

**Program Events:** The program-event functions are (see “DSS Monitoring,” in Section 1):

**&B**

Execution of a successful branch instruction in the monitored portion of OS/VS storage.

Access: N/A.

**&I**

Fetching and execution of any instruction in the monitored portion of OS/VS storage.

Access: N/A..

**&SA**

Alteration of monitored OS/VS storage. The contents of OS/VS storage are considered to have been altered whenever the CPU executes an instruction that causes the whole operand or part of it to be stored in the monitored portion of OS/VS storage.

Access: N/A.

**&RA**

Alteration of a designated general register by the execution of an instruction in the monitored portion of OS/VS storage. Recognition of this event is not contingent on the new value being different from the previous one.

Access: N/A.

**Note 1:** In conjunction with register monitoring, a mask of the registers to be monitored must be set by the `&PRM` function. Example:

```
SET  &PRM=&M( 3:6,9 )  
ON  &RA; DISPLAY &G( 3:6 ), &G( 9 );END
```

**Note 2:** All the above program-event functions are subject to an address-monitoring range (set by &RANGE) over which the requested program event recording is to be in effect. Example:

```
SET &RANGE=&LM( IGC004 )
```

indicates that the portion of OS/VIS storage to be monitored is the load module IGC004.

**System Events:** The system-event functions and the events that they symbolize are:

&LD(module-name[,...])

Loading of the specified modules.

Access: N/A.

&UNLD(module-name[,...])

Unloading of the specified modules.

Access: N/A.

**Example:**

```
ON &LD( IGC2903D ), &P( , SNAP );END
```

When module IGC2903D is loaded, DSS will save information for a snapshot dump.

**Note 1:** &LD and &UNLD are independent of &RANGE. They are, however, subject to the contents of the qualify table, set by &QT, at the time the ON command is executed. Each module name in the parenthesized list will be qualified by the map-id and the &ASID in the qualify table; the parenthesized list must not contain map-ids or &ASIDs. For &LD, the only map-ids that are meaningful are &JOB and &TCB. For &UNLD, the map-id &PRB may also be used.

**Note 2:** The number of system events that can be monitored at one time is limited only by the amount of storage available for DSS work areas. See Appendix A.

## Expressions

A DSS expression is an arithmetic, comparison, or boolean operation, or a combination of these operations. Any expression returns a single immediate data field. The attributes of this data field depend on the last operation performed.

Expressions are evaluated left to right, subject to these exceptions:

- When more than one prefix operator operates on the same data field, the prefix operators are evaluated right to left.
- The conventional algebraic rules for parentheses are applied.
- Operators of higher priority (where priority 1 is higher than priority 2, etc.) are evaluated before operators of lower priority. Figure 7 shows the hierarchy of operator priorities.

Priority	Operators
1	prefix +, prefix -, ~
2	*, /, //
3	+, -
4	>, ~>, >=, =>, =, ~=, =<, <=, <, ~<
5	!, &&

Figure 7. Operator Priorities

### Arithmetic Operators

All arithmetic operations can be performed on either type I or type X data fields. If an input data field is type C, an error message is issued.

All arithmetic operations are performed with four-byte data fields. (This forces type I data fields to represent integer values between  $-2^{31}$  and  $+2^{31}-1$ , and type X data fields to represent integer values between 0 and  $+2^{32}-1$ , before they can be operated on arithmetically.) However, the input data fields need not have length attributes of 4. If the length attribute of an input data field is not 4, a temporary four-byte data field of the same type (I) or (X) is established. The input data field is moved into the temporary data field, following the rules of data movement for I from I or X from X as shown in the table of truncation and padding for the SET and DEFINE commands, later in this section. If truncation cannot be performed, or if the length attribute is over 256, an error message is issued.

When the name of a data field is used in an arithmetic expression, the operation is performed on the contents of the data field, not on its address. For example,  $A+1$  results in one being added to the contents of the data field whose name is A. (This addition occurs in DSS working storage; the original value of data field A is unchanged.)  $A.(1)$ , on the other hand, specifies a one-byte offset from the address of A.

**Prefix Arithmetic Operators:** A prefix arithmetic operator produces an output data field with the attributes:

```
offset = 0
length = 4
type = same as input
size = 4
```

If the input field is type X, it must represent an integer between 0 and  $+2^{32}-1$ .

The prefix minus operator produces an output data field that is the algebraic negative of the input value.

The prefix plus operator produces an output data field that has the same algebraic value as the input value.

**Infix Arithmetic Operators:** The infix arithmetic operators perform their functions in a true algebraic sense.

If both input data fields are type I, the output data field has the attributes:

```
offset = 0
length = 4
type = I
size = 4
```

The output value from the operation must be between  $-2^{31}$  to  $+2^{31}-1$ .

If one or both of the input data fields are type X the output data field has the attributes:

offset = 0  
length = 4  
type = X  
size = 4

The operation's output value must be between 0 and  $+2^{32}-1$ .

### Comparison Operators

A comparison operator compares two input data fields and returns an output data field that indicates whether the specified relationship is true or false. The attributes of the output data field are:

offset = 0  
length = 1  
type = X  
size = 1

This data field contains X'FF' if the relationship is true and X'00' if the relationship is false.

If neither input data field is type C, the comparison is algebraic. An input data field with a type attribute of X is assumed to represent an integer between 0 and  $+2^{8n}-1$ , where n is the length attribute of the data field. An input data field with a type attribute of I is assumed to represent an integer between  $-2^{8n-1}$  to  $+2^{8n-1}-1$ , where n is the length attribute of the data field.

If both input data fields are type C or if one is type C and the other type X the comparison is binary. The shorter data field is padded on the right with blanks to make the lengths equal, and bit-by-bit comparison is performed from left to right. If all bits are the same, the operands are considered equal. The first occurrence of an unmatched pair of bits indicates that the operands are not equal, and the operand with a 1 in that bit position is considered larger.

A comparison between type C and type I data fields is invalid.

### Boolean Operators

Boolean operators can be used with data fields that have any type attribute and any length attribute.

**Prefix NOT:** The prefix NOT operator produces an output data field with the attributes:

offset = 0  
length = length of input data field  
type = X  
size = length

The output data field contains a bit-complement image of the input data field.

**Infix AND, Infix OR:** The infix boolean operators produce output data fields with the attributes:

offset = 0  
length = the length of the shortest input data field  
type = X  
size = length

The AND operator is applied bit by bit to the leftmost L bytes of the longer input data field and all of the shorter input data field, where L is the length of the shorter data field.

The OR operator is applied bit by bit to the leftmost L bytes of the longer input data field and all of the shorter input data field, where L is the length of the shorter data field.

### Examples of Expressions

$3+6*2$

produces an immediate data field with the attributes: offset=0, length=4, type=I, and size=4. The data field contains the integer +15.

$A+B>X'9E'$

does the algebraic addition of the contents of data fields A and B and compares the result algebraically with X'9E'. The data field produced has the attributes: offset=0, length=1, type=X, size=1. This data field contains X'FF' or X'00', depending on the results of the comparison.

$(A=B \& \& X'01') * 25 + 256$

produces a data field with the attributes: offset=0, length=4, type=X, size=4. The contents are X'00000100' if  $A \neq B$ , or X'00000119' if  $A=B$ .

$\neg(A < B)$

is the equivalent of  $A \geq B$ .

$A=B | A=C$

is evaluated as true if  $A=B$  or  $A=C$ .

$(A=B | A=C) \& \& D=X'FF'$

is evaluated as true if  $A=B$  or  $A=C$ , and  $D=X'FF'$ .

## Input/Output

DSS has a predefined set of I/O names to service I/O needs. These I/O names are assigned for the debugging session by the ASSIGN command and removed from the debugging session by the RELEASE command.

During the debugging session, these I/O names are used by the DIVERT command, the SNAP capabilities (AT and ON), and the DUMP command to indicate the I/O device, type (input or output), and record format of the I/O operation. The I/O names are:

I/O Name	Device
CARD	A card reader used to read the command stream
CMDT	A magnetic tape used to write a copy of the PROCEDURE command and its associated command procedure
LOG	A printer or magnetic tape used to create a hardcopy log of the DSS session
PRINT1	A printer used to print SNAP information (AT, ON) and dumps (DUMP)
PRINT2	Same as PRINT1
PUNCH	A card punch used to write a copy of the PROCEDURE command and its associated command procedure
TIN	A magnetic tape used to read the command stream
TOUT1	A magnetic tape used to write SNAP information (AT, ON) and dumps
TOUT2	Same as TOUT1

I/O names must be assigned, by means of the ASSIGN command, before they can be used.

**Note:** CMDT and PUNCH are for use with the DUMP command, when listing PROCEDURE commands and associated command procedures via the &PROC DSS function. Magnetic tapes (in card-image format) or card decks produced in this manner may be read by issuing the DIVERT TIN or DIVERT CARD commands.

Figure 8 lists the valid I/O-name and DSS-command combinations. Figure 9 shows how data is formatted on magnetic tapes used by DSS commands.



I/O Name \ Related Commands	ASSIGN i/o-name = device-address	DIVERT [i/o-name]	RELEASE i/o-name [ , . . . ]	SET & DOUT = 'i/o-name'	SET & SOUT = 'i/o-name'
CARD	X	X	X		
CMDT	X		X	X	
LOG	X		X	X	X
PRINT1 and PRINT2	X		X	X	X
PUNCH	X		X	X	
TIN	X	X	X		
TOUT1 and TOUT2	X		X	X	X
none (integrated operator's console)		X			
<p><b>Note:</b> The maximum size of a single DSS command, including continuation underscores, is 256 characters. This is also the maximum input record allowed from the integrated operator's console.</p>					

Figure 8. I/O Names and Related Commands

I/O Name \ Data Attributes	Logical Record Length	Blocking Factor	Physical Record Length
CMDT	80	20	1600
LOG (when magnetic tape)	121	1	121
TIN	80	1 to 25	2000
TOUT1 and TOUT2	121	1	121
<p><b>Notes:</b></p> <ul style="list-style-type: none"> <li>• DSS works with labeled and unlabeled 800 and 1600 BPI tapes for input but only with unlabeled tapes of those densities for output.</li> <li>• DSS does not support seven-track magnetic tapes.</li> <li>• TOUT1 and TOUT2 can be printed by means of the IEBPTPCH or IEBGENER utility program or the AMDPRDMP service aid program. Since DSS output tapes are unlabeled, DCB information must be supplied to the utility or service aid program.</li> </ul>			

Figure 9. Magnetic Tape Data Attributes

## The SNAP Dump

The SNAP capability is provided for the AT and ON commands only. This is done via the &P DSS function, which allows the specification of the SNAP suboperand.

There are two basic types of SNAP records:

Type 1 is for

AT	Any AT command with &P(SNAP)
ON &B	Branch
ON &I	Instruction Fetching
ON &RA	Register Alteration
ON &SA	Storage Alteration

Type 2 is for

ON &LD(module-name)	Loading of a module
ON &UNLD(module-name)	Unloading of a module

For a list of the output information provided for each type, see Figure 10.

Commands and Events SNAP Information Provided	AT	ON					
		&B	&I	&RA	&SA	&LD	&UNLD
SNAP Record Type	1	1	1	1	1	2	2
Activation code as Hexadecimal Digits	X	X	X	X	X	X	X
Address of Instruction Causing Activation, in Hexadecimal	X	X	X	X	X		
Basic Old PSW (Address of NSI, etc.), in Hexadecimal	X	X	X	X	X		
Instruction causing the Interrupt, in Hexadecimal	X	X	X	X	X		
General Purpose Registers 0-15 (After Instruction Causing Interrupt), in Hexadecimal	X	X	X	X	X		
Module Name Being Loaded or Unloaded as a Character String						X	X
Module Address (Beginning Location), in Hexadecimal						X	X
Module Length, in Hexadecimal						X	X
X = This Information is Provided							

Figure 10. SNAP Output Information

If the AT location or ON event is encountered and you specified the SNAP operand for the AT or ON command, then a SNAP record is stored in a SNAP buffer that is defined and maintained by DSS. SNAP records are accumulated in this SNAP buffer until it is full.

When the SNAP buffer is full, the following happens:

1. If the &SOUT DSS function has not been set to a valid I/O name (LOG, PRINT1, PRINT2, TOUT1, or TOUT2), then the buffer will wrap around by placing the next SNAP record over the first record in the SNAP buffer, the next over the second, etc., and then wrapping again and again as required.
2. If the &SOUT DSS function has been set to a valid I/O name, then when the SNAP buffer is full, the SNAP information is formatted as explained in "Section 5: Output Formats," and sent to the printer or magnetic tape. The next SNAP record is placed at the beginning of the now emptied SNAP buffer, and the entire procedure is repeated.

**Note:** When the DISCONNECT command is issued to terminate DSS, SNAP records in the partly filled buffer are formatted and written as explained above. Before issuing the DISCONNECT command, you must set &SOUT to one of the valid DSS output devices in order to obtain the records that are presently in the SNAP buffer. If this is not done, those records are lost.

Example:

```
ASSIGN PRINT1=14
SET     &SOUT='PRINT1'
AT     X, &P( , SNAP );END
```

SNAP information will be printed.

### The Hardcopy Log

The hardcopy log facility allows you to keep a separate record of input from the integrated operator's console and output to the integrated operator's console. Although this facility is primarily for hardcopy backup of display consoles, it can also be used if the integrated operator's console employs a printer-keyboard.

To start the hardcopy log, issue the command `ASSIGN LOG=`, where the equal sign is followed by a device address in the form of a decimal or hexadecimal literal. The addressed device must be a tape device or a printer that is supported by DSS and is not already assigned to an I/O name. For a complete log of the DSS session, this command should be the first one issued; however, it can be issued at any time during the session.

To combine console I/O with DUMP or SNAP output in the hardcopy log, issue `SET &DOUT='LOG'` or `SET &SOUT='LOG'`.

You can release LOG at any time. This stops the hardcopy log output and allows you to change the hardcopy log device or magnetic-tape volume.

## Commands

The commands, in the order in which they are described, are: \*(comment), ASSIGN, AT, COLLECT, DEFINE, DISABLE, DISCONNECT, DISPLAY, DIVERT, DUMP, ENABLE, END, EQUATE, GO, GOTO, IF, INVOKE, ON, PATCH, PROCEDURE, RELEASE, REMOVE, RETURN, REVERT, and SET.

Each command description includes

- A description of the command's function.
- An illustration of the command's syntax, excluding the label, semicolons, and blanks; the rules for using labels, semicolons, and blanks are explained earlier in this section, under "Blanks" and "Contents of a Command."
- Programming notes, if applicable.
- For most commands, one or more examples of usage.

The general format of DSS commands is:

```
{[label:]verb[boperand[,...]] [:...]
```

Most commands require one or more operands.

Any operand that is a representation of a data field can have an immediate attribute designation appended to it. See "Immediate Attribute Designation," under "Data Fields," earlier in this section.

**Note:** The &S DSS function must not be used in the label or verb field of a command.

DSS executes a command whenever possible. A command that specifies an incorrect action may be executed if it is syntactically correct.

Diagnostic messages are written to the console. These messages are listed and explained in Section 6. Error-handling procedures are described in Section 6, under the "Programmer Response" headings, and in Section 4, under "How to Correct an Invalid Command."

### The Comment Command

The comment command is a non-executable command that allows the placing of comments between commands; the comments can be stored in a command procedure or inserted on the console sheet. If the comments are stored in a command procedure, they will be written when the command procedure is displayed or dumped.

---

Command Verb	Operands
*	[comment]

---

comment

Any combination of characters from the DSS character set is valid (apostrophes must be paired); however, a semicolon cannot be part of the comment unless the semicolon is embedded in apostrophes, like this: ';

**Example:**

```
PROCEDURE SQUARE,A
* THIS PROCEDURE SQUARES THE VARIABLE A
SET &S(A)=&S(A)*&S(A);*THIS LINE CONTAINS THREE COMMANDS;END
```

## ASSIGN

The ASSIGN command allocates an I/O device to DSS, then opens the corresponding data set; these resource assignments remain effective until they are released by RELEASE or DISCONNECT.

Command Verb	Operands
{ ASSIGN ASGN }	( CARD PUNCH PRINT1 PRINT2 TIN CMDT TOUT1 TOUT2 LOG ) = device-address [...]

CARD  
PUNCH  
PRINT1  
PRINT2  
TIN  
CMDT  
TOUT1  
TOUT2  
LOG

An I/O name. See "Input/Output," earlier in this section.

device-address

The physical address of the device that is to be used.

Specified as: a hexadecimal or decimal literal.

**Note 1:** Any data set that is to be used by DSS must be allocated and opened by an ASSIGN command prior to its use.

**Note 2:** Return of control to OS/Vs by means of a GO command does not terminate allocation of the device for DSS use.

**Note 3:** If the device is already allocated to OS/Vs or was being activated at the time of DSS allocation, a message will inform you of this fact and give you the opportunity to cancel the ASSIGN command. A device should not be allocated to both DSS and OS/Vs unless absolutely necessary, because DSS cannot pass OS/Vs the I/O interrupts that are pending for OS/Vs on that device. In addition, if it is an output device, DSS and OS/Vs output could be mixed. Finally, the issuing of a GO command before the device is released from DSS use may lead to unpredictable results.

**Note 4:** For TIN, if the tape is labeled, ASSIGN spaces forward over the label.

**Example 1:**

```
ASSIGN PRINT1=14
```

allocates a printer with address 14 and opens an output data set (for use by DUMP or SNAP via the &DOUT and &SOUT DSS-function settings).

**Example 2:**

```
ASSIGN TIN=X'280'
```

allocates tape device 280 and opens the data set for input only (for use by a DIVERT command).

## AT

The AT command establishes breakpoints in executable code at which DSS is to be activated and a SNAP dump and/or command procedure is to be executed.

Command Verb	Operands	Text Commands
AT	location[,...][,&P([name][,SNAP][,DISABLE])]	[command procedure] END

### location

Specifies a breakpoint in executable code. Each time execution reaches the location, an interrupt occurs and actions indicated by the AT command are performed.

**Specified as:** any representation of a data field that is within an area of OS/VS virtual storage that is monitorable by DSS. (For a list of the areas monitorable by DSS, see "Appendix A: Restrictions".) It is assumed that the first byte of the data field's referenced string (base + offset) is the first byte of a machine instruction.

### name

Optional. If specified, it is the name of the AT for use in the DISABLE, DISPLAY, DUMP, ENABLE, and REMOVE commands. (In addition, a unique number is generated for the AT by DSS. This number, printed at the console when the AT is read, can be used instead of a name to refer to the AT.)

**Specified as:** an identifier.

### SNAP

Optional. If specified, each time one of the AT breakpoints occurs a predefined set of SNAP information will be recorded. If &SOUT is set to an I/O name that is assigned a device address, the SNAP information will be printed.

### DISABLE

Optional. If specified, it indicates that the breakpoints are not to be established at this time. However, the command procedure is saved. Subsequently, the ENABLE command can be used to establish the breakpoints. The ENABLE or DISABLE is by name or number and therefore applies to all locations specified in a single AT command.

command procedure

Optional if SNAP is specified. If SNAP is not specified, at least one DSS command must be specified before the END command. Any DSS commands are valid.

END

Delimits the command procedure; must be specified as the last or only text command.

**Note 1:** This sequence of events occurs if the breakpoint is encountered during OS/VS execution:

1. The instruction specified by the AT command's location operand is executed.
2. If a SNAP dump was requested, it is taken.
3. If a command procedure was specified, it is executed. If execution reaches the end of the text commands, control is returned to OS/VS; that is, a GO command is simulated.

**Note 2:** DSS checks the syntax of the text commands as you enter them. However, a nested command procedure is not syntax checked until the command procedure that contains it is executed.

**Note 3:** In OS/VS1, the qualification in the qualify table (&QT) when the command procedure is stored is the qualification that will be used when the command procedure is executed. In OS/VS2, the command procedure runs under the current &QT qualification.

**Note 4:** The AT remains effective (can be reused, if enabled) until a DISCONNECT command is executed, until the AT is removed by a REMOVE command, or until the module containing the breakpoint is unloaded. If the AT is specified for locations in two or more load modules, each unloading of one of these modules removes the AT for that module only.

**Note 5:** If the AT location is in the OS/VS1 machine check handler, the AT will be ignored when that location is reached.

**Note 6:** The AT location must not be an instruction that will be modified or is the subject of an Execute instruction.

**Note 7:** DSS does not accept an AT for a location containing:

- An invalid operation code.
- An OS/VS1 Diagnose instruction.
- An OS/VS1 Monitor Call instruction.
- An Execute instruction.
- An OS/VS1 Set System Mask instruction.
- An emulator compatibility-feature instruction.
- An enabled AT, unless the new AT is disabled.

**Note 8:** In OS/VS1, DSS does not accept an AT for a location in:

- The I/O supervisor.
- The first instruction of each first-level interrupt handler.
- NIP.
- DSS.

In OS/VS2, DSS does not accept an AT for a location in:

- Part of the restart interrupt handler.
- The part of the machine check handler that owns the new PSWs.
- Part of the stop/restart subroutine.
- Wake-up code in vary online.
- PSAs.
- NIP.
- DSS.

**Note 9:** DSS does not accept an AT for any OS/VS1 transient area.

**Note 10:** The number of ATs and ONs that can be enabled at the same time is limited by the amount of space that is free for storage of their control blocks. On the average, this is about 64 ATs and ONs in OS/VS1, and about 253 ATs and 92 ONs in OS/VS2.

**Note 11:** Use of a SIGP RESTART instruction patched into OS/VS by DSS is not recommended as a substitute for an AT command. Although it is very likely that this patched-in instruction would have the same effect as the RESTART key, it might instead lead to unpredictable results.

**Example 1:**

```
AT &LM(IGC2903D), &P(LP5, SNAP)
  DISPLAY 'IGC2903D TRAP', &G(3)
END
```

Execution of the breakpoint at load module IGC2903D will result in collecting SNAP information, displaying of a message and general register 3 at the console, and resumption of OS/VS execution.

**Example 2:**

```
AT &ASID(3).L'1EF20', &P(, SNAP); END
```

Execution of the breakpoint at location 1EF20 for the third address space will result in collecting SNAP information and the resumption of OS/VS execution.

**Example 3:**

```
AT L'4C'%. (160); DISPLAY AREA; END
```

Execution of the instruction 160 bytes from the location pointed to by L'4C'% will result in the display of AREA at the console and the resumption of OS/VS execution.

## COLLECT

The COLLECT command copies data from one data field into another data field. When more than one execution of COLLECT causes data to be copied into the same data field, the command causes the data to be placed in successive elements of the field.



Command Verb	Operands
{ COLLECT } COL	{ dss-data-field = data-field <sub>2</sub> } [...]

**dss-data-field**

The name of the data field to be collected into.

**Specified as:** an identifier; since the identifier represents a data field, you can append an immediate attribute designation to it.

**data field<sub>2</sub>**

The data field to be collected from.

**Specified as:** any representation of a data field that is no more than 2048 or 4096 bytes in length.

The data is moved in accordance with the length attribute of the second operand; the offset attribute of the first operand (that is, the starting point for subsequent data collection) is then incremented by the length attribute of the second operand. This is the mechanism by which successive elements of the collection area are filled by successive executions of COLLECT commands that specify the area.

When the length of the second operand is greater than the remaining space (size minus offset) in the collection area, the offset is reset to zero before the data is moved. Thus, a collection area is like a wrap-around buffer.

If, after the data is moved, the remaining space in the collection area is less than one collection-area element, that is, less than the length attribute of dss-data-field, the offset is reset to zero. This avoids accidental displaying or setting of data beyond the end of the collection area.

**Note 1:** If the COLLECT command specifies the collection area's offset with an immediate attribute designation, that offset is used as the new starting point for offset calculations.

**Note 2:** A single COLLECT command can be executed repetitively; each repetition moves the same amount of data into the collection area. On the other hand, if separate COLLECT commands move data into a common collection area, the amounts of collected data may differ in length.

**Note 3:** If you have used a data field as a collection area and you want to refer to it symbolically with assurance that its offset attribute is zero, qualify the data-field name with an immediate attribute designation that specifies an offset of zero. If the command that specifies an offset of zero is not COLLECT, the offset will return to its old value after the command is executed.

**Note 4:** To determine which element of a collection area is the next available space for a COLLECT command, display the offset attribute with DISPLAY &O(name). You might not know how many times the offset has been reset to zero, unless inspection of the collection area reveals that information, or unless you use an IF command to display the collection area periodically as in Example 3.

**Note 5:** If the length attribute of the second COLLECT operand exceeds the size attribute of the first operand, the data is truncated on the right.

**Note 6:** Negative subscripts are useful in referring to partly filled collection areas.  
 Example: To display the last six entries in collection area A, issue DISPLAY A(-6:-1) (If offset is less than length\*6, an error message is issued.)

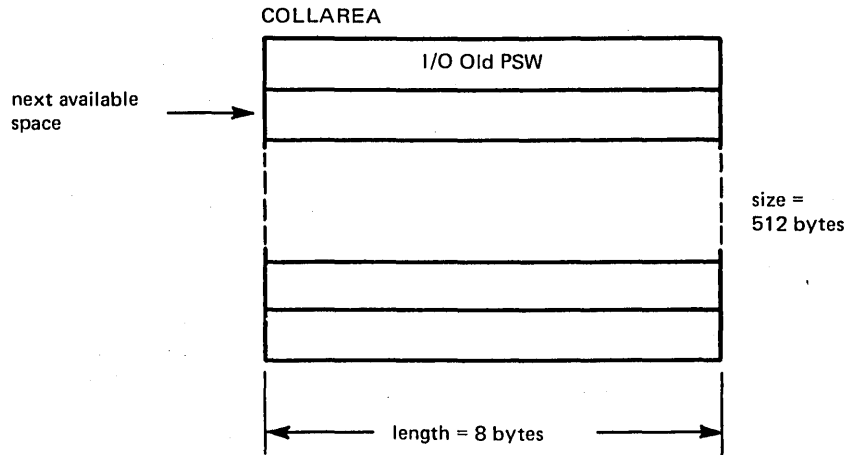
**Example 1:**

```
COLLECT FLDA=&C(1),FLDB=CODE
```

**Example 2:**

```
DEFINE COLLAREA.(,8,X,8*64)
EQUATE INTPROC=&SVM.IEANUC01.IEAQFX00.(X'100',2)
AT INTPROC;COLLECT COLLAREA=&IPSW.(,8);END
```

Each time the instruction location INTPROC is reached, the COLLECT command is executed and the I/O old PSW is stored in successive eight-byte fields of COLLAREA. The arrow in Figure 11 points to the next available space (that is, the referenced string) at base plus offset of COLLAREA after one execution of the COLLECT command.



**Figure 11. Collection Area**

In Figure 11, the next available space is referred to as COLLAREA(0), and the previous space is referred to as COLLAREA(-1). See Note 6, above.

**Example 3:**

```
AT INTPROC
COLLECT COLLAREA=&IPSW.(,8)
IF &O(COLLAREA)=0
  DISPLAY COLLAREA.(0,512)
END
END
```

## DEFINE

The DEFINE command creates a named data field in DSS storage and optionally sets the data field's contents to an initial value.

Command Verb	Operands
$\left\{ \begin{array}{l} \text{DEFINE} \\ \text{DEF} \end{array} \right\}$	{data-field <sub>1</sub> -name[ = data-field <sub>2</sub> ]}[,...]

data-field<sub>1</sub>-name

The name of the data field created.

**Specified as:** an identifier that is not already in use as the name of a data field; since the identifier represents a data field, you can append an immediate attribute designation to it.

data field<sub>2</sub>

A data field that contains a value to which the created data field is to be initialized.

**Specified as:** any representation of a data field that is not more than 2048 or 4096 bytes in length.

When no initialization is specified, the attributes of the created data field are either determined from the optional immediate attribute designation or by default. The default values are:

offset = 0  
length = 4  
type = X  
size = offset + length

When initialization is specified and immediate attributes are specified for the created data field, the referenced string of the created data field is initialized from the other data field, following the rules of padding and truncation given later in this section, with the description of the SET command. When initialization is specified and no immediate attributes are specified for the the created data field, the default attributes are:

offset = 0  
length = same as length of other data field  
type = same as type of other data field  
size = length

**Note 1:** Once defined, a data field exists until it is deleted by the REMOVE command or the DISCONNECT command.

**Note 2:** When length and size are not equal, it is usually because the data field is being defined as a collection area or an array. See Example 1. Although it is not mandatory that a collection area's length be a fraction of its size, this technique usually facilitates reference to the individually collected data items.

**Example 1:**

```
DEFINE AREA.(,8,,512)
```

As an array (or collection area), AREA consists of 64 eight-byte elements. DISPLAY AREA causes eight bytes to be written. The location of the eight bytes depends on AREA's offset, which does not remain zero if AREA is used as a collection area.

Assuming that offset=0,

```
DISPLAY AREA(2)
```

displays an eight-byte data field that consists of the third element of the array.

```
DISPLAY AREA(1:3)
```

displays a 24-byte data field that consists of the second through fourth elements of the array.

```
DISPLAY AREA(0:63) or DISPLAY AREA(,512)
```

displays the entire array—a field of 512 bytes.

**Example 2:**

```
DEFINE A
```

creates a data field in DSS storage with the attributes: offset=0, length=4, type=X, size=4. The contents of the data field are zeroed. Subsequent commands (SET, DISPLAY, etc.) can refer to the data field by using the identifier A.

**Example 3:**

```
DEFINE B=X'00'
```

creates a data field, named B, with the attributes: offset=0, length=1, type=X, size=1. The contents are X'00'.

**Example 4:**

```
DEFINE C.(2,2,C,8)=D
```

creates a data field, named C, with the attributes: offset=2, length=2, type=C, size=8. The third and fourth bytes of data field C are initialized from the first two bytes of data field D.

## DISABLE

The DISABLE command causes AT and ON breakpoints to be ignored until ENABLE commands reactivate them.

---

Command Verb	Operands
{DISABLE} {DSBL}	{&AT &ON} [( {name number} [...])][...]

---

&AT

&ON

Indicates whether ATs or ONs are to be disabled.

name

number

Indicates the enabled ATs or ONs that are to be disabled. If no name or number is specified, all ATs or ONs are disabled.

Specified as: an identifier or decimal literal.

**Note 1:** If any of the names or numbers specified are not found, or are already disabled, a warning message is issued.

**Note 2:** If an error is encountered in executing any operand, a warning message is issued and processing continues with the next operand until all operands are processed, at which time you are prompted for a command to correct the error.

**Example:**

```
DISABLE &AT(ATONE,005), &ON(004)  
GO
```

disables the AT named ATONE, the AT numbered 005, and the ON numbered 004; and returns control to OS/VS without the DSS monitoring for the disabled ATs and ON.

## DISCONNECT

The DISCONNECT command stops DSS operation. DISCONNECT removes all ATs, ONs, patches, PROCEDURES, and DSS user symbols; releases all assigned I/O names; and returns control to OS/VS. Use this command when you have completed debugging and want to terminate the DSS session.

---

Command Verb	Operands
{DISCONNECT} {DSC}	none

---

## DISPLAY

The DISPLAY command writes the contents of a data field or the information provided by a DSS function at the console.

---

Command Verb	Operands
{DISPLAY} D }	data-field [...]

---

data-field

The data field whose contents are to be displayed.

Specified as: any representation of a data field, or &QT.

**Note 1:** DISPLAY is normally used for short data fields. For extensive data fields, use DUMP.

**Note 2:** If an error is encountered in executing any operand, a warning message is issued and processing continues with the next operand until all operands are processed, at which time you are prompted for a command to correct the error.

**Note 3:** If multiple operands are used with DISPLAY, the data specified by each is written beginning on a new line.

**Note 4:** When the operand is a DSS function or includes a DSS function, one or more lines are written at the console. The contents and formatting of these lines depend on the DSS function. When the operand does not include a DSS function, the referenced string of the data field is displayed at the console. For specific output formats, see "Section 5: Output Formats".

**Example 1:**

```
DISPLAY 'IDENTIFIER'
```

causes this to be printed at the console: IDENTIFIER

**Example 2:**

```
DISPLAY &CPUID(1) &G(0:15)
```

displays all general registers for CPU 1 at the console; in this case, more than one line of output is required.

**Example 3:**

```
DISPLAY &O(AREA)
```

displays the offset attribute of AREA.

## DIVERT

The DIVERT command changes the source of DSS command input.

Command Verb	Operands
{DIVERT} {DVT}	{CARD} {TIN}

### CARD

### TIN

The I/O name of a card or tape data set that contains command input. If neither is specified, the console is assumed.

**Note 1:** The commands in the command stream established by DIVERT are executed as if each one was specified as a separate physical record. When you divert from CARD or TIN, the CARD or TIN input buffer is not cleared out. Therefore, CARD or TIN can divert to another data set and be diverted back to without the loss of any commands. However, when you divert from the console, the console's input buffer is cleared. Therefore, if a DIVERT command is specified at the console, it should be the last command in the input record, since all following commands in the input record are ignored.

**Note 2:** If a command procedure issues a DIVERT, the next command in the procedure can be reached by a subsequent REVERT.

**Note 3:** When physical end of input is detected on a diverted-to data set, DSS executes an implicit REVERT. Physical end of input is an end-of-file condition on a card reader or the detection of a tape mark on a magnetic tape.

**Note 4:** If DIVERT is executed as the last command in a command procedure, an attempt to REVERT to the command procedure returns control to the console. In addition, an attempt to REVERT to a command procedure before one has been invoked returns control to the console.

**Note 5:** The I/O name must have been allocated by the ASSIGN command.

### Example 1:

```
ASSIGN CARD=X'00C'  
DIVERT CARD
```

This causes the cards in the card reader to be used as command input.

### Example 2:

The console input:

```
A:ASSIGN CARD=X'00C'  
B:DIVERT CARD  
B1:RELEASE CARD
```

The diverted-to card input:

```
C:ASSIGN TIN=X'183'  
D:DIVERT TIN  
E:RELEASE TIN
```

The diverted-to magnetic tape input:

```
F:SET A=1  
G:SET B=2  
H:DIVERT CARD  
I:SET C=3
```

Commands A and B are executed in sequence. Command B causes commands to be read from the card reader; commands C and D are then executed. Command D causes commands to be read from the tape input device; commands F, G, and H are executed. Command H causes commands to be read from the card reader; command E is executed. When end of file is detected on the card reader, an implied REVERT causes the command stream to switch back to the console. Command B1 and all subsequent commands are then typed and executed. (See the first programming note under DIVERT). Note that command I has not been executed; it can be executed by DIVERT TIN.

Example 3:

```
A:PROC Z  
B:ASSIGN CARD=X'00C'  
C:DIVERT CARD  
D:RELEASE CARD  
END  
F:INVOKE Z  
G:SET C=1
```

The diverted-to card input:

```
C1:SET A=2  
C2:ASSIGN TIN=X'183'  
C3:DIVERT TIN  
C4:DIVERT TIN  
C5:SET B=3
```

The diverted-to magnetic tape input:

```
T1:SET B=2  
T2:DIVERT CARD  
T3:REVERT
```

Command A is executed. Commands B, C, and D are stored as the text commands of procedure Z. Command F is then executed to invoke the procedure. Commands B and C of procedure Z are then executed. Command C causes commands to be read from the card reader. Commands C1, C2, and C3 are executed. Command C3 causes commands to be read from the tape input device. Commands T1 and T2 are executed. Command T2 causes commands to be read from the card reader. Command C4 is executed, which causes commands to be read from the tape input device. Command T3 is executed; this causes control to return to the current command procedure, procedure Z. Command D is executed. Command D is the last command in the command procedure, so an implicit RETURN is executed, and command G is executed next.



# DUMP

The DUMP command writes the specified data to the I/O name specified by the last setting of the &DOUT DSS function. For specific output formats, see "Section 5: Output Formats."

Command Verb	Operands
DUMP	{ data-field[,...] & PRDMP[RM] }

## data-field

A data field whose contents are to be dumped as low-speed, formatted output. Specified as: any representation of a data field, or &QT. Immediate attribute designation can be used in the data-field representation.

## &PRDMP[RM]

Specifies a high-speed unformatted dump that can be used as input to the AMDPRDMP service aid program while OS/VS2 is running.

## &PRDMP

A dump of all virtual storage, consisting of all common storage and the private area indicated by the &ASID in &QT.

## &PRDMPRM

A dump of all real storage, including the PSA for the &CPUID in &QT.

&PRDMP or &PRDMPRM is valid only when &DOUT is set to TOUT1 or TOUT2.

&PRDMP[RM] speeds up the dumping of large areas of OS/VS2 storage. DUMP commands with this operand dump unformatted information consisting of blocked records, whereas DUMP commands without this operand dump formatted information consisting of unblocked records. Both types of dumps can be on the same tape; AMDRPRMP can print the preformatted information.

Each DUMP command with &PRDMP[RM] produces a separate tape data set. (That is, the first thing the DUMP command does is write a tapemark.) The communication vector table, address space vector table, and address space control block (ASCB) are dumped; the ASCB is dumped for virtual storage dumps only. DSS types the file sequence number at the console.

**Note 1:** When DUMP is executed, the I/O name of the output device (PRINT1, PRINT2, TOUT1, TOUT2, LOG, CMDT, or PUNCH) must be in the data field controlled by the &DOUT DSS function. Initializing &DOUT entails:

```
ASSIGN i/o-name = device-address
SET &DOUT = 'i/o-name'
```

For example,

```
ASSIGN PRINT1 = X'00E'
SET &DOUT = 'PRINT1'
```

**Note 2:** Only command procedures that have been stored by PROCEDURE can be dumped to the card punch (PUNCH) or to the output tape (CMDT).

**Note 3:** If an error is encountered in executing any operand, a warning message is provided and processing continues with the next operand until all operands are processed, at which time you are prompted for a command to correct previously mentioned errors.

**Note 4:** Every dump should have a subheading (specified by &HDR) that indicates what area of storage is dumped; every dump is on an offline device (&DOUT cannot specify the console), and it may be difficult to trace a dump back to the corresponding DUMP command.

**Example 1:**

```
DUMP L'30000:3FFFF'
```

This command dumps OS/Virtual Storage from location 30000 through location 3FFFF on the output device specified in &DOUT.

**Example 2:**

```
DUMP &AT, &ON
```

This command causes all ATs and ONs, with their associated command procedures, to be printed on the output device specified in &DOUT.

**Example 3:**

```
ASSIGN CMDT=X'180'  
SET &DOUT='CMDT'  
DUMP &PROC (COMS)
```

dumps command procedure COMS, which must have been stored by PROCEDURE, to magnetic tape CMDT for later command-stream input.

## ENABLE

The ENABLE command activates ATs and ONs that were disabled by either the DISABLE command or the DISABLE suboperand of the AT and ON commands.

Command Verb	Operands
{ ENABLE } { ENBL }	{ &AT } [( { name } { &ON } [ ... ] ) [ ... ]

&AT  
&ON

Indicates whether ATs or ONs are to be enabled.

name  
number

Indicates the ATs and ONs that are to be enabled. If no name or number is specified, all ATs and ONs are enabled.

Specified as: an identifier or decimal literal.

**Note 1:** If any of the names or numbers specified are not found, are already enabled, or conflict with other already-enabled AT breakpoints or ON events, a warning message is issued for each.

**Note 2:** If an error is encountered in executing any operand, a warning message is provided and processing continues with the next operand until all operands are processed, at which time you are prompted for a command to correct the error.

**Example:**

```
ENABLE  &AT, &ON( 5 )
GO
```

enables all ATs and ON number 5, and returns control to OS/V.S. DSS monitoring specified by the newly enabled ATs and ON is added to the AT and ON monitoring that was already specified.

## END

The END command is used as the final text command for an AT, ON, IF, or PROCEDURE command. END is not an executable command; it is a delimiter of the text commands associated with a definitional command.

Command Verb	Operands
END	none

**Note:** If END is found with no matching AT, ON, IF, or PROCEDURE command, it is ignored.

**Example 1:**

```
ON  &LD(HENRY);DUMP  &Q;ENABLE  &AT(AT1,AT2,AT3,_
AT4,AT5,AT6);GO;END
```

**Example 2:**

```
AT  L'1000'
IF  FIELD=X'FF'
INVOKE  SETPROC
END
END
```

## EQUATE

The EQUATE command establishes a secondary definition for an existing data field.

Command Verb	Operands
{ EQUATE } { EQU }	{data-field <sub>2</sub> -name = data-field <sub>1</sub> }[,...]

**data-field<sub>2</sub>-name**

The name of the secondary definition.

**Specified as:** an identifier that is not already in use as the name of a data field; you can append an immediate designation of any attribute except size.

data-field<sub>1</sub>

The data field for which you want a secondary definition.

Specified as: any representation of a previously defined or equated data field in DSS storage, or a data field in a load module in OS/VS virtual storage.

**Note 1:** The attributes of the secondary data field are as follows, where 1 designates the primary data field and 2 designates the secondary data field:

base2 = base1 plus offset1

offset2 = 0, or as specified by immediate attribute designation

length2 = in OS/VS1, length1 minus offset2, or as specified by immediate attribute designation

in OS/VS2, the lesser of (length1) or (size2 minus offset2), or as specified by immediate attribute designation

type2 = type1, or as specified by immediate attribute designation

size2 = size1 minus offset1

If offset2 or length2 is specified by immediate attribute designation, the referenced string must reside entirely within the bounds established by base2 and size2, or the command is rejected and an error message is issued.

**Note 2:** When more than one pair of operands is specified, the result is the same as if multiple EQUATE commands with single pairs of operands were specified.

**Note 3:** Once a secondary definition is established, it exists until it is deleted by the REMOVE command, or the primary data field ceases to exist, or the DISCONNECT command is issued. In addition, if the secondary definition applies to a data field in an OS/VS load module, DSS will remove the secondary definition if the module is unloaded.

**Example 1:**

```
EQUATE MYNAME=IGC116
DISPLAY &SYM(MYNAME)
```

DSS replies:

```
&SYM NAME=MYNAME REF=EQUATED OFFSET=00000000 LNG=000001F0 TYPE=X
SIZE=000001F0 SCOPE=EXTERNAL QUAL=&SVM.IEANUC01.
```

**Example 2:**

```
DEFINE FLDA='ABCDEFGHIJKLMNPOQRSTUVWXYZ'
EQUATE FLDB=FLDA.(9,3)
DISPLAY &SYM(FLDB)
```

DSS replies:

```
&SYM NAME FLDB REF=EQUATED OFFSET=00000000 LNG=00000003 TYPE=C
SIZE=0000001A SCOPE=INTERNAL QUAL=
```

## GO

The GO command relinquishes control to OS/V5 for resumption of program execution, without disconnecting the DSS user.

Command Verb	Operands
GO	[RECOVER]

### RECOVER

When DSS is monitoring OS/V52 and you want to give control to RTM (recovery termination management), press RESTART, and when DSS prompts you for a command, enter GO RECOVER. This command simulates pressing RESTART when DSS does not own the RESTART resource: DSS gives control to RTM with exactly the same interface as when the first-level interrupt handler gives control to RTM, except that DSS is monitoring the system.

**Note 1:** OS/V5 is restored as it was when DSS received control, except for any changes (including implanted AT and ON breakpoints) that you made.

**Note 2:** If you want to resume execution at a point other than RTM or the point of interruption when DSS gained control, change the current PSW and any related information (general registers, etc.) by means of the SET command and appropriate DSS functions before issuing the GO command.

**Note 3:** To regain control, other than through activated ATs and ONs, you must press the RESTART key; this activates the primary DSS control level.

**Note 4:** GO can be issued from any control level. (Control levels are explained in Section 4.)

### Example:

```
SET &RANGE=&LM( IGC2903D )
SET &SOUT=' PRINT1 '
ON &B, &P( LP3ON1, SNAP )
END
```

(DSS lists the name and number of the ON.)

```
GO
```

Control returns to OS/V5 with DSS monitoring. The monitoring is controlled by the commands entered before the GO.

## GOTO

The GOTO command allows nonsequential execution of the commands in a command procedure.

Command Verb	Operands
GOTO	label

### label

The label specified at the beginning of a DSS command, excluding the colon (:).

**Note 1:** The current command procedure is searched for the specified command label, starting at the first command in that command procedure. If the command label is not found within the command procedure, DSS issues an error message.

**Note 2:** If a GOTO command is issued from the command stream and the command label is found in the current command procedure, an implied REVERT command is executed and control is transferred to the specified command label.

**Example 1:**

```
GOTO JOE
```

The command labeled JOE will be executed next.

**Example 2:**

```
A:GOTO D
B:SET M=2
C:GO
D:SET J=1
E:GOTO B
```

The commands will be executed in the order A,D,E,B,C.

**Example 3:**

```
IF A=1
  GOTO EQUAL
END
SET B=2
INVOKE Q
EQUAL:SET C=1
GO
```

If A equals 1, the normal sequential processing of DSS commands will be interrupted and processing will continue at the command labeled EQUAL.

**Example 4:**

```
C: DISPLAY 'HERE'
A: SET F=1
B: GOTO C
D: SET E=2
C: SET G=3; GO
```

When the command labeled B is executed, the GOTO command will result in the first C (the DISPLAY command) being executed next; see Note 1 under GOTO.

## IF

The IF command supplies conditional text commands; that is, if the expression of the IF command is evaluated as “true,” its text commands are executed. If the expression is false, the specified text commands are bypassed and processing continues with the next command after the END command.

Command Verb	Operands	Text Commands
IF	expression	text-commands END

expression

Any expression that can be evaluated as true or false or returns a one-byte data field set to X'FF' or X'00'. The expression may contain arithmetic, relational, or boolean operators or a combination of these.

**Specified as:** a character string containing data-field representations, and usually one or more operators.

text-commands

Any DSS commands.

END

Indicates the end of the text commands.

**Note 1:** If an IF command (and its text commands) is nested within the text commands of another IF, execution of the inner IF is conditional on the truth of the outer IF's expression.

**Note 2:** Any expression that can be evaluated as true or false or returns a one-byte data field set to X'FF' or X'00' is valid in an IF command. Any other expression causes an error message.

**Example 1:**

```
IF BYTE=X'F0';DUMP TABLE;END
```

compares the contents of BYTE to the literal X'F0'; if they are equal, the DUMP command is executed. If they are not equal, the DUMP command is ignored.

The commands could also have been entered on separate lines:

```
IF BYTE=X'F0'  
DUMP TABLE  
END
```

**Example 2:**

```
AT INTPROC  
COLLECT COLLAREA=&MPSW  
IF %O(COLLAREA)=0  
    DISPLAY COLLAREA.(0,512)  
END  
END
```

## INVOKE

The INVOKE command causes the execution of a command procedure that a PROCEDURE command stored during the current DSS session.

Command Verb	Operands
{ INVOKE } { INV }	procedure-name [,argument]...

procedure-name

The name of the command procedure to be executed.

**Specified as:** the name of a command procedure that a PROCEDURE command stored during the current DSS session.

argument

The value to be assigned to the corresponding parameter variable of the procedure.

**Specified as:** A character literal or a character string (the same as a character literal, but not enclosed in apostrophes; consecutive blanks will be edited to single blanks).

The parameter variables of the procedure are established as **read-only** variables, with values equal to the arguments. The arguments in the INVOKE command must be specified in the same order as the parameter names in the associated PROCEDURE command.

**Note:** If the specified procedure cannot be located, DSS issues an error message.

**Example:**

```
PROCEDURE Z,W,A,P1
SET  εS(A)=εS(A)+1
IF W = 'NOGO'
    GOTO S1
    END
    SET B=εS(P1)
S1: RETURN
END
DEFINE L=0
INVOKE Z,GO,L,(M+7)*4
```

definition of command procedure Z

definition of the identifier  
to be passed as an argument

invocation of command procedure Z

The text commands following the PROCEDURE command are stored as a command procedure. The parameter variables of the command procedure are defined as if this command were executed at the beginning of the procedure:

```
DEFINE W = 'GO', A = 'L', P1 = '(M+7)*4'
```

These variables are automatically removed on return from the command procedure, as if this command were executed:

```
REMOVE εSYM(W,A,P1)
```

## ON

The ON command specifies the events for which OS/VS execution is to be monitored, and the actions to be taken on their occurrence.

Command Verb	Operands	Text Commands
ON	dss-function[,...][,&P([name] [.SNAP][.DISABLE])]	[command-procedure] END

**dss-function**

A program event or a system event.



### Program Events

&B        Successful branch  
&I        Execution of instruction  
&SA       Storage alteration  
&RA       General-purpose-register alteration

### System Events

&LD(module-name[,...])        The loading of the named modules  
&UNLD(module-name[,...])       The unloading of the named modules

### name

If specified, the name of the ON for use in the DISABLE, DISPLAY, DUMP, ENABLE, and REMOVE commands. (In addition, a unique number is generated for the ON by DSS. This number, printed at the console when the ON command is processed, can be used instead of a name to refer to the ON.)

### SNAP

If specified, each time one of the events occurs a predefined set of SNAP information will be recorded prior to the execution (if any) of the ON command procedure. If &SOUT is set to an I/O name that is assigned a device address, the SNAP information will be printed.

### DISABLE

If specified, indicates that the ON is to be inactive until activated by an ENABLE command.

### command-procedure

Optional only if SNAP is specified. If SNAP is not specified, at least one DSS command must be specified before the END command.

### END

Delimits the command procedure; must be specified as the last or only text command.

**Note 1:** The program events are monitored during the execution of the object program in the address range specified by the "SET &RANGE= address<sub>1</sub>;  
address<sub>2</sub>," command. The system events are monitored for the modules listed with &LD or &UNLD.

**Note 2:** The &RANGE DSS function applies to all program events. Only those areas where DSS has monitoring access may be specified by &RANGE; for a list of those areas, see "Appendix A: Restrictions."

**Note 3:** Registers to be monitored by &RA are indicated by setting the &PRM DSS function. Example: SET &PRM=&M(1,5:9). You can change &PRM at any time without disturbing the command procedure or the enable/disable status of the ON command.

**Note 4:** DSS checks the syntax of the text commands as you enter them. However, nested command procedures are not checked for syntax until the command procedures that contain them are executed.

**Note 5:** In OS/VS1, the qualification in the qualify table (&QT) when the command procedure is stored is the qualification that is used when the command procedure is executed. In OS/VS2, the command procedure runs under the current &QT qualification.

**Note 6:** DSS does not accept an enabled ON for a program event specified in an already-enabled ON.

**Note 7:** The number of ATs and ONs that can be enabled at the same time is limited by the amount of space that is free for storage of their control blocks. On the average, this is about 64 ATs and ONs in OS/VS1, and about 253 ATs and 92 ONs in OS/VS2.

**Example 1:**

```
SET &RANGE=&ASID(2),&SVM.IGC2903D.(0,100)
ON &B, &P(BRANCH,SNAP);END
```

Sets the program event range to the address of IGC2903D through the address IGC2903D+99, in the second address space. DSS will monitor for branches within that range. Each time a successful branch is executed within that range, SNAP information will be saved and program execution will continue.

**Example 2:**

```
ON &LD(PAYROLL), &P(LOAD,DISABLE)
  AT &LM(LMB).(X'64'), &P(LMAT,SNAP);DIVERT
  END
END
```

The AT command is stored along with related information for this ON command, but the event is not activated (and therefore not monitored) because DISABLE is specified. A subsequent ENABLE &ON(LOAD) is required to activate this event. Upon the enabling, the ON event will be monitored. When PAYROLL is loaded by OS/VS, the AT will be implanted at the logical breakpoint PAYROLL plus the offset of X'64'. When that breakpoint is executed, the SNAP information will be recorded and the DIVERT command will give control to the console. Note that the first END command delimits the AT command and the second END command delimits the ON command.

**PATCH**

The PATCH command changes the contents of a data field and saves the original contents.

Command Verb	Operands
{ PATCH } PAT	{ data-field <sub>1</sub> = data-field <sub>2</sub> } [...][,&P(name)]

**data-field<sub>1</sub>**

The data field where data is to be inserted. The base and offset attributes of this data field determine the first byte of OS/VS storage to be changed. Specified as: any representation of a data field that is in an OS/VS load module and in an area to which DSS has write access; for a list of those areas, see "Appendix A: Restrictions."

data-field<sub>2</sub>

The data field from which data is to be moved. The length attribute of this data field determines the number of bytes to be changed (an important difference between PATCH and SET) and cannot be greater than the smaller of 2048 or 4096 bytes or size<sub>1</sub> minus offset<sub>1</sub>.

Specified as: any representation of a data field.

name

If specified, the name applied to the patch for use in the DISPLAY, DUMP, and REMOVE commands. If more than one patch is given, the name applies to the entire group of patches. (In addition, a unique number is printed at the console when the PATCH command is processed; this number can be used instead of a name to refer to the patch.)

**Note 1:** The original contents of data-field<sub>1</sub> can be referred to by using the

&PATCH{ name  
          } number }

DSS function with the DISPLAY, DUMP, and REMOVE commands.

**Note 2:** Since data-field<sub>1</sub> must be in a load module, it cannot be in a storage area acquired by GETMAIN. GETMAIN areas can only be modified by SET.

**Note 3:** You cannot patch over an enabled AT breakpoint.

**Example:**

```
PATCH IEAQFX00.(X'1D0',2)=X'4770', &P(GOOD)
```

Replaces the two bytes at offset X'1D0' from the beginning of CSECT IEAQFX00 with X'4770' and saves the two bytes that were at that location. The name of this patch is GOOD. The original contents can be restored by REMOVE &PATCH(GOOD).

## PROCEDURE

The PROCEDURE command defines a command procedure that can be activated by the INVOKE command.

Command Verb	Operands	Text Commands
{ PROCEDURE } { PROC }	procedure-name[,parameter]...	command-procedure END

procedure-name

The name assigned to the command procedure.

Specified as: an identifier.

parameter

The name assigned to a parameter variable that will be created when the command procedure is invoked.

Specified as: an identifier that is not qualified by immediate attribute designation.

command-procedure

A series of DSS commands.

END

Delimits the command procedure; must be specified as the last text command.

**Note 1:** The syntax of the text commands will not be checked until the command procedure is invoked.

**Note 2:** The parameter variables will not be defined until the command procedure is invoked. They will be removed on return from the command procedure. See the first example under INVOKE.

**Note 3:** A parameter variable cannot be defined by DEFINE, since it is implicitly defined when you invoke the command procedure. Thus,

```
PROCEDURE PDEF, A
DEFINE A = 5
```

is an error.

**Note 4:** A parameter variable cannot be altered by SET, since it is read-only.

**Note 5:** Using &S, you can refer to the contents of the argument (in INVOKE) that corresponds to the parameter variable. Thus, after

```
PROCEDURE PDEF, A
SET &S(A)=&S(A)+1
END
DEFINE ABLE=0
INVOKE PDEF, ABLE
```

ABLE is incremented by 1.

**Note 6:** Every command procedure ends with an implicit RETURN command. Therefore, if the text is exhausted, the command procedure will terminate execution normally.

**Note 7:** To make any change to a command procedure, you must remove and reenter the entire procedure.

**Example:**

```
PROCEDURE STOP, LOC, WHERE
DEFINE PARM=WHERE
AT &S(LOC)
DISPLAY '&S(PARM) '
DIVERT
END
END
```

These commands define a command procedure with two parameter variables, LOC and WHERE, that are used as substitution variables in the text commands. The command DEFINE PARM=WHERE allows the AT-command procedure to display the value of WHERE. Although WHERE, a parameter variable, will be removed at the end of procedure STOP's execution, PARM will exist when the AT-command procedure is executed.

The text commands in this example include two END commands. The first END delimits the command procedure stored by AT; the second END delimits the command procedure stored by PROCEDURE. The latter command procedure could be invoked by:

```
INVOKE STOP,L'2C0','CSECT A ENTERED'
```

This causes STOP to be executed with parameter variable LOC set to L'2C0' and parameter variable WHERE set to 'CSECT A ENTERED'.

## RELEASE

The RELEASE command closes a data set and returns the associated I/O device to OS/VS.

Command Verb	Operands	
{ RELEASE REL }	{ CARD TIN CMDT PUNCH PRINT1 PRINT2 TOUT1 TOUT2 LOG }	[...]

CARD  
TIN  
CMDT  
PUNCH  
PRINT1  
PRINT2  
TOUT1  
TOUT2  
LOG

The I/O name of the device to be released.

**Note 1:** All data sets to be released must have been previously allocated by the ASSIGN command.

**Note 2:** For TOUT1, TOUT2, or CMDT, RELEASE writes a tape mark and does a rewind and unload. For TIN, RELEASE does a rewind and unload.

**Note 3:** If the I/O name being released was for dump output, &DOUT is set to blanks to indicate no device assignment.

**Note 4:** If the I/O name being released was for SNAP output, the SNAP buffer is dumped prior to closing the data set and &SOUT is set to blanks to indicate no device assignment.

**Note 5:** If the I/O name being released is used by the current command stream, a warning message is issued and the I/O name is not released.

**Note 6:** If an error is encountered in executing any operand, a warning message is issued and processing continues with the next operand until all operands are processed, at which time you are prompted for a command to correct the error.

**Example:**

```
RELEASE PRINT1, CARD
```

closes the PRINT1 and CARD data sets and releases the associated printer and card reader.

## REMOVE

The REMOVE command deletes ATs and ONs, along with their associated command procedures, as well as patches, command procedures stored by PROCEDURE, and DSS-user-defined data fields.

Command Verb	Operands
{ REMOVE } { REM }	{ { &AT } { &ON } { &PATCH } &PROC [(procedure-name[,...])] } &SYM [(dss-data-field[,...])] } [( { name } { number } [...])] } [...]

&AT

&ON

&PATCH

Indicates whether ATs, patches, or ONs are to be removed.

name

number

Indicates the ATs, patches, or ONs that are to be removed. If no name or number is specified, all ATs, patches, or ONs are removed.

Specified as: an identifier or decimal literal.

&PROC

Indicates that a command procedure stored by PROCEDURE is to be removed.

procedure-name

Names a PROCEDURE to be removed. If not specified, and &PROC is specified, all such command procedures are removed.

&SYM

Indicates that a data field is to be removed.

dss-data-field

Names a data field that was created by DEFINE or EQUATE and that is to be removed. If not specified, and &SYM is specified, all data fields that were created by DEFINE or EQUATE are removed.

**Note 1:** REMOVE is implied for ATs, ONs, PROCEDURES, patches, and user-defined data fields when DISCONNECT is issued.

**Note 2:** REMOVE is implied for an AT when the module that contains the breakpoint is unloaded. This removal is partial if the AT was specified for locations in more than one load module; the AT remains effective for the modules that remain loaded.

**Note 3:** REMOVE is implied for all symbols that are equated to data fields in a module being unloaded or equated to a data field being removed via REMOVE &SYM.

**Note 4:** You can remove a symbol equated to an OS/VS data field with no effect on the aliases equated to that symbol.

**Note 5:** If an error is encountered in executing any operand, a warning message is issued and processing continues with the next operand until all operands are processed, at which time you are prompted for a command to correct the error.

**Examples:**

```
REMOVE &ON, &AT(AT1), &SYM(FLDA, FLDB)
```

removes all of your ONs, the AT named AT1; and the symbols FLDA and FLDB.

```
REMOVE &AT(1)
```

removes AT number 1.

## RETURN

The RETURN command terminates the execution of a command procedure and returns control to the point from which the command procedure was invoked. Execution continues with the command following the INVOKE that invoked the terminated command procedure. If any parameter variables are used by the terminated command procedure, they are removed.

Command Verb	Operands
{ RETURN } RET	[ALL]

### ALL

Terminates all active command procedures and returns control to the console.

**Note 1:** If the INVOKE command was issued from a command stream, execution continues with the next sequential command in the stream. (This may not be the command following the INVOKE, since the invoked procedure could have diverted back to part of the command stream.)

**Note 2:** If command execution reaches the end of the command procedure before it reaches a RETURN command, an implicit RETURN is executed.

**Note 3:** Execution of a RETURN command in a command stream at the primary control level causes control to be returned to the console. (For an explanation of control levels, refer to "Control Levels," in Section 4.)

**Example 1:**

```
PROCEDURE B
  .
  .
  RETURN
END
```

The execution of procedure B is terminated and control returns to the point of invocation.

**Example 2:**

```
PROCEDURE C
.
.
RETURN ALL
END
```

The execution of procedure C and all other active procedures is terminated and control returns to the console.

## REVERT

The REVERT command returns control from the command stream to the current command procedure. Execution of the command procedure resumes with the command following the last-executed DIVERT command.

Command Verb	Operands
{REVERT} {RVT}	none

**Note 1:** If no DIVERT has been issued by the current command procedure, the REVERT command is ignored.

**Note 2:** Execution of a REVERT command in the primary control level returns control to the console. (For an explanation of control levels, refer to "Control Levels," in Section 4.)

**Example:**

```
A: PROCEDURE Z
B: ASSIGN CARD = 12
C: DIVERT CARD
END
E: INVOKE Z
F: RELEASE CARD
```

The diverted to card input:

```
C1: SET C = 1
C2: REVERT
C3: SET D=2
```

Command A is executed. It causes commands B and C to be stored as command procedure Z. Command E is executed, which causes command procedure Z to be executed. Commands B and C are now executed. Command C causes commands to be read from the card reader. Commands C1 and C2 are executed. The end of the command procedure is reached, and an implied RETURN is executed. Command F is then executed.

Command C3 has not been executed and cannot be executed by DIVERT CARD, since the card reader has been released.



## SET

The SET command is used to insert data into a data field. No record of the operation is kept by DSS.

Command Verb	Operands
SET	{ data-field <sub>1</sub> = data-field <sub>2</sub> } [,...]

### data-field<sub>1</sub>

The location where data is to be inserted. The length attribute of this data field determines the number of bytes to be changed. **This is an important difference between SET and PATCH;** in the PATCH command, the length of data field<sub>2</sub> determines the number of bytes to be changed.

Specified as: any representation of a data field to which DSS has write access (or &QT), and which is no more than 2048 or 4096 bytes long. For a list of areas to which DSS has write access, see "Appendix A: Restrictions."

### data-field<sub>2</sub>

The data field from which data is to be moved. If its length differs from that of datafield<sub>1</sub>, this data field is padded or truncated, according to data type, as shown in Figure 12.

Specified as: any representation of a data field.

**Note 1:** The REMOVE command removes only PATCH alterations, not SET alterations.

**Note 2:** When you alter OS/VS storage with the SET command, keep a record of the change by displaying the storage area before and after.

**Note 3:** SET { &O  
&L  
&T  
&SZ } (any-dss-function) is invalid.

Type Attribute		Length Attribute		
Data Field 1	Data Field 2	Data Field 2 Less	Data Field 2 Equal	Data Field 2 Greater
C C X	C X C	Data field 2 is padded on right with blanks.	Data field 2 is moved as is.	Data field 2 is truncated on right. Warning issued if any nonblanks are truncated.
C I	I C	Invalid operation raises error condition.	Invalid operation raises error condition.	Invalid operation raises error condition.
I	I	Data field 2 padded on left by propagating sign bit.	Data field 2 moved as is.	Data field 2 truncated on left. Warning issued if significant bits are lost.
X	X	Data field 2 padded on left with zeros.	Data field 2 moved as is.	Data field 2 truncated on left. Warning issued if significant bits are lost.
I	X	Data field 2 padded on left with zeros.	Data field 2 moved as is. Warning issued if result after move is negative.	Data field 2 truncated on left. Warning issued if significant bits are lost or result after move is negative.
X	I	Data field 2 padded on left by propagating sign bit. Warning issued if result is negative.	Data field 2 moved as is. Warning issued if result after move is negative.	Data field 2 truncated on left. Warning issued if significant bits are lost or result is negative.

Figure 12. Truncation and Padding by SET and DEFINE

**Example 1:**

```
SET &O(AREA)=0
```

Changes the offset value of the data field AREA to zero.

**Example 2:**

```
SET &PSW=X'0000000000000000' or SET &PSW=0
```

Sets the current resume PSW for the CPU identified in &QT to zero.

**Example 3:**

```
SET L'29000'=X'00'
```

Sets four bytes at location 29000 to zero.

**Example 4:**

```
SET FLAG=FLAG|X'40';
```

Sets the data field FLAG to the contents of the immediate data field that results from evaluating the expression FLAG|X'40':

**Example 5:**

```
SET &SOUT='PRINT1'
```

The device assigned to the PRINT1 I/O name is now the SNAP output device.

**Example 6:**

```
IF FLDA=6  
SET CTR=CTR+1  
END
```

Increments CTR if FLDA equals 6.



## Section 3: Command Summary

This section consists of five figures:

Figure 13. DSS Commands

The commands, their formats, and their uses.

Figure 14. Command Operands

How to type the operands that Figure 13 shows in lowercase letters.

Figure 15. DSS Functions

The DSS functions, their formats, and their meanings.

Figure 16. Using DSS Functions

The DSS functions and the commands that they can be used in.

Figure 17. I/O Names and Their Uses

The I/O names, the commands they can be used in, and their purposes.

Command Verb	Operands and Text Commands	Function
*	[comment]	Puts a comment into a command procedure or onto the console sheet.
{ASSIGN} {ASGN}	$\left( \begin{array}{l} \text{CARD} \\ \text{TIN} \\ \text{CMDT} \\ \text{PUNCH} \\ \text{PRINT1} \\ \text{PRINT2} \\ \text{TOUT1} \\ \text{TOUT2} \\ \text{LOG} \end{array} \right) = \text{device-address} \{, \dots\}$	Assigns an I/O name for use by DSS; allocates an I/O device to DSS.
AT	location[ , ... ] [&P({name} [,SNAP] [,DISABLE])] [command-procedure] END	Establishes AT breakpoints at which a SNAP dump and/or a command procedure will be executed.
{COLLECT} COL	{ dss-data-field = data-field <sub>2</sub> } [ , ... ]	Copies data from source data fields into successive elements of a receiver data field.
{DEFINE} DEF	{ data-field <sub>1</sub> -name [ = data-field <sub>2</sub> ] } [ , ... ]	Defines a data field in DSS work space and, optionally, initializes it.
{DISABLE} DSBL	{ {&AT} [ ( {name} [ , ... ] ) ] } { {&ON} [ ( {number} [ , ... ] ) ] } [ , ... ]	Disables ATs and ONs until ENABLE commands reactivate them.
{DISCONNECT} DSC		Terminates DSS and returns control to OS/VS.
{DISPLAY} D	data-field [ , ... ]	Writes data at the integrated operator's console.
{DIVERT} DVT	- [ {CARD} ]	Changes the source of DSS command input. The source may be the integrated operator's console, a card reader, or a tape drive.
DUMP	{ data-field [ , ... ] } {&PRDMP[RM]}	Dumps the specified data.
{ENABLE} ENBL	{ {&AT} [ ( {name} [ , ... ] ) ] } { {&ON} [ ( {number} [ , ... ] ) ] } [ , ... ]	Activates ATs and ONs.
END		Ends text commands for the AT, ON, IF, and PROCEDURE commands.

Figure 13. DSS Commands (Part 1 of 3)

Command Verb	Operands and Text Commands	Function										
{ EQUATE } EQU }	{ data-field <sub>2</sub> -name = data-field <sub>1</sub> } [ , ... ]	Applies an alias to a data field defined by the DSS user or to a data field in OS/VS storage.										
GO	[RECOVER]	Returns control to OS/VS with DSS monitoring.										
GOTO	command-label	Allows nonsequential execution of a command procedure.										
IF	expression text-commands END	Designates text commands whose execution depends on whether the expression is evaluated as true or false.										
{ INVOKE } INV }	procedure-name [,argument] ...	Invokes a command procedure stored by a PROCEDURE command.										
ON	dss-function [ , ... ] [ ,&P([name] [,SNAP] [,DISABLE])] [command-procedure] END	Specifies ON events on which a SNAP dump and/or a command procedure will be executed.										
{ PATCH } PAT }	{ data-field <sub>1</sub> = data-field <sub>2</sub> } [ , ... ] [ ,&P(name) ]	Alters the contents of a specified data field and keeps a record of the original data.										
{ PROCEDURE } PROC }	procedure-name [,parameter] ... command-procedure END	Stores a command procedure to be invoked later by an INVOKE command.										
{ RELEASE } REL }	<table style="display: inline-table; vertical-align: middle;"> <tr><td rowspan="9" style="font-size: 4em; vertical-align: middle;">}</td><td>CARD</td></tr> <tr><td>TIN</td></tr> <tr><td>CMDT</td></tr> <tr><td>PUNCH</td></tr> <tr><td>PRINT1 [ , ... ]</td></tr> <tr><td>PRINT2</td></tr> <tr><td>TOUT1</td></tr> <tr><td>TOUT2</td></tr> <tr><td>LOG</td></tr> </table>	}	CARD	TIN	CMDT	PUNCH	PRINT1 [ , ... ]	PRINT2	TOUT1	TOUT2	LOG	Closes a data set and returns the associated I/O device to OS/VS.
}	CARD											
	TIN											
	CMDT											
	PUNCH											
	PRINT1 [ , ... ]											
	PRINT2											
	TOUT1											
	TOUT2											
	LOG											
{ REMOVE } REM }	<table style="display: inline-table; vertical-align: middle;"> <tr><td rowspan="5" style="font-size: 4em; vertical-align: middle;">}</td><td>{ &amp;AT</td></tr> <tr><td>&amp;ON</td></tr> <tr><td>&amp;PATCH</td></tr> <tr><td>{ { name</td></tr> <tr><td>{ number } [ , ... ] }</td></tr> <tr><td rowspan="3" style="font-size: 4em; vertical-align: middle;">}</td><td>&amp;PROC [(procedure-name [ , ... ] )]</td></tr> <tr><td>&amp;SYM [(dss-data-field [ , ... ] )]</td></tr> <tr><td>[ , ... ]</td></tr> </table>	}	{ &AT	&ON	&PATCH	{ { name	{ number } [ , ... ] }	}	&PROC [(procedure-name [ , ... ] )]	&SYM [(dss-data-field [ , ... ] )]	[ , ... ]	Deletes: ATs, ONs, and associated command procedures; command procedures stored by PROCEDURE; patches; and DSS-user-defined data fields.
}	{ &AT											
	&ON											
	&PATCH											
	{ { name											
	{ number } [ , ... ] }											
}	&PROC [(procedure-name [ , ... ] )]											
	&SYM [(dss-data-field [ , ... ] )]											
	[ , ... ]											

Figure 13. DSS Commands (Part 2 of 3)

Command Verb	Operands	Function
{RETURN} RET }	[ALL]	Terminates the execution of a command procedure and returns control to the point from which the command procedure was invoked.
{REVERT} RVT }		Returns control from a command stream to the current command procedure.
SET	{ data-field <sub>1</sub> = data-field <sub>2</sub> } [ , ... ]	Alters the contents of a data field without saving the original data.

Figure 13. DSS Commands (Part 3 of 3)



Operand	Specified As	Format	Example
argument	character string or character literal	C <sup>1</sup>	ABCD
command-procedure	any DSS commands	Commands separated from each other and from definitional command by semicolons or by entry on separate lines	IF A=B DISPLAY & PSW,&G(0:9) GO; END
comment	character string	C <sup>1</sup>	THIS COMMENT
data-field	address literal	L'x' <sup>5</sup> [(o,l,t,sz)] <sup>4</sup>	L'3C000'
	character literal	'c' <sup>1</sup>	'TOUT1'
	decimal literal	n <sup>2</sup>	1234
	hexadecimal literal	X'x' <sup>5</sup>	X'89AB'
	name used previously as a data-field-name operand or parameter operand <sup>9</sup>	identifier <sup>6</sup> [(o,l,t,sz)] <sup>4</sup>	AREA5
	a DSS function that can be used in the operand's position; see Figure 16	See Figure 15	&PSW
	map expression <sup>7</sup>	See Figure 15	&SVM.MODULE1
indirect address	$\left\{ \begin{array}{l} \left\{ \begin{array}{l} \text{address-literal} \\ \text{dss-function } 8 \end{array} \right\} [(o,l,t,sz)]^4 \\ \text{data-field-name } [(n[:m])]^3 \end{array} \right\} \{ \% [(o,l,t,sz)]^4 \} \dots$	L'1234'%%	
indicator 1 : indicator 2	$\left\{ \begin{array}{l} \left\{ \begin{array}{l} \text{address-literal} \\ \text{dss-function } 8 \end{array} \right\} [(o,l,t,sz)]^4 \\ \text{indirect-address} \\ \text{data-field-name} \end{array} \right\} : \left\{ \begin{array}{l} \left\{ \begin{array}{l} \text{address-literal} \\ \text{dss-function } 8 \end{array} \right\} [(o,l,t,sz)]^4 \\ \text{indirect-address} \\ \text{data-field-name} \end{array} \right\}$	A:A(5)	

Figure 14. Command Operands (Part 1 of 3)

Operand	Specified As	Format	Example
data-field-name	name of data field being defined (must be a unique name)	identifier <sup>6</sup> [(n[:m])] <sup>3</sup> [(o,l,t,sz)] <sup>4</sup>	AREA5
device-address	decimal device address	n <sup>2</sup>	14
	hexadecimal device address	X'x' <sup>5</sup>	X'00E'
dss-data-field	name used previously as data-field-name operand or parameter operand <sup>9</sup>	identifier <sup>6</sup> [(n[:m])] <sup>3</sup> [(o,l,t,sz)] <sup>4</sup>	AREA5
dss-function	&B, &I, &SA, &RA, &LD, or &UNLD	$\left\{ \begin{array}{l} \&B \\ \&I \\ \&SA \\ \&RA \\ \&LD \text{ (module-name [ , . . . ] )} \\ \&UNLD \text{ (module-name [ , . . . ] )} \end{array} \right\}$	&LD(IGC07, IGC08)
expression	a character string, containing data-field representations and usually one or more operators, that can be evaluated as true or false or represents a one-byte data field set to X'FF' or X'00'	data-field { [[b] operator[b] data-field] } ...	A < B+9
label	name used as a label in the current command procedure	identifier <sup>6</sup>	A6
location	address literal	L'x' <sup>5</sup> [(o,l,t,sz)] <sup>4</sup>	L'1234'
	name specified by EQUATE as the name of a location in an OS/VS load module (not valid if qualified by &ASID)	identifier <sup>6</sup> [(n[:m])] <sup>3</sup> [(o,l,t,sz)] <sup>4</sup>	TESTLOC
	DSS function <sup>8</sup> indirect address	See Figure 15. $\left\{ \begin{array}{l} \{ \text{address-literal} \} [(o,l,t,sz)] 4 \\ \{ \text{dss-function } 8 \} [(o,l,t,sz)] 4 \\ \{ \text{data-field-name [(n[:m])] } 3 \} \{ \%[(o,l,t,sz)] 4 \} \dots \end{array} \right\}$	&Q L'1234'%%

Figure 14. Command Operands (Part 2 of 3)

Operand	Specified As	Format	Example
name	name of the AT breakpoint, ON breakpoint, or patch	identifier <sup>6</sup>	TEST
number	number of the AT breakpoint, ON breakpoint, or patch	n <sup>2</sup>	18
parameter	name of the parameter	identifier <sup>6</sup>	X
procedure-name	name of the command procedure	identifier <sup>6</sup>	C
text-commands	any DSS commands	See Figure 13	D A,B; INV C,X

Notes:

- 1 Where c is any character string.
- 2 Where n is a decimal integer.
- 3 A subscript (n) or subscript range (n:m), where n and m are decimal integers.
- 4 Immediate attribute designation, where
 

o = offset l = length sz = size t = type	}	in bytes, specified as a decimal literal, hexadecimal literal, symbol, or expression
		expressed as l, X, or C, where
		l = decimal X = hexadecimal C = character
- 5 Where x is a hexadecimal integer.
- 6 A character string that starts with an alphabetic or alphabet-extender character optionally followed by as many as seven alphanumeric characters.
- 7 The first operand of PATCH must be a map expression.
- 8 See Figures 15 and 16.
- 9
  - Except that REMOVE cannot remove parameters.
  - If a parameter, the command procedure that defines it must be active.

Figure 14. Command Operands (Part 3 of 3)

Type of DSS Function	Format	Meaning
Machine Register	&G {register range} &C {register range} &F {register range} &PREFIX	general-purpose registers control registers floating-point registers prefix register
Program Status	&EPSW &EPSWN &IPSW &IPSWN &PPSW &PPSWN &SPSW &SPSWN &MPSW &MPSWN &PSW &RPSW &RPSWN	external PSW (old) external PSW (new) I/O PSW (old) I/O PSW (new) program PSW (old) program PSW (new) SVC PSW (old) SVC PSW (new) machine check PSW (old) machine check PSW (new) current PSW restart PSW (old) restart PSW (new)
Machine Status	&AC &CAW &CID &CSW &IOEL &MC &PRM &RANGE &TEA &TOD	activation code channel address word channel identification channel status word I/O extended logout address monitor call register alteration mask program-event monitoring range translation exception address time-of-day clock
Command-Related	&AT[{name }{,...}] {number} &ON[{name }{,...}] {number} &PATCH[{name }{,...}] {number} &PROC[{procedure-name }{,...}] &SYM[dss-data-field {,...}] &P {name {name }[,SNAP] [,DISABLE]} &PRDMP[RM] &S[dss-data-field {,...}] &DOUT &SOUT &HDR &M {register }{,...} {range}	ATs ONs patches PROCEDURE-command procedures user-defined data fields operand for AT, ON, or PATCH operand for DUMP (&PRDMP or &PRDMPRM) character-string substitution DUMP output destination SNAP output destination DUMP subheading mask function

Figure 15. DSS Functions (Part 1 of 2)

Type of DSS Function	Format	Meaning
Attribute	&O(symbol) &L(symbol) &T(symbol) &SZ(symbol)	offset length type size
Location Qualifier	&ADDR[&ASID(number).] map-expression &ASID({ number, location } { job-name } )  &CPUID[{ number, { dss-function } { address-literal } } ] &ID(simple-address-literal) &RM.address-literal &SVM.module-name[ { csect-name { entry-point-name } } ]  &JOB( { job-name { simple-address-literal } }, module-name )  &TCB(simple-address-literal).module-name [ { csect-name { entry-point-name } } ] &PRB(simple-address-literal).module-name [ { csect-name { entry-point-name } } ] &LM(module-name) &Q[ { csect-name { entry-point-name } } ] &QT	hexadecimal address  address-space id  CPU id symbolic address real address literal SVM map-id; indicates that the module is in supervisor virtual storage  JOB map-id; indicates that the module is in the named job or addressed job-step task control block  TCB map-id; indicates that the module is associated with the addressed task control block  PRB (program request block) map-id; used in place of TCB when there are duplicate copies  load module qualify function qualify-table function
Map	&SVM MAP( { NUC { LPA { ALPA { RRMA } } } )  &JOBMAP( { job-name { simple-address-literal } } ) &TCBMAP(simple-address-literal)	SVM map  JOB map  TCB map
Event	&B &I &SA &RA &LD(module-name[ , . . . ] ) &UNLD(module-name[ , . . . ] )	branch instruction fetch and execute storage alteration register alteration load a module unload a module

Figure 15. DSS Functions (Part 2 of 2)

Note: Complete command formats are not shown.  
 ~~~ indicates where additional operands  
 are required.

|                                                            | &AC | &ADDR | &ASID | &AT | &B | &C | &CAW | &CID | &CPUID | &CSW | &DOUT | &EPSW | &EPSWN | &F | &G | &HDR | &I | &ID | &IOEL | &IPSW | &IPSWN | &JOB | &JOBMAP | &L | &LD | &LM | &M | &MC | &MPSW | &MPSWN | &O | &ON |   |
|------------------------------------------------------------|-----|-------|-------|-----|----|----|------|------|--------|------|-------|-------|--------|----|----|------|----|-----|-------|-------|--------|------|---------|----|-----|-----|----|-----|-------|--------|----|-----|---|
| AT dss-function; ~~~                                       |     |       | X     |     |    |    |      |      |        |      |       |       |        |    |    |      |    |     |       |       |        | X    |         |    | X   |     |    |     |       |        |    |     |   |
| AT ~~~ [,dss-function([name] [,SNAP] [,DISABLE])];<br>~~~~ |     |       |       |     |    |    |      |      |        |      |       |       |        |    |    |      |    |     |       |       |        |      |         |    |     |     |    |     |       |        |    |     |   |
| COLLECT data-field-name = dss-function                     | X   | X     | X     |     |    | X  | X    | X    | X      | X    | X     | X     | X      | X  | X  | X    |    | X   | X     | X     | X      | X    | X       | X  | X   | X   | X  | X   | X     | X      | X  | X   |   |
| DEFINE dss-data-field = dss-function                       | X   | X     | X     |     |    | X  | X    | X    | X      | X    | X     | X     | X      | X  | X  | X    |    | X   | X     | X     | X      | X    | X       | X  | X   | X   | X  | X   | X     | X      | X  | X   |   |
| DISABLE dss-function                                       |     |       |       | X   |    |    |      |      |        |      |       |       |        |    |    |      |    |     |       |       |        |      |         |    |     |     |    |     |       |        |    | X   |   |
| DISPLAY dss-function                                       | X   | X     | X     | X   |    | X  | X    | X    | X      | X    | X     | X     | X      | X  | X  | X    |    | X   | X     | X     | X      | X    | X       | X  | X   | X   | X  | X   | X     | X      | X  | X   | X |
| DUMP dss-function                                          | X   | X     | X     | X   |    | X  | X    | X    | X      | X    | X     | X     | X      | X  | X  | X    |    | X   | X     | X     | X      | X    | X       | X  | X   | X   | X  | X   | X     | X      | X  | X   | X |
| ENABLE dss-function                                        |     |       |       | X   |    |    |      |      |        |      |       |       |        |    |    |      |    |     |       |       |        |      |         |    |     |     |    |     |       |        |    | X   |   |
| EQUATE data-field-name = dss-function                      |     |       | X     |     |    |    |      |      |        |      |       |       |        |    |    |      |    |     |       |       |        | X    |         |    | X   |     |    |     |       |        |    |     |   |
| IF expression-including-dss-function                       | X   | X     | X     |     |    | X  | X    | X    | X      | X    | X     | X     | X      | X  | X  | X    |    |     | X     | X     | X      | X    | X       | X  | X   | X   | X  | X   | X     | X      | X  | X   | X |
| ON dss-function; ~~~                                       |     |       |       |     | X  |    |      |      |        |      |       |       |        |    |    |      | X  |     |       |       |        |      |         |    | X   |     |    |     |       |        |    |     |   |
| ON ~~~ [,dss-function([name] [,SNAP] [,DISABLE])];<br>~~~~ |     |       |       |     |    |    |      |      |        |      |       |       |        |    |    |      |    |     |       |       |        |      |         |    |     |     |    |     |       |        |    |     |   |
| PATCH dss-function = data-field                            |     |       | X     |     |    |    |      |      |        |      |       |       |        |    |    |      |    |     |       |       |        | X    |         |    |     | X   |    |     |       |        |    |     |   |
| PATCH data-field = dss-function                            | X   | X     | X     |     |    | X  | X    | X    | X      | X    | X     | X     | X      | X  | X  |      |    | X   | X     | X     | X      | X    | X       | X  | X   | X   | X  | X   | X     | X      | X  | X   | X |
| PATCH ~~~ [,dss-function(name)]                            |     |       |       |     |    |    |      |      |        |      |       |       |        |    |    |      |    |     |       |       |        |      |         |    |     |     |    |     |       |        |    |     |   |
| REMOVE dss-function                                        |     |       |       | X   |    |    |      |      |        |      |       |       |        |    |    |      |    |     |       |       |        |      |         |    |     |     |    |     |       |        |    |     | X |
| SET dss-function = data-field                              |     |       | X     |     |    | 1  | X    | X    |        | X    | X     | X     | X      | X  | X  | X    |    |     | X     | X     | X      | X    | X       | X  | X   | X   |    |     | X     | X      | X  | X   |   |
| SET data-field = dss-function                              | X   | X     | X     |     |    | X  | X    | X    | X      | X    | X     | X     | X      | X  | X  |      |    | X   | X     | X     | X      | X    | X       | X  | X   | X   | X  | X   | X     | X      | X  | X   | X |

<sup>1</sup>Excluding &C (9:11) in OS/VS1.

Figure 16. Using DSS Functions (Part 1 of 2)

|                                                           | &P | &PATCH | &PSSW | &PSSWN | &PRB | &PRDMP | &PREFIX | &PRM | &PROC | &PSW | &Q | &QT | &RA | &RANGE | &RM | &RPSW | &RPSWN | &S | &SA | &SOUT | &SPSW | &SPSWN | &SVM | &SVMAP | &SYM | &SZ | &T | &TCB | &TCBMAP | &TEA | &TOD | &UNLD |
|-----------------------------------------------------------|----|--------|-------|--------|------|--------|---------|------|-------|------|----|-----|-----|--------|-----|-------|--------|----|-----|-------|-------|--------|------|--------|------|-----|----|------|---------|------|------|-------|
| AT dss-function; ~~~                                      |    |        |       |        | X    |        |         |      |       | X    |    |     |     | X      |     |       |        | X  |     |       |       |        | X    |        |      |     |    | X    |         |      |      |       |
| AT ~~~ [,dss-function([name] [,SNAP] [,DISABLE])];<br>~~~ | X  |        |       |        |      |        |         |      |       |      |    |     |     |        |     |       |        | X  |     |       |       |        |      |        |      |     |    |      |         |      |      |       |
| COLLECT data-field-name = dss-function                    |    |        | X     | X      | X    |        | X       | X    |       | X    | X  | X   |     | X      | X   | X     | X      | X  |     | X     | X     | X      | X    |        |      | X   | X  | X    |         | X    | X    |       |
| DEFINE dss-data-field = dss-function                      |    |        | X     | X      | X    |        | X       | X    |       | X    | X  | X   |     | X      | X   | X     | X      | X  |     | X     | X     | X      | X    |        |      | X   | X  | X    |         | X    | X    |       |
| DISABLE dss-function                                      |    |        |       |        |      |        |         |      |       |      |    |     |     |        |     |       |        | X  |     |       |       |        |      |        |      |     |    |      |         |      |      |       |
| DISPLAY dss-function                                      | X  | X      | X     | X      |      |        | X       | X    | X     | X    | X  | X   |     | X      | X   | X     | X      | X  |     | X     | X     | X      | X    | X      | X    | X   | X  | X    | X       | X    | X    | X     |
| DUMP dss-function                                         | X  | X      | X     | X      | X    | X      | X       | X    | X     | X    | X  | X   |     | X      | X   | X     | X      | X  |     | X     | X     | X      | X    | X      | X    | X   | X  | X    | X       | X    | X    | X     |
| ENABLE dss-function                                       |    |        |       |        |      |        |         |      |       |      |    |     |     |        |     |       |        | X  |     |       |       |        |      |        |      |     |    |      |         |      |      |       |
| EQUATE data-field-name = dss-function                     |    |        |       |        | X    |        |         |      |       |      |    |     |     |        | X   |       |        | X  |     |       |       |        | X    |        |      |     |    | X    |         |      |      |       |
| IF expression-including-dss-function                      |    |        | X     | X      | X    |        | X       | X    |       | X    |    |     |     | X      | X   | X     | X      | X  |     | X     | X     | X      | X    |        |      | X   | X  | X    |         | X    | X    |       |
| ON dss-function; ~~~                                      |    |        |       |        |      |        |         |      |       |      |    |     | X   |        |     |       |        | X  | X   |       |       |        |      |        |      |     |    |      |         |      |      | X     |
| ON ~~~ [,dss-function([name] [,SNAP] [,DISABLE])];<br>~~~ | X  |        |       |        |      |        |         |      |       |      |    |     |     |        |     |       |        | X  |     |       |       |        |      |        |      |     |    |      |         |      |      |       |
| PATCH dss-function = data-field                           |    |        |       |        | X    |        |         |      |       |      | X  |     |     |        | X   |       |        | X  |     |       |       |        | X    |        |      |     |    | X    |         |      |      |       |
| PATCH data-field = dss-function                           |    |        | X     | X      | X    |        | X       |      |       | X    | X  |     |     |        | X   | X     | X      | X  |     |       |       | X      | X    | X      |      |     | X  | X    | X       |      | X    | X     |
| PATCH ~~~ [,dss-function(name)]                           | X  |        |       |        |      |        |         |      |       |      |    |     |     |        |     |       |        | X  |     |       |       |        |      |        |      |     |    |      |         |      |      |       |
| REMOVE dss-function                                       |    | X      |       |        |      |        |         |      | X     |      |    |     |     |        |     |       |        | X  |     |       |       |        |      |        |      | X   |    |      |         |      |      |       |
| SET dss-function = data-field                             |    |        | X     | X      | X    |        | X       |      |       | X    | X  | X   |     | X      | X   | X     | X      | X  |     | X     | X     | X      | X    |        |      |     | X  | X    |         | X    |      |       |
| SET data-field = dss-function                             |    |        | X     | X      | X    |        | X       |      |       | X    | X  |     |     |        | X   | X     | X      | X  |     |       |       | X      | X    | X      |      |     | X  | X    | X       |      | X    | X     |

Figure 16. Using DSS Functions (Part 2 of 2)

| I/O Name                                  | Related Commands | ASSIGN i/o-name<br>= device-address | DIVERT<br>[i/o-name] | RELEASE<br>i/o-name [,...] | SET &DOUT<br>= 'i/o-name' | SET &SOUT<br>= 'i/o-name' | Use                                     |
|-------------------------------------------|------------------|-------------------------------------|----------------------|----------------------------|---------------------------|---------------------------|-----------------------------------------|
| LOG                                       |                  | X                                   |                      | X                          | X                         | X                         | hardcopy log output                     |
| CMDT                                      |                  | X                                   |                      | X                          | X                         |                           | PROCEDURE output<br>(card-image format) |
| PUNCH                                     |                  | X                                   |                      | X                          | X                         |                           |                                         |
| TOUT1                                     |                  | X                                   |                      | X                          | X                         | X                         | displays and dumps                      |
| TOUT2                                     |                  | X                                   |                      | X                          | X                         | X                         |                                         |
| PRINT1                                    |                  | X                                   |                      | X                          | X                         | X                         |                                         |
| PRINT2                                    |                  | X                                   |                      | X                          | X                         | X                         |                                         |
| CARD                                      |                  | X                                   | X                    | X                          |                           |                           | command input                           |
| TIN                                       |                  | X                                   | X                    | X                          |                           |                           |                                         |
| none ( integrated<br>operator's console ) |                  |                                     | X                    |                            |                           |                           |                                         |

Figure 17. I/O Names and Their Uses



## Section 4: User's Guide

This section is a step-by-step description of how to use DSS; in addition, there is information on how DSS responds to errors.

The contents include:

- How to Gain Control
- How to Enter Commands
- DSS Command Processing
- How to Correct an Invalid Command
- How to Display Data
- How to Alter Data
- Command Procedures and Command Streams
- How to Do Simple Branching
- How to Return Control to OS/VS
- Control Levels
- System Errors
- Examples

**Note:** Shaded Information applies only to OS/VS2.

### How to Gain Control

Suppose you want to enter DSS commands at the integrated operator's console. You can:

- Press the RESTART key, which under proper conditions interrupts whatever OS/VS or DSS is doing and allows you to enter commands at the integrated operator's console.
- Use a stored DSS command to divert or return DSS control to the integrated operator's console at a selected point in DSS command execution.
- Combine the second technique, above, with a monitoring interrupt that gives DSS control at a selected point in OS/VS execution.

### When DSS is Dormant

To invoke DSS when OS/VS is running with DSS dormant, press the RESTART key.

Before pressing RESTART, store a nonzero value in the PSADSSGO byte (hexadecimal location 279) of that CPU's prefixed storage area. If this storage modification is done with the CPU's alter/display capability, the CPU must be brought into the manual state; if the QUIESCE operator's command is used to put the CPU in the manual state, the press of the RESTART key that brings the CPU back into operation will not initiate DSS. Therefore, you can set PSADSSGO and not initiate DSS until some later time.

**Note:** Do not press PSW RESTART to invoke DSS.

After you press RESTART to initiate DSS, DSS normally takes control immediately. However, any of these conditions will delay DSS initiation:

- The CPU is in a disabled wait state; DSS remains dormant.
- OS/VS is in a disabled loop; DSS remains dormant.

- The OS/VS1 paging supervisor is executing or has work pending; DSS waits for the completion of this condition, then takes control.
- The OS/VS1 system error task is executing or has work pending; DSS waits for the completion of this condition, then takes control.
- The master scheduler is executing or has work pending; DSS waits for completion of this condition, then takes control.
- The CPU is in the manual state because of a QUIESCE operator's command; a second press of RESTART will give DSS control. Make sure that the CPU is out of the manual state before you press RESTART again.
- PSADSSGO is set to zero; DSS remains dormant.

When DSS initialization is complete, DSS types a message at the integrated operator's console to inform you that it is ready, and types & to prompt you for a command.

**Example Directory**

Figure 1 in Section 1

Figure 23, first box ("press RESTART"), in Section 4

Figure 36 in Section 4

## When DSS is Executing Commands

To get control at the integrated operator's console when DSS is running, you can use either the RESTART key or a DIVERT command.

If you press RESTART while DSS is running, execution of the command stream or command procedure stops. Processing stops between commands, unless a dump, a display, or an input of a command procedure is in progress. DSS writes messages that indicate where processing stopped; in addition, if a command was interrupted, DSS allows you to specify whether the I/O for that command is to be completed. After processing stops and the partially executed command, if any, is either canceled or completed, DSS prompts you to enter commands at the console.

**Example Directory**

Figure 23, first box (RESTART), in Section 4

Figure 36, bottom of second control level ("or press RESTART") in Section 4

If you want, after pressing RESTART, you can return control to the point where DSS command processing was interrupted:

- If a command procedure was interrupted, type REVERT.
- If a command stream was interrupted, type DIVERT  $\left\{ \begin{array}{l} \text{CARD} \\ \text{TIN} \end{array} \right\}$

**Note 1:** When you are caught in an infinite command loop, you should use the RESTART key to get out.

**Note 2:** Command procedures and command streams are further explained under "Command Procedures and Command Streams," "How to Do Simple Branching," and "Control Levels," later in this section.

If you include a DIVERT command (with no operand) in a command procedure or command stream, execution of that command will transfer control to you at the integrated operator's console. In some cases, a RETURN command will have the same effect.

**Example Directory**

Figure 1 in Section 1

"RETURN," example 2, in Section 2

Figure 23, first box (DIVERT and RETURN), in Section 4

Figure 36, bottom of second control level (DIVERT and RETURN ALL), in Section 4

If you want, after entering DIVERT (and sometimes after RETURN), you can return control to the point where DSS command processing was interrupted.

- If a command procedure was interrupted by DIVERT, type REVERT.
- If a command stream was interrupted by DIVERT (or RETURN), type:

```
DIVERT {CARD}
        {TIN }
```

- If a command procedure was interrupted by RETURN, REVERT cannot return control directly to the point of interruption. The procedure can only be reentered from the beginning, with an INVOKE command.

## When DSS is Monitoring

To invoke DSS when OS/VS is running with DSS monitoring, you can use either the RESTART key or a monitoring interrupt.

As when DSS is dormant, use of the RESTART key when DSS is monitoring causes DSS to stop OS/VS, take full control of the CPU, and prompt you to enter commands.

The RESTART key should not be pressed several times quickly (or simultaneously on both CPUs in a multiprocessing system). If this rule is followed, the only times RESTART will fail to bring DSS out of the monitoring mode are when:

- The stop and restart subroutine is halting OS/VS2 (indicated by a wait-state code of CCC, for the QUIESCE command, or a wait-state code for a situation that must be dealt with).
- DSS was signaling another CPU at the instant the key was pressed (indicated by a continuation of normal OS/VS2 processing). Press RESTART again after a few seconds.

**Example Directory**

Figure 23, first box (RESTART), in Section 4

If you want a monitoring interrupt, you must follow these steps when DSS is active and OS/VS is stopped:

1. Enter an AT command or an ON command.
2. If you want a SNAP dump or OS/VS at the time of the interruption, specify SNAP in the &P operand of the AT or ON command.

3. Enter any DSS commands to be executed immediately after the interruption. Include a DIVERT or RETURN ALL command if you want the option of entering commands from the integrated operator's console at that time.
4. Enter an END command.
5. Do any other work that you want to do before DSS starts monitoring.
6. Enter GO to begin the monitoring.

If OS/V5 reaches the location specified by the first operand of the AT command, or performs the action specified by the first operand of the ON command, DSS will regain control and execute the commands that you entered in step 3.

**Example Directory**

- Figure 1 in Section 1
- "AT," examples 1 and 3, in Section 2
- "COLLECT," examples 1 and 2, in Section 2
- "DISABLE" example in Section 2
- "ENABLE" example in Section 2
- "ON," example 2, in Section 2
- "PROCEDURE" example in Section 2
- Figure 22, second box, in Section 4
- Figure 36 in Section 4
- Examples 1, 2, and 6 in Section 4

**Multiprocessing Considerations**

When DSS is dormant, you should press RESTART on the CPU that has I/O access to SYS1.DSSVM and the integrated operator's console. This will avoid much overhead in DSS command processing.

The configuration that gives the best DSS performance is where all devices that DSS is to use (paging data set, integrated operator's console, printers, tape drives, card reader, and card punch) are addressable from the same CPU.

**RESTART Summary**

Figure 18 summarizes the use of the RESTART key.

| Under these conditions               |                            | pressing RESTART gives control to                                                                                                                              |
|--------------------------------------|----------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------|
| DSS                                  | OS/V5                      |                                                                                                                                                                |
| dormant                              | quiesced                   | OS/V52                                                                                                                                                         |
|                                      | in control when PSADSSGO=0 | Recovery Termination Manager                                                                                                                                   |
|                                      | in control when PSADSSGO≠0 | DSS at the integrated operator's console                                                                                                                       |
| in control                           | dormant                    |                                                                                                                                                                |
| monitoring                           | running                    | OS/V52                                                                                                                                                         |
|                                      | quiesced                   |                                                                                                                                                                |
| monitoring and signaling another CPU | running                    | <ul style="list-style-type: none"> <li>● DSS at the integrated operator's console</li> <li>● RESTART is ignored</li> </ul> } Unpredictable; either could occur |

Figure 18. The RESTART Key

## How to Enter Commands

Type commands at the integrated operator's console, according to the format:

label: verb operands

If you type two or more commands on the same line, separate them with semicolons. If you type each command on its own line, semicolons are not required.

**Example Directory**

Example of definitional command and Note 5 under "Contents of a Command," in Section 2  
Comment command example in Section 2

If you want to continue a command on the next line, type `_` as the last character in the line.

**Example Directory**

"END," example 1, in Section 2

If, while typing a line, you notice a typing mistake in that line, you can either (1) cancel the line, by pressing CANCEL, or (2) backspace over the error and retype. If there is no backspace key on your keyboard, use the cursor or type one `?` for each backspace.

**Example Directory**

Second paragraph of "Character Literal," under "Data Fields," in Section 2

Complete the entering of the line by pressing the END key (not to be confused with the END command, which ends command procedures) or the RETURN key. If the integrated operator's console is a display console, end the line by pressing the ENTER key. If DSS types `&` on the next line, it has processed all of your previous input and is waiting for your next command. If DSS types `#` on the next line, it is waiting for you to continue a partially entered command or command procedure.

If DSS types an error message and then `?`, it is waiting for you to type a replacement for a canceled command.

## DSS Command Processing

Figure 19 is an overview of DSS operation.

**Note:** Figure 19 is a guide to the use of DSS, but it does not represent internal DSS logic.

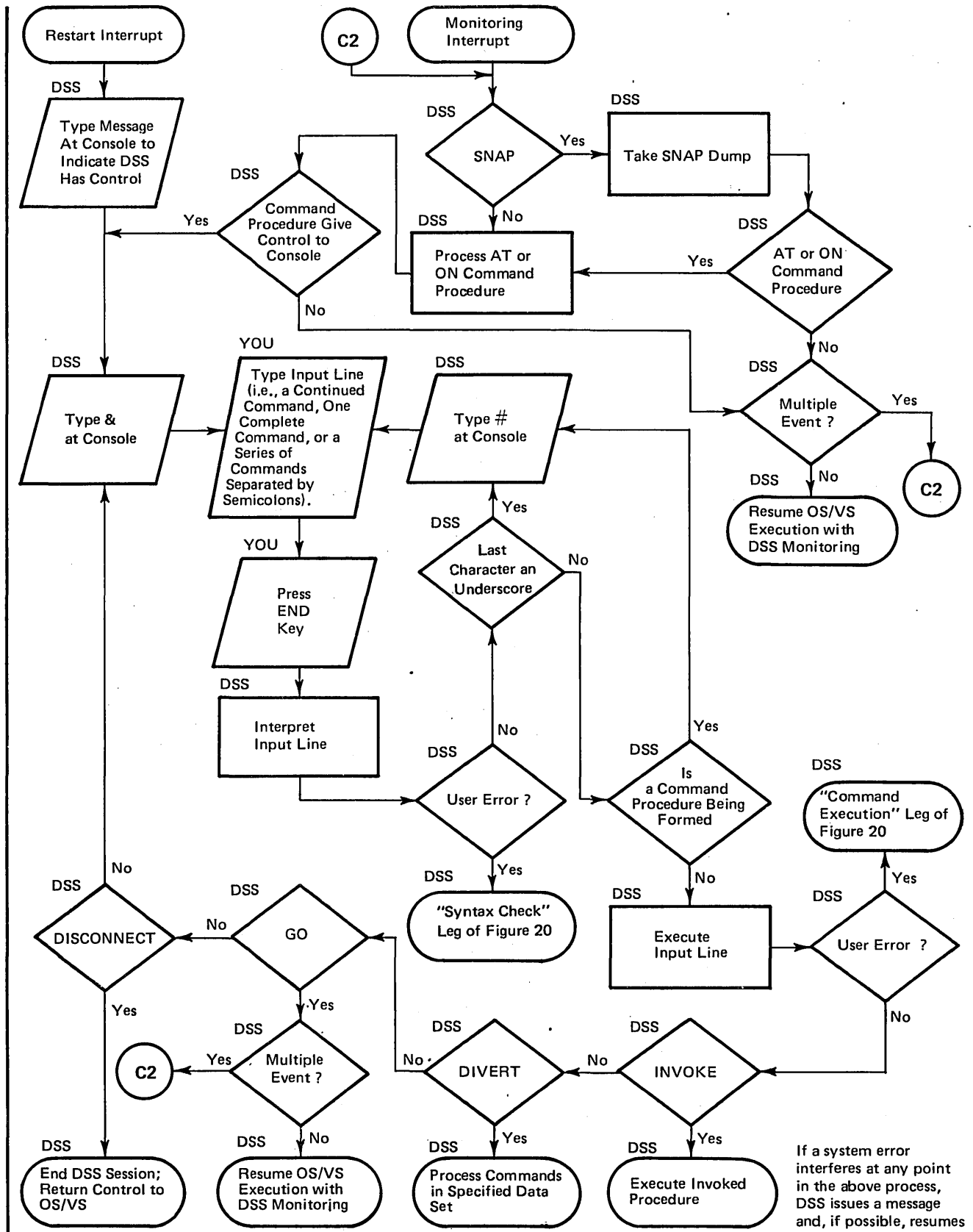


Figure 19. Command Processing

## | How to Correct an Invalid Command

After you enter the command, DSS checks it for syntactical validity:

- If the command is not part of a command procedure, the syntax check is performed immediately.
- If the command is in a command procedure stored by PROCEDURE, the syntax check will be performed when the command procedure is invoked.
- Command procedures stored by AT and ON are syntax-checked immediately, unless they are nested in other command procedures. A nested AT or ON procedure is syntax-checked when the command procedure that contains it is executed.

Some errors are not related to syntax, and are not detected until command execution. Example: An attempt to divert to an unassigned command stream.

When DSS detects the error, it reacts as described in Figure 20. You should then respond according to Figure 21.

**Note:** Figures 20 and 21 are guides to the use of DSS, but they do not represent internal DSS logic.

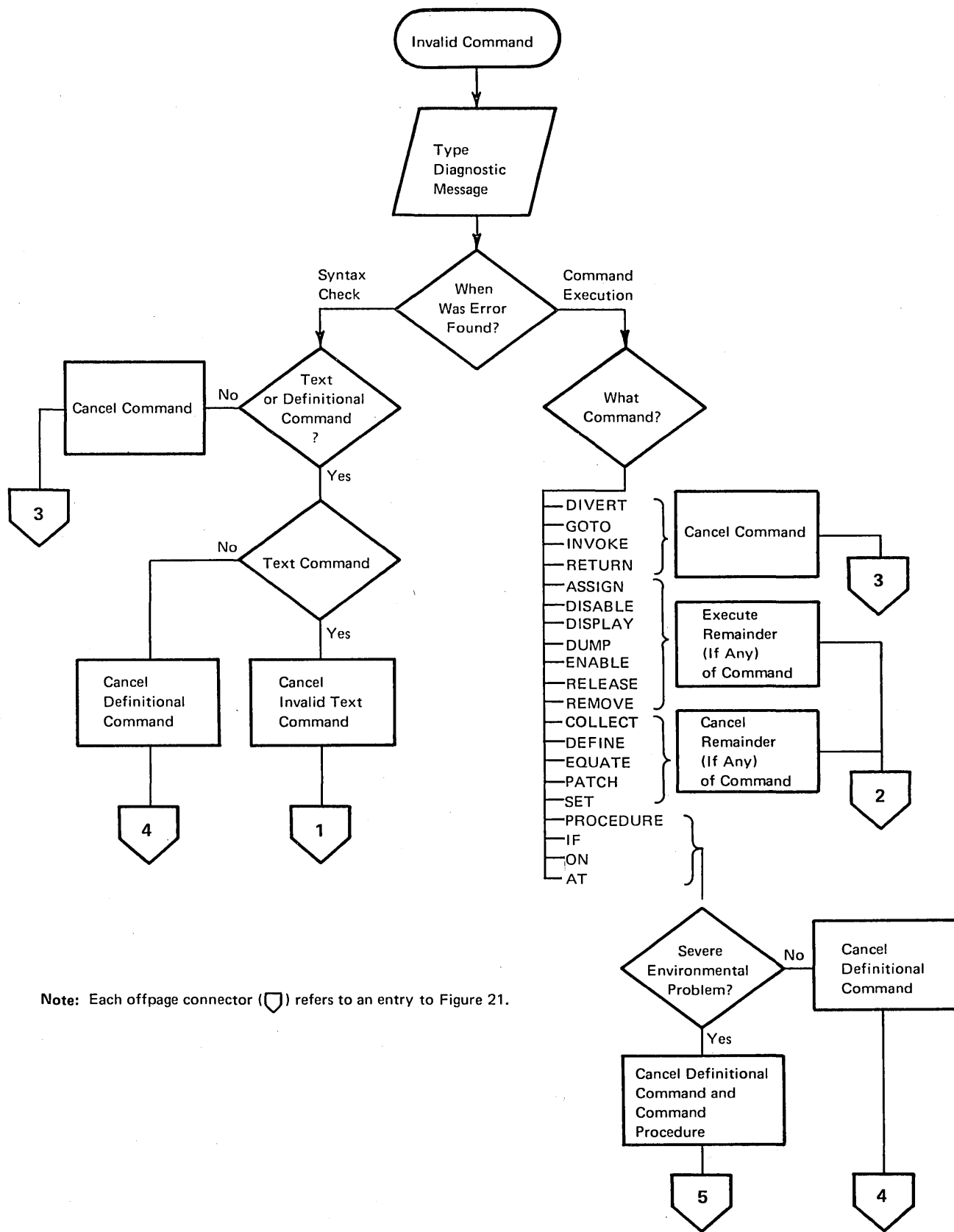


Figure 20. DSS Response to an Invalid Command



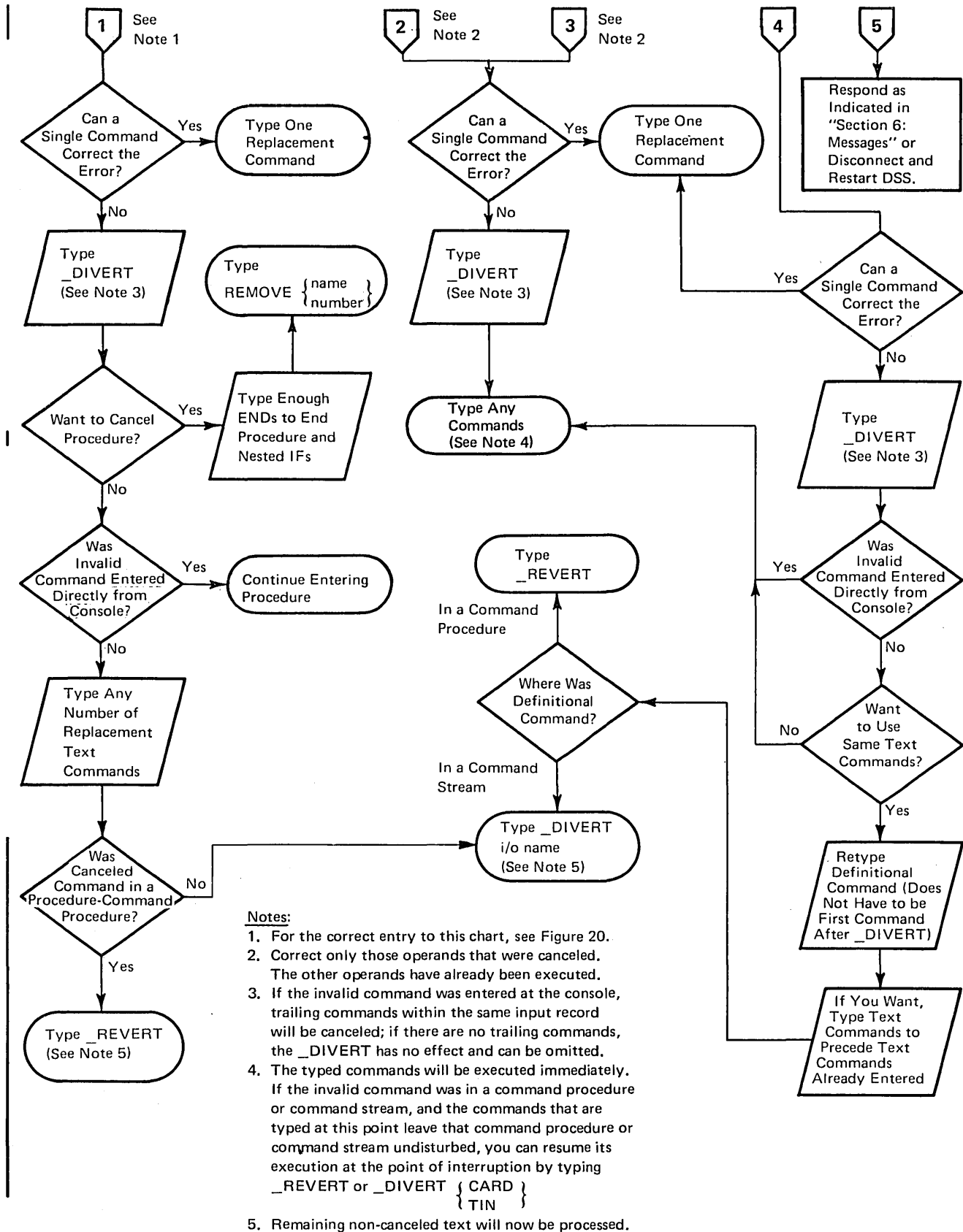


Figure 21. How to Respond to the Canceling of a Command or Command Operand

## How to Display Data

If you want to display a small data field at the integrated operator's console, use the DISPLAY command.

### Example Directory

Figure 1 in Section 1  
Example under "Mapped Data Fields," in Section 2  
"DEFINE," example 1, in Section 2  
"DISPLAY," examples 1-3, in Section 2  
"PROCEDURE" example in Section 2  
Examples 1-6 in Section 4  
DISPLAY examples in Section 5

If you want to dump a large data field to a magnetic tape or high-speed printer, use the DUMP command.

### Example Directory

"DUMP," examples 1-3, in Section 2  
Example 5 in Section 4  
DUMP examples in Section 5

If you want a brief summary of information (written to a printer or tape drive) associated with an OS/VS event, use the SNAP operand of the AT or ON command.

### Example Directory

Example of &S under "Command-Related Functions," in Section 2  
"AT," example 1, in Section 2  
"ON," examples 1 and 2, in Section 2  
Example 6 in Section 4

You can also associate a DISPLAY or DUMP command with an OS/VS event, by including the command in a command procedure that is controlled by an AT or ON command.

### Example Directory

Figure 1 in Section 1  
Example in Note 5 under "Contents of a Command," in Section 2  
Note 1 in "&RA" under "Event Functions," in Section 2  
"AT," examples 1 and 3, in Section 2  
"END," example 1, in Section 2  
"PROCEDURE" example in Section 2  
Examples 1 and 2 in Section 4

You can synchronize a DISPLAY or DUMP command with the filling of a collection area, by using an IF command to detect when the collection area is full.

### Example Directory

"COLLECT," example 2, in Section 2  
"IF," example 2, in Section 2

Before executing a DUMP command or SNAP operand, you should allocate an output I/O name, using the SET command, and an output device, using the ASSIGN command.

### Example Directory

"The SNAP Dump" example in Section 2  
"DUMP" note 1 and example 3, in Section 2  
Examples 5 and 6 in Section 4

## How to Alter Data

To alter data fields defined by you or referred to by DSS functions, use SET.

**Example Directory**

"SET," examples 1, 2, and 4-6, in Section 2  
Examples 3, 5, and 6, in Section 4

The contents of an OS/VS module can be altered by either SET or PATCH.

**Example Directory**

Figure 1 in Section 1  
& Q example under "Location Functions," in Section 2  
"PATCH," example 1, in Section 2  
"SET," example 3, in Section 2  
Example 6 (PATCH command) in Section 4

However, there are two differences between SET and PATCH:

- While DSS keeps no record of a SET operation, it does record the original contents of a patched data field. Thus, you can remove a patch by means of a REMOVE command.
- When the DSS session ends (that is, when you issue the DISCONNECT command), patches are removed but sets are not.

## Command Procedures and Command Streams

If there is a series of commands that you need to use more than once, you could simply type them at the integrated operator's console every time you want DSS to execute them. However, if the series is lengthy, it would be inconvenient to remember and type all the commands each time you use them. It would be easier to store them as a command procedure or a command stream that can be executed whenever you need it.

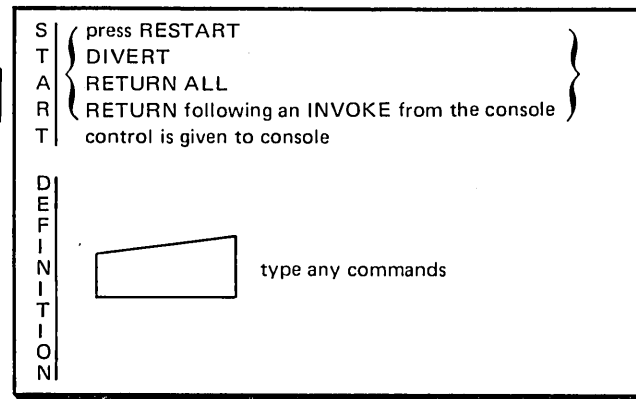
A **command procedure** is a set of commands that is stored in DSS virtual storage. Its advantages over a command stream are:

- There is no physical handling of a data set.
- The GOTO command works in it; you can execute it nonsequentially.

Figure 22 shows how to use the two types of command procedures.



Figure 23 shows how to use the two types of command streams.



OR

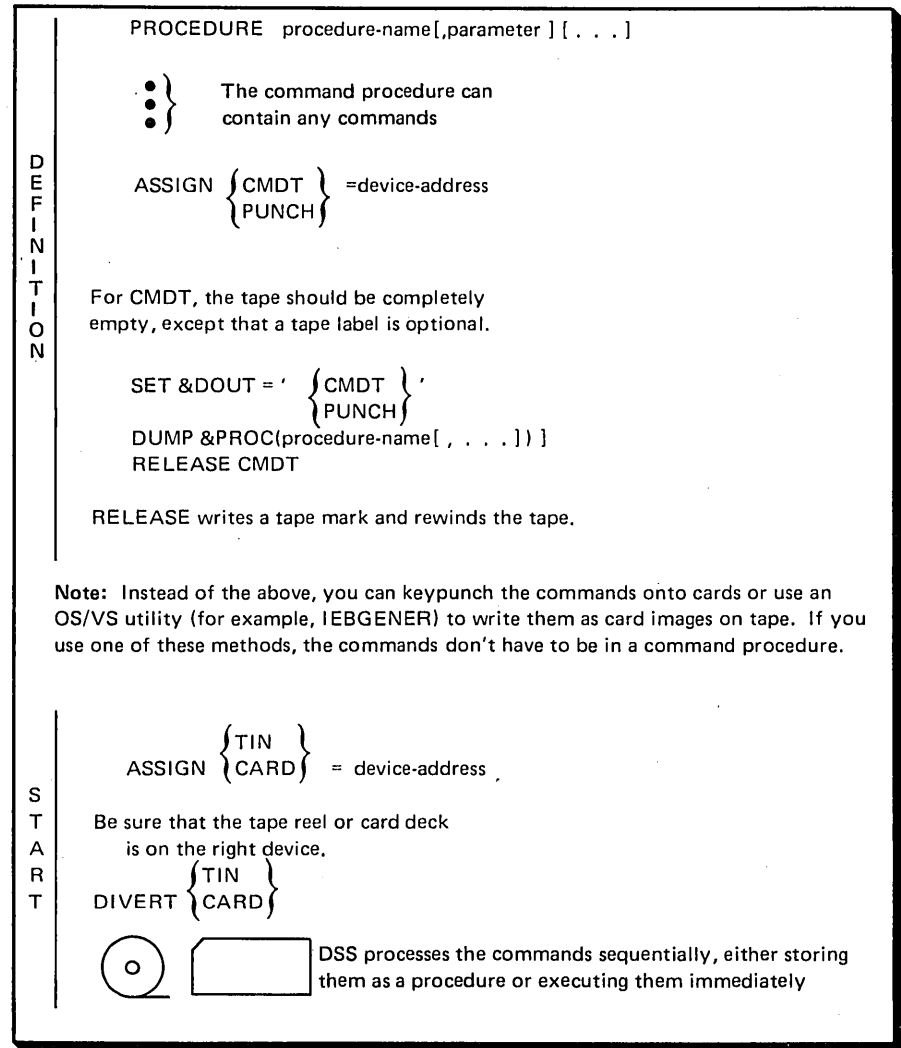


Figure 23. The Definition and Starting of a Command Stream

Use of a stored command stream is shown in the first part of Figure 23: Here, you put the commands on a tape reel or a card deck. Later place the tape reel or card deck on an input device (TIN or CARD) and read it with a DIVERT command.

Use of a console command stream is shown in the second part of Figure 23: to return control to the integrated operator's console from OS/VIS, press RESTART; from another command stream, press RESTART or use DIVERT; from a command procedure, press RESTART or use RETURN.

## How to Do Simple Branching

The GOTO command allows nonsequential execution of any command procedure. See Figure 24.

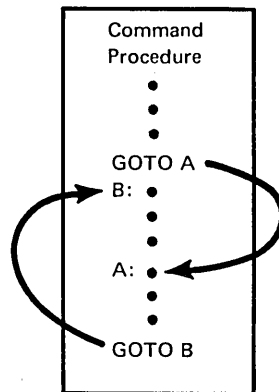


Figure 24. GOTO (Within a Command Procedure)

**Example Directory**  
 "GOTO," examples 1-4, in Section 2

The DIVERT command changes the source of command input from the command procedure to the command stream, if a command procedure was being executed, or from one stream-input device to another, if a command stream was being executed. See Figure 25.

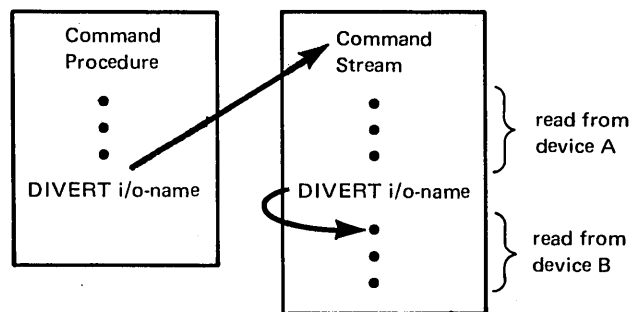


Figure 25. DIVERT

**Example Directory**  
 "DIVERT," examples 1-3, in Section 2

After a DIVERT command, either a REVERT or a GOTO command can be used to return control to the control level's command procedure. See Figures 26 and 27.

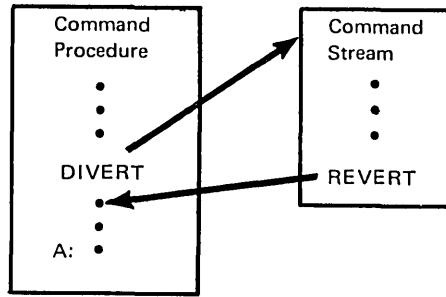


Figure 26. REVERT

Example Directory  
 "REVERT" example in Section 2

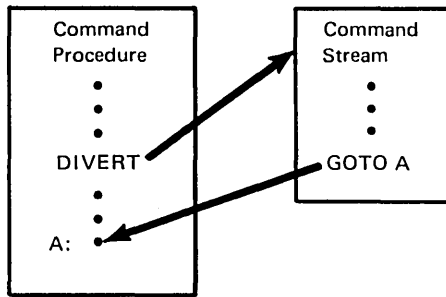


Figure 27. GOTO (Command Stream to Command Procedure)

Note: Compare Figure 27 with Figures 24 and 26.

## How to Return Control to OS/VS

There are two commands that return control to OS/VS: GO and DISCONNECT.

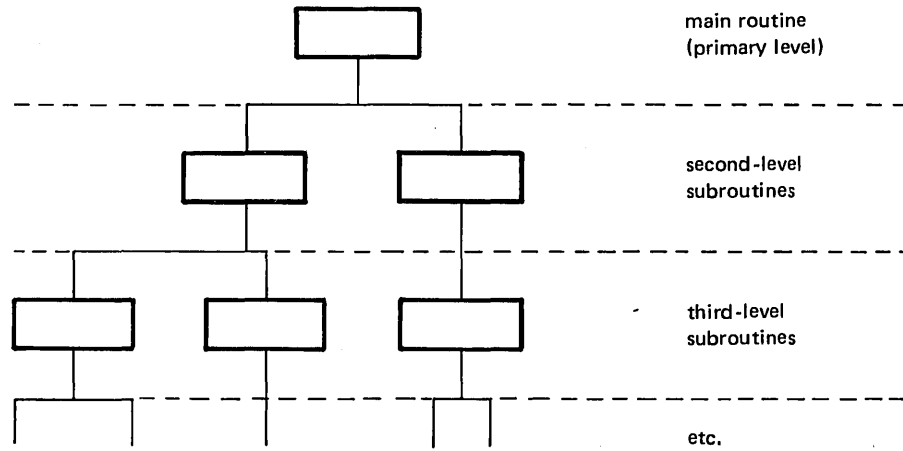
GO terminates DSS command processing, but does not terminate the DSS session. Any DSS AT or ON command that is enabled will cause a monitoring interrupt, and further DSS command processing, if the AT location is reached or the ON event occurs. In addition, the RESTART key on the console control panel can also be used to reactivate DSS.

**Note:** In an AT or ON command procedure, RETURN has the same effect as GO.

DISCONNECT terminates the DSS session. All ATs, ONs, patches, command procedures, and symbols that you have defined are removed. DSS will now be inactive until someone presses RESTART again.

# Control Levels

The concept of levels is sometimes applied to a hierarchy of programs:



This concept, when applied to a DSS session, helps illustrate your flexibility in choosing a source of DSS command input.

Here's a simple example: Each DSS session begins at the primary control level. If procedure A is then the first command procedure that you invoke with the INVOKE command, A is in the second control level. If A invokes procedure B, command execution goes to the third control level. If B contains a RETURN command that returns control to A, command execution returns to the second control level.

That's a general idea of what control levels are in DSS; now for the details. Although the subject may seem complex at first, it will give you a visual technique for remembering how to use all DSS branching commands.

The primary control level is the first control level after DSS initiation; see Figure 28.

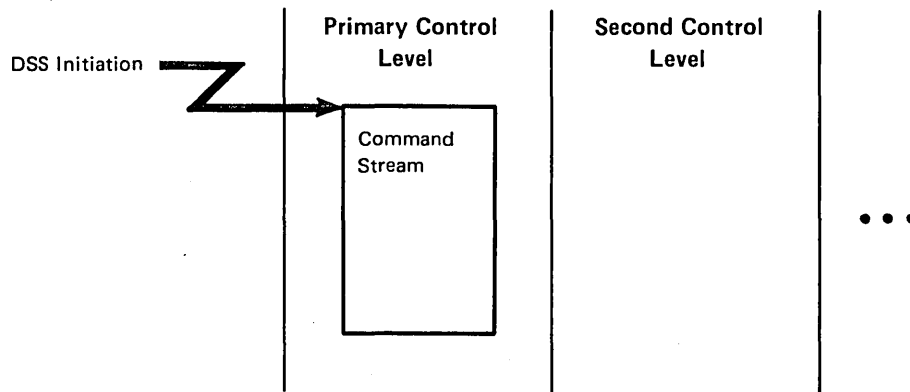


Figure 28. After DSS Initiation



At this point, DSS has typed DSS READY at the console, followed by a carrier return and an & that invites you to type your first command. As yet, no command text has been executed. The primary control level has no command procedure—it consists only of a command stream. Initially, this command stream is the conversational command stream that occurs when you type commands at the console and DSS replies at the console; however, the DIVERT command can change the source of the command stream to a card reader or tape drive.

Every control level except the primary control level has one command procedure and one command stream. See Figure 29.

A monitoring interrupt gives control to a command procedure at the second control level. See Figure 30.

Aside from the GO command, INVOKE and RETURN are the only commands that change control levels. They can be issued from either a command procedure or a command stream.

The RETURN command, when used with its ALL operand, allows you to return control to the primary control level from any control level. See Figure 31.

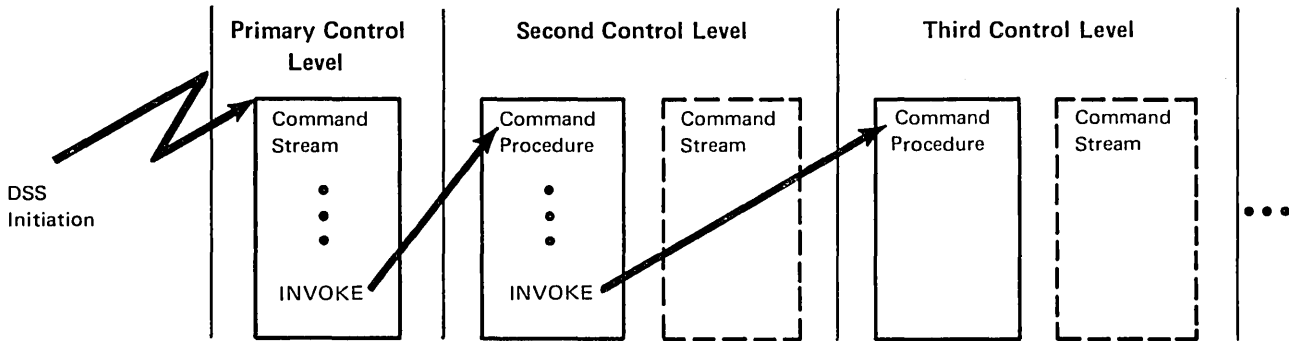


Figure 29. A Succession of Control Levels

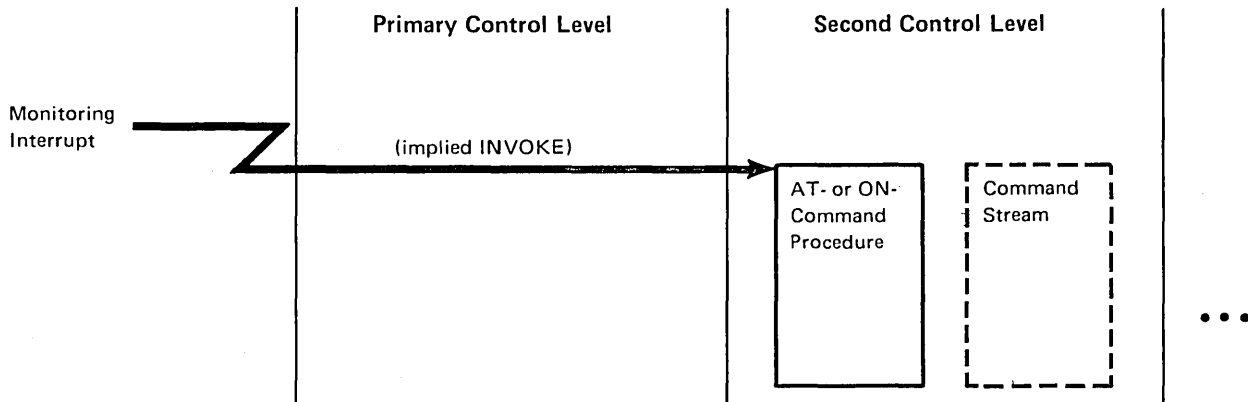


Figure 30. After a Monitoring Interrupt

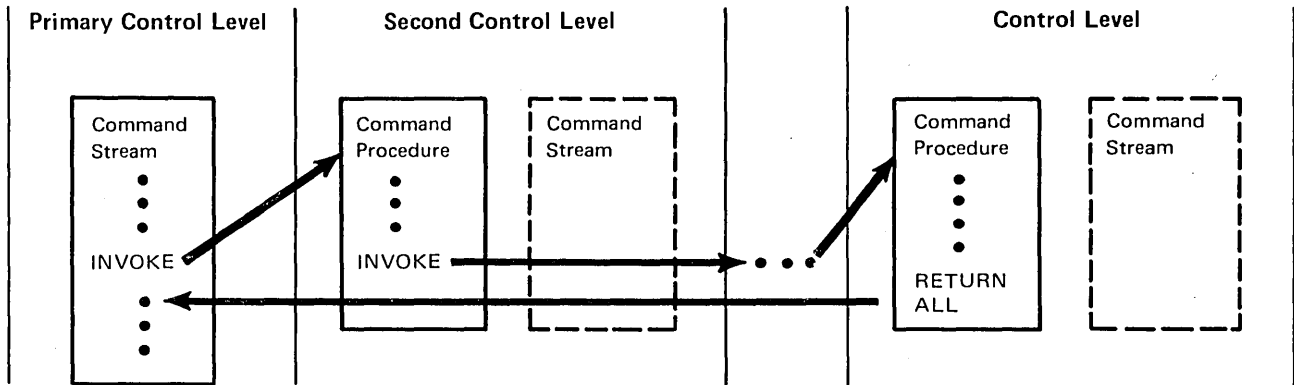


Figure 31. Skipping Control Levels on Return

**Notes for Figures 29, 30, and 31**

**Note 1:** The ALL operand must be used whenever control is to be returned to the primary control level from a command procedure stored by AT or ON.

**Note 2:** DIVERT, REVERT, GOTO, and the attention interrupt do not change control levels.

**Note 3:** In the primary control level, a REVERT is treated as a DIVERT with no operand (that is, a DIVERT to the integrated operator's console).

Figure 32 shows the sources of command input within any control level except the primary control level.

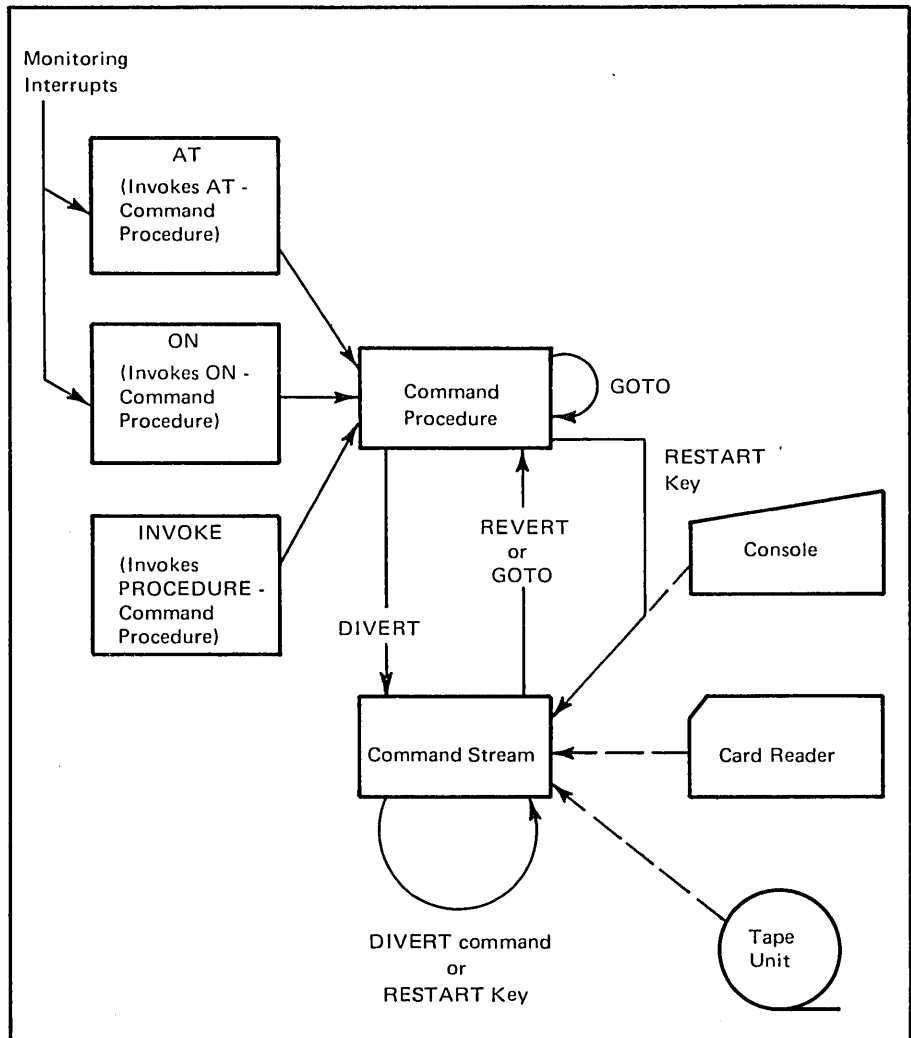


Figure 32. Command Inputs Within a DSS Control Level

Figure 33 shows the sources of command input within the primary control level.

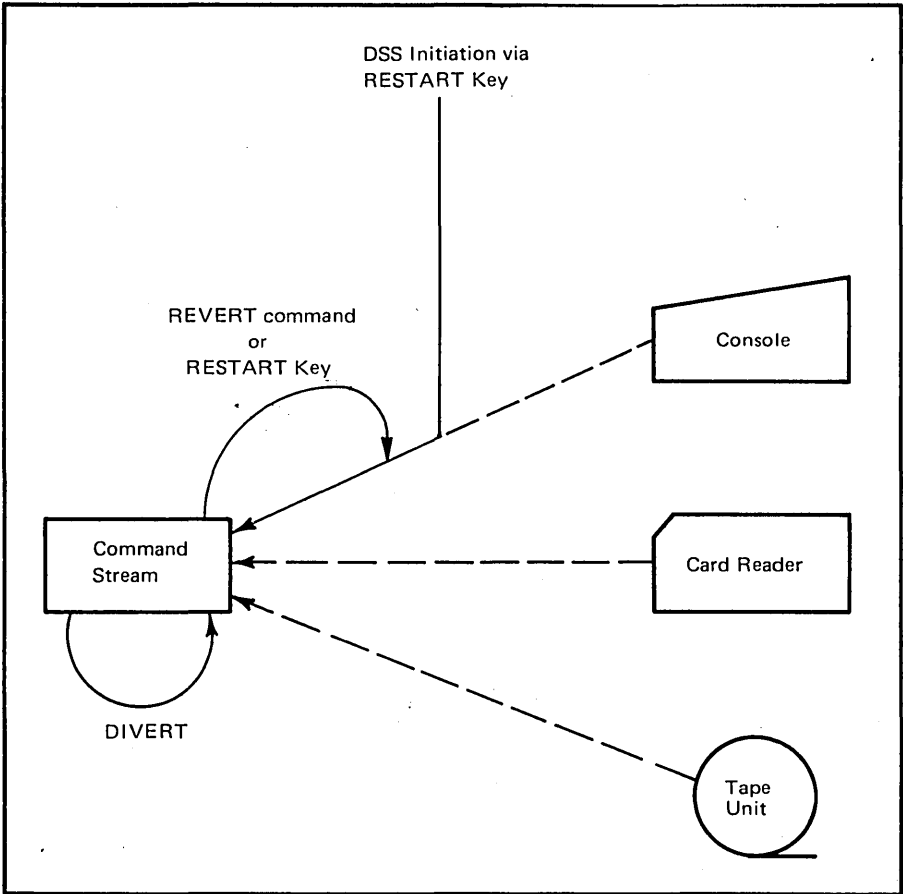


Figure 33. Command Inputs Within the Primary Control Level

Figure 34 shows how to get to any DSS control level.

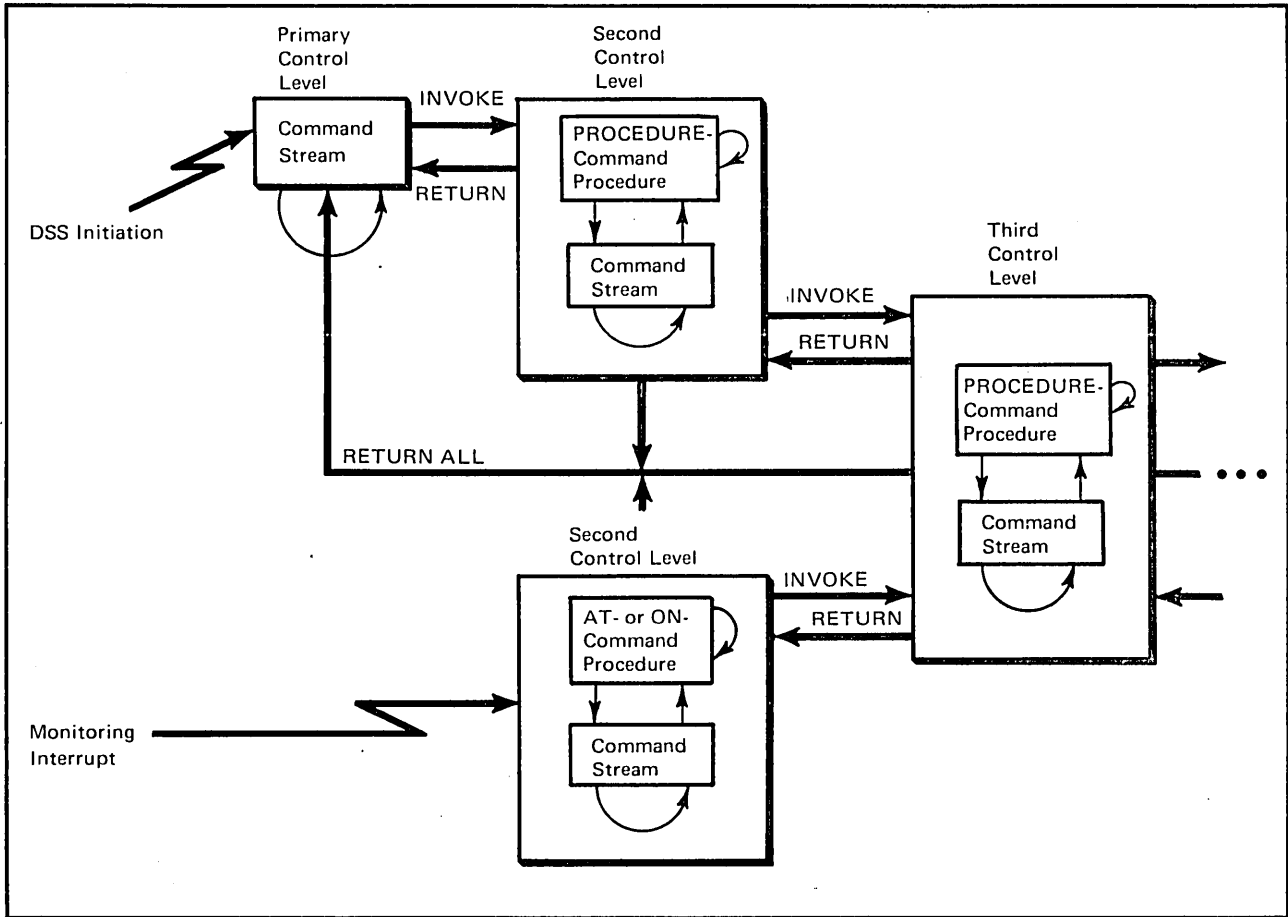


Figure 34. DSS Control Levels

Figure 35 shows how to return control to OS/VS from any control level.  
Figure 36 shows a hypothetical example of DSS command execution.

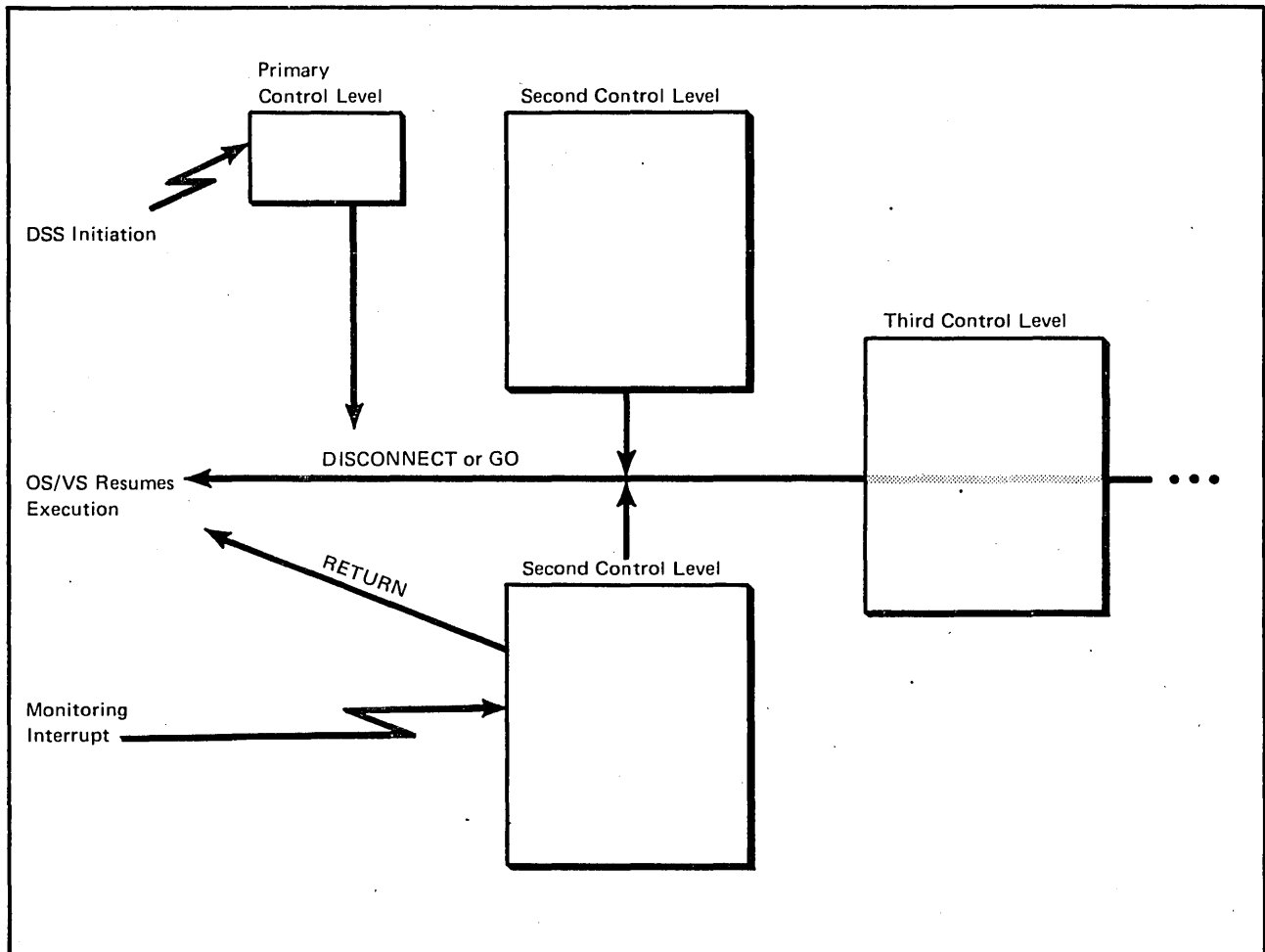


Figure 35. Return of Control To OS/VS

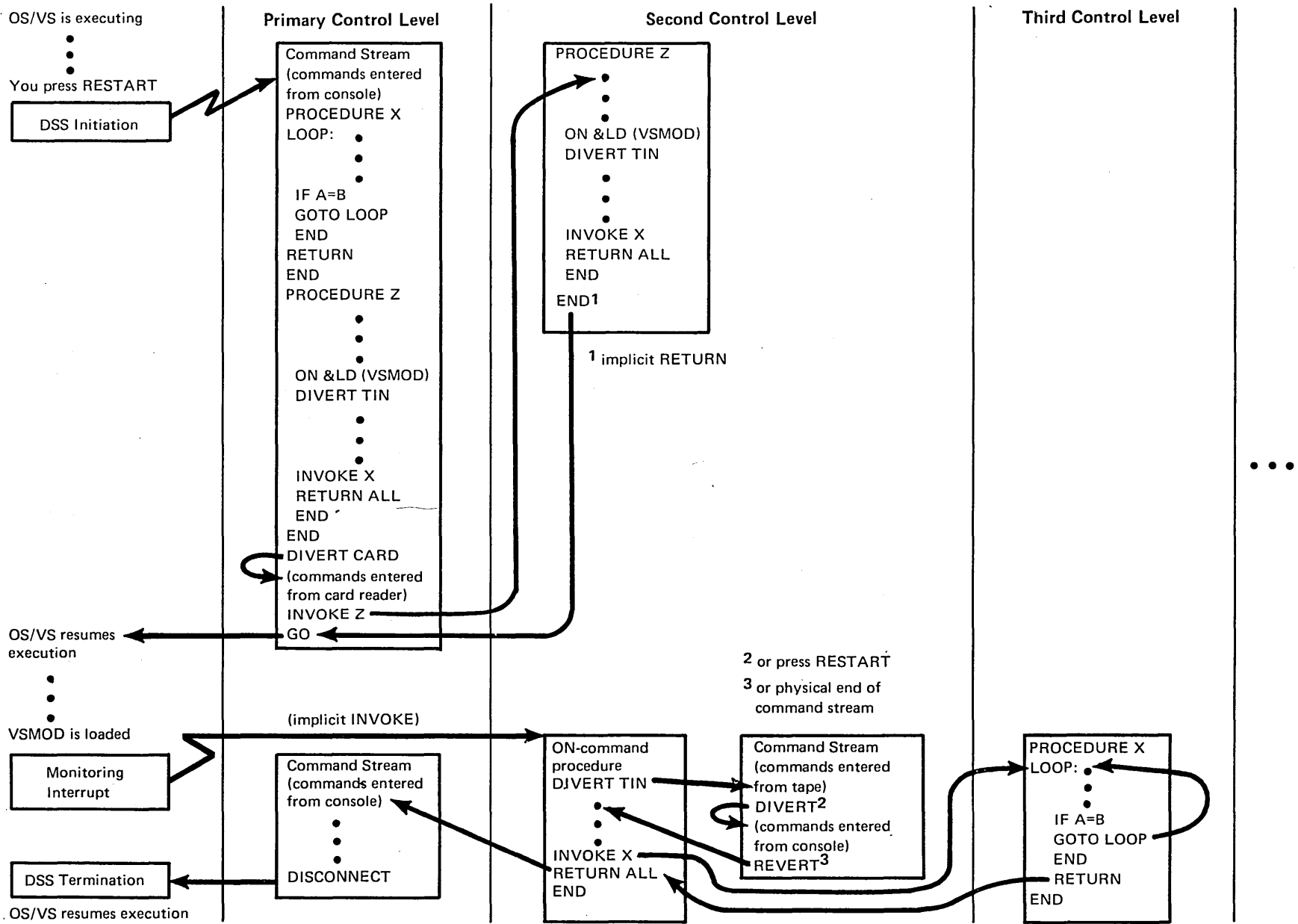


Figure 36. Example Of A DSS Session

## System Errors

The system-error messages, which are explained in “Section 6: Messages,” are IQA000 to IQA099.

When DSS detects a device error, a channel error, or an error in DSS logic that may threaten the integrity of DSS or OS/VS, DSS issues a system-error message in the following format:

| <u>IQA000D</u>         | <u>&lt;SEVERE&gt;</u> | <u>DSS PROGRAM CHECK,</u>                           | <u>LOC. ID. 02,</u>        | <u>IQAPIH00</u> | <u>IQAPAG00</u> |
|------------------------|-----------------------|-----------------------------------------------------|----------------------------|-----------------|-----------------|
| Message Identifier     | Error Severity        | Message Text                                        | Location In Current Module | Current Module  | Failing Module  |
| Second Line of Message |                       | <b>CANNOT TAKE DSS DUMP, CONTINUE? REPLY Y OR N</b> |                            |                 |                 |
|                        |                       | <b>DSS DUMP DESIRED? REPLY Y OR N</b>               |                            |                 |                 |

**Message Identifier:** IQAnnt, where IQA is the operating-system component identifier for DSS; nnn is a three-digit serial number; and t is one of these codes:

| Code | Meaning                          |
|------|----------------------------------|
| A    | Immediate action required        |
| D    | Immediate decision required      |
| E    | Eventual action required         |
| I    | Information (may provoke action) |
| W    | The system is in the wait state  |

**Error Severity:** The severity of the error that caused the message. See the description of error severities under “Resumption of Processing” later in this section.

**Message Text:** What happened.

**Location in Current Module:** The location where the error was detected; the current module’s listing relates this number to that location.

**Current Module:** The name of the module that detected the error.

**Failing Module:** The name of the module directly concerned with the error, if that name is available.

**Second Line of Message:** In OS/VS1, one of these lines is printed as the second line of the message. (In OS/VS2, these are issued as separate messages.)

The following situations could prevent DSS from taking a dump:

- The OS/VS1 paging supervisor is not loaded.
- Both OS/VS1 dump areas are full (see Note 1 under “How To Obtain A Dump,” which follows).
- A permanent I/O error (indicated by “DSS DUMP DESIRED?...” and then “CANNOT TAKE DSS DUMP...”).
- **Recursion into the DSS error processor while it was trying to dump DSS.**

In the case of a hard machine check interrupt or translation specification exception, DSS issues a different type of system-error message and then proceeds according to the severity of the error. See the description of catastrophic or major errors under “Resumption of Processing,” later in this section.



## How To Obtain A Dump

If DSS offers a dump and you type Y, DSS dumps itself, real storage from locations 0 to 4096, and registers. (The format of this dump is shown under "Error Dump," in Section 5.) In addition, this dump is taken automatically, without the Y or N prompt, if the system error prevents I/O to the console but allows the DSS dump. In OS/VS2, DSS dumps directly to a printer. The OS/VS1 dump, a page-formatted data set written to SYS1.DSSVM, could be printed by JCL such as:

```
//jobname JOB MSGLEVEL=(1,1)
//stepname EXEC PGM=IQAEP00
//SYSIN DD DSNAME=SYS1.DSSVM,DISP=OLD
//SYSPRINT DD SYSOUT=A,SPACE=(CYL,(5,5))
```

For more information on JCL options, refer to **OS/VS1 JCL Reference, GC24-5099**.

**Note 1:** In OS/VS1, space has been allocated for two separate dumps. When both areas are full, at least one dump must be printed before DSS can take another dump. When both areas have been printed, IQAEP00 prints NO DUMPS OUTSTANDING. Execution of IQAEP00 requires approximately 60K of real storage.

If DSS could not take the dump, you should dump real storage by running A/HMDSADMP as described in Section 6, Table I, item 11.

## Resumption of Processing

If DSS cannot take the dump and you type N in response to the CONTINUE? prompt, DSS gives control to the OS/VS machine check handler (MCH), which attempts to issue a message to the integrated operator's console and puts the system into a disabled wait state.

If you type Y after CONTINUE? or if DSS offers the option of a dump and you type Y or N, DSS attempts to resume processing. DSS attempts to recover from device and channel errors as described in Section 6, in the "System Action" paragraph for the message issued. Response to an error in DSS logic, or response after failure to recover from a device or channel error, depends on the severity of the error.

There are four severities of system error:

1. **Catastrophic (OS/VS1) or Major (OS/VS2):** Probable modification of DSS when successful return to the OS/VS environment is unlikely, a machine check interrupt during DSS operation when the machine check handler couldn't recover, a translation specification in DSS virtual memory, or loss of OS/VS integrity due to an AT that couldn't be removed from the nucleus or LPA. DSS issues another message (IQA016 or IQA010) and gives control to the OS/VS machine check handler; OS/VS then enters the disabled wait state.
2. **Severe:** The integrity of DSS is questionable, but it is likely that DSS can back out of the system and reinstate the OS/VS environment. DSS issues another message (IQA014) and relinquishes control as if a DISCONNECT command had been issued; OS/VS then runs with DSS terminated. The RESTART key can be used to reactivate DSS.

3. **Local:** Accidental modification of DSS or a DSS logic error, where the problem can be isolated; a successful reinstatement of the DSS environment is likely through reinitialization. DSS issues another message (IQA015), gives control to DSS Initialization, prints the DSS READY message at the console, prints an & , and waits for you to enter a command.
4. **Minor:** An error that does not affect further DSS processing.

## Examples

Following are examples of some of the ways DSS might be used to debug.

### Example 1

This will cause OS/VS1 to stop at entry to ABTERM when the ABEND code is 0C1.

```
PROCEDURE ABTERM,CC
DEF COMPCODE=&S(CC)                                ;*SAVE THE PARM VALUE
  AT L'4C'%. (52)%                                  ;*SET THE STOP
  IF &G(1)=COMPCODE                                ;*IS IT THE ABEND WE WANT
  DISPLAY 'ABEND HIT', &PSW, &G(0:15)             ;*TELL PROGRAMMER
  DVT                                              ;*GIVE PROGRAMMER CONTROL
  END                                              ;*END OF IF TEXT
END                                                ;*END OF AT TEXT
END                                                ;*END OF PROC
```

These commands would invoke the procedure and return control to OS/VS:

```
INVOKE ABTERM,X'800C1000'
GO
```

### Example 2

This will cause OS/VS1 or OS/VS2 to stop when message IEF244 is issued.

```
PROC MSG,NUMBER
DEF NUM.(0,6,C,6)=NUMBER                          ;*SAVE THE PARM VALUE
  AT &LM(IEAVVWTO).(2)                             ;*STOP ON ENTRY TO WTO1
  IF &G(1)%. (4,6,C)=NUM                            ;*IS THIS THE MSG
  DISPLAY 'MSG &S(NUM) ISSUED'                     ;*TELL PROGRAMMER
  DVT                                              ;*GIVE HIM CONTROL
  REMOVE &SYM(NUM)                                 ;*WIPE OUT THE PARM VALUE
  END                                              ;*END OF IF TEXT
END                                                ;*END OF AT TEXT
END                                                ;*END OF PROC
```

These commands would invoke the procedure and return control to OS/VS:

```
INVOKE MSG,IEF244
GO
```

### Example 3

A command procedure finds and displays an OS/VS1 or OS/VS2 UCB (unit control block).

```
PROC FINDUCB,UNIT
DEF TBLPTR=L'10'%. (40)                            ;*GET UCB TABLE POINTER
DEF UCBPTR                                         ;*DEFINE FULLWORD FOR INDIR ADDR
SEARCH: IF TBLPTR%. (0,2,X)=X'FFFF'                ;*IS THIS END OF UCB LOOK-UP TABLE
  DISPLAY 'UCB NOT FOUND'                          ;*YES, DISPLAY UCB NOT FOUND
  GOTO EXIT                                         ;*GET OUT
END                                                 ;*END OF IF TEXT
*
IF TBLPTR%. (0,2,X)=X'0000'                        ;*IS IT A VALID ADDR
SET TBLPTR=TBLPTR+2                                ;*NO, INCR THE TABLE ADDR
GOTO SEARCH                                         ;*LOOP BACK
END                                                 ;*END OF IF TEXT
```

---

<sup>1</sup> In OS/VS1, substitute IGC0003E for IEAVVWTO.

```

SET UCBPTR=TLBPTR%. (0,2,X)          ;*SET UCBPTR
IF UCBPTR%. (13,3,C)=UNIT           ;*IS THIS THE UCB
EQU UCB&S(UNIT). (0,64,X)=UCBPTR%  ;*PUT A LABEL ON IT
DISPLAY UCB&S(UNIT)                ;*DISPLAY THE UCB
GOTO EXIT                           ;*GET OUT
END                                  ;*END OF IF TEXT
SET TLBPTR=TLBPTR+2                 ;*INCR THE TABLE ADDR
GOTO SEARCH                          ;*GET NEXT UCB ENTRY
EXIT: REMOVE &SYM                   ;*REMOVE THIS PROC'S DATA FIELDS
END                                  ;*END OF PROC

```

Example of an INVOKE command for the above command procedure:

```
INVOKE FINDUCB,182
```

#### Example 4

This displays the address of the current OS/VS1 TCB (task control block) and, if the TIOT (task input/output table) pointer is not zero, displays the jobname, stepname, and procstep name. Note: If this command procedure is invoked when DSS is fully active (that is, after the DSS READY message), it will give the address of the DSS TCB. If the procedure is to give the address of the current OS/VS1 TCB, it must be invoked dynamically (via AT or ON).

```

PROC CURRENT
EQU TCB=L'4C'%%.(4)                 ;*INDIRECT ADDR OF NEWOLD
D TCB%. (0,32,X)                    ;*DISPLAY THE FIRST 32 BYTES
IF TCB%. (12,4,X)=0                 ;*TIOT POINTER ZERO
REMOVE &SYM(TCB)                    ;*YES, REMOVE THE SYMBOL
RETURN                               ;*GO BACK TO INVOKER
END                                  ;*END OF IF TEXT
EQU TIOT=TCB%. (12)                 ;*GET THE TIOT ADDR
DISPLAY TIOT%. (0,24,C)              ;*PRINT THE NAMES
REMOVE &SYM(TCB,TIOT)               ;*DELETE THE SYMBOLS
END                                  ;*END THE PROC

```

#### Example 5

This dumps the OS/VS1 TCB chain.

```

PROC DTCB
DEF PTR=L'10'%. (160,4)             ;*GET CVTHEAD
SET &HDR='TCB DUMP'                 ;*NAME THE DUMP
LOOP: DUMP PTR%. (,184)              ;*DUMP TCB
SET PTR=PTR%. (116,4)               ;*UPDATE TCB POINTER
IF PTR=X'0'                          ;*IS THIS END OF TCB CHAIN
REMOVE &SYM(PTR)                     ;*YES, REMOVE SYMBOL
D 'TCB DUMP COMPLETE'               ;*MESSAGE TO PROGRAMMER
RETURN                               ;*RETURN TO INVOKER
END                                  ;*END OF IF TEXT
GOTO LOOP                            ;*GET NEXT TCB
END                                  ;*END OF PROC

```

The following commands allow execution of the above procedure:

```

ASSIGN PRINT1=X'00E'
SET &DOUT='PRINT1'
INVOKE DTCB

```

### Example 6

DSS can be used to find random alterations of storage. For example, during the testing of DSS we found that somehow the restart PSW at location 0 was being altered by an OS/VIS module. We used DSS to find the module, bypass the problem, and apply the fix. The following DSS commands were entered from the integrated operator's console to trap the problem.

|                           |                                                            |
|---------------------------|------------------------------------------------------------|
| ASGN PRINT1=14            | Assign a printer to PRINT1.                                |
| SET &SOUT='PRINT1'        | Specify PRINT1 for SNAP output.                            |
| SET &RANGE=L'0'           | Monitor program events in loc 0-3.                         |
| ON &SA, &P(ZAPZERO, SNAP) | SNAP when PSW altered.                                     |
| REL PRINT1                | Release PRINT1; this causes the SNAP buffer to be printed. |
| END                       | End of ON text.                                            |
| GO                        | Resume OS/VIS execution.                                   |

The following SNAP output was produced.

```
SNAP AC=10 ADDR=00FC5A2C OPSW=470D100000FC5A30 INST=92000001
GPR=00 00118001 0007E760 0007E820 0007EF44 00000001 0000283C 00000000 0000283C
GPR=08 40FC51F6 00FC50AC 0007E880 0007E838 00DEF408 00DEF470 0007E820 00118000
```

From this, we learned which module contained the bug and applied the following temporary bypass.

|                                |                                                          |
|--------------------------------|----------------------------------------------------------|
| DEF TEMPFIX=X'47000000'        | No-op the bad instruction.                               |
| SET &QT=&SVM.IEFXX000          | Establish the module ID.                                 |
| DISPLAY IEFZZ000.(X'13A4')     | Verify that this is the location of the bad instruction. |
| PAT IEFZZ000.(X'13A4')=TEMPFIX | Patch in the bypass.                                     |
| GO                             | Return to OS/VIS.                                        |

This bypassed the problem until we received the permanent fix, which was zapped on the system and patched in using DSS.

### Example 7

This command procedure traps storage alteration over a range of virtual addresses.

```
PROC TRAPSA, RANGE, ASID
SET &QT=&ASID( &S(ASID) )
SET &RANGE=&S(RANGE)
ON &SA, &P(TRAPSA)
D 'MONITORED STORAGE ALTERED'
D &RANGE, &AC
D &ID( &AC.(2,3)% )
D '**INSTRUCTION RESPONSIBLE**'
IF &AC.(2,3)%.(,1)<X'CO'
D &AC.(2,3)%.(,4)
GOTO LOO100
END
D &AC.(2,3)%.(,6)
LOO100: D '**CONTENTS, AFTER ALTERATION, OF MONITORED STORAGE**', &S(RANGE)
D &G(0:15)
D '**LAST TRACE TABLE ENTRY**', L'54'%%.(,32)
DVT
END
RETURN
END
```

Example of an INVOKE command for the above command procedure:

```
INVOKE TRAPSA,L'100:200',0
```

### Example 8

This command procedure displays an ASID's ASCB and ASXB.

```
PROC ASCB,ASID
  IF &S(ASID)=X'FO'
    D 'ASID 0 NOT VALID' ;*ASID STARTS WITH 1
  GOTO END
  END
  DEF ASCB&S(ASID).(0,X'C8',X) ;*DEFINE ASCB FIELD
  DEF ASXB&S(ASID).(0,X'E8',X) ;*DEFINE ASXB FIELD
  DEF A=L'10'%.(X'22C')%.(528+4*( &S(ASID)-1),4) ;*DEFINE A FIELD TO POINT AT THE ASVT
  IF A.(0,1)=X'80' ;*IS THIS A VALID ASCB POINTER?
  D'ASID NOT ASSIGNED'
  GOTO END;
  END
  SET ASCB&S(ASID)=A%.(,X'C8') ;*SET ASCB FIELD = ASCB
  SET ASXB&S(ASID)=A%.(X'6C')%.(0,X'E8') ;*SET ASXB FIELD = ASXB
  D ASCB&S(ASID),ASXB&S(ASID) ;*DISPLAY ASID NO., ASCB, &ASXB
  END: REM &SYM(ASCB&S(ASID),ASXB&S(ASID))
  END
```

### Example 9

This command procedure dumps one virtual memory, except the LPA to a tape.

```
PROC DTAPE,TAPE,ASID,DESC
  SET &QT=&ASID( &S(ASID))
  ASGN TOUT1=X' &S(TAPE)'
  SET &HDR=' &S(DESC) '
  DUMP &SVMMAP
  DUMP &PSW,&G(0:15)
  DUMP L'0':IEACVT.(X'169',3)%
  DUMP IEACVT.(X'230',4)%.(X'18',4)%.(4,4)%.(9,3)%L'FFFFFF'
  RELEASE TOUT1
  END
```

Example of an INVOKE command for the above command procedure:

```
INVOKE DTAPE,280,0,TAPEDUMP
```

### Example 10

These command procedures define a DSS symbol for an OS/VS2 direct-access device UCB, then make the device either shared or nonshared.

```
PROC DEFUCB,UNIT
  *****DEFINE A DSS SYMBOL FOR A UCB*****
  DEF X1.(,2)=X' &S(UNIT)' ;*BINARY UNIT ADDRESS
  DEF X2=IEACVT.(X'24',4) ;*CHANNEL INDEX LIST ADDRESS
  DEF X3=X2+(X1/256) ;*CHANNEL ENTRY ADDRESS
  L01: IF X3>X2 ;*LOOP TO ENSURE SYSGENED CHANNEL ADDRESS
    IF X2%.(,1)=X'FF'
      D UNIT,'CHANNEL HIGHER THAN HIGHEST SYSGENED'
      GOTO EXIT
    END
  SET X2=X2+1
  GOTO L01
  END
  *
  IF X3%.(,1)=X'00'
    D UNIT,'CHANNEL NOT SYSGENED'
```

```

GOTO EXIT
END
SET X2=IEACVT.(X'24',4)+X3%.(,1)+2*((X1//256)/16) ;*CONTROL UNIT ENTRY ADDRESS
IF X2%.(,2)=X'0000'
D UNIT,'CONTROL UNIT NOT SYSGENED'
GOTO EXIT
END
SET X3=IEACVT.(X'28',4)+2*(X2%.(,2)+X1//16) ;*DEVICE ENTRY ADDRESS
IF X3%.(,2)=X'0000'
D UNIT,'UNIT NOT SYSGENED'
GOTO EXIT
END
EQU UCB&S(UNIT).(,64,X)=X3%.(,2)% ;*DEFINE THE SYMBOL
IF UCB&S(UNIT).(4,2)≠X1
D 'UNIT HAS MULTIPLE PATHS OR UCB DEFINITION INVALID',UCB&S(UNIT)
END
EXIT: REM &SYM(X1,X2,X3)
END

PROC SHARE,UNIT
*****MAKE A DIRECT ACCESS DEVICE SHARED*****
SET UCB&S(UNIT).(X'11',1)=UCB&S(UNIT).(X'11',1)|X'20'
END

PROC NONSHARE,UNIT
*****MAKE A DIRECT ACCESS DEVICE NONSHARED*****
SET UCB&S(UNIT).(X'11',1)=UCB&S(UNIT).X'11',1)&X'DF'
END

```

**This would invoke the command procedures to make unit 230 shared:**

```

INVOKE DEFUCB,230
INVOKE SHARE,230

```

**This would invoke the command procedures to make unit 230 nonshared:**

```

INVOKE DEFUCB,230
INVOKE NONSHARE,230

```





## Section 5: Output Formats

This section describes the formatted output produced by DSS commands. It includes the following topics:

- DISPLAY Output - produced by any DISPLAY command other than 'DISPLAY dss-function'
- DUMP Output - produced by any DUMP command other than 'DUMP dss-function'
- SNAP Output - produced by the SNAP operand of an AT or ON command, whenever the monitoring interrupt occurs
- Output Produced for DSS Functions - produced by 'DISPLAY dss-function' or 'DUMP dss-function'
- Error Dump - produced when DSS detects a DSS error and dumps itself

DSS messages are described in "Section 6: Messages."

### DISPLAY Output

The format of a displayed line is:

prefix      data

prefix

The content of the prefix depends on the type of DISPLAY command that was issued. See Figure 37.

| Command                                                                                        | Prefix                                                                                                                                                                                                                  |
|------------------------------------------------------------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| DISPLAY      identifier                                                                        | An integer that indicates the relative offset, from the beginning of the data field, of the first byte of data on the line. In addition, for the first line, the name of the integer is printed above the offset value. |
| DISPLAY      { identifier%<br>address-<br>literal      [%]<br>&RM.address<br>literal }         | The hexadecimal address of the first byte of data on the line.                                                                                                                                                          |
| DISPLAY      { expression<br>decimal-literal<br>hexadecimal-<br>literal<br>character-literal } | None.                                                                                                                                                                                                                   |

**Figure 37. DISPLAY Line Prefixes**

data

The data is formatted according to the type attribute of the data field. The formats are:

- X (hexadecimal)

A hexadecimal representation of from one to four words of storage, followed by the EBCDIC translation (a continuous character string, with a period inserted in any position for which there is no printable EBCDIC graphic):

hex   hex   hex   hex   character-string

- I (decimal integer)

An algebraic representation of from one to four words of storage:

± 10 digits ± 10 digits ± 10 digits ± 10 digits

A one- to four-byte data field is displayed as a signed ten-digit decimal integer.

Data fields longer than four bytes are split into fullwords on fullword boundaries; each fullword is displayed as a signed ten-digit decimal integer. Remainders from data field splitting, one to three bytes in length, are displayed as ten-digit decimal integers. See Example 8, in the set of examples that follows.

- C (character)

A continuous character string, with a period inserted in any position for which there is no printable EBCDIC graphic:

character-string

**Notes:**

1. If more than one operand is used with a DISPLAY command, the display for each operand begins on a new line and is formatted according to the type (X, I, or C).
2. If a range is specified as the DISPLAY command operand and the range limits have different type attributes, the display is in hexadecimal (X).
3. The maximum line length is 72 characters.
4. The information is displayed at the integrated operator's console.

**Examples**

1. DISPLAY MYSYM, where type=X, could produce

```
MYSYM
00000000    00000004
```

2. DISPLAY MYSYM, where type=C, could produce

```
MYSYM
00000000    ABC
```

3. DISPLAY MYSYM+4, where type=X for MYSYM, would produce type=X output:

```
+00000008
```

4. DISPLAY &G(2)\*MYSYM+4 could produce

```
+00000020
```

5. DISPLAY 'fred is great' produces capital-letter output, since DSS translates lowercase letters to uppercase:

```
FRED IS GREAT
```

6. DISPLAY L'4006':L'4024' would produce output in this format:

```
00004006          9B15 400021A4 00000000 * . . . . .u . . . *
00004010 0000BBE8 00040000 00002288 00040000 * . . . . .Y . . . . .h . . . *
00004020 0000B9E8 00000000 * . . . . .Y . . . . . *
```

7. DISPLAY ARRAY(10:19), where length=4 for ARRAY, could produce

```
ARRAY
00000028          00000000 00000000 * . . . . . *
00000030 00000000 44C1F0F0 00000000 00000000 * . . . . .A00 . . . . . *
00000040 TO 0000004F ALL CONTAIN 00000000
```

8. FIELD is this four-word data field:

```
02000001 00000004 00000005 00060000
```

DISPLAY FIELD.(1,13,I) would display FIELD as

```
FIELD
00000001 +0000000001 +0000000004 +0000000005 +0000000006
```

## DUMP Output

The output from a DUMP command with the &PRDMP[RM] operand must be to an unlabeled tape (TOUT1 or TOUT2). This tape can be formatted and printed, when OS/VS2 is running, by the AMDPRDMP service aid program. The format of the AMDPRDMP output varies, depending on the control statements that you use to invoke AMDPRDMP. The chapter on PRDMP in OS/VS2 Service Aids, GC28-0633, describes the JCL statements, the user control statements, and the output formats for AMDPRDMP listings of DSS dumps.

The output from a DUMP command without the &PRDMP[RM] operand is identical to that for the DISPLAY command, except that:

- Each line contains twice as much data: that is, eight eight-digit words of hexadecimal data, an algebraic sign followed by six ten-digit words of decimal integer data, or twice as many characters.
- The maximum line length is 121 bytes, where the first byte is a forms-motion control character. The remaining 120 characters contain the formatted data.
- Each DUMP command causes a new dump to be initiated, starting a new page with the header STORAGE PRINT and an indication whether virtual or real storage is being dumped. A line is skipped following the header and then the subheading, if specified via the DSS function &HDR, is printed and another line is skipped. The headings are repeated at the top of each page for that dump.
- The output device is determined by the DSS function &DOUT; therefore, &DOUT must be set to a valid DSS I/O name before you issue the DUMP command. The valid settings are PRINT1, PRINT2, TOUT1, TOUT2, and LOG. (In addition, CMDT and PUNCH are valid for command procedures created by the PROCEDURE command. See "&PROC," under "Output Produced for DSS Functions," later in this section.)

## Examples

1. DUMP MYSYM.(8,5), where MYSYM was defined by a DSS command, would produce output in this format:

```

***** DSS VM STORAGE PRINT *****
***** PAGE 001*****
MYSYM1
00000008                23456789 AB                *      . . .      *

```

2. DUMP L'5006':L'5060' would produce output in this format:

```

***** DSS VM STORAGE PRINT *****
***** PAGE 001*****
&SVM.IEANUC01.XCP064
00005006                921B 8978D200 A0003020 47F05710 91042001 47805702 58A07030 *      . . .K. . . . .0. . . . . *
00005020 50AC8980 92088980 47F0527E 91017006 471053A8 58F088A8 45E0F014 47F0533A *8. . . . .0.= . . . . .0..0..0. *
00005040 47F088BE 91017006 471053A8 58F7003C 587F005C 58F7003C 18C758AF 006058FA *.0 . . . . .7. . . .7. . .G. . . . *
00005060 003C1B77                * . . . . *

```

## SNAP Output

There are two basic types of SNAP output:

- Type 1, for ATs, ONs (&B, &I, &RA, &SA)
- Type 2, for ONs (&LD, &UNLD)

### Type 1—ATs, ONs (&B, &I, &RA, &SA)

```

SNAP AC=xx &CPUID=000x &ASID=xxxx ADDR=xxxxxxx
OPSW=xxxxxxxxxxxxxxxx INST=xxxxx etc.
GPR= 00 xxxxxxxx xxxxxxxx xxxxxxxx etc...
GPR= 08 xxxxxxxx xxxxxxxx xxxxxxxx etc...

```

where x = a hexadecimal digit

AC – two hexadecimal digits to indicate the event(s) that caused the SNAP record to be taken:

| Code | Event                                     |
|------|-------------------------------------------|
| 20   | Instruction fetch (ON &I)                 |
| 10   | Storage alteration (ON &SA)               |
| 08   | Successful branch instruction (ON &B)     |
| 04   | General register alteration (ON &RA)      |
| 02   | AT encountered                            |
| 01   | Asynchronous event (RESTART key or error) |

**Note:** When multiple events occur, the priority for processing their associated command procedures is the same as the order of the above list.

**&CPUID** — the number of the CPU in which the event occurred

**&ASID** — the number of the current OS/V2 address space for the CPU in which the event occurred

**ADDR** — the address of the instruction that caused the interrupt; this instruction was executed

**PSW** — the resume PSW

**INST** — four to twelve hexadecimal digits to indicate the instruction causing the interrupt

**GPR=00** — a display of general registers 0-7

**GPR=08** — a display of general registers 8-15

#### Example

```
SNAP AC=02 ADDR=001F4784 OPSW=00040005C01F4798 INST=07FE
GPR=00 00000000 001FR860 001F4800 FFFFFFFF F1F2F3F4 F5F6F7F8 C1C2C3C4 004789AB
GPR=08 FFFFFFFD9 0000147A 00000000 0ABCDEF0 001F4752 001F4954 001F4798 001F4750
```

## Type 2—ONs (&LD, &UNLD)

```
SNAP AC=xx &CPUID=000x &ASID=xxxx NAME=ccccccc
ADDR=xxxxxxxx LGN=xxxxxx
```

where x = a hexadecimal digit  
c = a character

**AC** — two hexadecimal digits to indicate the event(s) that caused the SNAP record to be taken.

| Code | Event                       |
|------|-----------------------------|
| 80   | Module loaded (ON & LD)     |
| 40   | Module unloaded (ON & UNLD) |

**&CPUID** — the number of the CPU in which the event occurred

**&ASID** — the number of the current OS/V2 address space for the CPU in which the event occurred

**NAME** — the name of the load module

**ADDR** — the beginning address of the named load module

**LGN** — the length of the load module

#### Example:

```
SNAP AC=80 NAME=TESTPGM1 ADDR=00AF4880 LGN=0001FA
```

## Output Produced for DSS Functions

When a DSS function is specified as an operand of a DISPLAY or DUMP command, the output appears as one or more lines of 72 bytes for the console, 121 bytes for printer or magnetic tape, or 80 bytes to obtain a copy of the PROCEDURE command and its associated command procedure in card-image format for a card punch or magnetic tape.

The output formats for the DSS functions are further described in the remainder of this appendix.

**Note:** If an expression containing a DSS function is specified as the operand of DISPLAY or DUMP, the output format is as described earlier in this appendix, under "DISPLAY Output" and "DUMP Output".

## &AT

```
&AT NO.=dddd NAME=ccccccc SNAP=ccc STATE=cccc  
QUAL=cccccccccccccccccccccc  
ADDR=xxxxxxx LM=ccccccc  
TXT=command-procedure
```

where d = a decimal digit  
x = a hexadecimal digit  
c = a character

NO. — the number that DSS assigned to this AT command

NAME — the name, if any, that you gave to this AT command

SNAP — a YES or NO that indicates whether or not a SNAP is to be taken

STATE — ENBL or DSBL, to indicate whether the AT is enabled or disabled

QUAL — the &QT qualification that will be applied to the command procedure when it is executed

ADDR — the address of the logical breakpoint

LM — the fully qualified name of the load module in which the breakpoint is planted, or, if the name is unavailable, the character string **\*\*NOT IN UNLOADABLE MODULE\*\***

TXT — the procedure stored by the AT command

### Notes:

1. A separate "ADDR=xxxxxxx LM=ccccccc" is written for each breakpoint that this AT command planted.
2. If &AT is specified without parentheses containing names or numbers, then all ATs are displayed. If the unqualified &AT is an operand of a DISPLAY command, the associated command text is not displayed. If you want to get the command text at the console, use the form &AT( ) with the numbers or names.

### Example

```
&AT NO.=0004 NAME=MYAT1 SNAP=YES STATE=ENBL  
QUAL=&SVM.TEST1  
ADDR=0001AF00 LM=&SVM.TEST1  
TXT=DISPLAY &G(7:10)  
SET &G(1)=X'00001AFO'  
DIVERT  
END
```

## &ON

```
&ON NO.=dddd NAME=ccccccc SNAP=ccc STATE=cccc  
QUAL=cccccccccccccc  
EVENTS=ccccccc etc.  
TXT=command-procedure
```

where d = a decimal digit  
c = a character

NO. — the number that DSS assigned to this ON command.

NAME — the name, if any, that you gave to this ON command.

SNAP — a YES or NO that indicates whether or not a SNAP is to be taken

STATE — ENBL or DISBL, to indicate whether the ON is enabled or disabled

QUAL — the &QT qualification that will be applied to the command procedure when it is executed.

EVENTS — the events to be monitored for by this ON command

TXT — the command procedure stored by the ON command

**Note:** If &ON is specified without parentheses containing names or numbers, then all ONs are displayed. If the unqualified &ON is an operand of a DISPLAY command, the associated command procedure is not displayed. If you want to get the command procedure at the console, use the form &ON ( ), with the numbers or names.

### Example

```
&ON NO.0001 NAME=MYON1 SNAP=YES STATE=ENBL  
QUAL= &SVM.TEST1  
EVENTS= &R  
TXT=END
```

## &PATCH

```
&PATCH NO.=dddd NAME=ccccccc  
ORIGINAL DATA LM=ccccccc  
location value value value value  
NEW DATA  
address value value value value
```

where d = a decimal digit  
c = a character

NO. — the number that DSS assigned to this PATCH command

NAME — the name, if any, that you gave to this PATCH command

LM — the fully qualified name of the patched load module, or, if the name is unavailable, the character string **\*\*NOT IN UNLOADABLE MODULE\*\***

location — eight hexadecimal digits that indicate the offset from the beginning of the module in OS/VS1, or the base address in OS/VS2 (in OS/VS2, location and address are the same)

address — eight hexadecimal digits that indicate the base address

value — eight hexadecimal digits; the series of values shows the original data and the new data

**Note:** A series of lines consisting of “address value value value value” may be required under both ORIGINAL DATA and NEW DATA to show these areas.

#### Example

```
&PATCH NO.=0022 NAME=MYPATCH1
ORIGINAL DATA LM=MYJOB.TEST47
00000000 0000001F C1C2C3C4 F1F2F3F4 FFFFFFFF
00000010 D1D2D3D4 001FF928
NEW DATA
00034F90 000000F4 C1C2C3C4 00000000 00000000
00034FA0 40404040 40404040
```

## &PROC

```
&PROC NAME=cccccccc PARM=cccccccc etc.
TXT= command procedure
```

where c = a character

NAME — the name of this command procedure

PARM — your parameter specification for this command procedure

TXT — the command procedure

Each of these commands produces different output for &PROC:

- DUMP &PROC(), when &DOUT is set to LOG, PRINT1, PRINT2, TOUT1, or TOUT2
- DUMP &PROC(), when &DOUT is set to CMDT or PUNCH
- DISPLAY &PROC()

#### Dump of &PROC to LOG, PRINT1, PRINT2, TOUT1, or TOUT2

The dump is in the format shown above; line length is 120 bytes. Where no parameters were specified, PARM= is blank.

#### Dump of &PROC to CMDT or PUNCH

The dump is in card-image format, that is, 80-character logical records that each contain a command. &PROC, NAME=, PARM=, and TXT= are omitted; only the commands themselves are dumped.



## Display of &PROC

The display is in the format shown above, except that if &PROC is unqualified, the texts of the command procedures are not displayed. (If you want to display the text of a command procedure at the console, use the form &PROC(name[,...]).) In addition, if no parameters were specified, PARM= is blank.

**Note:** If &PROC is specified without parentheses containing command-procedure names, all command procedures that PROCEDURE stored during the current DSS session are described.

### Example

```
&PROC NAME=MYPROC      PARM=
      TXT=DISPLAY &G(0:1)
      IF &G(0)=X'800049FF'
      SET MSG='OK'
      RETURN
      END
      SET MSG='NOGOOD'
      RETURN
      END
```

## &SYM

|                                                                                                                                    |
|------------------------------------------------------------------------------------------------------------------------------------|
| <pre>&amp;SYM NAME=cccccccc REF=ccccccc OFFSET=xxxxxxxx LNG=xxxxxxxx       TYPE=c SIZE=xxxxxxxx SCOPE=cccccccc QUAL=cccccccc</pre> |
|------------------------------------------------------------------------------------------------------------------------------------|

where c = a character  
x = a hexadecimal digit

NAME — the data field's name

REF — a character string that indicates whether the data field is defined, equated, or a procedure parameter

OFFSET — the offset

LNG — the length

TYPE — the type (C = character, I = decimal integer, X = hexadecimal)

SIZE — the size

SCOPE — INTERNAL or EXTERNAL, to indicate whether the name is internal or external

QUAL — for external symbols only, the fully qualified name of the load module

### Example

```
&SYM NAME=WORKAREA REF=DEFINED OFFSET=00000000 LNG=00000004
      TYPE=I SIZE=00000004 SCOPE=INTERNAL
```

**&PREFIX, &EPSW, &EPSWN, &IPSW, &IPSWN, &PPSW, &PPSWN, &SPSW, &SPSWN, &MPSW, &MPSWN, &RPSW, &RPSWN, &PSW, &CAW, &CID, &CSW, &IOEL, &MC, &PRM, &RANGE, &TEA, &ASID, &ASID (job-name), &CPUID, &AC, &ADDR, &M**

**&name value**

**&name** — the symbol to which the value applies

**value** — the value in hexadecimal format

**Examples**

**&RANGE 0001 0000AF00 0000AFB0**

**where the first four digits are the &ASID**

**&CAW 001DB0A8**

**&PRM 0472**

**For the output format of &ASID(number).operand or &CPUID(number).operand, see this section's output-format description for the operand.**

**&DOUT, &QT, &SOUT, &TOD**

**&name value**

**&name** — the name of the DSS function

**value** — a character string that indicates the value of the DSS function

**Example**

**&DOUT PRINT1**

**&HDR**

**&HDR  
subheading**

**subheading** — the character string that was stored in &HDR by the last SET &HDR command.

**Example**

**&HDR  
AREA ABC ON EXIT FROM XYZ**

**&ID**

**&ID address value**

**address** — eight hexadecimal digits that indicate the base address of the load module

value — the fully qualified symbolic address in the form:

```
{& ASID }{(number).map-id.module-name[. {csect-name }]  
{& CPUID} {entry-point-name}]
```

When the address that you specify with &ID is above X'1000', "value" is qualified by &ASID; when that address is X'1000' or below, "value" is qualified by &CPUID.

When the address that you specify with &ID does not reside in a mapped data field, "value" is the character string: \*NOT IN LOAD MODULE\*

#### Example

```
&ID 001FAB4 &ASID(4). &JOB(MYJOB).TEST1.PARTA
```

### &P, &S, &B, &I, &RA, &SA, &LD, &UNLD

These DSS functions cannot be displayed or dumped. Although &S can be specified in a DISPLAY or DUMP command, only the substituted item is displayed or dumped.

### Map Output Formats

&SVMMAP(NUC): The format is

| NUCLEUS STORAGE MAP nucleus-id |          |        |        |          |        |        |
|--------------------------------|----------|--------|--------|----------|--------|--------|
| TYPE                           | NAME     | ADDR   | LENGTH | ENTRY    | ADDR   | LENGTH |
| tt                             | nnnnnnnn | aaaaaa | llll   | nnnnnnnn | aaaaaa | llll   |

nucleus-id — Eight-character data field from the CVT.

tt — Type code —CS for a control section  
— blanks for an entry point.

nnnnnnnn — eight characters that indicate the control section or entry point name.

aaaaaa — six hexadecimal digits that indicate the address of the control section or entry point.

llll — four hexadecimal digits that indicate the length attribute of the control section or entry point.

The DISPLAY command writes two name-addr-length groups on each line; the DUMP command writes four name-addr-length groups on each line.

The map is ordered by address, so each control section entry in the map is followed by its entry points. The beginning of each control section entry is identified by a CS in the TYPE field.

**Example**

Following is the beginning of a printout that could be produced by DUMP &SVM MAP(NUC).

```

***** DSS  &SVM MAP PRINT ***** PAGE 001*****
                                NUCLEUS STORAGE MAP IEANUC01
TYPE NAME      ADDR    LENGTH  ENTRY    ADDR    LENGTH  ENTRY    ADDR    LENGTH  ENTRY    ADDR    LENGTH
CS  IEAQFX00   000000  FD78     SVCOPSW 000020  FD58     PIOPSW   000028  FD50     PINPSW   000068  FD10
      IEAPSW   000200  FB78     IECIERLC 00020A  FB6E     FLCAEQJ 000210  FB68
      FLCAEQS 000218  FB60     IEASCSAV 000220  FB58     IEAPKSAV 000260  FB18
      IEASAV   0002C8  FAB0     IQAPFXSV 000914  F464     IQAPFX00 000914  F464
      IQAPFXPC 00091C  F45C     IQAPFXLP 000930  F448     IQAPFXAT 000936  F442
      IECIOS   000940  F438     IECIOQET 004B5C  B21C     IECITSAR 00564C  A72C
      DRRRQE   005650  A728     IGC000   005698  A6E0     IGC114   0056A0  A6D8
      XCP064   005966  A412     IECHK4   005CA6  A0D2     IECHK1   00603C  9D3C
      IECINT   0061FC  987C     INT025   006464  9914     IECHK2   0065D2  97A6
      IECHK5   006940  9438     IECHK6   006942  9436     IGC015   006AF8  9280
      IFCCPL00 0069CE  8FAA     IGC092   006F40  8E38     IECIXAVL 006F94  8DE4
      IECOMS   007024  8D54     IECXAPG  00704C  8D2C     IECOLTVT 00707C  8CFC
      DDRAPNVT 0070A4  8CD4     CATAPP   007224  8B54     IECPESW  007268  8B10
      DEVTAB   00747C  88FC     IECCST   0074D4  88A4     IECILCH  0074FC  887C
      IECILK1  007534  8844     IECILK2  0075BA  87BE     IEAERPV  009102  6C76
      IECRMS   00930A  6A6E     DDRSRTO  009604  6774     IECIHIO  0097E6  6592
      IFCHK13  009A10  6368     IECSTB   00F170  0C08

CS  IGFCAT     00FD78  0740     CHRADTAB 0103E8  00D0     ERPIBFD  010418  00A0     IGFCCH   0104A5  0013

CS  IEAVGM00   0104B8  1EA0     IGC004   0104B8  1EA0     IGC005   0104B8  1EA0     GMBRANCH 010658  1D00
      CLBRANCH 01068A  1CCE     FMBRANCH 010690  1CC8     ABBRANCH 0106C2  1C96
      RMBRANCH 0106E8  1C70     IGC010   010710  1C48     QCBRANCH 01074E  1C0A
      SVCBYTE   010802  1B56     OBFPSW   010805  1B53     GETMAINB 01080C  1B4C
      MRELEASE  0112F8  1060     GMLPSWV  01161C  0D3C     GMLPSWR  011620  0D38
      CDPURGE   011746  0C12     FBQSRCH  0117C6  0B92     OBFSAVE  012178  01E0
      GOVRF1B   01233C  001C

CS  IEAVLK00   012358  0F88     IGC006   012358  0F88     IEAQCS02 01236C  0F74     IEAQCD5R 012456  0E8A
      DSS06     0128B6  0A2A     CDEPILOG 012968  0A78     IEAQCS03 012968  0978
      IGC007   012968  0A78     IGC008   012F98  0348
      IGC008   012F98  0348
      IGC009   013136  0114
  
```

**&SVM MAP(LPA), &SVM MAP(ALPA), &SVM MAP(RRMA), &JOB MAP(), &TCB MAP():** The format is:

| header-line |          |        |                                    |        |        |        |
|-------------|----------|--------|------------------------------------|--------|--------|--------|
| TYPE        | NAME     | EP     | EXTENTS(ADDR LENGTH) OR MAJOR NAME |        |        |        |
| tt          | nnnnnnnn | eeeeee | aaaaaa                             | llllll | aaaaaa | llllll |

header-line — A character string that depends on the map being printed.

tt — Type code — M Major and not shared  
 MS Major and shared  
 blanks Minor

nnnnnnnn — eight characters that indicate the load module or entry point name.

eeeeee — six hexadecimal digits that indicate the address of the entry point.

aaaaaa — six hexadecimal digits that indicate the starting address of an extent.

llllll — six hexadecimal digits that indicate the length attribute of the extent.

### Header-Line Formats:

```
&SVMMAP(LPA)
LINK PACK AREA MAP
&SVMMAP(ALPA)
ACTIVE LINK PACK AREA MAP
```

```
&SVMMAP(RRMA)
RESIDENT REENTRANT MODULE MAP
```

```
&JOBMAP( )
JOBMAP FOR nnnnnnnn AT aaaaaa
```

nnnnnnnn — eight characters that indicate the job step name.

aaaaaa — six hexadecimal digits that indicate the job step TCB address.

```
&TCBMAP( )
MAP OF TCB AT aaaaaa
```

aaaaaa — six hexadecimal digits that indicate the TCB address.

The DISPLAY command prints up to two extents on a line. The DUMP command prints up to five extents on a line. If more extents must be printed than can be fit on the line, one or more additional lines are used. The TYPE, NAME, and EP fields are set to blanks on these additional lines.

### Example

Following is the beginning of a printout that could be produced by DUMP &SVMMAP(LPA).

```
***** DSS  &SVM  MAP  PRINT *****          ***** PAGE 001*****
```

LINK PACK AREA MAP

| TYPE | NAME     | EP     | EXTENTS(ADDR LENGTH) OR MAJOR NAME |
|------|----------|--------|------------------------------------|
| MS   | IGC0A05A | ED3858 | ED3858 0007A8                      |
| MS   | IGG019V3 | DF9C60 | DF9C60 0001A0                      |
|      | IFG0239R | EE9400 | IFG0199R                           |
| MS   | IGG019V5 | DF81E0 | DF81E0 000E20                      |
| MS   | IGG01961 | DDB400 | DDB400 000400                      |
|      | IFG0239B | EEA000 | IFG0199B                           |
| MS   | IGC0308F | E98278 | E98278 000260                      |
| MS   | IEEVLOUT | FC2AA0 | FC2AA0 000148                      |
| MS   | IEFVGM2  | F80AE0 | F80AE0 000148                      |
| MS   | IGC04091 | E954D8 | E954D8 000380                      |
| MS   | IGC5207B | E778E8 | E778E8 0001D8                      |
| MS   | IGC0D05A | ED0678 | ED0678 0002E8                      |
| MS   | IGC6207B | E710D8 | E710D8 0003B0                      |
|      | IFG0239D | EE9C00 | IFG0199D                           |
|      | IFG0239E | EE9800 | IFG0199E                           |
| MS   | IGC0B06F | ED2230 | ED2230 000168                      |
| MS   | IGC0F01C | ECF6E8 | ECF6E8 000328                      |
| MS   | IGE0904G | E5F118 | E5F118 0000C0                      |
| MS   | IGG019CU | E512E0 | E512E0 0006D8                      |
|      | IFAB400  | FBA000 | FBA000 000148                      |
|      | IFAB11C  | ED1418 | ED1418 000148                      |
|      | IFAB078  | ED0678 | ED0678 000148                      |

**&O, &L, &SZ, &T**

| dss-function | name | value |
|--------------|------|-------|
|--------------|------|-------|

dss-function — a character string that indicates the dss function

name — a character string that indicates the symbol to which the value applies

value — eight hexadecimal digits or, for &T, a one-character code (X = character, I = decimal integer, X = hexadecimal).

**Examples**

```

&O ANSWER 000000A8
&SZ AREA 0000014E
&T MESSAGE C
    
```

**&G, &C, &F**

| dss-function | register # | value | value | value | value |
|--------------|------------|-------|-------|-------|-------|
|--------------|------------|-------|-------|-------|-------|

dss-function — a character string that indicates the DSS function

register # — a hexadecimal digit that indicates the initial register number on that line

value — a series of eight hexadecimal digits that indicate the present register value

**Examples:**

1. DISPLAY &G(0:5) would produce output in this format:

```

&G 00 000FABE4 000ABCDE 00000000 FFFFFFFF
    04 000000A8 000000AD
    
```

2. DISPLAY &C(2:15) would produce output in this format:

```

&C 2 FFFFFFFF FFFFFFFF
    4 00000000 00000000 00000000 00000000
    8 00000000 400007AD 00037340 000387E0
   12 00000000 00000000 E0C00000 0002F990
    
```

3. DISPLAY &G(1:4),&G(6:9) would produce output in this format:

```

&G 1 F0F0F840 00000000 00030001 00000000
&G 6 00000000 00000000 00000000 40F107AD
    
```

4. DUMP &G(2:15) would produce output in this format:

| ***** DSS VM STORAGE PRINT ***** |   |          |          |          |          |          |          |          |          | ***** PAGE 001***** |  |  |  |  |  |  |  |  |  |  |
|----------------------------------|---|----------|----------|----------|----------|----------|----------|----------|----------|---------------------|--|--|--|--|--|--|--|--|--|--|
| &G                               | 2 |          |          | 00000000 | 00000000 | 00000000 | 00000000 | 00000000 | 00000000 |                     |  |  |  |  |  |  |  |  |  |  |
|                                  | 8 | 00000000 | 00000000 | 00000000 | 00000000 | 00000000 | 00000000 | 00000000 | 00000000 |                     |  |  |  |  |  |  |  |  |  |  |

**&SVM, &JOB, &TCB, &LM, &PRB, &Q**

|                                 |
|---------------------------------|
| name                            |
| address value value value value |

name — a character string that indicates the symbol to which the values apply  
 address — eight hexadecimal digits that indicate the storage address specified  
 value — a series of eight hexadecimal digits that indicate the present value

**Example:**

```
| &JOB(MYJOB),MODULEA
00014AC8 F1F2F3F4 C1C2C3C4 00000000 F5F6F7F8
00014AD8 FFFFFFFF 000000AF D1D2D3D4 00000006
00014AE8 0000000B 4987ABCD
```

**&RM**

Output for &RM is covered under “DISPLAY Output,” earlier in this section.

**Error Dump**

- | In OS/VS1, the IQAEPR00 program prints
- 1. The dump identification that you entered in response to message IQA012A.
- 2. The contents of the general registers.
- | 3. Real storage from locations 0 to 4096.
- 4. The contents of all DSS-occupied segments from segment 5 up, in the form of one continuous storage dump.

Following is the beginning of a printout that could be produced by IQAEPR00.

```
ASGN1 ER
GPR 0 7 00012048 00000000 00050000 00000008 0000000F 00000001 00000206 8008B8F6 *.....6**
GPR 8 15 0008BCC3 00062000 0008B390 0008C390 0008C060 0006215C 5008BB3E 00054BAA *...C.....C.....*8.....**

00000000 040C0000 00019F88 070E0000 00000000 000170A0 00000000 070E0000 00000000 *.....H.....**
00000020 040C0000 00F3334A 060C0000 00F32E8A 00000000 00000000 070E0000 00000000 *...3...3.....**
00000040 0006D548 0C000000 0006D518 000170A0 FD64FFFF 0002FA54 04000000 00018C84 *..N...N.....D**
00000060 040C0000 00019604 000C0000 00018E36 04080000 0001EE48 040C0000 00018DB0 *...O.....**
00000080 00000000 00001004 0002006B 00060011 00F31000 00000000 00000000 00000000 *.....3.....**
000000A0 00000000 00000000 10000060 0002F458 FF000000 00000000 00000283 00000000 *.....4.....C.....**
000000C0 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 *.....**
      LINES 000000E0-00000140 SAME AS ABQVE
00000160 00000000 00000000 00000000 82000170 00000000 0002D798 00000000 00000000 *.....B.....PQ.....**
00000180 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 *.....**
      LINES 000001A0-000001E0 SAME AS ABOVE
00000200 00000000 00F3333C 0000020A 020A020A FF000210 00000210 FF000218 00000218 *...3.....**
00000220 00000000 00000003 00F34000 00000000 00000000 00000000 00000000 *...3...3.H...3...2J...**
```

In OS/V52, when DSS dumps itself to a printer, it displays:

1. The dump identification that you entered in response to message IQA012A.
2. The general registers.
3. The control registers.
4. The prefixed storage area (labeled FIXED LOW CORE in the dump).
5. The DSS vector table (IQADSV).
6. The DSS environment routines (IQAENV00).
7. The DSS language routines (IQALAN00), including the language work space.

Following is the beginning of a printout that could be produced by a DSS error dump in OS/V52:

```

*** DYNAMIC SUPPORT SYSTEM (DSS) ERROR DUMP FACILITY      DATE  73.122      TIME  04.09.16      PAGE  002  ***
***DSS ERROR DUMP***      *****DUMP TEST ON 40AJ*****      ***DSS ERROR DUMP***
*** S T A R T   O F   D U M P   ***

                *** G E N E R A L   R E G I S T E R S   ***
REG 0-7      00000000      0020C738      000A9714      400AEDAE      000A9000      00000000      000AF3B4      800AEE78
REG 8-15     00000009      00000040      00000400      000AA1CC      0009E000      0001D694      0001DBBC      A00AED9E

                *** C O N T R O L   R E G I S T E R S   ***
REG 0-7      00800000      0F09E140      FFFFFFFF      FFFFFFFF      00000000      00000000      00000000      00000000
REG 8-15     00000000      00000000      00000000      00000000      00000000      00000000      C0000000      00000200

                *** F I X E D   L O W   C O R E   ***
00000000      04080000      0000E65E      070F0000      00000000      0001D7F0      00000000      070F0000      00000000      *.....W.....P0.....*
00000020      070C1000      000B06C4      04082000      002286F0      00000000      00000000      070C0000      0023F3C      *.....D.....0.....*
00000040      000A30F0      04000000      000A30E8      0001D7F0      7CCA4705      00FD6760      000C0000      0005B3A8      *...0...Y...P0.....*
00000060      000C0000      0005BA00      000C0000      0005B7C8      00000000      00098624      040C0000      000A2D00      *.....H.....*
00000080      00000000      00000000      00020071      00060011      000286F0      00000000      00000000      00000000      *.....0.....*
000000A0      00000000      00000000      00000000      00000000      00000000      00000000      00000130      00000000      *.....*
000000C0      00000000      00000000      00000000      00000000      00000000      00000000      00000000      00000000      *.....*
                L I N E S   000000E0   T O   0000015F   S A M E   A S   A B O V E
00000160      00000000      00000000      00000000      8200015F      00000000      00022B58      00000000      00000000      *.....*
00000180      00000000      00000000      00000000      00000000      00000000      00000000      00000000      00000000      *.....*

```



## Section 6: Messages

DSS writes all of its messages at the integrated operator's console; in addition, if the LOG I/O name is assigned, DSS writes its messages in the hardcopy log. Most of the messages indicate either user error or system error; a few DSS messages are informational and do not imply error.

The following information is included for each DSS message:

| <p><b>Message:</b> The message identifier followed by the message text. The format of the message identifier is IQAnnt, where IQA is the operating-system component identifier for DSS; nnn is a three-digit serial number; and t is one of these codes:</p> <table border="0"> <thead> <tr> <th style="text-align: left;">Code</th> <th style="text-align: left;">Meaning</th> </tr> </thead> <tbody> <tr> <td>A</td> <td>Immediate action required</td> </tr> <tr> <td>D</td> <td>Immediate decision required</td> </tr> <tr> <td>E</td> <td>Eventual action required</td> </tr> <tr> <td>I</td> <td>Information (may provoke action)</td> </tr> <tr> <td>W</td> <td>The system is in the wait state</td> </tr> </tbody> </table> <p>Occasionally, a message is followed by one or more of these supplementary messages:</p> <table border="0"> <thead> <tr> <th style="text-align: left;">Format</th> <th style="text-align: left;">Meaning</th> </tr> </thead> <tbody> <tr> <td>CUA = xxxxxx</td> <td>Control Unit Address</td> </tr> <tr> <td>CSW = xxxxxxxx xxxxxxxx</td> <td>Channel Status Word</td> </tr> <tr> <td>SENSE = xxxx</td> <td>Sense Data</td> </tr> <tr> <td>CAW = xxxxxxxx</td> <td>Channel Address Word</td> </tr> <tr> <td>CCW = xxxxxxxx xxxxxxxx</td> <td>Failing Channel Command Word</td> </tr> </tbody> </table> | Code                             | Meaning | A | Immediate action required | D | Immediate decision required | E | Eventual action required | I | Information (may provoke action) | W | The system is in the wait state | Format | Meaning | CUA = xxxxxx | Control Unit Address | CSW = xxxxxxxx xxxxxxxx | Channel Status Word | SENSE = xxxx | Sense Data | CAW = xxxxxxxx | Channel Address Word | CCW = xxxxxxxx xxxxxxxx | Failing Channel Command Word |
|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|----------------------------------|---------|---|---------------------------|---|-----------------------------|---|--------------------------|---|----------------------------------|---|---------------------------------|--------|---------|--------------|----------------------|-------------------------|---------------------|--------------|------------|----------------|----------------------|-------------------------|------------------------------|
| Code                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                         | Meaning                          |         |   |                           |   |                             |   |                          |   |                                  |   |                                 |        |         |              |                      |                         |                     |              |            |                |                      |                         |                              |
| A                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                            | Immediate action required        |         |   |                           |   |                             |   |                          |   |                                  |   |                                 |        |         |              |                      |                         |                     |              |            |                |                      |                         |                              |
| D                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                            | Immediate decision required      |         |   |                           |   |                             |   |                          |   |                                  |   |                                 |        |         |              |                      |                         |                     |              |            |                |                      |                         |                              |
| E                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                            | Eventual action required         |         |   |                           |   |                             |   |                          |   |                                  |   |                                 |        |         |              |                      |                         |                     |              |            |                |                      |                         |                              |
| I                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                            | Information (may provoke action) |         |   |                           |   |                             |   |                          |   |                                  |   |                                 |        |         |              |                      |                         |                     |              |            |                |                      |                         |                              |
| W                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                            | The system is in the wait state  |         |   |                           |   |                             |   |                          |   |                                  |   |                                 |        |         |              |                      |                         |                     |              |            |                |                      |                         |                              |
| Format                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                       | Meaning                          |         |   |                           |   |                             |   |                          |   |                                  |   |                                 |        |         |              |                      |                         |                     |              |            |                |                      |                         |                              |
| CUA = xxxxxx                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                 | Control Unit Address             |         |   |                           |   |                             |   |                          |   |                                  |   |                                 |        |         |              |                      |                         |                     |              |            |                |                      |                         |                              |
| CSW = xxxxxxxx xxxxxxxx                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      | Channel Status Word              |         |   |                           |   |                             |   |                          |   |                                  |   |                                 |        |         |              |                      |                         |                     |              |            |                |                      |                         |                              |
| SENSE = xxxx                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                 | Sense Data                       |         |   |                           |   |                             |   |                          |   |                                  |   |                                 |        |         |              |                      |                         |                     |              |            |                |                      |                         |                              |
| CAW = xxxxxxxx                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                               | Channel Address Word             |         |   |                           |   |                             |   |                          |   |                                  |   |                                 |        |         |              |                      |                         |                     |              |            |                |                      |                         |                              |
| CCW = xxxxxxxx xxxxxxxx                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      | Failing Channel Command Word     |         |   |                           |   |                             |   |                          |   |                                  |   |                                 |        |         |              |                      |                         |                     |              |            |                |                      |                         |                              |
| <p><b>Explanation:</b> The meaning of the message text. What happened or is happening.</p>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                   |                                  |         |   |                           |   |                             |   |                          |   |                                  |   |                                 |        |         |              |                      |                         |                     |              |            |                |                      |                         |                              |
| <p><b>System Action:</b> How DSS responds to the event.</p>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |                                  |         |   |                           |   |                             |   |                          |   |                                  |   |                                 |        |         |              |                      |                         |                     |              |            |                |                      |                         |                              |
| <p><b>Programmer Response:</b> What the DSS user should normally do next.</p>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                |                                  |         |   |                           |   |                             |   |                          |   |                                  |   |                                 |        |         |              |                      |                         |                     |              |            |                |                      |                         |                              |
| <p><b>Problem Determination (if applicable):</b> What to do next if it becomes necessary to identify a failing hardware unit or program. Standard OS/VS problem determination actions are identified as items of table I. Actions unique to DSS are identified following the list of standard actions to be taken.</p>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                       |                                  |         |   |                           |   |                             |   |                          |   |                                  |   |                                 |        |         |              |                      |                         |                     |              |            |                |                      |                         |                              |

## Problem Determination

Table I. Problem Determination

|     |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                            |
|-----|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 2.  | Save the console sheet from the primary console. In systems with Multiple Console Support (MCS), save a copy of the hardcopy log.                                                                                                                                                                                                                                                                                                                                                          |
| 3.  | Save the job stream associated with the job.                                                                                                                                                                                                                                                                                                                                                                                                                                               |
| 4.  | Save the system output (SYSOUT) associated with the job.                                                                                                                                                                                                                                                                                                                                                                                                                                   |
| 11. | Execute the HMDSADMP or AMDSADMP service aid program to dump the contents of real storage and page data sets on magnetic tape.<br><br>After restarting the system, execute the appropriate function of the HMDPRDMP or AMDPRDMP service aid program to print the required portion of the dump tape produced by HMDSADMP or AMDSADMP.<br><br>Save both the tape from HMDSADMP or AMDSADMP (should further information from the tape be required) and the listing from HMDPRDMP or AMDPRDMP. |
| 16. | Save the dump.                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
| 20. | Save the control cards associated with the job.                                                                                                                                                                                                                                                                                                                                                                                                                                            |
| 22. | Save the source input associated with the job.                                                                                                                                                                                                                                                                                                                                                                                                                                             |
| 26. | Execute the IEBTPCH data set utility to<br>b. print the applicable data set.                                                                                                                                                                                                                                                                                                                                                                                                               |
| 29. | Contact IBM for programming support.                                                                                                                                                                                                                                                                                                                                                                                                                                                       |
| 30. | Contact IBM for hardware support.                                                                                                                                                                                                                                                                                                                                                                                                                                                          |

Figure 38. Problem Determination

**Note:** The discontinuities in the numbering of the Table I entries are caused by the fact that this table is a subset of the standard OS/VS Problem Determination table.

IQA000D

<severity> DSS PROGRAM CHECK, LOC.ID. xx IQAPIH00 IQAxxxxx.  
{ CANNOT TAKE DSS DUMP, CONTINUE? REPLY Y OR N. }  
{ DSS DUMP DESIRED? REPLY Y OR N. }

*Explanation:* A program interrupt (0-15) was encountered in the second module. For more information, refer to "System Errors," in Section 4.

*System Action:* DSS issues message IQA025D and waits for the programmer's response. A press of the END key with no preceding Y or N is treated as a Y.

*Programmer Response (OS/VS1):*

1. Cannot take DSS dump, continue? Reply Y or N
  - a. Y (yes)—Processing resumes as described under "Resumption of Processing," in Section 4.
  - b. N (no)—The system enters a disabled wait state. The system wait code is 0F1.
2. DSS dump desired? Reply Y or N
  - a. Y (yes)—Real storage from locations 0 through 4096 and DSS-occupied segments from segment 5 on up are written to SYS1.DSSVM for later printing by IQAEPR00. Processing resumes as described under "Resumption Of Processing," in Section 4.
  - b. N (no)—Processing resumes as described under "Resumption Of Processing," in Section 4.

*Programmer Response (OS/VS2):* See the programmer responses for the messages that follow this message at the console.

*Problem Determination (OS/VS1):*

After programmer response 1a or 2b—

Table I, item 2 and, if not already done, item 29.

After programmer response 1b—

Table I, item 2, item 11 while DSS is loaded, and, if not already done, item 29.

After programmer response 2a—

Table I, item 2 and, if not already done, item 29. In addition, print the DSS dump by means of the procedure given under "How To Obtain A Dump," in Section 4.

*Problem Determination (OS/VS2):* DSS software error. Reply Y to message IQA025D. Table I, items 2 and 16.

IQA001D

<severity> SVC IN DSS, LOC.ID. xx IQASVC00 IQAxxxxx.  
{ CANNOT TAKE DSS DUMP, CONTINUE? REPLY Y OR N. }  
{ DSS DUMP DESIRED? REPLY Y OR N. }

*Explanation:* An SVC instruction has been issued by the second module. This is a program error, since it is not the one case in which a DSS module is permitted to issue an SVC. For more information, see "System Errors," in Section 4.

*System Action:* The same as for message IQA000D.

*Programmer Response:* The same as for message IQA000D.

*Problem Determination:* The same as for message IQA000D.

IQA002D

<severity> DSS PAGE TABLE UNAVAILABLE, LOC.ID. xx IQAxxxxx IQAxxxxx.  
{ CANNOT TAKE DSS DUMP, CONTINUE? REPLY Y OR N. }  
{ DSS DUMP DESIRED? REPLY Y OR N. }

*Explanation:* A page exception occurred for a DSS page table entry marked unavailable. For more information, see "System Errors," in Section 4.

*System Action:* The same as for message IQA000D.

*Programmer Response:* The same as for message IQA000D.

*Problem Determination:* The same as for message IQA000D.

IQA003D <severity> DSS PAGE UNASSIGNED, LOC.ID. xx IQAxxxxx IQAxxxxx.

{ CANNOT TAKE DSS DUMP, CONTINUE? REPLY Y OR N. }  
{ DSS DUMP DESIRED? REPLY Y OR N. }

*Explanation:* The second module tried to refer to a DSS page entry that is marked unassigned. For more information, see "System Errors," in Section 4.

*System Action:* The same as for message IQA000D.

*Programmer Response:* The same as for message IQA000D.

*Problem Determination:* The same as for message IQA000D.

IQA004D <severity> INVALID DSS PAGE FAULT, LOC.ID. xx IQAxxxxx IQAxxxxx.

{ CANNOT TAKE DSS DUMP, CONTINUE? REPLY Y OR N. }  
{ DSS DUMP DESIRED? REPLY Y OR N. }

*Explanation:* The address of the page fault is not in a DSS-occupied segment. For more information, see "System Errors," in Section 4.

*System Action:* The same as for message IQA000D.

*Programmer Response:* The same as for message IQA000D.

*Problem Determination:* The same as for message IQA000D.

IQA005D <severity> UNRECOVERABLE DSS PAGING I/O ERROR, LOC.ID. xx IQAxxxxx IQAxxxxx.

{ CANNOT TAKE DSS DUMP, CONTINUE? REPLY Y OR N. }  
{ DSS DUMP DESIRED? REPLY Y OR N. }

*Explanation:* An unrecoverable I/O error has occurred on the DSS paging device. For more information, see "System Errors," in Section 4.

*System Action:* The same as for message IQA000D.

*Programmer Response:* The same as for message IQA000D.

*Problem Determination (OS/VS1):*

After programmer response 1a or 2b—

Table I, items 2 and 30.

After programmer response 1b—

Table I, item 2, item 11 while DSS is loaded, and item 30.

After programmer response 2a—

Table I, items 2 and 30. In addition, print the DSS dump by means of the procedure given under "How To Obtain A Dump," in Section 4.

*Problem Determination (OS/VS2):* Probable hardware error. Reply Y to message IQA025D. Table I, items 2, 16 and 30.

IQA006D <severity> INVALID RETURN CODE, LOC.ID. xx IQAxxxxx IQAxxxxx.

{ CANNOT TAKE DSS DUMP, CONTINUE? REPLY Y OR N. }  
{ DSS DUMP DESIRED? REPLY Y OR N. }

*Explanation:* At xx, the first module received an undefined return code from the second module. (In the prologue to the first module's listing, there is a directory that relates xx to a location in the first module.)

*System Action:* The same as for message IQA000D.

*Programmer Response:* The same as for message IQA000D.

*Problem Determination:* The same as for message IQA000D.

IQA007D <severity> CHANNEL ERROR INT xxxx CSW xxxx, LOC.ID. xx IQAxxxxx IQAxxxxx.  
{ CANNOT TAKE DSS DUMP, CONTINUE? REPLY Y OR N. }  
{ DSS DUMP DESIRED? REPLY Y OR N. }

*Explanation:* DSS encountered a channel error on the channel identified by the I/O address in the interrupt code (INT), and with the indicated CSW status. For further information, refer to "System Errors," in Section 4.

*System Action:* Before issuing this message, DSS does the following:

- If the bad channel status reflects an OS/VS I/O operation, the status information is saved for later return to OS/VS for recovery processing.
- If the bad channel status reflects a DSS I/O operation, the operation is retried. If this does not clear the error, the DSS command requiring the use of the channel is canceled and this message is issued; however, if the error prevents use of the console, and there is no device assignment for the LOG I/O name, this message is not issued.

After issuing this message, DSS issues message IQA025D and waits for the programmer's response. A press of the END key with no preceding Y or N is treated as a Y.

*Programmer Response:* The same as for message IQA000D.

*Problem Determination:* The same as for message IQA005D.

IQA008D <severity> DSS INTERNAL LOGIC ERROR, LOC.ID. xx IQAxxxxx IQAxxxxx.  
{ CANNOT TAKE DSS DUMP, CONTINUE? REPLY Y OR N. }  
{ DSS DUMP DESIRED? REPLY Y OR N. }

*Explanation:* At xx, the first module detected a logic error in the second module. (In the prologue to the first module's listing, there is a directory that relates xx to a location in the first module.)

*System Action:* The same as for message IQA000D.

*Programmer Response:* The same as for message IQA000D.

*Problem Determination:* The same as for message IQA000D.

IQA009D <severity> CONSOLE I/O ERROR, LOC.ID. xx IQAxxxxx IQAxxxxx.  
UNRECOVERABLE CONSOLE I/O ERROR, xx IQAxxxxx IQAxxxxx.  
{ CANNOT TAKE DSS DUMP, CONTINUE? REPLY Y OR N. }  
{ DSS DUMP DESIRED? REPLY Y OR N. }

*Explanation:* An unrecoverable I/O error occurred on the DSS console. For more information, refer to "System Errors," in Section 4.

*System Action:* The same as for message IQA000D.

*Programmer Response:* The same as for message IQA000D.

*Problem Determination:* The same as for message IQA005D.

\*IQA010W DSS TRANSLATION SPECIFICATION EXCEPTION.

*Explanation:* DSS encountered a translation specification exception while operating under DSS virtual storage.

*System Action:* The system goes into a wait state; the system wait code is OFA.

*Programmer Response:* None.

*Problem Determination:* Table I, item 11 while DSS is loaded, and, if not already done, item 29.

IQA011D

**<severity> LOGICAL PAGING ERROR, LOC.ID: xx IQAxxxxx IQAxxxxx.**

**{ CANNOT TAKE DSS DUMP, CONTINUE? REPLY Y OR N. }  
{ DSS DUMP DESIRED? REPLY Y OR N. }**

*Explanation:* There is a logic error in the DSS paging supervisor. As a result, there are not enough real page frames to bring in a page, or there are not enough page slots on the disk to write a page. For further information, refer to "System Errors," in Section 4.

*System Action:* The same as for message IQA000D.

*Programmer Response (OS/VS1):*

1. Cannot take DSS dump, continue? Reply Y or N
  - a. Y (yes)—DSS returns control to OS/VS as if a DISCONNECT command had been entered. The RESTART key can be used to reactivate DSS.
  - b. N (no)—The system enters a disabled wait state. The system wait code is OF1.
2. DSS dump desired? Reply Y or N
  - a. Y (yes)—Real storage from locations 0 through 4096 and DSS-occupied segments from segment 5 on up are written to SYS1.DSSVM for later printing by IQAEPR00. DSS then returns control to OS/VS as if a DISCONNECT command had been entered; the RESTART key can be used to reactivate DSS.
  - b. N (no)—DSS returns control to OS/VS as if a DISCONNECT command had been entered; the RESTART key can be used to reactivate DSS.

*Programmer Response (OS/VS2):* See the programmer responses for the messages that follow this message at the console.

*Problem Determination:* The same as for message IQA000D.

IQA012A

**ENTER DUMP IDENTIFICATION (8 CHARACTERS)**

*Explanation:* This message allows the programmer to enter an identification for a dump that he requested. The identification will be printed when the dump is printed.

*System Action:* DSS waits for the programmer's response.

*Programmer Response:* Enter up to eight characters, then press the END key.

IQA013E

**RUN DSS ERROR DUMP UTILITY**

*Explanation:* The requested DSS dump was successful; the DSS error print utility (IQAEPR00) must be run under OS/VS1 to print the DSS page-formatted data set.

*System Action:* None.

*Programmer Response:* The DSS error print utility must eventually be run according to the directions given under "How To Obtain A Dump," in Section 4.

IQA014I

**DSS DISCONNECT IN PROCESS, RETURNING TO SYSTEM.**

*Explanation:* The DSS error processor has encountered a severe error and is in the process of returning to OS/VS as if a DISCONNECT command had been entered.

*System Action:* OS/VS will run with DSS terminated.

*Programmer Response:* If more DSS processing is necessary, restart DSS. Avoid the command that caused the error.

*Problem Determination:* As specified for the first message issued regarding the error.

**IQA015I DSS SYSTEM DAMAGE. REINSTATEMENT IN PROGRESS.**

*Explanation:* The DSS error processor encountered a local error and is now reinitializing DSS as if the programmer had entered a DISCONNECT command and then restarted DSS.

*System Action:* DSS prints DSS READY, then prints & and waits for the programmer to enter a command.

*Programmer Response:* Start a new DSS session. Avoid the command that caused the error.

*Problem Determination:* As specified for the first message issued regarding the error.

**\*IQA016W DSS ERROR. SYSTEM WAIT STATE.**

*Explanation:* The DSS error processor encountered a catastrophic or **major** error, and the system has been placed in a disabled wait state. The system wait code is 0F1.

*System Action:* OS/VS attempts to write a record in the SYS1.LOGREC data set, places the system wait code in the PSW, and enters a disabled wait state.

*Programmer Response:* None.

*Problem Determination:* Table I, items 2 and (if not already done) 29 or 30. In addition, if a DSS dump was taken, follow the procedure given under "How To Obtain A Dump," in Section 4, to print the DSS page-formatted data set; if a DSS dump was not taken, Table I, item 11.

**IQA017D <severity> DSS ERROR ENCOUNTERED DURING DUMP PROCESSING. LOC.ID. xx  
IQAXxxxx IQAXxxxx.  
[CANNOT TAKE DSS DUMP, CONTINUE? REPLY Y OR N.]**

*Explanation:* While the DSS error dump was being written, an error prevented continuation of the dump.

*System Action:* DSS issues message IQA024D and waits for the programmer's response. A press of the END key with no preceding Y or N is treated as a Y.

*Programmer Response (OS/VS1):*

Y (yes)—Processing resumes as described under "Resumption of Processing," in Section 4.

N (no)—The system enters a disabled wait state. The system wait code is 0F1.

*Programmer Response (OS/VS2):* See the programmer responses for the messages that follow this message at the console.

*Problem Determination:* Table I, item 2, item 11 while DSS is loaded, and, if not already done, item 29.

**IQA018D <severity> I/O ERROR ENCOUNTERED WHILE SWAPPING OUT NUCLEUS, LOC.ID.  
xx IQAXxxxx IQAXxxxx.**

**{ CANNOT TAKE DSS DUMP, CONTINUE? REPLY Y OR N. }  
{ DSS DUMP DESIRED? REPLY Y OR N. }**

*Explanation:* During DSS initialization, while DSS was swapping out the OS/VS nucleus, an unrecoverable I/O error prevented continuation of the swapping process. For further information, refer to "System Errors," in Section 4.

*System Action:* Before issuing this message, DSS does the following:

- If the error was caused by an OS/VS I/O operation, DSS saves the status information for OS/VS recovery processing.
- If the error was caused by a DSS I/O operation, DSS retries the operation 10 times.

After issuing this message and message IQA025D, DSS waits for the programmer's response. A press of the END key with no preceding Y or N is treated as a Y.

*Programmer Response (OS/VS1):*

1. Cannot take DSS dump, continue? Reply Y or N
  - a. Y (yes)—DSS returns control to OS/VS at the point of interruption.
  - b. N (no)—The system enters a disabled wait state. The system wait code is OF1.
2. DSS dump desired? Reply Y or N
  - a. Y (yes)—Real storage from locations 0 through 4096 and DSS-occupied segments from segment 5 on up are written to SYS1.DSSVM for later printing by IQAEPR00. DSS then returns control to OS/VS at the point of interruption.
  - b. N (no)—DSS returns control to OS/VS at the point of interruption.

*Programmer Response (OS/VS2):* See the programmer responses for the messages that follow this message at the console.

*Problem Determination:* The same as for message IQA005D.

IQA019D

**<severity> NOT ENOUGH PAGE FRAMES FOR DSS ACTIVATION, LOC.ID: xx  
IQAxxxxx IQAxxxxx.**

**{ CANNOT TAKE DSS DUMP, CONTINUE? REPLY Y OR N. }  
{ DSS DUMP DESIRED? REPLY Y OR N. }**

*Explanation:* DSS cannot be activated, because there are not enough page frames. For further information, refer to "System Errors," in Section 4.

*System Action:* The same as for message IQA000D.

*Programmer Response:* The same as for message IQA018D.

*Problem Determination:* The same as for message IQA000D.

IQA020D

**<severity> I/O ERROR ENCOUNTERED WHILE SWAPPING IN DSS, LOC.ID: xx  
IQAxxxxx IQAxxxxx**

**{ CANNOT TAKE DSS DUMP, CONTINUE? REPLY Y OR N. }  
{ DSS DUMP DESIRED? REPLY Y OR N. }**

*Explanation:* During DSS initialization, an I/O error prevented the swapping in of DSS. For further information, refer to "System Errors," in Section 4.

*System Action:* Before issuing this message, DSS retries the operation unsuccessfully. After issuing this message and message IQA025D, DSS waits for the programmer's response. A press of the END key with no preceding Y or N is treated as Y.

*Programmer Response (OS/VS1):*

1. Cannot take DSS dump, continue? Reply Y or N
  - a. Y (yes)—DSS attempts to reinstate the OS/VS nucleus and return control to OS/VS at the point of interruption.
  - b. N (no)—The system enters a disabled wait state. The system wait code is OF1.
2. DSS dump desired? Reply Y or N
  - a. Y (yes)—Real storage from locations 0 through 4096 and DSS-occupied segments from segment 5 on up are written to SYS1.DSSVM for later printing by IQAEPR00. DSS then attempts to reinstate the OS/VS nucleus and return control to OS/VS at the point of interruption.
  - b. N (no)—DSS attempts to reinstate the OS/VS nucleus and return control to OS/VS at the point of interruption.

*Programmer Response (OS/VS2):* See the programmer responses for the messages that follow this message at the console.

*Problem Determination:* The same as for message IQA005D.



IQA021D

<severity> I/O ERROR ENCOUNTERED WHILE SWAPPING IN NUCLEUS, LOC.ID: xx  
IQAxxxxx IQAxxxxx

{ CANNOT TAKE DSS DUMP, CONTINUE? REPLY Y OR N. }  
{ DSS DUMP DESIRED? REPLY Y OR N. }

*Explanation:* During DSS exit processing, an unrecoverable I/O error prevented the nucleus from being swapped back in. For further information, refer to "System Errors," in Section 4.

*System Action:* Before printing this message, DSS retries the operation unsuccessfully. After issuing this message and message IQA025D, DSS waits for the programmer's response. A press of the END key with no preceding Y or N is treated as a Y.

*Programmer Response (OS/VS1):*

1. Cannot take DSS dump, continue? Reply Y or N
  - a. Y (yes)—The system enters a disabled wait state.
  - b. N (no)—The system enters a disabled wait state.
2. DSS dump desired? Reply Y or N
  - a. Y (yes)—Real storage from location 0 through 4096 and DSS-occupied segments from segment 5 on up are written to SYS1.DSSVM for later printing by IQAEP00. The system then enters a disabled wait state.
  - b. N (no)—The system enters a disabled wait state. The system wait code is OF1.

*Programmer Response (OS/VS2):* See the programmer responses for the messages that follow this message at the console.

*Problem Determination:* Table I, item 2; if a DSS dump was taken, follow the procedure given under "How To Obtain A Dump," in Section 4, to print the DSS page-formatted data set; if a DSS dump was not taken, Table I, item 11; if not already done, Table I, item 29 or 30.

IQA022A

INTERVENTION REQUIRED, DEVICE xxxx.

*Explanation:* The device at control unit address xxxx needs operator intervention.

*System Action:* The I/O operation is suspended until the required operator intervention is completed.

*Programmer Response:* Perform the action indicated by the I/O data in the supplementary messages.

IQA023I

STACK FULL, { INTERRUPT LOST } FOR DEVICE xxxx.  
{ MAY LOSE INTERRUPT }

*Explanation (VS1):* The DSS I/O interrupt stack (IQASTK) has overflowed. One interrupt has been lost; OS/VS1 integrity is questionable.

*Explanation (VS2):* The DSS I/O interrupt stack (IQASTK) is full. One interrupt may be lost, whether or not the next command is GO or DISCONNECT.

*System Action:* Processing continues. DSS may attempt to get the interrupt back to the system.

*Programmer Response:* Issue GO or DISCONNECT as soon as possible, so that the interrupts in the stack can be passed back to OS/VS.

*Problem Determination:* Table I, item 11 while DSS is loaded.

In addition,

- If it does not seem possible that many interrupts have occurred, item 29 (if not already done)
- If the error persists due to false interrupts, item 30

**IQA024D CANNOT TAKE DSS DUMP, CONTINUE? REPLY Y OR N.**

*Explanation:* One of the following situations has prevented a DSS error dump:

- A permanent I/O error (indicated by message IQA025D and then this message).
- Recursion into IQAERR00 (the DSS error processor) while IQAERR00 was trying to dump DSS.

*System Action:* DSS waits for the programmer's response. A press of the END key with no preceding Y or N is treated as a Y.

*Programmer Response:*

Y (yes)—Processing resumes as described under "Resumption of Processing," in Section 4.

N (no)—The system enters a disabled wait state. The system wait code is 0F1.

*Problem Determination:* Table I, item 2, item 11 while DSS is loaded, and, if not already done, item 29.

**IQA025D DSS DUMP DESIRED? REPLY Y OR N.**

*Explanation:* A DSS error, or an OS/VS2 error that affects DSS, occurred and is described by the preceding message. A DSS dump is probably necessary for complete diagnosis of the problem.

*System Action:* DSS waits for the programmer's response. A press of the END key with no preceding Y or N is treated as a Y.

*Programmer Response:*

Y (yes)—DSS-occupied storage is dumped to a high-speed printer; the programmer will be asked to enter the printer address. Processing then resumes as described under "Resumption of Processing," in Section 4.

N (no)—Processing resumes as described under "Resumption of Processing," in Section 4.

**IQA026I <severity> NO DSS SAVE AREAS AVAILABLE; LOC.ID. xx IQAxxxxx IQAxxxxx**

*Explanation:* DSS module IQATSS00 has returned with a nonzero return code indicating that there are no more save areas available in the DSS save area pool. This should not occur, as the pool is assembled with ample space. For further information refer to "System Errors," in Section 4.

*System Action:* DSS issues message IQA025D, to give the programmer the option of taking a DSS error dump. Processing then continues as described for errors in DSS logic under "Resumption of Processing," in Section 4.

*Programmer Response:* See the programmer responses for the messages that follow this message at the console.

*Problem Determination:* DSS software error. Reply Y to message IQA025D. Table I, item 16. Investigate the contents of the save area pool, which is assembled into module IQATSS00, to determine which DSS modules are not releasing save areas.

**IQA027I CANNOT SIGP CPU(S).**

*Explanation:* DSS attempted to gain control of one or more CPUs via a SIGNAL PROCESSOR instruction that ordered CPU restart. The attempt failed, as indicated by a return code of 8 or 12 from IQAICC00.

*System Action:* Processing continues; DSS cannot access data (registers, PSW, etc.) in the unavailable CPU. DSS will not be able to disconnect until it can reach all CPUs via SIGNAL PROCESSOR instructions.

*Programmer Response:* If DSS cannot control a CPU because of a correctable condition (for example, the operator is intervening, the CPU does not have its power on, or the CPU is in the stopped state), correct the condition.

*Problem Determination:* Table I, item 2.

**IQA028I** <SEVERE> UNEXPECTED PRIVILEGED OPERATION CODE;  
LOC.ID. xx IQAxxxxx IQA:xxxx

*Explanation:* The DSS privileged operation simulator (IQAPOS00) has encountered a privileged operation code that is not in the privileged operation translate table. For further information, refer to "System Errors," in Section 4.

*System Action:* DSS automatically disconnects, after issuing message IQA025D to give the programmer the option of taking a DSS error dump.

*Programmer Response:* See the programmer responses for the messages that follow this message at the console.

*Problem Determination:* DSS software error. Reply Y to message IQA025D. Table I, items 2 and 16.

**IQA029** { I } <SEVERE> UNIDENTIFIABLE DSS ERROR; LOC.ID. xx IQAxxxxx IQAxxxx  
          { D }

{ CANNOT TAKE DSS DUMP CONTINUE? REPLY Y OR N. }  
{ DSS DUMP DESIRED? REPLY Y OR N. }

*Explanation:* The DSS error processor (IQAERR00) has been called without a DSS error block (IQAERB) or with an invalid code in the ERBID field. For further information, refer to "System Errors," in Section 4.

*System Action:* The same as for IQA000D.

*Programmer Response:* The same as for IQA011D.

*Problem Determination:* The same as for IQA011D.

**IQA030I** <SEVERE> CANNOT ACCESS DSS PAGING DATA SET; LOC.ID. xx  
IQAxxxxx IQAxxxxx

*Explanation:* DSS cannot access SYS1.DSSVM (the DSS paging data set), either because of I/O errors or because DSS cannot get to the CPU that owns the path to the device. For further information refer to "System Errors," in Section 4.

*System Action:* DSS automatically disconnects, after issuing message IQA025D to give the programmer the option of taking a DSS error dump.

*Programmer Response:* Run the DSS sysgen utility (IQADVM00) to rebuild SYS1.DSSVM, then try to use DSS.

*Problem Determination:* Reply Y to IQA025D. Table I, items 2 and 16.

**IQA031I** <severity> INVALID INPUT PARAMETER; LOC.ID. xx IQAxxxxx IQAxxxxx

*Explanation:* A DSS module has received invalid input from another DSS module. For further information, refer to "System Errors," in Section 4.

*System Action:* DSS issues message IQA025D, to give the programmer the option of taking a DSS error dump. Processing then continues as described for errors in DSS logic under "Resumption of Processing," in Section 4.

*Programmer Response:* See the programmer responses for the messages that follow this message at the console.

*Problem Determination:* DSS software error. Reply Y to message IQA025D. Table I, item 16.

**IQA032I** <severity> LRA FAILED FOR A FIXED DSS OR SYSTEM PAGE;  
LOC.ID. xx IQAxxxxx IQAxxxxx

*Explanation:* The condition code from a LOAD REAL ADDRESS instruction indicates that a DSS or OS/VS2 page is not available. This should not occur, because the page is supposed to be fixed. For further information, refer to "System Errors," in Section 4.

*System Action:* DSS issues message IQA025D, to give the programmer the option of taking a DSS error dump. Processing then continues as described for errors in DSS logic under "Resumption of Processing," in Section 4.  
*Programmer Response:* See the programmer responses for the messages that follow this message at the console.  
*Problem Determination:* DSS software error. Reply Y to message IQA025D. Table I, item 16.

**IQA033A ENTER PRINTER ADDRESS (4 DIGITS)**

*Explanation:* The programmer replied Y to message IQA025D.  
*System Action:* DSS waits for the programmer's response.  
*Programmer Response:* Enter a printer address in the form of four hexadecimal digits. If the system uses three-digit device addresses, include a leading zero as the fourth digit.

**IQA034D INVALID PRINTER ADDRESS. REENTER OR REPLY N TO CANCEL.**

*Explanation:* The programmer replied incorrectly to message IQA033A.  
*System Action:* DSS waits for the programmer's response.  
*Programmer Response:* Enter a valid printer address in the form of four hexadecimal digits (including a leading zero if the system uses three-digit device addresses), or enter N to cancel the dump.

**IQA035I INVALID CCW IN IOS SENSE CHAIN.**

*Explanation:* During the return of stacked I/O error status indicators to OS/VS2, DSS encountered an unexpected command code in a CCW.  
*System Action:* DSS terminates status return for the invalid sense request, indicating an error to IOS. Processing then continues.  
*Programmer Response:* None.

**IQA036I AT ENCOUNTERED IS SUBJECT OF EXECUTE INSTRUCTION**

*Explanation:* While monitoring OS/VS2, DSS found an AT on an instruction that is the subject of an EXECUTE instruction.  
*System Action:* The AT is automatically removed; the programmer gets control at the integrated operator's console; GO or DISCONNECT will give control to the EXECUTE instruction.  
*Programmer Response:* Enter any DSS commands.  
*Problem Determination:* Table I, item 2 and, if not already done, item 30.

**IQA037I CPU xxxx NOT OPERATIONAL**

*Explanation:* The specified CPU has issued a malfunction alert, or the machine check handler has issued a SIGNAL PROCESSOR instruction that ordered an emergency signal to vary the specified CPU offline.  
*System Action:* Processing continues; DSS cannot access data (registers, PSW, etc.) in the unavailable CPU.  
*Programmer Response:* None.  
*Problem Determination:* Table I, items 2 and 30.

**IQA038I CPU xxxx NOW ONLINE**

*Explanation:* The specified CPU was varied online.  
*System Action:* Processing continues; DSS can now access data in the specified CPU.  
*Programmer Response:* None.

- IQA039I <SEVERE> NO DIRECT ACCESS SLOTS AVAILABLE FOR DSS PAGING;  
LOC.ID. xx IQAxxxxx IQAxxxxx**  
*Explanation:* There are no external page slots available in the SYS1.DSSVM data set for DSS paging. For further information, refer to "System Errors," in Section 4.  
*System Action:* DSS automatically disconnects, after issuing message IQA025D to give the programmer the option of taking a DSS error dump.  
*Programmer Response:* If the problem persists, reallocate IQADVM00 and SYS1.DSSVM.  
*Problem Determination:* Probable I/O errors in SYS1.DSSVM. Reply Y to message IQA025D. Table I, items 2, 16, and 30.
- IQA040I <SEVERE> CRITICAL MALFUNCTION DETECTED ON ANOTHER CPU;  
LOC.ID. xx IQAxxxxx IQAxxxxx**  
*Explanation:* A CPU necessary to DSS processing has issued a malfunction alert.  
*System Action:* DSS automatically disconnects, after issuing message IQA025D to give the programmer the option of taking a DSS error dump.  
*Programmer Response:* See the programmer responses for the messages that follow this message at the console.  
*Problem Determination:* Table I, items 2 and 30.
- IQA041I <SEVERE> DSS CANCELLED, REINSTATEMENT FAILED.  
LOC.ID. xx IQAxxxxx IQAxxxxx**  
*Explanation:* While processing a local error, DSS was unable to reinstate DSS. For further information, refer to "System Errors," in Section 4.  
*System Action:* DSS automatically disconnects, after issuing message IQA025D to give the programmer the option of taking a DSS error dump.  
*Programmer Response:* See the programmer responses for the messages that follow this message at the console.  
*Problem Determination:* Reply Y to message IQA025D. Table I, items 2 and 16.
- IQA042I INVALID STATUS/SENSE FOR ERP, DEV xxxxx, STAT xxxxx, SNS xxxx.**  
*Explanation:* The DSS I/O error recovery routine has detected invalid status or sense data.  
*System Action:* See the system actions for the messages that follow this message at the console.  
*Programmer Response:* None.  
*Problem Determination:* See the device address, status data, and sense data in this message to determine whether a hardware failure occurred. If a hardware failure occurred, Table I, item 30. If a hardware failure did not occur, reply Y to message IQA025D; Table I, items 2, 16, and 29.
- IQA043I <SEVERE> IEEVDSS ENTRY POINT IN IEEVWKVP NOT FOUND BY IEAVNPB6;  
LOC.ID. xx IQAxxxxx IQAxxxxx**  
*Explanation:* During a VARY CPU, ONLINE request, DSS could not find the entry point IEEVDSS. For further information, refer to "System Errors," in Section 4.  
*System Action:* DSS automatically disconnects, after issuing message IQA025D to give the programmer the option of taking a DSS error dump.  
*Programmer Response:* See the programmer responses for the messages that follow this message at the console.  
*Problem Determination:* DSS software error. Reply Y to message IQA025D. Table I, items 2 and 16.

**IQA0441**     **<SEVERE> NO DSS STATUS SAVE AREA AVAILABLE; LOC.ID. xx IQAxxxxx IQAxxxxx**

*Explanation:* During a VARY CPU, ONLINE request, DSS found that no AOS table exists for the CPU being varied online. For further information, refer to "System Errors," in Section 4.

*System Action:* DSS automatically disconnects, after issuing message IQA025D to give the programmer the option of taking a DSS error dump.

*Programmer Response:* Check the records of system generation to ensure that the CPU was included in the configuration.

*Problem Determination:* Reply Y to message IQA025D. Table I, items 2 and 16.

**IQA0451**     **ADDRESS TRANSLATION ERROR IN OS/VS**

*Explanation:* While DSS was monitoring, an error occurred during OS/VS2 address translation for an address in OS/VS2 storage.

*System Action:* DSS activates itself, to allow the programmer to take corrective action.

*Programmer Response:* Use DSS to correct the problem. When OS/VS2 is reentered, OS/VS2 will terminate the currently active task.

**IQA0461**     **SNAP OUTPUT INTEGRITY LOST. MISSING INFORMATION AVAILABLE FOR DISPLAY.**

*Explanation:* Due to simultaneous events requiring SNAP output, the SNAP buffers did not have room for all of the information.

*System Action:* DSS activates itself, to allow the programmer to display the information that was not included in the snapshot dumps.

*Programmer Response:* Enter any DSS commands.

**IQA1001**     **DSS READY**

*Explanation:* DSS initialization is complete.

*System Action:* DSS types & at the console to prompt the user for command input.

*Programmer Response:* Enter any DSS command.

**IQA1011**     **PAGE CANNOT BE ACCESSED**

*Explanation:* A page of OS/VS virtual storage could not be accessed.

*System Action:* The operand is canceled.

*Programmer Response:* Enter a GO command, let OS/VS execute a few instructions, press RESTART to give control back to DSS, and reenter the command.

*Problem Determination:* Probable OS/VS software error. Table I, items 2, 11 while DSS is loaded, 22, and, if not already done, 29.

**IQA1021**     **DSS INITIALIZED. SYSTEM IN JEOPARDY**

*Explanation:* During DSS initiation, system page fix limits were ignored to get the DSS control program established. Therefore, DSS cannot be used for monitoring.

*System Action:* Language routines prompt the user for command input. DSS cannot accept any AT, ON, or GO commands.

*Programmer Response:* Avoid the AT, ON, and GO commands. To return control to OS/VS, use the DISCONNECT command.

- IQA1031**      **LOAD MODULE name WAS UNLOADED BY OS/VS**  
*Explanation:* OS/VS unloaded a module that DSS was monitoring.  
*System Action:* DSS removes all AT breakpoints and patches in that module, as well as all names equated to areas in that module. If a breakpoint or patch belongs to a group of breakpoints or patches that have a single number or name but reside in different load modules, DSS removes only those that reside in the module being unloaded. This results in the group of ATs or patches being partially removed.  
*Programmer Response:* None.
- IQA1041**      **PERMANENT I/O ERROR ON SYSRES**  
*Explanation:* A permanent I/O error occurred while reading directory entries, scatter/translation records, or CESD control records of the SYS1.NUCLEUS library.  
*System Action:* The nucleus map is not built.  
*Programmer Response:* None.  
*Problem Determination:* Table I, item 16, and, if not already done, item 29.
- IQA1051**      **DSS PAGE FIX NOT POSSIBLE**  
*Explanation:* An unsuccessful attempt was made to fix a DSS page that contains a channel program used for building the nucleus map.  
*System Action:* The nucleus map is not built.  
*Programmer Response:* None.  
*Problem Determination:* Table I, items 2, 22. and, if not already done, 29.
- IQA1061**      **VIRTUAL ADDRESS NOT VALID**  
*Explanation:* A request by a DSS routine to refer to a page of OS/VS virtual storage was canceled because the page was marked unavailable or unassigned.  
*System Action:* The function in process is terminated.  
*Programmer Response:* None.  
*Problem Determination:* Table I, items 2, 11 while DSS is loaded, 22, and, if not already done, 29.
- IQA1071**      **NUCLEUS MAP NOT BUILT**  
*Explanation:* A map of the nucleus was not built due to the error described by a preceding message.  
*System Action:* None.  
*Programmer Response:* None.
- IQA1081**      **NUCLEUS DIRECTORY ENTRY NOT FOUND**  
*Explanation:* The nucleus member whose ID is in the communication vector table (CVT) was not found in the directory of SYS1.NUCLEUS.  
*System Action:* The nucleus map is not built.  
*Programmer Response:* Check the CVT at CVTNUCLS to ensure that the nucleus ID is valid.  
*Problem Determination:* Table I, item 16, and, if not already done, item 29.
- IQA1091**      **SYSRES DEVICE CANNOT BE ACCESSED**  
*Explanation:* The unit control block (UCB) for SYSRES that is identified in the communication vector table at label CVTSYSAD showed that an error recovery program was in control when DSS was being initialized.  
*System Action:* The nucleus map is not built.  
*Programmer Response:* None.  
*Problem Determination:* Table I, item 16, and, if not already done, item 29.

- IQA110I SNAP OUTPUT TERMINATED DUE TO I/O ERROR**  
*Explanation:* The message preceding this one describes the I/O error on the snap output device which prevents further output.  
*System Action:* The remaining records in the snap buffer are ignored.  
*Programmer Response:* None.  
*Problem Determination:* Table I, items 2 and 30.
- IQA111I &AT {name } HAS BEEN DISABLED BY DSS  
                                  {number }**  
*Explanation:* The AT could not be enabled after a machine check occurred, due to a paging error.  
*System Action:* None.  
*Programmer Response:* Remove the AT or try again to enable the AT.  
*Problem Determination:* Table I, items 2, 22, and 30.
- IQA112I MACHINE CHECK OCCURRED**  
*Explanation:* ATs and ONs were disabled while the machine check handler was in control. They have been enabled except where noted.  
*System Action:* Processing continues.  
*Programmer Response:* None.
- IQA113I PAGE NOT ASSIGNED**  
*Explanation:* A user-specified data field resides in a virtual storage page that has not been assigned by OS/VS.  
*System Action:* The rest of the command is canceled.  
*Programmer Response:* Correct the error and reenter the command.  
*Problem Determination:* Table I, items 2, 16, and 22.
- IQA114I PER INTEGRITY LOST**  
*Explanation:* Either PER (program event recording) hardware is not working properly, or during OS/VS operation the PER bit in the PSW under which OS/VS was operating or a PER related control registers (control register 9, 10, or 11) was changed from what the user requested and DSS processing required.  
*System Action:* PER interrupts, if they occurred, were ignored. ATs, if encountered, were ignored and may have been removed. No SNAP dumps were taken; no command procedures were executed.  
*Programmer Response:* Display control registers 9, 10, and 11, and the activation PSW to determine whether changes were made by OS/VS and what the effect was on PER monitoring.  
*Problem Determination:* Table I, item 29 (if not already done) or item 30.
- IQA115I DSS TRANSLATION SPECIFICATION EXCEPTION UNDER OS/VS VIRTUAL STORAGE**  
*Explanation:* DSS encountered a translation specification exception while operating under OS/VS virtual storage.  
*System Action:* DSS processing at the point of the exception is discontinued. Further DSS operation may be in jeopardy.  
*Programmer Response:* None.  
*Problem Determination:* Table I, item 29, if not already done.



- IQA1161**      **EVENT LOST DURING EXECUTION OF UNMONITORABLE CODE**  
*Explanation:* One or more ON breakpoints may have occurred while OS/VS was executing non-monitorable code. The actions specified for that event were not performed.  
*System Action:* None.  
*Programmer Response:* None.
- IQA1171**      **DSS READY. ASSIGN LOG IF HARDCOPY DESIRED**  
*Explanation:* DSS initialization is complete. When the integrated operator's console provides a display station rather than a printer, DSS writes this message to remind the programmer that he can start a hardcopy log.  
*System Action:* DSS prompts the programmer for his first command.  
*Programmer Response:* If a hardcopy log is desired, enter ASSIGN LOG = , where the equal sign is followed by a device address in the form of a decimal or hexadecimal literal. The addressed device must be a printer or a tape drive that is supported by DSS. For a complete log of the DSS session, this command should be issued first; however, it can be issued at any time during the session.
- IQA1181**      **PROGRAM EVENT RECORDING FAILURE**  
*Explanation:* The PER hardware has failed to generate an interrupt that DSS uses to detect when OS/VS1 execution reaches an AT location.  
*System Action:* Processing continues; the AT may be disabled.  
*Programmer Response:* Continue entering commands, but be aware that AT commands may not work.  
*Problem Determination:* Probable OS/VS1 hardware error. Table I, items 2 and 30.
- IQA1191**      **OS/VS HAS ENTERED A DISABLED WAIT STATE**  
*Explanation:* OS/VS1 has loaded a disabled PSW with the wait flag on.  
*System Action:* Processing continues.  
*Programmer Response:* Continue entering commands, but be aware that DISCONNECT will not work and that GO will only load the disabled-wait-state PSW, which can be displayed by DISPLAY & PSW.  
*Problem Determination:* Use DSS to dump, analyze, or repair OS/VS1. Table I, item 29 (if not already done).
- IQA1201**      **LPME ENTRY NOT ACCESSIBLE**  
*Explanation:* A DSS attempt to refer to OS/VS2 external page storage was canceled, because the LPME (logical page mapping entry) is not accessible.  
*System Action:* The function in process is terminated.  
*Programmer Response:* None.  
*Problem Determination:* Probable OS/VS2 software error. Table I, items 2, 11 while DSS is loaded, 22, and, if not already done, 29.
- IQA1211**      **LPID GREATER THAN CREATED ADDRESS SPACE**  
*Explanation:* A DSS attempt to refer to OS/VS2 external page storage was canceled, because the LPID (logical page identifier) is greater than the available address space.  
*System Action:* The function in process is terminated.  
*Programmer Response:* None.  
*Problem Determination:* Probable OS/VS2 software error. Table I, items 2, 11 while DSS is loaded, 22, and, if not already done, 29.

|                |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     |
|----------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>IQA122I</b> | <b>LPA IN TRANSIT</b><br><i>Explanation:</i> A page of the LPA (link pack area) could not be accessed.<br><i>System Action:</i> The operand is not executed.<br><i>Programmer Response:</i> Enter a GO command, let OS/VS2 execute a few instructions, press RESTART to give control back to DSS, and reenter the command.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |
| <b>IQA123I</b> | <b>BASE LGN INVALID</b><br><i>Explanation:</i> A DSS attempt to refer to OS/VS2 external page storage was canceled, because the base LGN (logical group number) is invalid.<br><i>System Action:</i> The function in process is terminated.<br><i>Programmer Response:</i> None.<br><i>Problem Determination:</i> Probable OS/VS2 software error. Table I, items 2, 11 while DSS is loaded, 22, and, if not already done, 29.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                       |
| <b>IQA124I</b> | <b>ASPCT NOT VALID OR NOT ACCESSIBLE</b><br><i>Explanation:</i> A DSS attempt to refer to OS/VS2 external page storage was canceled, because the ASPCT (auxiliary storage page correspondence table) is not valid or not accessible.<br><i>System Action:</i> The function in process is terminated.<br><i>Programmer Response:</i> None.<br><i>Problem Determination:</i> Probable OS/VS2 software error. Table I, items 2, 11 while DSS is loaded, 22, and, if not already done, 29.                                                                                                                                                                                                                                                                                                                                                                                                                              |
| <b>IQA125I</b> | <b>BAD REAL CORE ADDRESS OR IT HAS BEEN VARIED OFFLINE</b><br><i>Explanation:</i> DSS could not access a page frame, for one of these reasons: <ul style="list-style-type: none"> <li>• The programmer specified an invalid real address with &amp; RM.</li> <li>• OS/VS2 marked the page frame as bad, because of hardware problems.</li> <li>• The page frame is in a storage unit that has been varied offline.</li> </ul> <i>System Action:</i> The function in process is terminated.<br><i>Programmer Response:</i> Check to see which of the three reasons under Explanation applies. If an invalid address was specified, reenter the command with the correct address. If the page frame was marked as bad, be aware that it is not available. If the storage unit is varied offline, consider bringing it back online.<br><i>Problem Determination:</i> Probable OS/VS2 hardware error. Table I, item 30. |
| <b>IQA126I</b> | <b>TEMPORARY SLOTS FOR LPA PAGES ARE FULL</b><br><i>Explanation:</i> An AT or PATCH command has been rejected, because it would change a seventeenth LPA page. No more than sixteen LPA pages can be changed.<br><i>System Action:</i> The function in process is terminated.<br><i>Programmer Response:</i> Free one LPA page of all ATs and patches, and reenter the command.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     |
| <b>IQA127I</b> | <b>PREVIOUS READ WAS NOT ISSUED BEFORE PUT REQUEST</b><br><i>Explanation:</i> A PUT request by a DSS routine was rejected, because a GET request had not preceded it.<br><i>System Action:</i> The function in process is terminated.<br><i>Programmer Response:</i> Disconnect and restart DSS, if feasible; this may correct the problem.<br><i>Problem Determination:</i> Probable DSS software error. Table I, items 2, 11 while DSS is loaded, 22, and, if not already done, 29.                                                                                                                                                                                                                                                                                                                                                                                                                               |

**IQA128I****&ASID GIVEN WAS INVALID**

*Explanation:* The programmer attempted to use either &ASID(0) where it is not permitted, or an &ASID number that is larger than any address space number this system can assign.

*System Action:* The operand is not executed.

*Programmer Response:* Reenter the command with the correct &ASID number.

*Problem Determination:* Table I, items 2, 11 while DSS is loaded, and (if not already done) 29.

**IQA129I****ADDRESS SPACE NOT ACTIVE FOR DEFAULT OR REQUESTED ASID**

*Explanation:* The programmer specified, or accepted a default to, an address space that OS/VS2 has not assigned.

*System Action:* The operand is not executed.

*Programmer Response:* Reenter the command with the correct &ASID.

*Problem Determination:* Table I, items 2, 11 while DSS is loaded, and (if not already done) 29.

**IQA131I****RBA FOR PAGE DATASET IS OUT OF EXTENTS**

*Explanation:* A DSS attempt to refer to OS/VS2 external page storage was canceled, because the RBA (relative byte address) is outside of the page data set's extent.

*System Action:* The function in process is terminated.

*Programmer Response:* None.

*Problem Determination:* Probable OS/VS2 software error. Table I, items 2, 11 while DSS is loaded, 22, and, if not already done, 29.

**IQA200I****UNRECOVERABLE INPUT ERROR ON i/o-name**

*Explanation:* An unrecoverable input error was detected on the I/O-name data set while attempting to read a command stream.

*System Action:* All commands in the error record are ignored.

*Programmer Response:* If the data set is the integrated operator's console, reenter the command.

If the data set is a card reader, check the rejected card for a keypunching error. If the error persists, try another card reader.

If the data set is a tape drive, try another tape drive. If the error persists, recreate the tape.

*Problem Determination:* Table I, items 2, 22, and 30.

**IQA201I****PERMANENT I/O ERROR**

*Explanation:* An unrecoverable I/O error was encountered while trying to perform an I/O operation to the device specified in a supplementary message.

*System Action:* The requested I/O operation is not performed.

*Programmer Response:*

1. Try the request again.

2. Release the I/O name, assign it to another device, and retry the original request.

*Problem Determination:* Table I, items 2 and 30.

- IQA202I**      **device-address ALLOCATED TO DSS**  
*Explanation:* The programmer has attempted to assign an I/O name to a device that is already allocated to DSS.  
*System Action:* The command is canceled.  
*Programmer Response:* (1) Reenter the ASSIGN command with a different device address, or (2) enter a RELEASE command for the I/O name now assigned to the device and reenter the ASSIGN command with the same device address.  
*Problem Determination:* Table I, item 2, item 11 while DSS is loaded, and, if not already done, item 29.
- IQA203A**      **INTERVENTION REQUIRED**  
*Explanation:* The device at the control unit address shown in a supplementary message needs operator intervention.  
*System Action:* The I/O operation is suspended until the required operator intervention is completed.  
*Programmer Response:* Perform the action indicated by the I/O data in the supplementary messages.
- IQA204I**      **DATA CHECK**  
*Explanation:* The device at the control unit address shown in a supplementary message has encountered a data check.  
*System Action:* The I/O operation is canceled.  
*Programmer Response:* (1) Retry the operation, or (2) release the device and assign the I/O name to another device.  
*Problem Determination:* Table I, items 2 and 30.
- IQA205E**      **PARITY CHECK IN FCB. RELOAD BUFFER**  
*Explanation:* A parity error was detected in the forms control buffer.  
*System Action:* The I/O operation is canceled.  
*Programmer Response:* Reload the buffer.  
*Problem Determination:* Table I, items 22 and 30.
- IQA206I**      **I/O ERROR DURING PAGING**  
*Explanation:* A page of data from OS/Virtual Storage could not be accessed due to a permanent I/O error on an OS/Virtual Storage paging device.  
*System Action:* The operand is canceled.  
*Programmer Response:* Enter a GO command, let OS/Virtual Storage execute a few instructions, press RESTART to give control back to DSS, and reenter the command.  
*Problem Determination:* Table I, items 2 and 30.
- IQA207I**      **TAPE UNIT UNAVAILABLE**  
*Explanation:* The tape drive is disconnected, or a hardware malfunction prevented the system from finding it.  
*System Action:* The I/O operation is canceled.  
*Programmer Response:* Retry the operation. If this fails, release the device, assign the I/O name to another device, and retry the operation.  
*Problem Determination:* Table I, items 2 and 30.
- IQA208I**      **END OF TAPE**  
*Explanation:* The user attempted a dump for which there was not enough space remaining on the tape.  
*System Action:* DSS writes up to the end of the tape, prints this message, rewinds the tape, has the tape demounted, and issues message IQA393.  
*Programmer Response:* See message IQA393.

- IQA209A REENTER DATA**  
*Explanation:* DSS has encountered an input data check.  
*System Action:* DSS attempts to read the reentered data. After four unsuccessful attempts to read the data, the I/O operation is canceled.  
*Programmer Response:* Reenter the data.
- IQA210I HARDWARE MALFUNCTION IN PRINTER**  
*Explanation:* The printer has malfunctioned; there may be extraneous lines in the output.  
*System Action:* DSS retries the operation three times. If the retries fail, DSS issues message IQA201I.  
*Programmer Response:* None.  
*Problem Determination:* Table I, items 2 and 30.
- IQA211I MECHANICAL MOTION FAILURE**  
*Explanation:* A moving part in the IBM 3211 printer has malfunctioned.  
*System Action:* The I/O operation is canceled.  
*Programmer Response:* Retry the operation.  
*Problem Determination:* Table I, items 2 and 30.
- IQA212I DATA ERROR IN UNIVERSAL CHARACTER SET BUFFER**  
*Explanation:* DSS found faulty data in the Universal Character Set Buffer (UCSB) on the IBM 3211 printer. This could be due to a hardware error; however, it is probably due to faulty loading of the UCSB.  
*System Action:* Printing continues; the line printed at this point may be invalid.  
*Programmer Response:* If you want to reprint the line, retry the I/O operation. If the error recurs,  
  1. Return control to OS/VS by issuing a GO command, then
  2. Reload the UCSB using the SETPRT macro instruction as described in *OS/VS Data Management Macro Instructions, GC26-3793*, then
  3. Return control to DSS by pressing RESTART, then
  4. Retry the I/O operation.*Problem Determination:* Table I, items 2, 22, and 30.
- IQA213A PARITY CHECK IN FORMS CONTROL BUFFER. RELOAD BUFFER**  
*Explanation:* There was a parity check in the Forms Control Buffer on the IBM 3211 printer.  
*System Action:* The I/O operation is canceled.  
*Programmer Response:*  
  1. Return control to OS/VS by issuing a GO command, then
  2. Reload the FCB using the SETPRT macro instruction as described in *OS/VS Data Management Macro Instructions, GC26-3793*, then
  3. Return control to DSS by pressing RESTART, then
  4. Retry the I/O operation.*Problem Determination:* Table I, items 2, 22, and 30.
- IQA214A RELOAD THE UNIVERSAL CHARACTER SET BUFFER**  
*Explanation:* The Universal Character Set Buffer (UCSB) on the IBM 3211 printer is unreadable.  
*System Action:* The I/O operation is canceled.  
*Programmer Response:*  
  1. Return control to OS/VS by issuing a GO command, then
  2. Reload the UCSB using the SETPRT macro instruction as described in *OS/VS Data Management Macro Instructions, GC26-3793*, then
  3. Return control to DSS by pressing RESTART, then
  4. Retry the I/O operation.*Problem Determination:* Table I, items 2, 22, and 30.



**IQA303I UNPAIRED APOSTROPHES**

*Explanation:* A command contains an odd number of apostrophes.

*System Action:* The invalid command is canceled. Because of the unpaired apostrophes, the proper command delimiter may not have been detected and more than one command may be affected.

*Programmer Response:* Correct the error and reenter the command.

*Problem Determination:* Table I, item 2. In addition, enter DISPLAY &SYM or DUMP &SYM and save the resulting output.

**IQA304I INVALID CONTINUATION**

*Explanation:* An input record ended with an underscore, which indicates continuation, but an end of file was detected while trying to read the next record.

*System Action:* The invalid command is canceled.

*Programmer Response:* Reenter the entire command.

**IQA305I COMMAND TOO LONG**

*Explanation:* The command contains more than 256 characters.

*System Action:* The command is canceled.

*Programmer Response:* Break the command into smaller commands and reenter them.

*Problem Determination:* Table I, item 2. In addition, enter DISPLAY &SYM or DUMP &SYM and save the resulting output.

**IQA306I PROCESSING STOPPED AFTER ABOVE COMMAND**

*Explanation:* An attention interrupt has been recognized following execution of the displayed command.

*System Action:* DSS returns control to the console, as if a DIVERT command had been entered.

*Programmer Response:* Enter any DSS commands. To resume execution at the point of the attention interrupt, enter a DIVERT command specifying the interrupted command stream or enter a REVERT command to return to the interrupted command procedure.

**IQA307D CANCEL ABOVE COMMAND? REPLY Y or N**

*Explanation:* An attention request has been recognized during execution of the displayed command.

*System Action:* DSS waits for the programmer's response.

*Programmer Response:* Enter Y to cancel the command, or enter N to defer the attention request until the command finishes execution.

**IQA308I INVALID CHARACTER (character) IN { DECIMAL  
HEXADECIMAL } LITERAL  
ABSOLUTE ADDRESS**

*Explanation:* The specified character is not valid for this type of literal.

*System Action:* The command containing the literal is canceled.

*Programmer Response:* Correct the invalid literal and reenter the command.

*Problem Determination:* Table I, item 2.

**IQA309I DECIMAL LITERAL EXCEEDS MAXIMUM VALUE**

*Explanation:* A decimal literal was found whose value exceeds  $2^{31}-1$ .

*System Action:* The command containing the literal is canceled.

*Programmer Response:* Correct the invalid literal and reenter the command.

*Problem Determination:* Table I, item 2.

- IQA310I    INVALID ABSOLUTE ADDRESS**  
*Explanation:* A simple address literal or an address in an address range literal exceeds the maximum address in virtual storage.  
*System Action:* The command containing the literal is canceled.  
*Programmer Response:* Correct the invalid literal and reenter the command.  
*Problem Determination:* Table I, item 2, and item 11 while DSS is loaded.
- IQA311I    INVALID ABSOLUTE ADDRESS RANGE**  
*Explanation:* The second address specified in an address range literal is smaller than the first address.  
*System Action:* The command containing the literal is canceled.  
*Programmer Response:* Correct the invalid literal and reenter the command.  
*Problem Determination:* Table I, item 2.
- IQA312I    COMMAND CANCELED BY USER REQUEST**  
*Explanation:* The user replied YES to message IQA307.  
*System Action:* DSS cancels the command and returns control to the console.  
*Programmer Response:* Enter any DSS commands. To resume execution at the point of the attention interrupt, enter a DIVERT command specifying the interrupted command stream or enter a REVERT command to return to the interrupted command procedure.  
*Problem Determination:* Table I, items 2 and 22.
- IQA313I    COMMAND PARTIALLY CANCELED DUE TO ABOVE ERROR(S)**  
*Explanation:* The command had one or more invalid operands.  
*System Action:* The invalid operands of the command are not processed. The command is executed for all valid operands.  
*Programmer Response:* Correct the invalid operands and reenter the command with only the corrected operands.  
*Problem Determination:* Table I, items 2 and 22.
- IQA314I    REMAINDER OF COMMAND CANCELED DUE TO ABOVE ERROR**  
*Explanation:* The command had an invalid operand.  
*System Action:* The command is canceled at the point that the error is detected.  
*Programmer Response:* Correct the error and reenter the command with the corrected operand and all following operands.  
*Problem Determination:* Table I, items 2 and 22.
- IQA315I    COMMAND CANCELED DUE TO ABOVE ERROR**  
*Explanation:* The command had an invalid operand and could not be executed.  
*System Action:* The entire command is canceled.  
*Programmer Response:* Correct the error and reenter the entire command.  
*Problem Determination:* Table I, items 2 and 22.
- IQA316I    INVALID &S OPERAND**  
*Explanation:* The operand of a substitution function is not a single identifier.  
*System Action:* The command is canceled.  
*Programmer Response:* Correct the error and reenter the command.  
*Problem Determination:* Table I, item 2.



- IQA317I UNDEFINED &S OPERAND (symbol)**  
*Explanation:* The named variable appears as the operand of a substitution function, but is not a defined user symbol or a procedure variable.  
*System Action:* The rest of the command is scanned for substitution functions. When all substitutions have been processed, the command is canceled.  
*Programmer Response:* Correct the error (possibly by defining the substitution variable) and reenter the command.  
*Problem Determination:* Table I, item 2. In addition, enter DISPLAY &SYM or DUMP &SYM and save the resulting output.
- IQA318I &S OPERAND (symbol) NOT TYPE C**  
*Explanation:* The named variable appears as the operand of a substitution function but does not have the character (C) type attribute.  
*System Action:* The rest of the command is scanned for substitution functions. When all substitutions have been processed, the command is canceled.  
*Programmer Response:* Correct the error (possibly by redefining the variable's type attribute) and reenter the command.  
*Problem Determination:* Table I, item 2. In addition, enter DISPLAY &SYM or DUMP &SYM and save the resulting output.
- IQA319I INVALID CHARACTER (hex-code) IN &S OPERAND (symbol)**  
*Explanation:* The named variable appears as the operand of a substitution function but contains a character which is not in the DSS character set.  
*System Action:* The rest of the command is scanned for substitution functions. When all substitutions have been processed, the command is canceled.  
*Programmer Response:* Correct the error and reenter the command.  
*Problem Determination:* Table I, item 2. In addition, enter DISPLAY &SYM or DUMP &SYM and save the resulting output.
- IQA320I INVALID COMMAND LABEL (symbol)**  
*Explanation:* The specified symbol was found in the command-label field of a command, but is not an identifier.  
*System Action:* The command is canceled.  
*Programmer Response:* Correct the error and reenter the command.  
*Problem Determination:* Table I, items 2 and 22.
- IQA321I INVALID COMMAND VERB (symbol)**  
*Explanation:* The specified symbol was found in the command-verb field of a command, but is not a valid verb.  
*System Action:* The command is canceled.  
*Programmer Response:* Correct the error and reenter the command.  
*Problem Determination:* Table I, items 2 and 22.
- IQA322I INVALID IDENTIFIER (symbol)**  
*Explanation:* The specified identifier contains more than eight characters or contains a non-alphameric character.  
*System Action:* The command is canceled.  
*Programmer Response:* Correct the error and reenter the command.  
*Problem Determination:* Table I, items 2 and 22.
- IQA323I INVALID LITERAL TYPE (type-code)**  
*Explanation:* A literal was preceded by a character other than X, L, or a blank.  
*System Action:* The command is canceled.  
*Programmer Response:* Correct the error and reenter the command.  
*Problem Determination:* Table I, items 2 and 22.

**IQA324I      INVALID &P**

*Explanation:* The operand of a parameter function is not a list of identifiers, or the parameter function was specified twice in the same command.

*System Action:* The command is canceled.

*Programmer Response:* Correct the error and reenter the command.

*Problem Determination:* Table I, items 2 and 22.

**IQA325I      INVALID SYNTAX - item1 FOLLOWED BY item2**

*Explanation:* The command contains an invalid sequence of operands or operators (item1 cannot be followed by item2).

*System Action:* The command is canceled.

*Programmer Response:* Correct the error and reenter the command.

*Problem Determination:* Table I, items 2 and 22.

**IQA326I      IMPROPER USE OF UNDERSCORE**

*Explanation:* An underscore (    ) appears in a command other than (1) within a character literal, (2) as the first byte of a command verb, or (3) as the last byte of an input record.

*System Action:* The command is canceled.

*Programmer Response:* Correct the error and reenter the command.

*Problem Determination:* Table I, items 2 and 22.

**IQA327I      WRONG NUMBER OF OPERANDS**

*Explanation:* The wrong number of operands were specified for a command, or for an operation.

*System Action:* The rest of the command is canceled.

*Programmer Response:* Reenter the command with the proper number of operands.

*Problem Determination:* Table I, items 2, 11 while DSS is loaded, and 22.

**IQA328I      UNDEFINED IDENTIFIER (symbol)**

*Explanation:* The specified symbol was referred to but was not previously defined by the user, is not a valid DSS function, and is not in the system map.

*System Action:* The rest of the command is canceled.

*Programmer Response:* Define the symbol, or correct the reference to it, and reenter the command.

*Problem Determination:* Table I, item 2, 11 while DSS is loaded, and 22.

**IQA329I      OPERAND name NOT A DATA FIELD**

*Explanation:* The specified operand was used in place of a data field.

*System Action:* The rest of the command is canceled.

*Programmer Response:* Correct the operand and reenter the command.

*Problem Determination:* Table I, items 2 and 22.

**IQA330I      INVALID MAP OR QUALIFIED EXPRESSION**

*Explanation:* An improper symbolic location expression was specified.

1. Two map identifiers were specified.

2. Two load module names were specified.

3. The map-id was not specified first.

4. A required operand was not specified.

*System Action:* The rest of the command is canceled.

*Programmer Response:* Correct the error and reenter the command.

*Problem Determination:* Table I, items 2 and 22.

- IQA3311**      **INVALID ATTRIBUTE DESIGNATION FOR DATA FIELD name**  
*Explanation:* An immediate attribute designation is not valid.  
*System Action:* The rest of the command is canceled.  
*Programmer Response:* Correct the invalid immediate attribute designation and reenter the command.  
*Problem Determination:* Table I, items 2 and 22.
- IQA3321**      **INVALID RANGE**  
*Explanation:* The operands of the range operator are improperly specified for one of the following reasons:  
1. The address of the first operand is not less than the address of the second operand.  
2. One operand is a symbol internal to DSS and the other is an external symbol or location expression.  
3. Both operands are internal to DSS but they are within different data fields.  
4. One operand is in real storage and the other is in virtual storage.  
*System Action:* The rest of the command is canceled.  
*Programmer Response:* Correct the error and reenter the command.  
*Problem Determination:* Table I, items 2 and 22.
- IQA3331**      **INVALID INDIRECT ADDRESS**  
*Explanation:* A data field specified as an indirect address contains an address that exceeds the limit of virtual storage.  
*System Action:* The rest of the command is canceled.  
*Programmer Response:* Correct the error and reenter the command.  
*Problem Determination:* Table I, items 2 and 22.
- IQA3341**      **INVALID SUBSCRIPT FOR DATA FIELD name**  
*Explanation:* The subscript specified for the named data field is not within the allowable range for that data field.  
*System Action:* The rest of the command is canceled.  
*Programmer Response:* Correct the error and reenter the command.  
*Problem Determination:* Table I, items 2 and 22.
- IQA3351**      **INVALID OPERAND FOR name DSS FUNCTION**  
*Explanation:* The operand of the named DSS function was not a valid operand for that function.  
*System Action:* The rest of the command is canceled.  
*Programmer Response:* Correct the error and reenter the command.  
*Problem Determination:* Table I, items 2 and 22.
- IQA3361**      **DATA FIELD name NOT ARITHMETIC**  
*Explanation:* One of the operands of an arithmetic operator has the character (C) type attribute.  
*System Action:* The rest of the command is canceled.  
*Programmer Response:* Change the operand to hexadecimal or integer and reenter the command.  
*Problem Determination:* Table I, items 2, 22, and, if not already done, 29.
- IQA3371**      **ARITHMETIC OVERFLOW IN OPERATION**  
*Explanation:* An operand of an arithmetic expression exceeded the maximum allowable value, or the result of the indicated operation exceeded the maximum allowable value.  
*System Action:* The rest of the command is canceled.  
*Programmer Response:* Correct the arithmetic expression and reenter the command.  
*Problem Determination:* Table I, items 2 and 22.

- IQA338I DATA FIELD name1 CANNOT BE COMPARED TO DATA FIELD name2**  
*Explanation:* A comparison operation was specified between a character and a hexadecimal or integer operand.  
*System Action:* The rest of the command is canceled.  
*Programmer Response:* Make both operands the same type and reenter the command.  
*Problem Determination:* Table I, items 2 and 22.
- IQA339I COMMAND TOO COMPLEX**  
*Explanation:* The command requires too many operands and/or work areas; or its internal representation exceeds the maximum allowable size.  
*System Action:* The command is canceled.  
*Programmer Response:* Simplify the command or split it into two or more simpler parts.  
*Problem Determination:* Table I, items 2 and 22.
- IQA340I INSUFFICIENT DSS WORK SPACE**  
*Explanation:* A control block that is required to execute the command cannot be allocated because there is no available space in the DSS Work Space.  
*System Action:* The rest of the command is canceled.  
*Programmer Response:* Remove unneeded user symbols, procedures, etc., and reenter the command.  
*Problem Determination:* Table I, items 2, 11 while DSS is loaded, and 22.
- IQA341I MODULE name CANNOT BE MONITORED**  
*Explanation:* An AT breakpoint, patch, or equated symbol is in the named load module, which must therefore be monitored. However, the DSS Fixed Monitoring Queue is currently full, and the module cannot be monitored.  
*System Action:* The module is placed in the queue of modules to be monitored if space in the Fixed Monitoring Queue becomes available.  
*Programmer Response:* Remove or disable unneeded ATs, patches, or symbols equated to OS/VIS storage before issuing a GO command.  
*Problem Determination:* Table I, item 2.
- IQA342I INVALID EXPRESSION IN IF COMMAND**  
*Explanation:* The IF command cannot resolve the expression to X'00' (false) or X'FF' (true).  
*System Action:* The command is canceled, and the programmer is prompted to enter another command.  
*Programmer Response:* Correct the IF expression so that it can be evaluated as true or false or returns a one-byte data field set to X'00' or X'FF', and reenter the command.  
*Problem Determination:* Table I, items 2 and 22.
- IQA343I INVALID GOTO OPERAND**  
*Explanation:* The command label specified as an operand of the GOTO command is not an identifier.  
*System Action:* The command is canceled.  
*Programmer Response:* Correct the operand and reenter the command.  
*Problem Determination:* Table I, items 2 and 22.

- IQA344I NO COMMAND PROCEDURE TO SEARCH**  
*Explanation:* A GOTO command was entered from a command stream, but there is no command procedure to search.  
*System Action:* The command is canceled.  
*Programmer Response:* Enter a replacement command, if necessary.  
*Problem Determination:* Table I, item 2.
- IQA345I COMMAND LABEL NOT FOUND**  
*Explanation:* The current command procedure was searched, and the command label specified in the GOTO could not be found.  
*System Action:* The GOTO command is canceled.  
*Programmer Response:* Correct the operand and reenter the command.  
*Problem Determination:* Table I, items 2, 22, and, if not already done, 29.
- IQA346I i/o-name IS PRESENTLY ASSIGNED**  
*Explanation:* The I/O name has already been assigned.  
*System Action:* The ASSIGN command is canceled.  
*Programmer Response:* Do one of the following:  
1. Check the I/O name. If the wrong one was specified, reenter the ASSIGN command with the correct I/O name.  
2. Release the I/O name, using the RELEASE command, and reassign it to a new device address.  
*Problem Determination:* Table I, items 2 and 22.
- IQA347I INVALID DEVICE ADDRESS (device-address)**  
*Explanation:* The device address specified is not one of the allowable OS/VS system-defined device addresses.  
*System Action:* The command is canceled.  
*Programmer Response:* Correct the invalid device address and reenter the command.  
*Problem Determination:* Table I, items 2 and 22.
- IQA348I CONFLICTING ATTRIBUTES (i/o-name, device-address)**  
*Explanation:* The I/O name and the device address conflict with each other.  
*Example:* An attempt to assign CARD to a tape drive.  
*System Action:* The ASSIGN command is canceled.  
*Programmer Response:* Change the I/O name or the device address so that they don't conflict, and reenter the command.  
*Problem Determination:* Table I, items 2 and 22.
- IQA349I device-address PRESENTLY ALLOCATED TO THE SYSTEM.  
DO YOU WANT TO OVERRIDE ALLOCATION (YES OR NO)?**  
*Explanation:* The specified device is already allocated to OS/VS or was being allocated at the time of DSS activation.  
*System Action:* DSS waits for the programmer's response. A press of the END key with no preceding YES or NO is treated as a NO.  
*Programmer Response:*  
YES—The device becomes allocated to both DSS and OS/VS. This option should not be specified unless it is absolutely necessary, because DSS cannot pass OS/VS the I/O interrupts that are pending for OS/VS on the device. In addition, if it is an output device, DSS and OS/VS output could be mixed. Finally, the issuing of a GO command before the device is released from DSS use may lead to unpredictable results.  
NO—The ASSIGN command is canceled.  
*Problem Determination:* Table I, item 2, item 11 while DSS is loaded, and, if not already done, item 29.



**IQA356I**      { &AT  
                  &ON  
                  &PATCH }      **TABLE EMPTY**  
                  &PROC  
                  &SYM }

*Explanation:* The specified table has no entries. This is a warning message and is only issued when the entire table is requested to be displayed or dumped.

*System Action:* None

*Programmer Response:* None

**IQA357I**      **INVALID** { **RM** } **ADDRESS** (xxxxxxx)  
                  { **VM** }

*Explanation:* The address specified exceeds the size of real storage (if RM) or the maximum allowable virtual storage address (if VM).

*System Action:* The operand is canceled.

*Programmer Response:* Correct the error and reenter the command.

*Problem Determination:* Table I, items 2, 11 while DSS is loaded, and 22.

**IQA358I**      **i/o-name CURRENTLY IN USE**

*Explanation:* The I/O name cannot be released, as it belongs to a command stream that is currently in use.

*System Action:* The RELEASE command is canceled.

*Programmer Response:*

1. If the I/O name is incorrect, reenter the RELEASE command with the correct I/O name.

2. If the I/O name is correct, stop using the command stream and reissue the RELEASE command from a different command stream or from a command procedure.

*Problem Determination:* Table I, items 2 and 22.

**IQA359I**      **DATA FIELD** { **name** }      **EXCEEDS** { **2048** }      **BYTES**  
                  { **L'xxxxxxx'** }      { **4096** }

*Explanation:* The length of a data field being modified by a COLLECT, PATCH, or SET command exceeds one page in size.

*System Action:* The rest of the command is canceled.

*Programmer Response:* Correct the length attribute and reenter the command.

*Problem Determination:* Table I, item 2, and item 11 while DSS is loaded.

**IQA360I**      **DATA FIELD name1 CANNOT BE SET TO name2**

*Explanation:* The type attributes of the source and receiving data fields for a COLLECT or SET conflict. One is character and the other is integer.

*System Action:* The rest of the command is canceled.

*Programmer Response:* Change the type attribute of one of the data fields and reenter the command.

*Problem Determination:* Table I, item 2, and item 11 while DSS is loaded.

**IQA361I**      **NEITHER SNAP NOR COMMAND PROCEDURE SPECIFIED**

*Explanation:* An AT or ON command was entered without a SNAP parameter and no command procedure was specified.

*System Action:* The command is canceled.

*Programmer Response:* Reenter the command, specifying a SNAP parameter or command procedure or both.

*Problem Determination:* Table I, item 2.





**IQA368I** { AT } NAME = name, NUMBER = number  
{ ON }  
{ PATCH }

*Explanation:* This informational message is issued after each AT, ON, or PATCH command is executed to tell the user the number associated with the AT, ON, or PATCH for future use with the ENABLE, DISABLE, REMOVE, DUMP, or DISPLAY commands.

*System Action:* None.

*Programmer Response:* None.

*Problem Determination:* Table I, items 2, 11 while DSS is loaded, and 22.

**IQA369I** MONITORING QUEUE IS STILL FULL

*Explanation:* A GO command was entered but the DSS monitoring queue is full and one or more modules cannot be monitored.

*System Action:* The command is canceled.

*Programmer Response:* Remove or disable unneeded ATs, ONs, patches, or symbols equated to OS/VS storage before reentering the GO command.

*Problem Determination:* Table I, item 2. In addition, display or dump all &ATs, &ONs, &PATCHes, &PROCes, and &SYMes and save the resulting output.

**IQA370I** { &AT } { name }  
{ &ON } { number } CANNOT BE ENABLED AT THIS TIME

*Explanation:* An AT or ON was requested to be enabled but the DSS Monitoring Queue is full.

*System Action:* The rest of the command is canceled.

*Programmer Response:* Remove or disable activated ATs or ONs and reenter the command.

*Problem Determination:* Table I, item 2.

**IQA371I** { &AT } { name } ALREADY { ENABLED }  
{ &ON } { number } { DISABLED }

*Explanation:* An AT or ON was requested to be ENABLED or DISABLED and it already is in the state requested.

*System Action:* None.

*Programmer Response:* None.

*Problem Determination:* Table I, items 2 and 22.

**IQA372I** &AT L'xxxxxxx' CANNOT BE REMOVED

*Explanation:* A DISCONNECT command was entered, but an AT SVC still resides in OS/VS storage. Paging or I/O error prevents its removal.

*System Action:* The command is canceled.

*Programmer Response:* Issue the GO command. If the system is able to bring in the page and the AT SVC is executed, DSS will remove the SVC at that time, replace the operation code, and not execute the SNAP and/or command procedure associated with this AT. A DISCONNECT command can then be entered successfully.

*Problem Determination:* Table I, item 2.

**IQA373I** SIGN CHANGE WHILE SETTING name

*Explanation:* The sign of the specified data field changed as a result of the SET command.

*System Action:* None.

*Programmer Response:* None.

*Problem Determination:* Table I, items 2, 22, and, if not already done, 29.



- IQA382I**      **xxxxxx IS SYSTEM TASK TCB**  
*Explanation:* The TCB location specified in &TCBMAP() or &JOBMAP() was for a system task.  
*System Action:* The map is displayed or dumped.  
*Programmer Response:* The load modules used by system tasks are in the nucleus or the LPA/RRMA and can be referred to by the &SVM and &SVMMAP DSS functions.  
*Problem Determination:* Table I, items 2, 22, and, if not already done, 29.
- IQA383I**      **INVALID &DOUT SPECIFIED**  
*Explanation:* The &DOUT device has been specified as CMDT or PUNCH and the operand to be dumped is not &PROC.  
*System Action:* The operand is canceled and the DUMP/DISPLAY command processor continues with the next operand.  
*Programmer Response:* Reenter the command, specifying a different &DOUT device.  
*Problem Determination:* Table I, items 2, 22, and, if not already done, 29.
- IQA384I**      **xxxxxxxx NOT TCB ADDRESS**  
*Explanation:* 1) In &JOBMAP(), the address literal is not the address of a TCB, or 2) in &TCBMAP(), the item in parentheses is not an address literal or not the address of a TCB.  
*System Action:* The map is not displayed or dumped.  
*Programmer Response:* Reenter the command with a correct address.  
*Problem Determination:* Table I, items 2, 11 while DSS is loaded, and 22.
- IQA385I**      **INVALID PROCEDURE NAME**  
*Explanation:* The procedure name is not an identifier or is not unique.  
*System Action:* The command is canceled.  
*Programmer Response:* 1) Correct the procedure name and reenter the command. 2) Remove the identifier that duplicates your procedure name and reenter the command.  
*Problem Determination:* Table I, items 2 and 22.
- IQA386I**      **INVALID PARAMETER VARIABLE (parameter-variable)**  
*Explanation:* The parameter variable specified is not an identifier or is not a unique identifier.  
*System Action:* The command is canceled.  
*Programmer Response:* 1) Correct the parameter variable and reenter the command. 2) Change the parameter variable to a unique identifier and reenter the command. 3) Remove the identifier that conflicts with your parameter variable name and reenter the command.  
*Problem Determination:* Table I, items 2 and 22.
- IQA387I**      **NO COMMAND-VERB SPECIFIED**  
*Explanation:* The command verb could not be isolated in the command.  
*System Action:* The command is canceled.  
*Programmer Response:* Correct the error and reenter the command.  
*Problem Determination:* Table I, items 2 and 22.

**IQA3881      PROCEDURE NAME NOT FOUND**

*Explanation:* The procedure name could not be found in the list of available procedure definitions.

*System Action:* The command is canceled.

*Programmer Response:* Check the procedure name. If the procedure name is incorrect, correct it and reenter the command. If the procedure name is correct, the procedure must be defined before it can be invoked.

*Problem Determination:* Table I, items 2 and 22.

**IQA3891      TOO {MANY}      PARAMETER VARIABLES SPECIFIED**  
                  {FEW }

*Explanation:* The number of parameter variables does not correspond to the procedure definition.

*System Action:* The command is canceled.

*Programmer Response:* Reenter the command with the correct number of parameter variables.

*Problem Determination:* Table I, items 2 and 22.

**IQA3901      NO {&AT} TO {ENABLE}**  
                  {&ON}      {DISABLE}

*Explanation:* The ENABLE or DISABLE command processor did not find an AT or an ON. If an AT or ON command was previously entered, it was removed by a REMOVE or DISCONNECT command.

*System Action:* The command is canceled.

*Programmer Response:* Correct the &AT operand or &ON operand of the ENABLE or DISABLE command, or enter the appropriate AT or ON command.

*Problem Determination:* Table I, items 2 and 22. In addition, display or dump all ATs or all ONs, and save the resulting output.

**IQA3911      COMMAND PROCEDURE CANCELED [DUE TO ABOVE ERROR]**

*Explanation:* The command was invalid due to a serious environmental problem; as a result, the command procedure could not be defined or was of doubtful integrity. In OS/VS2, this message could also be issued if the programmer pressed RESTART while the command procedure was being processed.

*System Action:* The command procedure is canceled.

*Programmer Response:* Perform the action indicated for the message that precedes this message on the console sheet.

*Problem Determination:* As indicated for the preceding message.

**IQA392A      REWIND AND DEMOUNT OUTPUT TAPE**

*Explanation:* The DSS I/O control routine was unable to rewind the tape and have it demounted.

*System Action:* Immediately after this message, DSS issues message IQA393.

*Programmer Response:* Manually rewind and demount the tape.

*Problem Determination:* Table I, items 2 and 30.

**IQA393A      { PLEASE MOUNT UNLABELED TAPE OR PRESS RESTART TO CANCEL COMMAND }**  
                  { PLEASE MOUNT UNLABELED TAPE IF DESIRED, REPLY YES OR NO }

*Explanation:* An unlabeled tape should be mounted on the device to which DSS output is pending.

*System Action:* DSS waits for the tape to be mounted or the command to be canceled.

*Programmer Response (OS/VS2):* Either mount an unlabeled tape on the device to which DSS output is pending, or press RESTART, on the CPU control panel, to cancel the output-producing command.

*Programmer Response (OS/VS1):* Either reply YES and mount an unlabeled tape on the device to which DSS output is pending, or reply NO to cancel the output-producing command.

*Problem Determination:* Table I, items 2 and 30.

**IQA394I MOUNTED TAPE IS LABELED**

*Explanation:* A labeled tape was mounted on a device to which DSS output is pending. DSS supports only unlabeled tapes for output.

*System Action:* DSS waits for completion of the programmer response.

*Programmer Response:* Demount the labeled tape and mount an unlabeled tape.

*Problem Determination:* Table I, item 2, and, if not already done, item 29.

**IQA395I THE OS/VS2 QUICK START D. S. WILL BE MARKED INVALID. YES - CONTINUE, NO - CANCEL**

*Explanation:* If the pending PATCH, SET, or AT command is executed, more than five LPA pages will contain patches, SET alterations, or AT breakpoints.

*System Action:* DSS waits for the programmer's response. Any response other than YES is treated as a NO.

*Programmer Response:*

**YES**—DSS executes the command and changes quick-start record 2 to indicate that the quick-start data set is invalid. The next IPL cannot be a warm start unless (1) the command is PATCH or AT, and (2) the patch or AT is removed before the next IPL.

**NO**—DSS cancels the command and issues message IQA101.

*Problem Determination:* Do not reply YES to the message. Table I, item 2, item 11 while DSS is loaded, and, if not already done, item 29.

**IQA396I NO MODIFIER SUPPLIED FOR dss-function DSS FUNCTION**

*Explanation:* The format of the DSS function calls for an item in parentheses; this item was omitted.

*System Action:* The operand is rejected.

*Programmer Response:* Reenter the command, supplying the omitted item.

*Problem Determination:* Table I, item 2, item 11 while DSS is loaded, and, if not already done, item 29.

**IQA397A device-address PRESENTLY IN USE BY A SYSTEM FUNCTION DO YOU WISH TO OVERRIDE? REPLY YES OR NO.**

*Explanation:* An ASSIGN command requested a device that is allocated to an OS/VS2 component such as OLTEP or IEHDASDR.

*System Action:* DSS waits for the programmer's response.

*Programmer Response:*

**YES**—The ASSIGN command is executed. If the device is a tape drive, OS/VS2 data or DSS data could be lost.

**NO**—The ASSIGN command is not executed.

**IQA398A UNIT RECORD OR TAPE ALLOCATION IN PROGRESS. DO YOU WISH TO ALLOCATE? REPLY YES OR NO.**

*Explanation:* Dynamic device reconfiguration and/or device allocation was in progress for an OS/VS2 job step when OS/VS2 was interrupted by DSS. The ASSIGN command that was just entered may cause DSS to use the same I/O device that OS/VS2 will select for the job step.

*System Action:* DSS waits for the programmer's response.

*Programmer Response:*

**YES**—The ASSIGN command is executed. If the job step uses the same type of device, release the device before entering a GO command.

**NO**—The ASSIGN command is not executed.

|         |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        |
|---------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| IQA399I | <p><b>{&amp;DOUT }<br/>{&amp;SOUT } DEVICE NOT SET. BUFFER PURGED.</b></p> <p><i>Explanation:</i> The programmer attempted to use the DUMP command or the SNAP facility without specifying an output device.</p> <p><i>System Action:</i> The DUMP command is canceled, or the buffer scheduled for the snapshot dump is purged.</p> <p><i>Programmer Response:</i> Assign a device address to an output I/O name; then set &amp;DOUT or &amp;SOUT to that I/O name.</p>                                                                                                                                                                                               |
| IQA401I | <p><b>{&amp;LD } {REMOVED }<br/>{&amp;UNLD } FUNCTION CANNOT BE {IMPLANTED }</b></p> <p><i>Explanation:</i> A paging error will prevent DSS from executing any more:</p> <ul style="list-style-type: none"> <li>• ON &amp;LD commands.</li> <li>• ON &amp;UNLD commands.</li> <li>• REMOVE &amp;ON commands for any ON &amp;LD or ON &amp;UNLD commands that have already been executed.</li> </ul> <p><i>System Action:</i> Processing continues for all uses of DSS except those listed above.</p> <p><i>Programmer Response:</i> None.</p> <p><i>Problem Determination:</i> Possible OS/VS2 operator error, hardware error, or software error. Table I, item 2.</p> |
| IQA402I | <p><b>{&amp;ASID } {ASSIGNED }<br/>{&amp;CPUID } number NOT CURRENTLY {AVAILABLE }</b></p> <p><i>Explanation:</i> The address space in &amp;ASID has not yet been assigned by OS/VS2, or the CPU in &amp;CPUID is offline or cannot be halted.</p> <p><i>System Action:</i> The command is not executed.</p> <p><i>Programmer Response:</i> Reenter the command with the correct &amp;ASID or &amp;CPUID.</p> <p><i>Problem Determination:</i> Table I, items 2 and 11.</p>                                                                                                                                                                                            |
| IQA403I | <p><b>INVALID MODIFIER(S) SPECIFIED FOR DSS FUNCTION {&amp;ASID }<br/>{&amp;CPUID }</b></p> <p><i>Explanation:</i> More than one number has been given for &amp;ASID or &amp;CPUID, or the number is not a decimal or hexadecimal literal.</p> <p><i>System Action:</i> The command is not executed.</p> <p><i>Programmer Response:</i> Correct the operand and reenter the command.</p> <p><i>Problem Determination:</i> Table I, item 2.</p>                                                                                                                                                                                                                         |
| IQA404I | <p><b>INVALID &amp;ASID</b></p> <p><i>Explanation:</i> The programmer attempted to use an &amp;ASID number that is either negative or larger than any address space number this system can assign.</p> <p><i>System Action:</i> The command is not executed.</p> <p><i>Programmer Response:</i> Reenter the command with the correct &amp;ASID number.</p> <p><i>Problem Determination:</i> Table I, items 2, 11 while DSS is loaded, and (if not already done) 29.</p>                                                                                                                                                                                                |
| IQA405I | <p><b>&amp;ASID CONFLICTS WITH LOCATION</b></p> <p><i>Explanation:</i> A location is qualified by the number of an address space in which the location does not exist.</p> <p><i>System Action:</i> The command is not executed.</p> <p><i>Programmer Response:</i> Correct the location or the &amp;ASID number and reenter the command.</p> <p><i>Problem Determination:</i> Table I, item 2 and, if not already done, item 29.</p>                                                                                                                                                                                                                                  |

**IQA406I**

**I/O INTERRUPT STACK ALMOST FULL; GO COMMAND RECOMMENDED**

*Explanation:* There is room for only 10 more interrupts in the I/O interrupt stack. If the stack overflows, interrupts for devices used by OS/VIS could be lost and OS/VIS integrity could be damaged.

*System Action:* If DSS is processing an interruptable command, the programmer is given an opportunity to cancel the command.

*Programmer Response:* Enter GO or DISCONNECT as soon as possible, so that the interrupts in the stack can be passed back to OS/VIS.

*Problem Determination:* Table I, item 11 while DSS is loaded. In addition,

- If it does not seem possible that many interrupts have occurred, item 29 (if not already done)
- If the error persists due to false interrupts, item 30

**IQA407I**

**SIZE ATTRIBUTE SUPPLIED IS INVALID**

*Explanation:* The size of a new data field is zero, is too large, or is specified in an EQUATE command.

*System Action:* If possible, DSS supplies a default value for the data field's size; otherwise, the definition of the data field is canceled.

*Programmer Response:* If the system action is not satisfactory, try to adjust the size of the data field or recreate it in a way that satisfies both DSS and the programmer.

*Problem Determination:* Table I, item 2, and, if not already done, item 29.

**IQA408I**

**{ &B  
&I  
&SA  
&RA  
&LD  
&UNLD  
&AT }** **EVENT WAS PENDING FOR CPU xx**

*Explanation:* An AT or ON is being removed or disabled automatically or by a DISABLE command, REMOVE command, or a change of &RANGE. However, the monitoring interrupt has already occurred and is waiting to be processed.

*System Action:* DSS completes the removal or disabling, but gives control to the programmer at the integrated operator's console.

*Programmer Response:* Enter any DSS commands.

**IQA409I**

**ENTER DUMP IDENTIFICATION (100 CHARACTERS MAXIMUM)**

*Explanation:* This message allows the programmer to enter a title for the &PRDMP[RM] he requested.

*System Action:* DSS waits for the programmer's response.

*Programmer Response:* Enter a dump identification of up to 100 characters.

**IQA410I**

**DUMP COMPLETED. FILE NUMBER IS (nnnn).**

*Explanation:* The requested &PRDMP or &PRDMPRM dump has successfully completed. The tape file number for the dump is indicated in the parentheses.

*System Action:* Processing continues.

*Programmer Response:* None.

**IQA411I DUMP TERMINATED. RETURN CODE IS (nnnn).**

*Explanation:* The requested &PRDMP or &PRDMPRM dump has terminated abnormally. The return code in the parentheses indicates the reason for the termination:

- 4 -- canceled via RESTART
- 8 -- invalid &ASID in &QT
- 12 -- uncorrectable I/O error
- 20 -- internal logic error in IQAPDMP0
- 24 -- internal logic error in IQAMTP00
- 28 -- internal logic error in IQACTL00

*System Action:* The dump is terminated. If the return code is 20, 24, or 28, DSS issues messages IQA008D and IQA025D prior to this message.

*Programmer Response:*

| Return Code   | Action                                                                                  |
|---------------|-----------------------------------------------------------------------------------------|
| 4             | Reissue the DUMP command if the dump is still wanted.                                   |
| 8             | Correct the &ASID and reissue the DUMP command.                                         |
| 12            | Change the tape reel or change the tape drive, and reissue the DUMP command.            |
| 20, 24, or 28 | See the programmer responses for the messages that precede this message at the console. |

*Problem Determination:*

| Return Code   | Action                                                                  |
|---------------|-------------------------------------------------------------------------|
| 4 or 8        | Table I, items 2, and, if not already done, item 29.                    |
| 12            | Table I, items 2 and 30.                                                |
| 20, 24, or 28 | DSS software error. Table I, item 2, and, if not already done, item 29. |

**IQA412I NULL LITERAL INVALID**

*Explanation:* A hexadecimal, character, or address literal has no data; the apostrophes are empty.

*System Action:* The command is canceled.

*Programmer Response:* Correct the operand and reenter the command.

*Problem Determination:* Table I, item 2.

**IQA500I DATA SETS NOT OPENED SUCCESSFULLY FOR IQADVM00**

*Explanation:* The DSS utility that runs as a step during Stage II of system generation could not open the SYS1.LINKLIB or SYS1.DSSVM data set.

*System Action:* The job step is terminated.

*Programmer Response:* Determine why the data sets were not opened, correct the error, and rerun the job step.

*Problem Determination:* Table I, items 2, 3, 4, 16, 20, and 29.

**IQA501I IQALAN00 LOAD MODULE CSECTS NOT IN PROPER ORDER**

*Explanation:* The IQALAN00 load module on the SYS1.LINKLIB was not link edited with the proper control statements.

*System Action:* The job step is terminated.

*Programmer Response:* Correct the link edit control statements, link edit the IQALAN00 load module again, and rerun the job step.

*Problem Determination:* Table I, items 26b and 29.



- IQA502I DSS CANCELED (INTERNAL LOGIC ERROR)**  
*Explanation:* During DSS initialization, the DSS error exit routine was called with no errors indicated; that is, the error exit routine cannot determine why it was called.  
*System Action:* DSS returns control to OS/VS2.  
*Programmer Response:* None.  
*Problem Determination:* DSS software error. Table I, item 2 and, if not already done, item 29.
- IQA503I DSS CANCELED (CONSOLE STATUS ERROR)**  
*Explanation:* DSS is canceled during initialization, because the integrated operator's console is varied offline or allocated to a job.  
*System Action:* The system dispatches the next ready task and proceeds.  
*Programmer Response:* Correct the status of the console and reinvoke DSS.
- IQA504I DSS CANCELED (CONSOLE UCB NOT FOUND)**  
*Explanation:* DSS is canceled during initialization, because the UCB for the console sysgened as the integrated operator's console (IOC) was not found in the chain of console UCBs.  
*System Action:* The system dispatches the next ready task and proceeds.  
*Programmer Response:* Re-sysgen with the address of a supported console in the IOC parameter of the SCHEDULER macro.
- IQA505I DSS CANCELED (CONSOLE NOT SUPPORTED)**  
*Explanation:* DSS is canceled during initialization, because the console specified as the integrated operator's console is not a supported device.  
*System Action:* The system dispatches the next ready task and proceeds.  
*Programmer Response:* Restart the system with the address of a supported console in the IOC parameter of the SYSGEN macro.
- IQA506I DSS CANCELED (DAR ENTRY)**  
*Explanation:* DSS is canceled during initialization, due to an OS/VS paging error or DSS programming error detected by OS/VS and passed on by the damage assessment routine (DAR).  
*System Action:* The system dispatches the next ready task.  
*Programmer Response:* None.  
*Problem Determination:* Table I, items 2, 11, and, if not already done, 29.
- IQA507I DSS CANCELED (PAGES UNAVAILABLE)**  
*Explanation:* DSS is canceled during initialization, because OS/VS does not have sufficient pages available to fix the DSS Control Program.  
*System Action:* The system dispatches the next ready task and proceeds.  
*Programmer Response:* If the user can reduce the number of pages currently being used in the system, he should do so and re-invoke DSS by pressing the RESTART Key.  
*Problem Determination:* Table I, items 2, 4, and, if not already done, 29.
- IQA508I DSS CANCELED (PAGING DEVICE NOT SUPPORTED)**  
*Explanation:* DSS is canceled during initialization, because the device specified for the DSS data set is not supported by DSS.  
*System Action:* The system dispatches the next ready task.  
*Programmer Response:* If the programmer wants to use DSS, he must ensure that the DSS data set (SYS1.DSSVM) is on a supported device.

|                |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
|----------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>IQA510I</b> | <b>DSS CANCELED (NIP INITIALIZATION FAILED)</b><br><i>Explanation:</i> During NIP, the DSS resource initialization module, IEAVNPB6, was either unable to find IQAENV00 in the LPA or unable to find SYS1.DSSVM.<br><i>System Action:</i> DSS returns control to OS/VS2.<br><i>Programmer Response:</i> If the programmer wants to use DSS, he must re-IPL while ensuring that IQAENV00 is in the LPA and that SYS1.DSSVM is cataloged and available.                                                                                                                                                                                                                                                                                                                                                |
| <b>IQA511I</b> | <b>DSS CANCELED (UNABLE TO STOP CPUS)</b><br><i>Explanation:</i> During initialization of DSS in a multiprocessor environment, DSS was unable to stop all of the CPUs.<br><i>System Action:</i> DSS returns control to OS/VS2.<br><i>Programmer Response:</i> If DSS cannot control a CPU because of a correctable condition (for example, the CPU is in the stopped state), correct the condition.<br><i>Problem Determination:</i> Table I, item 2.                                                                                                                                                                                                                                                                                                                                                |
| <b>IQA513I</b> | <b>DSS CANCELED (NO ACCEPTABLE INTEGRATED OPERATOR'S CONSOLE FOUND)</b><br><i>Explanation:</i> DSS was unable to find an integrated operator's console for one of the following reasons: <ul style="list-style-type: none"> <li>• None of the consoles listed in OS/VS2 module IEEUCUM is marked as integrated.</li> <li>• None of the consoles marked as integrated has an I/O path to the CPU on which system restart was issued.</li> <li>• All of the consoles that are marked as integrated and have an I/O path to the restart CPU are either allocated or varied offline.</li> </ul> <i>System Action:</i> DSS returns control to OS/VS2.<br><i>Programmer Action:</i> Either ensure that a console is available to the restart CPU, or (in multiprocessing) activate DSS on a different CPU. |
| <b>IQA514I</b> | <b>DSS DISCONNECTED</b><br><i>Explanation:</i> A DISCONNECT command has been successfully executed.<br><i>System Action:</i> OS/VS2 resumes processing.<br><i>Programmer Response:</i> None.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                         |
| <b>IQA516I</b> | <b>DSS CANCELED (NO DSS SAVE AREA)</b><br><i>Explanation:</i> CSECT IQAIOC00 was unable to obtain a DSS save area from CSECT IQATSS00.<br><i>System Action:</i> DSS returns control to OS/VS2.<br><i>Programmer Response:</i> None.<br><i>Problem Determination:</i> DSS software error. Run the AMDPRDMP service aid program with the LPAMAP control statement, to locate IQAENV00 in the LPA. Then run AMDPRDMP with the PRINT STORAGE = control statement, to print the contents of IQAENV00.                                                                                                                                                                                                                                                                                                     |

## Appendix A: Restrictions

These restrictions are general; restrictions relating to specific commands are given with the command descriptions in Section 2.

1. The maximum size of an input record from the console is 256 characters.
2. The maximum size of a single DSS command, excluding continuation characters, is 256 characters.
3. DSS work space—the total number of ATs, ONs, procedures, patches, and user symbols—is limited to those that will fit, with their control blocks, in the 64K work space.
4. Enabled ATs and ONs (the total number of ATs and ONs that can be enabled at any time) is limited to those whose fixed control blocks will fit in a 2K storage area. On the average, this should be about 64 enabled ATs and ONs in OS/VS1, and about 253 ATs and 92 ONs in OS/VS2.
5. The maximum number of operands that can be specified for any command is 25.
6. The maximum size of a field that may be used as input to a comparison or boolean operator is 256 bytes in OS/VS1 or 4096 bytes in OS/VS2.
7. The maximum length of the initial value that can be stored in a data field defined by the DEFINE command is 2048 bytes in OS/VS1 or 4096 bytes in OS/VS2.
8. The maximum depth of definitional-command nesting is 256.
9. Limitations on DSS access to OS/VS1:
  - An AT or patch should not be placed in a transient area.
  - DSS cannot implant an AT or patch over the first instruction of a first-level interrupt handler; in addition, DSS cannot monitor these instructions via the ON command.
  - DSS cannot monitor the machine check handler using AT or ON.
  - ATs are not allowed in the I/O supervisor.
  - DSS has no access to OS/VS1 data sets.
  - DSS has no access to IPL or NIP processing.
  - The SET command should not be used to change control register 9, 10, or 11.
10. Limitations on DSS access to OS/VS2:
  - ATs are not allowed in prefixed storage areas (PSAs).
  - DSS cannot monitor part of the restart interrupt handler.
  - DSS cannot monitor the part of the machine check handler that owns the new PSWs.
  - DSS cannot monitor part of the stop/restart subroutine.

- DSS cannot monitor wake-up code in vary-online processing.
- No more than 16 LPA pages can be changed by AT, PATCH, or SET commands. See also restriction number 14.
- DSS has no access to OS/VS2 data sets.
- DSS has no access to IPL or NIP processing.

11. Limitations on DSS access to itself:

- DSS cannot monitor itself.
- The only write access that DSS has to itself is to the parts of DSS that are in the LPA and to DSS data fields, that is, to data fields defined by the DSS user or represented by DSS functions other than location functions (see “DSS Functions,” in “Section 2: Command Language Reference.”). (Note: Alteration of control registers 9-11 threatens DSS integrity in OS/VS1.)
- The only read access that DSS has to itself is (a) to the parts of DSS code that are mapped into OS/VS virtual storage, and (b) to DSS data fields, that is, to data fields defined by the DSS user or represented by DSS functions other than location qualifier functions (see “DSS Functions,” in “Section 2: Command Language Reference”).

12. Timing and accounting information is invalid after DSS is invoked. Thus, DSS should not be used while a time-sensitive program (for example, sensor-base or TP) is running. DSS could be used in development testing of these programs or in locating and repairing a critical problem that prevents the execution of such a program, but DSS should not be invoked during a time-dependent production run. When DSS is active, RES (Remote Entry Services) cannot be used because of teleprocessing timing considerations. That is, within one IPL, either DSS or RES can be used, but not both.

13. DSS fields all I/O interrupts belonging to OS/VS and stacks them. If there was a program controlled interrupt (PCI) among the interrupts stacked by DSS internally, the PCI user could act incorrectly—such as appending CCWs to a channel program that has terminated.

14. General rule: DSS cannot make permanent changes to OS/VS. Details:

- Any changes to OS/VS1 will not be saved for the next IPL, whether that IPL is a regular IPL or warm start.
- Changes to OS/VS2 cannot be saved for the next regular IPL, and can be saved for the next warm start only if all the changes are to the LPA and no more than five LPA pages are changed.

15. DSS works with labeled and unlabeled 800 and 1600 BPI tapes for input but only with unlabeled 800 and 1600 BPI tapes for output.

16. DSS does not support seven-track magnetic tapes.

## Appendix B: Command-Syntax Diagrams

Please Comment.  
Use Reader's Comment Form.

The diagrams in this appendix are offered as alternatives to the command-syntax illustrations in Section 2. As in those illustrations, the rules for using blanks, semicolons, and labels are not shown; these rules are covered in Section 2, under "Blanks" and "Contents of a Command."

**Note:** Shaded information applies only to OS/VS2.

### How to Use a Syntax Diagram

The rules for using a syntax diagram are:

- Start at the left of the diagram and read from left to right, except when an arrowhead indicates right-to-left flow.
- Lines represent paths through the diagram. Any word or symbol that interrupts a path is a codable item.
- A split in the path indicates a choice as to what is written next. If one of the paths in the branch does not contain codable items, that path represents a choice of coding nothing.
- Capital letters and special characters must be coded exactly as shown.
- Lowercase letters represent variables that are explained in the accompanying text.

Figure 39 gives an example of a syntax diagram. All the rules described above are illustrated here. To point out the relationships and rules, numbers and notes supplement the diagram.

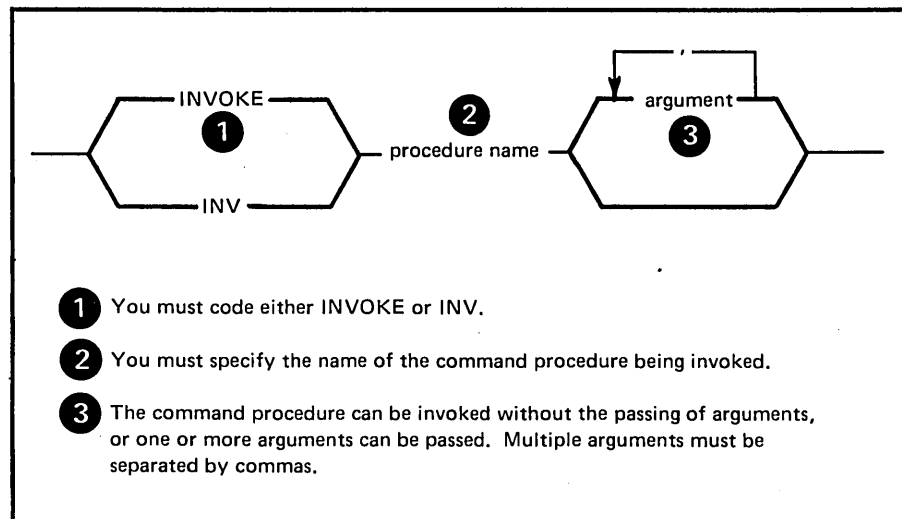
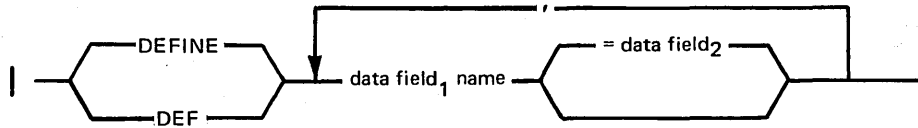
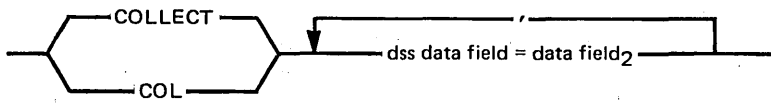
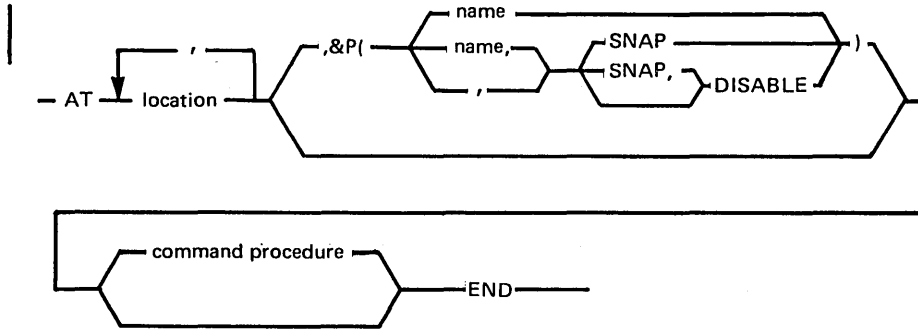
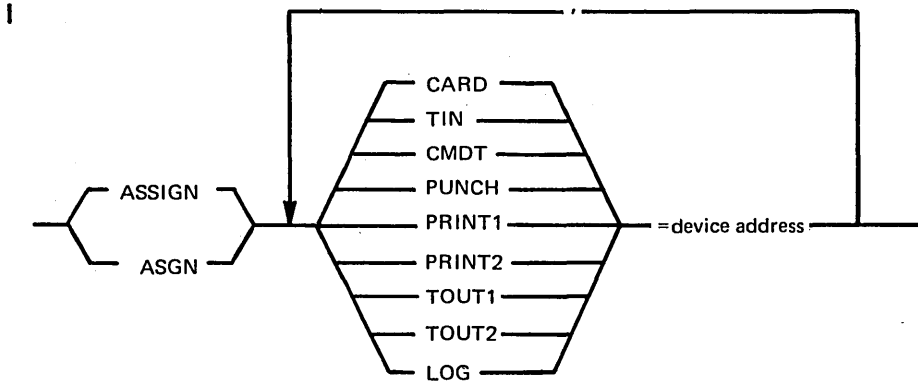
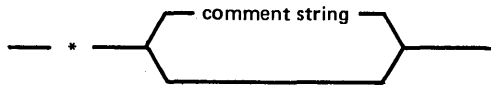
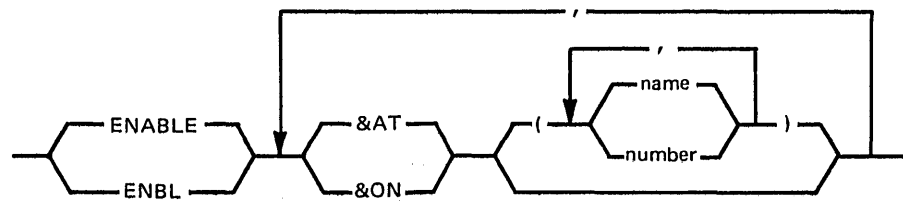
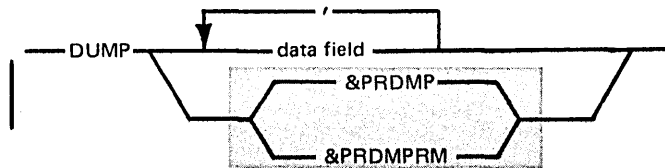
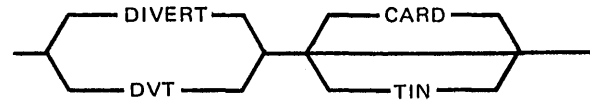
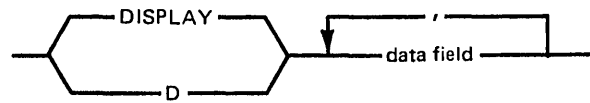
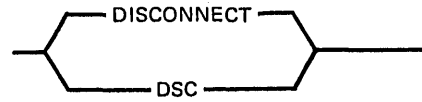
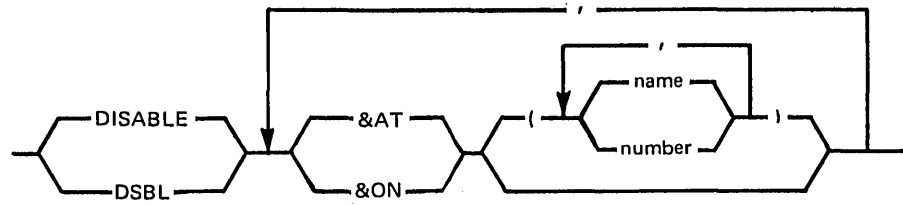


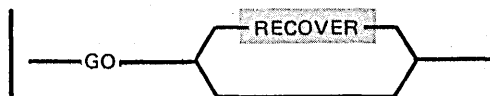
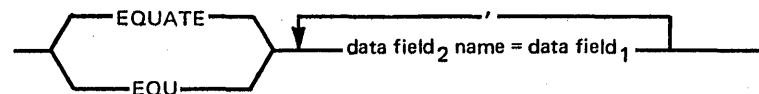
Figure 39. Syntax Diagram

# DSS Commands



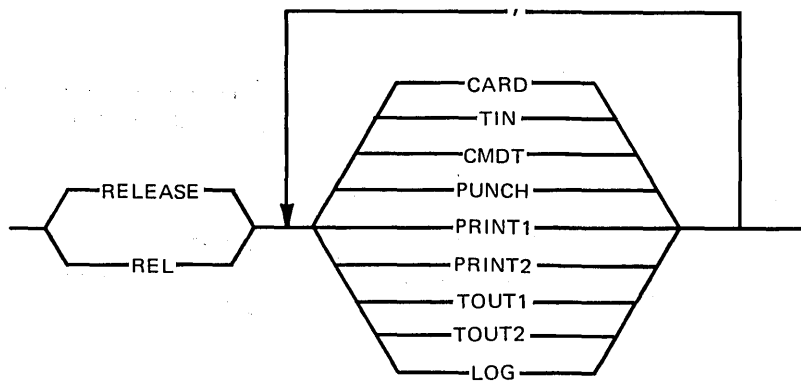
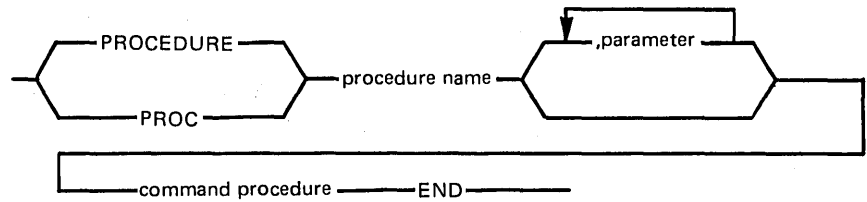
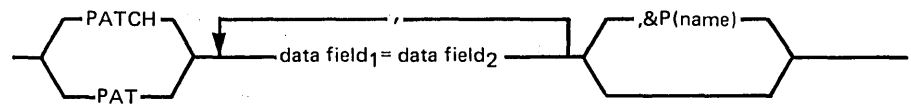
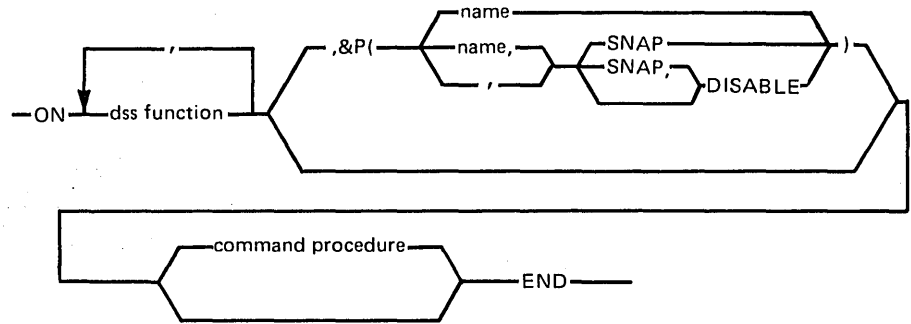
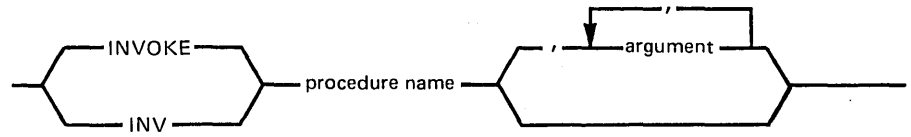


—END—

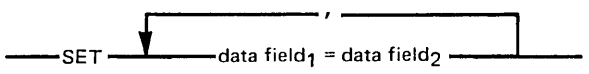
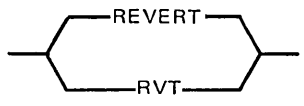
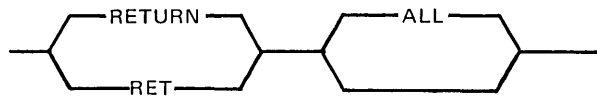
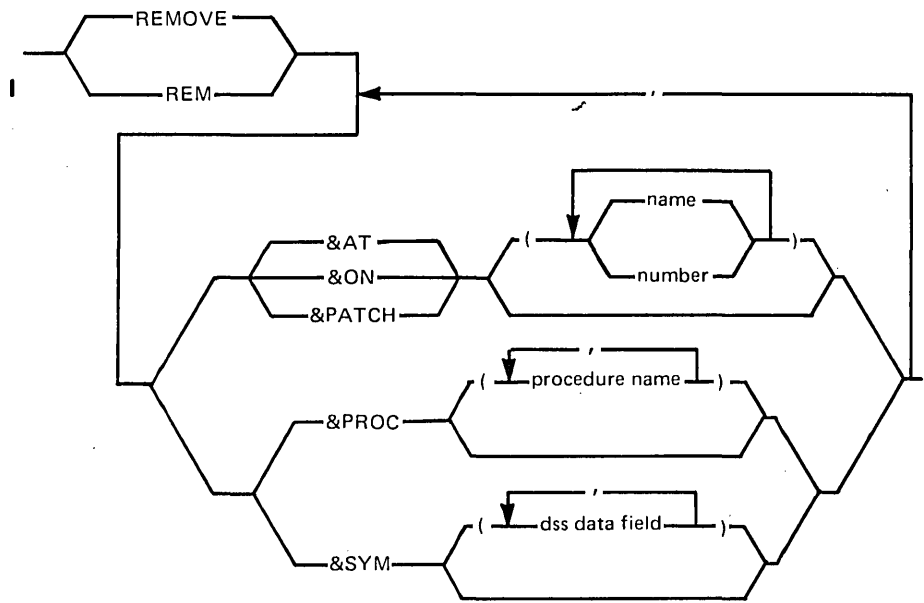


—GOTO— label —

—IF— expression — text commands — END









## Appendix C: Glossary

If the term you are seeking does not appear in this glossary, refer to **Data Processing Glossary, GC20-1699**.

IBM is grateful to the American National Standards Institute (ANSI) for permission to reprint its definitions from the American National Standard Vocabulary for Information Processing, which was prepared by Subcommittee X3K5 on Terminology and Glossary of American National Standards, Committee X3.

**alphanumeric characters:** In DSS, the characters A through Z in upper or lower case, digits 0 through 9, and €, !, \$, #, @, and ".

**asynchronous:** Without regular time relationship; unexpected or unpredictable with respect to the execution of a program's instructions.

**breakpoint:** (1)\* A place in a routine specified by an instruction, instruction digit, or other condition, where the routine may be interrupted by external intervention or by a monitor routine. (The DSS AT command can specify such a place.) (2) An event that causes an interrupt. (The DSS ON command can specify such an event.)

**command procedure:** A set of text commands that is stored for later execution. DSS text commands can be stored for later execution by an AT, ON, or PROCEDURE command.

**command stream:** A series of commands read from a console, card reader, or tape drive.

**command verb:** The name of a command, which tells a system what function it is to perform. In DSS, the command verb follows the optional command label and precedes the sometimes-optional operands.

**control level:** A level in the hierarchy of execution, where the invocation of a program or command procedure takes execution to the next highest level, and return to the invoking program or command procedure returns execution to the previous level.

**conversational:** Pertaining to a program or a system that carries on a dialog with a user, alternately accepting input and then responding to the input quickly enough for the user to maintain his train of thought. In DSS, pertaining to the dialog between DSS and the user when the user enters commands from the console.

**definitional command:** A command that makes the execution of text commands dependent upon a condition such as the occurrence of a specific type of program interrupt, the truth of an expression, or the explicit invocation of the text commands by a user. Definitional DSS commands are AT, IF, ON, and PROCEDURE.

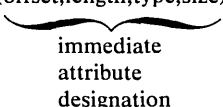
**DSS function:** A DSS command operand that is represented by an IBM-specified identifier which has & as its first character.

**hardcopy log:** In a system with multiple console support or a graphic console, a permanent record of system activity.

**I/O name:** In DSS, a name that indicates the I/O device, the record format, and the type (input or output) of a DSS I/O operation. The DSS I/O names are: CARD, CMDT, PRINT1, PRINT2, PUNCH, TIN, TOUT1, TOUT2 and LOG.

**\* identifier:** A symbol whose purpose is to identify, indicate, or name a body of data. (In DSS, a character string that consists of one to eight alphanumeric characters beginning with an alphabetic character.)

**immediate attribute designation:** In DSS, a designation of data-field attributes in this format:

data-field-name.(offset,length,type,size)  
  
immediate  
attribute  
designation

**immediate data field:** A data field that is defined only for the execution of a single command, by indirect addressing, a literal, an immediate attribute designation, or an expression.

**integrated operator's console:** The I/O device that is wired directly to the CPU.

**literal:** A symbol or quantity in a source program that is itself data, rather than a reference to data.

**map-id:** In DSS, any of the DSS functions &SVM, &JOB, &TCB, or &PRB; indicates a specific external-symbol map.

**monitoring interrupt:** An interrupt that occurs when execution reaches a breakpoint.

**prefixed storage area (PSA):** A 4K-byte area of real storage that begins at location zero and contains status information such as the PSW.

**primary control level:** In DSS, the first control level after the RESTART key is pressed.

**procedure:** See **command procedure**.

**referenced string:** In DSS, the byte string identified by a data field's offset and length attributes; the part of the data field that is being worked with.

**simple command:** A command that is neither definitional nor textual.

**text command:** A conditionally-executed command that follows a definitional command and precedes or is the END command corresponding to the definitional command.

---

\*American National Standard Definition



# Index

Where more than one page reference is given, the major reference is first.

Indexes to systems reference library manuals are consolidated in the *OS/VS1 Master Index*, GC24-5104, and in the *OS/VS2 Master Index*, GC28-0663. For additional information about any subject in this index, refer to other publications listed for the same subject in the *Master Index*.

- [ ] 29-30
- { } 29-30
- Ⓟ 17, 16
  - restrictions 19
- Ⓢ 16, 15
- 17, 16
- ... 29-30
- . ( ) 17, 21
- < 16-17, 49
- ( 16, 17
- ( ) 17
- + 16, 49
- | 16-17, 49
- & 17, 16
  - example 19
- && 17, 49
- &AC (activation code) 33, 142
- &ADDR 42, 142
- &ASID 42-43, 23
  - &ID 44
  - &JOBMAP, restriction 47
  - &LD 48
  - &PRDMP 69
  - &QT 45-46
  - &RANGE 35-36, 42
  - &TCBMAP, restriction 47
  - &UNLD 48
  - AT command, example 60
  - mapped data fields 25
  - ON command, example 78
  - output format 142
  - restrictions 186, 188
- &AT 36, 138
  - automatic removal 163
- &B 47-48, 143
- &C 31, 146
- &CAW 34, 142
- &CID 34, 142
- &CPUID 42-43, 23
  - &CAW 34
  - &CID 34
  - &CSW 34
  - &IOEL 35
  - &PRDMP 69
  - &PRM, restriction 35
  - &QT 45
  - &TEA 36
  - DISPLAY command, example 66
  - machine register functions 30
  - output format 142
  - program status functions 31
- &CSW 34, 142
- &DOUT 38, 142
- &EPSW 32, 142
- &EPSWN 33, 142
- &F 31, 146
- &G 30, 146
- &HDR 39, 142
- &I 47-48, 142
- &ID 44, 142-143
- &IOEL 35, 142
- &IPSW 32, 142
- &IPSWN 33, 142
- &JOB map-id 41, 147
- &JOBMAP 47, 144-145
- &L 40, 146
- &LD 48, 137
- &LM 44, 147
- &M (mask function) 39, 142
- &MC 35, 142
- &MPSW 32, 142
- &MPSWN 33, 142
- &O 39, 146
- &ON 37, 139
- &P 37, 143
- &PATCH 37, 139-140
  - automatic removal 163
- &PPSW 32, 142
- &PPSWN 33, 142
- &PRB map-id 41, 147
- &PRDMP[RM] 69, 38
  - output format 135
- &PREFIX 31, 142
- &PRM (register alteration mask) 35, 142
- &PROC 37, 140-141
- &PSW 33, 142
- &Q 45, 147
- &QT 44-46, 142
  - &PRDMP 69
  - AT command 59
  - ON command 78
  - SET command 85
- &RA 47-48, 142
- &RANGE (address range) 35-36, 142
  - &ASID 42
- &RM 44, 133
  - size of data field 22, 23
- &RPSW 33, 142
- &RPSWN 33, 142
- &S 38, 143
- &SA 47-48, 142
- &SOUT 38, 142
- &SPSW 32, 142
- &SPSWN 33, 142
- &SVM map-id 41, 147
- &SVMMAP 46-47, 143-145
- &SYM 37, 141
- &SZ 40, 146
- &T 40, 146
- &TCB map-id 41, 147
- &TCBMAP 47, 144-145
- &TEA 36, 142
- &TOD 36, 142
- &UNLD 48, 137
- ! 16, 15
- \$ 16, 15
- \* 16, 49
- ) 16, 17
- ; 17, 16
  - null command 18
  - when not needed 18
- ⌋ 16-17, 49
- 16, 49
- / 16, 49

- // 16, 49
- , 17, 16
- % 24-25, 16
- 17, 16
  - preceded by blank 19
- > 16-17, 49
- ? 17, 16
  - in character literal 22
- : 17, 16
- # 17, 16
  - example 19
- @ 16, 15
  - in character literal 22
- ' 17, 16
- = 16-17, 49
- " 16, 15
  
- A messages 149
- absolute address literal 22-24
- ABTERM 127
- accounting information 192
- activation code (&AC) 33
- active link pack area 46
- addition sign (+) 16, 49
- address literal 22-24
- address range (&RANGE) 35-36
- address space
  - control block (ASCB) 69, 130
  - extension block (ASXB) 130
  - identification (see &ASID)
  - vector table (ASVT) 69, 130
- alphabet-extender character 15, 16
- alphabetic characters 15, 16
- alphanumeric characters, glossary definition 199
- AMDPRDMP program 135, 69
  - &PRDMP[RM] 69, 135
  - problem determination 150
  - TOUT1 and TOUT2 53
- AMDSADMP program 150
- ampersand (&) 16-17
- AND, infix (&&) 51, 16-17
  - priority 49
- apostrophe (') 16, 17
- argument 76, 93
- array 26-27
- arithmetic operators 49-50
  - infix 49-50
  - prefix 49
- ASCB (address space control block) 69, 130
- ASSIGN command 57-58
  - (see also following I/O names: CARD, CMDT, LOG, PRINT1, PRINT2, PUNCH, TIN, TOUT1, TOUT2)
- and &DOUT 38
- conflicting I/O name and device address 177
- duplicate assignment prohibited 168, 177
- examples 58
  - DIVERT command 67, 84
  - DUMP command 69, 70
  - SNAP dump 55
  - use of a command stream 113
- RELEASE command 81
- required for all data sets 57
- SNAP dump 58, 77
- TIN 58
- asterisk (\*) 16, 49
- ASVT (address space vector table) 69, 130
- ASXB (address space extension block) 130
- asynchronous, glossary definition 199
- AT activation code 33-34
- AT command 58-60, 103-104
  - (see also &AT; breakpoint; command procedure, AT; definitional command; SNAP dump; text command)

- &QT 59
  - contents 58-59, 18
  - disabling of 65, 58
  - DISCONNECT command 82, 115
  - enabling of 70-71, 58
  - END command 71
  - examples 60
    - COLLECT command 62
    - END command 71
  - figures 112, 117
  - GO command 73
  - location operand 58-60
  - ON command, nesting of AT command in 78
  - OS/VS execution, effect of AT on 59, 103-104
  - PROCEDURE command, nesting of AT command in 80-81
  - procedure, storage of 112
  - qualify table 59
  - REMOVE command 82-83
  - restrictions
    - initiation of DSS, problems in 162
    - location 58-60
    - number of ATs 60
    - reuse of the AT 59
    - rules for definitional command 18-19
    - storage of command procedure 112
    - syntax checked, when 59
    - unloading of module 59
  - at sign (@) 15, 199
  - attention interrupt (see RESTART key)
  - attribute functions 39-40
    - (see also &L; &O; &SZ; &T)
  - automatic removal
    - DISCONNECT 65, 82
    - module unloaded 82, 163
- backspace equivalent (?) 17, 22
- base attribute 20
- blanks 19, 16-17
- boolean operators 50-51
  - AND, infix 51
  - NOT, prefix 50
  - OR, infix 51
- braces ( { } ) 29-30
- brackets ( [ ] ) 29-30
- branch instruction 47, 54
- breakpoint
  - glossary definition 199
  - name and number 181
  
- C data fields 20
- CANCEL key 105
- capital letters 29, 193
- CARD I/O name 52-53
- catastrophic error 125
- CAW (channel address word) 34
- CCW (channel command word) 192
- cent sign ( ¢ ) 15, 199
- central processing unit (CPU) 13, 23
  - manual state 101, 102
  - multiprocessing 104
  - (see also &CPUID)
- channel
  - address word (CAW) 34
  - command word (CCW) 192
  - identification (&CID) 34
  - status word (CSW) 34
- character literal 22, 76
- character string 76
- CMDT I/O name 52-53
- COLLECT command 60-62
  - restrictions
    - length of receiving data field 179

- type attributes of data fields 179
- colon (:) 16, 17
- comma (,) 16, 17
- command procedure 112
  - AT command 58-60
  - canceling, reasons for 184
  - glossary definition 199
  - interruption of 102-103
  - ON command 76-78
  - PROCEDURE command 79-81
- command stream
  - creation 113-114
  - glossary definition 199
- command verb
  - general examples 18, 56
  - glossary definition 199
- commands 15-100
  - (see also individual commands: \*, ASSIGN, AT, COLLECT, DEFINE, DISABLE, DISCONNECT, DISPLAY, DIVERT, DUMP, ENABLE, END, EQUATE, GO, GOTO, IF, INVOKE, ON, PATCH, PROCEDURE, RELEASE, REMOVE, RETURN, REVERT, and SET)
  - correcting 107, 109
    - while typing 105
  - invalid, DSS response to 108, 107
  - processing 105-106
  - summary 89-100
- command-related functions 36-39
  - (see also &AT; &DOUT; &HDR; &M; &ON; &P; &PROC; &S; &SOUT; &SYM)
- comment command 56-57
  - purpose 19, 90
  - use of semicolons in 56
- common storage 42, 69
- communication vector table (CVT) 69
- comparison operators 50
- composite symbols, restriction 19
- console (see integrated operator's console)
- contents of a command 18-19
- continuation underscore 17, 16
  - preceded by blank 19
- control levels 116-122
  - glossary definition 199
- control registers 31, 96
  - 9-11 restricted 191-192, 31
- conversational, glossary definition 199
- CPU (central processing unit) 13, 23
  - manual state 101, 102
  - multiprocessing 104
    - (see also &CPUID)
- CSECT 41
- CSW (channel status word) 34
- cursor 13
- CVT (communication vector table) 69
  
- D messages 149
- data field 19-29
  - attributes 20-21
  - name 71, 94
  - summary 93
- decimal literal 22
- DEFINE command 63-64
  - &SYM 37
  - default value of zero 64
  - truncation and padding 86
- definitional command 18-19
  - END always required 19
  - glossary definition 199
  - nesting, maximum depth of 19
- device address 94, 57
- device allocation (see ASSIGN command)
- Diagnose instruction 59
  
- DISABLE command 65
  - implied by machine check 164
- DISCONNECT command 65
  - figures 122-123, 106
  - implied by severe error 125
- DISPLAY command 66
- display console 13
- displays and dumps 133-148
- DIVERT command 67
  - figures 114-115, 106
  - implied by RESTART 119
  - implied by canceling of command 172
- division sign 16, 49
- dollar sign 15, 199
- dss-data-field 61, 94
- DSS, errors in 124-126, 11
- DSS functions 30-48, 12
  - (see also &AC; &ADDR; &AT; &B; &C; &CAW; &CSW; &DOUT; &EPSW; &EPSWN; &F; &G; &HDR; &I; &ID; &IPSW; &IPSWN; &JOB; &JOBMAP; &L; &LD; &LM; &M; &MC; &MPSW; &MPSWN; &O; &ON; &P; &PATCH; &PPSW; &PPSWN; &PRB; &PRM; &PROC; &PSW; &Q; &QT; &RA; &RANGE; &RM; &RPSW; &RPSWN; &S; &SA; &SOUT; &SPSW; &SPSWN; &SYM; &SVM; &SVM; &SYM; &SZ; &T; &TCB; &TCBMAP; &TOD; &UNLD)
  - formats 96-97
  - glossary definition 199
- DSS SVC 151
- dump
  - (see also DUMP command)
  - high-speed (unformatted) 69, 38
    - format 135
  - low-speed (formatted) 69, 135-136
    - of DSS 124-125
    - of real storage 69, 150
    - of virtual storage 69
- DUMP command 69-70
  - &DOUT 38
  
- E messages 149
- ellipses (...) 29-30
- emulator compatibility-feature instruction 59
- ENABLE command 70-71
  - monitoring queue full 181
  - redundant &AT or &ON 180
- END command 71
- ENTER key 13
- ENTRY 41
- equal sign (=) 16-17, 49
- EQUATE command 71-72
  - data field name must be unique 180
  - module that can't be monitored 176
- error handling 107-109, 124-126
- error severity 124-126
- establishing default qualification 45-46
- event functions 47-48
  - (see also &B; &I; &LD; &RA; &SA; &UNLD)
- exclamation mark (!) 16, 15
- Execute instruction 59
- expressions 48-51
  - examples 51
  - IF command 74, 94
- external page storage 165-167
- external PSW 32, 33
  
- file sequence number for &PRDMP[RM] 69
- first-level interrupt handler 12, 191
  - AT command 59
  - simulated by DSS after GO RECOVER 73
- Fixed Monitoring Queue 176
- floating-point registers 31, 96
- format notation 29-30

general registers 30, 96  
 GO command 73  
   figures 122-123, 106  
   module can't be monitored 181  
   page can't be accessed 162  
   restricted after page fix difficulties 162, 180  
 GOTO command 73-74  
   figures 114, 115  
   invalid 176  
 graphic console 13  
 greater-than symbol (>) 16-17, 49  
  
 hardcopy log 55  
   glossary definition 199  
 hexadecimal address (see address literal)  
 hexadecimal literal 22  
 HMDSADMP program 150  
 how to obtain a dump 125  
  
 I data fields 20  
 I messages 149  
 I/O (see I/O names; input/output)  
 I/O extended logout address (&IOEL) 35  
 I/O interrupt stack (IQASTK) 157, 187  
 I/O names 51-53  
   (see also CARD; CMDT; LOG; PRINT1; PRINT2; PUNCH;  
   TIN; TOUT1; TOUT2)  
   glossary definition 199  
 I/O PSW 32, 33  
 I/O supervisor 59, 191  
 identifier 16  
   definition 16, 199  
 IEBGENER program 53  
 IEBPTPCH program 53  
 IDENTIFY macro 41  
 IF command 74-75  
   from console 178  
 immediate attribute designation 21  
   glossary definition 199  
   restriction 30  
 immediate data field 22-25  
   glossary definition 199  
 indirect address 24-25  
 infix arithmetic operators 49-50  
 infix boolean operators 51  
 input/output 51-55, 13  
 instruction fetch and execute 47, 54  
 integrated operator's console 10  
   display console 13  
   glossary definition 199  
   restrictions 190  
 interrupt code (see activation code)  
 interrupt handler 12, 191  
 interrupts 192, 57  
 invalid commands 107-109  
 INVOKE command 75-76  
   figures 117-119, 106  
 invoking DSS 101-104  
 IPL 191, 192  
 IQAEPR00 125, 147  
 IQASTK (DSS I/O interrupt stack) 157, 187  
  
 JCL for printing dump of DSS 125  
  
 l (length) attribute 20, 40  
   &L output 146  
 label 18  
 left parenthesis [(] 16, 17  
 length attribute (l) 20, 40  
   &L output 146  
 less-than symbol (<) 16-17, 49  
 light pen, not supported 13  
  
 line  
   cancel with CANCEL key 105  
 link pack area (LPA) 46, 192  
 literal  
   address literal 22-24  
     (see also indirect address)  
   character literal 22  
   decimal literal 22  
   glossary definition 199  
   hexadecimal literal 22  
   loading of module 48, 77  
     SNAP dump 54  
   LOC. ID. 124  
   local error 126  
   location operand 58-60, 94  
   location qualifier functions 40-46  
     (see also &ADDR; &ASID; &CPUID; &ID; &JOB; &LM;  
     &PRB; &Q; &QT; &RM; &SVM; &TCB; map-ids; qualify  
     functions)  
   log 55  
   LOG I/O name 52-53, 55  
   lowercase letters 29, 193  
   LPA (link pack area) 46, 192  
  
 machine check  
   handler (MCH) 125, 191  
     AT command 59  
     monitoring of 12, 191  
   interrupt (MCI) 125  
   PSW 32, 33  
 machine register functions 30-31  
   (see also &C; &F; &G; &PREFIX)  
 machine status functions 33-36  
   (see also &AC; &CAW; &CID; &CSW; &IOEL; &MC;  
   &PRM; &RANGE; &TEA; &TOD)  
 magnetic tape  
   input 67-68, 113-114  
   output  
     high-speed 69, 135  
     low-speed 53, 113  
 major error 125  
 map expression 97  
 map functions 46-47  
   (see also &JOBMAP; &SVMMAP; &TCBMAP)  
 map-id 41  
   glossary definition 199  
 mapped data field 25-26  
 mask function (&M) 39, 142  
 mask, register (&PRM) 35, 39  
   output 142  
 master scheduler 102  
 MCH (machine check handler) 125, 191  
   AT command 59, 60  
   monitoring of 12, 191  
 MCI (machine check interrupt) 125  
 message  
   deletion 13  
   identifier 149  
   numbering 13  
 messages 149-190  
   DSS error 151-161, 124-126  
   supplementary 149  
   type codes 149  
 metalanguage 29-30, 193  
 minor error 126  
 minus sign (-) 16, 49  
 monitor call 35, 59  
 monitoring interrupt, glossary definition 199  
 multiplication symbol (\*) 16, 49  
 multiprocessing 23  
   performance 14  
   RESTART key 103



name operand 65, 70  
     summary 95  
 nesting of definitional commands 19  
 NIP (nucleus initialization program) 12, 191  
     AT command 59, 60  
     for DSS 189  
 no op patch, example 129  
 NOT, prefix (¬) 50, 16-17  
     priority 49  
 nucleus 46  
     initialization program (NIP) 12, 191  
         AT command 59, 60  
         for DSS 189  
         mapping 26, 13  
 NUCMAP system initialization parameter 26, 13  
 null command 18  
 number operand 65, 70  
     summary 95  
 number sign (#) 15-17  
  
 offset attribute (o) 20, 39  
     &O 39, 146  
 ON command 76-78  
     (see also &ON; breakpoint; command procedure; ON  
     command; definitional command; SNAP dump;  
     text command)  
     disabling of 65, 77  
     figures 112, 117  
     PSA monitoring 12  
     restrictions 78, 191-192  
         DSS initialization, page fix problems 162  
     syntax checked, when 77  
 operands 93-95  
     (see also DSS functions; I/O names)  
 operation of DSS 101-131  
 operator's console (see integrated operator's console)  
 OR, infix (|) 51, 16-17  
     priority 49  
  
 padding 86  
 page-formatted data set 125  
 parameter operand 79-81, 95  
 parameters (&P) 37, 143  
 parentheses [()] 16, 17  
 PATCH command 78-79  
     (see also &PATCH)  
     restrictions 79  
         length of modified data field 179  
         name must be unique 180  
 PCI (program controlled interrupt) 192  
 PER (program event recording) 13  
 percent sign (%) 16-17, 24-25  
 performance considerations 14  
 period (.) 16-17  
 plus sign (+) 16, 49  
 pointer 24  
 prefix arithmetic operators 49  
 prefix NOT 50  
 prefix register 31  
 prefixed storage area (PSA) 12, 191  
     addressing of 23, 44  
     AT command 60  
     dump of 69  
     glossary definition 199  
 PRINT1, PRINT2 I/O names 52, 53  
 private area, dump of 69  
 problem determination 149-150  
 PROCEDURE command 79-81  
     &PROC 37  
     figures 112, 121  
     syntax checked, when 80  
 procedure-name operand 75, 95  
  
 program controlled interrupt (PCI) 192  
 program event 12, 77  
     &AC 33  
     glossary definition 199  
     recording (PER) 13  
     register alteration mask 35, 39  
 program PSW 32, 33  
 program status functions 31-33  
     (see also &EPSW; &EPSWN; &IPSW; &IPSWN; &MPSW;  
     &MPSWN; &PPSW; &PPSWN; &PSW; &RPSW;  
     &RPSWN; &SPSW; &SPSWN)  
 prompting characters 19, 17  
 PSA (prefixed storage area) 12, 191  
     addressing of 23, 44  
     AT command 60  
 PSADSSGO byte 101, 102  
 PSW 31-33  
 PUNCH I/O name 52-53, 113  
  
 qualify table 44-46  
 quick-start data set 185  
 QUIESCE operator's command 101-103  
 quotation mark (") 15-16, 199  
 quote, single (') 16, 17  
  
 range  
     literal 23-24, 28  
     monitoring range 35-36  
     specifications 28-29  
 real-time conflicts 14  
 RECOVER operand of GO 73  
 recovery termination management (RTM) 73, 104  
 referenced string 20  
     glossary definition 199  
 register alteration 47, 54  
     mask 35, 39  
 registers  
     control 31, 96  
         9-11 restricted 191-192, 31  
     floating-point 31, 96  
     general 30, 96  
 RELEASE command 81  
     command-input device 179  
     DUMP or SNAP output device 178  
 remainder symbol (//) 16, 49  
 Remote Entry Services (RES) 192  
 REMOVE command 82-83  
     implied 82  
 RES (Remote Entry Services) 192  
 resident reenterable module area (RRMA) 46, 14  
     &SVM 41  
 restart interrupt handler 12, 191  
     AT command 60  
 RESTART key or function 101-104, 10  
     GO RECOVER 73  
     restrictions 191-192  
 resumption of processing 125-126  
 RETURN command 83-84  
     AT or ON command procedure 115  
     figures 118, 121-123  
     implied by end of text 83  
 REVERT command 84  
     after attention interrupt 171, 172  
     figures 115, 120  
     implied 67  
     invalid 180  
 right parenthesis 16, 17  
 RRMA (resident reenterable module area) 46, 14  
     &SVM 41  
 RTM (recovery termination management) 73, 104

- screen, display-console 13
- selector pen (see light pen)
- semicolon 17, 16
  - null command 18
  - when not needed 18
- sensor-base program 192
- SET command 85-87
  - &DOUT 38
  - &Q 44-46
  - &SOUT 38
  - other DSS functions 98-99
  - partial execution 182
- Set System Mask instruction 59
- severe error 125
- SIGNAL PROCESSOR instruction 158
- SIGP RESTART instruction 60
- simple address literal 22-23
- simple command 18
  - glossary definition 199
  - maximum size 19
- size attribute (sz) 20, 21
  - &SZ 40, 146
- slash (/) 16, 49
- SNAP dumps 54-55, 136-137
  - &SOUT 38
- special characters 16-17
- stacking of interrupts 192
- stop/restart subroutine 12, 191
  - AT command 60
  - RESTART, possible effect on 103
- storage alteration 47, 54
- stored data field 19-20
- subheading (&HDR) 39
- subscripts and arrays 26-27
- substitution function (&S) 38, 143
- subtraction sign (-) 16, 49
- summary of amendments 9
- supervisor virtual storage
  - &SVM 41, 147
  - &SVMMAP 46-47, 143-145
- supplementary messages 149
- SVC 151
- SVC PSW 32, 33
- symbol (&SYM) 37
- syntax notation 29-30
- system
  - error 124-126, 11
  - error task 102
  - event 77, 33
  - generation, reference to book on 13
  - initialization 13, 26
  - status displays 13
  - wait codes (see wait-state codes)
- SYS1.DSSVM 104, 125
- sz (size) attribute 20-21
  - &SZ 40, 146
- t (type) attribute 20-21
  - &T 40, 146
- table I 150
- tape
  - input 67-68, 113-114
  - output
    - high-speed 69, 135
    - low-speed 53, 113
- TCB (task control block), display of 128
- teleprocessing 192
- terminating DSS 115
- text command 18
  - glossary definition 199
  - maximum size 19
- time-of-day clock 36
- timing information 192
- TIN I/O name 52-53
- TOUT1, TOUT2 I/O names 52-53
- TP program 192
- transient area in OS/VS1 191
- translation exception address (&TEA) 36
- truncation 86
- type attribute (t) for data fields 20-21
  - &T 40, 146
- type code in messages 149
- UCB (unit control block)
  - display of 127-128
  - manipulation of 130-131
- underscore ( \_ ) 16, 17
  - preceded by blank 19
- unloading of module 48, 77
  - SNAP dump 54
- user's guide 101-131
- vary-online processing 192, 12
  - AT command 60
- virtual address given by &ADDR 42
- virtual storage
  - high-speed dump 69
  - low-speed dump 130
- W messages 149
- wait-state codes
  - CCC 103
  - OFA 153
  - OF1 151, 154-158
- wake-up code 192, 12
  - AT command 60
- warm start 192
- X data fields 20
- 3211 printer 169, 170



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)

GC28-0640-1

*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.*

Possible topics for comment are:

Clarity Accuracy Completeness Organization Index Figures Examples Legibility

Cut or Fold Along Line

In addition, please comment on:

|                                         | Very Dis-<br>satisfied   |                          | Neutral                  |                          | Very<br>Satisfied        |
|-----------------------------------------|--------------------------|--------------------------|--------------------------|--------------------------|--------------------------|
| "Where to Find" Directory (Front Cover) | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> |
| Command-Syntax Diagrams (Appendix B)    | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> |

What is your occupation? \_\_\_\_\_  
Number of latest Technical Newsletter (if any) concerning this publication: \_\_\_\_\_  
Please indicate 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 system 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 81  
Poughkeepsie  
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 D58, Building 706-2  
PO Box 390  
Poughkeepsie, New York 12602

Fold

Fold

OS/VS Dynamic Support System (S370-37)

Printed in U.S.A.

GC28-0640-1



**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)