DSS Command Language Reference Summary

This document includes material from *OS/VS Dynamic Support System,* GC28-0640-1, as summarized in the *OS/VS2 Release 3.7 Debugging Handbook*.

Key to Symbols in Command Illustrations

```
A-Z     *     &     ()     +
```

These must be typed as shown.

```
a-z
```

Words in these characters represent information supplied by you.

```
{}   []   . . .
```

These symbols, which do not represent typed information, have special meanings:

```
{item1}
{item2}
```

Choose one item.

```
[item]
```

The item is optional.

```
item . . .
```

You may specify two or more items.

```
item [, . . .]
```

```
[,item1] . . .
```

```
{item1}
{item2} [, . . .]
```

If you specify more than one item, separate the items by commas

```
{item1=item2}   [, . . .]
```

```
{item1[=item2]}
```

If you repeat the unit inside the {}, separate the units by commas.

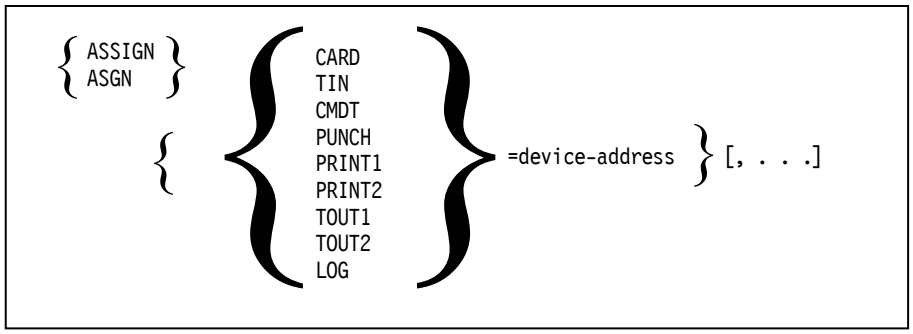
Commands

Comment

```
*     [comment]
```

Puts a comment into a command procedure or onto the console sheet.

ASSIGN



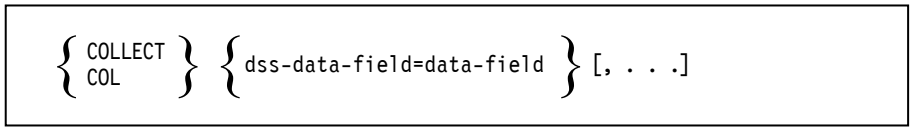
Assigns an I/O name for use by DSS; allocates an I/O device to DSS.

AT

```
AT  location [, . . .][,&P([name][,SNAP][,DISABLE])]
    [command-procedure]
    END
```

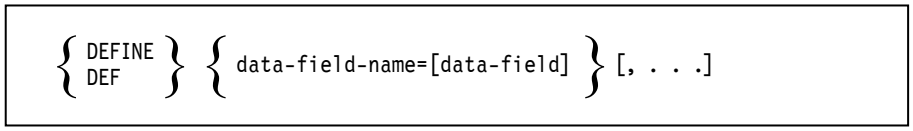
Establishes AT breakpoints at which a command procedure and/or a SNAP dump will be executed.

COLLECT



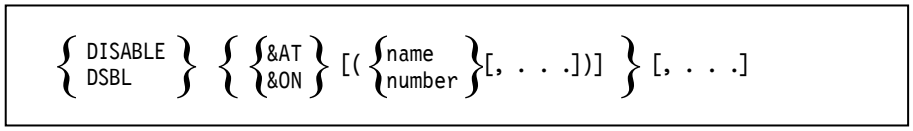
Copies data from source data fields into successive elements of a receiver data field.

DEFINE



Defines a data field in DSS work space and, optionally, initializes it.

DISABLE

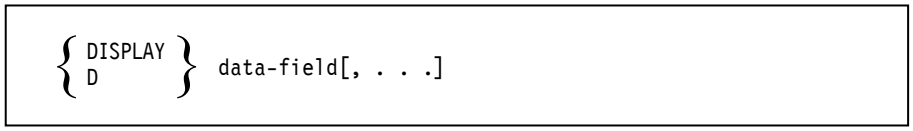


Disables ATs and ONs until ENABLE commands reactivate them.

DISCONNECT

Terminates DSS and returns control to OS/VS.

DISPLAY

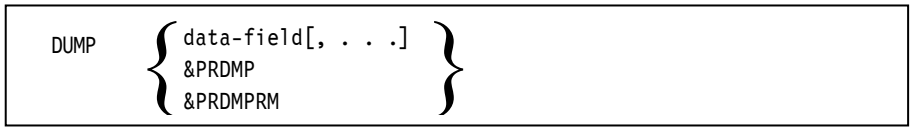


Writes data at the integrated operator's console.

DIVERT

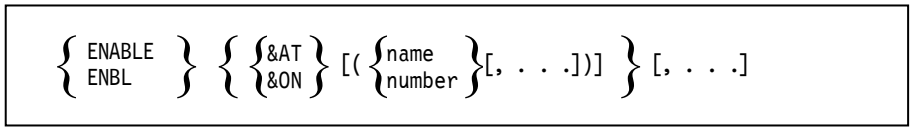
Changes the source of DSS command input. The source may be the integrated operator's console, a card reader, or a magnetic tape unit.

DUMP



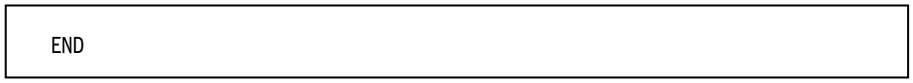
Dumps the contents of the data field.

ENABLE



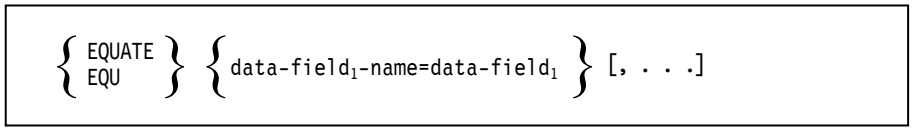
Activates ATs and ONs.

END



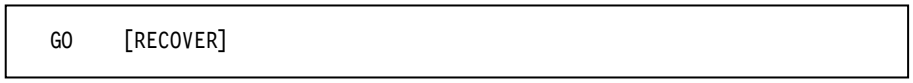
Ends text commands for the AT, ON, IF, and PROCEDURE commands.

EQUATE



Applies an alias to a data field defined by the DSS user or to a data field in OS/VS storage.

GO



Returns control to OS/VS with DSS monitoring. RECOVER simulates RTM restart.

GOTO

```
GOTO  command-label
```

Allows nonsequential execution of a command procedure.

IF

```
IF    expression
      text-commands
      END
```

Designates text commands whose execution depends on whether the expression is evaluated as true or false.

INVOKE

```
{INVOKE} procedure-name[,argument]. . .
{INV   }
```

Invokes a command procedure stored by a PROCEDURE command.

ON

```
ON  dss-function[, . . .][,&P([name][,SNAP][,DISABLE])]
    [command-procedure]
    END
```

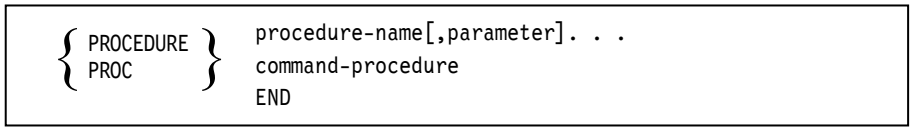
Specifies ON events where a SNAP dump and/or a command procedure will be executed.

PATCH

```
{PATCH} data-field1=data-field2[, . . .][,&P(name)]
{PAT  }
```

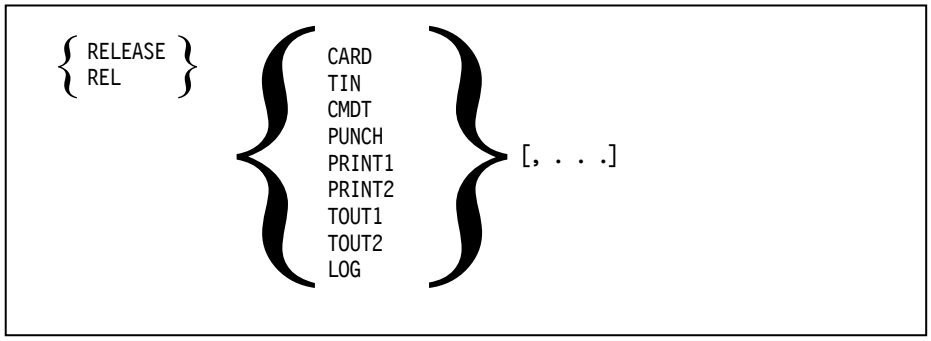
Changes the contents of data-field$_1$ and keeps a record of the original data. The length of data-field$_2$ determines the number of bytes to be changed.

PROCEDURE



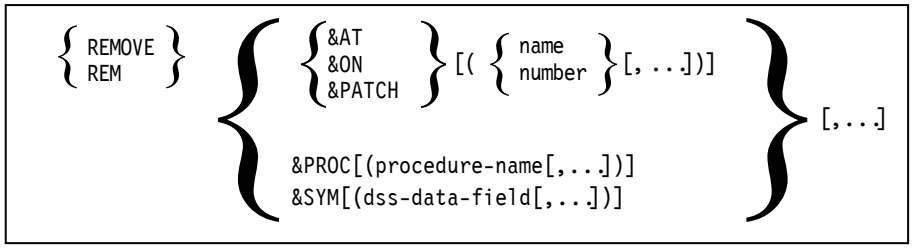
Stores a command procedure to be invoked later by an INVOKE command.

RELEASE



Closes a data set and returns the associated I/O device to OS/VS.

REMOVE

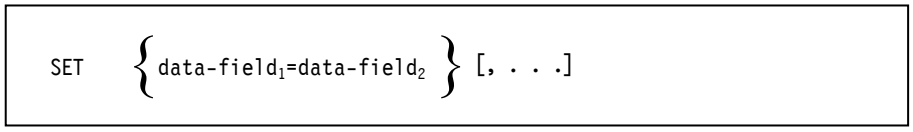


Deletes: ATs, ONs, and associated command procedures; command procedures stored by PROCEDURE; patches; aliases; and DSS-user-defined data fields.

RETURN

Terminates the execution of a command procedure and returns control to the point from which the command procedure was invoked.

SET



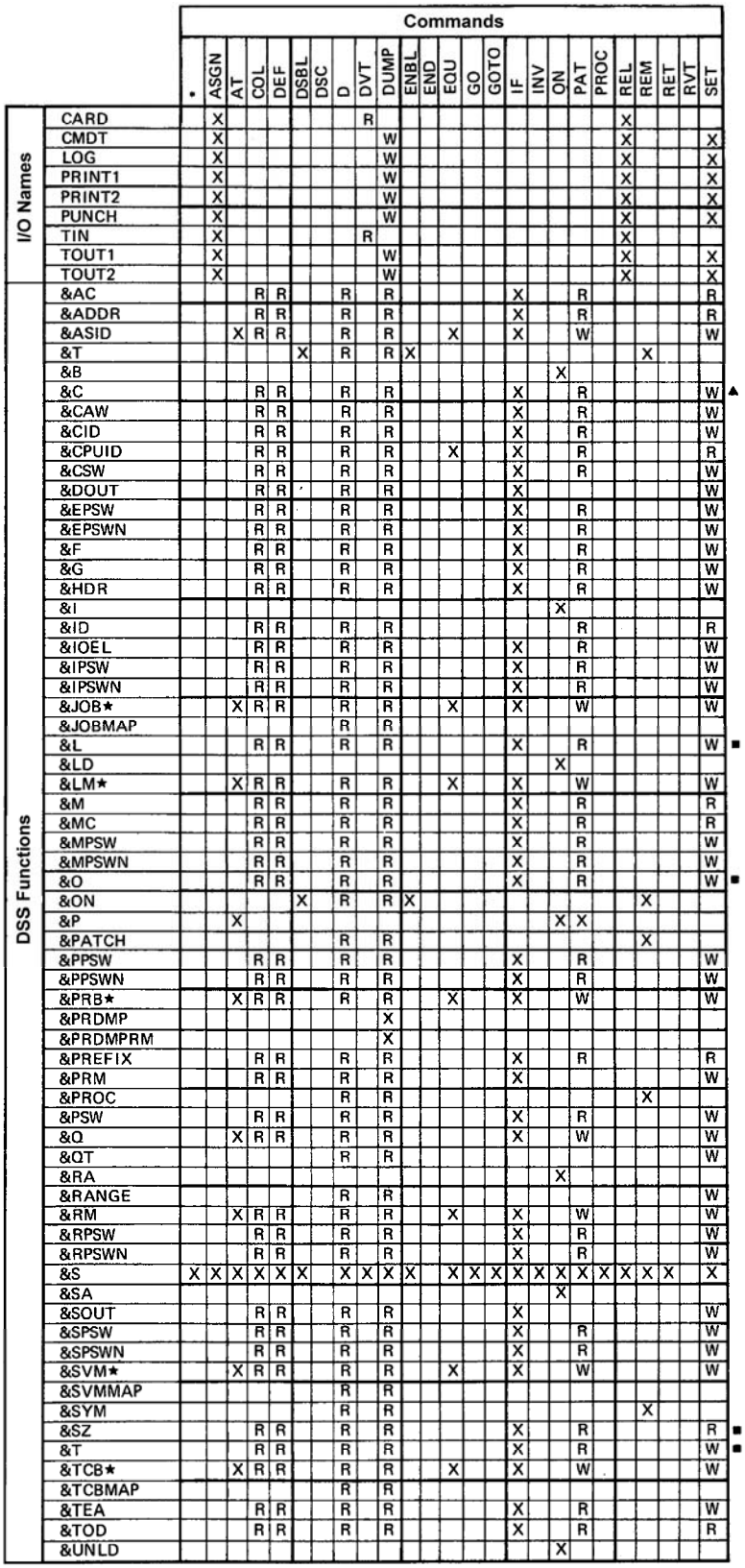
Changes the contents of data-field$_1$ without saving the original contents. The length of data-field$_1$ determines the number of bytes to be changed.

Table of Commands, I/O Names, and DSS Functions

Key to Table

X	Miscellaneous (Not Read or Write)
R	Read
W	Write or Read/Write
▲	Except that &C(9:11) must not be changed.
★	Must be part of a map expression.
■	Except that &O(y), &L(y), &T(y), and &SZ(y), where y is a DSS function, must not be changed.

Commands, I/O Names, and DSS Functions



		Commands																								
		*	ASGN	AT	COL	DEF	DSBL	DSC	D	DVT	DUMP	ENBL	END	EQU	GO	GOTO	IF	INV	ON	PAT	PROC	REL	REM	RET	RVT	SET
I/O Names	CARD		X							R												X				
	CMDT		X								W											X				X
	LOG		X								W											X				X
	PRINT1		X								W											X				X
	PRINT2		X								W											X				X
	PUNCH		X								W											X				X
	TIN		X							R												X				
	TOUT1		X								W											X				X
	TOUT2		X								W											X				X
DSS Functions	&AC				R	R			R		R						X			R						R
	&ADDR				R	R			R		R						X			R						R
	&ASID			X	R	R			R		R			X			X			W						W
	&T						X		R		R	X											X			
	&B																		X							
	&C				R	R			R		R						X			R						W
	&CAW				R	R			R		R						X			R						W
	&CID				R	R			R		R						X			R						W
	&CPUID				R	R			R		R			X			X			R						R
	&CSW				R	R			R		R						X			R						W
	&DOUT				R	R			R		R						X									W
	&EPSW				R	R			R		R						X			R						W
	&EPSWN				R	R			R		R						X			R						W
	&F				R	R			R		R						X			R						W
	&G				R	R			R		R						X			R						W
	&HDR				R	R			R		R						X			R						W
	&I																		X							
	&ID				R	R			R		R									R						R
	&IOEL				R	R			R		R						X			R						W
	&IPSW				R	R			R		R						X			R						W
	&IPSWN				R	R			R		R						X			R						W
	&JOB★			X	R	R			R		R			X			X			W						W
	&JOBMAP								R		R															
	&L				R	R			R		R						X			R						W
	&LD																		X							
	&LM★			X	R	R			R		R			X			X			W						W
	&M				R	R			R		R						X			R						R
	&MC				R	R			R		R						X			R						R
	&MPSW				R	R			R		R						X			R						W
	&MPSWN				R	R			R		R						X			R						W
	&O				R	R			R		R						X			R						W
	&ON						X		R		R	X											X			
	&P			X															X	X						
	&PATCH								R		R												X			
	&PPSW				R	R			R		R						X			R						W
	&PPSWN				R	R			R		R						X			R						W
	&PRB★			X	R	R			R		R			X			X			W						W
	&PRDMP										X															
	&PRDMPRM										X															
	&PREFIX				R	R			R		R						X			R						R
	&PRM				R	R			R		R						X									W
	&PROC								R		R												X			
	&PSW				R	R			R		R						X			R						W
	&Q			X	R	R			R		R						X			W						W
	&QT								R		R															W
	&RA																		X							
	&RANGE								R		R															W
	&RM			X	R	R			R		R			X			X			W						W
	&RPSW				R	R			R		R						X			R						W
	&RPSWN				R	R			R		R						X			R						W
	&S	X	X	X	X	X	X		X	X	X	X		X	X	X	X	X	X	X	X	X	X	X		X
	&SA																		X							
	&SOUT				R	R			R		R						X									W
	&SPSW				R	R			R		R						X			R						W
	&SPSWN				R	R			R		R						X			R						W
	&SVM★			X	R	R			R		R			X			X			W						W
	&SVMMAP								R		R															
	&SYM								R		R												X			
	&SZ				R	R			R		R						X			R						R
	&T				R	R			R		R						X			R						W
	&TCB★			X	R	R			R		R			X			X			W						W
	&TCBMAP								R		R															
	&TEA				R	R			R		R						X			R						W
	&TOD				R	R			R		R						X			R						R
	&UNLD																		X							

Activation Code (First Byte of &AC)

80	Module loaded (ON &LD)
40	Module unloaded (ON &UNLD)
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 interrupt (RESTART key)

Restart Key

<table>
<tr><th colspan="2">Under these conditions</th><th rowspan="2">pressing RESTART
gives control to</th></tr>
<tr><th>DSS</th><th>OS/VS2</th></tr>
<tr><td rowspan="3">dormant</td><td>quiesced</td><td>OS/VS2</td></tr>
<tr><td>in control
when
PSADSSGO=0</td><td>Recovery Termination Manager</td></tr>
<tr><td>in control
when
PSADSSGO≠0</td><td rowspan="3">DSS at the integrated operator's console</td></tr>
<tr><td>in control</td><td>dormant</td></tr>
<tr><td rowspan="2">monitoring</td><td>running</td></tr>
<tr><td>quiesced</td><td>OS/VS2</td></tr>
<tr><td>monitoring
and signaling
another CPU</td><td>running</td><td>• DSS at the integrated operator’s console
• RESTART is ignored
} Unpredictable; either could occur</td></tr>
</table>

DSS Examples

The following examples are of DSS procedures that can be read into the system at any time and then invoked as required.

The examples are also useful as a review of certain DSS facilities. Many of the statements could be entered whenever DSS is activated as individual instructions.

To read these or any procedures into the system, you must first activate DSS. Do this by storing any bit combination like X'FF' into the PSA+X'279'. RESTART will cause the Recovery Termination Manager to pass control to DSS and you will receive the message IQA1001 DSS READY and a prompt. Then type ASSGN CARD= 12; DVT CARD and end-of-block. (12 Is the card reader address in this example.)[1] Type in GO to return to VS2.

To invoke the procedures type INV proc-name and any necessary parameters whenever you have DSS active.

Example 1

This will cause OS/VS2 to stop when message IEF244I is issued.

These commands would invoke the procedure and return control to OS/VS:

```
INVOKE MSG,IEF244
GO
```

```
PROC MSG,NUMBER
DEF NUM.(0,6,C,6)=NUMBER                ;*SAVE THE PARM VALUE
SET &QT=&ASID(0)
  AT &LM(IGC0003E).(2)                  ;*STOP ON ENTRY TO WTO
    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
```

[1] If your card reader address is different, specify its address as either a decimal literal or as a hexadecimal literal (for example, 12 or X'00C'). DSS will accept either format.

Example 2

A command procedure finds and displays an OS/VS2 UCB (unit control block).

Example of an INVOKE command for the command procedure:

```
INVOKE FINDUCB,182
```

```
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
SET UCBPTR=TBLPTR%.(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 TBLPTR=TBLPTR+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 3

The following procedure writes all of real storage to a tape. The tape will be compatible with the AMDPRDMP service aid and can be completely formatted.

Only the tape address is variable.

Example of an INVOKE command for the procedure:

```
INV TDUMP
```

```
PROC TDUMP
  D 'REPLY RVT TO TAKE DUMP--OR GO TO SKIP';DVT
 ASGN TOUT1=X'482'
  SET &DOUT='TOUT1'
    DUMP &PRDMPRM
         RELEASE TOUT1
   D 'DUMP COMPLETE   REMOUNT NEW DUMP TAPE THEN ENTER GO'
  END
```

Example 4

The following procedure will print a virtual dump of any address space specified to a printer. All of the private area and all common storage except the LPA will be printed. This includes a Nucleus Map.

Example of an INVOKE command for the procedure:

```
INV PRINT,00E,7,TITLE
```

```
*DUMP ONE MEMORY, EXCEPT LPA, TO PRINTER
  PROC PRINT,PRT,ASID,DESC
  SET &QT=&ASID(&S(ASID))
  RELEASE PRINT1
  ASGN PRINT1=X'&S(PRT)'
  SET &DOUT='PRINT1'
  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'
END
```

Example 5

The following procedure can be used to trap storage alterations over a range of addresses and for a particular address space or all address spaces (ASID=0). When the trap is hit it will display messages, the range, the instruction altering storage, the module responsible, the registers, and the last trace table entry. Control will be passed to the operator.

To invoke the procedure, enter:

```
INV TRAPSA L'xxxxxx:yyyyyy',n
```

where `x` and `y` are the range limits and `n` is the ASID.

```
*TRAP 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
    D &AC
    D &ID(&AC.(2,3)%)
    D '**INSTRUCTION RESPONSIBLE**'
    IF &AC.(2,3)%.(.1)<X'C0'
      D &AC.(2,3)%.(,4)
      GOTO L00100
    END
    D &AC.(2,3)%.(,6)
    L00100: D '**CONTENTS OF ALTERED STORAGE**',&RANGE.(2,4)%:&RANGE.(6,4)%.(,1)
    D &G(0:15)
    D '**LAST TRACE TABLE ENTRY**',L'54'%%.(,32)
    DVT
  END
  RETURN
END
```

Example 6

This command procedure displays an ASID's ASCB and ASXB. To invoke the procedure, enter:

```
INV ASCB,7
```

Comment: This is to display the ASCB for ASID7.

```
PROC ASCB,ASID
   IF &S(ASID)=X'F0'
   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 FIELD TO POINT AT 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),A)
END
```

Example 7

These command procedures define a DSS symbol for an OS/VS2 direct-access device UCB, then make the device either shared or nonshared.

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
```

```
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
    IF X2%.(,1)=X'FF'                                ;       CHANNEL ADDRESS
    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
```

Example 8

The following command stream can be used to trap immediate program checks and optionally invoke TDUMP (see example 3). Any program check that would be an error will cause the display on the console of the current program old PSW, the module name, the translation exception address, the registers, and the last trace table entry. A dump will be taken and control passed to the operator.

The underlined addresses can vary from release to release and are defined as the BALR 14,15 (05EF) after the following five labels in module IEAVEPC: ERR, ERR3, DATERREP, IEAVRSET, TERMINAT. Placing the cards in the assigned reader and typing DVT CARD will put these ATs into the system. The INV TDUMP and DVT are optional statements.

```
*DEBUGGING ATS ON PROGRAM CHECKS
SET &QT=&ASID(0)
  AT IEAVEPC.(X'31C'),&P(PROGCHKA)                  ;*LABEL ERR
  D 'PROGRAM CHECK A'
  D &PPSW
D &ID(&PPSW.(4,4)%)
  D &TEA
  D '**REGISTERS AT PROGRAM CHECK**',L'210'%.(8,64)
  D '**LAST TRACE TABLE ENTRY**',L'54'%%.(,32)
INV TDUMP
  DVT
  END
  AT IEAVEPC.(X'25E'),&P(PROGCHKB)                  ;*LABEL ERR3
  D 'PROGRAM CHECK B'
  D &PPSW
D &ID(&PPSW.(4,4)%)
  D &TEA
  D '**REGISTERS AT PROGRAM CHECK**',L'210'%.(8,64)
  D '**LAST TRACE TABLE ENTRY**',L'54'%%.(,32)
  INV TDUMP
  DVT
  END
  AT IEAVEPC.(X'2A0'),&P(PROGCHKC)                  ;*LABEL DATERREP
  D 'PROGRAM CHECK C'
  D &PPSW
D &ID(&PPSW.(4,4)%)
  D &TEA
  D **REGISTERS AT PROGRAM CHECK**',L'210'%.(8,64)
  D **LAST TRACE TABLE ENTRY**',L'54'%%.(,32)
  INV TDUMP
  DVT
  END
```

Continued on next page

Continued from previous page

```
  AT IEAVEPC.(X'7AE'),&P(PROGCHKD)                 ;*LABEL IEAVRSET
  D 'PROGRAM CHECK D'
  D &PPSW
D &ID(&PPSW.(4,4)%)
  D &TEA
  D '**REGISTERS AT PROGRAM CHECK**',L'210'%.(8,64)
  D '**LAST TRACE TABLE ENTRY**',L'54'%%.(,32)
  INV TDUMP
  DVT
  END
  AT IEAVEPC.(X'600'),&P(PROGCHKE)                 ;*LABEL TERMINAT
  D 'PROGRAM CHECK F'
  D &PPSW
  D '**REGISTERS AT PROGRAM CHECK**',L'210'%.(8,64)
  D '**LAST TRACE TABLE ENTRY**',L'54'%%.(,32)
  INV TDUMP
  DVT
  END
```