F 5

```
   SSSSSSSSSSSSS DDDDDDDDDDDD        AAAAAAAAA
   SSSSSSSSSSSSS DDDDDDDDDDDD        AAAAAAAAA
   SSSSSSSSSSSSS DDDDDDDDDDDD        AAAAAAAAA
SSS              DDD         DDD  AAA         AAA
SSS              DDD         DDD  AAA         AAA
SSS              DDD         DDD  AAA         AAA
SSS              DDD         DDD  AAA         AAA
SSS              DDD         DDD  AAA         AAA
SSS              DDD         DDD  AAA         AAA
   SSSSSSSSS     DDD         DDD  AAA         AAA
   SSSSSSSSS     DDD         DDD  AAA         AAA
   SSSSSSSSS     DDD         DDD  AAA         AAA
            SSS  DDD         DDD  AAAAAAAAAAAAAAA
            SSS  DDD         DDD  AAAAAAAAAAAAAAA
            SSS  DDD         DDD  AAAAAAAAAAAAAAA
            SSS  DDD         DDD  AAA         AAA
            SSS  DDD         DDD  AAA         AAA
            SSS  DDD         DDD  AAA         AAA
SSSSSSSSSSSS     DDDDDDDDDDDD     AAA         AAA
SSSSSSSSSSSS     DDDDDDDDDDDD     AAA         AAA
SSSSSSSSSSSS     DDDDDDDDDDDD     AAA         AAA
```

```
  CCCCCCCC    OOOOOO    MM      MM  MM      MM    AAAAAA    NN      NN  DDDDDDDD      SSSSSSSS
  CCCCCCCC    OOOOOO    MM      MM  MM      MM    AAAAAA    NN      NN  DDDDDDDD      SSSSSSSS
CC          OO      OO  MMMM  MMMM  MMMM  MMMM  AA      AA  NN      NN  DD      DD  SS
CC          OO      OO  MMMM  MMMM  MMMM  MMMM  AA      AA  NN      NN  DD      DD  SS
CC          OO      OO  MM  MM  MM  MM  MM  MM  AA      AA  NNNN    NN  DD      DD  SS
CC          OO      OO  MM  MM  MM  MM  MM  MM  AA      AA  NNNN    NN  DD      DD  SS
CC          OO      OO  MM      MM  MM      MM  AA      AA  NN  NN  NN  DD      DD    SSSSSS
CC          OO      OO  MM      MM  MM      MM  AA      AA  NN  NN  NN  DD      DD    SSSSSS
CC          OO      OO  MM      MM  MM      MM  AAAAAAAAAA  NN    NNNN  DD      DD          SS
CC          OO      OO  MM      MM  MM      MM  AAAAAAAAAA  NN    NNNN  DD      DD          SS
CC          OO      OO  MM      MM  MM      MM  AA      AA  NN      NN  DD      DD          SS    ....
CC          OO      OO  MM      MM  MM      MM  AA      AA  NN      NN  DD      DD          SS    ....
  CCCCCCCC    OOOOOO    MM      MM  MM      MM  AA      AA  NN      NN  DDDDDDDD    SSSSSSSS      ....
  CCCCCCCC    OOOOOO    MM      MM  MM      MM  AA      AA  NN      NN  DDDDDDDD    SSSSSSSS      ....

LL            IIIIII      SSSSSSSS
LL            IIIIII      SSSSSSSS
LL              II      SS
LL              II      SS
LL              II      SS
LL              II      SS
LL              II        SSSSSS
LL              II        SSSSSS
LL              II              SS
LL              II              SS
LL              II              SS
LL              II              SS
LLLLLLLLLL    IIIIII    SSSSSSSS
LLLLLLLLLL    IIIIII    SSSSSSSS
```

Table of contents

```
0000            .TITLE  COMMANDS        PARSE AND EXECUTE SDA COMMANDS
0000            .SBTTL  COPYRIGHT NOTICE
0000            .IDENT  'V04-000'
0000    ;
0000    ;****************************************************************************
0000    ;*                                                                          *
0000    ;*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                                 *
0000    ;*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.                  *
0000    ;*  ALL RIGHTS RESERVED.                                                    *
0000    ;*                                                                          *
0000    ;*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED   *
0000    ;*  ONLY IN  ACCORDANCE WITH  THE  TERMS  OF  SUCH  LICENSE  AND WITH THE   *
0000    ;*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR  ANY  OTHER   *
0000    ;*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY   *
0000    ;*  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE IS  HEREBY   *
0000    ;*  TRANSFERRED.                                                            *
0000    ;*                                                                          *
0000    ;*  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE WITHOUT NOTICE   *
0000    ;*  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL EQUIPMENT   *
0000    ;*  CORPORATION.                                                            *
0000    ;*                                                                          *
0000    ;*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR  RELIABILITY OF ITS   *
0000    ;*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.                 *
0000    ;*                                                                          *
0000    ;*                                                                          *
0000    ;****************************************************************************
0000    ;
```

```
0000          .SBTTL  PROGRAM DESCRIPTION
0000  ;++
0000  ;  FACILITY
0000  ;
0000  ;       SYSTEM DUMP ANALYZER
0000  ;
0000  ;  ABSTRACT
0000  ;
0000  ;       THIS MODULE ACCEPTS COMMANDS FROM THE PRIMARY
0000  ;       INPUT DEVICE AND PERFORMS THE SYNTAX PARSING.
0000  ;       THE CORRECT ROUTINE IS THEN CALLED TO EXECUTE
0000  ;       THE COMMAND.
0000  ;
0000  ;  ENVIRONMENT
0000  ;
0000  ;       NATIVE MODE, USER MODE
0000  ;
0000  ;  AUTHOR
0000  ;
0000  ;       TIM HALVORSEN, JULY 1978
0000  ;
0000  ;  MODIFIED BY
0000  ;
0000  ;       V03-006 PRB0305          Paul Beck        10-Jan-1984  16:23
0000  ;               Add support for EXAMINE/INSTRUCTION/NOSKIP
0000  ;
0000  ;       V03-005 WMC0001          Wayne Cardoza            13-Jul-1983
0000  ;               Add EVAL/CONDITION.
0000  ;
0000  ;       V03-004 JLV0258          Jake VanNoy              23-MAY-1983
0000  ;               Replace $GET with call to SMG$ routine for keypad input.
0000  ;
0000  ;       V03-003 TMK0001          Todd M. Katz             21-Mar-1983
0000  ;               Implement the logging of interactive sessions. When a logging
0000  ;               file has been defined, and the output is going to the terminal
0000  ;               and not to a listing file, then log each command line to the log
0000  ;               file before processing it.
0000  ;
0000  ;       V03-002 CWH1002          CW Hobbs         2-Mar-1983
0000  ;               Save pix width in a local so that we can extract the
0000  ;               process index width from either an internal pid or an
0000  ;               extended pid without even checking which we have.
0000  ;
0000  ;       V03-001 JLV0221          Jake VanNoy              21-JAN-1983
0000  ;               Add Examine/PSL Examime/CONDITION_VALUE and Examine/TIME.
0000  ;               Repeat "escape" refreshes command line.
0000  ;
0000  ;       V007    MTR0002          Mike Rhodes      27-Jul-1981
0000  ;               Change SDA help to not prompt.
0000  ;
0000  ;       V006    MTR0001          Mike Rhodes      19-Jun-1981
0000  ;               Change all CMPW's referencing an MSG$_ symbol to CMPL's.
0000  ;
0000  ;               Change default addressing mode to longword.
0000  ;
0000  ;               1.  Remove references to macro $SDAMSGDEF.
0000  ;
```

```
0000    ;               2.  Modify routine GET_COMMANDS reference to condition
0000    ;               handler entry point (from CMD_HANDLER to HANDLER) for
0000    ;               standardized error handling.
0000    ;
0000    ;               3.  Rewrite routine DISPLAY_HELP to use the librarian
0000    ;               routine LBR$OUTPUT_HELP.
0000    ;
0000    ;       V005    TMH0005         Tim Halvorsen   21-May-1981
0000    ;               Remove READ/RELOCATE modifications in previous update,
0000    ;               since the relocation must be done within ADD_SYMBOL
0000    ;               based on whether the symbol already exists or not (we
0000    ;               can't relocate the values for symbols which already exist,
0000    ;               such as structure definitions).
0000    ;
0000    ;       V004    TMH0004         Tim Halvorsen   20-May-1981
0000    ;               Upcase all commands.  Add '@' and SEARCH commands.
0000    ;               Add READ/RELOCATE qualifier.  Increase size of symbolize
0000    ;               string buffer on EXAMINE command.
0000    ;
0000    ;       V003    TMH0003         Tim Halvorsen   07-Feb-1981
0000    ;               Add EXAMINE/INSTRUCTION.
0000    ;
0000    ;       V002    TMH0002         Tim Halvorsen   19-Dec-1980
0000    ;               Fix current process context so that it remembers /SYSTEM
0000    ;               from one SHOW PROCESS to another.
0000    ;
0000    ;       V001    TMH0001         Tim Halvorsen   30-Sep-1980
0000    ;               Use general addressing to reference LIB$SIGNAL
0000    ;--
```

```
0000          .SBTTL  DECLARATIONS
0000     ;
0000     ;        SYMBOL DEFINTIONS
0000     ;
0000          $TPADEF                         ; TPARSE DEFINITIONS
0000          $PCBDEF                         ; PROCESS CONTROL BLOCK
0000          $PHDDEF                         ; PROCESS HEADER DEFINITIONS
0000          $OPTDEF                         ; DEFINE BITS IN OPTIONS WORD
0000          $SHRDEF                         ; SHARED ERROR MESSAGES
0000          $DMPDEF                         ; DUMP HEADER DEFINITIONS
0000          $EMBDEF <CR>                    ; CRASH ERROR LOG ENTRY
0000          $FABDEF                         ; FILE ACCESS BLOCK
0000          $RABDEF                         ; RECORD ACCESS BLOCK
0000          $OBJDEF                         ; OBJECT MODULE DEFINITIONS
0000          $STSDEF                         ; MESSAGE FIELD DEFINITIONS
0000          $CHFDEF                         ; CONDITION HANDLER FIELD DEFINITIONS
0000          $SSDEF                          ; SYSTEM SERVICE CONDITION CODE DEF'S
0000          $HLPDEF                         ; HELP LIBRARY MASK DEFINITIONS
0000          $DVIDEF                         ; GETDVI DEFINITIONS
0000
```

```
                                      0000            .SBTTL  STORAGE DEFINITIONS
                                      0000      ;
                                      0000      ;       STORAGE DEFINITIONS
                                      0000      ;
                                      0000
                                  00000000            .PSECT  SDADATA,NOEXE,WRT
                                      0000
                         0000000E     0000  HELP_FLAGS:     .LONG   HLP$M_PROCESS!- ;Library search & Prompt options
                                      0004                          HLP$M_GROUP!-
                                      0004                          HLP$M_SYSTEM
45 48 24 53 59 53 0000000C'010E0000'  0004  HELP_LIBRARY:   .ASCID  /SYS$HELP:SDA.HLB/
      42 4C 48 2E 41 44 53 3A 50 4C   0012
                                      001C
                                      001C  TPARSE_BLOCK:
                         00000008     001C          .LONG   TPA$K_COUNT0                    ; LONGWORD IN BLOCK
                         00000002     0020          .LONG   TPA$M_ABBREV                    ; ABBREVIATIONS ALLOWED
                         00000040     0024          .BLKL   TPA$K_COUNT0-1                  ; REMAINDER OF BLOCK
                                      0040
                                      0040  ; STACK USED FOR EVALUATION OF EXPRESSIONS
                         0000010C'    0040  ESP::   .LONG   EXPR_STACK                      ; STACK POINTER
                         0000010C     0044          .BLKL   50
                                      010C  EXPR_STACK:                                     ; STARTING STACK POINTER
                                      010C
                                      010C  INPUT_RAB::
                         00000000     010C          .LONG   0                               ; ADDRESS OF CURRENT INPUT RAB
                                      0110                                                  ; INITIALLY 0
                                      0110                                                  ; (MAY BE CHANGED TO INDIRECT RAB)
                                      0110
                                      0110  OPTIONS::
                         00000114     0110          .BLKL   1                               ; OPTIONS FOR COMMAND
                                      0114
                                      0114  RELOCATE_BASE::
                         00000118     0114          .BLKL   1                               ; READ/RELOCATE BASE ADDRESS
                                      0118
                                      0118  PROC_NAME::
                         00000120     0118          .BLKQ   1                               ; PROCESS NAME
                                      0120
                                      0120  SYMBOL_NAME::
                         00000128     0120          .BLKQ   1                               ; SYMBOL NAME
                                      0128
                                      0128  FILE_DESC::
                         00000130     0128          .BLKQ   1                               ; FILE NAME DESCRIPTOR
                                      0130
                                      0130  PROC_INDEX::
                         FFFFFFFF     0130          .LONG   -1                              ; PROCESS INDEX
                                      0134
                                      0134  PIX_WIDTH::                                     ; USEFUL WIDTH OF PROCESS INDEX, SO
                         00000138     0134          .BLKL   1                               ; WE CAN INTERCHANGE IPIDS AND EPIDS
                                      0138
                                      0138  PROC_PID::
                         0000013C     0138          .BLKL   1                               ; PROCESS PID
                                      013C
                                      013C  CURRENT_PID:
                         00000140     013C          .BLKL   1                               ; PID OF PROCESS RUNNING DURING CRASH
                                      0140
                                      0140  ADDRESS::
                         00000000     0140          .LONG   0                               ; CURRENT MEMORY ADDRESS
```

```
                  0144
                  0144   DATALEN:
         00000000 0144           .LONG   0                         ; LENGTH OF PREVIOUS DATA ITEM
                  0148
                  0148   SYMBOL_DESC::
         00000150 0148           .BLKQ   1                         ; SYMBOL NAME TO BE DEFINED
                  0150
                  0150   LOG_BUFFER:
         00000055 0150   LOG_BUFFER_LENGTH = 85
   20 3E 41 44 53 0150           .ASCII  'SDA> '
         000001A5 0155           .BLKB   LOG_BUFFER_LENGTH-5
                  01A5
             00000000            .PSECT  COMMANDS,EXE,NOWRT,LONG
                  0000
                  0000   ;
                  0000   ; READ-ONLY STORAGE DEFINITIONS
                  0000   ;
                  0000
                  0000           .MACRO  NORMAL FIRST,LAST          ; MACRO TO DEFINE "NORMAL" INSTRUCTIONS
                  0000   .=NORMAL_INSTRS+FIRST
                  0000           .IF B   LAST                       ; IF LAST UNSPECIFIED, LOOP ONCE
                  0000           $$ = 1
                  0000           .IFF
                  0000           $$ = LAST+1-FIRST
                  0000           .ENDC
                  0000           .REPT   $$
                  0000           .BYTE   1
                  0000           .ENDR
                  0000           .ENDM
                  0000
                  0000   NORMAL_INSTRS:                             ; BYTEMASK OF REASONABLE INSTRUCTIONS
                  0000                                              ; (ITS ONLY A BYTEMASK CAUSE I DON'T
                  0000                                              ;  KNOW HOW TO CONSTRUCT THE BITMASK).
                  0000           .REPT   256/8
                  0000           .QUAD   0                          ; PRESET ALL TO "UNREASONABLE"
00000000 00000000 0000           .ENDR
                  0100
                  0100           .SAVE                              ; SAVE CURRENT "."
                  0100           NORMAL  4,5                        ; RET THRU RSB
                  0006           NORMAL  12,27                      ; PROBER THRU BLEQU
                  001C           NORMAL  30,31                      ; BGEQU THRU BLSSU
                  0020           NORMAL  40,41                      ; MOVC3 THRU CMPC3
                  002A           NORMAL  44,45                      ; MOVC5 THRU CMPC5
                  002E           NORMAL  48,50                      ; BSBW THRU CVTWL
                  0033           NORMAL  58                         ; LOCC
                  003B           NORMAL  60                         ; MOVZWL
                  003D           NORMAL  120                        ; ASHL
                  0079           NORMAL  124,139                    ; CLRQ THRU BICB3
                  008C           NORMAL  144,145                    ; MOVB THRU CMPB
                  0092           NORMAL  147,156                    ; BITB THRU ROTL
                  009D           NORMAL  158,171                    ; MOVAB THRU BICW3
                  00AC           NORMAL  176,183                    ; MOVW THRU DECW
                  00B8           NORMAL  186,187                    ; PUSHR THRU POPR
                  00BC           NORMAL  192,206                    ; ADDL2 THRU MNEGL
                  00CF           NORMAL  208,215                    ; MOVL THRU DECL
                  00D8           NORMAL  218,233                    ; MTPR THRU BLBC
                  00EA           NORMAL  236,240                    ; CMPV THRU INSV
```

```
    00F1   250        NORMAL  242,245                   ; AOBLSS THRU SOBGTR
    00F6   251        NORMAL  250,251                   ; CALLG THRU CALLS
    00FC   252        NORMAL  255                       ; BUGCHECK
00000100   253        .RESTORE
    0100   254
    0100   255        .DEFAULT DISPLACEMENT,LONG
```

```
                                  0100          .SBTTL  GET_COMMANDS -- ACCEPT AND EXECUTE COMMANDS
                                  0100  ;---
                                  0100  ;
                                  0100  ;       GET_COMMANDS
                                  0100  ;
                                  0100  ;       THIS ROUTINE ACCEPTS THE NEXT COMMAND FROM THE PRIMARY
                                  0100  ;       INPUT FILE, PARSES THE COMMAND AND EXECUTES IT.
                                  0100  ;
                                  0100  ; INPUTS:
                                  0100  ;
                                  0100  ;       NONE
                                  0100  ;
                                  0100  ; OUTPUTS:
                                  0100  ;
                                  0100  ;       R0 = TRUE IF COMMAND EXECUTED, FALSE IF EMPTY LINE
                                  0100  ;
                                  0100  ;---
                                  0100          .ENABL  LSB
                                  0100
                                  0100  PROMPT:
                        00000005  0100  PROMPT_LEN = 5
            20 3E 41 44 53 0A 00' 0100          .ASCIC  <10>'SDA> '
                              06  0100
                                  0107
                                  0107  GET_COMMANDS::
                            000C  0107          .WORD   ^M<R2,R3>
                                  0109
         6D   00000000'EF     DE  0109          MOVAL   HANDLER,(FP)            ; SETUP CONDITION HANDLER
         53   0000001C'EF     9E  0110          MOVAB   TPARSE_BLOCK,R3         ; SET ADDRESS OF TPARSE BLOCK
                                  0117  ;       WARNING -- THE HANDLER KNOWS ABOUT THE FOLLOWING 2 LINES
              00000110'EF     DD  0117          PUSHL   OPTIONS                 ; SAVE OPTIONS LONGWORD
         7E   00000000'EF     7D  011D          MOVQ    SUB_HEADING,-(SP)       ; SAVE CURRENT HEADING
                                  0124
              00000000'EF     D4  0124          CLRL    LINE_COUNT              ; 1ST PAGE EJECT W/O PROMPT
              0000010C'EF     D5  012A          TSTL    INPUT_RAB               ; indirect?
                       59     12  0130          BNEQ    20$
              00000000'EF     16  0132          JSB     GET_INPUT               ; Get a line of input
         50   00000000'8F     D1  0138          CMPL    #SMG$_EOF,R0            ; CHECK IF END OF FILE
                       12     12  013F          BNEQ    10$                     ; BRANCH IF NOT
                                  0141          SIGNAL  0,EOF                   ; SIGNAL END OF FILE
                                  0153  10$:
                                  0153          SIGNAL                          ; CHECK FOR OTHER ERRORS
                       7D     11  015F          BRB     45$
                                  0161  ;
                                  0161  ;       IF THE ESCAPE KEY WAS TYPED, AND LINE EDITING IS OFF,
                                  0161  ;       REPEAT THE LAST COMMAND.
                                  0161  ;
 1E 00000000'EF   00000000'8F     E1  0161          BBC     #DEV$V_TRM,DVI_DEVCHAR,20$ ; BR IF NOT TERMINAL
                  00000000'8F     E0  016D          BBS     #TT2$V_EDITING,-
               12 00000000'EF         0173                  DVI_DEVDEPND2,20$       ; BR IF EDITING
              1B      0C A2       B1  0179          CMPW    RAB$L_STV(R2),#^X1B     ; IS TERMINATOR ESCAPE KEY?
                          5F      12  017D          BNEQ    45$                     ; BRANCH IF NOT
      00000000'EF         63      FA  017F          CALLG   (R3),REPEAT_COMMAND     ; RESTORE PREV. COMMAND
                        0AAE      30  0186          BSBW    ECHO                    ; ECHO IF SCOPE
                          6A      11  0189          BRB     50$                     ; AND EXECUTE IT
                                  018B  ;
                                  018B  ; Use indirect file
```

```
                                018B     ;
                                018B 20$:
       52    0000010C'EF   D0   018B         MOVL    INPUT_RAB,R2               ; Set rab address
    24 A2    00000000'EF   9E   0192         MOVAB   INPUT_BUFFER,RAB$L_UBF(R2)
      20 A2      0000'8F   B0   019A         MOVW    #INPUT_BUF_LEN,RAB$W_USZ(R2)
                                01A0         $GET    (R2)
 00000000'EF       22 A2   B0   01A9         MOVW    RAB$W_RSZ(R2),INPUT_LEN    ; Set length
       50    00000000'8F   D1   01B1         CMPL    #RMS$_EOF,R0               ; CHECK IF END OF FILE
                      12   12   01B8         BNEQ    30$                        ; BRANCH IF NOT
                                01BA         SIGNAL  0,EOF                      ; SIGNAL END OF FILE
                                01CC 30$:
                                01CC         SIGNAL  RMS,(R2)                   ; CHECK FOR OTHER ERRORS
                                01DE     ;
                                01DE     ;       PARSE THE COMMAND
                                01DE     ;
    0C A3    00000000'EF   9E   01DE 45$:    MOVAB   INPUT_BUFFER,TPA$L_STRINGPTR(R3)
    08 A3    00000000'EF   3C   01E6         MOVZWL  INPUT_LEN,TPA$L_STRINGCNT(R3)
                      05   12   01EE         BNEQ    50$                        ; BRANCH IF NOT EMPTY
                      50   D4   01F0         CLRL    R0                         ; INDICATE EMPTY LINE
                    00CF   31   01F2         BRW     90$                        ; EXIT
                                01F5 50$:
             00000000'EF   D5   01F5         TSTL    OUTPUT_FILE                ; SKIP LOGGING IF RMS IS OUTPUTTING
                      4A   12   01FB         BNEQ    55$                        ; TO LISTING FILE OR LOGGING FILE
             00000000'EF   D5   01FD         TSTL    LOG_FILE                   ; HAS NOT BEEN DEFINED
                      42   13   0203         BEQL    55$
                                0205
                 007C 8F   BB   0205         PUSHR   #^M<R2,R3,R4,R5,R6>        ; SAVE SOME REGISTERS
       56    00000000'EF   9E   0209         MOVAB   LOGRAB,R6                  ; LOGGING FILE RAB
    28 A6    00000150'EF   9E   0210         MOVAB   LOG_BUFFER,RAB$L_RBF(R6); COMMAND INPUT LINE LOGGING BUFFER
                      05   A1   0218         ADDW3   #PROMPT_LEN,-              ; LENGTH OF LOGGING I/O IS SIZE OF
                   08 A3        021A                 TPA$L_STRINGCNT(R3),-      ; PROMPT PLUS SIZE OF COMMAND INPUT LINE
                   22 A6        021C                 RAB$W_RSZ(R6)
                   08 A3   28   021E         MOVC3   TPA$L_STRINGCNT(R3),-      ; APPEND THE COMMAND INPUT LINE AFTER
                   0C B3        0221                 @TPA$L_STRINGPTR(R3),-     ; THE PROMPT IN THE LOGGING BUFFER
             00000155'EF        0223                 LOG_BUFFER+PROMPT_LEN
                                0228         $PUT    (R6)                       ; LOG THE COMMAND INPUT LINE
                                0231         SIGNAL  RMS,(R6)                   ; SIGNAL ANY ERRORS
                 007C 8F   BA   0243         POPR    #^M<R2,R3,R4,R5,R6>        ; RESTORE SAVE REGISTERS
                                0247
                   08 A3   9F   0247 55$:    PUSHAB  TPA$L_STRINGCNT(R3)        ; ADDRESS OF INPUT STRING DESCRIPTOR
                      6E   DD   024A         PUSHL   (SP)                       ; OVERWRITE STRING WITH RESULT
      00000000'GF     02   FB   024C         CALLS   #2,G^STR$UPCASE            ; UPCASE COMMAND LINE
             00000110'EF   D4   0253         CLRL    OPTIONS                    ; INITIALIZE OPTIONS LONGWORD
             00000114'EF   D4   0259         CLRL    RELOCATE_BASE              ; INITIALIZE RELOCATION BASE
 00000040'EF 0000010C'EF   DE   025F         MOVAL   EXPR_STACK,ESP             ; INITIALIZE EXPRESSION STACK
              7E   08 A3   7D   026A         MOVQ    TPA$L_STRINGCNT(R3),-(SP)  ; SAVE DESCRIPTOR OF COMMAND
             00000000'EF   DF   026E         PUSHAL  L^SDA_KEY                  ; KEY TABLE ADDRESS
             00000000'EF   DF   0274         PUSHAL  L^SDA_STATE                ; STATE TABLE ADDRESS
                      53   DD   027A         PUSHL   R3                         ; TPARSE BLOCK
      00000000'GF     03   FB   027C         CALLS   #3,G^LIB$TPARSE            ; PARSE AND EXECUTE THE COMMAND
             00000000'EF   D4   0283         CLRL    HEADING_ROUTINE            ; CLEAR HEADING ROUTINE ADDRESS
              08 A3   8E   7D   0289         MOVQ    (SP)+,TPA$L_STRINGCNT(R3)  ; RESTORE COMMAND DESCRIPTOR
                   2D 50   E8   028D         BLBS    R0,80$                     ; BRANCH IF NO ERROR
                                0290     ;
                                0290     ;       SIGNAL SYNTAX ERROR FROM PARSE
                                0290     ;
    51    50    10    10   EF   0290         EXTZV   #16,#16,R0,R1              ; CHECK FACILITY CODE
```

```
                    0E    12  0295   370          BNEQ    60$                          ; BRANCH IF NOT SDA ERROR MSG
                              0297   371          SIGNAL                               ; SIGNAL ERROR CONDITION
                    18    11  02A3   372          BRB     80$
                              02A5   373  60$:
     50    000310F8 8F    D0  02A5   374          MOVL    #SHR$_SYNTAX!<3@16>,R0       ; CLI-W-SYNTAX MESSAGE
                   10 A3  9F  02AC   375          PUSHAB  TPA$L_TOKENCNT(R3)           ; DESCRIPTOR OF BAD TOKEN
                              02AF   376          SIGNAL  1                            ; SIGNAL WITH 1 FAO ARG.
                              02BD   377  80$:
                              02BD   378          STATUS  SUCCESS                      ; COMMAND EXECUTED
                              02C4   379  90$:
     00000000'EF  8E      7D  02C4   380          MOVQ    (SP)+,SUB_HEADING            ; RESTORE SUB_HEADING
          00000110'EF   8ED0  02CB   381          POPL    OPTIONS                      ; RESTORE OPTIONS WORD
                          04  02D2   382          RET
                              02D3   383
                              02D3   384          .DSABL  LSB
```

```
                               02D3        .SBTTL  INDIRECT_COMMAND --INDIRECT FILE PROCESSING
                               02D3   ;+
                               02D3   ;        INDIRECT_COMMAND
                               02D3   ;
                               02D3   ; INPUTS:
                               02D3   ;
                               02D3   ;        AP = ADDRESS OF TPARSE CONTROL BLOCK
                               02D3   ;
                               02D3   ; OUTPUTS:
                               02D3   ;
                               02D3   ;        ACTION ROUTINE TO OPEN AN INDIRECT COMMAND FILE
                               02D3   ;        AND PROCESS THE COMMANDS.
                               02D3   ;-
                         003C  02D3        .ENTRY  INDIRECT_COMMAND, ^M<R2, R3, R4, R5>
                               02D5
     53   00000000'EF    9E    02D5        MOVAB   INDRAB, R3                   ; R3 = ADDRESS OF RAB
             52   3C A3  D0    02DC        MOVL    RAB$L_FAB(R3),R2             ; R2 = ADDRESS OF FAB
     53   0000010C'EF    D1    02E0        CMPL    INPUT_RAB, R3                ; ONLY ONE LEVEL OF INDIRECTION
                     37  13    02E7        BEQL    10$                          ; BR IF RECURSIVE AND EXIT
                               02E9
                  10 AC  33    02E9        CVTWB   TPA$L_TOKENCNT(AP),-
                  34 A2        02EC                FAB$B_FNS(R2)                ; SET FILE NAME LENGTH
                  14 AC  D0    02EE        MOVL    TPA$L_TOKENPTR(AP),-
                  2C A2        02F1                FAB$L_FNA(R2)                ; SET FILE NAME ADDRESS
                               02F3
                               02F3        $OPEN   (R2)                         ; ATTEMPT TO OPEN THE FILE
                  24 50  E8    02FC        BLBS    R0, 20$                      ; BRANCH IF OK
                  2C A2  DD    02FF        PUSHL   FAB$L_FNA(R2)                ; CREATE DESCRIPTOR OF FILE NAME
             7E   34 A2  9A    0302        MOVZBL  FAB$B_FNS(R2), -(SP)
                 7E   50  7D   0306        MOVQ    R0, -(SP)                    ; RMS ERROR CODES
                     00  F0    0309        INSV    #STS$K_WARNING,-             ; CHANGE SEVERITY TO WARNING
                     00        030B                #STS$V_SEVERITY,-
                     03        030C                #STS$S_SEVERITY,-
                     6E        030D                (SP)
                  08 AE  9F    030E        PUSHAB  8(SP)                        ; ADDRESS OF FILE NAME DESCRIPTOR
                     01  DD    0311        PUSHL   #1                           ; NUMBER OF FAO ARGUMENTS
            00031098 8F  DD    0313        PUSHL   #SHR$_OPENIN!<3@16>          ; "ERROR OPENING INPUT FILE !AS"
         00000000'GF  05  FB   0319        CALLS   #5, G^LIB$SIGNAL             ; OUTPUT ERROR MESSAGE
                  00A4  31     0320   10$: BRW     50$                          ; EXIT WITH SUCCESS
                               0323
                               0323   20$: $CONNECT (R3)                        ; CONNECT FOR RECORD ACCESS
                               032C        SIGNAL  RMS, (R3)                    ; REPORT ANY ERRORS
                               033E
         00000000'EF  6C  FA   033E        CALLG   (AP),REPEAT_COMMAND          ; GET DESCRIPTOR OF @ COMMAND
             5E   08 AC  C2    0345        SUBL    TPA$L_STRINGCNT(AP),SP       ; ALLOCATE BUFFER TO HOLD @ COMMAND
                     0C  BB    0349        PUSHR   #^M<R2,R3>
 08 AE    0C BC   08 AC  28    034B        MOVC    TPA$L_STRINGCNT(AP),@TPA$L_STRINGPTR(AP),8(SP) ; PUSH ONTO STACK
                     0C  BA    0352        POPR    #^M<R2,R3>
                  08 AC  DD    0354        PUSHL   TPA$L_STRINGCNT(AP)          ; SAVE LENGTH OF @ COMMAND STRING
         0000010C'EF     DD    0357        PUSHL   INPUT_RAB                    ; SAVE RAB ACROSS PROCEDURE
         0000010C'EF  53  D0   035D        MOVL    R3, INPUT_RAB                ; SETUP INDIRECT FILES RAB AS INPUT
                               0364   30$:
         00000000'EF     D4    0364        CLRL    LINE_COUNT                   ; AVOID END OF PAGE PROMPTS
            FD98 CF   00  FB   036A        CALLS   #0, GET_COMMANDS             ; ACCEPT AND EXECUTE COMMANDS
         00000000'8F  50  D1   036F        CMPL    R0, #MSG$_EOF                ; CHECK IF END OF FILE
                     1E  13    0376        BEQL    40$                          ; BRANCH IF SO
         00000000'8F  50  D1   0378        CMPL    R0, #MSG$_EXITCMD            ; DID WE JUST EXIT A LEVEL?
```

```
                E3   12  037F        BNEQ    30$                     ; BRANCH IF NOT
       00000000'EF   7C  0381        CLRQ    SUB_HEADING             ; CLEAR CURRENT HEADING
       00000000'EF   D4  0387        CLRL    HEADING_ROUTINE         ; CLEAR HEADING ROUTINE ADDRESS
                         038D        SKIP    PAGE                    ; ERASE PREVIOUS JUNK
                8D   11  0394        BRB     20$
                         0396 40$:
       0000010C'EF 8ED0  0396        POPL    INPUT_RAB               ; RESTORE RAB ADDRESS
             08 AC 8ED0  039D        POPL    TPA$L_STRINGCNT(AP)     ; RESTORE LENGTH OF @ STRING
          0C AC   5E D0  03A1        MOVL    SP,TPA$L_STRINGPTR(AP)  ; SET ADDRESS OF COPY OF STRING
   00000000'EF   6C   FA  03A5        CALLG   (AP),SAVE_COMMAND       ; SAVE AS PREVIOUS COMMAND
                         03AC        $CLOSE  FAB=(R2)                ; CLOSE THE COMMAND FILE
                         03B5        SIGNAL  RMS,(R2)                ; REPORT ANY ERRORS
                         03C7 50$:
          50      01  D0  03C7        MOVL    #1, R0                  ; RETURN SUCCESS
                      04  03CA        RET
```

```
       03CB          .SBTTL  EXIT_COMMAND -- PROCESS EXIT COMMAND
       03CB  ;---
       03CB  ;
       03CB  ;       EXIT_COMMAND
       03CB  ;
       03CB  ;       EXIT COMMAND ACTION ROUTINE
       03CB  ;
       03CB  ;    OUTPUTS:
       03CB  ;
       03CB  ;       END OF FILE IS SIGNALED
       03CB  ;
       03CB  ;---
       03CB
       03CB  EXIT_COMMAND::
 0000  03CB          .WORD   0
       03CD
       03CD          SIGNAL  0,EOF                         ; SIGNAL END OF FILE
```

```
                              03DF        .SBTTL  DEFINE_SYMBOL -- DEFINE LOCAL SYMBOL
                              03DF   ;---
                              03DF   ;
                              03DF   ;        DEFINE_SYMBOL
                              03DF   ;
                              03DF   ;        THIS ROUTINE ADDS A SYMBOL TO THE LOCAL SYMBOL
                              03DF   ;        TABLE WITH AN ASSOCIATED VALUE.
                              03DF   ;
                              03DF   ;    INPUTS:
                              03DF   ;
                              03DF   ;        AP = ADDRESS OF TPARSE PARAMETER BLOCK
                              03DF   ;        SYMBOL_DESC = DESCRIPTOR OF SYMBOL
                              03DF   ;        TPA$L_NUMBER(AP) = VALUE TO BE ASSIGNED TO SYMBOL
                              03DF   ;
                              03DF   ;---
                              03DF
                              03DF   DEFINE_SYMBOL::
                        0000  03DF        .WORD   0
                              03E1
                  1C AC   DD  03E1        PUSHL   TPA$L_NUMBER(AP)         ; VALUE OF SYMBOL
        7E    00000148'EF  7D  03E4        MOVQ    SYMBOL_DESC,-(SP)        ; SYMBOL DESCRIPTOR
        00000000'EF    03   FB  03EB        CALLS   #3,ADD_SYMBOL            ; ADD TO SYMBOL TABLE
                          04  03F2        RET
```

```
                                  03F3          .SBTTL  SHOW_SYMBOL -- DISPLAY SYMBOL VALUE
                                  03F3   ;---
                                  03F3   ;
                                  03F3   ;      SHOW_SYMBOL
                                  03F3   ;
                                  03F3   ;      THIS ROUTINE DISPLAYS THE VALUE OF A SPECIFIED SYMBOL
                                  03F3   ;      AND THE CONTENTS OF THAT MEMORY LOCATION IF POSSIBLE.
                                  03F3   ;
                                  03F3   ; INPUTS:
                                  03F3   ;
                                  03F3   ;      SYMBOL_NAME = DESCRIPTOR OF SYMBOL
                                  03F3   ;
                                  03F3   ; OUTPUTS:
                                  03F3   ;
                                  03F3   ;      NONE
                                  03F3   ;
                                  03F3   ;---
                                  03F3
                                  03F3          .ENABL  LSB
                                  03F3
                                  03F3   SHOW_SYMBOL::
                            0004  03F3          .WORD   ^M<R2>
                                  03F5
         52  00000120'EF    9E    03F5          MOVAB   SYMBOL_NAME,R2          ; ADDRESS OF SYMBOL NAME DESC.
00000FFF 8F  00000110'EF    D1    03FC          CMPL    OPTIONS,#OPT$M_ALL      ; CHECK IF WANTS ALL SYMBOLS
                       6B   13    0407          BEQL    70$                     ; BRANCH IF SO
                       52   DD    0409          PUSHL   R2                      ; DESCRIPTOR
         00000000'EF   01   FB    040B          CALLS   #1,SYMBOL_VALUE         ; GET SYMBOL VALUE
                    4A 50   E9    0412          BLBC    R0,50$                  ; IF NOT FOUND
         C0000000 8F   51   D1    0415          CMPL    R1,#^XC0000000          ; CHECK IF INTERNAL SYMBOL
                       41   1E    041C          BGEQU   50$                     ; DO NOT SHOW INTERNAL SYMBOLS
         00000140'EF   51   D0    041E          MOVL    R1,ADDRESS              ; SAVE SYMBOL VALUE
                                  0425          TRYMEM  (R1)                    ; GET CONTENTS OF LOCATION
                    18 50   E9    042E          BLBC    R0,20$                  ; IF NOT AN ADDRESS
                       51   DD    0431          PUSHL   R1                      ; CONTENTS OF LOCATION
              00000140'EF   DD    0433          PUSHL   ADDRESS                 ; SYMBOL VALUE
                       52   DD    0439          PUSHL   R2                      ; DESCRIPTOR ADDRESS
                                  043B          PRINT   3,<!AS = !XL :  !XL>
                            04    0448          RET
                                  0449   20$:
              00000140'EF   DD    0449          PUSHL   ADDRESS                 ; SYMBOL VALUE
                       52   DD    044F          PUSHL   R2                      ; DESCRIPTOR ADDRESS
                                  0451          PRINT   2,<!AS = !XL>
                            04    045E          RET
                                  045F   50$:
                       52   DD    045F          PUSHL   R2                      ; DESCRIPTOR ADDRESS
                                  0461          SIGNAL  1,BADSYM                ; UNKNOWN SYMBOL
                            04    0473          RET
                                  0474   70$:
         00000000'EF   62   FA    0474          CALLG   (R2),PRINT_SYMBOLS      ; PRINT ALL SYMBOLS
                            04    047B          RET
                                  047C
                                  047C          .DSABL  LSB
```

```
                              047C           .SBTTL  SHOW_EXPR -- SHOW RESULT OF EVALUATE COMMAND
                              047C   ;---
                              047C   ;
                              047C   ;       SHOW_EXPR
                              047C   ;
                              047C   ;       DISPLAY THE RESULT OF AN EVALUATE COMMAND.
                              047C   ;
                              047C   ;  INPUTS:
                              047C   ;
                              047C   ;       TPA$L_NUMBER(AP) = RESULT OF EXPRESSION
                              047C   ;
                              047C   ;  OUTPUTS:
                              047C   ;
                              047C   ;       RESULT IS DISPLAYED.
                              047C   ;
                              047C   ;---
                              047C
                              047C   SHOW_EXPR::
                       0000   047C           .WORD   0
                              047E           .ENABL  LSB
  11 00000110'EF   08    E0   047E           BBS     #OPT$V_COND,OPTIONS,10$ ; IS IT /COND
              1C AC      DD   0486           PUSHL   TPA$L_NUMBER(AP)        ; GET RESULT OF EXPRESSION
                              0489           PRINT   1,<Hex = !XL   Decimal = !-!SL>
                         04   0496           RET
                              0497
                              0497   10$:    ALLOC   110,R2                  ; GET A BUFFER
                              04A9           $GETMSG_S -                     ; GET THE MESSAGE
                              04A9                   MSGID = TPA$L_NUMBER(AP),-
                              04A9                   MSGLEN  = (R2),-
                              04A9                   BUFADR = (R2)
              10 50      E9   04BB           BLBC    R0,20$
                 52      DD   04BE           PUSHL   R2
                              04C0           PRINT   1,<!AS>
                         04   04CD           RET
                              04CE
                 50      DD   04CE   20$:    PUSHL   R0                      ; ERROR
     00000000'GF   01    FB   04D0           CALLS   #1,G^LIB$SIGNAL
                         04   04D7           RET
                              04D8           .DSABL  LSB
```

```
                                04D8          .SBTTL  EXAM_MEMORY -- EXAMINE MEMORY LOCATIONS
                                04D8    ;---
                                04D8    ;
                                04D8    ;       EXAM_MEMORY
                                04D8    ;
                                04D8    ;       DISPLAY THE CONTENTS OF A SPECIFIED RANGE OF MEMORY
                                04D8    ;
                                04D8    ;    INPUTS:
                                04D8    ;
                                04D8    ;       EXPRESSION STACK CONTAINS ARGUMENTS OF RANGE
                                04D8    ;
                                04D8    ;    OUTPUTS:
                                04D8    ;
                                04D8    ;       NONE
                                04D8    ;
                                04D8    ;---
                                04D8
                                04D8          .ENABL  LSB
                                04D8
                                04D8    EXAM_MEMORY::
                        003C    04D8          .WORD   ^M<R2,R3,R4,R5>
                                04DA
       52   00000110'EF   D0    04DA          MOVL    OPTIONS,R2
  50   52   000003F8 8F   CB    04E1          BICL3   #OPT$M_RANGE!-
                                04E9                  OPT$M_LENGTH!-
                                04E9                  OPT$M_INST!-
                                04E9                  OPT$M_NOSKIP!-
                                04E9                  OPT$M_PSL!-
                                04E9                  OPT$M_COND!-
                                04E9                  OPT$M_TIME,R2,R0        ; CHECK IF ANY OPTIONS SPECIFIED
                  22      13    04E9          BEQL    40$                     ; BRANCH IF NOT
               07 52  00  E1    04EB          BBC     #OPT$V_P0,R2,10$        ; CHECK IF /P0 SPECIFIED
          00000000'EF 00  FB    04EF          CALLS   #0,SHOW_P0              ; SHOW P0 SPACE
               07 52  01  E1    04F6    10$:  BBC     #OPT$V_P1,R2,20$        ; CHECK IF /P1 SPECIFIED
          00000000'EF 00  FB    04FA          CALLS   #0,SHOW_P1              ; SHOW P1 SPACE
               07 52  02  E1    0501    20$:  BBC     #OPT$V_SYSTEM,R2,30$    ; CHECK IF /SYSTEM SPECIFIED
          00000000'EF 00  FB    0505          CALLS   #0,SHOW_SYSTEM          ; SHOW SYSTEM SPACE
                          04    050C    30$:  RET
                                050D
                                050D    40$:
       53   00000040'EF   D0    050D          MOVL    ESP,R3                  ; EXPRESSION STACK POINTER
               22 52  03  E0    0514          BBS     #OPT$V_RANGE,R2,50$     ; BRANCH IF RANGE SPECIFIED
               29 52  04  E0    0518          BBS     #OPT$V_LENGTH,R2,55$    ; BRANCH IF LENGTH SPECIFIED
       50   0000010C'EF   9E    051C          MOVAB   EXPR_STACK,R0
                  50  53  D1    0523          CMPL    R3,R0                   ; CHECK IF STACK EMPTY
                      5A  13    0526          BEQLU   60$                     ; IF NO ARGS, USE NEXT LONGWORD
          C0000000 8F 63  D1    0528          CMPL    (R3),#^XC0000000        ; CHECK IF INTERNAL REGISTER
                      5F  1E    052F          BGEQU   70$                     ; DO NOT SET ADDRESS IF SO
          00000140'EF 63  D0    0531          MOVL    (R3),ADDRESS            ; SET NEW "CURRENT" ADDRESS
                      56  11    0538          BRB     70$
                                053A    50$:
                      7E  D4    053A          CLRL    -(SP)                   ; DISABLE PAGING
       7E   63     04 A3  C3    053C          SUBL3   4(R3),(R3),-(SP)        ; LENGTH - 1 OF RANGE
                      6E  D6    0541          INCL    (SP)                    ; LENGTH TO DUMP
                      04  11    0543          BRB     58$
                                0545    55$:
                      7E  D4    0545          CLRL    -(SP)                   ; DISABLE PAGING
```

```
                  63  DD  0547          PUSHL   (R3)                            ; LENGTH TO DUMP
                          0549  58$:
               04 A3  DD  0549          PUSHL   4(R3)                           ; STARTING ADDRESS
   00000140'EF    6E  D0  054C          MOVL    (SP),ADDRESS                    ; SET NEW "CURRENT" ADDRESS
      00000000'EF     D5  0553          TSTL    OUTPUT_FILE                     ; AVOID SUBHEADNG IF NOT A FILE
                  14  13  0559          BEQL    59$                             ; BRANCH IF NOT
                          055B          SUBHD   <MEMORY DUMP>
                          0568          SKIP    PAGE
08 00000110'EF    05  E0  056F  59$:    BBS     #OPT$V_INST,OPTIONS,57$         ; BRANCH IF RANGE OF INSTRUCTIONS
   00000000'EF    03  FB  0577          CALLS   #3,DUMP                         ; DUMP MEMORY
                      04  057E          RET
                01A6  31  057F  57$:    BRW     150$                            ; GO SHOW RANGE OF DATA ITEMS
                          0582
                          0582  ;
                          0582  ; NO ADDRESS SPECIFIED - SKIP TO NEXT DATA ITEM
                          0582  ;
   53   00000140'EF   DE  0582  60$:    MOVAL   ADDRESS,R3
   63   00000144'EF   C0  0589          ADDL    DATALEN,(R3)                    ; SET "." TO NEXT DATA SEGMENT
                          0590  ;
                          0590  ; DISPLAY DATA AT "ADDRESS".
                          0590  ;
        00000000'EF   D4  0590  70$:    CLRL    LINE_COUNT                      ; INHIBIT PAGE EJECTS
   55   00000000'EF   9E  0596          MOVAB   BUFFER,R5                       ; ADDRESS OF INSTRUCTION/DATA BUFFER
                          059D          ALLOC   110,R2                          ; ALLOCATE SYMBOLIZE OUTPUT BUFFER
03 00000110'EF    05  E1  05AF          BBC     #OPT$V_INST,OPTIONS,71$         ; CONTINUE IF NOT INSTRUCTION
                00DF  31  05B7          BRW     120$                            ; BRANCH IF INSTRUCTION MODE
   00000144'EF    04  D0  05BA  71$:    MOVL    #4,DATALEN                      ; SET LENGTH OF DATA SEGMENT
07 00000110'EF    07  E1  05C1          BBC     #OPT$V_TIME,OPTIONS,72$         ; BRANCH IF NOT /TIME
   00000144'EF    08  D0  05C9          MOVL    #8,DATALEN                      ; SET LENGTH OF TIME QUADWORD
                          05D0  72$:    TRYMEM  @(R3),(R5),DATALEN              ; GET DATA
               42 50  E9  05E2          BLBC    R0,90$                          ; IF NOT AVAILABLE
       000001C0 8F    D3  05E5          BITL    #OPT$M_TIME!OPT$M_PSL!OPT$M_COND,-
       00000110'EF        05EB                  OPTIONS                         ; TIME, COND OR PSL?
                  4A  12  05F0          BNEQ    92$                             ; BRANCH IF EITHER
                  55  DD  05F2          PUSHL   R5                              ; ADDRESS OF COPY OF DATA ITEM
                  04  DD  05F4          PUSHL   #4                              ; LENGTH OF DATA ITEM
                  51  DD  05F6          PUSHL   R1                              ; CONTENTS OF LOCATION
                  52  DD  05F8          PUSHL   R2                              ; ADDRESS OF OUTPUT BUFFER
                  63  DD  05FA          PUSHL   (R3)                            ; ADDRESS TO TRANSLATE
      00000000'EF 02  FB  05FC          CALLS   #2,SYMBOLIZE                    ; TRANSLATE TO SYMBOL+OFFSET
                  61  D5  0603          TSTL    (R1)                            ; CHECK LENGTH OF RESULT STRING
                  10  13  0605          BEQL    80$                             ; BRANCH IF NOT TRANSLATED
                  51  DD  0607          PUSHL   R1                              ; ADDRESS OF STRING DESCRIPTOR
                          0609          PRINT   4,<!AS:  !XL    "!AF">
                      04  0616          RET
                          0617  80$:
                  63  DD  0617          PUSHL   (R3)
                          0619          PRINT   4,<!XL:  !XL    "!AF">
                      04  0626          RET
                          0627
                  63  DD  0627  90$:    PUSHL   (R3)                            ; ADDRESS
                          0629          SIGNAL  1,NOTINPHYS                     ; NOT IN PHYSICAL MEMORY
                      04  063B          RET
                          063C  ;
                          063C  ; EXAMINE /PSL or EXAMINE/TIME or EXAMINE/CONDITION_VALUE
                          063C  ;
                          063C
```

```
                             063C   92$:
24 00000110'EF    06   E0    063C           BBS     #OPT$V_PSL,OPTIONS,96$   ; BRANCH IF EXAM/PSL
29 00000110'EF    08   E0    0644           BBS     #OPT$V_COND,OPTIONS,98$  ; BRANCH IF EXAM/CONDITION_VALUE
                             064C           $ASCTIM_S TIMLEN = (R2),-
                             064C                   TIMBUF = (R2),-
                             064C                   TIMADR = (R5)            ; FORMAT TIME
            2B 50      E8    065B           BLBS    R0,110$                  ; BRANCH IF SUCCESS
                             065E   95$:
               50      DD    065E           PUSHL   R0                       ; ERROR
   00000000'GF    01   FB    0660           CALLS   #1,G^LIB$SIGNAL          ; SIGNAL
                       04    0667           RET
                             0668   96$:
               52      DD    0668           PUSHL   R2                       ; BUFFER ADDRESS
               65      DD    066A           PUSHL   (R5)                     ; PSL ADDRESS
   00000000'EF    02   FB    066C           CALLS   #2,FORMAT_PSL            ; FORMAT IT
               14      11    0673           BRB     110$                     ;
                             0675   98$:
                             0675           $GETMSG_S -
                             0675                   MSGID = (R5) -           ; MESSAGE CODE
                             0675                   MSGLEN = (R2) -          ; LENGTH
                             0675                   BUFADR = (R2)            ; ADDRESS
            D5 50      E9    0686           BLBC    R0,95$
                             0689   110$:
               52      DD    0689           PUSHL   R2
                             068B           PRINT   1,<!AS>
                       04    0698           RET
                             0699
                             0699   ;
                             0699   ; EXAMINE/INSTRUCTION
                             0699   ;
                             0699
                             0699   120$:   TRYMEM  @(R3),(R5),#64           ; GET MAXIMUM LENGTH OF INSTRUCTION
            03 50      E8    06AB           BLBS    R0,130$                  ; OK
             FF76      31    06AE           BRW     90$                      ; IF NOT AVAILABLE
                             06B1   130$:   ALLOC   128,R4                   ; BUFFER FOR INSTRUCTION TEXT
               55      DD    06C3           PUSHL   R5                       ; ADDRESS OF INSTRUCTION STREAM
     000007A1'EF       9F    06C5           PUSHAB  INSTR_VALUE              ; ADDRESS OF SYMBOLIZE ROUTINE
               54      DD    06CB           PUSHL   R4                       ; ADDRESS TO RETURN LENGTH
               54      DD    06CD           PUSHL   R4                       ; ADDRESS OF OUTPUT BUFFER DESCRIPTOR
            0C AE      DF    06CF           PUSHAL  12(SP)                   ; ADDRESS OF INSTRUCTION STREAM POINTER
   00000000'GF    04   FB    06D2           CALLS   #4,G^LIB$INS_DECODE      ; DECODE INSTRUCTION
            39 50      E9    06D9           BLBC    R0,190$                  ; AND SIGNAL ANY ERRORS
00000144'EF    8E 55   C3    06DC           SUBL3   R5,(SP)+,DATALEN         ; SAVE LENGTH OF INSTRUCTION
               54      DD    06E4           PUSHL   R4                       ; PUSH ADDRESS OF INSTRUCTION TEXT
               52      DD    06E6           PUSHL   R2                       ; ADDRESS OF RESULT BUFFER
               63      DD    06E8           PUSHL   (R3)                     ; ADDRESS TO TRANSLATE
   00000000'EF    02   FB    06EA           CALLS   #2,SYMBOLIZE             ; TRANSLATE TO SYMBOL+OFFSET
               61      D5    06F1           TSTL    (R1)                     ; CHECK LENGTH OF RESULT STRING
               10      13    06F3           BEQL    180$                     ; BRANCH IF NOT TRANSLATED
               51      DD    06F5           PUSHL   R1                       ; ADDRESS OF STRING DESCRIPTOR
                             06F7           PRINT   2,<!AS:  !AS>
                       04    0704           RET
               63      DD    0705   180$:   PUSHL   (R3)
                             0707           PRINT   2,<!XL:  !AS>
                       04    0714           RET
                             0715   190$:   SIGNAL  0,NOINSTRAN              ; SIGNAL ANY ERRORS FROM DECODE
                       04    0727           RET
```

```
                                    0728
                                    0728      ;
                                    0728      ; RANGE OF INSTRUCTIONS TO BE DUMPED
                                    0728      ;
                                    0728
        00000110'EF    18     CA    0728      150$:   BICL    #OPT$M_RANGE!OPT$M_LENGTH,OPTIONS ; INDICATE ONLY ADDRESS
00000040'EF   0000010C'EF     9E    072F              MOVAB   EXPR_STACK,ESP              ; POP ALL ARGUMENTS OFF STACK
     55    04 AE       6E     C1    073A              ADDL3   (SP),4(SP),R5               ; GET ENDING ADDRESS
           00000144'EF        D4    073F              CLRL    DATALEN                     ; DO NOT SKIP FIRST INSTRUCTION
     53    00000140'EF        DE    0745              MOVAL   ADDRESS,R3
                                    074C      ;
                                    074C      ; ATTEMPT TO LOCATE THE FIRST INSTRUCTION WHICH MAKES SENSE IN THE RANGE.
                                    074C      ; THIS IS DONE BY ATTEMPTING TO INTERPRET N INSTRUCTIONS IN A ROW WHICH
                                    074C      ; APPEAR IN THE LIST OF "NORMAL" INSTRUCTIONS.  IF WE CAN'T MAKE SENSE
                                    074C      ; OUT OF N IN A ROW, THEN WE INCREMENT "." BY 1 BYTE AND TRY AGAIN.
                                    074C      ; IF THE ENTIRE RANGE IS EXHAUSTED WITHOUT FINDING A VALID SEQUENCE,
                                    074C      ; THEN GIVE UP AND RESET "." BACK TO THE ORIGINALLY REQUESTED ADDRESS
                                    074C      ; AND SHOW THE USER THE CRAP.
                                    074C      ;
          10    04 AE         D1    074C              CMPL    4(SP),#16                   ; IS THE PRE-SCAN WORTH IT?
                   36         19    0750              BLSS    155$                        ; BRANCH IF NOT
   2E 00000110'EF  09         E0    0752              BBS     #OPT$V_NOSKIP,OPTIONS,155$ ; ALLOW USER TO OVERRIDE PRE-SCAN
      000007F3'EF  00         FB    075A      152$:   CALLS   #0,TRY_SEQUENCE ; EXAMINE NEXT N INSTRUCTIONS
                0C 50         E8    0761              BLBS    R0,154$                     ; BRANCH IF THEY ARE REASONABLE
                   63         D6    0764              INCL    (R3)                        ; SKIP AHEAD ONE BYTE
             55    63         D1    0766              CMPL    (R3),R5                     ; END OF RANGE?
                   EF         1F    0769              BLSSU   152$                        ; LOOP IF NOT
             63    6E         D0    076B              MOVL    (SP),(R3)                   ; RESET "." TO ORIGINAL ADDRESS
                   18         11    076E              BRB     155$
        7E   63    6E         C3    0770      154$:   SUBL3   (SP),(R3),-(SP)             ; NUMBER OF BYTES SKIPPED
                   12         13    0774              BEQL    155$                        ; SKIP MESSAGE IF NOTHING SKIPPED
                                    0776              SIGNAL  1,INSKIPPED                 ; INFORM NUMBER OF BYTES SKIPPED
                                    0788      ;
                                    0788      ; SHOW THE RANGE OF INSTRUCTIONS, UPDATING "." ON EACH INSTRUCTION
                                    0788      ;
          FD4B CF  00         FB    0788      155$:   CALLS   #0,EXAM_MEMORY              ; RECURSIVELY SHOW NEXT DATA ITEM
             55    63         D1    078D              CMPL    (R3),R5                     ; END OF RANGE YET?
                   F6         1F    0790              BLSSU   155$                        ; LOOP UNTIL DONE
             50    01         D0    0792              MOVL    #1,R0                       ; SUCCESS
                              04    0795              RET
                                    0796
                                    0796              .DSABL  LSB
```

```
                                  0796          .SBTTL  INSTR_VALUE, SYMBOLIZE ADDRESS INTO BUFFER
                                  0796  ;---
                                  0796  ;
                                  0796  ;       OUTPUT A STRING DESCRIBING AN ADDRESS.  THE ADDRESS IS CONVERTED
                                  0796  ;       TO A ''REAL'' ADDRESS BY COMPUTING THE OFFSET FROM THE INSTRUCTION
                                  0796  ;       BUFFER.  THEN THE SYMBOL TABLE IS SEARCHED FOR THE CLOSEST SYMBOL.
                                  0796  ;       IF NOT FOUND IN THE SYMBOL TABLE, A LONGWORD HEX STRING IS WRITTEN.
                                  0796  ;
                                  0796  ; INPUTS:
                                  0796  ;
                                  0796  ;       4(AP) = ADDRESS OF LONGWORD CONTAINING ADDRESS
                                  0796  ;       8(AP) = ADDRESS OF RESULT BUFFER DESCRIPTOR
                                  0796  ;       12(AP) = ADDRESS OF WORD TO RECEIVE RESULT LENGTH
                                  0796  ;       16(AP) = ADDRESS OF FLAG: 1 IF ABSOLUTE ADDRESS, 0 IF RELATIVE ADDRESS
                                  0796  ;
                                  0796  ; OUTPUTS:
                                  0796  ;
                                  0796  ;       THE STRING IS WRITTEN INTO THE RESULT BUFFER.
                                  0796  ;---
                                  0796
                                  0796  XL_STRING:
      4C 58 21 0000079E'010E0000' 0796          .ASCID  '!XL'
                                  07A1
                                  07A1  INSTR_VALUE:                                    ; SYMBOLIZE INSTRUCTION VALUE
                            000C  07A1          .WORD   ^M<R2,R3>
                                  07A3
           52    04 BC     D0     07A3          MOVL    @4(AP),R2               ; GET VALUE (ARGUMENT BY REFERENCE)
           53    08 AC     D0     07A7          MOVL    8(AP),R3                ; GET ADDRESS OF DESCRIPTOR
              11 10 BC     E8     07AB          BLBS    @16(AP),5$              ; BRANCH IF ABSOLUTE ADDRESS
     50    00000000'EF     9E     07AF          MOVAB   BUFFER,R0               ; ADDRESS OF INSTRUCTION BUFFER
                 52  50    C2     07B6          SUBL    R0,R2                   ; GET OFFSET FROM INSTRUCTION
     52    00000140'EF     C0     07B9          ADDL    ADDRESS,R2              ; AND COMPUTE ''REAL'' ADDRESS
                     53    DD     07C0  5$:     PUSHL   R3                      ; ADDRESS OF RESULT BUFFER
                     52    DD     07C2          PUSHL   R2                      ; VALUE TO BE CONVERTED
        00000000'EF  02    FB     07C4          CALLS   #2,SYMBOLIZE            ; CONVERT TO SYMBOL+OFFSET
                     61    D5     07CB          TSTL    (R1)                    ; ANYTHING RETURNED IN BUFFER?
                     08    13     07CD          BEQL    10$                     ; BRANCH IF NOT
              0C BC  61    B0     07CF          MOVW    (R1),@12(AP)            ; RETURN LENGTH TO CALLER
                 50  01    D0     07D3          MOVL    #1,R0
                     04           07D6          RET
                 5E  02    C2     07D7  10$:    SUBL    #2,SP                   ; ADDRESS TO RETURN FAO LENGTH
                     52    DD     07DA          PUSHL   R2                      ; ABSOLUTE VALUE
                     53    DD     07DC          PUSHL   R3                      ; ADDRESS OF RESULT BUFFER
                 08  AE    DF     07DE          PUSHAL  8(SP)                   ; ADDRESS OF WORD TO RECEIVE LENGTH
                 B2  AF    9F     07E1          PUSHAB  XL_STRING               ; ADDRESS OF FAO CONTROL STRING
        00000000'GF  04    FB     07E4          CALLS   #4,G^SYS$FAO            ; CONVERT INTO BUFFER
              0C BC  8E    B0     07EB          MOVW    (SP)+,@12(AP)           ; RETURN LENGTH TO CALLER
                 50  01    D0     07EF          MOVL    #1,R0                   ; SUCCESS
                     04           07F2          RET
```

```
                          07F3        .SBTTL  TRY_SEQUENCE, DETERMINE IF INSTRUCTIONS VALID
                          07F3   ;---
                          07F3   ;
                          07F3   ; SCAN THE CURRENT INSTRUCTION STREAM LOOKING FOR N INSTRUCTIONS IN
                          07F3   ; A ROW WHICH "MAKE SENSE", ACCORDING TO A LIST OF "NORMAL" INSTRUCTIONS.
                          07F3   ;
                          07F3   ; INPUTS:
                          07F3   ;
                          07F3   ;       ADDRESS = "REAL" ADDRESS OF INSTRUCTION STREAM
                          07F3   ;
                          07F3   ; OUTPUTS:
                          07F3   ;
                          07F3   ;       R0 = TRUE IF STREAM REASONABLE, ELSE FALSE
                          07F3   ;---
                          07F3
                          07F3   TRY_SEQUENCE:
                   0038   07F3        .WORD   ^M<R3,R4,R5>
                          07F5
55  00000000'EF     9E    07F5        MOVAB   BUFFER,R5                ; GET ADDRESS OF INSTRUCTION BUFFER
53  00000140'EF     D0    07FC        MOVL    ADDRESS,R3               ; GET CURRENT "."
          54  04    D0    0803        MOVL    #4,R4                    ; INITIALIZE LOOP COUNT
                          0806   10$: TRYMEM  (R3),(R5),#64            ; GET MAXIMUM LENGTH OF INSTRUCTION
          28  50    E9    0817        BLBC    R0,90$                   ; IF NOT AVAILABLE
       50     65    9A    081A        MOVZBL  (R5),R0                  ; PICK UP OPCODE
     21 0000'C0     E9    081D        BLBC    NORMAL_INSTRS(R0),80$    ; BRANCH IF UNREASONABLE INSTRUCTION
              55    DD    0822        PUSHL   R5                       ; ADDRESS OF INSTRUCTION STREAM
              7E    7C    0824        CLRQ    -(SP)                    ; NO SYMBOLIZE ROUTINE OR RETLEN
              5E    DD    0826        PUSHL   SP                       ; ADDRESS OF NULL BUFFER DESCRIPTOR
          0C  AE    DF    0828        PUSHAL  12(SP)                   ; ADDRESS OF INSTRUCTION STREAM POINTER
00000000'GF   04    FB    082B        CALLS   #4,G^LIB$INS_DECODE      ; DECODE INSTRUCTION
          0D  50    E9    0832        BLBC    R0,90$                   ; IF ERROR, RETURN UNREASONABLE
    50    8E  55    C3    0835        SUBL3   R5,(SP)+,R0              ; CALCULATE LENGTH OF INSTRUCTION
          53  50    C0    0839        ADDL    R0,R3                    ; AND POINT TO NEXT INSTRUCTION
          C7  54    F5    083C        SOBGTR  R4,10$                   ; TRY NEXT INSTRUCTION
       50     01    D0    083F        MOVL    #1,R0                    ; RETURN SEQUENCE IS REASONABLE
                    04    0842   90$: RET
              50    D4    0843   80$: CLRL    R0                       ; RETURN SEQUENCE UNREASONABLE
                    04    0845        RET
```

```
                                  0846        .SBTTL  SET_PROCESS -- SELECT SPECIFIED PROCESS
                                  0846  ;---
                                  0846  ;
                                  0846  ;       SET_PROCESS
                                  0846  ;
                                  0846  ;       SETUP FOR DISPLAY OF A SPECIFIED PROCESS
                                  0846  ;
                                  0846  ; INPUTS:
                                  0846  ;
                                  0846  ;       OPTIONS = OPTIONS MASK (ONLY SYSPROC IS CHECKED)
                                  0846  ;       PROC_NAME = PROCESS NAME
                                  0846  ;       PROC_INDEX = PROCESS INDEX (MUTALLY EXCLUSIVE)
                                  0846  ;
                                  0846  ; OUTPUTS:
                                  0846  ;
                                  0846  ;       PROC_PID = PROCESS PID (0 IF CURRENT PROCESS)
                                  0846  ;       BADPROC IF NOT FOUND
                                  0846  ;
                                  0846  ;       ALL ERRORS ARE SIGNALLED IMMEDIATELY
                                  0846  ;
                                  0846  ;---
                                  0846
                         01FC     0846        .ENTRY  SET_PROCESS,-
                                  0848                ^M<R2,R3,R4,R5,R6,R7,R8>
                                  0848
                                  0848  ;
                                  0848  ;       IF /SYSTEM SPECIFIED, SET TO "SYSTEM PROCESS"
                                  0848  ;
       10 00000110'EF   06   E1   0848        BBC     #OPT$V_SYSPROC,OPTIONS,10$ ; BRANCH IF NOT SPECIFIED
                  58    01   CE   0850        MNEGL   #1,R8                     ; SET TO -1 (SYSTEM)
00000000'EF  00000000'FF     D0   0853  5$:   MOVL    @MMG$AL_SYSPCB,PCBADR     ; SET SYSTEM PCB ADDRESS
                        44   11   085E        BRB     20$
                                  0860  ;
                                  0860  ;       IF PROCESS INDEX GIVEN, OBTAIN PCB ADDRESS FROM VECTOR
                                  0860  ;
                                  0860  10$:
             00000118'EF     D5   0860        TSTL    PROC_NAME                 ; WAS NAME SPECIFIED
                        5B   12   0866        BNEQ    50$                       ; BRANCH IF SO
                                  0868        REQMEM  @SCH$GL_MAXPIX            ; MAXIMUM INDEX NUMBER
          58 00000130'EF     32   0875        CVTWL   PROC_INDEX,R8             ; CHECK IF "SYSTEM PROCESS"
                        D5   19   087C        BLSS    5$                        ; BRANCH IF SO
                  51    58   D1   087E        CMPL    R8,R1                     ; CHECK IF IN RANGE
                        3D   1A   0881        BGTRU   40$                       ; ERROR IF BAD INDEX
                                  0883        REQMEM  @SCH$GL_PCBVEC            ; VECTOR OF PCB ADDRESSES
                51    6148   DE   0890        MOVAL   (R1)[R8],R1               ; ADDRESS OF POINTER TO PCB
                                  0894        REQMEM  (R1),PCBADR               ; STORE ADDRESS OF PCB
                                  08A4  20$:
                                  08A4        REQMEM  @PCBADR,PCB,#PCB$C_LENGTH ; GET ENTIRE PCB
                      0074   31   08BD        BRW     70$
                                  08C0  40$:
                      00F8   31   08C0        BRW     80$                       ; FAILURE
                                  08C3  ;
                                  08C3  ;       LOOKUP SPECIFIED NAME IN ALL PCBS
                                  08C3  ;
                                  08C3  50$:
                                  08C3        REQMEM  @SCH$GL_PCBVEC            ; VECTOR OF PCB ADDRESSES
                  56    51   D0   08D0        MOVL    R1,R6                     ; SAVE ADDRESS OF VECTOR
```

```
                                   08D3          REQMEM  @SCH$GL_MAXPIX                 ; MAXIMUM PROCESS INDEX
                  57     51   D0   08E0          MOVL    R1,R7                          ; SAVE NUMBER OF SLOTS
                         58   D4   08E3          CLRL    R8                             ; INITIALIZE CURRENT INDEX
                                   08E5      60$:
              51       6648   DE   08E5          MOVAL   (R6)[R8],R1                    ; ADDRESS OF POINTER TO PCB
                                   08E9          REQMEM  (R1),PCBADR                    ; GET ADDRESS OF PCB
                                   08F9          REQMEM  @PCBADR,PCB,#PCB$C_LENGTH      ; READ ENTIRE PCB
           50   00000070'EF   9A   0912          MOVZBL  PCB+PCB$T_LNAME,R0             ; GET LENGTH OF NAME
     20   00000071'EF    50   2D   0919          CMPC5   R0,PCB+PCB$T_LNAME+1,#^A' ', -
                00000118'EF        0921
                0000011C'FF        0926                  PROC_NAME,@PROC_NAME+4         ; CHECK IF CORRECT PROCESS
                         07   13   092B          BEQL    70$                            ; BRANCH IF YES
               B4 58     57   F3   092D          AOBLEQ  R7,R8,60$                      ; LOOP UNTIL FOUND
                       0087   31   0931          BRW     80$                            ; FAILURE
                                   0934      70$:
                00000118'EF   7C   0934          CLRQ    PROC_NAME                      ; CLEAR SO WE DON'T RE-SEARCH
           51   0000006C'EF   D0   093A          MOVL    PCB+PCB$L_PHD,R1               ; ADDRESS OF PROCESS HEADER
                         3B   13   0941          BEQL    75$                            ; BRANCH IF NOT RESIDENT
                                   0943          REQMEM  (R1),PHD,#PHD$C_LENGTH         ; READ ENTIRE PHD
  00000000'EF   0000006C'EF   D0   0958          MOVL    PCB+PCB$L_PHD,PHDADR           ; SAVE PHD ADDRESS
00000000'EF   000000C8'EF    10   28   0963          MOVC    #16,PHD+PHD$L_P0BR,P0BR        ; GET RELOCATION REGISTERS
     00000000'EF    18   00   EF   096F          EXTZV   #PHD$V_P0LR,#PHD$S_P0LR,P0LR,P0LR
                00000000'EF        0977
                         10   11   097C          BRB     79$
                00000000'EF   D4   097E      75$: CLRL   PHDADR                         ; SET PHD NOT RESIDENT
       10    00     6E   00   2C   0984          MOVC5   #0,(SP),#0,#16,P0BR            ; ZERO RELOCATION REGISTERS
                00000000'EF        0989
       00000130'EF       58   D0   098E      79$: MOVL   R8,PROC_INDEX                  ; SAVE PROCESS INDEX
  00000138'EF   00000060'EF   D0   0995          MOVL    PCB+PCB$L_PID,PROC_PID         ; SAVE PROCESS PID
                                   09A0      ;
                                   09A0      ;       ZERO THE PROC_PID IF THE PROCESS IS THE CURRENT PROCESS.
                                   09A0      ;       THIS AVOIDS UNNECESSARY OVERHEAD ON THE RUNNING SYSTEM DISPLAYS.
                                   09A0      ;
  00000138'EF   0000013C'EF   D1   09A0          CMPL    CURRENT_PID,PROC_PID           ; IS PROC_PID SAME AS CUR. PROCESS?
                         06   12   09AB          BNEQ    90$                            ; BRANCH IF NOT
                00000138'EF   D4   09AD          CLRL    PROC_PID                       ; IF SO, ZERO PROC_PID FOR QAST
                                   09B3      90$: STATUS SUCCESS
                              04   09BA          RET
                                   09BB      80$:
                00000118'EF   7C   09BB          CLRQ    PROC_NAME                      ; CLEAR BAD PROCESS NAME
                                   09C1          SIGNAL  0,BADPROC                      ; NO SUCH PROCESS
                              04   09D3          RET
```

```
                                     09D4        .SBTTL  CURPROC -- SET RELOCATION TO CURRENT PROCESS
                                     09D4  ;---
                                     09D4  ;
                                     09D4  ;         CURPROC
                                     09D4  ;
                                     09D4  ;         THIS ROUTINE SETS UP THE RELOCATION REGISTERS TO
                                     09D4  ;         REFER TO THE PROCESS EXECUTING WHEN THE SYSTEM
                                     09D4  ;         CRASHED.
                                     09D4  ;
                                     09D4  ;     INPUTS:
                                     09D4  ;
                                     09D4  ;         NONE
                                     09D4  ;
                                     09D4  ;     OUTPUTS:
                                     09D4  ;
                                     09D4  ;         P0BR-P1LR ARE SETUP FOR ''CRASH'' PROCESS
                                     09D4  ;
                                     09D4  ;---
                                     09D4
                                     09D4  CURPROC::
                              003C   09D4        .WORD   ^M<R2,R3,R4,R5>
                                     09D6
                                     09D6        GETMEM  @SCH$GL_CURPCB           ; ADDRESS OF CURRENT PCB
                                     09E3        RETIFERR
           00000000'EF    51    D0   09E7        MOVL    R1,PCBADR                ; SAVE ADDRESS OF PCB
       53    00000000'EF        9E   09EE        MOVAB   L^PCB,R3                 ; ADDRESS OF OWN PCB
                                     09F5        GETMEM  (R1),(R3),#PCB$C_LENGTH  ; READ ENTIRE PCB
                                     0A06        RETIFERR
       00000130'EF    60 A3     32   0A0A        CVTWL   PCB$L_PID(R3),PROC_INDEX ; SAVE PROCESS INDEX
                                     0A12  ;     MOVL    PCB$L_PID(R3),PROC_PID   ; SAVE PROCESS PID
       0000013C'EF    60 A3     D0   0A12        MOVL    PCB$L_PID(R3),CURRENT_PID ; SAVE PID OF CURRENT PROCESS
                 00000138'EF    D4   0A1A        CLRL    PROC_PID                 ; DO NOT SWITCH CONTEXT IN GETMEM
       00000000'EF    6C A3     D0   0A20        MOVL    PCB$L_PHD(R3),PHDADR     ; SAVE PHD ADDRESS
       54    00000000'EF        9E   0A28        MOVAB   L^PHD,R4                 ; ADDRESS OF OWN PHD
                                     0A2F        GETMEM  @PCB$L_PHD(R3),(R4),#PHD$C_LENGTH  ; READ ENTIRE PHD
                                     0A41        RETIFERR
00000000'EF    00C8 C4    10    28   0A45        MOVC    #16,PHD$L_P0BR(R4),P0BR  ; GET P0BR - P1LR
       00000000'EF    18    00  EF   0A4F        EXTZV   #PHD$V_P0LR,#PHD$S_P0LR,P0LR,P0LR   ; ELIM. ASTLVL
                 00000000'EF         0A57
                                     0A5C        GETMEM  @SCH$GL_MAXPIX           ; GET MAXIMUM PROCESS INDEX
                                     0A69        RETIFERR
                 50    1F       D0   0A6D        MOVL    #31,R0                   ; START AT HIGH BIT
              03 51    50       E0   0A70  10$:  BBS     R0,R1,11$                ; FIND THE HIGHEST SET BIT
                 F9    50       F5   0A74        SOBGTR  R0,10$                   ; WE KNOW WE WILL FIND ONE
       00000134'EF    50    01  C1   0A77  11$:  ADDL3   #1,R0,PIX_WIDTH          ; BIT ''N'' SET MEANS THAT PIX WIDTH FOR
                                     0A7F                                         ;  EXTENDED PID IS ''N+1'' BITS
                                04   0A7F        RET
```

```
                                   0A80         .SBTTL  SHOW_PROCESS -- DISPLAY SPECIFIED PROCESS
                                   0A80  ;---
                                   0A80  ;
                                   0A80  ;      SHOW_PROCESS
                                   0A80  ;
                                   0A80  ;      DISPLAY SELECTED FIELDS FROM A SPECIFIED PROCESS
                                   0A80  ;
                                   0A80  ;  INPUTS:
                                   0A80  ;
                                   0A80  ;      PROC_NAME OR PROC_INDEX CONTAINS WHICH PROCESS
                                   0A80  ;
                                   0A80  ;  OUTPUTS:
                                   0A80  ;
                                   0A80  ;      NONE
                                   0A80  ;
                                   0A80  ;---
                                   0A80
                         4C 4C 41  0A80  ALL:   .ASCII  'ALL'
                                   0A83
                                   0A83  SHOW_PROCESS::
                             003C  0A83         .WORD   ^M<R2,R3,R4,R5>
                                   0A85
00000118'EF   20   F7 AF   03   2D  0A85         CMPC5   #3,ALL,#^A' ',PROC_NAME,@PROC_NAME+4
              0000011C'FF           0A8F
                           08   12  0A94         BNEQ    10$                         ; BRANCH IF NOT ALL
        00000000'EF        00   FB  0A96         CALLS   #0,DISPLAY_PROCS            ; DISPLAY ALL PROCESSES
                                04  0A9D         RET
                                    0A9E  10$:
              FDA3 CF      00   FB  0A9E         CALLS   #0,SET_PROCESS              ; SETUP TO READ PROCESS
                        07 50   E9  0AA3         BLBC    R0,90$                      ; BRANCH IF ERROR
        00000000'EF        00   FB  0AA6         CALLS   #0,DISPLAY_PROCESS          ; DISPLAY INFORMATION
                                    0AAD  90$:
                                    0AAD         STATUS  SUCCESS
                                04  0AB4         RET
```

```
                              0AB5          .SBTTL  DISPLAY_HELP -- DISPLAY HELP INFORMATION
                              0AB5  ;---
                              0AB5  ;
                              0AB5  ;       DISPLAY_HELP
                              0AB5  ;
                              0AB5  ;       DISPLAY PORTIONS OF THE HELP FILE BASED ON THE TYPE
                              0AB5  ;       OF INFORMATION REQUESTED IN THE HELP COMMAND.
                              0AB5  ;
                              0AB5  ;   INPUTS:
                              0AB5  ;
                              0AB5  ;       AP POINTS TO THE TPARSE BLOCK
                              0AB5  ;
                              0AB5  ;   OUTPUTS:
                              0AB5  ;
                              0AB5  ;       NONE
                              0AB5  ;
                              0AB5  ;---
                              0AB5
                              0AB5          .ENABL  LSB
                              0AB5
                              0AB5  DISPLAY_HELP::
                        0000  0AB5          .WORD   ^M<>
   6D   00000AF6'EF     DE    0AB7          MOVAL   HELPHAND,(FP)           ;Establish Condition Handler
        00000000'GF     DF    0ABE          PUSHAL  G^LIB$GET_INPUT         ;Address of input routine
        00000000'EF     DF    0AC4          PUSHAL  HELP_FLAGS              ;Library search & Prompt options
        00000004'EF     7F    0ACA          PUSHAQ  HELP_LIBRARY            ;Address of library name descriptor
              10 AC     DF    0AD0          PUSHAL  TPA$L_TOKENCNT(AP)      ;Address of input line descriptor
                 00     DD    0AD3          PUSHL   #0                      ;Text line width -- Defaulted
        00000000'GF     DF    0AD5          PUSHAL  G^LIB$PUT_OUTPUT        ;Address of output routine
   00000000'GF  06      FB    0ADB          CALLS   #6,G^LBR$OUTPUT_HELP    ;Does the obvious...
              09 50     E8    0AE2          BLBS    R0,10$                  ;Any problems?
                 50     DD    0AE5          PUSHL   R0                      ;YES,...
   00000000'GF  01      FB    0AE7          CALLS   #1,G^LIB$SIGNAL         ;     Send it to the condition handler
                              0AEE  10$:    STATUS  SUCCESS                 ;All's well that ends well...
                 04           0AF5          RET
                              0AF6
                              0AF6          .DSABL  LSB
                              0AF6  ;
                              0AF6  ;       Condition Handler for Display_Help.
                              0AF6  ;
                              0AF6
                              0AF6          .ENABL  LSB
                              0AF6
                              0AF6  HELPHAND:
                        0000  0AF6          .WORD   ^M<>
        50    04 AC     D0    0AF8          MOVL    4(AP),R0                ;Get address of Signal Array
                              0AFC  ;
                              0AFC  ;       Check to see if this is a SEVERE or FATAL exception...
                              0AFC  ;       if it is, then CHANGE the severity to ERROR and resignal the
                              0AFC  ;       condition, else, just resignal the condition.
                              0AFC  ;
                 00     ED    0AFC          CMPZV   #STS$V_SEVERITY,-       ;Position of Severity Field
                 03           0AFE                  #STS$S_SEVERITY,-       ;Size of the Severity Field
              04 A0           0AFF                  CHF$L_SIG_NAME(R0),-    ;Base address of comparison field
                 04           0B01                  #STS$K_SEVERE           ;Test value
                 06     12    0B02          BNEQ    10$                     ;Its not a FATAL severity.
                              0B04  ;
```

```
                        0B04  1133 ;         If we are here, then we do have a FATAL severity code in our
                        0B04  1134 ;         condition value, sooo... we must CHANGE the severity to ERROR.
                        0B04  1135 ;
            02     F0   0B04  1136           INSV    #STS$K_ERROR,-             ;The desired severity code.
            00          0B06  1137                   #STS$V_SEVERITY,-          ;Position of Severity Field
            03          0B07  1138                   #STS$S_SEVERITY,-          ;Size of the Severity Field
         04 A0          0B08  1139                   CHF$L_SIG_NAME(R0)         ;Base address of Signal Name
                        0B0A  1140
  50    0918 8F    3C   0B0A  1141 10$:      MOVZWL  #SS$_RESIGNAL,R0           ;Send condition to HANDLER.
                   04   0B0F  1142           RET
                        0B10  1143
                        0B10  1144           .DSABL  LSB
```

```
                          0B10          .SBTTL  READ_SYMFILE, Read symbols from given file
                          0B10   ;----
                          0B10   ;
                          0B10   ;       READ_SYMFILE
                          0B10   ;
                          0B10   ;       This command can be executed to add symbol definitions
                          0B10   ;       to the working symbol table by reading object modules
                          0B10   ;       containing GST entries and defining all symbols found.
                          0B10   ;
                          0B10   ;   Inputs:
                          0B10   ;
                          0B10   ;       FILE_DESC - Descriptor of file specification
                          0B10   ;
                          0B10   ;   Outputs:
                          0B10   ;
                          0B10   ;       The symbols are added to the symbol table.
                          0B10   ;
                          0B10   ;----
                          0B10
                          0B10   sym_default:
             42 54 53 2E  0B10          .ascii  '.STB'
                00000004  0B14   sym_def_len = .-sym_default
                          0B14
                    003C  0B14          .entry  read_symfile,-
                          0B16                  ^m<r2,r3,r4,r5>
                          0B16
  50   00000128'EF    9E  0B16          movab   file_desc,r0            ; Address of file descriptor
  52   00000000'EF    9E  0B1D          movab   stbf,r2                 ; R2 = Address of FAB
  53   00000000'EF    9E  0B24          movab   stb,r3                  ; R3 = Address of RAB
        34 A2    60   33  0B2B          cvtwb   (r0),fab$b_fns(r2)      ; Set file name length
     2C A2    04 A0   D0  0B2F          movl    4(r0),fab$l_fna(r2)     ; Set file name address
        35 A2    04   90  0B34          movb    #sym_def_len,fab$b_dns(r2)      ; Set default length
     30 A2    D5 AF   9E  0B38          movab   sym_default,fab$l_dna(r2)       ; Set default address
                          0B3D          $open   (r2)                    ; Attempt to open the file
           29 50      E8  0B46          blbs    r0,5$                   ; branch if ok
           2C A2      DD  0B49          pushl   fab$l_fna(r2)           ; Create descriptor of file name
        7E 34 A2      9A  0B4C          movzbl  fab$b_fns(r2),-(sp)
           7E 50      7D  0B50          movq    r0,-(sp)                ; RMS error codes
              00      F0  0B53          insv    #sts$k_warning,-        ; Change severity to warning
      6E   03 00          0B55                  #sts$v_severity,#sts$s_severity,(sp)
           08 AE      9F  0B58          pushab  8(sp)                   ; Address of file name descriptor
  50   00031098 8F    D0  0B5B          movl    #shr$_openin!<3@16>,r0  ; "Error opening input file !AS"
                          0B62          signal  1                       ; Signal with 1 FAO argument
              5A      11  0B70          brb     95$                     ; exit with success
                          0B72   5$:    $connect (r3)                   ; Connect for record access
                          0B7B          signal  RMS,(r3)                ; Report any errors
  00000000'EF    00   FB  0B8D          calls   #0,rewind_stb           ; Rewind the file
                          0B94   10$:
  00000000'EF    00   FB  0B94          calls   #0,get_symbol           ; Get next symbol in file
           13 50      E9  0B9B          blbc    r0,90$                  ; Branch if end of file
           05 A1      DD  0B9E          pushl   obj$l_sym_value(r1)     ; Value of symbol
           0A A1      DF  0BA1          pushal  obj$t_sym_name+1(r1)    ; Address of symbol
        7E 09 A1      9A  0BA4          movzbl  obj$t_sym_name(r1),-(sp); Length of symbol
  00000000'EF    03   FB  0BA8          calls   #3,add_symbol           ; Add to symbol table
              E3      11  0BAF          brb     10$
                          0BB1   90$:
                          0BB1          $close  (r2)                    ; Close the file
```

```
        0BBA  1203          signal  RMS,(r2)                ; Signal any errors
        0BCC  1204 95$:     status  success
    04  0BD3  1205          ret
```

```
                              0BD4         .sbttl  SEARCH_MEMORY
                              0BD4   ;+
                              0BD4   ;
                              0BD4   ;     search memory for the longword pattern
                              0BD4   ;
                              0BD4   ; Inputs:
                              0BD4   ;
                              0BD4   ;     expr_stack      -->      pattern
                              0BD4   ;                              end_addr or length
                              0BD4   ;                              start_addr
                              0BD4   ;
                              0BD4   ; Outputs:
                              0BD4   ;
                              0BD4   ;     None
                              0BD4   ;-
                              0BD4         .enabl  lsb
                              0BD4
                        01CC  0BD4         .entry  search_memory,^M<r2,r3,r6,r7,r8>
                              0BD6
 52  00000110'EF          D0  0BD6         movl    options, r2               ; Get options
 53  00000040'EF          D0  0BDD         movl    esp, r3          ; point to expression stack
    56     08 A3          D0  0BE4         movl    8(r3), r6                 ; r6 --> start addr
    57     04 A3          D0  0BE8         movl    4(r3), r7                 ; r7 --> end addr or length
    03 52     03          E0  0BEC         bbs     #opt$v_range, r2, 10$     ; br if range option
       57     56          C0  0BF0         addl    r6, r7                    ; add start addr to length
       58     63          D0  0BF3  10$:   movl    (r3), r8                  ; r8 --> value to search for
                              0BF6   ;
                              0BF6   ;     Search from (r6) to (r7) for r8
                              0BF6   ;
              58          DD  0BF6         pushl   r8
              57          DD  0BF8         pushl   r7
              56          DD  0BFA         pushl   r6
                              0BFC         print   3,<Searching from !XL to !XL for !XL...>
                              0C09  20$:
       57     56          D1  0C09         cmpl    r6, r7                    ; check for end of search
              25          1A  0C0C         bgtru   40$                       ; GTR if range searched
                              0C0E         trymem  (r6)                      ; get contents of location
       14     50          E9  0C17         blbc    r0, 30$                   ; if not an address
       51     58          D1  0C1A         cmpl    r8, r1                    ; does this location match?
              0F          12  0C1D         bneq    30$                       ; Br if not
                              0C1F   ;
                              0C1F   ;     Here with a match
                              0C1F   ;
              56          DD  0C1F         pushl   r6                        ; push address of match
                              0C21         print   1,<Match at !XL>          ; print it out
       56     04          C0  0C2E  30$:   addl    #4, r6                    ; step on a longword
              D6          11  0C31         brb     20$                       ; go look at the next
                              0C33  40$:
       50     01          D0  0C33         movl    #1, r0                    ; O.K. status
                          04  0C36         ret                               ;
                              0C37
                              0C37         .dsabl  lsb
```

```
                           0C37         .sbttl  ECHO
                           0C37   ;+
                           0C37   ;
                           0C37   ;       ECHO last command
                           0C37   ;
                           0C37   ; Inputs:
                           0C37   ;
                           0C37   ;       R2 - RAB address
                           0C37   ;       R3 - TPARSE BLOCK
                           0C37   ;
                           0C37   ;-
                           0C37
                           0C37   ECHO:
             54     DD     0C37         PUSHL   R4                              ; SAVE
      54  3C A2     D0     0C39         MOVL    RAB$L_FAB(R2),R4                ; R2 = ADDRESS OF FAB
        00000000'8F E1     0C3D         BBC     #DEV$V_TRM,-
           3D 40 A4        0C43                 FAB$L_DEV(R4),20$               ; BRANCH IF NOT TERMINAL
        00000000'EF B5     0C46         TSTW    TT_CHAN                         ; CHANNEL ASSIGNED?
                35  13     0C4C         BEQL    20$
                           0C4E
        00000000'8F E1     0C4E         BBC     #TT$V_SCOPE,-
     29 00000000'EF        0C54                 DVI_DEVDEPEND,20$               ; IGNORE IF NOT SCOPE
                           0C5A
                           0C5A         $QIOW_S CHAN=TT_CHAN,-
                           0C5A                 FUNC=#IO$_WRITEVBLK,-
                           0C5A                 P1=@TPA$L_STRINGPTR(R3),-
                           0C5A                 P2=TPA$L_STRINGCNT(R3),-
                           0C5A                 P4=#^X00880000                  ; BACKSPACE FIRST!
                           0C83   20$:
             54 8ED0       0C83         POPL    R4                              ; RESTORE
                 05        0C86         RSB
                           0C87
                           0C87         .END
```

```
$$                            = 00000001                  LIB$INS_DECODE                  ********   X   03
$$.TMP1                       = 00000001                  LIB$PUT_OUTPUT                  ********   X   03
$$.TMP2                       = 00000062                  LIB$SIGNAL                      ********   X   03
$$T1                          = 00000001                  LIB$TPARSE                      ********   X   03
ADDRESS                         00000140 RG      02       LINE_COUNT                      ********   X   03
ADD_SYMBOL                      ********    X    03       LOGRAB                          ********   X   03
ALL                             00000A80 R       03       LOG_BUFFER                      00000150 R     02
ARGS                          = 00000001                  LOG_BUFFER_LENGTH             = 00000055
BUFFER                          ********    X    03       LOG_FILE                        ********   X   03
CHF$L_SIG_NAME                = 00000004                  MMG$AL_SYSPCB                   ********   X   03
CURPROC                         000009D4 RG      03       MSG$_BADPROC                    ********   X   03
CURRENT_PID                     0000013C R       02       MSG$_BADSYM                     ********   X   03
DATALEN                         00000144 R       02       MSG$_EOF                        ********   X   03
DEFINE_SYMBOL                   000003DF RG      03       MSG$_EXITCMD                    ********   X   03
DEV$V_TRM                       ********    X    03       MSG$_INSKIPPED                  ********   X   03
DISPLAY_HELP                    00000AB5 RG      03       MSG$_NOINSTRAN                  ********   X   03
DISPLAY_PROCESS                 ********    X    03       MSG$_NOTINPHYS                  ********   X   03
DISPLAY_PROCS                   ********    X    03       MSG$_SUCCESS                    ********   X   03
DUMP                            ********    X    03       NEW_PAGE                        ********   X   03
DVI_DEVCHAR                     ********    X    03       NORMAL_INSTRS                   00000000 R     03
DVI_DEVDEPEND                   ********    X    03       OBJ$L_SYM_VALUE               = 00000005
DVI_DEVDEPND2                   ********    X    03       OBJ$T_SYM_NAME                = 00000009
ECHO                            00000C37 R       03       OPT$M_ALL                     = 00000FFF
ESP                             00000040 RG      02       OPT$M_COND                    = 00000100
EXAM_MEMORY                     000004D8 RG      03       OPT$M_INST                    = 00000020
EXIT_COMMAND                    000003CB RG      03       OPT$M_LENGTH                  = 00000010
EXPR_STACK                      0000010C R       02       OPT$M_NOSKIP                  = 00000200
FAB$B_DNS                     = 00000035                  OPT$M_PSL                     = 00000040
FAB$B_FNS                     = 00000034                  OPT$M_RANGE                   = 00000008
FAB$L_DEV                     = 00000040                  OPT$M_TIME                    = 00000080
FAB$L_DNA                     = 00000030                  OPT$V_COND                    = 00000008
FAB$L_FNA                     = 0000002C                  OPT$V_INST                    = 00000005
FAB$L_STV                     = 0000000C                  OPT$V_LENGTH                  = 00000004
FILE_DESC                       00000128 RG      02       OPT$V_NOSKIP                  = 00000009
FORMAT_PSL                      ********    X    03       OPT$V_P0                      = 00000000
GETMEM                          ********    X    03       OPT$V_P1                      = 00000001
GET_COMMANDS                    00000107 RG      03       OPT$V_PSL                     = 00000006
GET_INPUT                       ********    X    03       OPT$V_RANGE                   = 00000003
GET_SYMBOL                      ********    X    03       OPT$V_SYSPROC                 = 00000006
HANDLER                         ********    X    03       OPT$V_SYSTEM                  = 00000002
HEADING_ROUTINE                 ********    X    03       OPT$V_TIME                    = 00000007
HELPHAND                        00000AF6 R       03       OPTIONS                         00000110 RG    02
HELP_FLAGS                      00000000 R       02       OUTPUT_FILE                     ********   X   03
HELP_LIBRARY                    00000004 R       02       P0BR                            ********   X   03
HLP$M_GROUP                   = 00000004                  P0LR                            ********   X   03
HLP$M_PROCESS                 = 00000002                  PCB                             ********   X   03
HLP$M_SYSTEM                  = 00000008                  PCB$C_LENGTH                  = 00000120
INDIRECT_COMMAND                000002D3 RG      03       PCB$L_PHD                     = 0000006C
INDRAB                          ********    X    03       PCB$L_PID                     = 00000060
INPUT_BUFFER                    ********    X    03       PCB$T_LNAME                   = 00000070
INPUT_BUF_LEN                   ********    X    03       PCBADR                          ********   X   03
INPUT_LEN                       ********    X    03       PHD                             ********   X   03
INPUT_RAB                       0000010C RG      02       PHD$C_LENGTH                  = 0000017C
INSTR_VALUE                     000007A1 R       03       PHD$L_P0BR                    = 000000C8
IO$_WRITEVBLK                   ********    X    03       PHD$S_P0LR                    = 00000018
LBR$OUTPUT_HELP                 ********    X    03       PHD$V_P0LR                    = 00000000
LIB$GET_INPUT                   ********    X    03       PHDADR                          ********   X   03
```

```
PIX_WIDTH                00000134 RG    02
PRINT                    ********    X  03
PRINT_SYMBOLS            ********    X  03
PROC_INDEX               00000130 RG    02
PROC_NAME                00000118 RG    02
PROC_PID                 00000138 RG    02
PROMPT                   00000100 R     03
PROMPT_LEN             = 00000005
RAB$L_FAB              = 0000003C
RAB$L_RBF              = 00000028
RAB$L_STV              = 0000000C
RAB$L_UBF              = 00000024
RAB$W_RSZ              = 00000022
RAB$W_USZ              = 00000020
READ_SYMFILE             00000B14 RG    03
RELOCATE_BASE            00000114 RG    02
REPEAT_COMMAND           ********    X  03
REQMEM                   ********    X  03
REWIND_STB               ********    X  03
RMS$_EOF                 ********    X  03
SAVE_COMMAND             ********    X  03
SCH$GL_CURPCB            ********    X  03
SCH$GL_MAXPIX            ********    X  03
SCH$GL_PCBVEC            ********    X  03
SDA_KEY                  ********    X  03
SDA_STATE                ********    X  03
SEARCH_MEMORY            00000BD4 RG    03
SET_HEADING              ********    X  03
SET_PROCESS              00000846 RG    03
SHOW_EXPR                0000047C RG    03
SHOW_P0                  ********    X  03
SHOW_P1                  ********    X  03
SHOW_PROCESS             00000A83 RG    03
SHOW_SYMBOL              000003F3 RG    03
SHOW_SYSTEM              ********    X  03
SHR$_OPENIN            = 00001098
SHR$_SYNTAX            = 000010F8
SMG$_EOF                 ********    X  03
SS$_RESIGNAL           = 00000918
STB                      ********    X  03
STBF                     ********    X  03
STR$UPCASE               ********    X  03
STS$K_ERROR            = 00000002
STS$K_SEVERE           = 00000004
STS$K_WARNING          = 00000000
STS$S_SEVERITY         = 00000003
STS$V_SEVERITY         = 00000000
SUB_HEADING              ********    X  03
SYMBOLIZE                ********    X  03
SYMBOL_DESC              00000148 RG    02
SYMBOL_NAME              00000120 RG    02
SYMBOL_VALUE             ********    X  03
SYM_DEFAULT              00000B10 R     03
SYM_DEF_LEN            = 00000004
SYS$ASCTIM               ********   GX  03
SYS$CLOSE                ********   GX  03
SYS$CONNECT              ********   GX  03
SYS$FAO                  ********    X  03
SYS$GET                  ********   GX  03
SYS$GETMSG               ********   GX  03
SYS$OPEN                 ********   GX  03
SYS$PUT                  ********   GX  03
SYS$QIOW                 ********   GX  03
TPA$K_COUNT0           = 00000008
TPA$L_NUMBER           = 0000001C
TPA$L_STRINGCNT        = 00000008
TPA$L_STRINGPTR        = 0000000C
TPA$L_TOKENCNT         = 00000010
TPA$L_TOKENPTR         = 00000014
TPA$M_ABBREV           = 00000002
TPARSE_BLOCK             0000001C R     02
TRYMEM                   ********    X  03
TRY_SEQUENCE             000007F3 R     03
TT$V_SCOPE               ********    X  03
TT2$V_EDITING            ********    X  03
TT_CHAN                  ********    X  03
XL_STRING                00000796 R     03
```

```
                              +----------------+
                              ! Psect synopsis !
                              +----------------+

PSECT name                       Allocation          PSECT No.  Attributes
----------                       ----------          ---------  ----------
.  ABS  .                        00000000  (    0.)  00 (  0.)  NOPIC   USR   CON   ABS   LCL NOSHR NOEXE NORD   NOWRT NOVEC BYTE
$ABS$                            00000000  (    0.)  01 (  1.)  NOPIC   USR   CON   ABS   LCL NOSHR   EXE   RD     WRT NOVEC BYTE
SDADATA                          000001A5  (  421.)  02 (  2.)  NOPIC   USR   CON   REL   LCL NOSHR NOEXE   RD     WRT NOVEC BYTE
COMMANDS                         00000C87  ( 3207.)  03 (  3.)  NOPIC   USR   CON   REL   LCL NOSHR   EXE   RD   NOWRT NOVEC LONG
LITERALS                         00000109  (  265.)  04 (  4.)  NOPIC   USR   CON   REL   LCL NOSHR   EXE   RD   NOWRT NOVEC BYTE

                           +------------------------+
                           ! Performance indicators !
                           +------------------------+

Phase                    Page faults    CPU Time       Elapsed Time
-----                    -----------    --------       ------------
Initialization                    29    00:00:00.05    00:00:01.07
Command processing               153    00:00:00.59    00:00:04.52
Pass 1                           528    00:00:13.88    00:00:47.83
Symbol table sort                  0    00:00:01.92    00:00:08.76
Pass 2                           233    00:00:03.32    00:00:13.17
Symbol table output               23    00:00:00.14    00:00:00.45
Psect synopsis output              2    00:00:00.02    00:00:00.02
Cross-reference output             0    00:00:00.00    00:00:00.00
Assembler run totals             970    00:00:19.94    00:01:15.82

The working set limit was 2250 pages.
124778 bytes (244 pages) of virtual memory were used to buffer the intermediate code.
There were 100 pages of symbol table space allocated to hold 1844 non-local and 112 local symbols.
1290 source lines were read in Pass 1, producing 42 object records in Pass 2.
50 pages of virtual memory were used to define 48 macros.

                          +--------------------------+
                          ! Macro library statistics !
                          +--------------------------+

Macro library name                          Macros defined
------------------                          --------------
_$255$DUA28:[SDA.OBJ]SDALIB.MLB;1                    13
_$255$DUA28:[SYS.OBJ]LIB.MLB;1                        6
_$255$DUA28:[SYSLIB]STARLET.MLB;2                    25
TOTALS (all libraries)                               44

2088 GETS were required to define 44 macros.

There were no errors, warnings or information messages.

MACRO/LIS=LIS$:COMMANDS/OBJ=OBJ$:COMMANDS MSRC$:COMMANDS/UPDATE=(ENH$:COMMANDS)+EXECML$/LIB+LIB$:SDALIB/LIB
```

0351 AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY

DEVICE
LIS

CRASH
LIS

COMMANDS
LIS

DECODE
LIS