```
NNN         NNN     CCCCCCCCCCCC  PPPPPPPPPPPP
NNN         NNN     CCCCCCCCCCCC  PPPPPPPPPPPP
NNN         NNN     CCCCCCCCCCCC  PPPPPPPPPPPP
NNN         NNN  CCC              PPP         PPP
NNN         NNN  CCC              PPP         PPP
NNN         NNN  CCC              PPP         PPP
NNNNNN      NNN  CCC              PPP         PPP
NNNNNN      NNN  CCC              PPP         PPP
NNNNNN      NNN  CCC              PPP         PPP
NNN   NNN   NNN  CCC              PPPPPPPPPPPP
NNN   NNN   NNN  CCC              PPPPPPPPPPPP
NNN   NNN   NNN  CCC              PPPPPPPPPPPP
NNN      NNNNNN  CCC              PPP
NNN      NNNNNN  CCC              PPP
NNN      NNNNNN  CCC              PPP
NNN         NNN  CCC              PPP
NNN         NNN  CCC              PPP
NNN         NNN  CCC              PPP
NNN         NNN     CCCCCCCCCCCC  PPP
NNN         NNN     CCCCCCCCCCCC  PPP
NNN         NNN     CCCCCCCCCCCC  PPP
```

```
**FILE**ID**NCPPRSACT




NN      NN    CCCCCCCC  PPPPPPPP    PPPPPPPP    RRRRRRRR      SSSSSSSS    AAAAAA      CCCCCCCC  TTTTTTTTTT
NN      NN    CCCCCCCC  PPPPPPPP    PPPPPPPP    RRRRRRRR      SSSSSSSS    AAAAAA      CCCCCCCC  TTTTTTTTTT
NN      NN  CC          PP      PP  PP      PP  RR      RR  SS          AA      AA  CC              TT
NN      NN  CC          PP      PP  PP      PP  RR      RR  SS          AA      AA  CC              TT
NNNN    NN  CC          PP      PP  PP      PP  RR      RR  SS          AA      AA  CC              TT
NNNN    NN  CC          PP      PP  PP      PP  RR      RR  SS          AA      AA  CC              TT
NN  NN  NN  CC          PPPPPPPP    PPPPPPPP    RRRRRRRR      SSSSSS    AA      AA  CC              TT
NN  NN  NN  CC          PPPPPPPP    PPPPPPPP    RRRRRRRR      SSSSSS    AA      AA  CC              TT
NN    NNNN  CC          PP          PP          RR  RR              SS  AAAAAAAAAA  CC              TT
NN    NNNN  CC          PP          PP          RR  RR              SS  AAAAAAAAAA  CC              TT
NN      NN  CC          PP          PP          RR    RR            SS  AA      AA  CC              TT
NN      NN  CC          PP          PP          RR    RR            SS  AA      AA  CC              TT      ....
NN      NN    CCCCCCCC  PP          PP          RR      RR  SSSSSSSS    AA      AA    CCCCCCCC      TT      ....
NN      NN    CCCCCCCC  PP          PP          RR      RR  SSSSSSSS    AA      AA    CCCCCCCC      TT      ....

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
```

```
0001  0 %TITLE  'Parse Data and Action Routines'
0002  0 MODULE NCPPRSACT (IDENT = 'V04-000',
0003  0                 ADDRESSING_MODE(EXTERNAL=GENERAL),
0004  0                 ADDRESSING_MODE(NONEXTERNAL=GENERAL)) =
0005  1 BEGIN
0006  1
0007  1 !
0008  1 !****************************************************************************
0009  1 !*                                                                          *
0010  1 !*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                                 *
0011  1 !*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.                  *
0012  1 !*  ALL RIGHTS RESERVED.                                                    *
0013  1 !*                                                                          *
0014  1 !*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED   *
0015  1 !*  ONLY IN  ACCORDANCE WITH  THE  TERMS  OF  SUCH  LICENSE  AND WITH THE   *
0016  1 !*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR  ANY  OTHER   *
0017  1 !*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY   *
0018  1 !*  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE IS  HEREBY   *
0019  1 !*  TRANSFERRED.                                                            *
0020  1 !*                                                                          *
0021  1 !*  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE WITHOUT NOTICE   *
0022  1 !*  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL EQUIPMENT   *
0023  1 !*  CORPORATION.                                                            *
0024  1 !*                                                                          *
0025  1 !*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR  RELIABILITY OF ITS   *
0026  1 !*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.                 *
0027  1 !*                                                                          *
0028  1 !*                                                                          *
0029  1 !****************************************************************************
0030  1 !
0031  1
0032  1 !++
0033  1 ! FACILITY:     Network Control Program (NCP)
0034  1 !
0035  1 ! ABSTRACT:
0036  1 !
0037  1 !       Data and Action routines for parsing
0038  1 !
0039  1 ! ENVIRONMENT:  VAX/VMS Operating System
0040  1 !
0041  1 ! AUTHOR:       Darrell Duffy   , CREATION DATE: 28-August-79
0042  1 !
0043  1 ! MODIFIED BY:
0044  1 !
0045  1 !       V03-002 RPG0002         Bob Grosso              29-Jul-1982
0046  1 !               Add new routine to supply prompting help.
0047  1 !
0048  1 !       V03-001 RPG0001         Bob Grosso              30-Jun-1982
0049  1 !               Add routines to support channel lists.
0050  1 !
0051  1 !       V002    TMH0002         Tim Halvorsen   04-Nov-1981
0052  1 !               Remove duplicate definition of NCP$_NXT_STATE as
0053  1 !               both an external and a global.  It should have been
0054  1 !               one or the other.
0055  1 !
0056  1 !       V001    TMH0001         Tim Halvorsen   18-Jun-1981
0057  1 !               Make all external references longword relative.
```

```
:    58          0058  1 !--
```

```
1 %SBTTL  'Definitions'
1
1 !
1 ! TABLE OF CONTENTS:
1 !
1
1 FORWARD ROUTINE
1         NCP$SIG_CMDERR;                    ! Signal a command error
1
1 !
1 ! INCLUDE FILES:
1 !
1
1 LIBRARY 'LIB$:NCPLIBRY.L32';
1 LIBRARY 'SYS$LIBRARY:STARLET.L32';
1
1 !
1 ! OWN STORAGE
1 !
1
1 GLOBAL LITERAL
1         ACT$C_RNGLSTMAX = 2 * MAX_RNGLST_PAIRS;
1
1 GLOBAL
1         ACT$GA_RNGLST : VECTOR [ACT$C_RNGLSTMAX + 1, WORD],
1                                            ! Channel list vector,
1                                            !  ACT$GA_RNGLST [0] contains count
1         NCP$_NXT_STATE : VECTOR [2],       ! Next state table and keytable to use
1         NCP$_PRSCMD_DSC : VECTOR [2];      ! Descriptor of the parsed command
1
1 !
1 ! EXTERNAL REFERENCES:
1 !
1
1 EXTERNAL LITERAL
1         NCP$_AMBCMD,                       ! Error status for ambiguous command
1         NCP$_INVCMD,                       ! Error status for invalid command
1         NCP$_CMDCAN,                       ! Command canceled
1         NCP$_CMDERR,                       ! I/O error
1         NCP$_FIELDLIM,                     ! Too many fields
1         NCP$_NOTDONE,                      ! Prompt for non-terminal command
1         NCP$_PRMRNG,                       ! Parameter range status code
1         NCP$_PRMLEN,                       ! Parameter length status code
1         NCP$_SYNTAX,                       ! Syntax error status
1         LIB$_SYNTAXERR                     ! Syntax error status from LIB$TPARSE
1         ;
1
1 EXTERNAL ROUTINE
1         LBR$OUTPUT_HELP,                   ! Prompting help
1         LIB$GET_INPUT,                     ! for prompting for help
1         LIB$PUT_OUTPUT,                    ! for printing help
1         LIB$TPARSE;                        ! The table parser
1
1 EXTERNAL ROUTINE
1         NCP$WRITE_LINE,                    ! Write a line to sys$output
1         NCP$READ_CMD,                      ! Read a command with continuation
1         NCP$CMD_TERM_Q                     ! Is input device a terminal?
```

```
        ;
EXTERNAL
        NCP$_CMDBUF_DSC : VECTOR,        ! Command buffer descriptor
        ACT$GL_PMT_Q                     ! True for prompting active
        ;
```

```
1 %SBTTL 'NCP$PARSE_CMD  Parse Command'
1 GLOBAL ROUTINE NCP$PARSE_CMD (INP_DSC, ST_TBL, KEY_TBL, RTN_DSC) =       !
1
1 !++
1 ! FUNCTIONAL DESCRIPTION:
1 !
1 !       Calls LIB$TPARSE with the parse state table
1 !
1 ! FORMAL PARAMETERS:
1 !
1 !       INP_DSC         Address of descriptor of line
1 !       ST_TBL          Address of state table to use
1 !       KEY_TBL         Address of keyword table
1 !       RTN_DSC         Address of descriptor to receive remainder
1 !                       of command line
1 !
1 ! IMPLICIT INPUTS:
1 !
1 !       NONE
1 !
1 ! IMPLICIT OUTPUTS:
1 !
1 !       NONE
1 !
1 ! ROUTINE VALUE:
1 ! COMPLETION CODES:
1 !
1 !       Return status from LIB$TPARSE is signaled if syntax error
1 !
1 ! SIDE EFFECTS:
1 !
1 !       NONE
1 !
1 !--
1
2    BEGIN
2
2    MAP
2       INP_DSC : REF VECTOR [2],        ! Dsc of input line
2       RTN_DSC : REF VECTOR [2]         ! Returned dsc of remainder
2       ;
2
2    LOCAL
2       STATUS                           ! Returned status
2       ;            !
2
2    OWN
2       PARSE_STATE :                    ! Parse state table to LIB$TPARSE
2               BBLOCK [TPA$K_LENGTH0]
2       ;
2
2    NCP$_PRSCMD_DSC [0] = .INP_DSC [0]; ! Save the descriptor of the command
2    NCP$_PRSCMD_DSC [1] = .INP_DSC [1];
2
2    PARSE_STATE [TPA$L_COUNT] =         ! Set count of arg block
2               TPA$K_COUNT0;
2    PARSE_STATE [TPA$L_OPTIONS] =       ! Set for minimum abbreviation
```

```
2                  TPA$M_ABBREV;
2      PARSE_STATE [TPA$L_STRINGCNT] =         ! Setup the initial command string
2                  .INP_DSC [0];
2      PARSE_STATE [TPA$L_STRINGPTR] =
2                  .INP_DSC [1];
2
2      NCP$_NXT_STATE [0] = .ST_TBL;           ! Setup the first round of state
2      NCP$_NXT_STATE [1] = .KEY_TBL;
2
2      DO
3          BEGIN
3          LOCAL
3              STATES,                         ! Temp for state table address
3              KEYS                            ! Temp for keyword table address
3              ;
3          STATES = .NCP$_NXT_STATE [0];       ! Set up this round
3          KEYS   = .NCP$_NXT_STATE [1];
3
3          NCP$_NXT_STATE = 0;                 ! Zero next round
3
3          STATUS = LIB$TPARSE (PARSE_STATE,! Parse the string
3                      .STATES, .KEYS);
3          END
3                                              ! While no error and there is a next
3      WHILE .STATUS AND (.NCP$_NXT_STATE [0] NEQ 0) ! round, keep going
2      ;
2
2      IF NOT .STATUS                          ! Returned an error?
2      THEN
3          BEGIN                               ! Yes, then signal it somehow
3          IF .STATUS EQLU LIB$_SYNTAXERR      ! Is it a vanilla syntax error?
3          THEN                                ! Yes, then build the arguments
3              NCP$SIG_CMDERR                  ! Signal a command syntax error
4                  (   (IF .PARSE_STATE [TPA$V_AMBIG]
4                      THEN NCP$_AMBCMD        ! Ambiguous keyword or
4                      ELSE NCP$_SYNTAX        ! Syntax error
3                      ) OR STS$K_SEVERE,      ! Severe error to stop
3                  .PARSE_STATE [TPA$L_TOKENCNT],
3                  .PARSE_STATE [TPA$L_TOKENPTR],
3                  .PARSE_STATE [TPA$L_STRINGCNT],
3                  .PARSE_STATE [TPA$L_STRINGPTR]
3                  )
3          ELSE
3              SIGNAL (.STATUS)                ! Punt the signal of anything else
3
3          END
2      ;
2
2      RTN_DSC [0] = .PARSE_STATE              ! Return the remainder of the string
2                  [TPA$L_STRINGCNT];
2      RTN_DSC [1] = .PARSE_STATE
2                  [TPA$L_STRINGPTR];
2      RETURN .STATUS                          ! and the status of the call
2
1      END;
```

```
                                                  .TITLE  NCPPRSACT Parse Data and Action Routines
                                                  .IDENT  \V04-000\

                                                  .PSECT  $OWN$,NOEXE,2

                                  00000 PARSE_STATE:
                                                  .BLKB   36

                                                  .PSECT  $GLOBAL$,NOEXE,2

                                  00000 ACT$GA_RNGLST::
                                                  .BLKB   66
                                  00042           .BLKB   2
                                  00044 NCP$_NXT_STATE::
                                                  .BLKB   8
                                  0004C NCP$_PRSCMD_DSC::
                                                  .BLKB   8

                                        ACT$C_RNGLSTMAX==    32
                                                  .EXTRN  NCP$_AMBCMD, NCP$_INVCMD
                                                  .EXTRN  NCP$_CMDCAN, NCP$_CMDERR
                                                  .EXTRN  NCP$_FIELDLIM, NCP$_NOTDONE
                                                  .EXTRN  NCP$_PRMRNG, NCP$_PRMLEN
                                                  .EXTRN  NCP$_SYNTAX, LIB$_SYNTAXERR
                                                  .EXTRN  LBR$OUTPUT_HELP
                                                  .EXTRN  LIB$GET_INPUT, LIB$PUT_OUTPUT
                                                  .EXTRN  LIB$TPARSE, NCP$WRITE_LINE
                                                  .EXTRN  NCP$READ_CMD, NCP$CMD_TERM_Q
                                                  .EXTRN  NCP$_CMDBUF_DSC
                                                  .EXTRN  ACT$GL_PMT_Q

                                                  .PSECT  $CODE$,NOWRT,2

                             001C 00000           .ENTRY  NCP$PARSE_CMD, Save R2,R3,R4                          ; 0123
            54 00000000'   00   9E 00002          MOVAB   NCP$_NXT_STATE, R4                                    ;
            53 00000000'   00   9E 00009          MOVAB   PARSE_STATE+8, R3                                     ;
            50        04   AC   D0 00010          MOVL    INP_DSC, R0                                           ; 0173
         08 A4             60   7D 00014          MOVQ    (R0), NCP$_PRSCMD_DSC                                 ;
         F8 A3             08   D0 00018          MOVL    #8, PARSE_STATE                                       ; 0176
         FC A3             02   D0 0001C          MOVL    #2, PARSE_STATE+4                                     ; 0178
            63             60   7D 00020          MOVQ    (R0), PARSE_STATE+8                                   ; 0181
            64        08   AC   7D 00023          MOVQ    ST_TBL, NCP$_NXT_STATE                                ; 0185
            51             64   D0 00027 1$:      MOVL    NCP$_NXT_STATE, STATES                                ; 0194
            50        04   A4   D0 0002A          MOVL    NCP$_NXT_STATE+4, KEYS                                ; 0195
                           64   D4 0002E          CLRL    NCP$_NXT_STATE                                        ; 0197
                           50   DD 00030          PUSHL   KEYS                                                  ; 0200
                           51   DD 00032          PUSHL   STATES                                                ;
                      F8   A3   9F 00034          PUSHAB  PARSE_STATE                                           ; 0199
   00000000G 00            03   FB 00037          CALLS   #3, LIB$TPARSE                                        ;
            52             50   D0 0003E          MOVL    R0, STATUS                                            ;
            07             52   E9 00041          BLBC    STATUS, 2$                                            ; 0203
                           64   D5 00044          TSTL    NCP$_NXT_STATE                                        ;
                           DF   12 00046          BNEQ    1$                                                    ;
            3A             52   E8 00048          BLBS    STATUS, 6$                                            ; 0206
   00000000G 8F            52   D1 0004B 2$:      CMPL    STATUS, #LIB$_SYNTAXERR                               ; 0209
                           28   12 00052          BNEQ    5$                                                    ;
            7E             63   7D 00054          MOVQ    PARSE_STATE+8, -(SP)                                  ; 0218
```

```
                    7E        08  A3  7D 00057       MOVQ    PARSE_STATE+16, -(SP)              ; 0216
                    09        FE  A3  E9 0005B       BLBC    PARSE_STATE+6, 3$                  ; 0212
                    50 00000000G  8F  D0 0005F       MOVL    #NCP$_AMBCMD, R0                   ;
                                  07  11 00066       BRB     4$                                 ;
                    50 00000000G  8F  D0 00068 3$:   MOVL    #NCP$_SYNTAX, R0                   ;
             7E     50            04  C9 0006F 4$:   BISL3   #4, R0, -(SP)                      ; 0215
          00000000V 00            05  FB 00073       CALLS   #5, NCP$SIG_CMDERR                 ;
                                  09  11 0007A       BRB     6$                                 ; 0212
                                  52  DD 0007C 5$:   PUSHL   STATUS                             ; 0222
          00000000G 00            01  FB 0007E       CALLS   #1, LIB$SIGNAL                     ;
                    50        10  AC  D0 00085 6$:   MOVL    RTN_DSC, R0                        ; 0227
                    60            63  7D 00089       MOVQ    PARSE_STATE+8, (R0)                ;
                    50            52  D0 0008C       MOVL    STATUS, R0                         ; 0231
                                      04 0008F       RET                                        ; 0233

; Routine Size:  144 bytes,    Routine Base:  $CODE$ + 0000
```

```
%SBTTL  'NCP$SIG_CMDERR  Signal a command syntax error'
GLOBAL ROUTINE NCP$SIG_CMDERR (CODE, TKN_CNT, TKN_PTR, STR_CNT, STR_PTR) =

!++
! FUNCTIONAL DESCRIPTION:
!
!       A command error is signalled for printing.  The signal name is
!       code. The remainder of the arguments give the user context for his
!       error.  The context is cleaned up and passed on for printing.
!
!       If prompting is not active, we signal stop to avoid further error
!       messages being printed.  If prompting is active, we signal so that
!       the prompt will allow the user to correct his mistake.
!
! FORMAL PARAMETERS:
!
!       CODE            Value of status code to signal
!       TKN_CNT         Value of size of token in error
!       TKN_PTR         Address of token in error
!       STR_CNT         Value of size of remaining part of command
!       STR_PTR         Address of remaining part of command
!
! IMPLICIT INPUTS:
!
!       ACT$GL_PMT_Q    True for prompting active
!       NCP$_PRSCMD_DSC Descriptor of parsed command
!
! IMPLICIT OUTPUTS:
!
!       NONE
!
! ROUTINE VALUE:
! COMPLETION CODES:
!
!       Error condition signalled
!
! SIDE EFFECTS:
!
!       NONE
!
!--
    BEGIN

    LITERAL
        WDO_SIZ = 30                        ! Window size for error text
        ;

        LOCAL
        BFR_CNT,                            ! Before counter for error
        BFR_PTR,                            ! Before pointer for error
        AFT_CNT
        ;

    IF  (                                   ! Check token position for reasonable
        ( (.TKN_PTR + .TKN_CNT) GEQA .STR_PTR )
        AND
        ( (.TKN_PTR + .TKN_CNT) LSSA (.STR_PTR + .STR_CNT) )
;  294         0291 3            )
;  295         0292 2        THEN                                    ! If so then fix it up
;  296         0293 3            BEGIN
;  297         0294 4            STR_CNT = (.STR_PTR + .STR_CNT) ! Position string beyond token
;  298         0295 3                    - (.TKN_PTR + .TKN_CNT);
;  299         0296 3            STR_PTR = .TKN_PTR + .TKN_CNT
;  300         0297 3            END
;  301         0298 2        ELSE                                    ! If not reasonable, then punt it
;  302         0299 3            BEGIN
;  303         0300 3            TKN_PTR = .STR_PTR;                 ! Make token zero length
;  304         0301 3            TKN_CNT = 0
;  305         0302 3            END
;  306         0303 2        ;
;  307         0304 2
;  308         0305 3        IF (BFR_CNT =                           ! Use some characters on either side
;  309         0306 3            .TKN_PTR - .NCP$_PRSCMD_DSC [1]) ! of the bad token
;  310         0307 2            GTRU
;  311         0308 2            WDO_SIZ
;  312         0309 2        THEN                                    ! But not too many since the command
;  313         0310 3            BEGIN                               ! may be quite long
;  314         0311 3            BFR_PTR = .TKN_PTR - WDO_SIZ;
;  315         0312 3            BFR_CNT = WDO_SIZ
;  316         0313 3            END
;  317         0314 2        ELSE                                    ! On short commands use it all
;  318         0315 2            BFR_PTR = .NCP$_PRSCMD_DSC [1]
;  319         0316 2        ;
;  320         0317 3        IF (AFT_CNT =                           ! Compute the after part too
;  321         0318 3            .STR_CNT)
;  322         0319 2            GTRU
;  323         0320 2            WDO_SIZ
;  324         0321 2        THEN                                    ! for some following context
;  325         0322 2            AFT_CNT = WDO_SIZ
;  326         0323 2        ;
;  327         0324 2        IF .ACT$GL_PMT_Q                        ! Is prompting active?
;  328         0325 2        THEN
;  329         0326 2            SIGNAL                              ! Signal to allow correction of errors
;  330         0327 2                (.CODE, 6,                      ! Signal a syntax error
;  331         0328 2                .BFR_CNT, .BFR_PTR,             ! with all the context
;  332         0329 2                .TKN_CNT, .TKN_PTR,
;  333         0330 2                .AFT_CNT, .STR_PTR
;  334         0331 2                )
;  335         0332 2        ELSE
;  336         0333 2            SIGNAL_STOP                         ! Signal stop to prevent further msgs
;  337         0334 2                (.CODE, 6,                      ! Signal a syntax error
;  338         0335 2                .BFR_CNT, .BFR_PTR,             ! with all the context
;  339         0336 2                .TKN_CNT, .TKN_PTR,
;  340         0337 2                .AFT_CNT, .STR_PTR
;  341         0338 2                )
;  342         0339 2
;  343         0340 1        END;                    ! End of routine



                                      0004 00000          .ENTRY  NCP$SIG_CMDERR, Save R2                              ; 0235
                51     0C   AC     08   AC  C1 00002          ADDL3   TKN_CNT, TKN_PTR, R1                                 ; 0288
```

```
                                                               M 9
NCPPRSACT       Parse Data and Action Routines                 15-Sep-1984 23:51:04    VAX-11 Bliss-32 V4.0-742              Page  11
V04-000         NCP$SIG_CMDERR  Signal a command syntax error  14-Sep-1984 12:48:15    [NCP.SRC]NCPPRSACT.B32;1                   (4)

                     14   AC          51  D1 00008          CMPL    R1, STR_PTR                                      ;
                                      1C  1F 0000C          BLSSU   1$                                               ;
            50       14   AC     10   AC  C1 0000E          ADDL3   STR_CNT, STR_PTR, R0                             ; 0290
                          50          51  D1 00014          CMPL    R1, R0                                           ;
                                      11  1E 00017          BGEQU   1$                                               ;
            50       14   AC     10   AC  C1 00019          ADDL3   STR_CNT, STR_PTR, R0                             ; 0294
       10   AC            50          51  C3 0001F          SUBL3   R1, R0, STR_CNT                                  ; 0295
                     14   AC          51  D0 00024          MOVL    R1, STR_PTR                                      ; 0296
                                      08  11 00028          BRB     2$                                               ;
                     0C   AC     14   AC  D0 0002A 1$:      MOVL    STR_PTR, TKN_PTR                                 ; 0300
                                 08   AC  D4 0002F          CLRL    TKN_CNT                                          ; 0301
                          50 00000000'  00  D0 00032 2$:    MOVL    NCP$_PRSCMD_DSC+4, R0                            ; 0306
            51       0C   AC          50  C3 00039          SUBL3   R0, TKN_PTR, BFR_CNT                             ;
                          1E          51  D1 0003E          CMPL    BFR_CNT, #30                                     ; 0307
                                      0A  1B 00041          BLEQU   3$                                               ;
            52       0C   AC          1E  C3 00043          SUBL3   #30, TKN_PTR, BFR_PTR                            ; 0311
                          51          1E  D0 00048          MOVL    #30, BFR_CNT                                     ; 0312
                                      03  11 0004B          BRB     4$                                               ;
                          52          50  D0 0004D 3$:      MOVL    R0, BFR_PTR                                      ; 0315
                          50     10   AC  D0 00050 4$:      MOVL    STR_CNT, AFT_CNT                                 ; 0318
                          1E          50  D1 00054          CMPL    AFT_CNT, #30                                     ; 0319
                                      03  1B 00057          BLEQU   5$                                               ;
                          50          1E  D0 00059          MOVL    #30, AFT_CNT                                     ; 0322
                          18 00000000G  00  E9 0005C 5$:    BLBC    ACT$GL_PMT_Q, 6$                                 ; 0324
                                 14   AC  DD 00063          PUSHL   STR_PTR                                          ; 0330
                                      50  DD 00066          PUSHL   AFT_CNT                                          ;
                          7E     08   AC  7D 00068          MOVQ    TKN_CNT, -(SP)                                   ; 0329
                                      06  BB 0006C          PUSHR   #^M<R1,R2>                                       ; 0328
                                      06  DD 0006E          PUSHL   #6                                               ; 0327
                                 04   AC  DD 00070          PUSHL   CODE                                             ;
              00000000G   00          08  FB 00073          CALLS   #8, LIB$SIGNAL                                   ;
                                      04 0007A              RET                                                      ;
                                 14   AC  DD 0007B 6$:      PUSHL   STR_PTR                                          ; 0337
                                      50  DD 0007E          PUSHL   AFT_CNT                                          ;
                          7E     08   AC  7D 00080          MOVQ    TKN_CNT, -(SP)                                   ; 0336
                                      06  BB 00084          PUSHR   #^M<R1,R2>                                       ; 0335
                                      06  DD 00086          PUSHL   #6                                               ; 0334
                                 04   AC  DD 00088          PUSHL   CODE                                             ;
              00000000G   00          08  FB 0008B          CALLS   #8, LIB$STOP                                     ;
                                      04 00092              RET                                                      ; 0340

; Routine Size:  147 bytes,    Routine Base:  $CODE$ + 0090
```

```
%SBTTL  'ACT$INV_COMMAND  Action routine for invalid command'
GLOBAL ROUTINE ACT$INV_COMMAND (OPT, STRCNT, STRPTR, TKNCNT, TKNPTR,
                CHR, NUM, PARAM) =      !

!++
! FUNCTIONAL DESCRIPTION:
!
!       Action routine to signal an invalid command.  Signal is either
!       for invalid command or ambiguous command if the AMBIG bit is
!       set in the parse options.
!
! FORMAL PARAMETERS:
!
!       Parse state table
!       OPT             Value of the parse options
!       STRCNT          Size of the remainder of the command line
!       STRPTR          Address of the remainder of the command line
!       TKNCNT          Size of the token in error
!       TKNPTR          Address of the token in error
!
! IMPLICIT INPUTS:
!
!       NONE
!
! IMPLICIT OUTPUTS:
!
!       NONE
!
! ROUTINE VALUE:
! COMPLETION CODES:
!
!       NONE
!
! SIDE EFFECTS:
!
!       NONE
!
!--

    BEGIN

    NCP$SIG_CMDERR                          ! Signal the error
        (   ( IF (.OPT AND TPA$M_AMBIG)     ! Use the appropriate code
                NEQ 0
              THEN NCP$_AMBCMD              ! based on the ambiguous option bit
              ELSE NCP$_INVCMD
            ) OR STS$K_SEVERE,              ! Make severe to stop
        .TKNCNT, .TKNPTR,                   ! the token in error
        .STRCNT, .STRPTR                    ! the remainder of the command line
        )

    END;                            ! End of routine
```

```
                                  0000 00000          .ENTRY  ACT$INV_COMMAND, Save nothing             ; 0342
                  7E      08   AC   7D 00002          MOVQ    STRCNT, -(SP)                             ; 0389
                  7E      10   AC   7D 00006          MOVQ    TKNCNT, -(SP)                             ; 0388
        09        6C           30   E1 0000A          BBC     #48, OPT, 1$                              ; 0384
                  50 00000000G 8F   D0 0000E          MOVL    #NCP$_AMBCMD, R0                          ; 0383
                               07   11 00015          BRB     2$                                        ;
                  50 00000000G 8F   D0 00017 1$:      MOVL    #NCP$_INVCMD, R0                          ;
        7E        50           04   C9 0001E 2$:      BISL3   #4, R0, -(SP)                             ; 0387
             FF46 CF           05   FB 00022          CALLS   #5, NCP$SIG_CMDERR                        ;
                                    04 00027          RET                                               ; 0392

; Routine Size:  40 bytes,    Routine Base:  $CODE$ + 0123
;  398      0393  1 %SBTTL  'ACT$TMPSTR  Save a temporary string'
;  399      0394  1 GLOBAL ROUTINE ACT$TMPSTR (OPT, STRCNT, STRPTR, TKNCNT, TKNPTR,
;  400      0395  1                         CHR, NUM, PARAM) =       !
;  401      0396  1
;  402      0397  1 !++
;  403      0398  1 ! FUNCTIONAL DESCRIPTION:
;  404      0399  1 !
;  405      0400  1 !       Action routine to save a temporary string in a descriptor
;  406      0401  1 !
;  407      0402  1 ! FORMAL PARAMETERS:
;  408      0403  1 !
;  409      0404  1 !       Parse state table
;  410      0405  1 !       TKNCNT          Count of string
;  411      0406  1 !       TKNPTR          Address of string
;  412      0407  1 !       PARAM           Address of string descriptor to return
;  413      0408  1 !
;  414      0409  1 ! IMPLICIT INPUTS:
;  415      0410  1 !
;  416      0411  1 !       NONE
;  417      0412  1 !
;  418      0413  1 ! IMPLICIT OUTPUTS:
;  419      0414  1 !
;  420      0415  1 !       NONE
;  421      0416  1 !
;  422      0417  1 ! ROUTINE VALUE:
;  423      0418  1 ! COMPLETION CODES:
;  424      0419  1 !
;  425      0420  1 !       SUCCESS
;  426      0421  1 !
;  427      0422  1 ! SIDE EFFECTS:
;  428      0423  1 !
;  429      0424  1 !       NONE
;  430      0425  1 !
;  431      0426  1 !--
;  432      0427  1
;  433      0428  2     BEGIN
;  434      0429  2
;  435      0430  2     MAP
;  436      0431  2         PARAM : REF VECTOR [2]           ! Address of string descriptor
;  437      0432  2         ;
;  438      0433  2     PARAM [0] = .TKNCNT;                 ! Fill in the string descriptor
;  439      0434  2     PARAM [1] = .TKNPTR;
;  440      0435  2     RETURN SUCCESS
;  441      0436  1     END;



                                    0000 00000          .ENTRY  ACT$TMPSTR, Save nothing                        : 0394
                   50       20   AC   D0 00002          MOVL    PARAM, R0                                       : 0433
                   60       10   AC   7D 00006          MOVQ    TKNCNT, (R0)                                    :
                   50            01   D0 0000A          MOVL    #1, R0                                          : 0435
                                      04 0000D          RET                                                     : 0436

; Routine Size:  14 bytes,    Routine Base:  $CODE$ + 014B
```

```
;  443          0437  1 %SBTTL  'ACT$BLNK_SIG  Blanks are significant'
;  444          0438  1 GLOBAL ROUTINE ACT$BLNK_SIG (OPTIONS) = !
;  445          0439  1
;  446          0440  1 !++
;  447          0441  1 ! FUNCTIONAL DESCRIPTION:
;  448          0442  1 !
;  449          0443  1 !       Set parse options so blanks (spaces, tabs) are significant
;  450          0444  1 !
;  451          0445  1 ! FORMAL PARAMETERS:
;  452          0446  1 !
;  453          0447  1 !       Parse state table
;  454          0448  1 !       OPT             Options longword
;  455          0449  1 !
;  456          0450  1 ! IMPLICIT INPUTS:
;  457          0451  1 !
;  458          0452  1 !       NONE
;  459          0453  1 !
;  460          0454  1 ! IMPLICIT OUTPUTS:
;  461          0455  1 !
;  462          0456  1 !       NONE
;  463          0457  1 !
;  464          0458  1 ! ROUTINE VALUE:
;  465          0459  1 ! COMPLETION CODES:
;  466          0460  1 !
;  467          0461  1 !       Success
;  468          0462  1 !
;  469          0463  1 ! SIDE EFFECTS:
;  470          0464  1 !
;  471          0465  1 !       NONE
;  472          0466  1 !
;  473          0467  1 !--
;  474          0468  1
;  475          0469  2     BEGIN
;  476          0470  2
;  477          0471  2                                             ! Blanks are significant
;  478          0472  2     OPTIONS = .OPTIONS OR TPA$M_BLANKS ;
;  479          0473  2     RETURN SUCCESS
;  480          0474  2
;  481          0475  1     END;



                                        0000 00000          .ENTRY  ACT$BLNK_SIG, Save nothing                          ; 0438
                      04   AC        01  88 00002           BISB2   #1, OPTIONS                                          ; 0472
                           50        01  D0 00006           MOVL    #1, R0                                               ; 0473
                                         04 00009           RET                                                          ; 0475

; Routine Size:  10 bytes,    Routine Base:  $CODE$ + 0159
;  483           0476   1 %SBTTL  'ACT$BLNK_NSIG  Blanks are not significant'
;  484           0477   1 GLOBAL ROUTINE ACT$BLNK_NSIG (OPTIONS) =          !
;  485           0478   1
;  486           0479   1 !++
;  487           0480   1 ! FUNCTIONAL DESCRIPTION:
;  488           0481   1 !
;  489           0482   1 !       Parse options are set so blanks (spaces, tabs) are not significant
;  490           0483   1 !
;  491           0484   1 ! FORMAL PARAMETERS:
;  492           0485   1 !
;  493           0486   1 !       Parse state table
;  494           0487   1 !       OPT             Options longword
;  495           0488   1 !
;  496           0489   1 ! IMPLICIT INPUTS:
;  497           0490   1 !
;  498           0491   1 !       NONE
;  499           0492   1 !
;  500           0493   1 ! IMPLICIT OUTPUTS:
;  501           0494   1 !
;  502           0495   1 !       NONE
;  503           0496   1 !
;  504           0497   1 ! ROUTINE VALUE:
;  505           0498   1 ! COMPLETION CODES:
;  506           0499   1 !
;  507           0500   1 !       Success
;  508           0501   1 !
;  509           0502   1 ! SIDE EFFECTS:
;  510           0503   1 !
;  511           0504   1 !       NONE
;  512           0505   1 !
;  513           0506   1 !--
;  514           0507   1
;  515           0508   2     BEGIN
;  516           0509   2
;  517           0510   2     OPTIONS = .OPTIONS AND (NOT TPA$M_BLANKS) ; ! Blanks not significant
;  518           0511   2     RETURN SUCCESS
;  519           0512   2
;  520           0513   1     END;



                                  0000 00000          .ENTRY  ACT$BLNK_NSIG, Save nothing                          ; 0477
                 04    AC      01  8A 00002           BICB2   #1, OPTIONS                                          ; 0510
                       50      01  D0 00006           MOVL    #1, R0                                               ; 0511
                                   04 00009           RET                                                          ; 0513

; Routine Size:  10 bytes,    Routine Base:  $CODE$ + 0163
;   522         0514  1 %SBTTL  'ACT$ZAPTMPDSC  Zero a temporary descriptor'
;   523         0515  1 GLOBAL ROUTINE ACT$ZAPTMPDSC (OPT, STRCNT, STRPTR, TKNCTR,TKNPTR,
;   524         0516  1                                     CHR, NUM, PARAM) =        !
;   525         0517  1
;   526         0518  1 !++
;   527         0519  1 ! FUNCTIONAL DESCRIPTION:
;   528         0520  1 !
;   529         0521  1 !       Zero a list of descriptors for temporary strings
;   530         0522  1 !
;   531         0523  1 ! FORMAL PARAMETERS:
;   532         0524  1 !
;   533         0525  1 !       Parse state table
;   534         0526  1 !       PARAM           Address of PLIT for addresses of descriptor
;   535         0527  1 !
;   536         0528  1 ! IMPLICIT INPUTS:
;   537         0529  1 !
;   538         0530  1 !       NONE
;   539         0531  1 !
;   540         0532  1 ! IMPLICIT OUTPUTS:
;   541         0533  1 !
;   542         0534  1 !       NONE
;   543         0535  1 !
;   544         0536  1 ! ROUTINE VALUE:
;   545         0537  1 ! COMPLETION CODES:
;   546         0538  1 !
;   547         0539  1 !       SUCCESS
;   548         0540  1 !
;   549         0541  1 ! SIDE EFFECTS:
;   550         0542  1 !
;   551         0543  1 !       NONE
;   552         0544  1 !
;   553         0545  1 !--
;   554         0546  1
;   555         0547  2     BEGIN
;   556         0548  2
;   557         0549  2     MAP
;   558         0550  2         PARAM : REF VECTOR [10]             ! Expect a rather long list
;   559         0551  2         ;           !
;   560         0552  2
;   561         0553  2     INCRU IDX    FROM 0                      ! Scan the PLIT for addresses
;   562         0554  2                  TO .PARAM [-1] - 1          ! PLIT count
;   563         0555  2     DO
;   564         0556  2         .PARAM [.IDX] = 0                   ! Zap the count for the descriptor
;   565         0557  2         ;
;   566         0558  2     RETURN SUCCESS
;   567         0559  2
;   568         0560  1     END;


                                 000C 00000          .ENTRY  ACT$ZAPTMPDSC, Save R2,R3                       ; 0515
                    52      20 AC  D0 00002          MOVL    PARAM, R2                                       ; 0554
        53     FC   A2         01  C3 00006          SUBL3   #1, -4(R2), R3
                               50  D4 0000B          CLRL    IDX                                             ; 0553
                               08  11 0000D          BRB     2$                                              ;
```

```
                     51          6240   D0 0000F 1$:      MOVL    (R2)[IDX], R1                                    ; 0556
                                   61   D4 00013          CLRL    (R1)                                             ;
                                   50   D6 00015          INCL    IDX                                              ;
                     53            50   D1 00017 2$:      CMPL    IDX, R3                                          ;
                                   F3   1B 0001A          BLEQU   1$                                               ;
                     50            01   D0 0001C          MOVL    #1, R0                                           ; 0558
                                        04 0001F          RET                                                      ; 0560

; Routine Size:  32 bytes,    Routine Base:  $CODE$ + 016D
```

```
1 %SBTTL  'ACT$PRMPT  Action routine to prompt'
1 GLOBAL ROUTINE ACT$PRMPT (OPT, STRCNT, STRPTR, TKNCNT, TKNPTR,
1                 CHR, NUM, PARAM) =      !
1
1 !++
1 ! FUNCTIONAL DESCRIPTION:
1 !
1 !       Prompt for the remaining portion of a command if input is
1 !       interactive.
1 !
1 ! FORMAL PARAMETERS:
1 !
1 !       Parse state table
1 !       PARAM           Address of descriptor of prompt string
1 !
1 ! IMPLICIT INPUTS:
1 !
1 !       NCP$CMD_BUF_DSC Descriptor of command line buffer
1 !
1 ! IMPLICIT OUTPUTS:
1 !
1 !       NCP$_PRSCMD_DSC Descriptor of new command line portion
1 !
1 ! ROUTINE VALUE:
1 ! COMPLETION CODES:
1 !
1 !       Success or error signaled
1 !
1 ! SIDE EFFECTS:
1 !
1 !       NONE
1 !
1 !--
1
2   BEGIN
2
2   MAP
2       PARAM : REF VECTOR              ! Descriptor of prompt string
2       ;
2
2   LOCAL
2       STATUS
2       ;
2
2   IF NOT NCP$CMD_TERM_Q ()            ! If we are not on a terminal,
2   THEN
2       SIGNAL_STOP (NCP$_NOTDONE)      ! Dump off with a special error
2   ;
2
2   STATUS = NCP$READ_CMD               ! Read another command portion
2               (NCP$_CMDBUF_DSC,
2               .PARAM,
2               NCP$_PRSCMD_DSC
2               )
2   ;
2   IF NOT .STATUS                      ! Any error dumps us off
2   THEN
```

```
;  627      0618  3          BEGIN
;  628      0619  3          IF .STATUS EQL RMS$_EOF
;  629      0620  3          THEN SIGNAL_STOP (NCP$_CMDCAN) ! Command canceled if EOF
;  630      0621  3                                         ! Report any other error too
;  631      0622  3          ELSE SIGNAL_STOP (NCP$_CMDERR, 0, .STATUS)
;  632      0623  3          END
;  633      0624  2      ;
;  634      0625  2
;  635      0626  2      STRPTR = .NCP$_PRSCMD_DSC [1];          ! Set parse state to the portion
;  636      0627  2      STRCNT = .NCP$_PRSCMD_DSC [0];
;  637      0628  2
;  638      0629  2      RETURN SUCCESS                          ! And continue with parse
;  639      0630  2
;  640      0631  1      END;



                                     000C 00000          .ENTRY  ACT$PRMPT, Save R2,R3                          ; 0562
                    53 00000000'  00   9E 00002          MOVAB   NCP$_PRSCMD_DSC, R3                            ;
                    52 00000000G  00   9E 00009          MOVAB   LIB$STOP, R2                                   ;
           00000000G  00               00   FB 00010          CALLS   #0, NCP$CMD_TERM_Q                             ; 0605
                  09               50   E8 00017          BLBS    R0, 1$                                         ;
                 00000000G  8F     DD 0001A          PUSHL   #NCP$_NOTDONE                                  ; 0607
                    62             01   FB 00020          CALLS   #1, LIB$STOP                                   ;
                                   53   DD 00023  1$:    PUSHL   R3                                             ; 0611
                            20  AC     DD 00025          PUSHL   PARAM                                          ; 0612
                 00000000G  00     9F 00028          PUSHAB  NCP$_CMDBUF_DSC                                ; 0611
           00000000G  00               03   FB 0002E          CALLS   #3, NCP$READ_CMD                               ;
                    21             50   E8 00035          BLBS    STATUS, 3$                                     ; 0616
           0001827A  8F               50   D1 00038          CMPL    STATUS, #98938                                 ; 0619
                                   0B   12 0003F          BNEQ    2$                                             ;
                 00000000G  8F     DD 00041          PUSHL   #NCP$_CMDCAN                                   ; 0620
                    62             01   FB 00047          CALLS   #1, LIB$STOP                                   ;
                                   0D   11 0004A          BRB     3$                                             ;
                                   50   DD 0004C  2$:    PUSHL   STATUS                                         ; 0622
                                   7E   D4 0004E          CLRL    -(SP)                                          ;
                 00000000G  8F     DD 00050          PUSHL   #NCP$_CMDERR                                   ;
                    62             03   FB 00056          CALLS   #3, LIB$STOP                                   ;
              08    AC             63   7D 00059  3$:    MOVQ    NCP$_PRSCMD_DSC, STRCNT                        ; 0627
                    50             01   D0 0005D          MOVL    #1, R0                                         ; 0629
                                        04 00060          RET                                                    ; 0631

; Routine Size:  97 bytes,    Routine Base:  $CODE$ + 018D
```

```
%SBTTL  'ACT$NUM_RNG  Check numeric ranges'
GLOBAL ROUTINE ACT$NUM_RNG (OPT, STRCNT, STRPTR, TKNCNT, TKNPTR,
                CHR, NUM, PARAM) =          !

!++
! FUNCTIONAL DESCRIPTION:
!
!       Action routine to check numeric range of numeric parameter
!
! FORMAL PARAMETERS:
!
!       Parse state table
!       NUM             Value of the numeric token
!       PARAM           Address of range as two long words, low first
!
! IMPLICIT INPUTS:
!
!       NONE
!
! IMPLICIT OUTPUTS:
!
!       NONE
!
! ROUTINE VALUE:
! COMPLETION CODES:
!
!       Success or error signal
!
! SIDE EFFECTS:
!
!       NONE
!
!--

    BEGIN

    MAP
        PARAM : REF VECTOR [2]              ! Address of UPLIT (low, high)
        ;

    IF  .NUM GEQU .PARAM [0] AND            ! If inbounds return success
        .NUM LEQU .PARAM [1]
    THEN
        RETURN SUCCESS
    ELSE
        BEGIN
        NCP$SIG_CMDERR                      ! Signal parameter out of range
            (NCP$_PRMRNG,
            .TKNCNT, .TKNPTR,
            .STRCNT, .STRPTR
            )
        ;
        RETURN FAILURE                      ! Fail transition in parse table
        END;
    END;
```

```
                                   0000 00000          .ENTRY  ACT$NUM_RNG, Save nothing                    ; 0633
                  50        20   AC  D0 00002          MOVL    PARAM, R0                                    ; 0672
                  60        1C   AC  D1 00006          CMPL    NUM, (R0)                                    ;
                                 0B  1F 0000A          BLSSU   1$                                           ;
          04      A0        1C   AC  D1 0000C          CMPL    NUM, 4(R0)                                   ; 0673
                                 04  1A 00011          BGTRU   1$                                           ;
                  50             01  D0 00013          MOVL    #1, R0                                       ; 0677
                                     04 00016          RET                                                  ;
                  7E        08   AC  7D 00017  1$:     MOVQ    STRCNT, -(SP)                                ; 0681
                  7E        10   AC  7D 0001B          MOVQ    TKNCNT, -(SP)                                ; 0680
                       00000000G 8F  DD 0001F          PUSHL   #NCP$_PRMRNG                                 ; 0679
        FE78      CF             05  FB 00025          CALLS   #5, NCP$SIG_CMDERR                           ;
                                 50  D4 0002A          CLRL    R0                                           ; 0684
                                     04 0002C          RET                                                  ; 0686

; Routine Size:  45 bytes,    Routine Base:  $CODE$ + 01EE
```

```
1 %SBTTL  'ACT$NUM_RNGSAV  Check numeric ranges and store value in vector'
1 GLOBAL ROUTINE ACT$NUM_RNGSAV (OPT, STRCNT, STRPTR, TKNCNT, TKNPTR,
1                 CHR, NUM, PARAM) =        !
1
1 !++
1 ! FUNCTIONAL DESCRIPTION:
1 !
1 !       Action routine to check numeric range of numeric parameter
1 !       and store value in vector.
1 !
1 ! FORMAL PARAMETERS:
1 !
1 !       Parse state table
1 !       NUM             Value of the numeric token
1 !       PARAM           Address of range as two long words, low first
1 !
1 ! IMPLICIT INPUTS:
1 !
1 !       NONE
1 !
1 ! IMPLICIT OUTPUTS:
1 !
1 !       NONE
1 !
1 ! ROUTINE VALUE:
1 ! COMPLETION CODES:
1 !
1 !       Success or error signal
1 !
1 ! SIDE EFFECTS:
1 !
1 !       NONE
1 !
1 !--
1
2     BEGIN
2
2     MAP
2       PARAM : REF VECTOR [2]          ! Address of UPLIT (low, high)
2       ;
2
2     IF  .NUM GEQU .PARAM [0] AND          ! If inbounds then store
2       .NUM LEQU .PARAM [1]
2     THEN
3       BEGIN                           ! Store the value
3       IF .ACT$GA_RNGLST [0]  LSS ACT$C_RNGLSTMAX
3       THEN
4           BEGIN
4           ACT$GA_RNGLST [0]  = .ACT$GA_RNGLST [0]  + 1;
4           ACT$GA_RNGLST [.ACT$GA_RNGLST [0]]  = .NUM;
4           END
3       ELSE
4           BEGIN
4           !
4           !   If the vector is full then complain
4           !
4           NCP$SIG_CMDERR                  ! Signal too many ranges
```

```
;  755          0744  4                  (NCP$_FIELDLIM,
;  756          0745  4                  .TKNCNT, .TKNPTR,
;  757          0746  4                  .STRCNT, .STRPTR
;  758          0747  4                  );
;  759          0748  4              RETURN FAILURE;                        ! Fail transition in parse table
;  760          0749  3              END;
;  761          0750  3          END
;  762          0751  2      ELSE
;  763          0752  3          BEGIN
;  764          0753  3          NCP$SIG_CMDERR                      ! Signal parameter out of range
;  765          0754  3              (NCP$_PRMRNG,
;  766          0755  3              .TKNCNT, .TKNPTR,
;  767          0756  3              .STRCNT, .STRPTR
;  768          0757  3              )
;  769          0758  3          ;
;  770          0759  3          RETURN FAILURE;                      ! Fail transition in parse table
;  771          0760  2          END;
;  772          0761  2
;  773          0762  2      RETURN SUCCESS;
;  774          0763  1      END;


                                           0004 00000        .ENTRY  ACT$NUM_RNGSAV, Save R2                       ; 0688
                 52 00000000'   00    9E 00002        MOVAB   ACT$GA_RNGLST, R2                              ;
                 50         20   AC    D0 00009        MOVL    PARAM, R0                                      ; 0728
                 60         1C   AC    D1 0000D        CMPL    NUM, (R0)                                      ;
                                 28    1F 00011        BLSSU   2$                                             ;
          04     A0         1C   AC    D1 00013        CMPL    NUM, 4(R0)                                     ; 0729
                                 21    1A 00018        BGTRU   2$                                             ;
                 20              62    B1 0001A        CMPW    ACT$GA_RNGLST, #32                             ; 0732
                                 0C    1E 0001D        BGEQU   1$                                             ;
                                 62    B6 0001F        INCW    ACT$GA_RNGLST                                  ; 0735
                 50              62    3C 00021        MOVZWL  ACT$GA_RNGLST, R0                              ; 0736
               6240         1C   AC    B0 00024        MOVW    NUM, ACT$GA_RNGLST[R0]                         ;
                                 25    11 00029        BRB     4$                                             ; 0732
                 7E         08   AC    7D 0002B 1$:    MOVQ    STRCNT, -(SP)                                  ; 0746
                 7E         10   AC    7D 0002F        MOVQ    TKNCNT, -(SP)                                  ; 0745
                    00000000G    8F    DD 00033        PUSHL   #NCP$_FIELDLIM                                 ; 0744
                                 0E    11 00039        BRB     3$                                             ;
                 7E         08   AC    7D 0003B 2$:    MOVQ    STRCNT, -(SP)                                  ; 0756
                 7E         10   AC    7D 0003F        MOVQ    TKNCNT, -(SP)                                  ; 0755
                    00000000G    8F    DD 00043        PUSHL   #NCP$_PRMRNG                                   ; 0754
          FE27   CF              05    FB 00049 3$:    CALLS   #5, NCP$SIG_CMDERR                             ;
                                 04    11 0004E        BRB     5$                                             ; 0759
                 50              01    D0 00050 4$:    MOVL    #1, R0                                         ; 0762
                                       04 00053        RET                                                    ;
                           50    D4 00054 5$:    CLRL    R0                                             ; 0763
                                       04 00056        RET                                                    ;

; Routine Size:  87 bytes,    Routine Base:  $CODE$ + 021B
```

```
1 %SBTTL  'ACT$NUM_SAV  Store value in vector'
1 GLOBAL ROUTINE ACT$NUM_SAV (OPT, STRCNT, STRPTR, TKNCNT, TKNPTR,
1                       CHR, NUM) =     !
1
1 !++
1 ! FUNCTIONAL DESCRIPTION:
1 !
1 !       Action routine to store value in vector.
1 !
1 ! FORMAL PARAMETERS:
1 !
1 !       Parse state table
1 !       NUM             Value of the numeric token
1 !
1 ! IMPLICIT INPUTS:
1 !
1 !       NONE
1 !
1 ! IMPLICIT OUTPUTS:
1 !
1 !       NONE
1 !
1 ! ROUTINE VALUE:
1 ! COMPLETION CODES:
1 !
1 !       Success or error signal
1 !
1 ! SIDE EFFECTS:
1 !
1 !       NONE
1 !
1 !--
1
2     BEGIN
2     IF .ACT$GA_RNGLST [0] LSS ACT$C_RNGLSTMAX
2     THEN
3         BEGIN                           ! Store the value
3         ACT$GA_RNGLST [0] = .ACT$GA_RNGLST [0] + 1;
3         ACT$GA_RNGLST [.ACT$GA_RNGLST [0] ]= .NUM;
3         END
2     ELSE
3         BEGIN
3         !
3         !   If the vector is full then complain
3         !
3         NCP$SIG_CMDERR                  ! Signal too many ranges
3             (NCP$_FIELDLIM,
3             .TKNCNT, .TKNPTR,
3             .STRCNT, .STRPTR
3             );
3         RETURN FAILURE;                 ! Fail transition in parse table
2         END;
2
2     RETURN SUCCESS;
1     END;
```

```
                                       0004 00000          .ENTRY  ACT$NUM_SAV, Save R2                        ; 0765
                  52 00000000'    00   9E 00002          MOVAB   ACT$GA_RNGLST, R2                           ;
                  20              62   B1 00009          CMPW    ACT$GA_RNGLST, #32                          ; 0798
                                  0C   1E 0000C          BGEQU   1$                                          ;
                                  62   B6 0000E          INCW    ACT$GA_RNGLST                               ; 0801
                  50              62   3C 00010          MOVZWL  ACT$GA_RNGLST, R0                           ; 0802
                6240        1C    AC   B0 00013          MOVW    NUM, ACT$GA_RNGLST[R0]                      ;
                                  15   11 00018          BRB     2$                                          ; 0798
                  7E        08    AC   7D 0001A 1$:      MOVQ    STRCNT, -(SP)                               ; 0812
                  7E        10    AC   7D 0001E          MOVQ    TKNCNT, -(SP)                               ; 0811
                     00000000G    8F   DD 00022          PUSHL   #NCP$_FIELDLIM                              ; 0810
            FDF1    CF            05   FB 00028          CALLS   #5, NCP$SIG_CMDERR                          ;
                                  04   11 0002D          BRB     3$                                          ; 0814
                  50              01   D0 0002F 2$:      MOVL    #1, R0                                      ; 0817
                                       04 00032          RET                                                 ;
                                  50   D4 00033 3$:      CLRL    R0                                          ; 0818
                                       04 00035          RET                                                 ;

; Routine Size:  54 bytes,    Routine Base:  $CODE$ + 0272
```

```
%SBTTL  'ACT$STR_LEN  Check string length'
GLOBAL ROUTINE ACT$STR_LEN (OPT, STRCNT, STRPTR, TKNCNT, TKNPTR,
                CHR, NUM, PARAM) =      !

!++
! FUNCTIONAL DESCRIPTION:
!
!       Action routine to check length of a string token.
!       Length is checked for strings with trailing spaces and
!       quoted strings.  Trailing spaces and tabs are removed and
!       quoted strings which must begin with a " are counted with
!       the quotes parsed correctly.
!
! FORMAL PARAMETERS:
!
!       Parse state table
!       TKNCNT                  Length of the string token
!       PARAM                   Value of the maximum length of token
!
! IMPLICIT INPUTS:
!
!       NONE
!
! IMPLICIT OUTPUTS:
!
!       NONE
!
! ROUTINE VALUE:
! COMPLETION CODES:
!
!       Success or error signal
!
! SIDE EFFECTS:
!
!       NONE
!
!--

    BEGIN

    LOCAL
        PTR,                                    ! Point into token
        PEND,                                   ! End of token
        CTR                                     ! Size of string
        ;

    IF CH$RCHAR (.TKNPTR) EQL '"'               ! Quoted string?
    THEN
        BEGIN
        CTR = 0;                                ! Setup counters and pointers
        PTR = .TKNPTR + 1;
        PEND = .TKNPTR + .TKNCNT;
        WHILE .PTR LSS .PEND                    ! Scan string
        DO
            BEGIN
            IF  CH$RCHAR_A (PTR) EQL '"' ! Quote inside?
            THEN
```

```
;  889          0876  5                     BEGIN
;  890          0877  5                     IF   CH$RCHAR (.PTR) EQL '"'
;  891          0878  5                          AND
;  892          0879  5                          .PTR LSS .PEND
;  893          0880  5                     THEN PTR = .PTR + 1         ! Count one quote for two
;  894          0881  5                     ELSE EXITLOOP               ! Single quote ends it
;  895          0882  5                     END
;  896          0883  4                 ;
;  897          0884  4                 CTR = .CTR + 1
;  898          0885  4                 END
;  899          0886  3             END
;  900          0887  2         ELSE
;  901          0888  3             BEGIN
;  902          0889  3             PTR = .TKNPTR + .TKNCNT - 1;      ! Strip trailing spaces from token
;  903          0890  3             WHILE PTR GTRU .TKNPTR
;  904          0891  3                 AND
;  905          0892  4                 (
;  906          0893  4                 CH$RCHAR (.PTR) EQL ' '       ! Space
;  907          0894  4                 OR
;  908          0895  4                 CH$RCHAR (.PTR) EQL 9         ! Tab
;  909          0896  4                 )
;  910          0897  3             DO
;  911          0898  3                 PTR = .PTR - 1
;  912          0899  3             ;
;  913          0900  3             CTR = (.PTR + 1) - .TKNPTR        ! Compute real size of token
;  914          0901  3             END
;  915          0902  2         ;
;  916          0903  2
;  917          0904  2         IF .CTR LEQU .PARAM                   ! Check size of token
;  918          0905  2         THEN
;  919          0906  2             RETURN SUCCESS
;  920          0907  2         ELSE
;  921          0908  3             BEGIN
;  922          0909  3             NCP$SIG_CMDERR                    ! Signal to print error message
;  923          0910  3                 (NCP$_PRMLEN,
;  924          0911  3                 .TKNCNT, .TKNPTR,
;  925          0912  3                 .STRCNT, .STRPTR
;  926          0913  3                 )
;  927          0914  3             ;
;  928          0915  3             RETURN FAILURE                    ! Fail transition in parse table
;  929          0916  3             END
;  930          0917  3
;  931          0918  1         END;


                                       000C 00000          .ENTRY  ACT$STR_LEN, Save R2,R3                        ; 0820
                         5E              04  C2 00002          SUBL2   #4, SP
                         52       14     AC  D0 00005          MOVL    TKNPTR, R2                                   ; 0865
           51            52       10     AC  C1 00009          ADDL3   TKNCNT, R2, R1                               ; 0870
                         22              62  91 0000E          CMPB    (R2), #34                                    ; 0865
                                         27  12 00011          BNEQ    3$
                                         50  D4 00013          CLRL    CTR                                          ; 0868
                         6E       01     A2  9E 00015          MOVAB   1(R2), PTR                                   ; 0869
                         51              6E  D1 00019 1$:      CMPL    PTR, PEND                                    ; 0871
```

```
                                        40  18 0001C            BGEQ    7$                                              ;
                        53        00    BE  9A 0001E            MOVZBL  @PTR, R3                                        ; 0874
                                        6E  D6 00022            INCL    PTR                                             ;
                        22              53  91 00024            CMPB    R3, #34                                         ;
                                        0D  12 00027            BNEQ    2$                                              ;
                        22        00    BE  91 00029            CMPB    @PTR, #34                                       ; 0877
                                        2F  12 0002D            BNEQ    7$                                              ;
                        51              6E  D1 0002F            CMPL    PTR, PEND                                       ; 0879
                                        2A  18 00032            BGEQ    7$                                              ;
                                        6E  D6 00034            INCL    PTR                                             ; 0880
                                        50  D6 00036 2$:        INCL    CTR                                             ; 0884
                                        DF  11 00038            BRB     1$                                              ;
                        6E        FF    A1  9E 0003A 3$:        MOVAB   -1(R1), PTR                                     ; 0889
                        51              6E  9E 0003E 4$:        MOVAB   PTR, R1                                         ; 0890
                        51              52  D1 00041            CMPL    R2, R1                                          ;
                                        10  1E 00044            BGEQU   6$                                              ;
                        20        00    BE  91 00046            CMPB    @PTR, #32                                       ; 0893
                                        06  13 0004A            BEQL    5$                                              ;
                        09        00    BE  91 0004C            CMPB    @PTR, #9                                        ; 0895
                                        04  12 00050            BNEQ    6$                                              ;
                                        6E  D7 00052 5$:        DECL    PTR                                             ; 0898
                                        E8  11 00054            BRB     4$                                              ;
            51                6E        52  C3 00056 6$:        SUBL3   R2, PTR, R1                                     ; 0900
                        50        01    A1  9E 0005A            MOVAB   1(R1), CTR                                      ;
                  20    AC              50  D1 0005E 7$:        CMPL    CTR, PARAM                                      ; 0904
                                        04  1A 00062            BGTRU   8$                                              ;
                        50              01  D0 00064            MOVL    #1, R0                                          ; 0908
                                            04 00067            RET                                                     ;
                        7E        08    AC  7D 00068 8$:        MOVQ    STRCNT, -(SP)                                   ; 0912
                                        52  DD 0006C            PUSHL   R2                                              ; 0911
                                  10    AC  DD 0006E            PUSHL   TKNCNT                                          ;
                          00000000G     8F  DD 00071            PUSHL   #NCP$_PRMLEN                                    ; 0910
                FD6C    CF              05  FB 00077            CALLS   #5, NCP$SIG_CMDERR                              ;
                                        50  D4 0007C            CLRL    R0                                              ; 0915
                                            04 0007E            RET                                                     ; 0918

; Routine Size:  127 bytes,    Routine Base:  $CODE$ + 02A8
;  933      0919  1 %SBTTL  'ACT$NXT_STATE  Set next state table for parse'
;  934      0920  1 GLOBAL ROUTINE ACT$NXT_STATE (OPT, STRCNT, STRPTR, TKNCNT, TKNPTR,
;  935      0921  1                         CHR, NUM, PARAM) =        !
;  936      0922  1
;  937      0923  1 !++
;  938      0924  1 ! FUNCTIONAL DESCRIPTION:
;  939      0925  1 !
;  940      0926  1 !       Setup pointers to skip to another state table to finish parse
;  941      0927  1 !
;  942      0928  1 ! FORMAL PARAMETERS:
;  943      0929  1 !
;  944      0930  1 !       Parse state table
;  945      0931  1 !       PARAM           Address of address pair - state_table, key_table
;  946      0932  1 !
;  947      0933  1 ! IMPLICIT INPUTS:
;  948      0934  1 !
;  949      0935  1 !       NONE
;  950      0936  1 !
;  951      0937  1 ! IMPLICIT OUTPUTS:
;  952      0938  1 !
;  953      0939  1 !       NONE
;  954      0940  1 !
;  955      0941  1 ! ROUTINE VALUE:
;  956      0942  1 ! COMPLETION CODES:
;  957      0943  1 !
;  958      0944  1 !       Success
;  959      0945  1 !
;  960      0946  1 ! SIDE EFFECTS:
;  961      0947  1 !
;  962      0948  1 !       NONE
;  963      0949  1 !
;  964      0950  1 !--
;  965      0951  1
;  966      0952  2     BEGIN
;  967      0953  2
;  968      0954  2     MAP
;  969      0955  2         PARAM : REF VECTOR
;  970      0956  2         ;               !
;  971      0957  2
;  972      0958  2     NCP$_NXT_STATE [0] = .PARAM [0];      ! State table address
;  973      0959  2     NCP$_NXT_STATE [1] = .PARAM [1];      ! Key table address
;  974      0960  2     RETURN SUCCESS
;  975      0961  2
;  976      0962  1     END;



                                              0000 00000          .ENTRY  ACT$NXT_STATE, Save nothing                   ; 0920
                          50         20   AC   D0 00002          MOVL    PARAM, R0                                     ; 0958
              00000000'   00              60   7D 00006          MOVQ    (R0), NCP$_NXT_STATE
                          50              01   D0 0000D          MOVL    #1, R0                                        ; 0960
                                               04 00010          RET                                                   ; 0962

; Routine Size:  17 bytes,    Routine Base:  $CODE$ + 0327
```

```
;  978          0963  1 %SBTTL  'ACT$WRI_STR  Write a string to the output device'
;  979          0964  1 GLOBAL ROUTINE ACT$WRI_STR (OPT, STRCNT, STRPTR, TKNCNT, TKNPTR,
;  980          0965  1                 CHR, NUM, PARAM) =      !
;  981          0966  1
;  982          0967  1 !++
;  983          0968  1 ! FUNCTIONAL DESCRIPTION:
;  984          0969  1 !
;  985          0970  1 !       Write a string to SYS$OUTPUT
;  986          0971  1 !
;  987          0972  1 ! FORMAL PARAMETERS:
;  988          0973  1 !
;  989          0974  1 !       Parse state table
;  990          0975  1 !       PARAM           Address of descriptor of the string
;  991          0976  1 !
;  992          0977  1 ! IMPLICIT INPUTS:
;  993          0978  1 !
;  994          0979  1 !       NONE
;  995          0980  1 !
;  996          0981  1 ! IMPLICIT OUTPUTS:
;  997          0982  1 !
;  998          0983  1 !       NONE
;  999          0984  1 !
; 1000          0985  1 ! ROUTINE VALUE:
; 1001          0986  1 ! COMPLETION CODES:
; 1002          0987  1 !
; 1003          0988  1 !       Success
; 1004          0989  1 !
; 1005          0990  1 ! SIDE EFFECTS:
; 1006          0991  1 !
; 1007          0992  1 !       NONE
; 1008          0993  1 !
; 1009          0994  1 !--
; 1010          0995  1
; 1011          0996  2     BEGIN
; 1012          0997  2
; 1013          0998  2
; 1014          0999  2     NCP$WRITE_LINE (.PARAM);                ! Write the line
; 1015          1000  2     RETURN SUCCESS
; 1016          1001  2
; 1017          1002  1     END;



                                     0000 00000        .ENTRY  ACT$WRI_STR, Save nothing                          ; 0964
                              20  AC  DD 00002         PUSHL   PARAM                                              ; 0999
              00000000G  00       01  FB 00005         CALLS   #1, NCP$WRITE_LINE                                 ;
                         50       01  D0 0000C         MOVL    #1, R0                                             ; 1000
                                  04 0000F             RET                                                        ; 1002

; Routine Size:  16 bytes,     Routine Base:  $CODE$ + 0338
; 1019       1003   1 %SBTTL  'ACT$SIGNAL  Signal and error from parse'
; 1020       1004   1 GLOBAL ROUTINE ACT$SIGNAL (OPT, STRCNT, STRPTR, TKNCNT, TKNPTR,
; 1021       1005   1                    CHR, NUM, PARAM) =        !
; 1022       1006   1
; 1023       1007   1 !++
; 1024       1008   1 ! FUNCTIONAL DESCRIPTION:
; 1025       1009   1 !
; 1026       1010   1 !       Signal and error from parse and return to parse
; 1027       1011   1 !
; 1028       1012   1 ! FORMAL PARAMETERS:
; 1029       1013   1 !
; 1030       1014   1 !       Parse state table
; 1031       1015   1 !       PARAM           Value of status code to signal
; 1032       1016   1 !
; 1033       1017   1 ! IMPLICIT INPUTS:
; 1034       1018   1 !
; 1035       1019   1 !       NONE
; 1036       1020   1 !
; 1037       1021   1 ! IMPLICIT OUTPUTS:
; 1038       1022   1 !
; 1039       1023   1 !       NONE
; 1040       1024   1 !
; 1041       1025   1 ! ROUTINE VALUE:
; 1042       1026   1 ! COMPLETION CODES:
; 1043       1027   1 !
; 1044       1028   1 !       Success
; 1045       1029   1 !
; 1046       1030   1 ! SIDE EFFECTS:
; 1047       1031   1 !
; 1048       1032   1 !       NONE
; 1049       1033   1 !
; 1050       1034   1 !--
; 1051       1035   1
; 1052       1036   2     BEGIN
; 1053       1037   2
; 1054       1038   2
; 1055       1039   2     SIGNAL (.PARAM);             ! Signal the condition to print the message
; 1056       1040   2
; 1057       1041   2     RETURN SUCCESS
; 1058       1042   2
; 1059       1043   1     END;



                                    0000 00000          .ENTRY  ACT$SIGNAL, Save nothing                                 ; 1004
                         20      AC   DD 00002          PUSHL   PARAM                                                    ; 1039
          00000000G  00          01   FB 00005          CALLS   #1, LIB$SIGNAL                                           ;
                     50          01   D0 0000C          MOVL    #1, R0                                                   ; 1041
                                      04 0000F          RET                                                              ; 1043

; Routine Size:  16 bytes,    Routine Base:  $CODE$ + 0348
; 1061       1044  1 %SBTTL  'ACT$PMT_ON  Enable prompting'
; 1062       1045  1 GLOBAL ROUTINE ACT$PMT_ON =        !
; 1063       1046  1
; 1064       1047  1 !++
; 1065       1048  1 ! FUNCTIONAL DESCRIPTION:
; 1066       1049  1 !
; 1067       1050  1 !       Action routine to enable prompting for command prompting
; 1068       1051  1 !
; 1069       1052  1 ! FORMAL PARAMETERS:
; 1070       1053  1 !
; 1071       1054  1 !       NONE
; 1072       1055  1 !
; 1073       1056  1 ! IMPLICIT INPUTS:
; 1074       1057  1 !
; 1075       1058  1 !       NONE
; 1076       1059  1 !
; 1077       1060  1 ! IMPLICIT OUTPUTS:
; 1078       1061  1 !
; 1079       1062  1 !       ACT$GL_PMT_Q
; 1080       1063  1 !
; 1081       1064  1 ! ROUTINE VALUE:
; 1082       1065  1 ! COMPLETION CODES:
; 1083       1066  1 !
; 1084       1067  1 !       Success
; 1085       1068  1 !
; 1086       1069  1 ! SIDE EFFECTS:
; 1087       1070  1 !
; 1088       1071  1 !       NONE
; 1089       1072  1 !
; 1090       1073  1 !--
; 1091       1074  1
; 1092       1075  2     BEGIN
; 1093       1076  2
; 1094       1077  2
; 1095       1078  2     GLOBAL
; 1096       1079  2         ACT$GL_PMT_Q                        ! True for prompting enabled
; 1097       1080  2         ;
; 1098       1081  2
; 1099       1082  2     ACT$GL_PMT_Q = TRUE;                    ! Enable prompting
; 1100       1083  2
; 1101       1084  2     RETURN SUCCESS                          ! Continue the parse
; 1102       1085  2
; 1103       1086  1     END;


                                                              .PSECT  $GLOBAL$,NOEXE,2

                                            00054 ACT$GL_PMT_Q::
                                                              .BLKB   4


                                                              .PSECT  $CODE$,NOWRT,2

                                       0000 00000             .ENTRY  ACT$PMT_ON, Save nothing                     ; 1045
        00000000'  00               01  D0 00002              MOVL    #1, ACT$GL_PMT_Q                             ; 1082
```

```
                           50             01  D0 00009            MOVL     #1, R0                                    ; 1084
                                              04 0000C            RET                                                ; 1086

; Routine Size:  13 bytes,    Routine Base:  $CODE$ + 0358
```

```
1087  1 %SBTTL  'ACT$PMT_OFF  Disable prompting'
1088  1 GLOBAL ROUTINE ACT$PMT_OFF =     !
1089  1
1090  1 !++
1091  1 ! FUNCTIONAL DESCRIPTION:
1092  1 !
1093  1 !       Action routine to disable command prompting
1094  1 !
1095  1 ! FORMAL PARAMETERS:
1096  1 !
1097  1 !       NONE
1098  1 !
1099  1 ! IMPLICIT INPUTS:
1100  1 !
1101  1 !       NONE
1102  1 !
1103  1 ! IMPLICIT OUTPUTS:
1104  1 !
1105  1 !       ACT$GL_PMT_Q
1106  1 !
1107  1 ! ROUTINE VALUE:
1108  1 ! COMPLETION CODES:
1109  1 !
1110  1 !       Success
1111  1 !
1112  1 ! SIDE EFFECTS:
1113  1 !
1114  1 !       NONE
1115  1 !
1116  1 !--
1117  1
1118  2     BEGIN
1119  2
1120  2     ACT$GL_PMT_Q = FALSE;                   ! Disable prompting
1121  2
1122  2     RETURN SUCCESS                          ! Continue the parse
1123  2
1124  1     END;



                                0000 00000          .ENTRY  ACT$PMT_OFF, Save nothing                         : 1088
                    00000000G  00  D4 00002          CLRL    ACT$GL_PMT_Q                                      : 1120
              50              01  D0 00008          MOVL    #1, R0                                            : 1122
                                  04 0000B          RET                                                       : 1124

; Routine Size:  12 bytes,    Routine Base:  $CODE$ + 0365
```

```
1 %SBTTL  'ACT$PMT_Q  Control command prompting'
1 GLOBAL ROUTINE ACT$PMT_Q (OPT, STRCNT, STRPTR, TKNCNT, TKNPTR,
1                   CHR, NUM, PARAM) =      !
1
1 !++
1 ! FUNCTIONAL DESCRIPTION:
1 !
1 !       Action routine to control command prompting
1 !       These three routines work with the COMMAND_PROMPT
1 !       macro to control command prompting.  Keywords or
1 !       an entity is prompted if the state is entered with
1 !       the TPA$_EOS condition.  ACT$PMT_ON and _OFF are used
1 !       to set prompting on or off.  If prompting is on and
1 !       any other transition fails, this action routine is called
1 !       to signal an error and set the EOS condition so ACT$PRMPT
1 !       will obtain the next string to try.  The parse loops in the
1 !       state until an acceptable response is given or EOF causes
1 !       return to the initial command level.
1 !
1 ! FORMAL PARAMETERS:
1 !
1 !       Parse state table
1 !       PARAM           Value of status code to signal if non-zero
1 !
1 ! IMPLICIT INPUTS:
1 !
1 !       ACT$GL_PMT_Q
1 !
1 ! IMPLICIT OUTPUTS:
1 !
1 !       NONE
1 !
1 ! ROUTINE VALUE:
1 ! COMPLETION CODES:
1 !
1 !       Success of prompting, failure if not promting
1 !
1 ! SIDE EFFECTS:
1 !
1 !       NONE
1 !
1 !--
1
2     BEGIN
2
2     IF .ACT$GL_PMT_Q                          ! If prompting
2     THEN
3         BEGIN
3         IF .PARAM NEQ 0                       ! If condition to signal
3         THEN  SIGNAL (.PARAM)                 ! Signal the condition
3         ;
3         STRCNT = 0;                           ! Set for EOS parse to occur
3         RETURN SUCCESS                        ! Continue parse
3         END
3
2     ELSE
2         RETURN FAILURE                        ! Cause failure in state
```

```
; 1201          1182  2
; 1202          1183  1      END;



                                            0000 00000                 .ENTRY   ACT$PMT_Q, Save nothing                           ; 1126
                        16 00000000G    00  E9 00002                   BLBC     ACT$GL_PMT_Q, 2$                                  ; 1170
                                20      AC  D5 00009                   TSTL     PARAM                                             ; 1173
                                        0A  13 0000C                   BEQL     1$                                                ;
                                20      AC  DD 0000E                   PUSHL    PARAM                                             ; 1174
                00000000G   00          01  FB 00011                   CALLS    #1, LIB$SIGNAL                                    ;
                                08      AC  D4 00018  1$:              CLRL     STRCNT                                            ; 1176
                            50          01  D0 0001B                   MOVL     #1, R0                                            ; 1181
                                            04 0001E                   RET                                                        ;
                                50          D4 0001F  2$:              CLRL     R0                                                ;
                                            04 00021                   RET                                                        ; 1183

; Routine Size:  34 bytes,    Routine Base:  $CODE$ + 0371
; 1204         1184  1 %SBTTL  'ACT$EXECQ  Test if Component is Executor'
; 1205         1185  1 GLOBAL ROUTINE ACT$EXECQ (OPT, STRCNT, STRPTR, TKNCNT, TKNPTR,
; 1206         1186  1                 CHR, NUM, PDB) =            !
; 1207         1187  1
; 1208         1188  1 !++
; 1209         1189  1 ! FUNCTIONAL DESCRIPTION:
; 1210         1190  1 !
; 1211         1191  1 !       Action routine to test if the current node component is the
; 1212         1192  1 !       executor node.  Called from the ncpstanod module to select
; 1213         1193  1 !       the correct parameters for prompting.
; 1214         1194  1 !       The executor is coded as an address of zero, that is three
; 1215         1195  1 !       data bytes of zero.
; 1216         1196  1 !
; 1217         1197  1 ! FORMAL PARAMETERS:
; 1218         1198  1 !
; 1219         1199  1 !       Parse state table
; 1220         1200  1 !       PDB             Address of PDB data block for the component.
; 1221         1201  1 !
; 1222         1202  1 ! IMPLICIT INPUTS:
; 1223         1203  1 !
; 1224         1204  1 !       NONE
; 1225         1205  1 !
; 1226         1206  1 ! IMPLICIT OUTPUTS:
; 1227         1207  1 !
; 1228         1208  1 !       NONE
; 1229         1209  1 !
; 1230         1210  1 ! ROUTINE VALUE:
; 1231         1211  1 ! COMPLETION CODES:
; 1232         1212  1 !
; 1233         1213  1 !       Success if component is executor node, failure if not
; 1234         1214  1 !
; 1235         1215  1 ! SIDE EFFECTS:
; 1236         1216  1 !
; 1237         1217  1 !       NONE
; 1238         1218  1 !
; 1239         1219  1 !--
; 1240         1220  1
; 1241         1221  2    BEGIN
; 1242         1222  2
; 1243         1223  2    MAP
; 1244         1224  2       PDB : REF BBLOCK [PDB$C_SIZE]   ! Pointer to the component PDB
; 1245         1225  2       ;
; 1246         1226  2
; 1247         1227  2    IF .PDB [1, 0, 24, 0] EQL 0          ! Look at three bytes of the data
; 1248         1228  2    THEN
; 1249         1229  2       RETURN SUCCESS                    ! It is the executor
; 1250         1230  2
; 1251         1231  2    ELSE
; 1252         1232  2       RETURN FAILURE                    ! Cause failure in state
; 1253         1233  2
; 1254         1234  1    END;



                                          0000 00000          .ENTRY  ACT$EXECQ, Save nothing                      ; 1185
```

```
                                   50      20   AC  D0 00002         MOVL    PDB, R0                               ; 1227
        00       01   A0           18           00  ED 00006         CMPZV   #0, #24, 1(R0), #0                    ;
                                                04  12 0000C         BNEQ    1$                                    ;
                                   50           01  D0 0000E         MOVL    #1, R0                                ; 1232
                                                    04 00011         RET                                           ;
                                           50   D4 00012 1$:         CLRL    R0                                    ;
                                                    04 00014         RET                                           ; 1234

; Routine Size:  21 bytes,    Routine Base:  $CODE$ + 0393
```

```
%SBTTL  'ACT$PMTDONEQ  Terminate Prompts?'
GLOBAL ROUTINE ACT$PMTDONEQ (OPT, STRCNT, STRPTR, TKNCNT, TKNPTR,
                CHR, NUM, PDB) =                 !

!++
! FUNCTIONAL DESCRIPTION:
!
!       This routine checks the parsed token to see if it is "_DONE".
!       If so the routine returns success and the parse tables exit the
!       prompt sequence.  Otherwise the routine returns false and the
!       remainder of the prompts are processed.
!
! FORMAL PARAMETERS:
!
!       Parse state
!       TKNCNT                          Descriptor of token just parsed
!       TKNPTR
!
! IMPLICIT INPUTS:
!
!       NONE
!
! IMPLICIT OUTPUTS:
!
!       NONE
!
! ROUTINE VALUE:
! COMPLETION CODES:
!
!       Success or failure
!
! SIDE EFFECTS:
!
!       NONE
!
!--

   BEGIN

   IF CH$EQL (.TKNCNT, .TKNPTR, 5, UPLIT BYTE ('_DONE'), 0)
   THEN
       RETURN SUCCESS
   ELSE
       RETURN FAILURE

   END;


                                                  .PSECT  $PLIT$,NOWRT,NOEXE,2

                     45  4E  4F  44  5F  00000 P.AAA:  .ASCII  \_DONE\                      ;


                                                  .PSECT  $CODE$,NOWRT,2
```

```
                                          000C 00000          .ENTRY  ACT$PMTDONEQ, Save R2,R3                        ; 1236
        05          00      14   BC    10,  AC  2D 00002          CMPC5   TKNCNT, @TKNPTR, #0, #5, P.AAA                  ; 1274
                              00000000'  00     00009                                                                  ;
                                         04  12 0000E          BNEQ    1$                                              ;
                                 50      01  D0 00010          MOVL    #1, R0                                          ; 1278
                                             04 00013          RET                                                     ;
                                         50  D4 00014 1$:      CLRL    R0                                              ;
                                             04 00016          RET                                                     ; 1280

; Routine Size:  23 bytes,    Routine Base:  $CODE$ + 03A8
```

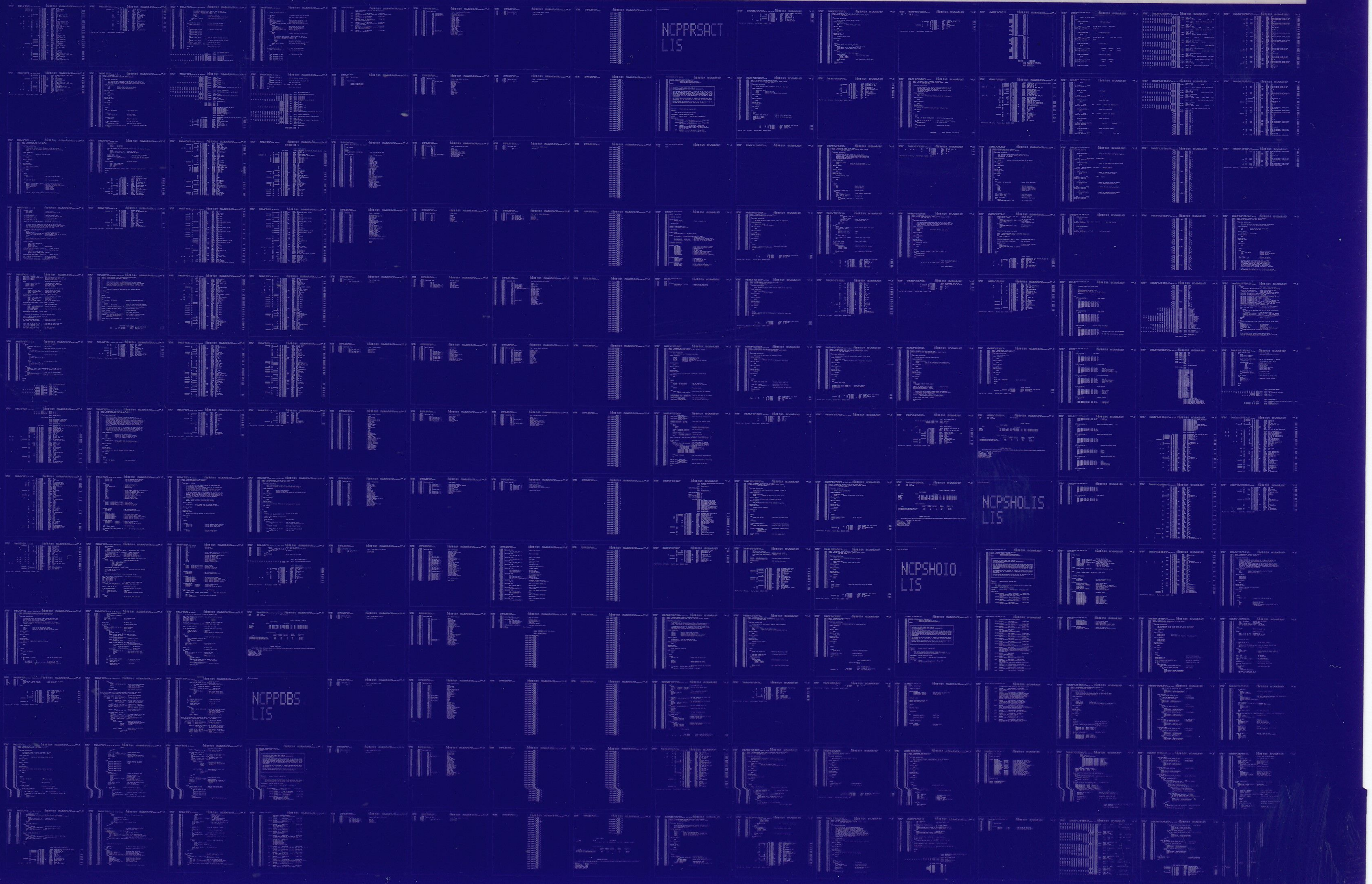
```
1
1  %SBTTL  'ACT$HELP  Provide prompting help'
1  GLOBAL ROUTINE ACT$HELP (OPT, STRCNT, STRPTR, TKNCNT, TKNPTR,
1                           CHR, NUM, PARAM) =          !
1
1  !++
1  ! FUNCTIONAL DESCRIPTION:
1  !
1  !       Action routine to provide prompting help
1  !
1  ! FORMAL PARAMETERS:
1  !
1  !       Parse state table
1  !       STRCNT          Size of the remainder of the command line
1  !       STRPTR          Address of the remainder of the command line
1  !
1  ! IMPLICIT INPUTS:
1  !
1  !       NONE
1  !
1  ! IMPLICIT OUTPUTS:
1  !
1  !       NONE
1  !
1  ! ROUTINE VALUE:
1  ! COMPLETION CODES:
1  !
1  !       SUCCESS
1  !
1  ! SIDE EFFECTS:
1  !
1  !       NONE
1  !
1  !--
1
2     BEGIN
2     LOCAL
2       HLP_DESC : BBLOCK [DSC$C_S_BLN],
2       STATUS;
2
2     CH$FILL (0, DSC$C_S_BLN, HLP_DESC);               ! zero descriptor
2     HLP_DESC [DSC$W_LENGTH] = .STRCNT;
2     HLP_DESC [DSC$A_POINTER] = .STRPTR;
2
2     !
2     !     Request help be printed by lib$put_output to sys$output,
2     !     from library SYS$HELP:NCPHELP.HLB.  Query for additional help
2     !     to sys$input using lib$get_input.
2     !
2     STATUS = LBR$OUTPUT_HELP (LIB$PUT_OUTPUT, 0, HLP_DESC,
2                 $DESCRIPTOR('NCPHELP'), 0,LIB$GET_INPUT);
2
2     IF NOT .STATUS THEN SIGNAL (.STATUS);
2
2     RETURN SUCCESS
1     END;
```

```
                                                                        .PSECT  $PLIT$,NOWRT,NOEXE,2

                        50  4C  45  48  50  43  4E  00005 P.AAC:    .ASCII  \NCPHELP\                              ;
                                         00000007  0000C P.AAB:    .LONG   7                                      ;
                                         00000000' 00010           .ADDRESS P.AAC                                 ;



                                                                        .PSECT  $CODE$,NOWRT,2

                                              003C 00000                .ENTRY  ACT$HELP, Save R2,R3,R4,R5        ; 1283
                               5E        08   C2 00002                SUBL2   #8, SP                             ;
        08             00      6E        00   2C 00005                MOVC5   #0, (SP), #0, #8, HLP_DESC         ; 1321
                                         6E      0000A                                                           ;
                               6E    08  AC   B0 0000B                MOVW    STRCNT, HLP_DESC                   ; 1322
                         04    AE    0C  AC   D0 0000F                MOVL    STRPTR, HLP_DESC+4                 ; 1323
                              00000000G  00   9F 00014                PUSHAB  LIB$GET_INPUT                      ; 1330
                                         7E   D4 0001A                CLRL    -(SP)                              ;
                              00000000'  00   9F 0001C                PUSHAB  P.AAB                              ; 1331
                                    0C   AE   9F 00022                PUSHAB  HLP_DESC                           ; 1330
                                         7E   D4 00025                CLRL    -(SP)                              ;
                              00000000G  00   9F 00027                PUSHAB  LIB$PUT_OUTPUT                     ;
                   00000000G  00         06   FB 0002D                CALLS   #6, LBR$OUTPUT_HELP                ;
                              09         50   E8 00034                BLBS    STATUS, 1$                         ; 1333
                                         50   DD 00037                PUSHL   STATUS                             ;
                   00000000G  00         01   FB 00039                CALLS   #1, LIB$SIGNAL                     ;
                              50         01   D0 00040 1$:            MOVL    #1, R0                             ; 1335
                                              04 00043                RET                                        ; 1336

; Routine Size:  68 bytes,    Routine Base:  $CODE$ + 03BF
; 1360          1337  1 END                                    !End of module
; 1361          1338  0 ELUDOM




                                                                                   .EXTRN  LIB$SIGNAL, LIB$STOP

;                                            PSECT SUMMARY
;
;       Name                            Bytes                           Attributes
;
;   $GLOBAL$                               88  NOVEC,  WRT,  RD ,NOEXE,NOSHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)
;   $OWN$                                  36  NOVEC,  WRT,  RD ,NOEXE,NOSHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)
;   $CODE$                               1027  NOVEC,NOWRT,  RD ,  EXE,NOSHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)
;   .  ABS  .                               0  NOVEC,NOWRT,NORD ,NOEXE,NOSHR,  LCL,  ABS,  CON,NOPIC,ALIGN(0)
;   $PLIT$                                 20  NOVEC,NOWRT,  RD ,NOEXE,NOSHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)




;                                     Library Statistics
;
;                                                 -------- Symbols --------        Pages        Processing
;       File                                      Total    Loaded   Percent        Mapped       Time
;
;   _$255$DUA28:[NCP.OBJ]NCPLIBRY.L32;1              373         7         1            52         00:00.1
;   _$255$DUA28:[SYSLIB]STARLET.L32;1               9776        18         0           581         00:01.4





;                                              COMMAND QUALIFIERS

;       BLISS/CHECK=(FIELD,INITIAL,OPTIMIZE)/LIS=LIS$:NCPPRSACT/OBJ=OBJ$:NCPPRSACT MSRC$:NCPPRSACT/UPDATE=(ENH$:NCPPRSACT)

; Size:           1027 code + 144 data bytes
; Run Time:          00:21.0
; Elapsed Time:      01:23.1
; Lines/CPU Min:       3819
; Lexemes/CPU-Min:    11009
; Memory Used:   87 pages
; Compilation Complete
```

**F

0268 AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY


NCPPRSACT
LIS

NCPSHOLIS
LIS

NCPSHOIO
LIS

NCPPDBS
LIS