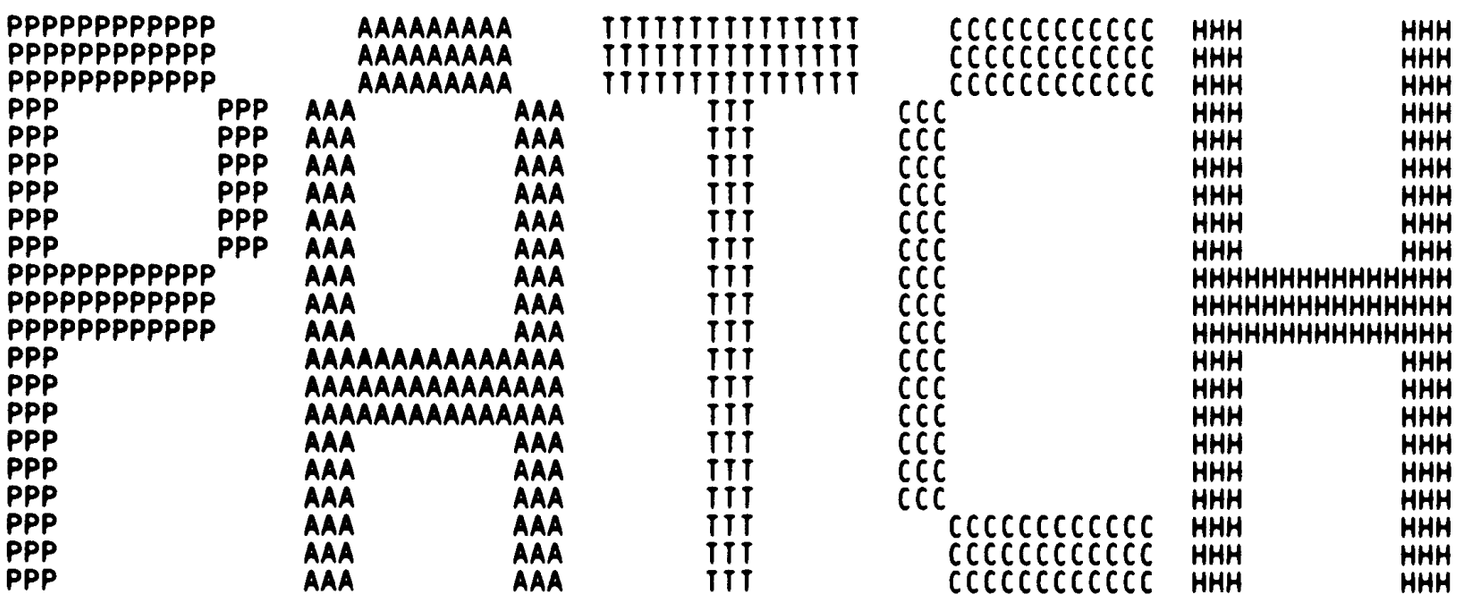
PPPPPPPPPPPP AAAAAAAAA TTTTTTTTTTTTTTT CCCCCCCCCCCC HHH HHH
PPP PPP AAA AAA TTT CCC HHH HHH
PPPPPPPPPPPP AAAAAAAAAAAAAAAA TTT CCC HHHHHHHHHHHHHHHH
PPP AAA AAA TTT CCC HHH HHH
PPP AAA AAA TTT CCCCCCCCCCCC HHH HHH

```
PPPPPPPP      AAAAAA    TTTTTTTTTT    CCCCCCCC  MM      MM  DDDDDDDD
PPPPPPPP      AAAAAA    TTTTTTTTTT    CCCCCCCC  MM      MM  DDDDDDDD
PP      PP   AA    AA       TT      CC          MMMM  MMMM  DD      DD
PP      PP   AA    AA       TT      CC          MMMM  MMMM  DD      DD
PP      PP   AA    AA       TT      CC          MM  MM  MM  DD      DD
PP      PP   AA    AA       TT      CC          MM  MM  MM  DD      DD
PPPPPPPP     AA    AA       TT      CC          MM      MM  DD      DD
PPPPPPPP     AA    AA       TT      CC          MM      MM  DD      DD
PP           AAAAAAAAAA     TT      CC          MM      MM  DD      DD
PP           AAAAAAAAAA     TT      CC          MM      MM  DD      DD
PP           AA    AA       TT      CC          MM      MM  DD      DD      ....
PP           AA    AA       TT      CC          MM      MM  DD      DD      ....
PP           AA    AA       TT        CCCCCCCC  MM      MM  DDDDDDDD        ....
PP           AA    AA       TT        CCCCCCCC  MM      MM  DDDDDDDD        ....

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
```

00

```
;     1          0001  0 MODULE PATCMD (
;     2        L 0002  0                 %IF %VARIANT EQL 1
;     3          0003  0                 %THEN
;     4          0004  0                         ADDRESSING_MODE (EXTERNAL = LONG_RELATIVE, NONEXTERNAL = LONG_RELATIVE),
;     5          0005  0                 %FI
;     6          0006  0                 IDENT = 'V04-000') =
;     7          0007  1 BEGIN
;     8          0008  1
;     9          0009  1 !*****************************************************************************
;    10          0010  1 !*                                                                           *
;    11          0011  1 !*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                                  *
;    12          0012  1 !*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.                    *
;    13          0013  1 !*  ALL RIGHTS RESERVED.                                                      *
;    14          0014  1 !*                                                                           *
;    15          0015  1 !*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED     *
;    16          0016  1 !*  ONLY IN  ACCORDANCE WITH  THE  TERMS  OF  SUCH  LICENSE  AND WITH THE    *
;    17          0017  1 !*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR  ANY  OTHER     *
;    18          0018  1 !*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY     *
;    19          0019  1 !*  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE IS  HEREBY     *
;    20          0020  1 !*  TRANSFERRED.                                                              *
;    21          0021  1 !*                                                                           *
;    22          0022  1 !*  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE WITHOUT NOTICE     *
;    23          0023  1 !*  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL EQUIPMENT     *
;    24          0024  1 !*  CORPORATION.                                                              *
;    25          0025  1 !*                                                                           *
;    26          0026  1 !*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR  RELIABILITY OF ITS     *
;    27          0027  1 !*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.                   *
;    28          0028  1 !*                                                                           *
;    29          0029  1 !*                                                                           *
;    30          0030  1 !*****************************************************************************
;    31          0031  1 !
;    32          0032  1 ! FACILITY:     PATCH
;    33          0033  1 !
;    34          0034  1 !++
;    35          0035  1 ! FUNCTIONAL DESCRIPTION:
;    36          0036  1 !
;    37          0037  1 !       PATCH COMMAND LINE HANDLER
;    38          0038  1 !
;    39          0039  1 !
;    40          0040  1 ! History:
;    41          0041  1 !       Author:
;    42          0042  1 !               Carol Peters, 05 Oct 1976: DBGEXC.B32 Version 01
;    43          0043  1 !
;    44          0044  1 !
;    45          0045  1 ! MODIFIED BY:
;    46          0046  1 !
;    47          0047  1 !       V03-001 MTR0007         Mike Rhodes             14-Jun-1982
;    48          0048  1 !               Use shared system messages.  Affected modules include:
;    49          0049  1 !               DYNMEM.B32, PATBAS.B32, PATCMD.B32, PATIHD.B32, PATINT.B32,
;    50          0050  1 !               PATIO.B32, PATMAI.B32, PATMSG.MSG, PATWRT.B32, and PATSPA.B32.
;    51          0051  1 !
;    52          0052  1 !               The shared messages are defined by DYNMEM.B32's invocation of
;    53          0053  1 !               SHRMSG.REQ and we simply link against these symbols.  They are
;    54          0054  1 !               declared as external literals below.
;    55          0055  1 !
;    56          0056  1 !       V02-011 MTR0001         Mike Rhodes             20-AUG-1981
;    57          0057  1 !               Modify module PAT$PROMPT_READ\GET_LINE: write comments
```

```
;    58          0058  1 !                to the Journal file instead of trashing them.
;    59          0059  1 !
;    60          0060  1 !        V02-010 PCG0001         Peter George             02-FEB-1981
;    61          0061  1 !                Add require statement for LIB$:PATDEF.REQ
;    62          0062  1 !
;    63          0063  1 ! MODIFICATIONS:
;    64          0064  1 !
;    65          0065  1 !  NO   DATE            PROGRAMMER          PURPOSE
;    66          0066  1 !  --   ----            ----------          -------
;    67          0067  1 !
;    68          0068  1 !  00   13-OCT-77       K.D. MORSE          CONVERT FOR PATCH
;    69          0069  1 !  01   15-DEC-77       K.D. MORSE          ADD JOURNAL OUTPUT,
;    70          0070  1 !                                           CHANGE PROMPT
;    71          0071  1 !  02   3-JAN-78        K.D. MORSE          ADD CHECK FOR EOF.
;    72          0072  1 !  03   5-JAN-78        K.D. MORSE          TURN OFF TYPE AHEAD AFTER
;    73          0073  1 !                                           FIRST READ.
;    74          0074  1 !  04   16-JAN-78       K.D. MORSE          MAKE PAT$PROMPT READ TO
;    75          0075  1 !                                           HANDLE COMMAND PROMPTING.
;    76          0076  1 !  05   23-JAN-78       K.D. MORSE          CHANGE PAT$CP_INP_STR TO
;    77          0077  1 !                                           PAT$CP_INP_DSCS, A BLOCK OF INPUT
;    78          0078  1 !                                           LINE DESCRIPTORS TO HANDLE
;    79          0079  1 !                                           PROMPTING AND MULTIPLE INPUT LINES.
;    80          0080  1 !  06   21-MAR-78       K.D. MORSE          ADD CHECK FOR <CR><LF> AND
;    81          0081  1 !                                           NO PATCH COMMAND.
;    82          0082  1 !  07   25-APR-78       K.D. MORSE          CONVERT TO NATIVE COMPILER.
;    83          0083  1 !  08   13-JUN-78       K.D. MORSE          ADD FAO COUNT TO SIGNALS.
;    84          0084  1 !  09   19-JUN-78       K.D. MORSE          ADD CHECK TO IGNORE COMMENTS.
;    85          0085  1 !
;    86          0086  1 !--
```

```
;    88          0087  1 FORWARD ROUTINE
;    89          0088  1           PAT$COM_PROC : NOVALUE,                    ! Accepts a command
;    90          0089  1           PAT$PROMPT_READ;                           ! Prompts and reads a command
;    91          0090  1
;    92          0091  1 LIBRARY 'SYS$LIBRARY:LIB.L32';
;    93          0092  1 REQUIRE 'SRC$:VXSMAC.REQ';
;    94          0157  1 REQUIRE 'LIB$:PATDEF.REQ';                           ! Defines literals
;    95          0211  1 REQUIRE 'LIB$:PATMSG.REQ';
;    96          0385  1 REQUIRE 'SRC$:PATPCT.REQ';
;    97          0425  1 REQUIRE 'SRC$:PATGEN.REQ';
;    98          0647  1 REQUIRE 'SRC$:PATTER.REQ';
;    99          0854  1 REQUIRE 'SRC$:PATTOK.REQ';
;   100          0924  1 REQUIRE 'SRC$:SYSSER.REQ';
```

```
;  R0956  1     SWITCHES LIST (SOURCE);
;  R0957  1
;  R0958  1     EXTERNAL ROUTINE
;  R0959  1              PAT$fao_out;           ! formats a line and outputs to the terminal
;  R0960  1
```

```
;  101          1006  1
;  102          1007  1 EXTERNAL ROUTINE
;  103          1008  1         PAT$ERR_HANDLER,                          ! Exception handler
;  104          1009  1         PAT$FREERELEASE,                          ! Releases free storage
;  105          1010  1         PAT$FREEZ,                                ! Allocates free storage
;  106          1011  1         PAT$PARS_A_LINE,                          ! Parses a line of input text
;  107          1012  1         PAT$WRITEFILE,                            ! Writes the journal file
;  108          1013  1         GETFILDSC;                                ! Returns the address of a file name descrip
;  109          1014  1
;  110          1015  1 EXTERNAL
;  111          1016  1         PAT$CP_INP_DSCS : REF VECTOR [,LONG],     ! Points to vector of input string descripto
;  112          1017  1         PAT$GB_TAKE_CMD,                          ! Indicator to accept more commands
;  113          1018  1         PAT$GB_INPNAME,                           ! Input file name
;  114          1019  1         PAT$GL_ERRCODE,                           ! Global error code
;  115          1020  1         PAT$GL_INPFAB: BLOCK [, BYTE],            ! FAB for 'SYS$INPUT'
;  116          1021  1         PAT$GL_INPRAB: BLOCK [, BYTE],            ! RAB for 'SYS$INPUT'
;  117          1022  1         PAT$GL_JNLRAB: BLOCK [, BYTE];            ! RAB for journal file
;  118          1023  1
;  119          1024  1 EXTERNAL LITERAL
;  120          1025  1         PAT$K_INPNAMLNG,                          ! Input file name length
;  121          1026  1 !
;  122          1027  1 ! Define shared message references.  (resolved @ link time)
;  123          1028  1 !
;  124          1029  1         PAT$_CLOSEIN,                             ! Error closing input file.
;  125          1030  1         PAT$_CLOSEOUT,                            ! Error closing output file.
;  126          1031  1         PAT$_OPENIN,                              ! Error opening input file.
;  127          1032  1         PAT$_OPENOUT,                             ! Error opening output file.
;  128          1033  1         PAT$_READERR,                             ! Error reading from file.
;  129          1034  1         PAT$_SYSERROR,                            ! System Service error.
;  130          1035  1         PAT$_WRITEERR;                            ! Error writing to file.
;  131          1036  1
;  132          1037  1 LITERAL
;  133          1038  1         PATCH_PMT = 90;                           ! Patch command prompt offset
```

```
;  135     1039  1 GLOBAL ROUTINE PAT$COM_PROC : NOVALUE =
;  136     1040  1
;  137     1041  1 !++
;  138     1042  1 ! FUNCTIONAL DESCRIPTION:
;  139     1043  1 !
;  140     1044  1 !       Accepts a single command line from the user at patch command level.
;  141     1045  1 !       A command is read from SYS$INPUT.
;  142     1046  1 !
;  143     1047  1 ! CALLING SEQUENCE:
;  144     1048  1 !
;  145     1049  1 !       PAT$COM_PROC
;  146     1050  1 !
;  147     1051  1 ! INPUTS:
;  148     1052  1 !
;  149     1053  1 !       none
;  150     1054  1 !
;  151     1055  1 ! IMPLICIT INPUTS:
;  152     1056  1 !
;  153     1057  1 !       none
;  154     1058  1 !
;  155     1059  1 ! OUTPUTS:
;  156     1060  1 !
;  157     1061  1 !       none
;  158     1062  1 !
;  159     1063  1 ! IMPLICIT OUTPUTS:
;  160     1064  1 !
;  161     1065  1 !       none
;  162     1066  1 !
;  163     1067  1 ! ROUTINE VALUE:
;  164     1068  1 !
;  165     1069  1 !       NOVALUE
;  166     1070  1 !
;  167     1071  1 ! SIDE EFFECTS:
;  168     1072  1 !
;  169     1073  1 !       The parser is called with the contents of the input buffer.
;  170     1074  1 !
;  171     1075  1 !--
;  172     1076  1
;  173     1077  2 BEGIN
;  174     1078  2
;  175     1079  2 LOCAL
;  176     1080  2         STG_DESC: BLOCK [8, BYTE];                              ! String descriptor for parser
;  177     1081  2
;  178     1082  2 ENABLE PAT$ERR_HANDLER;
;  179     1083  2
;  180     1084  2 !++
;  181     1085  2 ! Request patch command.
;  182     1086  2 !--
;  183     1087  2 STG_DESC[DSC$W_LENGTH] = 0;
;  184     1088  3 WHILE (.STG_DESC[DSC$W_LENGTH] EQL 0)
;  185     1089  2 DO
;  186     1090  2         IF NOT PAT$PROMPT_READ (PATCH_PMT, STG_DESC)
;  187     1091  2         THEN
;  188     1092  2                 RETURN;
;  189     1093  2
;  190     1094  2 !++
;  191     1095  2 ! Parse the command and execute it.
```

```
;  192          1096  2 !--
;  193          1097  2 PAT$PARS_A_LINE (STG_DESC);
;  194          1098  2 RETURN;
;  195          1099  1 END;


                                                        .TITLE  PATCMD
                                                        .IDENT  \V04-000\

                                                        .EXTRN  PAT$FAO_OUT, PAT$ERR_HANDLER
                                                        .EXTRN  PAT$FREERELEASE
                                                        .EXTRN  PAT$FREEZ, PAT$PARS_A_LINE
                                                        .EXTRN  PAT$WRITEFILE, GETFILDSC
                                                        .EXTRN  PAT$CP_INP_DSCS
                                                        .EXTRN  PAT$GB_TAKE_CMD
                                                        .EXTRN  PAT$GB_INPNAME, PAT$GL_ERRCODE
                                                        .EXTRN  PAT$GL_INPFAB, PAT$GL_INPRAB
                                                        .EXTRN  PAT$GL_JNLRAB, PAT$K_INPNAMLNG
                                                        .EXTRN  PAT$_CLOSEIN, PAT$_CLOSEOUT
                                                        .EXTRN  PAT$_OPENIN, PAT$_OPENOUT
                                                        .EXTRN  PAT$_READERR, PAT$_SYSERROR
                                                        .EXTRN  PAT$_WRITEERR

                                                        .PSECT  _PAT$CODE,NOWRT,2

                                          0000 00000    .ENTRY  PAT$COM_PROC, Save nothing                       ; 1039
                                    5E    08   C2 00002  SUBL2  #8, SP                                           ;
                                    6D    0022 CF   DE 00005  MOVAL  3$, (FP)                                    ; 1077
                                          6E   B4 0000A  CLRW   STG_DESC                                        ; 1087
                                          6E   B5 0000C  1$:    TSTW   STG_DESC                                 ; 1088
                                          11   12 0000E  BNEQ   2$                                              ;
                                          5E   DD 00010  PUSHL  SP                                              ; 1090
                                    7E    5A   8F 9A 00012  MOVZBL #90, -(SP)                                    ;
                              00000000V EF         02 FB 00016  CALLS  #2, PAT$PROMPT_READ                      ;
                                    EC         50 E8 0001D  BLBS   R0, 1$                                      ;
                                               04 00020  RET                                                    ; 1092
                                          5E   DD 00021  2$:    PUSHL  SP                                       ; 1097
                              00000000G EF         01 FB 00023  CALLS  #1, PAT$PARS_A_LINE                      ;
                                               04 0002A  RET                                                    ; 1099
                                        0000 0002B  3$:    .WORD  Save nothing                                  ; 1077
                                          7E   D4 0002D  CLRL   -(SP)                                           ;
                                          5E   DD 0002F  PUSHL  SP                                              ;
                                    7E    04   AC 7D 00031  MOVQ   4(AP), -(SP)                                 ;
                              00000000G EF         03 FB 00035  CALLS  #3, PAT$ERR_HANDLER                      ;
                                               04 0003C  RET                                                    ;

; Routine Size:  61 bytes,    Routine Base:  _PAT$CODE + 0000
```

```
;  197          1100  1 GLOBAL ROUTINE PAT$PROMPT_READ(PROMPT_CODE, STG_DESC) =
;  198          1101  1
;  199          1102  1 !++
;  200          1103  1 ! FUNCTIONAL DESCRIPTION:
;  201          1104  1 !
;  202          1105  1 !       This routine requests a line of input from SYS$INPUT.  The prompt
;  203          1106  1 !       string is based on the prompt code passed as input.  The command
;  204          1107  1 !       line is read.
;  205          1108  1 !
;  206          1109  1 !       If the command read from the device SYS$INPUT is interpreted by
;  207          1110  1 !       RMS as EOF, or any other nonsuccessful return from RMS is seen,
;  208          1111  1 !       THEN cancel the command taking flag, and return.
;  209          1112  1 !
;  210          1113  1 !
;  211          1114  1 ! CALLING SEQUENCE:
;  212          1115  1 !
;  213          1116  1 !       PAT$PROMPT_READ(PROMPT_CODE, STG_DESC)
;  214          1117  1 !
;  215          1118  1 ! INPUTS:
;  216          1119  1 !
;  217          1120  1 !       PROMPT_CODE - Code for which prompt to display
;  218          1121  1 !       STG_DESC - Input line string descriptor
;  219          1122  1 !
;  220          1123  1 ! IMPLICIT INPUTS:
;  221          1124  1 !
;  222          1125  1 !       SYS$INPUT is already opened as the input channel.
;  223          1126  1 !
;  224          1127  1 ! OUTPUTS:
;  225          1128  1 !
;  226          1129  1 !       The input line string descriptor is set.
;  227          1130  1 !
;  228          1131  1 ! IMPLICIT OUTPUTS:
;  229          1132  1 !
;  230          1133  1 !       The new input line is read.
;  231          1134  1 !
;  232          1135  1 ! ROUTINE VALUE:
;  233          1136  1 !
;  234          1137  1 !       Status code from read.
;  235          1138  1 !
;  236          1139  1 ! SIDE EFFECTS:
;  237          1140  1 !
;  238          1141  1 !       The command line is written to the journal file.
;  239          1142  1 !
;  240          1143  1 !--
;  241          1144  1
;  242          1145  2 BEGIN
;  243          1146  2
;  244          1147  2 LITERAL
;  245          1148  2         NUM_OF_DSCS = 5,                              ! Number of default input line descriptors
;  246          1149  2         MORE_DSCS = 2,                                ! Size to increment input line table
;  247          1150  2         RIGHT_ANGLE = %X'3E',
;  248          1151  2         SPACES = %X'20',
;  249          1152  2         COMMENT_CHAR = %X'21',                        ! Comment delimiter
;  250          1153  2         NULL_BYTE_LOC   = 1,
;  251          1154  2         MAX_PMT_SIZ = 8,
;  252          1155  2         CONTINUE_PMT = 7;
;  253          1156  2
```

```
;  254          1157  2 MACRO
;  255        M 1158  2         PROMPT_MSG(PROMPT_STR) = UPLIT BYTE (%ASCII %STRING
;  256        M 1159  2                                   (%CHAR (CARRIAGE_RET),%CHAR(LINEFEED),
;  257          1160  2                                    PROMPT_STR,%CHAR(RIGHT_ANGLE)))%,
;  258        M 1161  2         PROMPT_SIZ(PROMPT_STR) = %CHARCOUNT (%ASCII %STRING
;  259        M 1162  2                                   (%CHAR (CARRIAGE_RET),%CHAR(LINEFEED),
;  260          1163  2                                    PROMPT_STR,%CHAR(RIGHT_ANGLE)))%;
;  261          1164  2
;  262          1165  2 OWN
;  263          1166  2         PROMPT_MSG_PTR : VECTOR[8,LONG] INITIAL (                ! Prompt texts
;  264          1167  2                                 PROMPT_MSG('PATCH'),
;  265          1168  2                                 PROMPT_MSG('OLD'),
;  266          1169  2                                 PROMPT_MSG('NEW'),
;  267          1170  2                                 PROMPT_MSG('LOC'),
;  268          1171  2                                 PROMPT_MSG('NAM'),
;  269          1172  2                                 PROMPT_MSG('EXP'),
;  270          1173  2                                 PROMPT_MSG('ECO'),
;  271          1174  2                                 PROMPT_MSG('_')),
;  272          1175  2         PROMPT_MSG_SIZ : VECTOR[8,BYTE] INITIAL ( BYTE(          ! Prompt text lengths
;  273          1176  2                                 PROMPT_SIZ('PATCH'),
;  274          1177  2                                 PROMPT_SIZ('OLD'),
;  275          1178  2                                 PROMPT_SIZ('NEW'),
;  276          1179  2                                 PROMPT_SIZ('LOC'),
;  277          1180  2                                 PROMPT_SIZ('NAM'),
;  278          1181  2                                 PROMPT_SIZ('EXP'),
;  279          1182  2                                 PROMPT_SIZ('ECO'),
;  280          1183  2                                 PROMPT_SIZ('_')));
;  281          1184  2
;  282          1185  2 LOCAL
;  283          1186  2         PMT_OFFSET,                                          ! Offset into prompt tables
;  284          1187  2         PROMPT_BUFFER: VECTOR [MAX_PMT_SIZ-2+NO_OF_INP_CHARS+A_LONGWORD, BYTE],
;  285          1188  2         INPUT_BUFFER: REF VECTOR[,BYTE],
;  286          1189  2         PREV_COUNT,                                          ! Current character count
;  287          1190  2         OLD_POINTER,                                         ! Pointer to previous buffer
;  288          1191  2         LOOP,                                                ! Loop counter
;  289          1192  2         NEW_DSCS,                                            ! Pointer to new input line descriptor table
;  290          1193  2         NEW_POINTER : REF VECTOR [, BYTE];                   ! Pointer to current buffer
;  291          1194  2
;  292          1195  2 LABEL
;  293          1196  2         GET_LINE,                                            ! Block to get a command line
;  294          1197  2
;  295          1198  2         FIND_COMMENT;                                        ! Block to find comment lines
;  296          1199  2
;  297          1200  2 MAP
;  298          1201  2         STG_DESC: REF BLOCK [8, BYTE];                       ! String descriptor for parser
;  299          1202  2
;  300          1203  2 !++
;  301          1204  2 ! Compute the offset into the prompt tables for the prompt code.
;  302          1205  2 ! Set the INPUT_BUFFER to be the local input buffer.
;  303          1206  2 !--
;  304          1207  2 PMT_OFFSET = .PROMPT_CODE - MIN_PMT_CODE;
;  305          1208  2 INPUT_BUFFER = CH$PTR(PROMPT_BUFFER[MAX_PMT_SIZ-2], 0);
;  306          1209  2
;  307          1210  2 !++
;  308          1211  2 ! Collect an entire command line.  Enter a loop that collects multiple lines of
;  309          1212  2 ! input, ceasing only when a line ends with other than a hyphen ("-"), which
;  310          1213  2 ! is the line continuation character. Buffer the possibly multiple lines into
```

```
;  311          1214  2 ! free storage.
;  312          1215  2 !--
;  313          1216  2 PREV_COUNT = 0;
;  314          1217  2 OLD_POINTER = 0;
;  315          1218  2 PAT$GL_INPRAB [RAB$L_PBF] = CH$PTR(.PROMPT_MSG_PTR[.PMT_OFFSET],0);
;  316          1219  2 PAT$GL_INPRAB [RAB$B_PSZ] = .PROMPT_MSG_SIZ[.PMT_OFFSET];
;  317          1220  2 PAT$GL_INPRAB [RAB$L_UBF] = CH$PTR(.INPUT_BUFFER, 0);
;  318          1221  2
;  319          1222  2 !++
;  320          1223  2 ! Now get a command line.  If the first character is a comment delimiter
;  321          1224  2 ! then get another command line.
;  322          1225  2 !--
;  323          1226  2 GET_LINE:
;  324          1227  3 BEGIN
;  325          1228  3 REPEAT
;  326          1229  4         BEGIN
;  327          1230  4         PAT$GL_ERRCODE = $GET (RAB = PAT$GL_INPRAB);
;  328          1231  4         IF NOT .PAT$GL_ERRCODE
;  329          1232  4         THEN
;  330          1233  5                 BEGIN
;  331          1234  5                 !++
;  332          1235  5                 ! The $GET failed. Set status so that PATCH returns to CLI.
;  333          1236  5                 !--
;  334          1237  6                 IF (.PAT$GL_ERRCODE EQL RMS$_EOF)
;  335          1238  5                 THEN
;  336          1239  6                         BEGIN
;  337          1240  6                         PAT$GB_TAKE_CMD = FALSE;
;  338          1241  6                         RETURN(.PAT$GL_ERRCODE);
;  339          1242  6                         END
;  340          1243  5                 ELSE
;  341          1244  5                         SIGNAL (PAT$_READERR,1,GETFILDSC(PAT$GL_INPFAB),.PAT$GL_INPRAB[RAB$I_STS],.PAT$GL_IN
;  342          1245  5                 END
;  343          1246  4         ELSE
;  344          1247  4             FIND_COMMENT:
;  345          1248  5                 BEGIN
;  346          1249  5                 LOCAL
;  347          1250  5                         MSG_SIZ;
;  348          1251  5                 MSG_SIZ=.PAT$GL_INPRAB[RAB$W_RSZ];
;  349          1252  6                 WHILE (.MSG_SIZ GTR 0)
;  350          1253  5                 DO
;  351          1254  6                         BEGIN
;  352          1255  7                         IF (.INPUT_BUFFER[.PAT$GL_INPRAB[RAB$W_RSZ]-.MSG_SIZ] EQL %X'20')
;  353          1256  7                             OR (.INPUT_BUFFER[.PAT$GL_INPRAB[RAB$W_RSZ]-.MSG_SIZ] EQL %X'09')
;  354          1257  6                         THEN
;  355          1258  6                                 MSG_SIZ = .MSG_SIZ - 1
;  356          1259  6                         ELSE
;  357          1260  7                                 IF (.INPUT_BUFFER[.PAT$GL_INPRAB[RAB$W_RSZ]-.MSG_SIZ] EQL COMMENT_CHAR)
;  358          1261  6                                 THEN
;  359          1262  7                                         BEGIN
;  360          1263  7                                         PAT$WRITEFILE  (.PAT$GL_INPRAB[RAB$W_RSZ],
;  361          1264  7                                                         CH$PTR(.INPUT_BUFFER, 0),
;  362          1265  7                                                         PAT$GL_JNLRAB),
;  363          1266  7                                         LEAVE FIND_COMMENT;
;  364          1267  7                                         END
;  365          1268  6                                 ELSE
;  366          1269  6                                         LEAVE GET_LINE;
;  367          1270  5                         END;
```

```
;  368          1271  4                    END;
;  369          1272  3            END;
;  370          1273  2 END;
;  371          1274  2
;  372          1275  2 !++
;  373          1276  2 ! Insert the prompt (without the <CR><LF>) at the start of the input buffer and
;  374          1277  2 ! write the prompt plus the command to the journal file.
;  375          1278  2 !--
;  376          1279  2 CH$COPY(.PROMPT_MSG_SIZ[.PMT_OFFSET]-2, CH$PTR(.PROMPT_MSG_PTR[.PMT_OFFSET],2),
;  377          1280  2                    SPACES, MAX_PMT_SIZ-2, CH$PTR(PROMPT_BUFFER,0));
;  378          1281  2 PAT$WRITEFILE(.PAT$GL_INPRAB[RAB$W_RSZ]+MAX_PMT_SIZ-2, PROMPT_BUFFER,
;  379          1282  2                    PAT$GL_JNLRAB);
;  380          1283  2
;  381          1284  2 !++
;  382          1285  2 ! Now change the prompt to an underscore in case there are continuation lines.
;  383          1286  2 !--
;  384          1287  2 CH$COPY(.PROMPT_MSG_SIZ[CONTINUE_PMT]-2, CH$PTR(.PROMPT_MSG_PTR[CONTINUE_PMT],2),
;  385          1288  2                    SPACES, MAX_PMT_SIZ-2,  CH$PTR(PROMPT_BUFFER,0));
;  386          1289  2 PAT$GL_INPRAB [RAB$V_PTA] = FALSE;
;  387          1290  2 PAT$GL_INPRAB [RAB$L_PBF] = CH$PTR(.PROMPT_MSG_PTR[CONTINUE_PMT],0);
;  388          1291  2 PAT$GL_INPRAB [RAB$B_PSZ] = .PROMPT_MSG_SIZ[CONTINUE_PMT];
;  389          1292  2
;  390          1293  2 !++
;  391          1294  2 ! Loop, reading all continuation lines for this command.
;  392          1295  2 !--
;  393          1296  2 REPEAT
;  394          1297  3         BEGIN
;  395          1298  3
;  396          1299  3         LOCAL
;  397          1300  3                 CONT_LINE;                                  ! Boolean test for end of line character
;  398          1301  3
;  399          1302  4         IF (.INPUT_BUFFER [.PAT$GL_INPRAB [RAB$W_RSZ] - 1] EQL '-')
;  400          1303  3         THEN
;  401          1304  4                 BEGIN
;  402          1305  4                 PAT$GL_INPRAB [RAB$W_RSZ] = .PAT$GL_INPRAB [RAB$W_RSZ] - 1;
;  403          1306  4                 CONT_LINE = TRUE;
;  404          1307  4                 END
;  405          1308  3         ELSE CONT_LINE = FALSE;
;  406          1309  3
;  407          1310  3         !++
;  408          1311  3         ! Allocate space for this buffer plus all previous buffers.
;  409          1312  3         ! If the space can be found, write the old and new buffers
;  410          1313  3         ! into the new space.
;  411          1314  3         !--
;  412          1315  4         NEW_POINTER = PAT$freez ((.PREV_COUNT + NULL_BYTE_LOC +
;  413          1316  3                 .PAT$GL_INPRAB [RAB$W_RSZ] + 3) / 4);
;  414          1317  3
;  415          1318  4         IF (.OLD_POINTER NEQ 0)                                                                    ; I
;  416          1319  3         THEN
;  417          1320  4                 BEGIN
;  418          1321  4                 CH$MOVE (.PREV_COUNT, .OLD_POINTER, .NEW_POINTER);
;  419          1322  4                 PAT$FREERELEASE (.OLD_POINTER, (.PREV_COUNT + NULL_BYTE_LOC + 3) / 4);
;  420          1323  3                 END;
;  421          1324  3         CH$MOVE (.PAT$GL_INPRAB [RAB$W_RSZ], CH$PTR(.INPUT_BUFFER,0),
;  422          1325  3                         CH$PLUS (.NEW_POINTER, .PREV_COUNT));
;  423          1326  3         PREV_COUNT = .PREV_COUNT + .PAT$GL_INPRAB [RAB$W_RSZ];
;  424          1327  3         NEW_POINTER [.PREV_COUNT] = 0;
```

```
;  425          1328  3          OLD_POINTER = .NEW_POINTER;
;  426          1329  3
;  427          1330  3          !++
;  428          1331  3          ! See whether this line ends with a continuation character.
;  429          1332  3          ! If so, get another line.
;  430          1333  3          !--
;  431          1334  3          IF NOT .CONT_LINE
;  432          1335  3          THEN EXITLOOP;
;  433          1336  3          PAT$GL_ERRCODE = $GET (RAB = PAT$GL_INPRAB);
;  434          1337  3          IF NOT .PAT$GL_ERRCODE
;  435          1338  3          THEN
;  436          1339  3                  SIGNAL (PAT$_READERR,1,GETFILDSC(PAT$GL_INPFAB),.PAT$GL_INPRAB[RAB$L_STS],.PAT$GL_INPRAB[RAB
;  437          1340  3          PAT$WRITEFILE(.PAT$GL_INPRAB[RAB$W_RSZ]+MAX_PMT_SIZ-2, PROMPT_BUFFER,
;  438          1341  3                        PAT$GL_JNLRAB);
;  439          1342  2          END;
;  440          1343  2
;  441          1344  2  !++
;  442          1345  2  ! A complete line has been collected.  Now set up the return input line descriptor.
;  443          1346  2  !--
;  444          1347  2  STG_DESC [dsc$w_length] = .PREV_COUNT;
;  44_          1348  2  STG_DESC [dsc$a_pointer] = .NEW_POINTER;
;  446          1349  2
;  447          1350  2  !++
;  448          1351  2  ! The input line must be entered in the input descriptor table for deallocation
;  449          1352  2  ! when the command is over.  PAT$CP_INP_DSCS is a pointer to a vector of
;  450          1353  2  ! longwords.  The first longword is the number of string descriptors in the
;  451          1354  2  ! vector.  The remaining space is string descriptors, one for each input line
;  452          1355  2  ! used for this command.  Any unused descriptors are filled with zeros.
;  453          1356  2  ! PAT$END_OF_LINE in PATACT.B32 will deallocate the input lines.
;  454          1357  2  !--
;  455          1358  2
;  456          1359  2  !++
;  457          1360  2  ! First check that there is a descriptor table allocated.
;  458          1361  2  ! If not, then allocate one and initialize the count of descriptors.
;  459          1362  2  !--
;  460          1363  3  IF (.PAT$CP_INP_DSCS EQLA 0)
;  461          1364  2  THEN
;  462          1365  3          BEGIN
;  463          1366  3          PAT$CP_INP_DSCS = PAT$FREEZ(((A_QUADWORD * NUM_OF_DSCS) + A_LONGWORD +3)/4);
;  464          1367  3          PAT$CP_INP_DSCS[0] = NUM_OF_DSCS;
;  465          1368  2          END;
;  466          1369  2
;  467          1370  2  !++
;  468          1371  2  ! Now enter a string descriptor for the input line just read.  Use the first
;  469          1372  2  ! zero entry in the descriptor table.  If there is none, then a larger table
;  470          1373  2  ! must be allocated and the old table deallocated.
;  471          1374  2  !--
;  472          1375  2  INCR LOOP FROM 1 TO .PAT$CP_INP_DSCS[0]*2 BY 2
;  473          1376  2  DO
;  474          1377  3          IF (.PAT$CP_INP_DSCS[.LOOP] EQL 0)
;  475          1378  2          THEN
;  476          1379  3                  BEGIN
;  477          1380  3                  PAT$CP_INP_DSCS [.LOOP] = .PREV_COUNT + 1;
;  478          1381  3                  PAT$CP_INP_DSCS [.LOOP + 1] = .NEW_POINTER;
;  479          1382  3                  RETURN(.PAT$GL_ERRCODE);
;  480          1383  2                  END;
;  481          1384  2
```

```
;  482          1385  2 !++
;  483          1386  2 ! There was not enough room in the descriptor table.  Therefore allocate a
;  484          1387  2 ! larger table, copy in the old table, deallocate the old table, and insert
;  485          1388  2 ! the new descriptor into the table.
;  486          1389  2 !--
;  487          1390  3 NEW_DSCS = PAT$FREEZ( (((.PAT$CP_INP_DSCS[0] + MORE_DSCS)* A_QUADWORD) + A_LONGWORD
;  488          1391  2                                                 +3) / 4);
;  489          1392  2 CH$MOVE(((.PAT$CP_INP_DSCS[0] * A_QUADWORD) + A_LONGWORD), CH$PTR(.PAT$CP_INP_DSCS, 0),
;  490          1393  2         CH$PTR(.NEW_DSCS, 0));
;  491          1394  3 PAT$FREERELEASE(.PAT$CP_INP_DSCS, ((.PAT$CP_INP_DSCS[0] * A_QUADWORD) + A_LONGWORD
;  492          1395  2                                 + 3) / 4);
;  493          1396  2 PAT$CP_INP_DSCS = CH$PTR (.NEW_DSCS, 0);
;  494          1397  2 PAT$CP_INP_DSCS[(.PAT$CP_INP_DSCS[0]*2)+1] = .PREV_COUNT + 1;
;  495          1398  2 PAT$CP_INP_DSCS[(.PAT$CP_INP_DSCS[0]*2)+2] = .NEW_POINTER;
;  496          1399  2 PAT$CP_INP_DSCS[0] = .PAT$CP_INP_DSCS[0] + MORE_DSCS;
;  497          1400  2 RETURN(.PAT$GL_ERRCODE);
;  498          1401  1 END;


                                                                .PSECT  _PAT$PLIT,NOWRT,NOEXE,0

                    3E  48  43  54  41  50  0A  0D  00000 P.AAA:  .ASCII  <13><10>\PATCH>\                          ;
                            3E  44  4C  4F  0A  0D  00008 P.AAB:  .ASCII  <13><10>\OLD>\                           ;
                            3E  57  45  4E  0A  0D  0000E P.AAC:  .ASCII  <13><10>\NEW>\                           ;
                            3E  43  4F  4C  0A  0D  00014 P.AAD:  .ASCII  <13><10>\LOC>\                           ;
                            3E  4D  41  4E  0A  0D  0001A P.AAE:  .ASCII  <13><10>\NAM>\                           ;
                            3E  50  58  45  0A  0D  00020 P.AAF:  .ASCII  <13><10>\EXP>\                           ;
                            3E  4F  43  45  0A  0D  00026 P.AAG:  .ASCII  <13><10>\ECO>\                           ;
                                    3E  5F  0A  0D  0002C P.AAH:  .ASCII  <13><10>\_>\                             ;

                                                                .PSECT  _PAT$OWN,NOEXE,2

00000000' 00000000' 00000000' 00000000' 00000000' 00000000' 00000 PROMPT_MSG_PTR:
                                                                .ADDRESS P.AAA, P.AAB, P.AAC, P.AAD, P.AAE, -          ;
                                    00000000' 00000000' 00018       P.AAF, P.AAG, P.AAH                            ;
                        04  06  06  06  06  06  06  08  00020 PROMPT_MSG_SIZ:
                                                                .BYTE   8, 6, 6, 6, 6, 6, 6, 4                         ;

                                                                .EXTRN  SYS$GET

                                                                .PSECT  _PAT$CODE,NOWRT,2

                                                                .ENTRY  PAT$PROMPT_READ, Save R2,R3,R4,R5,R6,R7,R8,-; 1100
                                                                        R9,R10,R11                                     ;
                                5B 00000000G   EF  9E 00002     MOVAB   PAT$GL_INPRAB+34, R11                          ;
                                5E      FF70   CE  9E 00009     MOVAB   -144(SP), SP                                   ;
                        52      04     AC 0000005A   8F  C3 0000E     SUBL3   #90, PROMPT_CODE, PMT_OFFSET           ; 1207
                                57           06   AE  9E 00017     MOVAB   PROMPT_BUFFER+6, INPUT_BUFFER          ; 1208
                                                  56  D4 0001B     CLRL    PREV_COUNT                             ; 1216
                                                  5A  D4 0001D     CLRL    OLD_POINTER                            ; 1217
                                0E     AB 00000000'EF42  D0 0001F     MOVL    PROMPT_MSG_PTR[PMT_OFFSET], -          ; 1218
                                                                        PAT$GL_INPRAB+48                               ;
                                12     AB 00000000'EF42  90 00028     MOVB    PROMPT_MSG_SIZ[PMT_OFFSET], -          ; 1219
                                                                        PAT$GL_INPRAB+52                               ;
                                02     AB           57  D0 00031     MOVL    INPUT_BUFFER, PAT$GL_INPRAB+36         ; 1220
                                             DE     AB  9F 00035 1$:  PUSHAB  PAT$GL_INPRAB                          ; 1230
```

```
                00000000G  00              01  FB 00038          CALLS   #1, SYS$GET
                00000000G  EF              50  D0 0003F          MOVL    R0, PAT$GL_ERRCODE
                           34              50  E8 00046          BLBS    R0, 3$                                  ;  1231
                00018`7A   8F              50  D1 00049          CMPL    R0, #98938                              ;  1237
                                           07  12 00050          BNEQ    2$
                               00000000G   EF  D4 00052          CLRL    PAT$GB_TAKE_CMD                         ;  1240
                                           04  00058          RET                                             ;  1241
                           7E        E6    AB  7D 00059 2$:     MOVQ    PAT$GL_INPRAB+8, -(SP)                  ;  1244
                               00000000G   EF  9F 0005D          PUSHAB  PAT$GL_INPFAB
                00000000G  EF              01  FB 00063          CALLS   #1, GETFILDSC
                                           50  DD 0006A          PUSHL   R0
                                           01  DD 0006C          PUSHL   #1
                               00000000G   8F  DD 0006E          PUSHL   #PAT$_READERR
                00000000G  00              05  FB 00074          CALLS   #5, LIB$SIGNAL
                                           B8  11 0007B          BRB     1$                                      ;  1231
                           53              6B  3C 0007D 3$:     MOVZWL  PAT$GL_INPRAB+34, MSG_SIZ               ;  1231
                                           B3  15 00080 4$:     BLEQ    1$                                      ;  1232
                           50              6B  3C 00082          MOVZWL  PAT$GL_INPRAB+34, R0                    ;  1233
                           50              53  C2 00085          SUBL2   MSG_SIZ, R0
                           20          6047    91 00088          CMPB    (R0)[INPUT_BUFFER], #32
                                           06  13 0008C          BEQL    5$
                           09          6047    91 0008E          CMPB    (R0)[INPUT_BUFFER], #9                  ;  1256
                                           04  12 00092          BNEQ    6$
                                           53  D7 00094 5$:     DECL    MSG_SIZ                                 ;  1258
                                           E8  11 00096          BRB     4$
                           21          6047    91 00098 6$:     CMPB    (R0)[INPUT_BUFFER], #33                 ;  1260
                                           14  12 0009C          BNEQ    7$
                               00000000G   EF  9F 0009E          PUSHAB  PAT$GL_JNLRAB                           ;  1264
                                           57  DD 000A4          PUSHL   INPUT_BUFFER
                           7E              6B  3C 000A6          MOVZWL  PAT$GL_INPRAB+34, -(SP)
                00000000G  EF              03  FB 000A9          CALLS   #3, PAT$WRITEFILE
                                           83  11 000B0          BRB     1$                                      ;  1266
                           51  00000000'EF42  9A 000B2 7$:     MOVZBL  PROMPT_MSG_SIZ[PMT_OFFSET], R1          ;  1279
                           51              02  C2 000BA          SUBL2   #2, R1
                           50  00000000'EF42  D0 000BD          MOVL    PROMPT_MSG_PTR[PMT_OFFSET], R0
06              20      02     A0              51  2C 000C5          MOVC5   R1, 2(R0), #32, #6, PROMPT_BUFFER       ;  1280
                                               6E      000CB
                               00000000G   EF  9F 000CC          PUSHAB  PAT$GL_JNLRAB                           ;  1281
                                       04  AE  9F 000D2          PUSHAB  PROMPT_BUFFER
                           7E              6B  3C 000D5          MOVZWL  PAT$GL_INPRAB+34, -(SP)
                           6E              06  C0 000D8          ADDL2   #6, (SP)
                               00000000G   EF  03  FB 000DB          CALLS   #3, PAT$WRITEFILE
                           50  00000000'  EF  9A 000E2          MOVZBL  PROMPT_MSG_SIZ+7, R0                    ;  1287
                           50              02  C2 000E9          SUBL2   #2, R0
                           58  00000000'  EF  D0 000EC          MOVL    PROMPT_MSG_PTR+28, R8
06              20      02     A8              50  2C 000F3          MOVC5   R0, 2(R8), #32, #6, PROMPT_BUFFER       ;  1288
                                               6E      000F9
                           E5  AB              20  8A 000FA          BICB2   #32, PAT$GL_INPRAB+7                    ;  1289
                           0E  AB              58  D0 000FE          MOVL    R8, PAT$GL_INPRAB+48                    ;  1290
                           12  AB  00000000'    EF  90 00102          MOVB    PROMPT_MSG_SIZ+7, PAT$GL_INPRAB+52      ;  1291
                                           50  6B  3C 0010A 8$:     MOVZWL  PAT$GL_INPRAB+34, R0                    ;  1302
                                       2D      FF A047  91 0010D          CMPB    -1(R0)[INPUT_BUFFER], #45
                                               07  12 00112          BNEQ    9$
                                               6B  B7 00114          DECW    PAT$GL_INPRAB+34                        ;  1305
                                           59  01  D0 00116          MOVL    #1, CONT_LINE                           ;  1306
                                               02  11 00119          BRB     10$                                     ;  1302
                                               59  D4 0011B 9$:     CLRL    CONT_LINE                               ;  1308
```

```
                50          6B  3C 00011D 10$:    MOVZWL  PAT$GL_INPRAB+34, R0                    ; 1316
                50    04 A046   9E 00120          MOVAB   4(R0)[PREV_COUNT], R0                   ; 1315
          7E    50          04  C7 00125          DIVL3   #4, R0, -(SP)                           ; 1316
    00000000G   EF          01  FB 00129          CALLS   #1, PAT$FREEZ                           ;
                58          50  D0 00130          MOVL    R0, NEW_POINTER                         ;
                            5A  D5 00133          TSTL    OLD_POINTER                             ; 1318
                            15  13 00135          BEQL    11$                                     ;
          68    6A          56  28 00137          MOVC3   PREV_COUNT, (OLD_POINTER), (NEW_POINTER) ; 1321
                50    04    A6  9E 0013B          MOVAB   4(R6), R0                               ; 1322
          7E    50          04  C7 0013F          DIVL3   #4, R0, -(SP)                           ;
                            5A  DD 00143          PUSHL   OLD_POINTER                             ;
    00000000G   EF          02  FB 00145          CALLS   #2, PAT$FREERELEASE                     ;
        6648    67          6B  28 0014C 11$:     MOVC3   PAT$GL_INPRAB+34, (INPUT_BUFFER), -     ; 1325
                                                          (PREV_COUNT)[NEW_POINTER]               ;
                50          6B  3C 00151          MOVZWL  PAT$GL_INPRAB+34, R0                    ; 1326
                56          50  C0 00154          ADDL2   R0, PREV_COUNT                          ;
                          6648  94 00157          CLRB    (PREV_COUNT)[NEW_POINTER]               ; 1327
                5A          58  D0 0015A          MOVL    NEW_POINTER, OLD_POINTER                ; 1328
                53          59  E9 0015D          BLBC    CONT_LINE, 13$                          ; 1334
                      DE    AB  9F 00160          PUSHAB  PAT$GL_INPRAB                           ; 1336
    00000000G   00          01  FB 00163          CALLS   #1, SYS$GET                             ;
    00000000G   EF          50  D0 0016A          MOVL    R0, PAT$GL_ERRCODE                      ;
                22 00000000G  EF  E8 00171          BLBS    PAT$GL_ERRCODE, 12$                     ; 1337
                7E        E6  AB  7D 00178          MOVQ    PAT$GL_INPRAB+8, -(SP)                  ; 1339
                   00000000G  EF  9F 0017C          PUSHAB  PAT$GL_INPFAB                           ;
    00000000G   EF          01  FB 00182          CALLS   #1, GETFILDSC                           ;
                            50  DD 00189          PUSHL   R0                                      ;
                            01  DD 0018B          PUSHL   #1                                      ;
                00000000G   8F  DD 0018D          PUSHL   #PAT$_READERR                           ;
    00000000G   00          05  FB 00193          CALLS   #5, LIB$SIGNAL                          ;
                00000000G   EF  9F 0019A 12$:     PUSHAB  PAT$GL_JNLRAB                           ; 1340
                      04    AE  9F 001A0          PUSHAB  PROMPT_BUFFER                           ;
                7E          6B  3C 001A3          MOVZWL  PAT$GL_INPRAB+34, -(SP)                 ;
                6E          06  C0 001A6          ADDL2   #6, (SP)                                ;
    00000000G   EF          03  FB 001A9          CALLS   #3, PAT$WRITEFILE                       ;
                          FF57  31 001B0          BRW     8$                                      ; 1291
                50    08    AC  D0 001B3 13$:     MOVL    STG_DESC, R0                            ; 1347
                60          56  B0 001B7          MOVW    PREV_COUNT, (R0)                        ;
          04    A0          58  D0 001BA          MOVL    NEW_POINTER, 4(R0)                      ; 1348
                00000000G   EF  D5 001BE          TSTL    PAT$CP_INP_DSCS                         ; 1363
                            17  12 001C4          BNEQ    14$                                     ;
                            0B  DD 001C6          PUSHL   #11                                     ; 1366
    00000000G   EF          01  FB 001C8          CALLS   #1, PAT$FREEZ                           ;
    00000000G   EF          50  D0 001CF          MOVL    R0, PAT$CP_INP_DSCS                     ;
    00000000G   FF          05  D0 001D6          MOVL    #5, @PAT$CP_INP_DSCS                    ; 1367
                51 00000000G  EF  D0 001DD 14$:     MOVL    PAT$CP_INP_DSCS, R1                     ; 1375
                50          61  D0 001E4          MOVL    (R1), R0                                ;
          53    50          01  78 001E7          ASHL    #1, R0, R3                              ;
                52          01  CE 001EB          MNEGL   #1, LOOP                                ;
                            11  11 001EE          BRB     16$                                     ;
                          6142  D5 001F0 15$:     TSTL    (R1)[LOOP]                              ; 1377
                            0C  12 001F3          BNEQ    16$                                     ;
                      6142  01  A6  9E 001F5          MOVAB   1(R6), (R1)[LOOP]                   ; 1380
                04    A142        58  D0 001FA          MOVL    NEW_POINTER, 4(R1)[LOOP]            ; 1381
                                  6A  11 001FF          BRB     17$                                 ; 1382
    FFE9    52          02          53  F1 00201 16$:     ACBL    R3, #2, LOOP, 15$               ; 1377
                50          08      C4  00207          MULL2   #8, R0                          ; 1390
```

```
                         50              17  C0 0020A          ADDL2   #23, R0
                      7E 50              04  C7 0020D          DIVL3   #4, R0, -(SP)                   ; 1391
                00000000G EF             01  FB 00211          CALLS   #1, PAT$FREEZ
                         5A              50  D0 00218          MOVL    R0, NEW_DSCS
                         57 00000000G    EF  D0 0021B          MOVL    PAT$CP_INP_DSCS, R7             ; 1392
                         59              67  D0 00222          MOVL    (R7), R9
                   50    59              03  78 00225          ASHL    #3, R9, R0
                         50              04  C0 00229          ADDL2   #4, R0
                   6A    67              50  28 0022C          MOVC3   R0, (R7), (NEW_DSCS)            ; 1393
                   50    59              03  78 00230          ASHL    #3, R9, R0                      ; 1394
                         50              07  C0 00234          ADDL2   #7, R0
                      7E 50              04  C7 00237          DIVL3   #4, R0, -(SP)                   ; 1395
                         57              DD  0023B          PUSHL   R7                              ; 1394
                00000000G EF             02  FB 0023D          CALLS   #2, PAT$FREERELEASE
                00000000G EF             5A  D0 00244          MOVL    NEW_DSCS, PAT$CP_INP_DSCS       ; 1396
                         51 00000000G    EF  D0 0024B          MOVL    PAT$CP_INP_DSCS, R1             ; 1397
                         52              61  D0 00252          MOVL    (R1), R2
                   50    52              01  78 00255          ASHL    #1, R2, R0
                      04 A140        01   A6  9E 00259          MOVAB   1(R6), 4(R1)[R0]
                   50    52              01  78 0025F          ASHL    #1, R2, R0                      ; 1398
                      08 A140             58  D0 00263          MOVL    NEW_POINTER, 8(R1)[R0]
                         61              02  C0 00268          ADDL2   #2, (R1)                        ; 1399
                         50 00000000G    EF  D0 0026B 17$:     MOVL    PAT$GL_ERRCODE, R0              ; 1400
                                             04 00272          RET                                     ; 1401

; Routine Size:  627 bytes,    Routine Base:  _PAT$CODE + 003D
```

```
;  500          1402  1 END
;  501          1403  0 ELUDOM


                                                .EXTRN  LIB$SIGNAL

;                               PSECT SUMMARY

;       Name                    Bytes                   Attributes

;   _PAT$CODE                     688  NOVEC,NOWRT,  RD ,  EXE,NOSHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)
;   _PAT$PLIT                      48  NOVEC,NOWRT,  RD ,NOEXE,NOSHR,  LCL,  REL,  CON,NOPIC,ALIGN(0)
;   _PAT$OWN                       40  NOVEC,  WRT,  RD ,NOEXE,NOSHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)


;                           Library Statistics

;                                    -------- Symbols --------      Pages       Processing
;       File                         Total    Loaded   Percent      Mapped      Time

;   _$255$DUA28:[SYSLIB]LIB.L32;1    18619        18         0      1000          00:01.8


;                                   COMMAND QUALIFIERS

;       BLISS/CHECK=(FIELD,INITIAL,OPTIMIZE)/VARIANT:1/LIS=LIS$:PATCMD/OBJ=OBJ$:PATCMD MSRC$:PATCMD/UPDATE=(ENH$:PATCMD)

; Size:         688 code + 88 data bytes
; Run Time:         00:23.0
; Elapsed Time:     01:31.8
; Lines/CPU Min:     3666
; Lexemes/CPU-Min: 38226
; Memory Used:  236 pages
; Compilation Complete
```

PATARI
LIS

PATCMD
LIS

PATECO
LIS

PATCON
LIS

PATENC
LIS

PATBAS
LIS

PATBLD
LIS