```
DDDDDDDDDDD         CCCCCCCCCCC  LLL
DDDDDDDDDDDD        CCCCCCCCCCCC LLL
DDDDDDDDDDDD        CCCCCCCCCCCC LLL
DDD          DDD  CCC           LLL
DDD          DDD  CCC           LLL
DDD          DDD  CCC           LLL
DDD          DDD  CCC           LLL
DDD          DDD  CCC           LLL
DDD          DDD  CCC           LLL
DDD          DDD  CCC           LLL
DDD          DDD  CCC           LLL
DDD          DDD  CCC           LLL
DDD          DDD  CCC           LLL
DDD          DDD  CCC           LLL
DDD          DDD  CCC           LLL
DDD          DDD  CCC           LLL
DDD          DDD  CCC           LLL
DDDDDDDDDDDD        CCCCCCCCCCCC LLLLLLLLLLLLLLL
DDDDDDDDDDDD        CCCCCCCCCCCC LLLLLLLLLLLLLLL
DDDDDDDDDDDD        CCCCCCCCCCCC LLLLLLLLLLLLLLL
```

FILEID**INDIRECT

```
IIIIII    NN      NN  DDDDDDDD     IIIIII    RRRRRRRR    EEEEEEEEEE    CCCCCCCC  TTTTTTTTTT
IIIIII    NN      NN  DDDDDDDD     IIIIII    RRRRRRRR    EEEEEEEEEE    CCCCCCCC  TTTTTTTTTT
  II      NN      NN  DD      DD     II      RR      RR  EE          CC              TT
  II      NN      NN  DD      DD     II      RR      RR  EE          CC              TT
  II      NNNN    NN  DD      DD     II      RR      RR  EE          CC              TT
  II      NNNN    NN  DD      DD     II      RR      RR  EE          CC              TT
  II      NN  NN  NN  DD      DD     II      RRRRRRRR    EEEEEEEE    CC              TT
  II      NN  NN  NN  DD      DD     II      RRRRRRRR    EEEEEEEE    CC              TT
  II      NN    NNNN  DD      DD     II      RR  RR      EE          CC              TT
  II      NN    NNNN  DD      DD     II      RR  RR      EE          CC              TT
  II      NN      NN  DD      DD     II      RR    RR    EE          CC              TT
  II      NN      NN  DD      DD     II      RR    RR    EE          CC              TT
IIIIII    NN      NN  DDDDDDDD     IIIIII    RR      RR  EEEEEEEEEE    CCCCCCCC      TT
IIIIII    NN      NN  DDDDDDDD     IIIIII    RR      RR  EEEEEEEEEE    CCCCCCCC      TT


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

Table of contents

```
0000     1          .TITLE  INDIRECT - INDIRECT FILE MANIPULATION ROUTINES
0000     2          .IDENT  'V04-000'
0000     3
0000     4  ;
0000     5  ;*****************************************************************************
0000     6  ;*                                                                           *
0000     7  ;*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                                  *
0000     8  ;*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.                   *
0000     9  ;*  ALL RIGHTS RESERVED.                                                     *
0000    10  ;*                                                                           *
0000    11  ;*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED     *
0000    12  ;*  ONLY IN  ACCORDANCE WITH  THE  TERMS  OF  SUCH  LICENSE  AND WITH THE    *
0000    13  ;*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR  ANY  OTHER     *
0000    14  ;*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY     *
0000    15  ;*  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE IS  HEREBY     *
0000    16  ;*  TRANSFERRED.                                                             *
0000    17  ;*                                                                           *
0000    18  ;*  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE WITHOUT NOTICE     *
0000    19  ;*  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL EQUIPMENT    *
0000    20  ;*  CORPORATION.                                                             *
0000    21  ;*                                                                           *
0000    22  ;*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR  RELIABILITY OF ITS     *
0000    23  ;*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.                  *
0000    24  ;*                                                                           *
0000    25  ;*                                                                           *
0000    26  ;*****************************************************************************
0000    27  ;
0000    28  ; D. N. CUTLER 2-MAY-77
0000    29  ;
0000    30  ; INDIRECT FILE MANIPULATION ROUTINES
0000    31  ;
0000    32  ; MODIFIED BY:
0000    33  ;
0000    34  ;       V03-016 HWS0100         Harold Schultz  06-AUG-1984
0000    35  ;               Open any new indirect frame with Carriagecontrol
0000    36  ;               attributes.
0000    37  ;
0000    38  ;       V03-015 HWS0081         Harold Schultz  15-Jul-1984
0000    39  ;               When closing the current indirect frame and unstacking
0000    40  ;               the previous frame, set NAM block to not use the RSA and
0000    41  ;               ESA fields left by the indirect frame just closed. Using
0000    42  ;               these values cause the free dynamic memory list to become
0000    43  ;               corrupted. Add support for execute-only command procedures.
0000    44  ;
0000    45  ;       V03-014 HWS0080         Harold Schultz  12-Jul-1984
0000    46  ;               When allocating room in symbol table for resultant
0000    47  ;               name string, don't use constant of 256; use the value
0000    48  ;               of NAM$C_MAXRSS+1 rounded up to a long word boundry
0000    49  ;               instead. Remove the ASSUME of NAM$C_MAXRSS. Bypass
0000    50  ;               deallocation of unused buffer if none to deallocate.
0000    51  ;
0000    52  ;       V03-013 HWS0066         Harold Schultz  21-May-1984
0000    53  ;               Correct the error handling when a SYMOVF error is encountered
0000    54  ;               while setting up the new indirect level. Reenable password
0000    55  ;               masking after opening an indirect input file.
0000    56  ;
0000    57  ;       V03-012 HWS0015         Harold Schultz  21-Feb-1984
```

```
0000     58 ;                  Check status after $FIND.
0000     59 ;                  Initialize file spec. size fields in NAM block before reusing.
0000     60 ;                  Deassign SYS$INPUT prior to reopening a file at a prior indirect
0000     61 ;                  level.
0000     62 ;
0000     63 ;       V03-011 PCG0013         Peter George    12-Jan-1984
0000     64 ;                  Fix broken branch.
0000     65 ;
0000     66 ;       V03-010 PCG0012         Peter George    17-Aug-1983
0000     67 ;                  Correctly clear RMS F$SEARCH context.
0000     68 ;                  Manage concealed logical name attribute using the new services.
0000     69 ;
0000     70 ;       V03-009 PCG0011         Peter George    27-May-1983
0000     71 ;                  Fix bug in unstacking when restored command procedure
0000     72 ;                  is already positioned to EOF.
0000     73 ;
0000     74 ;       V03-009 PCG0011         Peter George    27-May-1983
0000     75 ;                  Fix bug in file name saving logic.
0000     76 ;                  Fix bugs in SYS$OUTPUT processing.
0000     77 ;
0000     78 ;       V03-008 KRM0099         Karl Malik      29-Apr-1983
0000     79 ;                  Disable password masking for network.
0000     80 ;
0000     81 ;       V03-007 PCG0010         Peter George    10-Apr-1983
0000     82 ;                  Finish making remote open work.
0000     83 ;
0000     84 ;       V03-006 PCG0009         Peter George    22-Feb-1983
0000     85 ;                  Add DCL$DEFINE_P1_TO_P8.
0000     86 ;                  Clear FAB$M_SQO bit.
0000     87 ;                  Clear FAB$v_NAM and FAB$W_IFI when performing
0000     88 ;                  remote reopen.
0000     89 ;
0000     90 ;       V03-005 PCG0008         Peter George    28-Jan-1983
0000     91 ;                  Remove reference to ONEXIT bit.
0000     92 ;
0000     93 ;       V03-004 PCG0007         Peter George    13-Jan-1983
0000     94 ;                  Call SYS$OUTPUT routines.
0000     95 ;                  Save name of command procedure.
0000     96 ;                  Use saved file name spec to reopen command procedures
0000     97 ;                  on remote nodes.
0000     98 ;
0000     99 ;       V03-003 PCG0006         Peter George    30-Dec-1982
0000    100 ;                  Clear PRC_V_ONEXIT when unstacking.
0000    101 ;
0000    102 ;       V03-002 PCG0005         Peter George    28-Oct-1982
0000    103 ;                  Fix CLRBIT typo.
0000    104 ;
0000    105 ;       V03-001 PCG0004         Peter George    15-Jul-1982
0000    106 ;                  Allow execute-only command procedures.
0000    107 ;---
0000    108
0000    109 ;
0000    110 ; MACRO LIBRARY CALLS
0000    111 ;
0000    112
0000    113          PRCDEF                                  ;DEFINE PROCESS WORK AREA
0000    114          WRKDEF                                  ;DEFINE COMMAND WORK AREA
```

```
                                0000   115          PTRDEF                          ;DEFINE RESULT PARSE DESCRIPTOR FORMAT
                                0000   116          IDFDEF                          ;DEFINE INDIRECT FRAME OFFSETS
                                0000   117          PRDDEF                          ;PROCESS RMS DATA
                                0000   118          SYMDEF                          ;DEFINE TYPES OF SYMBOLS
                                0000   119          $CLIMSGDEF                      ;DEFINE ERROR/STATUS VALUES
                                0000   120          $DEVDEF                         ;DEFINE DEVICE CHARACTERISTIC BITS
                                0000   121          $FABDEF                         ;DEFINE FAB OFFSETS
                                0000   122          $RABDEF                         ;DEFINE RAB OFFSETS
                                0000   123          $LOGDEF                         ;DEFINE LOG OFFSETS
                                0000   124          $NAMDEF                         ;DEFINE NAM OFFSETS
                                0000   125          $PSLDEF                         ;DEFINE PROCESSOR STATUS FIELDS
                                0000   126
                                0000   127 ;
                                0000   128 ; LOCAL SYMBOLS
                                0000   129 ;
                                0000   130
                   00000008     0000   131 SYMBOLS=8                               ;MAXIMUM NUMBER OF INDIRECT FILE SYMBOLS
                                0000   132
                                0000   133 ;
                                0000   134 ; LOCAL DATA
                                0000   135 ;
                                0000   136
                   00000000     0000   137          .PSECT  DCL$ZCODE,BYTE,RD,NOWRT
                                0000   138 INPFILE:                               ; INPUT FILE DEFAULT NAME STRING
                      4D 4F 43 2E 0000   139          .ASCII  /.COM/                  ;
                   54 55 50 54 55 4F 0004   140 OUTQUAL:.ASCII  /OUTPUT/           ; REST OF NAME AND THE QUALIFIER
                                000A   141 SYS_INPUT_NAME:
54 55 50 4E 49 24 53 59 53 00'  000A   142          .ASCIC  /SYS$INPUT/             ; LOGICAL NAME FOR SYS$INPUT
                             09 000A
                                0014   143 LNM$PROCESS:
53 53 45 43 4F 52 50 24 4D 4E 4C 00' 0014   144          .ASCIC  /LNM$PROCESS/           ; PROCESS LOGICAL NAME TABLF
                             0B 0014
                   3A 30 41 4C 4E 5F 00' 0020   145 NLA0:    .ASCIC  /_NLA0:/                ; NULL DEVICE
                             06 0020
```

```
                                0027   147          .SBTTL  STACK INDIRECT FILE
                                0027   148 ;+
                                0027   149 ; DCL$STACKIND - STACK INDIRECT FILE
                                0027   150 ;
                                0027   151 ; THIS ROUTINE IS CALLED TO STACK THE CURRENT INDIRECT FILE LEVEL AND TO PARSE
                                0027   152 ; AND OPEN THE NEXT INDIRECT FILE.
                                0027   153 ;
                                0027   154 ; INPUTS
                                0027   155 ;
                                0027   156 ;       IT IS ASSUMED THAT THE INDIRECT FILE PROCESSING FLAG IS SET.
                                0027   157 ;
                                0027   158 ; OUTPUTS:
                                0027   159 ;
                                0027   160 ;       THE CURRENT INDIRECT FILE SPECIFICATION IS SAVED ON THE INDIRECT FILE
                                0027   161 ;       STACK AND THE NEXT INDIRECT FILE IS PROCESSED.
                                0027   162 ;
                                0027   163 ;       R0 LOW BIT CLEAR INDICATES INDIRECT FILE PROCESSING FAILURE.
                                0027   164 ;
                                0027   165 ;               R0 = DCL$_ATLAST - INDIRECT FILE SPECIFICATION NOT LAST ITEM ON
                                0027   166 ;                    COMMAND LINE.
                                0027   167 ;               R0 = DCL$_DEFOVF - ATTEMPT TO DEFINE MORE THAN EIGHT PARAMETERS.
                                0027   168 ;               R0 = DCL$_STKOVF - INDIRECT FILE INTERNAL STACK OVERFLOW.
                                0027   169 ;
                                0027   170 ;       R0 LOW BIT SET INDICATES SUCCESSFUL COMPLETION.
                                0027   171 ;
                                0027   172 ;               R0 = DCL$_NORMAL - NORMAL COMPLETION.
                                0027   173 ;-
                                0027   174
                                0027   175 DCL$STACKIND::                                ;STACK INDIRECT FILE
                  0346    30    0027   176          BSBW    SETIND                      ;SET INDIRECT PROCESSING UP
              5E    C0 AE    9E 002A   177          MOVAB   -<SYMBOLS*8>(SP),SP         ;ALLOCATE SPACE FOR SYMBOL DESCRIPTO
                     7E    D4 002E   178          CLRL    -(SP)                       ;CLEAR COUNT OF GENERATED SYMBOLS
                 F48E CA    D7 0030   179          DECL    WRK_L_CHARPTR(R10)          ;BACK UP TO AT SIGN
                   FFC9'    30 0034   180 10$:     BSBW    DCL$MARK                    ;MARK CURRENT PARSE POSITION
                  53    03    9A 0037   181          MOVZBL  #PTR_K_PARAMETR,R3          ;SET TOKEN CONTEXT FOR FILESPEC
                   FFC3'    30 003A   182          BSBW    DCL$PROCFILE                ;PROCESS FILE SPECIFICATION
                    3C 50    E9 003D   183          BLBC    R0,15$                      ;IF LBC PARSE FAILURE
                   FFBD'    30 0040   184          BSBW    DCL$SETCHAR                 ;PEEK AT NEXT CHARACTER IN INPUT BUF
                  50    2F    91 0043   185          CMPB    #^A\/\,R0                   ;SLASH?
                        37    12 0046   186          BNEQ    20$                         ;IF NEQ NO
                   FFB5'    30 0048   187          BSBW    DCL$MOVTOKN                 ;MOVE TERMINATOR AND GET NEXT TOKEN
                  04    51    D1 004B   188          CMPL    R1,#4                       ;MORE THAN MAX MATCH NAME
                        03    19 004E   189          BLSS    13$                         ;BR IF NO
                  51    04    D0 0050   190          MOVL    #4,R1                       ;ONLY CHECK FOR 4 CHARS
                        50    DD 0053   191 13$:     PUSHL   R0                          ;SAVE TERMINATION CHARACTER
          AA AF   62    51    29 0055   192          CMPC    R1,(R2),OUTQUAL             ;CHECK FOR VALID QUAL
                        0C    13 005A   193          BEQL    14$                         ;BR IF OK
                     50 8ED0    005C   194          POPL    R0                          ;RESTORE TERMINATION CHARACTER
                                005F   195          STATUS  IVQUAL                      ;SET ILLEGAL QUALIFIER CODE
                        14    11 0066   196          BRB     15$                         ;
                     50 8ED0    0068   197 14$:     POPL    R0                          ;RESTORE TERMINATION CHARACTER
                  50    3D    91 006B   198          CMPB    #^A/=/,R0                   ;EQUAL SIGN TERMINATOR?
                        C4    13 006E   199          BEQL    10$                         ;IF EQL YES
                  50    3A    91 0070   200          CMPB    #^A/:/,R0                   ;COLON TERMINATOR?
                        BF    13 0073   201          BEQL    10$                         ;IF EQL YES
                                0075   202          STATUS  IVVALU                      ;SET INVALID VALUE SYNTAX
                   007D    31    007C   203 15$:     BRW     80$                         ;
```

```
                                                C 15
                     007F    204
                     007F    205 ;
                     007F    206 ; FILE SPECIFICATIONS PARSED - PARSE SYMBOL DEFINITIONS
                     007F    207 ;
                     007F    208 ; IF THE FILESPEC WAS FOLLOWED BY A SPACE, THAT SPACE MAY HAVE BEEN THROWN
                     007F    209 ; AWAY IF THE FIRST CHARACTER IN P1 MAKES IT INSIGNIFICANT.
                     007F    210 ;
58   04 AE    9E     007F    211 20$:    MOVAB   4(SP),R8                ;GET ADDRESS OF SYMBOL DESCRIPTOR ST
      FF7A'   30     0083    212         BSBW    DCL$SETNBLK             ;IGNORE BLANKS AFTER FILESPEC
      FF77'   30     0086    213 30$:    BSBW    DCL$MARK                ;MARK POSITION OF FIRST NON-BLANK
      FF74'   30     0089    214 40$:    BSBW    DCL$MOVCHAR             ;COPY A CHARACTER FROM INPUT BUUFER
        04    E0     008C    215         BBS     #WRK_V_QUOTE,-          ;LOOP IF IN A QUOTED STRING
  F8 F0 AA            008E    216                 WRK_W_FLAGS(R10),40$    ;
        05    13     0091    217         BEQL    45$                     ;BR IF END OF LINE
   50   20    91     0093    218         CMPB    #^A' ',R0               ;IS THIS A TERMINATOR
        F1    12     0096    219         BNEQ    40$                     ;BR IF NO - KEEP LOOKING FOR TERMINA
      FF65'   30     0098    220 45$:    BSBW    DCL$MARKEDTOKEN         ;GET DESCRIPTOR OF PARAMETER
        51    D7     009B    221         DECL    R1                      ;REMOVE COUNT FOR TERMINATOR
        18    13     009D    222         BEQL    60$                     ;IF NULL STRING - NO MORE SYMBOLS
   62   22    91     009F    223         CMPB    #^A/''/,(R2)            ;SYMBOL START WITH A QUOTE
        03    12     00A2    224         BNEQ    50$                     ;IF NO - LEAVE THE SYMBOL ALONE
      FF59'   30     00A4    225         BSBW    DCL$COMPRESS            ;ELSE REMOVE THE QUOTE PAIRS
   88   51    7D     00A7    226 50$:    MOVQ    R1,(R8)+                ;STORE SYMBOL DESCRIPTOR
D8 6E    08    F3     00AA    227         AOBLEQ  #SYMBOLS,(SP),30$       ;ANY MORE SYMBOL DEFINITIONS ALLOWED
                     00AE    228         STATUS  DEFOVF                  ;SET SYMBOL DEFINITION OVERFLOW
        45    11     00B5    229         BRB     80$                     ;
                     00B7    230
                     00B7    231 ;
                     00B7    232 ; RUN DOWN ANY IMAGE CURRENTLY RUNNING
                     00B7    233 ;
      BA AA    DD     00B7    234 60$:    PUSHL   WRK_L_RSLNXT(R10)       ;SAVE POINTER INTO WRK AREA
       FF43'   30     00BA    235         BSBW    DCL$RUNDOWN             ;RUN DOWN IMAGE AND INDIRECT LEVELS
50    BA AA    8E     C3     00BD    236         SUBL3   (SP)+,WRK_L_RSLNXT(R10),R0  ;CALCULATE LENGTH OF STACK SHIFT
      68 AE    50     C0     00C2    237         ADDL    R0,<<SYMBOLS*8>+4+<9*4>>(SP)  ;RELOCATE SAVED WRK_L_RSLNXT
      6C AE    50     C0     00C6    238         ADDL    R0,<<SYMBOLS*8>+4+<10*4>>(SP) ;RELOCATE SAVED WRK_L_RSLEND
      70 AE    50     C0     00CA    239         ADDL    R0,<<SYMBOLS*8>+4+<11*4>>(SP) ;RELOCATE SAVED WRK_L_EXPANDPTR
      74 AE    50     C0     00CE    240         ADDL    R0,<<SYMBOLS*8>+4+<12*4>>(SP) ;RELOCATE SAVED WRK_L_MARKPTR
                     00D2    241
                     00D2    242 ;
                     00D2    243 ; STACK COMMAND PROCEDURE
                     00D2    244 ;
      68 AE    D0     00D2    245         MOVL    <<SYMBOLS*8>+4+<9*4>>(SP),-     ;RETRIEVE ADDRESS OF DESCRIPTORS
      BA AA            00D5    246                 WRK_L_RSLNXT(R10)       ;
       FF26'   30     00D7    247         BSBW    DCL$GETDVAL             ;GET INPUT FILE DESCRIPTOR VALUES
   7E   51    7D     00DA    248         MOVQ    R1,-(SP)                ;SAVE INPUT FILESPEC
        54    D4     00DD    249         CLRL    R4                      ;ASSUME NO OUTPUT FILESPEC
       FF1E'   30     00DF    250         BSBW    DCL$GETDVAL             ;GET OUTPUT FILESPEC
      03 50    E9     00E2    251         BLBC    R0,65$                  ;IF NONE, PASS IN NULL FILESPEC
   54   51    7D     00E5    252         MOVQ    R1,R4                   ;SET OUTPUT FILESPEC ARGUMENT
   52   8E    7D     00E8    253 65$:    MOVQ    (SP)+,R2                ;SET INPUT FILESPEC ARGUMENT
        51    D4     00EB    254         CLRL    R1                      ;SIGNAL ALL RMS ERRORS
      0049    30     00ED    255         BSBW    DCL$PUSHPROC            ;PUSH PROCEDURE ONTO INDIRECT STACK
      09 50    E9     00F0    256         BLBC    R0,80$                  ;BRANCH IF ERROR DETECTED
                     00F3    257
                     00F3    258 ;
                     00F3    259 ; CREATE SYMBOLS P1-P8
                     00F3    260 ;
```

```
                     00F3   261          MOVL    (SP),R6                         ;GET NUMBER OF SYMBOL DEFINITIONS
                     00F6   262          MOVAB   4(SP),R8                        ;GET ADDRESS OF VALUE DESCRIPTORS
                     00FA   263          BSBB    DCL$DEFINE_P1_TO_P8             ;DEFINE P1 THROUGH P8
                     00FC   264
                     00FC   265 ;
                     00FC   266 ; RESTORE THE STACK AND PREPARE TO EXIT
                     00FC   267 ;
                     00FC   268 80$:     MOVAB   <SYMBOLS*8+4>(SP),SP            ;DEALLOCATE SYMBOL DESCRIPTOR STORAG
                     0100   269          PUSHL   R0                              ;SAVE FINAL STATUS
                     0102   270          BRW     STKXIT                          ;
```

```
                     0105   272          .SBTTL  DEFINE SYMBOLS P1-P8
                     0105   273 ;+
                     0105   274 ; DCL$DEFINE_P1_TO_P8 - DEFINE SYMBOLS P1-P8
                     0105   275 ;
                     0105   276 ; THIS ROUTINE IS CALLED TO DEFINE THE LOCAL SYMBOLS P1-P8.
                     0105   277 ;
                     0105   278 ; INPUTS:
                     0105   279 ;
                     0105   280 ;       R6 = NUMBER OF SYMBOLS THAT HAVE ASSIGNED VALUES
                     0105   281 ;       R8 = ADDRESS OF LIST OF Pn VALUE DESCRIPTORS
                     0105   282 ;
                     0105   283 ; OUTPUTS:
                     0105   284 ;
                     0105   285 ;       R1-R8 TRASHED
                     0105   286 ;
                     0105   287 ;-
                     0105   288
                     0105   289 DCL$DEFINE_P1_TO_P8::
7E   3050 8F   3C    0105   290          MOVZWL  #^A/P0/,-(SP)           ;CREATE PROTOTYPE OF GENERATED SYMBO
     57   08   D0    010A   291          MOVL    #SYMBOLS,R7             ;SET NUMBER OF SYMBOLS TO GENERATE
          51   D4    010D   292 10$:     CLRL    R1                      ;ASSUME NO MORE SYMBOLS DEFINED
          56   D7    010F   293          DECL    R6                      ;ARE THERE ANY MORE TO DEFINE
          03   19    0111   294          BLSS    20$                     ;BR IF NO - DEFINE AS NULL STRING
     51   88   7D    0113   295          MOVQ    (R8)+,R1                ;GET VALUE DESCRIPTOR
       01 AE   96    0116   296 20$:     INCB    1(SP)                   ;INCREMENT SYMBOL NUMBER
     53   02   D0    0119   297          MOVL    #2,R3                   ;SET LENGTH OF SYMBOL NAME
     54   6E   9E    011C   298          MOVAB   (SP),R4                 ;SET ADDRESS OF SYMBOL NAME
  55   38 AB   9E    011F   299          MOVAB   PRC_Q_LOCAL(R11),R5     ;GET ADDRESS OF LOCAL SYMBOL TABLE L
     50   00   D0    0123   300          MOVL    #SYM_K_STRING,R0        ;SET SYMBOL TYPE IS STRING
        FED7'  30    0126   301          BSBW    DCL$ALLOCSYM            ;ALLOCATE AND INSERT SYMBOL TABLE EN
       0A 50   E9    0129   302          BLBC    R0,90$                  ;IF LBC ALLOCATION FAILURE
       DE 57   F5    012C   303          SOBGTR  R7,10$                  ;ANY MORE SYMBOL TO PROCESS?
                     012F   304          STATUS  NORMAL                  ;SET NORMAL COMPLETION STATUS
     8E   D5   0136   305 90$:     TSTL    (SP)+                   ;RESTORE THE STACK
          05   0138   306          RSB                             ;RETURN
```

```
                     0139   308          .SBTTL  PUSH PROCEDURE ONTO INDIRECT STACK
                     0139   309 ;+
                     0139   310 ; DCL$PUSHPROC - PUSH PROCEDURE ONTO INDIRECT STACK
                     0139   311 ;
                     0139   312 ; THIS ROUTINE IS CALLED TO INITIALIZE A NEW INDIRECT FRAME
                     0139   313 ; ON THE INDIRECT PROCEDURE STACK.
                     0139   314 ;
                     0139   315 ; INPUTS:
                     0139   316 ;
                     0139   317 ;       R1 = 1 IF RMS ERRORS SHOULD NOT BE SIGNALED, ELSE 0
                     0139   318 ;       R2/R3 = DESCRIPTOR OF INPUT FILESPEC
                     0139   319 ;       R4/R5 = DESCRIPTOR OF OUTPUT FILESPEC
                     0139   320 ;       R11 = ADDRESS OF PROCESS WORK AREA
                     0139   321 ;
                     0139   322 ; OUTPUTS:
                     0139   323 ;
                     0139   324 ;       R0 = STATUS (NOT SIGNALED)
                     0139   325 ;
                     0139   326 ;-
                     0139   327
                     0139   328 DCL$PUSHPROC::
      13FE 8F    BB  0139   329          PUSHR   #^M<R1,R2,R3,R4,R5,R6,R7,R8,R9,AP>
56    00A0 CB    D0  013D   330          MOVL    PRC_L_STACKPT(R11),R6   ;GET CURRENT INDIRECT STACK POINTER
   58    8C A6    9E  0142   331          MOVAB   -IDF_K_LENGTH(R6),R8    ;CALCULATE NEW INDIRECT STACK POINTER
00A4 CB    58    D1  0146   332          CMPL    R8,PRC_L_STACKLM(R11)   ;INDIRECT STACK OVERFLOW?
         0C    1E  014B   333          BGEQU   2$                      ;BRANCH IF OK
                     014D   334          STATUS  STKOVF                  ;SET INDIRECT STACK OVERFLOW
      13FE 8F    BA  0154   335 80$:     POPR    #^M<R1,R2,R3,R4,R5,R6,R7,R8,R9,AP>
         05    0158   336          RSB
                     0159   337
                     0159   338 ;
                     0159   339 ; ALLOCATE ROOM IN SYMBOL TABLE FOR RESULTANT NAME STRING PRIOR TO CLOSING
                     0159   340 ; CURRENT INDIRECT FRAME.
                     0159   341 ;
                     0159   342
51   00000100 8F    D0  0159   343 2$:      MOVL    #<<<NAM$C_MAXRSS+1>+7>&^C<7>>,R1 ;SET MAXIMUM SIZE OF
                     0160   344                                             ;RESULTANT NAME STRING
         FE9D'   30  0160   345          BSBW    DCL$ALLDYNMEM           ;ALLOCATE ROOM IN THE SYMBOL TABLE
      09 50    E8  0163   346          BLBS    R0,3$                   ;BR IF NO ALLOCATION ERROR
                     0166   347          STATUS  SYMOVF                  ;INDICATE NO MORE ROOM IN SYM TAB
         E5    11  016D   348          BRB     80$                     ;JUST EXIT
   59    52    D0  016F   349 3$:      MOVL    R2,R9                   ;SAVE ADDRESS OF ALLOCATED BLOCK
                     0172   350
                     0172   351 ;
                     0172   352 ; THE NEW INDIRECT FILE FRAME IS FORMED ON THE STACK AND LINKED TO ANY
                     0172   353 ; PREVIOUS FRAMES.  THE STACK OVERFLOW CHECK HAS BEENY PERFORMED AT THIS POINT
                     0172   354 ;
                     0172   355
      5C AB    D6  0172   356          INCL    PRC_L_INDEPTH(R11)      ; SET NEW INSTACK STACK DEPTH
      7C AB    D6  0175   357          INCL    PRC_L_INDCLOCK(R11)     ; INCREMENT TOTAL STACKS OR UNSTACKS
00A0 CB    58    D0  0178   358          MOVL    R8,PRC_L_STACKPT(R11)   ; ALLOCATE NEW STACK FRAME
56   00BC CB    D0  017D   359          MOVL    PRC_L_IDFLNK(R11),R6    ; GET ADDRESS OF CURRENT INDIRECT FRAME
   68    56    D0  0182   360          MOVL    R6,IDF_L_LNK(R8)        ; LINK NEW FRAME INTO TOP OF
00BC CB    68    DF  0185   361          MOVAL   IDF_L_LNK(R8), -        ; INDIRECT FILE FRAME LIST
                     018A   362                  PRC_L_IDFLNK(R11)
                     018A   363 ;
                     018A   364 ; R6 = Pointer to current stack frame
```

```
                                 018A    365 ; R8 = Pointer to new stack frame
                                 018A    366 ;
      5C    1C AB    D0          018A    367         MOVL    PRC_L_INDFAB(R11),AP    ;GET ADDRESS OF INDIRECT FAB
   10 A6    38 AB    7D          018E    368         MOVQ    PRC_Q_LOCAL(R11),IDF_Q_LOCAL(R6) ;SAVE LOCAL SYMBOL TABLE LISTHEAD
   18 A6    30 AB    7D          0193    369         MOVQ    PRC_Q_LABEL(R11),IDF_Q_LABEL(R6) ;SAVE LABEL SYMBOL TABLE LISTHEAD
   06 A6    6A AB    B0          0198    370         MOVW    PRC_W_ONLEVEL(R11),IDF_W_ONLEVEL(R6) ;SAVE ON ERROR LEVEL NUMBER
   08 A6    6C AB    D0          019D    371         MOVL    PRC_L_ONERROR(R11),IDF_L_ONERROR(R6) ;SAVE ON ERROR COMMAND TEXT
      50    38 AB    9E          01A2    372         MOVAB   PRC_Q_LOCAL(R11),R0     ;GET ADDRESS OF LOCAL TABLE LISTHEAD
         60    50    D0          01A6    373         MOVL    R0,(R0)                 ;SET ADDRESS OF LISTHEAD AS FORWARD LINK
         80    80    D0          01A9    374         MOVL    (R0)+,(R0)+             ;SET ADDRESS OF LISTHEAD AS BACKWARD LINK
      50    30 AB    9E          01AC    375         MOVAB   PRC_Q_LABEL(R11),R0     ;GET ADDRESS OF LABEL TABLE LISTHEAD
         60    50    D0          01B0    376         MOVL    R0,(R0)                 ;SET ADDRESS OF LISTHEAD AS FORWARD LINK
         80    80    D0          01B3    377         MOVL    (R0)+,(R0)+             ;SET ADDRESS OF LISTHEAD AS BACKWARD LINK
           6C AB    D4          01B6    378         CLRL    PRC_L_ONERROR(R11)      ;CLEAR ADDRESS OF ON ERROR COMMAND TEXT
   6A AB    0202 8F    B0          01B9    379         MOVW    #2@8!2,PRC_W_ONLEVEL(R11) ;RESET ON ERROR LEVEL TO ERROR
   60 A6    00B8 CB    D0          01BF    380         MOVL    PRC_L_ONCTLY(R11),IDF_L_ONCTLY(R6) ;SAVE ON CONTROL Y COMMAND
               07    13          01C5    381         BEQL    5$                      ;BR IF THERE WAS NONE
00B8 CB    0000'CF    9E          01C7    382         MOVAB   W^DCL$T_DEFONTXT,PRC_L_ONCTLY(R11) ;SET DEFUALT FOR NEXT LEVEL
      5E AB    01    B0          01CE    383 5$:     MOVW    #1@IDF_V_INPOPN,IDF_W_FLAG(R8) ;SET INPUT FILE OPEN FLAG
                                 01D2    384                                     ;ASSUME FILE IS OPENED LOCALLY
           64 A8    D4          01D2    385         CLRL    IDF_L_SEARCHCTX(R8)     ;INITIALIZE F$SEARCH CONTEXT LIST
                                 01D5    386
                                 01D5    387 ;
                                 01D5    388 ; CLOSE INPUT FILE FROM PREVIOUS INDIRECT LEVEL AND REMEMBER THE CURRENT
                                 01D5    389 ; POSITION IN THE FILE, SO THAT ON RETURN, WE CAN RESET THE POSITION.
                                 01D5    390 ;
      52    14 AB    D0          01D5    391         MOVL    PRC_L_INDINPRAB(R11),R2 ;SET CURRENT INDIRECT RAB POINTER
      08 AB    52    D1          01D9    392         CMPL    R2,PRC_L_INPRAB(R11)    ;IS THIS THE PRIMARY INPUT STREAM?
               31    13          01DD    393         BEQL    7$                      ;BR IF YES-THAT NEVER GETS CLOSED
   1E 18 A2    1C    E1          01DF    394         BBC     #DEV$V_RND,RAB$L_CTX(R2),6$ ;SKIP IF NOT A DISK FILE
      58 A6    01    AE          01E4    395         MNEGW   #1,IDF_W_INPRFA(R6)     ;ASSUME END OF FILE
                                 01E8    396         $FIND   RAB=(R2)                ;GET THE CURRENT RECORD POSITION (IT
    0000'8F    50    B1          01F1    397         CMPW    R0,#RMS$_EOF&^XFFFF     ;MAY HAVE BEEN ADVANCED BY AN INDIRECT
               0A    13          01F6    398         BEQL    6$                      ;ACCESSOR SINCE OUR LAST $GET).
   58 A6    10 A2    D0          01F8    399         MOVL    RAB$W_RFA(R2),IDF_W_INPRFA(R6) ;SAVE RECORD POSITION IN FILE
   5C A6    14 A2    B0          01FD    400         MOVW    RAB$W_RFA+4(R2),IDF_W_INPRFA+4(R6)
   02 AC    04 A6    B0          0202    401 6$:     MOVW    IDF_W_INPIFI(R6),FAB$W_IFI(AP) ;SET INTERNAL FILE IDENTIFICATION
                                 0207    402         $CLOSE  FAB=(AP)                ;
                                 0210    403
                                 0210    404 ;
                                 0210    405 ; OPEN INPUT PROCEDURE FILE
                                 0210    406 ;
         04 A8    B4          0210    407 7$:     CLRW    IDF_W_INPIFI(R8)        ;CLEAR INPUT FILE INTERNAL INDEX
         02 AC    B4          0213    408         CLRW    FAB$W_IFI(AP)           ;CLEAR INTERNAL FILE INDEX
                                 0216    409
      51    04 AE    7D          0216    410         MOVQ    4(SP),R1                ;GET INPUT FILESPEC (R2/R3 ON ENTRY)
      34 AC    51    90          021A    411         MOVB    R1,FAB$B_FNS(AP)        ;SET SIZE OF FILE NAME STRING
      2C AC    52    D0          021E    412         MOVL    R2,FAB$L_FNA(AP)        ;SET ADDRESS OF FILE NAME STRING
      35 AC    04    90          0222    413         MOVB    #4,FAB$B_DNS(AP)        ;SET SIZE OF DEFAULT NAME STRING
30 AC    FDD6 CF    9E          0226    414         MOVAB   INPFILE,FAB$L_DNA(AP)   ;SET ADDRESS OF DEFAULT NAME STRING
                                 022C    415
      57    28 AC    D0          022C    416         MOVL    FAB$L_NAM(AP),R7        ;GET ADDRESS OF INDIRECT NAME BLOCK
                                 0230    417
      69    FF 8F    90          0230    418         MOVB    #255,(R9)               ;STORE LENGTH OF BUFFER (IN SYM TAB)
      68 A8    59    D0          0234    419         MOVL    R9,IDF_L_FILENAME(R8)   ;STORE ADDRESS OF BUFFER (IN SYM TAB)
                                 0238    420         ASSUME  NAM$B_RSS EQ NAM$B_RSS+1
         FF 8F    9B          0238    421         MOVZBW  #NAM$C_MAXRSS,-         ;SAVE THE SIZE OF THE BUFFER
```

```
                02 A7          023B   422                      NAM$B_RSS(R7)              ;(NOTE, NOT THE ALLOCATED SIZE)
        04 A7   01 A9    9E    023D   423          MOVAB       1(R9),NAM$L_RSA(R7)        ;SAVE THE ADDRESS OF THE BUFFER
                               0242   424          ASSUME      NAM$B_RSL EQ NAM$B_RSS+1
                FF 8F    9B    0242   425          MOVZBW      #NAM$C_MAXRSS,-            ;SET UP EXPANDED STRING TOO
                0A A7          0245   426                      NAM$B_ESS(R7)              ;
                04 A7    D0    0247   427          MOVL        NAM$L_RSA(R7),-            ;
                0C A7          024A   428                      NAM$L_ESA(R7)              ;
                               024C   429
           08 A7   01    90    024C   430          MOVB        #NAM$M_PWD,NAM$B_NOP(R7);DISABLE PASSWORD MASKING
        16 AC   80 8F    90    0250   431          MOVB        #FAB$M_EXE,FAB$B_FAC(AP);SET FILE ACCESS TYPE
          000C0000 8F    D0    0255   432          MOVL        #FAB$M_INP!FAB$M_PPF,-     ;SET FILE OPEN OPTIONS
                04 AC          025B   433                      FAB$L_FOP(AP)              ;
            1E AC   04    90    025D   434          MOVB        #FAB$M_PRN,FAB$B_RAT(AP);SET CARRIAGE CONTROL
            1F AC   03    90    0261   435          MOVB        #FAB$C_VFC,FAB$B_RFM(AP);SET VERT. FORMS CONTROL
                17 AC   94    0265   436          CLRB        FAB$B_SHR(AP)              ;CLEAR FILE SHARING OPTIONS
                50   5C   D0    0268   437          MOVL        AP,R0                      ;ADDRESS OF FAB
                               026B   438
                51   04   D0    026B   439          MOVL        #4,R1                      ;ASSUME OPEN WITH ERROR REPORTING
                  03 6E   E9    026E   440          BLBC        (SP),8$                    ;IF ERROR REPORTING DISABLED,
                51   02   C8    0271   441          BISL        #2,R1                      ;DO OPEN WITHOUT ERROR REPORTING
                   FD89'  30    0274   442 8$:      BSBW        DCL$OPEN_CREATE            ;OPEN INDIRECT INPUT FILE
                               0277   443          CLRBIT      NAM$V_PWD,NAM$B_NOP(R7) ;UNCONDITIONALLY REENABLE PASSWORD MASKINNG
                  3D 50   E9    027B   444          BLBC        R0,9$                      ;IF LBC OPEN FAILURE
                               027E   445
        04 A8   02 AC   B0    027E   446          MOVW        FAB$W_IFI(AP),IDF_W_INPIFI(R8) ;SAVE INPUT FILE INTERNAL INDEX
         56   00F4 CC   9E    0283   447          MOVAB       PRD_G_ALTINPRAB(AP),R6  ;GET ALTERNATE INPUT RAB
        18 A6   40 AC   D0    0288   448          MOVL        FAB$L_DEV(AP),RAB$L_CTX(R6) ;SAVE DEVICE CHARACTERISTICS
        0C A8   18 A6   D0    028D   449          MOVL        RAB$L_CTX(R6),IDF_L_INPRABCTX(R8) ;AND A COPY IN THE STACK FRAME
                   11   E1    0292   450          BBC         #NAM$V_NODE,-              ;BRANCH IF NOT A REMOTE OPEN
             04 34 A7          0294   451                      NAM$L_FNB(R7),10$          ;
                               0297   452          SETBIT      IDF_V_REMOTE,-             ;SET REMOTE OPEN FLAG
                               0297   453                      IDF_W_FLAG(R8)             ;
                               029B   454
                               029B   455 ;
                               029B   456 ; GET THE DEVICE NAME.  PROPAGATE CONCEALED ATTRIBUTES.
                               029B   457 ;
                               029B   458 10$:     CLRBIT      IDF_V_INPCCL,IDF_B_OUTFLAGS(R8) ;CLEAR CONCEALED BIT IN IDF
                      0C   E1    029F   459          BBC         #NAM$V_CNCL_DEV,-          ;IS DEVICE CONCEALED?
             04 34 A7          02A1   460                      NAM$L_FNB(R7),11$          ;
                               02A4   461          SETBIT      IDF_V_INPCCL,IDF_B_OUTFLAGS(R8) ;SET CONCEALED BIT IN IDF
                               02A8   462          ASSUME      IDF_W_INPFID EQ IDF_T_INPDVI+16
                               02A8   463          ASSUME      IDF_W_INPDID EQ IDF_W_INPFID+6
        14 A7   1C    28       02A8   464 11$:     MOVC        #28,NAM$T_DVI(R7),-        ;COPY FILE INFORMATION
             3C A8               02AC   465                      IDF_T_INPDVI(R8)           ;INTO INDIRECT STACK FRAME
        3C A6   5C    D0       02AE   466          MOVL        AP,RAB$L_FAB(R6)           ;LINK FAB TO RAB
                               02B2   467          $CONNECT RAB=(R6)                      ;CONNECT TO NEW INPUT
                4A 50   E9    02BB   468 9$:      BLBC        R0,50$                     ;IF LBC CONNECT FAILURE
                               02BE   469          CLRBIT      RAB$V_PPF_IND,RAB$W_ISI(R6) ;MAKE SURE INDIRECT FLAG IS CLEAR
        14 AB   56    D0       02C3   470          MOVL        R6,PRC_L_INDINPRAB(R11)    ;SET INDIRECT INPUT RAB
                0272  30       02C7   471          BSBW        SAV_EXE_ONLY               ;SAVE VER. FLAGS IF EXE-ONLY PROCEDURE.
                               02CA   472
                               02CA   473 ;
                               02CA   474 ; CREATE OUTPUT FILE, IF SPECIFIED
                               02CA   475 ;
        51   0C AE   7D       02CA   476          MOVQ        12(SP),R1                  ;GET OUTPUT FILESPEC (R4/R5 ON ENTRY)
        7E   03 A7   9A       02CE   477          MOVZBL      NAM$B_RSL(R7),-(SP)        ;SAVE LENGTH OF INPUT FILE NAME
           56   68   D0       02D2   478          MOVL        IDF_L_LNK(R8),R6           ;SET ADDRESS OF DEFAULT SYS$OUTPUT INFO
```

```
                 FD28'   30   02D5    479          BSBW    DCL$OPEN_OUTPUT          ;CONDITIONALLY OPEN SYS$OUTPUT
                   51 8ED0   02D8    480          POPL    R1                       ;RESTORE LENGTH OF INPUT FILE NAME
                2A 50   E9   02DB    481          BLBC    R0,50$                   ;RETURN ANY ERRORS
                             02DE    482
                             02DE    483 ;
                             02DE    484 ; DEALLOCATE UNUSED BUFFER.
                             02DE    485 ;
             50   68 A8   D0   02DE    486          MOVL    IDF_L_FILENAME(R8),R0    ;GET ADDRESS OF BUFFER
                60   51   90   02E2    487          MOVB    R1,(R0)                  ;SAVE FILE NAME LENGTH IN FIRST BYTE OF BUFF
                51   08   C0   02E5    488          ADDL    #8,R1                    ;ROUND UP SIZE TO QUADWORD BOUNDARY(INCLUDE
                51   07   CA   02E8    489          BICL    #7,R1                    ;TRUNCATE DOWN SIZE TO QUADWORD BOUNDARY
                50   51   C0   02EB    490          ADDL    R1,R0                    ;CALCULATE ADDRESS OF UNUSED BUFFER
     51   00000100 8F   51   C3   02EE    491          SUBL3   R1,#<<<NAM$C_MAXRSS+1>+7>&^C<7>>,R1 ;CALCULATE SIZE OF UNUSED BUFFER
                   03   13   02F6    492          BEQL    40$                      ;DON'T DEALLOCATE IF NO UNUSED BUFFER
                 FD05'   30   02F8    493          BSBW    DCL$DEADYNMEM            ;DEALLOCATE UNUSED BUFFER
                             02FB    494
                             02FB    495 ;
                             02FB    496 ; CREATE LOGICAL NAMES FOR 'INPUT' AND 'OUTPUT' AND EXIT WITH SUCCESS.
                             02FB    497 ;
                 FD02'   30   02FB    498 40$:     BSBW    DCL$CREATE_IO            ;CREATE LOGICAL NAMES FOR 'INPUT' AND 'OUTPU
                             02FE    499          STATUS  NORMAL                   ;
                 FE4C    31   0305    500          BRW     80$                      ;EXIT WITH SUCCESS
                             0308    501
                             0308    502 ;
                             0308    503 ; OPEN, CONNECT, OR SYMBOL ALLOCATION FAILURE
                             0308    504 ;
                02 A7    B4   0308    505 50$:     CLRW    NAM$B_RSS(R7)            ;INVALIDATE RESULTANT STRINGS
                0A A7    B4   030B    506          CLRW    NAM$B_ESS(R7)            ;INVALIDATE EXPANDED STRINGS
                   50    DD   030E    507          PUSHL   R0                       ;SAVE ERROR/STATUS VALUE
                 0080    30   0310    508          BSBW    UNSTACK                  ;UNSTACK PREVIOUS INDIRECT FILE
                   50 8ED0   0313    509          POPL    R0                       ;RETRIEVE ERROR/STATUS VALUE
                 FE3B    31   0316    510          BRW     80$                      ;EXIT WITH STATUS
```

```
                                                                J 15
                     0319   512          .SBTTL  UNSTACK INDIRECT FILE SPECIFICATION
                     0319   513 ;+
                     0319   514 ; DCL$UNSTACK - UNSTACK INDIRECT FILE SPECIFICATION
                     0319   515 ;
                     0319   516 ; THIS ROUTINE IS CALLED TO CLOSE THE CURRENT INDIRECT FILE AND TO UNSTACK THE
                     0319   517 ; PREVIOUS SPECIFICATION.
                     0319   518 ;
                     0319   519 ; INPUTS:
                     0319   520 ;
                     0319   521 ;       NONE.
                     0319   522 ;
                     0319   523 ; OUTPUTS:
                     0319   524 ;
                     0319   525 ;       THE CURRENT INDIRECT FILE IS CLOSED AND ALL LOCAL SYMBOLS FOR THE LEVEL
                     0319   526 ;       ARE DEALLOCATED. THE PREVIOUS INDIRECT FILE IS THEN UNSTACKED AND REOPENED.
                     0319   527 ;
                     0319   528 ;       R0 LOW BIT CLEAR INDICATES UNSUCCESSFUL COMPLETION.
                     0319   529 ;
                     0319   530 ;       R0 LOW BIT SET INDICATES SUCCESSFUL COMPLETION.
                     0319   531 ;
                     0319   532 ;       ALL ERRORS ARE SIGNALED BEFORE RETURNING TO CALLER.
                     0319   533 ;-
                     0319   534
                     0319   535 DCL$UNSTACK::                            ;UNSTACK INDIRECT FILE SPECIFICATION
                     0319   536          $DELLOG_S       TBLFLG=#LOG$C_PROCESS,- ;DELETE ANY USER DEFINED
                     0319   537                          ACMODE=#PSL$C_USER      ;LOGICAL NAMES.
           48   10   0326   538          BSBB    SETIND                  ;SETUP INDIRECT PROCESSING
           00'  DD   0328   539          PUSHL   S^#SS$_NORMAL           ;ASSUME NORMAL COMPLETION
13 68 AB   04   E5   032A   540          BBCC    #PRC_V_GOTO,PRC_W_FLAGS(R11),10$ ;IF CLR, NO GOTO IN PROGRESS
        FCCE'  30   032F   541          BSBW    DCL$DEALGOTO            ;DEALLOCATE GOTO SYMBOL
                     0332   542          STATUS  USGOTO                  ;SET UNSATISFIED GOTO STATUS
        6E   50   D0   0339   543          MOVL    R0,(SP)                 ;SET COMPLETION STATUS
                     033C   544          ERRMSG                          ;OUTPUT ERROR MESSAGE
           FCBE'  30   033F   545          BSBW    DCL$SET_STATUS          ;GIVE ERROR HANDLER'S A CHANCE
             4F   10   0342   546 10$:     BSBB    UNSTACK                 ;UNSTACK TO PREVIOUS LEVEL
        F895 CA   9E   0344   547          MOVAB   WRK_G_INPBUF-1(R10),-   ;SET STARTING ADDRESS OF INPUT
        F48E CA        0348   548                  WRK_L_CHARPTR(R10)      ;BUFFER AS LAST BYTE FETCHED
        F896 CA   94   034B   549          CLRB    WRK_G_INPBUF(R10)       ;SET EOL AS NEXT BYTE TO FETCH
        11FF 8F   BA   034F   550 STKXIT: POPR    #^M<R0,R1,R2,R3,R4,R5,R6,R7,R8,AP> ;RESTORE REGISTERS
                     0353   551                                          ; R0=STATUS, R1=ORIGINAL FLAGS
          BA AA 8ED0 0353   552          POPL    WRK_L_RSLNXT(R10)       ;RESTORE TOKEN DESCRIPTORS BACK TO
          B6 AA 8ED0 0357   553          POPL    WRK_L_RSLEND(R10)       ;WHERE THEY WERE WHEN WE STARTED
        F486 CA 8ED0 035B   554          POPL    WRK_L_EXPANDPTR(R10)    ;RESTORE EXPANSION BUFFER POINTER
        F48A CA 8ED0 0360   555          POPL    WRK_L_MARKPTR(R10)      ;RESTORE MARKER POINTER
                     0365   556          ENABLE                          ;ENABLE CONTROL Y/C AST'S
04 51    01   E0   0367   557          BBS     #WRK_V_COMMAND,R1,10$   ;BR IF COMMAND WAS SET
F0 AA    02   AA   036B   558          BICW    #WRK_M_COMMAND,WRK_W_FLAGS(R10) ;CLEAR COMMAND IN PROGRESS
             05   036F   559 10$:     RSB                             ;
                     0370   560
                     0370   561 ;
                     0370   562 ; SETIND - SETUP INDIRECT
                     0370   563 ;
                     0370   564 ; SAVE THE NON-VOLATILE REGISTERS AND THE COMMAND WORK FLAGS, THEN SET COMMAND
                     0370   565 ;
                     0370   566
           01   BA   0370   567 SETIND: POPR    #^M<R0>                 ;GET RETURN PC
                     0372   568          DISABLE                         ;DISABLE CONTROL Y/C AST'S
```

```
                F48A CA    DD  0378    569         PUSHL   WRK_L_MARKPTR(R10)      ;SAVE CURRENT MARKER POINTER
                F486 CA    DD  037C    570         PUSHL   WRK_L_EXPANDPTR(R10)    ;SAVE CURRENT EXPANSION BUFFER POINTER
                  B6 AA    DD  0380    571         PUSHL   WRK_L_RSLEND(R10)       ;SAVE CURRENT ENDING TOKEN ADDRESS
                  BA AA    DD  0383    572         PUSHL   WRK_L_RSLNXT(R10)       ;SAVE CURRENT POSITION IN TOKEN ARRAY
                11FC 8F    BB  0386    573         PUSHR   #^M<R2,R3,R4,R5,R6,R7,R8,AP> ;SAVE REGISTERS
                  F0 AA    DD  038A    574         PUSHL   WRK_W_FLAGS(R10)        ;SAVE PREVIOUS COMMAND FLAGS
                                0390    575         SETBIT  WRK_V_COMMAND,WRK_W_FLAGS(R10) ;SET COMMAND IN PROGRESS
                      60    17  0394    576         JMP     (R0)                    ;RETURN TO CALLER
```

```
                     0393    578          .SBTTL  UNSTACK NEXT INDIRECT FILE
                     0393    579 ;---
                     0393    580 ;
                     0393    581 ; UNSTACK - UNSTACK NEXT INDIRECT FILE
                     0393    582 ;
                     0393    583 ; THIS ROUTINE IS CALLED TO CLOSE THE CURRENT INDIRECT FILE AND UNSTACK THE
                     0393    584 ; CONTEXT INFORMATION FOR THE PREVIOUS LEVEL INDIRECT FILE.
                     0393    585 ;
                     0393    586 ; INPUTS:
                     0393    587 ;
                     0393    588 ;       R11 = ADDRESS OF PROCESS WORK AREA
                     0393    589 ;
                     0393    590 ; OUTPUTS:
                     0393    591 ;
                     0393    592 ;       NONE
                     0393    593 ;
                     0393    594 ;       R0-R8,AP ARE DESTROYED.
                     0393    595 ;---
                     0393    596
                     0393    597 UNSTACK:                                        ;UNSTACK INDIRECT FILE
   5C    1C AB   D0  0393    598          MOVL    PRC_L_INDFAB(R11),AP           ;GET ADDRESS OF SCRATCH FAB
 58    00BC CB   D0  0397    599          MOVL    PRC_L_IDFLNK(R11),R8           ;GET ADDRESS OF CURRENT INDIRECT FRAME
          58   DD  039C    600          PUSHL   R8                             ;SAVE THAT ADDRESS
                     039E    601
                     039E    602 ;
                     039E    603 ; CLOSE CURRENT INPUT PROCEDURE FILE
                     039E    604 ;
 02 AC    04 A8   B0  039E    605          MOVW    IDF_W_INPIFI(R8),FAB$W_IFI(AP) ;RESTORE INTERNAL FILE INDEX
                     03A3    606          $CLOSE  FAB=(AP)                       ;CLOSE INDIRECT INPUT FILE
                     03AC    607
                     03AC    608 ;
                     03AC    609 ; DEALLOCATE LOCAL SYMBOLS AND LABELS FOR CURRENT LEVEL
                     03AC    610 ;
   53    38 BB    0F  03AC    611 10$:     REMQUE  @PRC_Q_LOCAL(R11),R3           ;REMOVE NEXT ENTRY FROM LOCAL SYMBOL TABLE
            06    1C  03B0    612          BVC     20$                            ;IF VC ENTRY REMOVED
   53    30 BB    0F  03B2    613          REMQUE  @PRC_Q_LABEL(R11),R3           ;REMOVE NEXT ENTRY FROM LOCAL LABEL TABLE
            05    1D  03B6    614          BVS     30$                            ;IF VS TABLE EMPTY
          FC45'   30  03B8    615 20$:     BSBW    DCL$DEALLOCSYM                 ;DEALLOCATE SYMBOL ENTRY
            EF    11  03BB    616          BRB     10$                            ;
                     03BD    617 ;
                     03BD    618 ; DEALLOCATE F$SEARCH CONTEXT BLOCKS FOR CURRENT LEVEL
                     03BD    619 ;
   53    64 A8   D0  03BD    620 30$:     MOVL    IDF_L_SEARCHCTX(R8),R3         ;GET FIRST ENTRY OFF F$SEARCH LIST
            26    13  03C1    621          BEQL    32$                            ;BRANCH IF NONE LEFT
 64 A8    63   D0  03C3    622          MOVL    (R3),IDF_L_SEARCHCTX(R8)       ;REMOVE FROM LINKED LIST
3C A3    FC55 CF  90  03C7    623          MOVB    NLA0,FAB$B_FNS+8(R3)           ;SET NULL DEVICE NAME
34 A3    FC50 CF  9E  03CD    624          MOVAB   NLA0+1,FAB$L_FNA+8(R3)         ;
                     03D3    625          $PARSE  FAB=8(R3)                      ;TERMINATE SEARCH SEQUENCE
   50    53   D0  03DD    626          MOVL    R3,R0                          ;SET ADDRESS OF BLOCK TO DEALLOCATE
   51    04 A0   D0  03E0    627          MOVL    4(R0),R1                       ;GET SIZE OF ENTRY IN BYTES
          FC19'   30  03E4    628          BSBW    DCL$DEADYNMEM                  ;DEALLOCATE CONTEXT BLOCK
            D4    11  03E7    629          BRB     30$                            ;LOOP UNTIL LIST CLEANED OUT
                     03E9    630
                     03E9    631 ;
                     03E9    632 ; DEALLOCATE FILE NAME STRING
                     03E9    633 ;
   50    68 A8   D0  03E9    634 32$:     MOVL    IDF_L_FILENAME(R8),R0          ;GET ADDRESS OF ASCIC FILENAME
```

```
                51    60    9A  03ED    635          MOVZBL  (R0),R1                  ;GET SIZE OF FILENAME
                51    08    C0  03F0    636          ADDL    #8,R1                   ;ROUND UP SIZE TO QUAD BOUNDARY
                51    07    CA  03F3    637          BICL    #7,R1                   ;TRUNCATE SIZE TO QUAD BOUNDARY
                   FC07'    30  03F6    638          BSBW    DCL$DEADYNMEM           ;DEALLOCATE BUFFER
                               03F9    639
                               03F9    640 ;
                               03F9    641 ; RESET ON CONDITIONS BACK TO DEFAULTS
                               03F9    642 ;
                   FC04'    30  03F9    643          BSBW    DCL$ONRESET             ;RESET ON ERROR PARAMETERS
                   FC01'    30  03FC    644          BSBW    DCL$ONCTLYRST           ;AND THE ON CONTROL Y HANDLER
                               03FF    645 ;
                               03FF    646 ; CHECK IF THE FRAME JUST CLOSED WAS THE FIRST EXE-ONLY FRAME ENCOUNTERED.
                               03FF    647 ; IF SO, RESTORE VERIFICATION STATE FROM SAVED FLAGS.
                               03FF    648 ;
                    0178    30  03FF    649          BSBW    RES_EXE_ONLY            ;CHECK EXE-ONLY PARAMETERS.
                               0402    650 ;
                               0402    651 ; POINT BACK TO THE PREVIOUS INDIRECT FRAME
                               0402    652 ;
          00BC CB     68    D0  0402    653          MOVL    IDF_L_LNK(R8), -        ; UNLINK FRAME FROM INDIRECT LIST
                               0407    654                  PRC_L_IDFLNK(R11)       ; AND RESET FRAME POINTER
        00A0 CB     74 A8   9E  0407    655          MOVAB   IDF_K_LENGTH(R8), -     ; REMOVE CURRENT INDIRECT FRAME FROM
                               040D    656                  PRC_L_STACKPT(R11)      ; STACK AND RESET STACK POINTER
                 5C AB      D7  040D    657          DECL    PRC_L_INDEPTH(R11)      ; SET NEW INDIRECT STACK DEPTH
                 7C AB      D6  0410    658          INCL    PRC_L_INDCLOCK(R11)     ; COUNT TOTAL STACKS OR UNSTACKS
              58   00BC CB    D0  0413    659          MOVL    PRC_L_IDFLNK(R11),R8    ; POINT TO PREVIOUS INDIRECT FRAME
                               0418    660
                               0418    661 ;
                               0418    662 ; RESTORE THE SAVED CONTEXT FROM THE PREVIOUS INDIRECT FRAME
                               0418    663 ;
           38 AB    10 A8    7D  0418    664          MOVQ    IDF_Q_LOCAL(R8),PRC_Q_LOCAL(R11) ;RESTORE LOCAL SYMBOL TABLE LISTHEA
           30 AB    18 A8    7D  041D    665          MOVQ    IDF_Q_LABEL(R8),PRC_Q_LABEL(R11) ;RESTORE LOCAL LABEL TABLE LISTHEAD
           6A AB    06 A8    B0  0422    666          MOVW    IDF_W_ONLEVEL(R8),PRC_W_ONLEVEL(R11) ;RESTORE ON ERROR LEVEL NUMBER
           6C AB    08 A8    D0  0427    667          MOVL    IDF_L_ONERROR(R8),PRC_L_ONERROR(R11) ;RESTORE ADDRESS OF COMMAND TEX
        00B8 CB     60 A8    D0  042C    668          MOVL    IDF_L_ONCTLY(R8),PRC_L_ONCTLY(R11) ;AND THE ON CONTROL T HANDLER
                               0432    669
                               0432    670 ;
                               0432    671 ; RE-OPEN THE INPUT PROCEDURE FILE ASSOCIATED WITH THE PREVIOUS
                               0432    672 ; INDIRECT FRAME.
                               0432    673 ;
        FFFF 8F     58 A8    B1  0432    674          CMPW    IDF_W_INPRFA(R8),#^XFFFF;IS THE INPUT FILE ALREADY AT EOF?
                       03    12  0438    675          BNEQ    35$                     ;NO, THEN BRANCH
                     00EA    31  043A    676          BRW     50$                     ;YES, THEN DO NOT REOPEN
                  08 AB      D0  043D    677 35$:     MOVL    PRC_L_INPRAB(R11),-     ;ASSUME RETURNING TO LEVEL ZERO AND-
                  14 AB          0440    678                  PRC_L_INDINPRAB(R11)    ;SET INPUT AS INDIRECT INPUT ALSO
           03 5E A8    00    E0  0442    679          BBS     #IDF_V_INPOPN,IDF_W_FLAG(R8),351$ ;CONTINUE IF NOT GOING TO LEVEL 0
                     008A    31  0447    680          BRW     371$                    ;SKIP IF GOING TO LEVEL 0
                               044A    681
                       02    DD  044A    682 351$:    PUSHL   #PSL$C_SUPER            ;PUSH ACCESS MODE
             7E    FBBB CF    9E  044C    683          MOVAB   SYS_INPUT_NAME+1,-(SP)  ;BUILD LOGICAL NAME DESCRIPTOR
             7E    FBB5 CF    9A  0451    684          MOVZBL  SYS_INPUT_NAME,-(SP)    ;
             7E    FBBB CF    9E  0456    685          MOVAB   LNM$PROCESS+1,-(SP)     ;BUILD TABLE NAME DESCRIPTOR
             7E    FBB5 CF    9A  045B    686          MOVZBL  LNM$PROCESS,-(SP)       ;
                   51    5E   D0  0460    687          MOVL    SP,R1                   ;SAVE ADDR. OF DESCRIPTORS
                               0463    688          $DELLNM_S       TABNAM=(R1),-   ;DELETE SYS$INPUT
                               0463    689                  LOGNAM=8(R1),-
                               0463    690                  ACMODE=16(R1)
                   5E    14   C0  0472    691          ADDL    #4*5,SP                 ;CLEAN STACK
```

```
                        0475   692
 56   00F4 CC    9E     0475   693          MOVAB   PRD_G_ALTINPRAB(AP),R6  ;GET THE ALTERNATE INPUT RAB
   14 AB    56    D0     047A   694          MOVL    R6,PRC_L_INDINPRAB(R11) ;SET THAT IS INDIRECT INPUT RAB
      0C A8    D0     047E   695          MOVL    IDF_L_INPRABCTX(R8),-   ;RESTORE STACKED DEVICE CHARACTERISTICS-
         18 A6           0481   696                  RAB$L_CTX(R6)           ;VALUE FROM STACK FRAME
 57   28 AC    D0     0483   697          MOVL    FAB$L_NAM(AP),R7        ;ADDRESS OF NAME BLOCK
                      0487   698          ASSUME  IDF_W_INPFID EQ IDF_T_INPDVI+16
                      0487   699          ASSUME  IDF_W_INPDID EQ IDF_W_INPFID+6
 3C A8    1C    28     0487   700          MOVC    #28,IDF_T_INPDVI(R8),=  ;COPY PREVIOUS INPUT DEVICE,FILE AND-
      14 A7           048B   701                  NAM$T_DVI(R7)           ;DIRECTORY ID'S INTO NAME BLOCK
                      048D   702          ASSUME  NAM$B_DEV EQ NAM$B_NODE+1
                      048D   703          ASSUME  NAM$B_DIR EQ NAM$B_DEV+1
                      048D   704          ASSUME  NAM$B_NAME EQ NAM$B_DIR+1
                      048D   705          ASSUME  NAM$B_TYPE EQ NAM$B_NAME+1
                      048D   706          ASSUME  NAM$B_VER EQ NAM$B_TYPE+1
      38 A7    D4     048D   707          CLRL    NAM$B_NODE(R7)          ;INIT. FILE SPEC. SIZE FIELDS BEFORE
                      0490   708                                          ;REUSING NAM BLOCK.
      3C A7    B4     0490   709          CLRW    NAM$B_TYPE(R7)          ;
                      0493   710
                      0493   711          ASSUME  NAM$B_RSL EQ NAM$B_RSS+1
                      0493   712          ASSUME  NAM$B_ESL EQ NAM$B_ESS+1
      02 A7    B4     0493   713          CLRW    NAM$B_RSS(R7)           ;SET RESULT RESULTANT AND EXPANDED
      0A A7    B4     0496   714          CLRW    NAM$B_ESS(R7)           ;STRING SIZES TO NULL SO THAT THE
                      0499   715                                          ;RSA AND ESA WON'T BE USED.
                      0499   716
16 AC    82 8F    90     0499   717          MOVB    #FAB$M_EXE!FAB$M_GET,FAB$B_FAC(AP) ;SET FILE ACCESS TYPE
  010C0000 8F    D0     049E   718          MOVL    #FAB$M_INP!FAB$M_PPF!FAB$M_NAM,- ;SET FILE OPEN OPTIONS
         04 AC           04A4   719                  FAB$L_FOP(AP)
         34 AC    94     04A6   720          CLRB    FAB$B_FNS(AP)           ;REMOVE RESIDUAL FILE NAME SIZE
         17 AC    94     04A9   721          CLRB    FAB$B_SHR(AP)           ;CLEAR FILE SHARING OPTIONS
         02 AC    B4     04AC   722          CLRW    FAB$W_IFI(AP)           ;CLEAR INPUT IFI
            01    E1     04AF   723          BBC     #IDF_V_REMOTE,-         ;SKIP IF NOT REMOTE ACCESS
      11 5E A8           04B1   724                  IDF_W_FLAG(R8),36$      ;
                      04B4   725          CLRBIT  FAB$V_NAM,FAB$L_FOP(AP) ;CLEAR OPEN BY NAM BLOCK FLAG
   50   68 A8    D0     04B9   726          MOVL    IDF_L_FILENAME(R8),R0   ;GET ADDRESS OF ASCIC FILENAME
   34 AC    80    90     04BD   727          MOVB    (R0)+,FAB$B_FNS(AP)     ;GET LENGTH OF FILE NAME
   2C AC    50    D0     04C1   728          MOVL    R0,FAB$L_FNA(AP)        ;GET ADDRESS OF FILE NAME
                      04C5   729 36$:     $OPEN   FAB=(AP)                ;OPEN PREVIOUS INPUT
         05 50    E8     04CE   730          BLBS    R0,38$                  ;BRANCH IF SUCCESSFUL
          FB2C'   30     04D1   731 37$:     BSBW    DCL$ERRORMSG            ;REPORT ERROR MESSAGE
            3D    11     04D4   732 371$:    BRB     40$
                      04D6   733 38$:     CLRBIT  FAB$V_NAM,FAB$L_FOP(AP) ;REMOVE OPEN BY NAME BLOCK FLAG
04 A8    02 AC    B0     04DB   734          MOVW    FAB$W_IFI(AP),IDF_W_INPIFI(R8) ;SET NEW INPUT IFI
         02 A6    B4     04E0   735          CLRW    RAB$W_ISI(R6)           ;ZERO PREVIOUS INTERNAL SEQUENCE NUMBER
                      04E3   736          $CONNECT RAB=(R6)               ;CONNECT TO PREVIOUS INPUT
         E2 50    E9     04EC   737          BLBC    R0,37$                  ;BRANCH IF UNSUCCESSFUL
1F 18 A6    1C    E1     04EF   738          BBC     #DEV$V_RND,RAB$L_CTX(R6),40$ ;SKIP IF NOT A DISK FILE
14 A6    5C A8    B0     04F4   739          MOVW    IDF_W_INPRFA+4(R8),RAB$W_RFA4(R6) ;COPY RECORD FILE ADDRESS FROM
10 A6    58 A8    D0     04F9   740          MOVL    IDF_W_INPRFA(R8),RAB$W_RFA(R6) ;FROM INDIRECT STACK TO RAB
            13    13     04FE   741          BEQL    40$                     ;BR IF PREVIOUS FILE AT TOP OF FILE
   1E A6    02    90     0500   742          MOVB    #RAB$C_RFA,RAB$B_RAC(R6) ;SET ACCESS MODE TO RECORD FICE ADR
                      0504   743          $FIND   RAB=(R6)                ;SET TO NEXT RECORD POSITION
         C1 50    E9     050D   744          BLBC    R0,37$                  ;BRANCH IF UNSUCCESSFUL
                      0510   745          ASSUME  RAB$C_SEQ EQ 0
         1E A6    94     0510   746          CLRB    RAB$B_RAC(R6)           ;SET ACCESS TO SEQUENTIAL
                      0513   747
                      0513   748 ;
```

```
                     0513   749 ; CLOSE CURRENT OUTPUT FILE IF THE CURRENT OUTPUT FILE IS DIFFERENT
                     0513   750 ; FROM THE PREVIOUS LEVEL.  CREATE SYS$INPUT AND SYS$OUTPUT LOGICAL NAMES.
                     0513   751 ;
        58 8ED0      0513   752 40$:     POPL    R8                              ;GET ADDR OF JUST CLOSED IDF FRAME
52  0094 C8   9E     0516   753         MOVAB   IDF_W_OUTIFI+IDF_K_LENGTH(R8),R2 ;GET ADDR OF OUTPUT FILE INFO
      FAE2'   30     0518   754         BSBW    DCL$RESTORE_OUTPUT              ;RESET OLD SYS$OUTPUT
58  00BC CB   D0     051E   755         MOVL    PRC_L_IDFLNR(R11),R8            ;GET ADDR OF CURRENT IDF FRAME
      FADA'   30     0523   756         BSBW    DCL$CREATE_IO                   ;CREATE 'INPUT' AND 'OUTPUT' LOGICAL NAMES
              05     0526   757         RSB
                     0527   758
                     0527   759 ;
                     0527   760 ; DO NOT OPEN THIS INPUT FILE.  REOPEN THE NEXT ONE.
                     0527   761 ;
        58 8ED0      0527   762 50$:     POPL    R8                              ;GET ADDR OF JUST CLOSED IDF FRAME
52  0094 C8   9E     052A   763         MOVAB   IDF_W_OUTIFI+IDF_K_LENGTH(R8),R2 ;GET ADDR OF OUTPUT FILE INFO
      FACE'   30     052F   764         BSBW    DCL$RESTORE_OUTPUT              ;RESET OLD SYS$OUTPUT
58  00BC CB   D0     0532   765         MOVL    PRC_L_IDFLNR(R11),R8            ;GET ADDR OF CURRENT IDF FRAME
          58   DD     0537   766         PUSHL   R8                              ;SAVE THAT ADDRESS
        FE70   31     0539   767         BRW     10$                             ;REOPEN NEXT INPUT FILE
```

```
                     053C   769          .SBTTL  SAVE VERIFICATION STATE
                     053C   770 ;+
                     053C   771 ; SAV_EXE_ONLY - SAVE EXECUTE ONLY VERIFICATION STATE
                     053C   772 ;
                     053C   773 ; THIS ROUTINE CHECKS IF PROCEDURE THAT IS ABOUT TO BE EXECUTED IS THE FIRST
                     053C   774 ; EXECUTE-ONLY PROCEDURE ENCOUTERED SO FAR. IF SO, IT SAVES THE VERIFICATION
                     053C   775 ; STATES AND THE LEVEL NUMBER.
                     053C   776 ;
                     053C   777 ; INPUTS:
                     053C   778 ;
                     053C   779 ;       R11 - ADDRESS OF PROCESS WORK AREA
                     053C   780 ;
                     053C   781 ; OUTPUTS:
                     053C   782 ;
                     053C   783 ;       NONE
                     053C   784 ;-
                     053C   785
                     053C   786 SAV_EXE_ONLY:
          7E   56    DD  053C   787          MOVL    R6,-(SP)                        ;SAVE WORK REGISTER
    2B 16 AC   01    E0  053F   788          BBS     #FAB$V_GET,FAB$B_FAC(AP),30$    ;SKIP IF READ ACCESS
        012D CB      95  0544   789          TSTB    PRC_B_EXONLY(R11)               ;FIRST ONE ENCOUNTERED?
             25      12  0548   790          BNEQ    30$                             ;NO, JUST SKIP IT
                         054A   791
             03      8A  054A   792          BICB    #PRC_V_SAVCMDV!PRC_V_SAVIMGV,-          ;PRESET SAVED VERIF. FLAGS
        012C CB          054C   793          PRC_B_OUTFLAGS(R11)
             56      D4  054F   794          CLRL    R6                              ;TURN OFF IMG. VERIF.
    05 68 AB   07    E1  0551   795          BBC     #PRC_V_VERIFY,PRC_W_FLAGS(R11),10$      ;SKIP IF NO VERIFY
                         0556   796          SETBIT  PRC_V_SAVCMDV,PRC_B_OUTFLAGS(R11)       ;SET CMD. VERIFY
 C5 00AF CB  07    E1  055B   797 10$:     BBC     #PRC_V_VERIMAGE,PRC_B_FLAGS2(R11),20$   ;SKIP IF NO VERIFY
                         0561   798          SETBIT  PRC_V_SAVIMGV,PRC_B_OUTFLAGS(R11)       ;SET IMG. VERIFY
           FA97'     30  0566   799 20$:     BSBW    DCL$SETVERIFY_IMAGE
 012D CB    5C   AB    90  0569   800          MOVB    PRC_L_INDEPTH(R11),PRC_B_EXONLY(R11)    ;SAVE LEVEL NUMBER
        56    8E       DO  056F   801 30$:     MOVL    (SP)+,R6                        ;RESTORE WORK REGISTERS
                         0572   802          STATUS  NORMAL                          ;ALWAYS EXIT WITH SUCCESS
                     05  0579   803          RSB                                     ;EXIT
```

```
                     057A   805          .SBTTL  RESTORE VERIFICATION STATE
                     057A   806 ;+
                     057A   807 ;
                     057A   808 ; RES_EXE_ONLY - RESTORE EXECUTE ONLY VERIFICATION STATE
                     057A   809 ;
                     057A   810 ; THIS ROUTINE CHECKS IF THE FRAME CURRENTLY BEING UNSTACKED IS THE FIRST
                     057A   811 ; EXE-ONLY PROCEDURE ENCOUNTERED. IF SO, IT RESTORES THE VERIFICATION STATES
                     057A   812 ; TO WHAT THEY WERE PRIOR TO THE EXECUTE ONLY PROCEDURE.
                     057A   813 ;
                     057A   814 ; INPUTS:
                     057A   815 ;
                     057A   816 ;       R11 = ADDRESS OF PROCESS WORK AREA
                     057A   817 ;
                     057A   818 ; OUTPUTS:
                     057A   819 ;
                     057A   820 ;       NONE
                     057A   821 ;-
                     057A   822
                     057A   823 RES_EXE_ONLY:
                7E   56   D0 057A   824          MOVL    R6,-(SP)                ;SAVE WORK REGISTER
                  5C AB   91 057D   825          CMPB    PRC_L_INDEPTH(R11),-    ;IS THIS 1ST EX-ONLY LEVEL?
                012D CB      0580   826                  PRC_B_EXONLYL(R11)
                     27   12 0583   827          BNEQ    30$                     ;NO, SKIP THIS ONE
                          0585   828
                012D CB   94 0585   829          CLRB    PRC_B_EXONLYL(R11)      ;CLEAR EXE-ONLY FLAG
                          0589   830          CLRBIT  PRC_V_VERIFY,PRC_W_FLAGS(R11)   ;INIT. CMD. VERIF. FLAG
 05 012C CB   02   E1 058E   831          BBC     #PRC_V_SAVCMDV,PRC_B_OUTFLAGS(R11),10$  ;SKIP IF NO VERIFY
                          0594   832          SETBIT  PRC_V_VERIFY,PRC_W_FLAGS(R11)   ;SET CMD. VERIFICATION
                          0599   833
                56   D4 0599   834 10$:     CLRL    R6                      ;ASSUME NO IMAGE VERIFICATION
 02 012C CB   03   E1 059B   835          BBC     #PRC_V_SAVIMGV,PRC_B_OUTFLAGS(R11),20$  ;SKIP IF NO VERIFY
                56   D6 05A1   836          INCL    R6                      ;SET FLAG TO SET IMG. VERIFICATION
                          05A3   837 20$:     CLRBIT  PRC_V_VERIMAGE,PRC_B_FLAGS2(R11) ;SYNC FLAG WITH LAST SET STATE
             FA54'   30 05A9   838          BSBW    DCL$SETVERIFY_IMAGE     ;SET/RESET IMAGE VERIFICATION
          56   8E   D0 05AC   839 30$:     MOVL    (SP)+,R6                ;RESTORE WORK REGISTER
                          05AF   840          STATUS  NORMAL                  ;ALWAYS SIGNAL SUCCESS
                     05   05B6   841          RSB                             ;EXIT
                          05B7   842
                          05B7   843          .END
```

```
$$.TMP1                 = 00000001
$$.TMP2                 = 00000066
$$T1                    = 00000000
CLI$_DEFOVF             = 00038028
CLI$_IVQUAL             = 00038240
CLI$_IVVALU             = 00038C98
CLI$_NORMAL             = 00030001
CLI$_STKC               = 00038128
CLI$_SYMOVF             = 00038138
CLI$_USGOTO             = 00038148
DCL$ALLDYNMEM           ******** X   02
DCL$ALLOCSYM            ******** X   02
DCL$COMPRESS            ******** X   02
DCL$CREATE_IO           ******** X   02
DCL$DEADYNMEM           ******** X   02
DCL$DEALGOTO            ******** X   02
DCL$DEALLOCSYM          ******** X   02
DCL$DEFINE_P1_TO_P8     00000105 RG    02
DCL$DISABLE             ******** X   02
DCL$ERRORMSG            ******** X   02
DCL$GETDVAL             ******** X   02
DCL$MARK                ******** X   02
DCL$MARKEDTOKEN         ******** X   02
DCL$MOVCHAR             ******** X   02
DCL$MOVTOKN             ******** X   02
DCL$ONCTLYRST           ******** X   02
DCL$ONRESET             ******** X   02
DCL$OPEN_CREATE         ******** X   02
DCL$OPEN_OUTPUT         ******** X   02
DCL$PROCFILE            ******** X   02
DCL$PUSHPROC            00000139 RG    02
DCL$RESTORE_OUTPUT      ******** X   02
DCL$RUNDOWN             ******** X   02
DCL$SETCHAR             ******** X   02
DCL$SETNBLK             ******** X   02
DCL$SETVERIFY_IMAGE     ******** X   02
DCL$SET_STATUS          ******** X   02
DCL$STACKIND            00000027 RG    02
DCL$T_DEFONTXT          ******** X   02
DCL$UNSTACK             00000319 RG    02
DEV$V_RND               = 0000001C
FAB$B_DNS               = 00000035
FAB$B_FAC               = 00000016
FAB$B_FNS               = 00000034
FAB$B_RAT               = 0000001E
FAB$B_RFM               = 0000001F
FAB$B_SHR               = 00000017
FAB$C_VFC               = 00000003
FAB$L_DEV               = 00000040
FAB$L_DNA               = 00000030
FAB$L_FNA               = 0000002C
FAB$L_FOP               = 00000004
FAB$L_NAM               = 00000028
FAB$M_EXE               = 00000080
FAB$M_GET               = 00000002
FAB$M_INP               = 00080000
FAB$M_NAM               = 01000000
FAB$M_PPF               = 00040000
FAB$M_PRN               = 00000004
FAB$V_GET               = 00000001
FAB$V_NAM               = 00000018
FAB$W_IFI               = 00000002
IDF_B_OUTFLAGS            00000038
IDF_C_LENGTH              00000074
IDF_K_LENGTH              00000074
IDF_L_FILENAME            00000068
IDF_L_INPRABCTX           0000000C
IDF_L_LNK                 00000000
IDF_L_ONCTLY              00000060
IDF_L_ONERROR             00000008
IDF_L_OUTRABCTX           00000024
IDF_L_SEARCHCTX           00000064
IDF_Q_LABEL               00000018
IDF_Q_LOCAL               00000010
IDF_T_INPDVI              0000003C
IDF_T_OUTDVI              00000028
IDF_V_INPCCL            = 00000001
IDF_V_INPOPN            = 00000000
IDF_V_REMOTE            = 00000001
IDF_W_FLAG                0000005E
IDF_W_INPDID              00000052
IDF_W_INPFID              0000004C
IDF_W_INPIFI              00000004
IDF_W_INPRFA              00000058
IDF_W_ONLEVEL             00000006
IDF_W_OUTIFI              00000020
IDF_W_OUTISI              00000022
INPFILE                   00000000 R     03
LNM$PROCESS               00000014 R     02
LOG$C_PROCESS           = 00000002
NAM$B_DEV               = 00000039
NAM$B_DIR               = 0000003A
NAM$B_ESL               = 0000000B
NAM$B_ESS               = 0000000A
NAM$B_NAME              = 0000003B
NAM$B_NODE              = 00000038
NAM$B_NOP               = 00000008
NAM$B_RSL               = 0C000003
NAM$B_RSS               = 00000002
NAM$B_TYPE              = 0000003C
NAM$B_VER               = 0000003D
NAM$C_MAXR              = 000000FF
NAM$L_ESA               = 0000000C
NAM$L_FNB               = 00000034
NAM$L_RSA               = 00000004
NAM$M_PWD               = 00000001
NAM$T_DVI               = 00000014
NAM$V_CNCL_DEV          = 0000000C
NAM$V_NODE              = 00000011
NAM$V_PWD               = 00000000
NLA0                      00000020 R     02
OUTQUAL                   00000004 R     02
PRC_B_CONTINUE            000000F3
PRC_B_DEFRADIX            000000AE
```

```
PRC_B_EXMDEPMOD  000000AD      PRC_T_OUTDVI              0000011C
PRC_B_EXMDEPWID  000000AC      PRC_V_GOTO              = 00000004
PRC_B_EXONLYL    0000012D      PRC_V_SAVCMDV           = 00000002
PRC_B_FLAGS2     000000AF      PRC_V_SAVIMGV           = 00000003
PRC_B_IMGFLAG    00000078      PRC_V_VERIFY            = 00000007
PRC_B_OUTFLAGS   0000012C      PRC_V_VERIMAGE          = 00000007
PRC_B_PROMPTLEN  000000F0      PRC_W_ASTIOSB             00000C6
PRC_C_LENGTH     00000534      PRC_W_ASTRETN             000000C8
PRC_G_COMMANDS   00000133      PRC_W_ASTSTATUS           000000C4
PRC_G_PROMPT     000000F4      PRC_W_ATTMBX              0000007A
PRC_K_LENGTH     00000534      PRC_W_FLAGS               00000068
PRC_L_CURRKEY    00000048      PRC_W_INPCHAN             00000064
PRC_L_EXMDEPADR  000000A8      PRC_W_ONLEVEL             0000006A
PRC_L_EXTARG     00000094      PRC_W_OUTIFI              00000114
PRC_L_EXTBLK     0000008C      PRC_W_OUTISI              00000116
PRC_L_EXTCOD     0000009C      PRC_W_OUTMBXCHN           000000CA
PRC_L_EXTHND     00000090      PRC_W_OUTMBXREF           000000CE
PRC_L_EXTPRM     00000098      PRC_W_OUTMBXSIZ           000000CC
PRC_L_IDFLNK     000000BC      PRC_W_PMPTCTRL            000000F1
PRC_L_IMGACTSTS  00000080      PRC_W_WAITIOSB            00000066
PRC_L_INDCLOCK   0000007C      PRD_C_LENGTH              00000214
PRC_L_INDEPTH    0000005C      PRD_C_XLENGTH             00000244
PRC_L_INDFAB     0000001C      PRD_G_ALTINPRAB           000000F4
PRC_L_INDINPRAB  00000014      PRD_G_ALTOUTRAB           00000138
PRC_L_INDOUTRAB  00000018      PRD_G_FAB                 00000000
PRC_L_INPRAB     00000008      PRD_G_INPRAB              000000B0
PRC_L_LASTKEY    0000004C      PRD_G_NAM                 00000050
PRC_L_LSTSTATUS  000000B0      PRD_G_OUTRAB              0000017C
PRC_L_ONCTLY     000000B8      PRD_G_TRMLIST             000001E4
PRC_L_ONERROR    0000006C      PRD_G_XABTRM              000001C0
PRC_L_OUTOFBAND  000000B4      PRD_K_LENGTH              00000214
PRC_L_OUTRAB     00C0000C      PRD_K_XLENGTH             00000244
PRC_L_OUTRABCTX  00000118      PRD_T_OUTDVI              00000214
PRC_L_PPFLIST    00000070      PRD_T_OUTFNM              00000230
PRC_L_RECALLPTR  0000012F      PRD_W_OUTDID              0000022A
PRC_L_RESTART    00000058      PRD_W_OUTFID              00000224
PRC_L_SAVAP      00000000      PSL$C_SUPER             = 00000002
PRC_L_SAVFP      00000004      PSL$C_USER              = 00000003
PRC_L_SEVERITY   00000050      PTR_B_LEVEL               00000004
PRC_L_SPWN       000000C0      PTR_B_NUMBER              00000005
PRC_L_STACKLM    000000A4      PTR_B_PARMCNT             00000006
PRC_L_STACKPT    000000A0      PTR_B_VALUE               00000000
PRC_L_STATUS     00000054      PTR_C_LENGTH              0000000C
PRC_L_STS        00000084      PTR_K_LENGTH              0000000C
PRC_L_STV        00000088      PTR_K_PARAMETR          = 00000003
PRC_L_SYMBOL     00000060      PTR_L_DESCR               00000000
PRC_L_TMBX       00000074      PTR_L_ENTITY              00000008
PRC_L_TRMLIST    00000010      RAB$B_RAC               = 0000001E
PRC_Q_ALLOCREG   00000020      RAB$C_RFA               = 00000002
PRC_Q_COMMAND    000000E0      RAB$C_SEQ               = 00000000
PRC_Q_FLUSHTIME  000000D0      RAB$L_CTX               = 00000018
PRC_Q_GLOBAL     00000028      RAB$L_FAB               = 0000003C
PRC_Q_IMAGENAME  000000D8      RAB$V_PPF_IND           = 0000000E
PRC_Q_KEYPAD     00000040      RAB$W_ISI               = 00000002
PRC_Q_LABEL      00000030      RAB$W_RFA               = 00000010
PRC_Q_LOCAL      00000038      RAB$W_RFA4              = 00000014
PRC_Q_SAVEPRIV   000000E8      RES_EXE_ONLY              0000057A R      02
```

```
RMS$_EOF                ********    X   02
SAV_EXE_ONLY            0000053C R      02
SETIND                  00000370 R      02
SS$_NORMAL              ********    X   02
STKXIT                  0000034F R      02
SYMBOLS               = 00000008
SYM_B_FLAGS             0000000B
SYM_B_NONUNIQUE         0000000B
SYM_B_TYPE              0000000A
SYM_K_STRING          = 00000000
SYM_L_BL                00000004
SYM_L_FL                00000000
SYM_T_SYMBOL            0000000C
SYM_W_SIZE              00000008
SYS$CLOSE               ********   GX   02
SYS$CONNECT             ********   GX   02
SYS$DELLNM              ********   GX   02
SYS$DELLOG              ********   GX   02
SYS$FIND                ********   GX   02
SYS$OPEN                ********   GX   02
SYS$PARSE               ********   GX   02
SYS_INPUT_NAME          0000000A R      02
UNSTACK                 00000393 R      02
WRK_B_CMDOPT            FFFFFFC3
WRK_B_MAXPARM           FFFFFFD0
WRK_B_MINPARM           FFFFFFD1
WRK_B_PARMCNT           FFFFFFCE
WRK_B_PARMSUM           FFFFFFCF
WRK_B_RECALLCNT         FFFFFFC5
WRK_B_VALLEV            FFFFFFC4
WRK_B_VERBTYP           FFFFFFC2
WRK_C_LENGTH            FFFFF486
WRK_G_BUFFER            FFFFF492
WRK_G_INPBUF            FFFFF896
WRK_G_RESULT            FFFFF9B6
WRK_K_LENGTH            FFFFF486
WRK_L_CHARPTR           FFFFF48E
WRK_L_DISALLOW          FFFFFFE6
WRK_L_ERRORRTN          FFFFF9AE
WRK_L_EXPANDPTR         FFFFF486
WRK_L_IMAGE             FFFFFFE2
WRK_L_MARKPTR           FFFFF48A
WRK_L_PAROUT            FFFFFFD2
WRK_L_PMPTADDR          FFFFF9A2
WRK_L_PROMPTRTN         FFFFF9A6
WRK_L_PROPTR            FFFFFFC6
WRK_L_QUABLK            FFFFFFCA
WRK_L_READRTN           FFFFF9AA
WRK_L_RECALLPTR         FFFFFFEA
WRK_L_RSLEND            FFFFFFB6
WRK_L_RSLNXT            FFFFFFBA
WRK_L_SAVAP             FFFFFFF8
WRK_L_SAVFP             FFFFFFFC
WRK_L_SAVSP             FFFFFFF4
WRK_L_SIGNALRTN         FFFFFFD6
WRK_L_SPECRTN           FFFFF9B2
WRK_L_TAB_VEC           FFFFFFDE
WRK_L_VERB              FFFFFFBE
WRK_M_COMMAND         = 00000002
WRK_V_COMMAND         = 00000001
WRK_V_QUOTE           = 00000004
WRK_W_FLAGS             FFFFFFF0
WRK_W_FLAGS2            FFFFFFF2
WRK_W_IMGCHAN           FFFFFFEE
WRK_W_PMPTLEN           FFFFF99E
_$$_                  = 000000EF
```

Psect synopsis

PSECT name	Allocation	PSECT No.	Attributes
. ABS .	00000000 (0.)	00 (0.)	NOPIC USR CON ABS LCL NOSHR NOEXE NORD NOWRT NOVEC BYTE
ABS	FFFFFFFC (0.)	01 (1.)	NOPIC USR CON ABS LCL NOSHR EXE RD WRT NOVEC BYTE
DCL$ZCODE	000005B7 (1463.)	02 (2.)	NOPIC USR CON REL LCL NOSHR EXE RD NOWRT NOVEC BYTE

Performance indicators

Phase	Page faults	CPU Time	Elapsed Time
Initialization	9	00:00:00.08	00:00:02.40
Command processing	83	00:00:00.67	00:00:05.41
Pass 1	355	00:00:15.13	00:00:44.00
Symbol table sort	0	00:00:01.61	00:00:03.90
Pass 2	157	00:00:03.15	00:00:10.87
Symbol table output	34	00:00:00.27	00:00:01.15
Psect synopsis output	2	00:00:00.02	00:00:00.02
Cross-reference output	0	00:00:00.00	00:00:00.00
Assembler run totals	640	00:00:20.93	00:01:07.75

The working set limit was 1500 pages.
76452 bytes (150 pages) of virtual memory were used to buffer the intermediate code.
There were 60 pages of symbol table space allocated to hold 1153 non-local and 47 local symbols.
843 source lines were read in Pass 1, producing 18 object records in Pass 2.
62 pages of virtual memory were used to define 43 macros.

Macro library statistics

Macro library name	Macros defined
_255DUA28:[SYSLIB]SYSBLDMLB.MLB;1	0
_255DUA28:[DCL.OBJ]DCL.MLB;1	14
_255DUA28:[SYS.OBJ]LIB.MLB;1	1
_255DUA28:[SYSLIB]STARLET.MLB;2	19
TOTALS (all libraries)	34

1417 GETS were required to define 34 macros.

There were no errors, warnings or information messages.

MACRO/LIS=LIS$:INDIRECT/OBJ=OBJ$:INDIRECT MSRC$:INDIRECT/UPDATE=(ENH$:INDIRECT)+EXECML$/LIB+LIB$:DCL/LIB+SYS$LIBRARY:SYSBLDMLB/LIB

GETKEYNAM
LIS

GOTO
LIS

EXPRESS
LIS

HANDLE
LIS

IMAGECTRL
LIS

INITIAL
LIS

INDIRECT
LIS

FILECMDS
LIS

IF
LIS

IMAGEXECT
LIS