F 1

```
   CCCCCCCCCCCC     OOOOOOOOO      PPPPPPPPPPPP     YYY         YYY
   CCCCCCCCCCCC     OOOOOOOOO      PPPPPPPPPPPP     YYY         YYY
   CCCCCCCCCCCC     OOOOOOOOO      PPPPPPPPPPPP     YYY         YYY
CCC              OOO          OOO  PPP         PPP  YYY         YYY
CCC              OOO          OOO  PPP         PPP  YYY         YYY
CCC              OOO          OOO  PPP         PPP  YYY         YYY
CCC              OOO          OOO  PPP         PPP     YYY   YYY
CCC              OOO          OOO  PPP         PPP     YYY   YYY
CCC              OOO          OOO  PPP         PPP     YYY   YYY
CCC              OOO          OOO  PPPPPPPPPPPP           YYY
CCC              OOO          OOO  PPPPPPPPPPPP           YYY
CCC              OOO          OOO  PPPPPPPPPPPP           YYY
CCC              OOO          OOO  PPP                    YYY
CCC              OOO          OOO  PPP                    YYY
CCC              OOO          OOO  PPP                    YYY
CCC              OOO          OOO  PPP                    YYY
CCC              OOO          OOO  PPP                    YYY
CCC              OOO          OOO  PPP                    YYY
   CCCCCCCCCCCC     OOOOOOOOO      PPP                    YYY
   CCCCCCCCCCCC     OOOOOOOOO      PPP                    YYY
   CCCCCCCCCCCC     OOOOOOOOO      PPP                    YYY
```

```
                                        E 14
**FILE**ID**COPYSPECS




  CCCCCCCC    OOOOOO    PPPPPPPP    YY      YY    SSSSSSSS  PPPPPPPP    EEEEEEEEEE    CCCCCCCC    SSSSSSSS
  CCCCCCCC    OOOOOO    PPPPPPPP    YY      YY    SSSSSSSS  PPPPPPPP    EEEEEEEEEE    CCCCCCCC    SSSSSSSS
CC          OO      OO  PP      PP  YY      YY  SS          PP      PP  EE          CC          SS
CC          OO      OO  PP      PP  YY      YY  SS          PP      PP  EE          CC          SS
CC          OO      OO  PP      PP    YY  YY    SS          PP      PP  EE          CC          SS
CC          OO      OO  PP      PP    YY  YY    SS          PP      PP  EE          CC          SS
CC          OO      OO  PPPPPPPP        YY        SSSSSS    PPPPPPPP    EEEEEEEE    CC            SSSSSS
CC          OO      OO  PPPPPPPP        YY        SSSSSS    PPPPPPPP    EEEEEEEE    CC            SSSSSS
CC          OO      OO  PP              YY              SS  PP          EE          CC                  SS
CC          OO      OO  PP              YY              SS  PP          EE          CC                  SS
CC          OO      OO  PP              YY              SS  PP          EE          CC                  SS      ....
CC          OO      OO  PP              YY              SS  PP          EE          CC                  SS      ....
  CCCCCCCC    OOOOOO    PP              YY      SSSSSSSS    PP          EEEEEEEEEE    CCCCCCCC  SSSSSSSS        ....
  CCCCCCCC    OOOOOO    PP              YY      SSSSSSSS    PP          EEEEEEEEEE    CCCCCCCC  SSSSSSSS        ....

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

```
0 MODULE copyspecs (   ! Manipulates input and output specifications for COPY utility
0                      LANGUAGE (BLISS32),
0                      IDENT = 'V04-000'
0                      ) =
1 BEGIN
1
1 !
1 !****************************************************************************
1 !*                                                                          *
1 !*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                                 *
1 !*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.                  *
1 !*  ALL RIGHTS RESERVED.                                                    *
1 !*                                                                          *
1 !*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED   *
1 !*  ONLY IN  ACCORDANCE WITH  THE  TERMS  OF  SUCH  LICENSE  AND WITH THE   *
1 !*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR  ANY  OTHER   *
1 !*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY   *
1 !*  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE IS  HEREBY   *
1 !*  TRANSFERRED.                                                            *
1 !*                                                                          *
1 !*  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE WITHOUT NOTICE   *
1 !*  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL EQUIPMENT   *
1 !*  CORPORATION.                                                            *
1 !*                                                                          *
1 !*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR  RELIABILITY OF ITS   *
1 !*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.                 *
1 !*                                                                          *
1 !*                                                                          *
1 !****************************************************************************
1
1 !++
1 ! FACILITY:     COPY Command
1 !
1 ! ABSTRACT:
1 !
1 !       This module obtains input and output specifications from the CLI and opens
1 !       the associated files.
1 !
1 ! ENVIRONMENT:
1 !
1 !       VAX/VMS operating system, unprivileged user mode utility,
1 !       operates at non-AST level.
1 !
1 !--
1 !++
1 !
1 ! AUTHOR:       Carol Peters,   CREATION DATE:  14 April 1978 14:17
1 !
1 ! Modified by:
1 !
1 !       V03-011 TSK0010         Tamar Krichevsky          8-May-1984
1 !               Rearrange the calls to CLI$GET_VALUE and LIB$FIND_FILE, for
1 !               input filename processing.  This will fix the problem of
1 !               COPY a.a,a.a,a.a,a.a NL: copying every other file, instead of
1 !               every file.
1 !
1 !       V03-010 TSK0009         Tamar Krichevsky         20-Apr-1984
```

```
1 !                     Before the input file is opened, clear the longest record
1 !                     length field in the input file's file header XAB.  This will
1 !                     insure that the LRL value will be correct for record oriented
1 !                     devices.  RMS does not clear this field if it is inappropriate.
1 !                     As a result, the LRL could be carried from one file to another.
1 !                     For example, given the command -- COPY foo.txt,SYS$INPUT a.a --
1 !                     SYS$INPUT inherited the LRL form foo.txt.  (Not kosher!)
1 !
1 !             V03-009 TSK0008         Tamar Krichevsky        28-Mar-1984
1 !                     Fix IF statement in COPY$OPN_OUTFIL which sets up the
1 !                     default name string as ";*".  It was broken by TSK007.
1 !
1 !             V03-008 TSK0007         Tamar Krichevsky        2-Mar-1984
1 !                     Convert input file parsing and searching to LIB$FIND_FILE.
1 !                     Place the check for WILD_OUTPUT before the potential reparse
1 !                     of the output file.  RMS changed how it set the bits in the
1 !                     NAM$L_FNB field.
1 !
1 !             V03-007 TSK0006         Tamar Krichevsky        16-Feb-1984
1 !                     Copy the input and output file names form the command line
1 !                     into the appropriate buffers.  They were getting lost and
1 !                     some error messages were being displayed like so:
1 !                     "Error opening  as input"
1 !
1 !                     Also add in check to see if the input file's record format
1 !                     is VFC and the fixed control region size is zero.  The SOS
1 !                     editor created files like this.  It knew that the smallest
1 !                     fixed header size was two bytes; so it assumed 2 when it
1 !                     saw 0.  RMS compenstated for this by setting the size to
1 !                     two bytes.  Unfortunately, the incompatible attributes
1 !                     comparison would fail because the input file's HSZ field in
1 !                     the XAB$FHC was zero, but the output file's HSZ was two.
1 !                     When COPY encounters such an input file, it will change the
1 !                     HSZ field to two.
1 !
1 !             V03-006 TSK0005         Tamar Krichevsky        3-Oct-1983
1 !                     Move the $DISPLAY, which was added in V03-005, to after the
1 !                     the check for a successful file $CREATE or $OPEN.  Otherwise,
1 !                     an extra message is issued when the file can not be accessed
1 !                     for the $DISPLAY.
1 !
1 !             V03-005 LMP0150         L. Mark Pilant,         9-Sep-1983  11:19
1 !                     Add a $DISPLAY to COPY$OPN_OUTFIL so that the protection
1 !                     of the created file may be obtained.
1 !
1 !             V03-004 TSK0004         Tamar Krichevsky        8-Aug-1983
1 !                     Fix ACCVIO during append operations.  Output file's XABPRO
1 !                     should not be removed from XAB chain until file is closed.
1 !
1 !             V03-003 TSK0004         Tamar Krichevsky        8-Aug-1983
1 !                     Modify COPY$OPN_OUTFILE, SETUP_OUTXAB and APPLY_OUT_QUAL so
1 !                     that file protection and revision inforamtion is not propogated
1 !                     to the output file from the input file.  Fix bug which clears
1 !                     the expiration date when the output device is mag-tape. Fix
1 !                     bug in /PROTECTION qualifier so that unspecified fields are
1 !                     left alone.
1 !
```

```
1 !     V03-002 TSK0003         Tamar Krichevsky        4-Feb-1982
1 !             Change over to the new CLI.  Move external declarations from
1 !             COPY.REQ into this module.
1 !
1 !     V03-001 TSK0002         Tamar Krichevsky        4-Feb-1982
1 !             Copy the buckets size from the input FAB in the output XAB to
1 !             insure that the file is created with the correct bucket size.
1 !             When a file is created, if there are any allocation XABs, the
1 !             bucket size in the FAB is ignored.  Therefore, if the input file
1 !             has several areas, and area 0 does not have largest BKZ, something
1 !             other than the BKZ in the first (and only, in COPY's case) XABALL
1 !             must be used.  The largest bucket size is kept in the input file's
1 !             FAB. ****************** NOTE:  This works only if the ISAM files (the
1 !             worst offenders) are copied block mode.  IF FOR ANY REASON ISAM FILES
1 !             ARE COPIED USING RECORD MODE IN THE FUTURE, THIS PROCEDURE WILL HAVE TO
1 !             BE CHANGED.
1 !
1 !     X00025  TSK0001         Tamar Krichevsky        5-Feb-1982
1 !             Have Global Buffer Count (GBC) transferred from input FAB to
1 !             outout FAB.
1 !
1 !     X00024  KRM0038         Karl Malik      12-Jan-1982
1 !             Warn the user (in COPY$OPN_OUTFIL) if the output file
1 !             was forced to stream format ( in a network copy to
1 !             a 10,20 or RT system ).
1 !
1 !     X00023  KRM0035         Karl Malik      31-Dec-1981
1 !             Check for network quoted string in single output filespec
1 !             & if found, do not force multiple output files.
1 !
1 !     X00022  WMC0030         Wayne Cardoza   15-Dec-1981
1 !             Disallow output directory wildcards remaining after the output
1 !             file parse with the related input file.
1 !
1 !     X00021  WMC0021         Wayne Cardoza   8-Dec-1981
1 !             Set no_output_spec if only directory is wild and no explicit
1 !             filename components.
1 !
1 !     X00020  KFH0001         Ken Henderson   28-Sep-1981
1 !             Expiration and Backup dates are not copied from input file,
1 !             but instead are defaulted.
1 !
1 !     X00019  WMC0001         Wayne Cardoza   22-Jul-1981
1 !             Explicit protection specification should not cause old dates
1 !             to be preserved if a file spec is also present.
1 !
1 !     X00018  SPF0001         S. Forgey       27-Jan-1981
1 !             Allow wildcard directories in output file specifications to
1 !             go along with RMS now handling "sticky" directories.
1 !
1 !     X00017  JAK0017         J. Krycka       18-Sep-1980
1 !             Alter the X00006 special check for network access in setting up
1 !             the output Allocation XAB (i.e., gat ALQ and DEQ values from the
1 !             FHC XAB).
1 !
1 !     X00016  TMH0015         Tim Halvorsen   24-Mar-1980
1 !             Force creation of a new file (creation date, owner, prot)
```

```
;    1 !                         if the output file specification is explicit to maintain
;    1 !                         compatibility with release 1 behavior.  This involves changing
;    1 !                         the previous update to remove remove xabpro,rdt,dat if
;    1 !                         explicit output filespec as long as /PROT was not specified
;    1 !                         (If /PROT specified, xabpro must not be removed to allow it
;    1 !                         to work).
;    1 !
;    1 !                 X00015  TMH0014          Tim Halvorsen   19-Mar-1980
;    1 !                         Do not remove output XABPRO,RDT,DAT blocks if concat_follows
;    1 !                         flag is set because we were only trying to prevent changing
;    1 !                         characteristics on existing files -- concatenation always
;    1 !                         produces a new file.  Also, inhibit wildcard directories on
;    1 !                         output file specifications.
;    1 !
;    1 !                 X00014  TMH0013          Tim Halvorsen   17-Mar-1980
;    1 !                         Issue ENDPRM2 call at the same time as ENDPRM1 call
;    1 !                         to eliminate problems with parameter ordering (in MCR,
;    1 !                         the parameters appear in reverse order).
;    1 !
;    1 !                 X00013  JAK0003          J. Krycka       14-Jan-1980
;    1 !                         Undo X00005 change so that COPY will be able to use block I/O
;    1 !                         to copy relative and indexed files over the network.
;    1 !
;    1 !                 X00012  TMH0012          T. Halvorsen    29-Dec-1979
;    1 !                         Remove XABPRO on appends since changing both owner or
;    1 !                         protection is prohibited (see X00010)
;    1 !
;    1 !                 X00011  TMH0011          T. Halvorsen    15-Nov-1979
;    1 !                         Call CLI back with ENDPRM2 after output filespec is
;    1 !                         obtained to signal any unprocessed qualifiers.
;    1 !
;    1 !                 X00010  TMH0010          T. Halvorsen    13-Nov-1979
;    1 !                         Zero the owner UIC field of the XABPRO on appends since
;    1 !                         changing the owner UIC for an existing file is prohibited.
;    1 !
;    1 !                 X00009  TMH0009          T. Halvorsen    24-Oct-1979
;    1 !                         Test for output spec of only an explicit nodename
;    1 !                         so that the filename is defaulted correctly.
;    1 !                         Fix relative volume placement control to be hard (issue an
;    1 !                         error if the file cannot completely be placed on the volume).
;    1 !
;    1 !                 X00008  T. Halvorsen    25-Jul-1979
;    1 !                         Add relative volume placement control.
;    1 !                         Fix message to indicate contiguous-best-try is being tried
;    1 !                         when there is not enough contigous space rather than issuing
;    1 !                         an error message.
;    1 !
;    1 !                 X00007  T. Halvorsen    14-Jul-1979
;    1 !                         Fix problem copying ISAM files after another file (BIO
;    1 !                         was left on from previous file).
;    1 !
;    1 !                 X00006  JAK0002          J. Krycka       16-Mar-1978     14:00
;    1 !                         To support copy of files over the network, get ALQ and DEQ
;    1 !                         values from input XABALL if NET bit is set.
;    1 !
;    1 !                 X00005  JAK0001          J. Krycka       16-Mar-1978     14:00
;    1 !                         To support copy of relative files over the network, set
;  1 !               BRO bit in output FAB if NET bit is set.
;  1 !
;  1 !       X00004  CHP20339      C. Peters      25-Oct-1978     14:10
;  1 !               In COPY$GET_INFILE, zero ESL and RSL fields to avoid
;  1 !               reporting wrong file specification on error.
;  1 !
;  1 !       X00003  CHP19547      C. Peters      7-Oct-1978      14:27
;  1 !               Don't make version numbers sticky in an APPEND command.
;  1 !
;  1 !--
```

```
  1 !
  1 ! Table of Contents
  1 !
  1 FORWARD ROUTINE
  1     copy$get_infile,                                          ! Obtains the input file specification
  1     copy$opn_infile,                                          ! Opens the current input file
  1     copy$get_outfil,                                          ! Obtains the output file specification
  1     copy$opn_outfil,                                          ! Opens the current output file
  1     setup_extend,                                             ! Sets up an output file to be extended.
  1     setup_outxab        : NOVALUE,                            ! Sets up XAB fields for an output file.
  1     apply_out_qual      : NOVALUE;                            ! Sets output fields depending on file qualifiers.
  1
  1 !
  1 ! Include files
  1 !
  1
  1 LIBRARY 'SYS$LIBRARY:STARLET.L32';                             ! VAX/VMS system definitions
  1 REQUIRE 'SRC$:COPYMSG.REQ';                                   ! Definition of macros to SIGNAL a message
  1
  1 !
  1 ! Macros
  1 !
  1 MACRO
  1
  1     ! Check to see if the global or local qualifier flag is set without the
  1     ! local negation flag being set.
  1     !
M 1     qualifier_active( global_qual, local_qual, locally_negated ) =
M 1         (IF (.global_qual AND NOT .locally_negated) OR .local_qual
M 1          THEN true
  1          ELSE false )%
  1     ;
  1
  1 !
  1 ! External variables
  1 !
  1 EXTERNAL
  1     copy$cli_status : $BBLOCK,
  1     copy$sem_status : $BBLOCK,
  1
  1     curr_allocation_value,
  1     curr_extension_value,
  1     curr_protection_or,
  1     curr_protection_and,
  1     curr_file_max_value,
  1     curr_volume_value,
  1
  1     infile_cli_desc     : $BBLOCK[],                          ! Descriptor for input file name returned by CLI
  1     in_name_desc        : VECTOR,                             ! Descriptor of input file specification
  1     out_name_desc       : VECTOR                              ! descriptor for output file specification
  1     ;
  1
  1 REQUIRE
  1     'SRC$:COPY.REQ'                                           ! Field definitions for  COPY$CLI_STATUS and COPY$SE
  1     ;
```

```
; %PRINT:         File: VMSMAC.B32, Version V04-000, Edit 1, WWC, 09-JAN-1978
;  295        0828  1
;  296        0829  1 EXTERNAL ROUTINE
;  297        0830  1      cli$get_value : addressing_mode( general ),
;  298        0831  1      copy$get_global_qual,                    ! Retrieves command level qualifiers
;  299        0832  1      copy$get_local_qual,                     ! Retrieves local qualifiers
;  300        0833  1      copy$check_file_for_match,               ! See if input file matches command line criteria
;  301        0834  1      copy$calc_alq,                           ! Calculates a file extension quantity.
;  302        0835  1      copy$close_outf,                         ! Closes an output file
;  303        0836  1      copy$inopn_err,                          ! Handles an input $OPEN error
;  304        0837  1      copy$log_msg,                            ! Logs a message about COPY's activities
;  305        0838  1      copy$oclose_err,                         ! Handles an output file close error.
;  306        0839  1      copy$outopn_err,                         ! Handles an output $OPEN error
;  307        0840  1      copy$find_input_file,                    ! Finds and parses an input file specification
;  308        0841  1      copy$semantics;                          ! Determines semantics of a command
```

```
GLOBAL ROUTINE copy$get_infile (input_fab, input_nam, input_xaball) =
                                                    ! Obtain input file specification

!++
! Functional description:
!
!       This routine gets an input file specification and all
!       related qualifiers from the Command Language Interpreter. Then
!       the file specification is parsed.
!
!       If a wildcard specification is still being processed, or if
!       no more input specifications are available, this routine just
!       returns successfully.
!
!       A series of flags are set if certain conditions obtain. These
!       conditions describe the current list of files that are candidates
!       for concatenation. The flags are set if the file specification
!       contains input wildcards, an explicit wildcard version number, or an explicit version number.
!
!       Another flag applies only to this specification and says whether it contains any wildcards.
!
! Calling sequence:
!
!       copy$get_infile (input_fab.ra.v, input_nam.ra.v, input_xaball.ra.v)
!
! Input parameters:
!
!       input_fab       - the FAB to use for this input specification
!       input_nam       - the NAM to use for this input specification
!       input_xaball    - the XABALL to use for this input specification
!
! Implicit inputs:
!
!       wildcard_active - a bit in COPY$CLI_STATUS that says that we are
!                         already processing an input wildcard.
!
! Output parameters:
!
!       none
!
! Implicit outputs:
!
!       The fields of the FAB and the NAM block are filled in according
!       to the CLI call and the $PARSE function call.
!
!       The RSL field of the dummy_nam_blk is filled in by the routine COPY$FIND_INPUT_FILE. This is later
!       used in parsing the name additional input files or output files.
!
!       A bit in COPY$CLI_STATUS may be set:
!
!               multiple_input  - more than one input file specification in the command
!               wildcard_active - if a wildcard is present
!
!       Some bits in COPY$SEM_STATUS may be set:
!
!               wild_input      - wildcard fields exist
!               wild_inp_ver    - a wildcard version number exists
```

```
1 !                     exp_inp_ver      - an explicit version number exists
1 !
1 ! Routine value:
1 !
1 !       OK              - success
1 !       NO_MORE_FILES   - success, no more input specifications
1 !       NO_FILE         - failure
1 !
1 ! Side effects:
1 !
1 !       none
1 !
1 !--
1
2     BEGIN
2
2     LOCAL
2         rtn_status;                                       ! Retrun status from external calls
2
2     MAP
2         input_fab       : REF BLOCK [, BYTE],             ! FAB to use with input file
2         input_nam       : REF BLOCK [, BYTE],             ! NAM to use with input file
2         input_xaball    : REF BLOCK [, BYTE];             ! XABALL to use with input file
2
2
2
2     ! Return if a wildcard file specification is currently being processed or the
2     ! last input file name has been retrieved from the command line.  Otherwise,
2     ! set the flag which indicates that more input files have been found.
2     !
2
2     IF .wildcard_active                                   ! If a wildcard specification is currently
2     THEN                                                  ! being processed, then just return to caller.
2         RETURN ok;
2
2     !
2     ! Reinitialize the RSL and ESL fields of the NAM block so that a parsing
2     ! error does not report an error in the previous file processed.
2     !
2
2     input_nam [nam$b_esl] = 0;                            ! Expanded string length of zero.
2     input_nam [nam$b_rsl] = 0;                            ! Resultant string length of zero.
2
2
2     !
2     ! Call LIB$FIND_FILE to parse the input file specification. This resolves
2     ! logical names and determines if there are wildcards present, or explicit
2     ! named fields present.
2     !
2
3     IF NOT (rtn_status = copy$find_input_file ( infile_cli_desc ))
2     THEN
2         IF .rtn_status NEQ RMS$_NMF
2         THEN
2             RETURN .rtn_status;
2
2     !
```

```
    ! Initialize the input file FAB.
    !

    $FAB_INIT (                                                  ! Setup the input file FAB as follows:
                  FAB = .input_fab,                              !    FAB address is the input parameter
                  FAC = <GET,BRO>,                               !        !     Input file, mixed block and record acce
                  SHR = GET,                                     !    Allow others to read the input file
                  DNA = 0,                                       !    No default file specification
                  RTV = 0,                                       !    Use default retrieval window size
                  RAT = CR,                                      !    Carriage control in case unit record input
                  FOP = <SQO,NAM>,                               !    Sequential I/O only, open by name block
                  NAM = .input_nam,                              !    NAM block address
                  XAB = .input_xaball);                          !    XABALL block address.

!
! If there were no more files for the current inout specification, get the next
! one from the command line.
!
    IF .rtn_status EQL RMS$_NMF
    THEN
        BEGIN

        IF NOT (rtn_status = CLI$GET_VALUE( $DESCRIPTOR('infile'), infile_cli_desc))
        THEN
            RETURN no_more_files;

        ! Get the qualifiers for this input file.
        !
        COPY$GET_LOCAL_QUAL();

        ! Check to see if more than one input file has been given.
        !
        IF .rtn_status NEQ SS$_NORMAL
        THEN
            multiple_input = TRUE;

        !
        ! Reinitialize the RSL and ESL fields of the NAM block so that a parsing
        ! error does not report an error in the previous file processed.
        !

        input_nam [nam$b_esl] = 0;                              ! Expanded string length of zero.
        input_nam [nam$b_rsl] = 0;                              ! Resultant string length of zero.

        !
        ! Call LIB$FIND_FILE to parse the input file specification. This resolves
        ! logical names and determines if there are wildcards present, or explicit
        ! named fields present.
        !

        IF NOT (rtn_status = copy$find_input_file ( infile_cli_desc ))
        THEN
            RETURN .rtn_status;
        END;

!
```

```
2 ! Now test the type of expanded name string that we have. Does it contain wildcards? Were
2 ! certain fields explicitly named?
2 !
2
2     IF .input_nam [nam$v_wildcard]                                  ! If there were any wildcards,
2     THEN
3         BEGIN
3         wildcard_active = TRUE;                                     !     set WILDCARD_ACTIVE. This says current file
3                                                                     !     specification contains wildcards.
3         wild_input = TRUE;                                          !     Also set WILD_INPUT. This says that the current
3                                                                     !     input list contains wildcard specs somewhere.
3         first_wild_infile = TRUE;                                   ! Indicate this is the first wild input file
3         END
2     ELSE                                                            ! If no input wildcards in this spec, turn off
2         wildcard_active = FALSE;                                    !     the WILDCARD_ACTIVE flag.
2
2     IF .input_nam [nam$v_wild_ver]                                  ! If an explicit wildcard version number
2     THEN                                                            !     was specified,
2         wild_inp_ver = TRUE                                         !     set the WILD_INP_VER flag.
2     ELSE                                                            ! Otherwise,
3         BEGIN
3         IF .input_nam [nam$v_exp_ver]                               !     see if an explicit version number was specified
3         THEN                                                        !     If it is, set the EXP_INP_VER flag, meaning
3             exp_inp_ver = TRUE;                                     !     that there is an explicit input version number.
2         END;
2
2 !
2 ! Return with success.
2 !
2
2     RETURN ok;
1     END;


                                                        .TITLE  COPYSPECS
                                                        .IDENT  \V04-000\

                                                        .PSECT  $PLIT$,NOWRT,NOEXE,2

                      65  6C  69  66  6E  69  00000 P.AAB:  .ASCII  \infile\                                      ;
                                              00006         .BLKB   2
                                00000006      00008 P.AAA:  .LONG   6                                             ;
                                00000000'     0000C         .ADDRESS P.AAB                                        ;

                                                        .EXTRN  COPY$MSG_NUMBER
                                                        .EXTRN  COPY$CLI_STATUS
                                                        .EXTRN  COPY$SEM_STATUS
                                                        .EXTRN  CURR_ALLOCATION_VALUE
                                                        .EXTRN  CURR_EXTENSION_VALUE
                                                        .EXTRN  CURR_PROTECTION_OR
                                                        .EXTRN  CURR_PROTECTION_AND
                                                        .EXTRN  CURR_FILE_MAX_VALUE
                                                        .EXTRN  CURR_VOLUME_VALUE
                                                        .EXTRN  INFILE_CLI_DESC
                                                        .EXTRN  IN_NAME_DESC, OUT_NAME_DESC
                                                        .EXTRN  CLI$_PRESENT, CLI$_NEGATED
                                                        .EXTRN  CLI$_LOCPRES, CLI$_LOCNEG
```

```
                                                              .EXTRN   CLI$GET_VALUE, COPY$GET_GLOBAL_QUAL
                                                              .EXTRN   COPY$GET_LOCAL_QUAL
                                                              .EXTRN   COPY$CHECK_FILE_FOR_MATCH
                                                              .EXTRN   COPY$CALC_ALQ, COPY$CLOSE_OUTF
                                                              .EXTRN   COPY$INOPN_ERR, COPY$LOG_MSG
                                                              .EXTRN   COPY$OCLOSE_ERR
                                                              .EXTRN   COPY$OUTOPN_ERR
                                                              .EXTRN   COPY$FIND_INPUT_FILE
                                                              .EXTRN   COPY$SEMANTICS

                                                              .PSECT   $CODE$,NOWRT,2

                                              07FC 00000      .ENTRY   COPY$GET_INFILE, Save R2,R3,R4,R5,R6,R7,R8,-; 0842
                                                                       R9,R10                                      ;
                        5A      0000G   CF     9E 00002       MOVAB    INFILE_CLI_DESC, R10                        ;
                        59      0000G   CF     9E 00007       MOVAB    COPY$SEM_STATUS, R9                         ;
           03       02  A9              05     E1 0000C       BBC      #5, COPY$SEM_STATUS+2, 1$                   ; 0930
                                      00B5     31 00011       BRW      10$                                         ;
                        57      08      AC     D0 00014 1$:   MOVL     INPUT_NAM, R7                               ; 0939
                                0B      A7     94 00018       CLRB     11(R7)                                      ;
                                03      A7     94 0001B       CLRB     3(R7)                                       ; 0940
                                        5A     DD 0001E       PUSHL    R10                                         ; 0949
                  0000G  CF             01     FB 00020       CALLS    #1, COPY$FIND_INPUT_FILE                    ;
                         58             50     D0 00025       MOVL     R0, RTN_STATUS                              ;
                         09             58     E8 00028       BLBS     RTN_STATUS, 2$                              ;
               000182CA  8F             58     D1 0002B       CMPL     RTN_STATUS, #99018                          ; 0951
                                        6F     12 00032       BNEQ     5$                                          ;
                         56     04      AC     D0 00034 2$:   MOVL     INPUT_FAB, R6                               ; 0968
    0050   8F       00   6E             00     2C 00038       MOVC5    #0, (SP), #0, #80, (R6)                     ;
                         66                       0003F                                                            ;
                         66   5003      8F     B0 00040       MOVW     #20483, (R6)                                ;
                   04    A6 01000040    8F     D0 00045       MOVL     #16777280, 4(R6)                            ;
                   16    A6   0242      8F     B0 0004D       MOVW     #578, 22(R6)                                ;
                   1E    A6   0202      8F     B0 00053       MOVW     #514, 30(R6)                                ;
                   24    A6     0C      AC     D0 00059       MOVL     INPUT_XABALL, 36(R6)                        ;
                   28    A6             57     D0 0005E       MOVL     R7, 40(R6)                                  ;
               000182CA  8F             58     D1 00062       CMPL     RTN_STATUS, #99018                          ; 0974
                                        3C     12 00069       BNEQ     6$                                          ;
                                        5A     DD 0006B       PUSHL    R10                                         ; 0978
                              0000'     CF     9F 0006D       PUSHAB   P.AAA                                       ;
              00000000G  00             02     FB 00071       CALLS    #2, CLI$GET_VALUE                           ;
                         58             50     D0 00078       MOVL     R0, RTN_STATUS                              ;
                         04             58     E8 0007B       BLBS     RTN_STATUS, 3$                              ;
                         50             03     D0 0007E       MOVL     #3, R0                                      ; 0980
                                               04 00081       RET                                                  ;
                  0000G  CF             00     FB 00082 3$:   CALLS    #0, COPY$GET_LOCAL_QUAL                     ; 0984
                         01             58     D1 00087       CMPL     RTN_STATUS, #1                              ; 0988
                                        04     13 0008A       BEQL     4$                                          ;
                   01    A9             02     88 0008C       BISB2    #2, COPY$SEM_STATUS+1                       ; 0990
                                0B      A7     94 00090 4$:   CLRB     11(R7)                                      ; 0997
                                03      A7     94 00093       CLRB     3(R7)                                       ; 0998
                                        5A     DD 00096       PUSHL    R10                                         ; 1007
                  0000G  CF             01     FB 00098       CALLS    #1, COPY$FIND_INPUT_FILE                    ;
                         58             50     D0 0009D       MOVL     R0, RTN_STATUS                              ;
                         04             58     E8 000A0       BLBS     RTN_STATUS, 6$                              ;
                         50             58     D0 000A3 5$:   MOVL     RTN_STATUS, R0                              ; 1009
                                               04 000A6       RET                                                  ;
                              09          35    A7   E9 000A7 6$:       BLBC    53(R7), 7$                                     ; 1017
                              69 02200010       8F   C8 000AB           BISL2   #35651600, COPY$SEM_STATUS                     ; 1024
                                                04   11 000B2           BRB     8$                                             ; 1017
                        02    A9                20   8A 000B4 7$:       BICB2   #32, COPY$SEM_STATUS+2                         ; 1027
            05          34    A7                03   E1 000B8 8$:       BBC     #3, 52(R7), 9$                                 ; 1029
                              69                20   88 000BD           BISB2   #32, COPY$SEM_STATUS                           ; 1031
                                                07   11 000C0           BRB     10$                                            ;
                              03          34    A7   E9 000C2 9$:       BLBC    52(R7), 10$                                    ; 1034
                              69                02   88 000C6           BISB2   #2, COPY$SEM_STATUS                            ; 1036
                              50                01   DC 000C9 10$:      MOVL    #1, R0                                         ; 1043
                                                     04 000CC           RET                                                    ; 1044

; Routine Size:  205 bytes,    Routine Base:  $CODE$ + 0000
```

```
1 GLOBAL ROUTINE copy$opn_infile (input_fab) =              ! Open the current input file
1
1 !++
1 ! Functional description:
1 !
1 !       This routine opens the current input file. If the input file
1 !       specification contains a wildcard field, an RMS $SEARCH for the
1 !       next wildcard match occurs before the actual file open.
1 !
1 !       Any input parameter qualifiers are applied to the file's RMS blocks before
1 !       the open is performed. For now, the only valid qualifier is /READ_CHECK.
1 !
1 !       If the OPEN fails, an error is reported to SYS$ERROR. When input wildcards are present,
1 !       two types of failure are permitted:
1 !
1 !               RMS$_NMF        - no more files match given wildcard
1 !               open failure    - allowed when a file matching a wildcard spec cannot be
1 !                                 opened, as long as that file would have been copied without concatenation.
1 !
1 ! Calling sequence:
1 !
1 !       copy$opn_infile (input_fab.ra.v)
1 !
1 ! Input parameters:
1 !
1 !       input_fab       - the FAB associated with the input file
1 !
1 ! Implicit inputs:
1 !
1 !       COPY$CLI_STATUS bits are checked:
1 !
1 !               iread_check_bit - This bit is set if the /READ_CHECK qualifier was specified for this file.
1 !               wildcard_active - This specification contains wildcards.
1 !                                 Find the next file with a $SEARCH function call.
1 !
1 !       input file NAM block is read to obtain the length of the resultant name string
1 !       input file XABFHC to check the HSZ for VFC files.
1 !
1 !       COPY$SEM_STATUS bits are checked:
1 !
1 !               multiple_output - Multiple files are being produced. This is checked to allow for
1 !                                 open failure on a wildcard specified file.
1 !
1 ! Output parameters:
1 !
1 !       none
1 !
1 ! Implicit outputs:
1 !
1 !       in_name_desc    - the length field of the input name descriptor is written from the RSL
1 !                         field in the NAM block
1 !
1 !       The FAB$V_RCK bit in the input FAB is set if /READ_CHECK was specified.
1 !
1 !       COPY$CLI_STATUS bit settings may be altered:
1 !
1 !               wildcard_active - turned off if no more files that match wildcard are found.
```

```
1 !                infile_open     - set if the file is opened successfully
1 !
1 ! Routine value:
1 !
1 !       OK               - input file open
1 !       NO_MORE_FILES    - no further wildcard match found
1 !       NO_WILD_OPEN     - open failure on wildcard match file
1 !       NO_FILE          - input file not found
1 !
1 ! Side effects:
1 !
1 !       The input file is opened.
1 !       If an RMS SEARCH function fails, then an error is reported on SYS$ERROR.
1 !
1 !--
1
2    BEGIN
2
2    MAP
2       input_fab        : REF BLOCK [, BYTE];            ! input FAB block
2
2    BIND
2       input_xaball     =                                ! input file XABALL block
2           .input_fab [fab$l_xab]           : BLOCK [, BYTE],
2       input_xabdat     =                                ! input file XABDAT block
2           .input_xaball [xab$l_nxt]        : BLOCK [, BYTE],
2       input_xabfhc     =                                ! input file XABFHC block
2           .input_xabdat [xab$l_nxt]        : BLOCK [, BYTE],
2       input_nam        =                                ! input NAM block address
2                   .input_fab [fab$l_nam] : BLOCK [, BYTE];
2
2    LOCAL
2       status;                                           ! RMS status code variable
2
2 !
2 ! If a wildcard specification is active, call RMS to search for the next wildcard match.
2 !
2
2    IF .wildcard_active                                  ! If an input wildcard field is present,
2    THEN
2       IF NOT .first_wild_infile
2       THEN
3           BEGIN
3           status = COPY$FIND_INPUT_FILE( infile_cli_desc );
3
3           IF .status EQL rms$_nmf                       ! If no more wildcard matches exist,
3           THEN
4               BEGIN
4               wildcard_active = FALSE;                  !    turn off the WILDCARD_ACTIVE flag,
4               RETURN no_more_files;                     !    and return with success status of NO_MORE_FILES
3               END;
3
3           IF NOT .status                                ! If RMS returned some other error code,
3           THEN
4               BEGIN
4               copy$inopn_err (.input_fab);              !    then call the RMS error action routine.
4               wildcard_active = FALSE;                  !    Turn off the wildcard flag so that we don't loo
```

```
  4               RETURN no_file;                               !    for the file again. Return to caller with NO_FI
  3               END;                                          !    error code.
  3           END                                               ! End of special wildcard search processing.
  2       ELSE
  2           first_wild_infile = FALSE;
  2
  2 !
  2 ! If the user specified the input read checking qualifier, turn on the appropriate bit in the FAB.
  2 !
  2
  3   IF qualifier_active( read_chk_qual, loc_read_chk_qual, neg_read_chk_qual)
  2   THEN
  2       input_fab [fab$v_rck] = TRUE                          !    then turn on the FAB read check indicator.
  2   ELSE
  2       input_fab [fab$v_rck] = FALSE;                        !    Otherwise, turn it off.
  2
  2 !
  2 ! Open the input file.  First, zero the LRL field in the file header XAB.  This
  2 ! insures that it will have the appropriate value if the input device is record
  2 ! oriented (i.e. SYS$INPUT).
  2 !
  2
  2   input_xabfhc[ XAB$W_LRL ] = 0;
P 2   IF $RMS_OPEN (                                            ! Open the input file with RMS.
P 2                     FAB = .input_fab,                       !    Specify the input parameter for the FAB,
  3                     ERR = copy$inopn_err)                   !    and an error action routine.
  2   THEN                                                      ! If the OPEN is successful,
  3       BEGIN
  3       infile_open = TRUE;                                   !    indicate that the file is open
  3       in_name_desc [0] = .input_nam [nam$b_rsl];            !    and set the length of the input file name descr
  3
  3       ! If record format is VFC and the HSZ is 0, then set the HSZ to 2.
  3       ! If this isn't done, the incompatible attributes check will
  3       ! incorrectly fail.
  3       !
  3       IF .input_fab [FAB$B_RFM] EQL FAB$C_VFC
  3               AND
  3           .input_xabfhc [XAB$B_HSZ] EQL 0
  3       THEN
  3           input_xabfhc [XAB$B_HSZ] = 2;
  3       RETURN ok;                                            ! Return to caller with success code.
  3       END                                                   ! End of successful OPEN processing
  3
  2   ELSE
  3       BEGIN
  3
  3 !
  3 ! If multiple output files are being produced, and this is a file that matches a wildcard specification,
  3 ! allow the open to fail. This means that one file that matches the wildcard specification is not copied
  3 ! to a new output file.
  3 !
  3
  3       IF .wildcard_active AND
  4           (.multiple_output OR NOT .explicit_concat_qual )
  3       THEN
  3           RETURN no_wild_open
  3       ELSE
```

```
:  685         1216  3              RETURN no_file;
:  686         1217  2          END;
:  687         1218  2
:  688         1219  1      END;



                                                                              .EXTRN    SYS$OPEN

                                                          007C 00000          .ENTRY    COPY$OPN_INFILE, Save R2,R3,R4,R5,R6        ; 1045
                        56      0000G  CF   9E 00002          MOVAB     COPY$CLI_STATUS+4, R6                       ;
                        55      0000G  CF   9E 00007          MOVAB     COPY$SEM_STATUS, R5                         ;
                        52         04  AC   D0 0000C          MOVL      INPUT_FAB, R2                               ; 1125
                        50         24  A2   D0 00010          MOVL      36(R2), R0                                  ;
                        50         04  A0   D0 00014          MOVL      4(R0), R0                                   ; 1127
                        53         04  A0   D0 00018          MOVL      4(R0), R3                                   ; 1129
                        54         28  A2   D0 0001C          MOVL      40(R2), R4                                  ; 1131
        33         02   A5              05  E1 00020          BBC       #5, COPY$SEM_STATUS+2, 3$                   ; 1140
        2A         03   A5              01  E0 00025          BBS       #1, COPY$SEM_STATUS+3, 2$                   ; 1142
                                0000G  CF   9F 0002A          PUSHAB    INFILE_CLI_DESC                             ; 1145
               0000G   CF               01  FB 0002E          CALLS     #1, COPY$FIND_INPUT_FILE                    ;
           000182CA    8F               50  D1 00033          CMPL      STATUS, #99018                              ; 1147
                                        08  12 0003A          BNEQ      1$                                          ;
                   02   A5              20  8A 0003C          BICB2     #32, COPY$SEM_STATUS+2                      ; 1150
                        50              03  D0 00040          MOVL      #3, R0                                      ; 1151
                                            04 00043          RET                                                   ;
                        11              50  E8 00044  1$:     BLBS      STATUS, 3$                                  ; 1154
                                        52  DD 00047          PUSHL     R2                                          ; 1157
               0000G   CF               01  FB 00049          CALLS     #1, COPY$INOPN_ERR                          ;
                  02   A5               20  8A 0004E          BICB2     #32, COPY$SEM_STATUS+2                      ; 1158
                                        5D  11 00052          BRB       11$                                         ; 1159
                  03   A5               02  8A 00054  2$:     BICB2     #2, COPY$SEM_STATUS+3                       ; 1163
                       04               66  E9 00058  3$:     BLBC      COPY$CLI_STATUS+4, 4$                       ; 1169
        04             66               02  E1 0005B          BBC       #2, COPY$CLI_STATUS+4, 5$                   ;
        07             66               01  E1 0005F  4$:     BBC       #1, COPY$CLI_STATUS+4, 6$                   ;
                  06   A2       80      8F  88 00063  5$:     BISB2     #128, 6(R2)                                 ; 1171
                                        05  11 00068          BRB       7$                                          ;
                  06   A2       80      8F  8A 0006A  6$:     BICB2     #128, 6(R2)                                 ; 1173
                                0A      A3  B4 0006F  7$:     CLRW      10(R3)                                      ; 1181
                                0000G   CF  9F 00072          PUSHAB    COPY$INOPN_ERR                              ; 1184
                                        52  DD 00076          PUSHL     R2                                          ;
           00000000G   00               02  FB 00078          CALLS     #2, SYS$OPEN                                ;
                       1D               50  E9 0007F          BLBC      R0, 9$                                      ;
                  02   A5               04  88 00082          BISB2     #4, COPY$SEM_STATUS+2                       ; 1187
               0000G   CF       03      A4  9A 00086          MOVZBL    3(R4), IN_NAME_DESC                         ; 1188
                       03       1F      A2  91 0008C          CMPB      31(R2), #3                                  ; 1194
                                        09  12 00090          BNEQ      8$                                          ;
                                17      A3  95 00092          TSTB      23(R3)                                      ; 1196
                                        04  12 00095          BNEQ      8$                                          ;
                  17   A3               02  90 00097          MOVB      #2, 23(R3)                                  ; 1198
                       50               01  D0 0009B  8$:     MOVL      #1, R0                                      ; 1203
                                            04 0009E          RET                                                   ;
        0D        02   A5               05  E1 0009F  9$:     BBC       #5, COPY$SEM_STATUS+2, 11$                  ; 1211
                       05       01      A5  E8 000A4          BLBS      COPY$SEM_STATUS+1, 10$                      ; 1212
        04        FC   A6               02  E0 000A8          BBS       #2, COPY$CLI_STATUS, 11$                    ;
                       50               05  D0 000AD  10$:    MOVL      #5, R0                                      ; 1216
                                            04 000B0          RET                                                   ;
```

```
                                          50  D4 000B1 11$:     CLRL    R0                                   ; 1219
                                              04 000B3          RET                                          ;

; Routine Size:  180 bytes,    Routine Base:  $CODE$ + 00CD
```

```
1 GLOBAL ROUTINE copy$get_outfil (output_fab, output_nam, output_xabfhc) =
1                                                         ! Obtain the output file specification
1
1 !++
1 ! Functional description:
1 !
1 !       This routine obtains the output file specification and all
1 !       related qualifiers from the Command Language Interpreter. Then
1 !       the file specification is parsed without any help from related input file name
1 !       blocks. This initial parse determines whether the file specification had null file
1 !       name, type, and version number fields.
1 !
1 !       If no output file name, type, or version number is given, a flag
1 !       is set in COPY$SEM_STATUS.
1 !
1 ! Calling sequence:
1 !
1 !       copy$get_outfil (output_fab.ra.v, output_nam.ra.v, output_xabfhc.ra.v)
1 !
1 ! Input parameters:
1 !
1 !       output_fab      - the FAB to use for this output specification
1 !       output_nam      - the NAM to use for this output specification
1 !       output_xabfhc   - the XABFHC to use for this output specification
1 !
1 ! Implicit inputs:
1 !
1 !       The RLF field of the output NAM block contains the address of the input file NAM block.
1 !
1 ! Output parameters:
1 !
1 !       none
1 !
1 ! Implicit outputs:
1 !
1 !       The fields of the FAB and the NAM block are filled in according
1 !       to the CLI call, FAB initialization, and the $PARSE function call.
1 !
1 !       A bit may be set in COPY$SEM_STATUS:
1 !
1 !               no_output_spec  - no output name, type, or version number specified.
1 !
1 ! Routine value:
1 !
1 !       OK              - success
1 !       NO_FILE         - the $PARSE function call returned an error code
1 !
1 ! Side effects:
1 !
1 !       An error is reported if the $PARSE function returns an error status code and
1 !       COPY$OUTOPN_ERR is called.
1 !
1 !--
1
2    BEGIN
2
2    MAP
```

```
        output_fab      : REF BLOCK [, BYTE],                   ! FAB to use with output file
        output_nam      : REF BLOCK [, BYTE],                   ! NAM to use with output file
        output_xabfhc   : REF BLOCK [, BYTE];                   ! XABFHC to use with output file

    LOCAL
        cli_desc : $BBLOCK[ DSC$C_S_BLN ],                      ! Descriptor for qualifier values
        temp_rlf;                                               ! Holds the output RLF field



    ! Initialize descriptor.  Retrieve the output file specification.
    !
    CH$FILL( 0, DSC$C_S_BLN, cli_desc );
    cli_desc[ DSC$B_CLASS ] = DSC$K_CLASS_D;

    CLI$GET_VALUE( $DESCRIPTOR('OUTFILE'), cli_desc);

    ! Save the file name in the output name descriptor; in case the name
    ! doesn't parse.  The name given on the command line will be used
    ! in the error message returned to the user.
    !
    out_name_desc[0] = .cli_desc[DSC$W_LENGTH];
    CH$MOVE(.cli_desc[DSC$W_LENGTH], .cli_desc[DSC$A_POINTER], .out_name_desc[1]);

    ! Get the qualifiers for the output file.
    !
    COPY$GET_GLOBAL_QUAL();

!
! Initialize the output file FAB.
!

    $FAB_INIT (                                                 ! Setup the output file FAB as follows:
                FAB = .output_fab,                              !     FAB address is the output parameter
                FAC = <PUT,TRN>,                                !     Output file
                SHR = NIL,                                      !     No file sharing
                FNA = .cli_desc [DSC$A_POINTER],                !     File name address from CLI
                FNS = .cli_desc [DSC$W_LENGTH],                 !     File name size from CLI also
                RTV = 0,                                        !     Use the system default retrieval window size
                FOP = <SQO,OFP,NAM>,                            !     Sequential operations only, output file parse,
                NAM = .output_nam,                              !     NAM block address
                XAB = .output_xabfhc);                          !     XABFHC block address
                                                                !         name block open

!
! Zero the expanded string length so that the COPY error routine, copy$outopn_err, can
! decide if an expanded name string was created by RMS.
!

    output_nam [nam$b_esl] = 0;                                 ! Zero the output expanded string length.

!
! Temporarily remove the RLF field of the output NAM block so that the
! output file specification can be tested for null name, type, and
! version number fields.
!

```

```
   2      temp_rlf = .output_nam [nam$l_rlf];               ! Save the RLF field because it may be needed later.
   2      output_nam [nam$l_rlf] = 0;                        ! Set the RLF field to null.
   2
   2  !
   2  ! Parse the output file specification.
   2  !
   2
P  2      IF NOT $RMS_PARSE (                                ! Call the RMS function that parses file specificati
P  2                         FAB = .output_fab,              !     specifying the output FAB parameter,
   3                         ERR = copy$outopn_err)          !     and an error routine.
   2      THEN                                               ! If the PARSE is not successful,
   2          RETURN no_file;                                !     then return an error code to the caller.
   2
   2  !
   2  ! Test for an absence of the file name, type, and version number fields
   2  ! (or the presence of a network quoted string).
   2  !
   2
   2      IF (NOT .output_nam [nam$v_wild_name]) AND          ! If no output wildcards are present,
   2          (NOT .output_nam [nam$v_wild_type]) AND
   2          (NOT .output_nam [nam$v_wild_ver]) AND
   2          (NOT .output_nam [nam$v_quoted]) AND           !     and no quoted string
   2          (NOT .output_nam [nam$v_exp_name]) AND         !     and no output name,
   2          (NOT .output_nam [nam$v_exp_type]) AND         !     and no output type,
   2          (NOT .output_nam [nam$v_exp_ver]) AND          !     and no output version number,
   3          (.output_nam [nam$v_exp_dir] OR                !     and an explicit directory
   3          .output_nam [nam$v_exp_dev] OR                 !     or device name
   3          .output_nam [nam$v_node])                      !     or node name is given,
   2      THEN
   2          no_output_spec = TRUE;                         !     then set NO_OUTPUT_SPEC bit.
   2
   2
   2      ! If the file name, file type or version fields are ALL either wild or no specified and
   2      ! the output file spec does not contain a quoted string, then set the flag which indicates
   2      ! that the output file spec was completely wild.
   2      !
   3      IF  (.output_nam [nam$v_wild_name] OR NOT .output_nam [nam$v_exp_name])
   2              AND
   3          (.output_nam [nam$v_wild_type] OR NOT .output_nam [nam$v_exp_type])
   2              AND
   3          (.output_nam [nam$v_exp_ver] OR NOT .output_nam [nam$v_wild_ver])
   2              AND
   2          NOT .output_nam [nam$v_quoted]
   2      THEN
   2          no_expl_out_fields = TRUE;
   2  !
   2  ! Reload the RLF field. Another PARSE will be performed later in the routine
   2  ! COPY$OPN_OUTFIL and may take fields from the input resultant file string.
   2  !
   2
   2      output_nam [nam$l_rlf] = .temp_rlf;
   2
   2  !
   2  ! Return with a success code.
   2  !
   2
   2      RETURN ok;                                         ! Return successfully.
```

```
;  861        1391  2
;  862        1392  1      END;



                                                                   .PSECT   $PLIT$,NOWRT,NOEXE,2

                      45  4C  49  46  54  55  4F  00010 P.AAD:     .ASCII   \OUTFILE\                               ;
                                                  00017            .BLKB    1
                                        00000007  00018 P.AAC:     .LONG    7                                       ;
                                        00000000' 0001C            .ADDRESS P.AAD                                   ;

                                                                   .EXTRN   SYS$PARSE

                                                                   .PSECT   $CODE$,NOWRT,2

                                            007C  00000            .ENTRY   COPY$GET_OUTFIL, Save R2,R3,R4,R5,R6    ; 1220
                             5E            08  C2 00002            SUBL2    #8, SP                                  ;
        08         00        6E            00  2C 00005            MOVC5    #0, (SP), #0, #8, CLI_DESC              ; 1289
                                           6E     0000A                                                             ;
                      03     AE            02  90 0000B            MOVB     #2, CLI_DESC+3                          ; 1290
                             5E            5E  DD 0000F            PUSHL    SP                                      ; 1292
                                    0000'  CF  9F 00011            PUSHAB   P.AAC                                   ;
                   00000000G 00            02  FB 00015            CALLS    #2, CLI$GET_VALUE                       ;
                       0000G CF            6E  3C 0001C            MOVZWL   CLI_DESC, OUT_NAME_DESC                 ; 1298
              0000G  DF   04 BE            6E  28 00021            MOVC3    CLI_DESC, @CLI_DESC+4, @OUT_NAME_DESC+4 ; 1299
                       0000G CF            00  FB 00028            CALLS    #0, COPY$GET_GLOBAL_QUAL                ; 1303
                             56      04    AC  D0 0002D            MOVL     OUTPUT_FAB, R6                          ; 1318
   0050   8F         00      6E            00  2C 00031            MOVC5    #0, (SP), #0, #80, (R6)                 ;
                                           66     00038                                                             ;
                             66    5003    8F  B0 00039            MOVW     #20483, (R6)                            ;
                      04     A6 21000040   8F  D0 0003E            MOVL     #553648192, 4(R6)                       ;
                      16     A6    2011    8F  B0 00046            MOVW     #8209, 22(R6)                           ;
                      1F     A6            02  90 0004C            MOVB     #2, 31(R6)                              ;
                      24     A6      0C    AC  D0 00050            MOVL     OUTPUT_XABFHC, 36(R6)                   ;
                             52      08    AC  D0 00055            MOVL     OUTPUT_NAM, R2                          ;
                      28     A6            52  D0 00059            MOVL     R2, 40(R6)                              ;
                      2C     A6      04    AE  D0 0005D            MOVL     CLI_DESC+4, 44(R6)                      ;
                      34     A6            6E  90 00062            MOVB     CLI_DESC, 52(R6)                        ;
                                     0B    A2  94 00066            CLRB     11(R2)                                  ; 1326
                             53      10    A2  D0 00069            MOVL     16(R2), TEMP_RLF                        ; 1334
                                     10    A2  D4 0006D            CLRL     16(R2)                                  ; 1335
                                    0000G  CF  9F 00070            PUSHAB   COPY$OUTOPN_ERR                         ; 1343
                                           56  DD 00074            PUSHL    R6                                      ;
                   00000000G 00            02  FB 00076            CALLS    #2, SYS$PARSE                           ;
                             58            50  E9 0007D            BLBC     R0, 7$                                  ;
                             50      34    A2  9E 00080            MOVAB    52(R2), R0                              ; 1352
                 30          60            05  E0 00084            BBS      #5, (R0), 3$                            ;
                 24          60            04  E0 00088            BBS      #4, (R0), 2$                            ; 1353
                 20          60            03  E0 0008C            BBS      #3, (R0), 2$                            ; 1354
                 1C          60            12  E0 00090            BBS      #18, (R0), 2$                           ; 1355
                 18          60            02  E0 00094            BBS      #2, (R0), 2$                            ; 1356
                 14          60            01  E0 00098            BBS      #1, (R0), 2$                            ; 1357
                             11            60  E8 0009C            BLBS     (R0), 2$                                ; 1358
                 08          60            06  E0 0009F            BBS      #6, (R0), 1$                            ; 1359
                                           60  95 000A3            TSTB     (R0)                                    ; 1360
                                           04  19 000A5            BLSS     1$                                      ;
```

```
                                                                  C 16
COPYSPECS                                                         15-Sep-1984 23:42:51    VAX-11 Bliss-32 V4.0-742              Page  24
V04-000                                                           14-Sep-1984 12:14:19    [COPY.SRC]COPYSPECS.B32;1                  (5)

                          05            60      11  E1 000A7          BBC     #17, (R0), 2$                                   ; 1361
                                 0000G  CF      08  88 000AB  1$:     BISB2   #8, COPY$SEM_STATUS                             ; 1363
                          04            60      05  E0 000B0  2$:     BBS     #5, (R0), 3$                                    ; 1370
                          18            60      02  E0 000B4          BBS     #2, (R0), 6$                                    ;
                          04            60      04  E0 000B8  3$:     BBS     #4, (R0), 4$                                    ; 1372
                          10            60      01  E0 000BC          BBS     #1, (R0), 6$                                    ;
                                        04      60  E8 000C0  4$:     BLBS    (R0), 5$                                        ; 1374
                          09            60      03  E0 000C3          BBS     #3, (R0), 6$                                    ;
                          05            60      12  E0 000C7  5$:     BBS     #18, (R0), 6$                                   ; 1376
                                 0000G  CF      01  88 000CB          BISB2   #1, COPY$SEM_STATUS+3                           ; 1378
                                    10  A2      53  D0 000D0  6$:     MOVL    TEMP_RLF, 16(R2)                                ; 1384
                                        50      01  D0 000D4          MOVL    #1, R0                                          ; 1390
                                                    04 000D7          RET                                                     ;
                                                50  D4 000D8  7$:     CLRL    R0                                              ; 1392
                                                    04 000DA          RET                                                     ;

; Routine Size:  219 bytes,    Routine Base:  $CODE$ + 0181
```

```
1 GLOBAL ROUTINE copy$opn_outfil (output_fab, output_rab, input_fab, out_file_count) =
1                                                ! Opens the current output file
1
1 !++
1 ! Functional description:
1 !
1 !       This routine opens the current output file. If it is already open due
1 !       to input file concatenation, the output file RAB is simply disconnected from
1 !       the FAB to permit switching from block mode I/O to record mode I/O.
1 !
1 !       Many of the fields in the input FAB and XAB blocks are copied into the corresponding
1 !       output FAB and XAB blocks. Also, bits and values are set in the output XAB and FAB blocks
1 !       because of output file qualifiers specified on the command.
1 !
1 !       If the output file already exists, and is being overwritten, it is opened
1 !       for output. If the output file does not exist, it is allocated and then opened.
1 !
1 ! Calling sequence:
1 !
1 !       copy$opn_outfil (output_fab.ra.v, output_rab.ra.v, input_fab.ra.v, out_file_count.wl.r)
1 !
1 ! Input parameters:
1 !
1 !       output_fab      - the address of the FAB associated with the output file
1 !       output_rab      - the address of the RAB to be used with the output file
1 !       input_fab       - the address of the FAB associated with the input file
1 !
1 ! Implicit inputs:
1 !
1 !       copy$cli_status - the OUTFILE_OPEN bit indicates whether an output file is already open.
1 !                       - bits indicate the settings of the output file qualifiers
1 !
1 !       Fields from the input NAM and XAB block are used in the output NAM and XAB blocks.
1 !
1 ! Output parameters:
1 !
1 !       out_file_count  - a counter that is incremented if a new file is opened.
1 !
1 ! Implicit outputs:
1 !
1 !       copy$cli_status - OUTFILE_OPEN is set once the file is opened.
1 !                       - EXTEND_OUTFILE is set if the output file is being extended.
1 !
1 !       Fields are written in the output_fab and its associated NAM and XAB blocks.
1 !
1 !       out_name_desc   - a descriptor for the output file. Its length field is written.
1 !
1 !       When the output file name is parsed, various bits are set in
1 !       COPY$SEM_STATUS. These include:
1 !
1 !               wild_output     - output spec includes explicit wildcards
1 !               wild_out_ver    - explicit wildcard version number
1 !
1 ! Routine value
1 !
1 !       OK              - output file successfully created or readied for more output
1 !       NO_FILE         - output file could not be opened, created, or readied for output
```

```
!
! Side effects:
!
!       The routine SETUP_EXTEND is called if the output file is open. The value of this call
!               is returned to the caller.
!       The routine SETUP_OUTXAB is called to write most of the output XAB block fields.
!       Messages are output if a file was created during an APPEND command, if versions were
!               slipped under higher existing versions, or if files were replaced or overlaid.
!
!--

    BEGIN

    MAP
        output_fab      : REF BLOCK [, BYTE],                   ! FAB to use with output file
        output_rab      : REF BLOCK [, BYTE],                   ! RAB to use with output file
        input_fab       : REF BLOCK [, BYTE],                   ! FAB of the current input file
        out_file_count  : REF VECTOR;                           ! pointer to number of output files written

    BIND
        output_nam      =                                       ! output NAM block address
                .output_fab [fab$l_nam]         : BLOCK [, BYTE],
        output_xabfhc   =                                       ! output XAB file header characteristics block
                .output_fab [fab$l_xab]         : BLOCK [, BYTE],
        output_xaball   =                                       ! output XAB date block
                .output_xabfhc [xab$l_nxt]      : BLOCK [, BYTE],
        output_xabdat   =                                       ! output XAB date block
                .output_xaball [xab$l_nxt]      : BLOCK [, BYTE],
        output_xabrdt   =                                       ! output XAB date block
                .output_xabdat [xab$l_nxt]      : BLOCK [, BYTE],
        output_xabpro   =                                       ! output XAB date block
                .output_xabrdt [xab$l_nxt]      : BLOCK [, BYTE];

    LOCAL
        status;                                                 ! Status variable for calling semantic routine.

!
! If the output file is already open (due to input file concatenation), call a routine,
! SETUP_EXTEND, to prepare the file to contain more data.
!

    IF .outfile_open                                            ! If the output file is already open,
    THEN
        RETURN setup_extend (                                   !    call a routine to set the file up
                            .output_rab);                       !    to be extended.

!
! Copy a set of FAB attributes from the input to the output FAB.
!

    output_fab [fab$b_org] = .input_fab [fab$b_org];            ! The fields copied are file organization,
    output_fab [fab$b_rat] = .input_fab [fab$b_rat];            !    record attributes
    output_fab [fab$w_mrs] = .input_fab [fab$w_mrs];            !    maximum record size
    output_fab [fab$l_mrn] = .input_fab [fab$l_mrn];            !    maximum record number
    output_fab [fab$b_rfm] = .input_fab [fab$b_rfm];            !    record format
    output_fab [fab$b_fsz] = .input_fab [fab$b_fsz];            !    fixed control area size
    output_fab [fab$b_bks] = .input_fab [fab$b_bks];            !    bucket size
    output_fab [fab$w_gbc] = .input_fab [fab$w_gbc];    !    global buffer count

!
! If the input file has read or write checking options, copy them to the output file.
!

    output_fab [fab$l_fop] = .output_fab [fab$l_fop] OR ! OR together the current FOP output field
        (.input_fab [fab$l_fop] AND (fab$m_rck OR fab$m_wck));
                                                        !    and the read and write check bits of the
                                                        !    FOP input field.

!
! Decide on block or record I/O.
!

    IF .input_fab [fab$b_org] EQL fab$c_seq             ! If the input file is a sequential file,
    THEN
        output_fab [fab$v_bro] = TRUE                   !    then indicate mixed block and record I/O.
    ELSE
        BEGIN
        output_fab [fab$v_bio] = true;                  ! Otherwise, indicate only block I/O.
        output_fab [fab$v_bro] = false;                 ! and turn off block/record I/O
        END;

!
! Copy input blocksize for tapes. Otherwise let RMS set the output blocksize.
!

    IF .input_fab [$FAB_DEV (sqd)]                      ! If input device is a tape,
    THEN
        output_fab [fab$w_bls] = .input_fab [fab$w_bls] !    then copy the blocksize to the output FAB.
    ELSE
        output_fab [fab$w_bls] = 0;                     ! Otherwise, let RMS choose blocksize.

!
! Test the expanded name string for the output file. Does it contain wildcards? If so,
! is there an explicit wildcard version number?
!

    IF .output_nam [nam$v_wildcard]                     ! If there were any wildcards,
    THEN
        wild_output = TRUE;                             !    set flag saying that the file specification
                                                        !    contained some wildcard fields.

    IF .output_nam [nam$v_wild_ver]                     ! If the version number is a wildcard,
    THEN
       wild_out_ver = TRUE                              !    output version number, remember it.
    ELSE
        IF .output_nam [nam$v_exp_ver]                  ! Otherwise, see if an explicit version number was s
        THEN
            exp_out_ver = TRUE;                         !    If so, set the EXP_OUT_VER flag.

!
! Reparse the output string with a wildcard version number, if this is not
! an APPEND operation and one of the following cases is true:
!       - no output file name, type or version number was given
!         (e.g. COPY x.x [dir])
!           - wild or explicit version numbers were given for the input file, but
!             the version field for the output file was not specified
!             (e.g. COPY x.x;* a.a)
!           - the output spec is wild (e.g. COPY x.x *, or COPY x.x *.*)

    IF NOT .append_command
                AND
        (.no_output_spec
                OR
        ((.wild_inp_ver OR .exp_inp_ver)
        AND NOT .output_nam [nam$v_wild_ver]
        AND NOT .output_nam [nam$v_exp_ver])
                OR
        (NOT .output_nam [nam$v_exp_ver]
        AND (.output_nam [nam$v_wild_type] OR NOT .output_nam [nam$v_exp_type])
        AND .output_nam [nam$v_wild_name]))
    THEN
        BEGIN                                           ! Then provide a default name string
        output_fab [fab$l_dna] = UPLIT (';*');          !     of an explicit output wildcard
        output_fab [fab$b_dns] = 2;                     !     version number,
        END;


    ! Now $PARSE (this may be a reparse) the output file specification.
    !
    IF NOT $RMS_PARSE ( FAB = .output_fab, ERR = copy$outopn_err)
    THEN
            RETURN no_file;                             ! On failure, return with an error code.


    !
    ! No director wildcards allowed to remain at this time
    !
    BEGIN
    BIND
        lastchar = .output_nam[nam$l_dir] + .output_nam[nam$b_dir] - 2 : byte;
    IF .lastchar EQL %C'*' OR
        .lastchar EQL %C'.'
    THEN
        BEGIN
        LOCAL
           outputstr : vector[2];
        outputstr[0] = .output_nam [nam$b_esl];
        outputstr[1] = .output_nam [nam$l_esa];
        PUT_MESSAGE( MSG$_SYNTAX,
                     1,
                     outputstr,
                     0 );
        RETURN no_file;
        END;
    END;

    !
    ! See if the output file fits the criteria given on the command line.
    !
    IF NOT (status = copy$check_file_for_match())
    THEN
```

```
        RETURN .status;

    !
    ! Call the routine SETUP_OUTXAB to copy output XAB fields from the corresponding input XAB fields.
    !

    setup_outxab (                                             ! Write output XAB fields by calling
                    .output_fab,                               !     a routine that selects the necessary fields fro
                    .input_fab);                               !     the input FAB and writes them into the output F

    !
    ! Call the routine APPLY_OUT_QUAL to write RMS fields according to output parameter qualifiers.
    !

    apply_out_qual (                                           ! Process output file qualifiers
                    .output_fab);

    !
    ! Call the routine COPY$SEMANTICS to determine the semantic effects of
    ! this particular combination of input and output file specifications and qualifiers.
    !

    IF NOT copy$semantics (                                    ! Decide what semantic behavior is required.
                             copy$sem_status,                  !     Pass the status variable copy$sem_status,
                             .input_fab,                       !     the input FAB block address,
                             .output_fab)                      !     and the output FAB block address.
    THEN                                                       ! If the input/output spec combination makes no sens
        RETURN no_file;                                        !     then return with error status code.

    !
    ! Perform special XAB setup if a concatenated file is being created.
    !

    IF (.append_command                                        ! If appending to existing file,
        OR .concat_follows                                     !     or concatenating
        OR NOT .no_expl_out_fields                             !     or if explicit field in output spec
        OR NOT .input_fab [$fab_dev (fod)])                    !     or the input device is not file structured,
    THEN
        output_xaball [xab$l_nxt] = .output_xabrdt [xab$l_nxt] ! Do not provide any date information
    ELSE
        BEGIN
        output_xaball [xab$l_nxt] = output_xabdat;             ! Otherwise, include the output date/time XAB block
        output_xabdat [xab$l_nxt] = output_xabrdt;             !     and the revision date/time XAB block;
        END;

    !
    ! Create (or simply open) the output file.
    !

    extend_outfile = FALSE;                                    ! Assume that the output file is not being extended.

    !
    ! If a file needn't be created, just open an existing file.
    !

    IF .append_command AND                                     ! If this is an APPEND command and
        NOT .new_version_qual                                  !     and output file creation was not requested,
  2     THEN
  3         BEGIN
  4             IF NOT (status = $RMS_OPEN ( FAB = .output_fab, ERR = copy$outopn_err))
  3             THEN RETURN .status;
  3         END
  2     ELSE
  3         BEGIN
  3         status = $RMS_CREATE (FAB = .output_fab);       ! Else, create (or open if it exists) file
  3
  3 !
  3 ! If the file could not be created as a contiguous file because the disk was too full,
  3 ! then try to create it contiguous best try.
  3 !
  3
  3         IF .status EQL rms$_ful
  3             AND .output_xaball [xab$v_ctg]
  4             AND NOT qualifier_active( contig_qual, loc_contig_qual, neg_contig_qual )
  3         THEN
  4             BEGIN
  4             output_xaball [xab$v_ctg] = FALSE;          !    then turn off the contiguous indicator,
  4             output_xaball [xab$v_cbt] = TRUE;           !    turn on the contiguous best try indicator,
P 4             status = $RMS_CREATE (                      !    and retry the create.
P 4                                     FAB = .output_fab,  ! Specify the address of the FAB block
  4                                     ERR = copy$outopn_err);
  4             IF .status                                  ! If contig-best-try ok,
  4             THEN
  4                 put_message (msg$_cbt);                 ! then issue message
  4             END                                         !    and an error action routine.
  3         ELSE
  3         IF NOT .status                                  ! Else, if error,
  3         THEN
  3             copy$outopn_err (.output_fab);              ! issue error message
  3
  3 !
  3 ! Change the RMS return status to "created" if indeed the file was created.
  3 !
  3
  3         IF NOT .output_fab [fab$v_cif] AND              ! Since RMS returns RMS$_NORMAL whether or not the
  3             .status EQL rms$_normal                     !    file was created, for internal reporting, chang
  3         THEN                                            !    the status code to RMS$_CREATED if appropriate.
  3             status = rms$_created;                      !    (I.e., if the file was created.)
  3
  3 ! If the file was indeed created, issue a $DISPLAY to obtain information
  3 ! about the newly created file.
  3
  3         IF NOT .status                                  ! If the open or create failed,
  3         THEN
  3             RETURN no_file;                             !    then return an error status code.
  3
  4         IF NOT $RMS_DISPLAY (FAB = .output_fab)
  3         THEN
  3             copy$outopn_err (.output_fab);
  3 !
  3 ! If the output file was copied to a 10,20 or RT node and it was forced to a
  3 ! stream format file, then (if the /LOG qualifier was specified) warn the user
  3 ! of the conversion.
  3 !
```

```
      IF .status EQL rms$_cre_stm AND .LOG_MSG_QUAL
      THEN
         BEGIN
         out_name_desc [0] = .output_nam [nam$b_rsl]; ! Store the length of the filespec
         put_message (msg$_createdstm,1,              ! Issue the message
                      out_name_desc);
         status = rms$_created;                       ! Change the status as above
         END;
      END;

  outfile_open = TRUE;                                 ! Otherwise, set a flag saying that an output file i
  out_name_desc [0] = .output_nam [nam$b_rsl];         !    and store the length of the file specification.

!
! Clean up the output open procedure by reporting to the user if necessary and
! updating more fields.
!

  SELECTONE .status OF                                 ! Select additional processing based on the

      SET                                              !    RMS completion code from the OPEN or CREATE.

      [rms$_created]:                                  ! Output file was created.
          BEGIN
          out_file_count [0] =                         !    Update count of files created.
              .out_file_count [0] + 1;

          IF .append_command                           !    If this is an APPEND command,
          THEN
              copy$log_msg (                           !    send the following message to the user:
                          msg$_created);               !        "<file-name> created" because creation is u

          IF .output_nam [nam$v_highver] AND           !    If a higher version of this file exists,
              NOT .quiet_slip                          !    and warnings about versions are not suppressed,
          THEN
              put_message (                            !    send the following message to the user:
                              msg$_highver, 1,         !        "higher version of <file-name> exists"
                              out_name_desc);          !    because this may cause version confusion.

          END;


      [rms$_supersede]:                                ! Output file caused deletion of file of same name.
          BEGIN
          out_file_count [0] =                         !    Update count of files created.
              .out_file_count [0] + 1;

          copy$log_msg (                               !    Send the following message to the user:
                      msg$_replaced);                  !        "<file-name> replaced" because
                                                       !        supersession is unusual.

          END;


      [rms$_normal]:                                   ! Output file existed previously and was opened.
          BEGIN
          IF .append_command                           !    If this is an APPEND command,
; 1263        1792  3              THEN
; 1264        1793  4                  BEGIN
; 1265        1794  4                  extend_outfile = TRUE;                    !     set a flag saying that the file is being extend
; 1266        1795  4
; 1267        1796  4                  output_xaball [xab$l_alq] =              !     Calculate the necessary extension quantity
; 1268        1797  4                      copy$calc_alq ();                     !     with a call to COPY$CALC_ALQ.
; 1269        1798  4
; 1270        1799  4                  IF .output_xaball [xab$l_alq] NEQ 0      !     If the extension quantity is not null,
; 1271        1800  4                  THEN
; 1272      P 1801  4                      IF NOT $RMS_EXTEND (                  !     then try to extend the file.
; 1273      P 1802  4                                      FAB = .output_fab,
; 1274        1803  5                                      ERR = copy$outopn_err)
; 1275        1804  4                      THEN                                  !     If the extend fails,
; 1276        1805  4                          RETURN no_file;                   !     then return an error status code.
; 1277        1806  4
; 1278        1807  4                  END
; 1279        1808  4
; 1280        1809  3              ELSE                                          !     If this is a COPY command,
; 1281        1810  4                  BEGIN
; 1282        1811  4                  copy$log_msg (                            !     send the following message to the user:
; 1283        1812  4                              msg$_overlay);                !         "<file-name> being overwritten"
; 1284        1813  4
; 1285        1814  4 ! ******
; 1286        1815  4 !       Omitted here is the revision of the output file's attributes. Ward had this
; 1287        1816  4 !       commented out.
; 1288        1817  4 ! ******
; 1289        1818  4
; 1290        1819  3                  END;
; 1291        1820  3
; 1292        1821  2              END;
; 1293        1822  2
; 1294        1823  2          TES;                                      ! End of SELECT expression.
; 1295        1824  2
; 1296        1825  2 !
; 1297        1826  2 ! Return to the caller with a success status code.
; 1298        1827  2 !
; 1299        1828  2
; 1300        1829  2      RETURN ok;                                    ! Return with a success code.
; 1301        1830  1      END;


                                                    .PSECT  $PLIT$,NOWRT,NOEXE,2

                  00  00  2A  3B  00020 P.AAE:      .ASCII  \;*\<0><0>                                  ;

                                                    .EXTRN  SYS$CREATE, SYS$DISPLAY
                                                    .EXTRN  SYS$EXTEND

                                                    .PSECT  $CODE$,NOWRT,2

                               0FFC 00000           .ENTRY  COPY$OPN_OUTFIL, Save R2,R3,R4,R5,R6,R7,R8,-; 1393
                                                            R9,R10,R11                                  ;
              5B      0000G  CF  9E 00002           MOVAB   COPY$CLI_STATUS, R11                        ;
              5A      0000G  CF  9E 00007           MOVAB   COPY$SEM_STATUS, R10                        ;
              5E         08  C2  0000C              SUBL2   #8, SP                                      ;
              53         04  AC  D0 0000F           MOVL    OUTPUT_FAB, R3                              ; 1471
```

```
                  56        28   A3   D0 00013           MOVL    40(R3), R6                                ;
                  50        24   A3   D0 00017           MOVL    36(R3), R0                                ; 1473
                  55        04   A0   D0 0001B           MOVL    4(R0), R5                                 ; 1475
                  58        04   A5   D0 0001F           MOVL    4(R5), R8                                 ; 1477
                  59        04   A8   D0 00023           MOVL    4(R8), R9                                 ; 1479
        09        02   AA             01   E1 00027      BBC     #1, COPY$SEM_STATUS+2, 1$                 ; 1491
                            08   AC   DD 0002C           PUSHL   OUTPUT_RAB                                ; 1494
             0000V   CF             01   FB 0002F        CALLS   #1, SETUP_EXTEND                          ;
                                      04 00034           RET                                               ; 1493
                  52        0C   AC   D0 00035 1$:       MOVL    INPUT_FAB, R2                             ; 1500
             1D   A3        1D   A2   B0 00039           MOVW    29(R2), 29(R3)                            ;
             36   A3        36   A2   B0 0003E           MOVW    54(R2), 54(R3)                            ; 1502
             38   A3        38   A2   D0 00043           MOVL    56(R2), 56(R3)                            ; 1503
             1F   A3        1F   A2   90 00048           MOVB    31(R2), 31(R3)                            ; 1504
             3E   A3        3E   A2   B0 0004D           MOVW    62(R2), 62(R3)                            ; 1506
             48   A3        48   A2   B0 00052           MOVW    72(R2), 72(R3)                            ; 1507
        50        04   A2 FF7FFDFF    8F   CB 00057      BICL3   #-8389121, 4(R2), R0                      ; 1514
             04   A3                  50   C8 00060      BISL2   R0, 4(R3)                                 ;
                            1D   A2   95 00064           TSTB    29(R2)                                    ; 1522
                                 07   12 00067           BNEQ    2$                                        ;
             16   A3        40   8F   88 00069           BISB2   #64, 22(R3)                               ; 1524
                                 09   11 0006E           BRB     3$                                        ;
             16   A3             20   88 00070 2$:       BISB2   #32, 22(R3)                               ; 1527
             16   A3        40   8F   8A 00074           BICB2   #64, 22(R3)                               ; 1528
        07   40   A2             05   E1 00079 3$:       BBC     #5, 64(R2), 4$                            ; 1535
             3C   A3        3C   A2   B0 0007E           MOVW    60(R2), 60(R3)                            ; 1537
                                 03   11 00083           BRB     5$                                        ;
                            3C   A3   B4 00085 4$:       CLRW    60(R3)                                    ; 1539
                  57        34   A6   9E 00088 5$:       MOVAB   52(R6), R7                                ; 1546
                  04        01   A7   E9 0008C           BLBC    1(R7), 6$                                 ;
                  6A        40   8F   88 00090           BISB2   #64, COPY$SEM_STATUS                      ; 1548
        06        67             03   E1 00094 6$:       BBC     #3, (R7), 7$                              ; 1551
                  6A        80   8F   88 00098           BISB2   #128, COPY$SEM_STATUS                     ; 1553
                                 06   11 0009C           BRB     8$                                        ;
                  03             67   E9 0009E 7$:       BLBC    (R7), 8$                                  ; 1555
                  6A             04   88 000A1           BISB2   #4, COPY$SEM_STATUS                       ; 1557
                  2C             6B   E8 000A4 8$:       BLBS    COPY$CLI_STATUS, 13$                      ; 1569
        1E        6A             03   E0 000A7           BBS     #3, COPY$SEM_STATUS, 12$                  ; 1571
        04        6A             05   E0 000AB           BBS     #5, COPY$SEM_STATUS, 9$                   ; 1573
        07        6A             01   E1 000AF           BBC     #1, COPY$SEM_STATUS, 10$                  ;
        03        67             03   E0 000B3 9$:       BBS     #3, (R7), 10$                             ; 1574
                  0F             67   E9 000B7           BLBC    (R7), 12$                                 ; 1575
                  16             67   E8 000BA 10$:      BLBS    (R7), 13$                                 ; 1577
        04        67             04   E0 000BD           BBS     #4, (R7), 11$                             ; 1578
        0E        67             01   E0 000C1           BBS     #1, (R7), 13$                             ;
        0A        67             05   E1 000C5 11$:      BBC     #5, (R7), 13$                             ; 1579
             30   A3      0000'  CF   9E 000C9 12$:      MOVAB   P.AAE, 48(R3)                             ; 1582
             35   A3             02   90 000CF           MOVB    #2, 53(R3)                                ; 1583
                          0000G  CF   9F 000D3 13$:      PUSHAB  COPY$OUTOPN_ERR                           ; 1589
                                 53   DD 000D7           PUSHL   R3                                        ;
          00000000G  00          02   FB 000D9           CALLS   #2, SYS$PARSE                             ;
                  37             50   E9 000E0           BLBC    R0, 15$                                   ;
                  50        3A   A6   9A 000E3           MOVZBL  58(R6), R0                                ; 1599
                  50        48   A6   C0 000E7           ADDL2   72(R6), R0                                ;
                  2A        FE   A0   91 000EB           CMPB    -2(R0), #42                               ; 1600
                                 06   13 000EF           BEQL    14$                                       ;
                  2E        FE   A0   91 000F1           CMPB    -2(R0), #46                               ; 1601
```

```
                                                   M 16
COPYSPECS                                          15-Sep-1984 23:42:51    VAX-11 Bliss-32 V4.0-742           Page  34
V04-000                                            14-Sep-1984 12:14:19    [COPY.SRC]COPYSPECS.B32;1                (6)

                                         26  12 000F5          BNEQ    16$                                    ;
                        6E         0B    A6  9A 000F7 14$:     MOVZBL  11(R6), OUTPUTSTR                      ; 1606
           04           AE         0C    A6  D0 000FB          MOVL    12(R6), OUTPUTSTR+4                    ; 1607
                                         7E  D4 00100          CLRL    -(SP)                                  ; 1611
                                   04    AE  9F 00102          PUSHAB  OUTPUTSTR                              ;
                                         01  DD 00105          PUSHL   #1                                     ;
                        7E       10FC    8F  3C 00107          MOVZWL  #4348, -(SP)                           ;
        0000G           CF               01  FB 0010C          CALLS   #1, COPY$MSG_NUMBER                    ;
                                         50  DD 00111          PUSHL   R0                                     ;
    00000000G           00               04  FB 00113          CALLS   #4, LIB$STOP                           ;
                                       01C7  31 0011A 15$:     BRW     35$                                    ; 1612
        0000G           CF               00  FB 0011D 16$:     CALLS   #0, COPY$CHECK_FILE_FOR_MATCH          ; 1619
                        54               50  D0 00122          MOVL    R0, STATUS                             ;
                        5E               54  E9 00125          BLBC    STATUS, 20$                            ;
                                         52  DD 00128          PUSHL   R2                                     ; 1629
                                         53  DD 0012A          PUSHL   R3                                     ; 1628
        0000V           CF               02  FB 0012C          CALLS   #2, SETUP_OUTXAB                       ;
                                         53  DD 00131          PUSHL   R3                                     ; 1636
        0000V           CF               01  FB 00133          CALLS   #1, APPLY_OUT_QUAL                     ;
                                         0C  BB 00138          PUSHR   #^M<R2,R3>                             ; 1645
                                         5A  DD 0013A          PUSHL   R10                                    ; 1643
        0000G           CF               03  FB 0013C          CALLS   #3, COPY$SEMANTICS                     ;
                        D6               50  E9 00141          BLBC    R0, 15$                                ;
                        0E               6B  E8 00144          BLBS    COPY$CLI_STATUS, 17$                   ; 1654
  09       02           AA               03  E0 00147          BBS     #3, COPY$SEM_STATUS+2, 17$             ; 1655
                        05         03    AA  E9 0014C          BLBC    COPY$SEM_STATUS+3, 17$                 ; 1656
  07       41           A2               06  E0 00150          BBS     #6, 65(R2), 18$                        ; 1657
           04           A5         04    A9  D0 00155 17$:     MOVL    4(R9), 4(R5)                           ; 1659
                                         08  11 0015A          BRB     19$                                    ;
           04           A5               58  D0 0015C 18$:     MOVL    R8, 4(R5)                              ; 1662
           04           A8               59  D0 00160          MOVL    R9, 4(R8)                              ; 1663
           02           AA         80    8F  8A 00164 19$:     BICB2   #128, COPY$SEM_STATUS+2                ; 1670
                        1E               6B  E9 00169          BLBC    COPY$CLI_STATUS, 21$                   ; 1676
  1A                    6B               04  E0 0016C          BBS     #4, COPY$CLI_STATUS, 21$               ; 1677
                                0000G    CF  9F 00170          PUSHAB  COPY$OUTOPN_ERR                        ; 1680
                                         53  DD 00174          PUSHL   R3                                     ;
    00000000G           00               02  FB 00176          CALLS   #2, SYS$OPEN                           ;
                        54               50  D0 0017D          MOVL    R0, STATUS                             ;
                        03               54  E9 00180          BLBC    STATUS, 20$                            ;
                                       00C9  31 00183          BRW     28$                                    ;
                        50               54  D0 00186 20$:     MOVL    STATUS, R0                             ; 1681
                                         04 00189              RET                                            ;
                                         53  DD 0018A 21$:     PUSHL   R3                                     ; 1685
    00000000G           00               01  FB 0018C          CALLS   #1, SYS$CREATE                         ;
                        54               50  D0 00193          MOVL    R0, STATUS                             ;
    00018544            8F               54  D1 00196          CMPL    STATUS, #99652                         ; 1692
                                         45  12 0019D          BNEQ    23$                                    ;
                                   08    A5  95 0019F          TSTB    8(R5)                                  ; 1693
                                         40  18 001A2          BGEQ    23$                                    ;
  05       02           AB               03  E1 001A4          BBC     #3, COPY$CLI_STATUS+2, 22$             ; 1694
  36       02           AB               06  E1 001A9          BBC     #6, COPY$CLI_STATUS+2, 23$             ;
  31       02           AB               05  E0 001AE 22$:     BBS     #5, COPY$CLI_STATUS+2, 23$             ;
           08           A5         80    8F  8A 001B3          BICB2   #128, 8(R5)                            ; 1697
           08           A5               20  88 001B8          BISB2   #32, 8(R5)                             ; 1698
                                0000G    CF  9F 001BC          PUSHAB  COPY$OUTOPN_ERR                        ; 1701
                                         53  DD 001C0          PUSHL   R3                                     ;
    00000000G           00               02  FB 001C2          CALLS   #2, SYS$CREATE                         ;
```

```
                    54              50  D0 001C9          MOVL    R0, STATUS                                ;
                    1F              54  E9 001CC          BLBC    STATUS, 24$                               ; 1702
                    7E      1288    8F  3C 001CF          MOVZWL  #4744, -(SP)                              ; 1704
          0000G     CF              01  FB 001D4          CALLS   #1, COPY$MSG_NUMBER                       ;
                                    50  DD 001D9          PUSHL   R0                                        ;
      00000000G     00              01  FB 001DB          CALLS   #1, LIB$SIGNAL                            ;
                                    0A  11 001E2          BRB     24$                                       ; 1692
                    07              54  E8 001E4   23$:   BLBS    STATUS, 24$                               ; 1707
                                    53  DD 001E7          PUSHL   R3                                        ; 1709
          0000G     CF              01  FB 001E9          CALLS   #1, COPY$OUTOPN_ERR                       ;
   10          07   A3              01  E0 001EE   24$:   BBS     #1, 7(R3), 25$                            ; 1715
      00010001      8F              54  D1 001F3          CMPL    STATUS, #65537                            ; 1716
                                    07  12 001FA          BNEQ    25$                                       ;
                    54  00010619    8F  D0 001FC          MOVL    #67097, STATUS                            ; 1718
                    03              54  E8 00203   25$:   BLBS    STATUS, 26$                               ; 1723
                                  00DB  31 00206          BRW     35$                                       ;
                                    53  DD 00209   26$:   PUSHL   R3                                        ; 1727
      00000000G     00              01  FB 0020B          CALLS   #1, SYS$DISPLAY                           ;
                    07              50  E8 00212          BLBS    R0, 27$                                   ;
                                    53  DD 00215          PUSHL   R3                                        ; 1729
          0000G     CF              01  FB 00217          CALLS   #1, COPY$OUTOPN_ERR                       ;
      00018069      8F              54  D1 0021C   27$:   CMPL    STATUS, #98409                            ; 1735
                                    2A  12 00223          BNEQ    28$                                       ;
   26               6B              01  E1 00225          BBC     #1, COPY$CLI_STATUS, 28$                  ;
          0000G     CF        03    A6  9A 00229          MOVZBL  3(R6), OUT_NAME_DESC                      ; 1738
                            0000G   CF  9F 0022F          PUSHAB  OUT_NAME_DESC                             ; 1740
                                    01  DD 00233          PUSHL   #1                                        ;
                    7E      12FB    8F  3C 00235          MOVZWL  #4859, -(SP)                              ;
          0000G     CF              01  FB 0023A          CALLS   #1, COPY$MSG_NUMBER                       ;
                                    50  DD 0023F          PUSHL   R0                                        ;
      00000000G     00              03  FB 00241          CALLS   #3, LIB$SIGNAL                            ;
                    54  00010619    8F  D0 00248          MOVL    #67097, STATUS                            ; 1741
             02     AA              02  88 0024F   28$:   BISB2   #2, COPY$SEM_STATUS+2                     ; 1745
          0000G     CF        03    A6  9A 00253          MOVZBL  3(R6), OUT_NAME_DESC                      ; 1746
      00010619      8F              54  D1 00259          CMPL    STATUS, #67097                            ; 1757
                                    33  12 00260          BNEQ    30$                                       ;
                              10    BC  D6 00262          INCL    @OUT_FILE_COUNT                           ; 1760
                    0A              6B  E9 00265          BLBC    COPY$CLI_STATUS, 29$                      ; 1762
                    7E      1073    8F  3C 00268          MOVZWL  #4211, -(SP)                              ; 1764
          0000G     CF              01  FB 0026D          CALLS   #1, COPY$LOG_MSG                          ;
                                    67  B5 00272   29$:   TSTW    (R7)                                      ; 1767
                                    6A  18 00274          BGEQ    34$                                       ;
                    66        02    AA  E8 00276          BLBS    COPY$SEM_STATUS+2, 34$                    ; 1768
                            0000G   CF  9F 0027A          PUSHAB  OUT_NAME_DESC                             ; 1772
                                    01  DD 0027E          PUSHL   #1                                        ;
                    7E      1148    8F  3C 00280          MOVZWL  #4424, -(SP)                              ;
          0000G     CF              01  FB 00285          CALLS   #1, COPY$MSG_NUMBER                       ;
                                    50  DD 0028A          PUSHL   R0                                        ;
      00000000G     00              03  FB 0028C          CALLS   #3, LIB$SIGNAL                            ;
                                    4B  11 00293          BRB     34$                                       ; 1753
      00010631      8F              54  D1 00295   30$:   CMPL    STATUS, #67121                            ; 1777
                                    0A  12 0029C          BNEQ    31$                                       ;
                              10    BC  D6 0029E          INCL    @OUT_FILE_COUNT                           ; 1780
                    7E      10BB    8F  3C 002A1          MOVZWL  #4283, -(SP)                              ; 1782
                                    33  11 002A6          BRB     33$                                       ;
      00010001      8F              54  D1 002A8   31$:   CMPL    STATUS, #65537                            ; 1789
                                    2F  12 002AF          BNEQ    34$                                       ;
```

```
                 22              6B  E9 002B1            BLBC    COPY$CLI_STATUS, 32$                       ; 1791
          02     AA        80    8F  88 002B4            BISB2   #128, COPY$SEM_STATUS+2                    ; 1794
       0000G     CF              00  FB 002B9            CALLS   #0, COPY$CALC_ALQ                          ; 1797
          10     A5              50  D0 002BE            MOVL    R0, 16(R5)                                 ;
                                 1C  13 002C2            BEQL    34$                                        ; 1799
                        0000G    CF  9F 002C4            PUSHAB  COPY$OUTOPN_ERR                            ; 1803
                                 53  DD 002C8            PUSHL   R3                                         ;
   00000000G     00              02  FB 002CA            CALLS   #2, SYS$EXTEND                             ;
                 0C              50  E8 002D1            BLBS    R0, 34$                                    ;
                                 0E  11 002D4            BRB     35$                                        ; 1805
                 7E      10AB    8F  3C 002D6 32$:       MOVZWL  #4267, -(SP)                               ; 1811
       0000G     CF              01  FB 002DB 33$:       CALLS   #1, COPY$LOG_MSG                           ;
                 50              01  D0 002E0 34$:       MOVL    #1, R0                                     ; 1829
                                     04 002E3            RET                                                ;
                                 50  D4 002E4 35$:       CLRL    R0                                         ; 1830
                                     04 002E6            RET                                                ;

; Routine Size:  743 bytes,    Routine Base:  $CODE$ + 025C
```

```
1 ROUTINE setup_extend (output_rab) =                      ! Setup a file to be extended.
1
1 !++
1 ! Functional description:
1 !
1 !       This routine takes an open file and prepares it to be extended.
1 !
1 !       First, a DISCONNECT is performed. This permits switching from block mode I/O
1 !       to record mode I/O, if desired. Then update the output file allocation information,
1 !       set a bit in COPY$CLI_STATUS saying that the file is being extended, calculate
1 !       the file extension quantity, and extend the file.
1 !
1 ! Calling sequence:
1 !
1 !       setup_extend (output_rab.ra.v)
1 !
1 ! Input parameters
1 !
1 !       output_rab      - the RAB connected to the output FAB
1 !
1 ! Implicit inputs
1 !
1 !       The FAB and XAB blocks associated with the specified output RAB block.
1 !
1 ! Output parameters
1 !
1 !       none
1 !
1 ! Implicit outputs
1 !
1 !       The allocation information in the FAB is updated.
1 !       The EXTEND_OUTFILE bit in COPY$CLI_STATUS is set.
1 !       The ALQ field in the output XAB block is set to an appropriate extension quantity.
1 !
1 ! Routine value
1 !
1 !       OK              - success
1 !       NO_FILE         - failure
1 !
1 ! Side effects
1 !
1 !       If the file cannot be extended, the file is closed.
1 !
1 !--
1
2     BEGIN
2
2     MAP
2       output_rab      : REF BLOCK [, BYTE];               ! output FAB of the open output file
2
2     BIND
2       output_fab      =                                   ! associated output FAB block
2               .output_rab [rab$l_fab]         : BLOCK [, BYTE],
2       output_xabfhc   =                                   ! associated output XAB block
2               .output_fab [fab$l_xab]         : BLOCK [, BYTE],
2       output_xaball   =                                   !   second XAB in XAB chain
2               .output_xabfhc [xab$l_nxt]      : BLOCK [, BYTE];
```

```
    LOCAL
        status;                                         ! Holds RMS status values

    ! See if the input file fits the criteria given on the command line.
    !
    IF NOT (status = copy$check_file_for_match())
    THEN
        RETURN .status;

    ! Disconnect the RAB from the FAB. On error, close the file and return
    ! with error status code.
    !
    IF NOT $RMS_DISCONNECT (                            ! Disconnect the output file RAB from its FAB.
                            RAB = .output_rab,          !     Specify the RAB block address
                            ERR = copy$oclose_err)      !     and an error routine.
    THEN
        BEGIN                                           ! If the DISCONNECT fails,
        copy$close_outf (                               !     close the output file,
                            output_fab);
        RETURN no_file;                                 !     and return with an error code.
        END;
!
! Shortening the XAB chain to include only the FHC (file header characteristics) XAB,
! call the RMS function $DISPLAY to update the output file allocation information
! as recorded in the XABFHC.
!

    output_xabfhc [xab$l_nxt] = 0;                      ! Leave only the FHC XAB on the XAB chain.

    status = $RMS_DISPLAY (                             ! Call DISPLAY to update the XAB information
                            FAB = output_fab,           !     about the file's allocation.
                            ERR = copy$outopn_err);     !     Specify an error action routine.

    output_xabfhc [xab$l_nxt] = output_xaball;          ! Restore the XAB chain.
!
! See if the $DISPLAY function succeeded. If not, close the output file and return
! an error status code.
!

    IF NOT .status                                      ! If the $DISPLAY function failed,
    THEN
        BEGIN
        copy$close_outf (                               !     then close the output file,
                            output_fab);
        RETURN no_file;                                 !     and return an error status code.
        END;
!
! Set the bit in COPY$CLI_STATUS that indicates that the file is to be extended.
!

    extend_outfile = TRUE;                              ! Set EXTEND_OUTFILE bit.
```

```
; 1417        1945  2
; 1418        1946  2 !
; 1419        1947  2 ! Calculate the file extension quantity and extend the file with an RMS $EXTEND function call.
; 1420        1948  2 ! The routine COPY$CALC_ALQ does the calculation. It returns a "zero" in the following cases:
; 1421        1949  2 !
; 1422        1950  2 !     The output file is on a magtape or a nonfile-structured device.
; 1423        1951  2 !     The output file is already long enough to hold the size of the file to be appended.
; 1424        1952  2 !
; 1425        1953  2
; 1426        1954  2    output_xaball [xab$l_alq] = copy$calc_alq ();          ! Setup the output file extension quantity in the XA
; 1427        1955  2
; 1428        1956  2    IF .output_xaball [xab$l_alq] EQL 0                    ! If the input file is of zero length,
; 1429        1957  2    THEN
; 1430        1958  2        RETURN ok;                                         !     then return with success code.
; 1431        1959  2
; 1432      P 1960  2    IF $RMS_EXTEND (                                       ! If the output file can be extended successfully,
; 1433      P 1961  2                    FAB = output_fab,
; 1434        1962  3                    ERR = copy$outopn_err)                 !     (specify an error routine)
; 1435        1963  2    THEN
; 1436        1964  2        RETURN ok                                          !     then return with success code.
; 1437        1965  2    ELSE
; 1438        1966  2        RETURN no_file;                                    ! Otherwise, return with error code.
; 1439        1967  2
; 1440        1968  1    END;


                                                      .EXTRN  SYS$DISCONNECT

                                    007C 00000 SETUP_EXTEND:
                                                      .WORD   Save R2,R3,R4,R5,R6                             ; 1831
               54       04   AC   D0 00002            MOVL    OUTPUT_RAB, R4                                  ; 1883
               55       3C   A4   D0 00006            MOVL    60(R4), R5                                      ;
               52       24   A5   D0 0000A            MOVL    36(R5), R2                                      ; 1885
               53       04   A2   D0 0000E            MOVL    4(R2), R3                                       ; 1887
       0000G   CF            00   FB 00012            CALLS   #0, COPY$CHECK_FILE_FOR_MATCH                   ; 1895
               56            50   D0 00017            MOVL    R0, STATUS                                      ;
               04            56   E8 0001A            BLBS    STATUS, 1$                                      ;
               50            56   D0 0001D            MOVL    STATUS, R0                                      ; 1897
                                  04 00020            RET                                                     ;
                    0000G   CF    9F 00021 1$:        PUSHAB  COPY$OCLOSE_ERR                                 ; 1905
                            54    DD 00025            PUSHL   R4                                              ;
    00000000G  00            02   FB 00027            CALLS   #2, SYS$DISCONNECT                              ;
               1A            50   E9 0002E            BLBC    R0, 2$                                          ;
                        04   A2   D4 00031            CLRL    4(R2)                                           ; 1919
                    0000G   CF    9F 00034            PUSHAB  COPY$OUTOPN_ERR                                 ; 1923
                            55    DD 00038            PUSHL   R5                                              ;
    00000000G  00            02   FB 0003A            CALLS   #2, SYS$DISPLAY                                 ;
               56            50   D0 00041            MOVL    R0, STATUS                                      ;
          04   A2            53   D0 00044            MOVL    R3, 4(R2)                                       ; 1925
               09            56   E8 00048            BLBS    STATUS, 3$                                      ; 1932
                             55   DD 0004B 2$:        PUSHL   R5                                              ; 1935
       0000G   CF            01   FB 0004D            CALLS   #1, COPY$CLOSE_OUTF                             ;
                             25   11 00052            BRB     5$                                              ; 1937
       0000G   CF       80   8F   88 00054 3$:        BISB2   #128, COPY$SEM_STATUS+2                         ; 1944
       0000G   CF            00   FB 0005A            CALLS   #0, COPY$CALC_ALQ                               ; 1954
          10   A3            50   D0 0005F            MOVL    R0, 16(R3)                                      ;
```

```
                                   10  13 00063         BEQL    4$                                    ; 1956
                          0000G    CF  9F 00065         PUSHAB  COPY$OUTOPN_ERR                       ; 1962
                                   55  DD 00069         PUSHL   R5                                    ;
               00000000G  00       02  FB 0006B         CALLS   #2, SYS$EXTEND                        ;
                          04       50  E9 00072         BLBC    R0, 5$                                ;
                          50       01  D0 00075 4$:     MOVL    #1, R0                                ; 1966
                                       04 00078         RET                                           ;
                                   50  D4 00079 5$:     CLRL    R0                                    ; 1968
                                       04 0007B         RET                                           ;

; Routine Size:  124 bytes,    Routine Base:  $CODE$ + 0543
```

```
ROUTINE setup_outxab (output_fab, input_fab) : NOVALUE =
                                                   ! Setup output XAB fields from input XAB fields

!++
! Functional description:
!
!       This routine copies input XAB fields into corresponding output XAB fields.
!
! Calling sequence:
!
!       setup_outxab (output_fab.ra.v, input_fab.ra.v)
!
! Input parameters:
!
!       output_fab      - FAB block associated with the output file
!       input_fab       - FAB block associated with the input file
!
! Implicit inputs:
!
!       output_xaball   - XABALL block for output file
!       output_xabdat   - XABDAT block for output file
!       output_xabfhc   - XABFHC block for output file
!       output_xabpro   - XABPRO block for output file
!       output_xabrdt   - XABRDT block for output file
!
!       input_xaball    - XABALL block for input file
!       input_xabdat    - XABDAT block for input file
!       input_xabfhc    - XABFHC block for input file
!       input_xabpro    - XABPRO block for input file
!
! Output parameters
!
!       none
!
! Implicit outputs
!
!       The relevant fields in the output XABs are written.
!
! Routine value
!
!       none
!
! Side effects
!
!       none
!
!--

    BEGIN

    MAP
        output_fab      : REF BLOCK [, BYTE],           ! output file FAB block
        input_fab       : REF BLOCK [, BYTE];           ! input file FAB block

    BIND
        output_nam      =                               ! output NAM block address
                .output_fab [fab$l_nam]         : BLOCK [, BYTE],
        output_xabfhc    =                                  ! output XAB file header characteristics block
                .output_fab [fab$l_xab]         : BLOCK [, BYTE],
        output_xaball    =                                  ! output XAB date block
                .output_xabfhc [xab$l_nxt]      : BLOCK [, BYTE],
        output_xabdat    =                                  ! output XAB date block
                .output_xaball [xab$l_nxt]      : BLOCK [, BYTE],
        output_xabrdt    =                                  ! output XAB date block
                .output_xabdat [xab$l_nxt]      : BLOCK [, BYTE],
        output_xabpro    =                                  ! output XAB date block
                .output_xabrdt [xab$l_nxt]      : BLOCK [, BYTE],

        input_xaball     =                                  ! input file XABALL block
                .input_fab [fab$l_xab]          : BLOCK [, BYTE],
        input_xabdat     =                                  ! input file XABDAT block
                .input_xaball [xab$l_nxt]       : BLOCK [, BYTE],
        input_xabfhc     =                                  ! input file XABFHC block
                .input_xabdat [xab$l_nxt]       : BLOCK [, BYTE],
        input_xabpro     =                                  ! input file XABPRO block
                .input_xabfhc [xab$l_nxt]       : BLOCK [, BYTE];

!
! Write the output allocation XAB.
!

    output_xaball [xab$b_aop] =
                        .input_xaball [xab$b_aop];          ! Write the allocation options,
    output_xaball [xab$b_aln] =
                        .input_xaball [xab$b_aln];          !    and the alignment type.

    output_xaball [xab$l_alq] = copy$calc_alq ();           ! Calculate and write in the allocation quantity.

    output_xaball [xab$w_deq] =
                        .input_xabfhc [xab$w_dxq];          ! Write the default extension quantity.
    output_xaball [xab$b_bkz] =                             ! Write the default bucket size
                        .input_fab [fab$b_bks];             !    from the input FAB bucket size.
                                                            !    This insures the file is created with
                                                            !    correct bucksize.  Area 0 not may have
                                                            !    the largest bucket size.

    output_xaball [xab$w_vol] = 0;                          ! Zero the related volume number,
    output_xaball [xab$l_loc] = 0;                          !    the allocation location,
    output_xaball [xab$b_aid] = 0;                          !    the area id number,
    output_xaball [xab$w_rfi0] = 0;                         !    the related file number
    output_xaball [xab$w_rfi2] = 0;                         !    the related file sequence number
    output_xaball [xab$w_rfi4] = 0;                         !    and the related file revision number.

    IF .input_fab [$fab_dev(net)] AND                       ! If this is a network operation
       .output_xaball [xab$l_alq] EQL 0                     !    and the calculated ALQ = 0,
    THEN output_xaball [xab$l_alq] =                        !    then get ALQ from the FHC XAB
                        .input_xabfhc [xab$l_hbk];          !

!
! Write the output Date/Time XAB.
!

    output_xabdat [xab$w_rvn] =                             ! Increment the revision number
                        .input_xabdat [xab$w_rvn ] + 1;
    output_xabdat [xab$l_rdt0] = 0;                               ! Clear the revision date
    output_xabdat [xab$l_rdt4] = 0;
    output_xabdat [xab$l_cdt0] =                                  ! Copy the creation date
                        .input_xabdat [xab$l_cdt0];
    output_xabdat [xab$l_cdt4] =                                  !    and the creation time
                        .input_xabdat [xab$l_cdt4];
!
! These values are not copied from the input, but defaulted instead,
! so the user will get new backup and expiration dates.
!

    ! If the output device is tape, then propogate the expiration date.
    ! Otherwise, clear it.

    IF .output_fab[ $FAB_DEV(sqd) ]
    THEN
        BEGIN
        output_xabdat [xab$l_edt0] = .input_xabdat [xab$l_edt0];
        output_xabdat [xab$l_edt4] = .input_xabdat [xab$l_edt4];
        END
    ELSE
        BEGIN
        output_xabdat [xab$l_edt0] = 0;
        output_xabdat [xab$l_edt4] = 0;
        END;

    output_xabdat [xab$l_bdt0] = 0;                               !    the backup date
    output_xabdat [xab$l_bdt4] = 0;                               !    and the backup time

!
! Write the output File Header Characteristics XAB block.
!

    output_xabfhc [xab$b_rfo] =                                   ! The XABFHC includes the
                        .input_xabfhc [xab$b_rfo];                !    record format and file organization,
    output_xabfhc [xab$b_atr] =                                   !    the record attributes,
                        .input_xabfhc [xab$b_atr];
    output_xabfhc [xab$w_lrl] =                                   !    the length of the longest record,
                        .input_xabfhc [xab$w_lrl];
    output_xabfhc [xab$b_bkz] =                                   !    the bucket size,
                        .input_xabfhc [xab$b_bkz];
    output_xabfhc [xab$b_hsz] =                                   !    the VFC header size,
                        .input_xabfhc [xab$b_hsz];
    output_xabfhc [xab$w_mrz] =                                   !    the maximum record length,
                        .input_xabfhc [xab$w_mrz];
    output_xabfhc [xab$w_dxq] =                                   !    and the default extension quantity.
                        .input_xabfhc [xab$w_dxq];

    output_xabfhc [xab$l_sbn] = 0;                                ! Zero the starting virtual block number.

!
! Write the output Protection XAB block. Most of this XAB can only be setup
! after the output file has been opened or created. Therefore, it is not done here.
!

    output_xabpro [xab$l_uic] = 0;                                ! Clear the file owner field.

```

```
; 1613          2140  2 !
; 1614          2141  2 ! Write the output Revision Date/Time XAB block.
; 1615          2142  2 !
; 1616          2143  2
; 1617          2144  2      output_xabrdt [xab$w_rvn] =                            ! Increment revision number
; 1618          2145  2                 .input_xabdat [xab$w_rvn ] + 1;
; 1619          2146  2      output_xabrdt [xab$l_rdt0] = 0;                        ! Do not propogate the the input revision date,
; 1620          2147  2      output_xabrdt [xab$l_rdt4] = 0;
; 1621          2148  2
; 1622          2149  2 ! ******
; 1623          2150  2 !        Temporarily, I omit the special saving of XABDAT and XABFHC fields
; 1624          2151  2 !        of a file that may be overwritten. This must go back in.
; 1625          2152  2 ! ******
; 1626          2153  2
; 1627          2154  1      END;



                                            07FC 00000 SETUP_OUTXAB:
                                                               .WORD    Save R2,R3,R4,R5,R6,R7,R8,R9,R10              ; 1969
                        58        04   AC  D0 00002            MOVL     OUTPUT_FAB, R8                                ; 2025
                        56        24   A8  D0 00006            MOVL     36(R8), R6                                    ; 2027
                        52        04   A6  D0 0000A            MOVL     4(R6), R2                                     ; 2029
                        53        04   A2  D0 0000E            MOVL     4(R2), R3                                     ; 2031
                        59        04   A3  D0 00012            MOVL     4(R3), R9                                     ; 2033
                        5A        04   A9  D0 00016            MOVL     4(R9), R10                                    ; 2035
                        57        08   AC  D0 0001A            MOVL     INPUT_FAB, R7                                 ; 2038
                        50        24   A7  D0 0001E            MOVL     36(R7), R0                                    ;
                        54        04   A0  D0 00022            MOVL     4(R0), R4                                     ; 2040
                        55        04   A4  D0 00026            MOVL     4(R4), R5                                     ; 2042
                  08    A2        08   A0  B0 0002A            MOVW     8(R0), 8(R2)                                  ; 2051
               0000G    CF             00  FB 0002F            CALLS    #0, COPY$CALC_ALQ                             ; 2055
                  10    A2             50  D0 00034            MOVL     R0, 16(R2)                                    ;
                  14    A2        1A   A5  B0 00038            MOVW     26(R5), 20(R2)                                ; 2058
                                  0A   A2  B4 0003D            CLRW     10(R2)                                        ; 2065
                                  0C   A2  D4 00040            CLRL     12(R2)                                        ; 2066
                  16    A2        3E   A7  9B 00043            MOVZBW   62(R7), 22(R2)                                ; 2060
                                  18   A2  D4 00048            CLRL     24(R2)                                        ; 2068
                                  1C   A2  B4 0004B            CLRW     28(R2)                                        ; 2070
         0A       41    A7             05  E1 0004E            BBC      #5, 65(R7), 1$                                ; 2072
                                  10   A2  D5 00053            TSTL     16(R2)                                        ; 2073
                                       05  12 00056            BNEQ     1$                                            ;
                  10    A2        0C   A5  D0 00058            MOVL     12(R5), 16(R2)                                ; 2075
                        50        08   A4  3C 0005D 1$:        MOVZWL   8(R4), R0                                     ; 2082
                                       50  D6 00061            INCL     R0                                            ;
                  08    A3             50  B0 00063            MOVW     R0, 8(R3)                                     ;
                                  0C   A3  7C 00067            CLRQ     12(R3)                                        ; 2083
                  14    A3        14   A4  7D 0006A            MOVQ     20(R4), 20(R3)                                ; 2086
         07       40    A8             05  E1 0006F            BBC      #5, 64(R8), 2$                                ; 2097
                  1C    A3        1C   A4  7D 00074            MOVQ     28(R4), 28(R3)                                ; 2100
                                       03  11 00079            BRB      3$                                            ; 2097
                                  1C   A3  7C 0007B 2$:        CLRQ     28(R3)                                        ; 2105
                                  24   A3  7C 0007E 3$:        CLRQ     36(R3)                                        ; 2109
                  08    A6        08   A5  D0 00081            MOVL     8(R5), 8(R6)                                  ; 2117
                  16    A6        16   A5  D0 00086            MOVL     22(R5), 22(R6)                                ; 2123
```

```
                        1A    A6       1A    A5  B0 0008B          MOVW    26(R5), 26(R6)                     ; 2129
                                       28    A6  D4 00090          CLRL    40(R6)                             ; 2131
                                       0C    AA  D4 00093          CLRL    12(R10)                            ; 2138
                        08    A9             50  B0 00096          MOVW    R0, 8(R9)                          ; 2145
                                       0C    A9  7C 0009A          CLRQ    12(R9)                             ; 2146
                                                 04 0009D          RET                                        ; 2154

; Routine Size:  158 bytes,    Routine Base:  $CODE$ + 05BF
```

```
ROUTINE apply_out_qual (output_fab) : NOVALUE =          ! Applies output parameter qualifiers to FAB and XAB

!++
! Functional description
!
!       This routine looks for the presence of qualifiers on the output file specification,
!       and sets RMS fields according to the semantics of each qualifier.
!
! Calling sequence:
!
!       apply_out_qual (output_fab.ra.v)
!
! Input parameters:
!
!       output_fab      - the FAB block related to the output file specification
!
! Implicit inputs:
!
!       output_xaball   - The XABALL block associated with the output FAB
!
!       The following bits in COPY$CLI_STATUS:
!
!               alignment_bit
!               allocation_bit
!               contiguous_bit
!               extension_bit
!               file_max_bit
!               overlay_bit
!               oread_check_bit
!               replace_bit
!               truncate_bit
!               write_check_bit
!               volume_bit
!
!       Some values associated with qualifiers specified for the output file specification:
!
!               align_type
!               align_option
!               align_location
!               alloc_value
!               extension_value
!               file_max_value
!               volume_value
!
! Output parameters
!
!       none
!
! Implicit outputs
!
!       Some fields in the output XABALL block are written:
!
!               ALN     - alignment type
!               AOP     - alignment option
!               LOC     - alignment location
!               ALQ     - allocation quantity
!               CTG     - contiguous file
```

```
1 !                 CBT      - contiguous best try file
1 !                 DEQ      - file extension quantity
1 !                 VOL      - relative volume number
1 !
1 !         Some fields in the output FAB are written:
1 !
1 !                 MRN      - maximum record number
1 !                 CIF      - create if nonexistent file
1 !                 RCK      - read check
1 !                 TEF      - truncate files at EOF mark
1 !                 SUP      - supersede
1 !                 WCK      - write check
1 !
1 ! Routine value
1 !
1 !         novalue
1 !
1 ! Side effects
1 !
1 !         none
1 !
1 !--
1
2     BEGIN
2
2     MAP
2         output_fab       : REF BLOCK [, BYTE];              ! Output file FAB block
2
2     BIND
2         output_nam       =                                  ! output NAM block address
2                 .output_fab [fab$l_nam]          : BLOCK [, BYTE],
2         output_xabfhc    =                                  ! output XAB file header characteristics block
2                 .output_fab [fab$l_xab]          : BLOCK [, BYTE],
2         output_xaball    =                                  ! output XAB date block
2                 .output_xabfhc [xab$l_nxt]       : BLOCK [, BYTE],
2         output_xabdat    =                                  ! output XAB date block
2                 .output_xaball [xab$l_nxt]       : BLOCK [, BYTE],
2         output_xabrdt    =                                  ! output XAB date block
2                 .output_xabdat [xab$l_nxt]       : BLOCK [, BYTE],
2         output_xabpro    =                                  ! output XAB date block
2                 .output_xabrdt [xab$l_nxt]       : BLOCK [, BYTE];
2
2 !
2 ! Apply the effects of the output file qualifiers to the appropriate XAB blocks.
2 !
2
2      ! /ALLOCATION = n
2      !
3      IF qualifier_active( alloc_qual, loc_alloc_qual, neg_alloc_qual )
2      THEN
2          output_xaball [xab$l_alq] = .curr_allocation_value;
2
3      IF qualifier_active( contig_qual, loc_contig_qual, neg_contig_qual )
2      THEN
3          BEGIN
3          output_xaball [xab$v_ctg] = TRUE;
3          output_xaball [xab$v_cbt] = FALSE;
```

```
3               END
2           ELSE
3               BEGIN
3               IF .contig_negated OR .neg_contig_qual
3               THEN
4                   BEGIN
4                   output_xaball [xab$v_ctg] = FALSE;
4                   output_xaball [xab$v_cbt] = FALSE;
3                   END;
2               END;
2
3           IF qualifier_active( extend_qual, loc_extend_qual, neg_extend_qual )
2           THEN
2               output_xaball [xab$w_deq] = .curr_extension_value;
2
3           IF qualifier_active( file_max_qual, loc_file_max_qual, neg_file_max_qual )
2           THEN
2               output_fab [fab$l_mrn] = .curr_file_max_value;
2
2           IF qualifier_active( overlay_qual, loc_overlay_qual, neg_overlay_qual ) OR
2               .new_version_qual
2           THEN
2               output_fab [fab$v_cif] = TRUE;
2
3           IF qualifier_active( replace_qual, loc_replace_qual, neg_replace_qual )
2           THEN
2               output_fab [fab$v_sup] = TRUE;
2
3           IF qualifier_active( truncate_qual, loc_truncate_qual, neg_truncate_qual )
2           THEN
2               output_fab [fab$v_tef] = TRUE;
2
3           IF qualifier_active( volume_qual, loc_volume_qual, neg_volume_qual )
2           THEN
3               BEGIN
3               output_xaball [xab$w_vol] = .curr_volume_value;
3               output_xaball [xab$b_aln] = xab$c_lbn;
3               output_xaball [xab$v_hrd] = 1;
2               END;
2
3           IF qualifier_active( write_chk_qual, loc_write_chk_qual, neg_write_chk_qual )
2           THEN
2               output_fab [fab$v_wck] = TRUE
2           ELSE
3               BEGIN
3               IF .write_chk_negated
3               THEN
3                   output_fab [fab$v_wck] = FALSE;
2               END;
2
2       !
2       ! Return to caller.
2       !
2
1           END;                                                ! Return without a value.
```

```
                         000C 00000 APPLY_OUT_QUAL:
                                          .WORD   Save R2,R3                                    ; 2155
            53  0000G  CF  9E 00002       MOVAB   COPY$CLI_STATUS+4, R3                         ;
            51     04  AC  D0 00007       MOVL    OUTPUT_FAB, R1                                ; 2242
            50     24  A1  D0 0000B       MOVL    36(R1), R0                                    ; 2244
            50     04  A0  D0 0000F       MOVL    4(R0), R0                                     ; 2246
            52     04  A0  D0 00013       MOVL    4(R0), R2                                     ; 2248
            05     FE  A3  E9 00017       BLBC    COPY$CLI_STATUS+2, 1$                         ; 2260
05    FE    A3         02  E1 0001B       BBC     #2, COPY$CLI_STATUS+2, 2$                     ;
06    FE    A3         01  E1 00020 1$:   BBC     #1, COPY$CLI_STATUS+2, 3$                     ;
      10    A0  0000G  CF  D0 00025 2$:   MOVL    CURR_ALLOCATION_VALUE, 16(R0)                 ; 2262
05    FE    A3         03  E1 0002B 3$:   BBC     #3, COPY$CLI_STATUS+2, 4$                     ; 2264
05    FE    A3         06  E1 00030       BBC     #6, COPY$CLI_STATUS+2, 5$                     ;
07    FE    A3         05  E1 00035 4$:   BBC     #5, COPY$CLI_STATUS+2, 6$                     ;
      08    A0     80  8F  88 0003A 5$:   BISB2   #128, 8(R0)                                   ; 2267
                       0F  11 0003F       BRB     8$                                            ; 2268
05    FE    A3         04  E0 00041 6$:   BBS     #4, COPY$CLI_STATUS+2, 7$                     ; 2272
09    FE    A3         06  E1 00046       BBC     #6, COPY$CLI_STATUS+2, 9$                     ;
      08    A0     80  8F  8A 0004B 7$:   BICB2   #128, 8(R0)                                   ; 2275
      08    A0         20  8A 00050 8$:   BICB2   #32, 8(R0)                                    ; 2276
                   FE  A3  95 00054 9$:   TSTB    COPY$CLI_STATUS+2                             ; 2280
                       05  18 00057       BGEQ    10$                                           ;
04    FF    A3         01  E1 00059       BBC     #1, COPY$CLI_STATUS+3, 11$                    ;
            06     FF  A3  E9 0005E 10$:  BLBC    COPY$CLI_STATUS+3, 12$                        ;
      14    A0  0000G  CF  B0 00062 11$:  MOVW    CURR_EXTENSION_VALUE, 20(R0)                  ; 2282
05    FF    A3         02  E1 00068 12$:  BBC     #2, COPY$CLI_STATUS+3, 13$                    ; 2284
05    FF    A3         04  E1 0006D       BBC     #4, COPY$CLI_STATUS+3, 14$                    ;
06    FF    A3         03  E1 00072 13$:  BBC     #3, COPY$CLI_STATUS+3, 15$                    ;
      38    A1  0000G  CF  D0 00077 14$:  MOVL    CURR_FILE_MAX_VALUE, 56(R1)                   ; 2286
                       63  95 0007D 15$:  TSTB    COPY$CLI_STATUS+4                             ; 2288
                       05  18 0007F       BGEQ    16$                                           ;
09    01    A3         01  E1 00081       BBC     #1, COPY$CLI_STATUS+5, 17$                    ;
            05     01  A3  E8 00086 16$:  BLBS    COPY$CLI_STATUS+5, 17$                        ;
04    FC    A3         04  E1 0008A       BBC     #4, COPY$CLI_STATUS, 18$                      ; 2289
      07    A1         02  88 0008F 17$:  BISB2   #2, 7(R1)                                     ; 2291
05    02    A3         01  E1 00093 18$:  BBC     #1, COPY$CLI_STATUS+6, 19$                    ; 2293
05    02    A3         03  E1 00098       BBC     #3, COPY$CLI_STATUS+6, 20$                    ;
04    02    A3         02  E1 0009D 19$:  BBC     #2, COPY$CLI_STATUS+6, 21$                    ;
      04    A1         04  88 000A2 20$:  BISB2   #4, 4(R1)                                     ; 2295
04    01    A3         05  E1 000A6 21$:  BBC     #5, COPY$CLI_STATUS+5, 22$                    ; 2297
            05     02  A3  E9 000AB       BLBC    COPY$CLI_STATUS+6, 23$                        ;
                   01  A3  95 000AF 22$:  TSTB    COPY$CLI_STATUS+5                             ;
                       04  18 000B2       BGEQ    24$                                           ;
      07    A1         10  88 000B4 23$:  BISB2   #16, 7(R1)                                    ; 2299
05    01    A3         02  E1 000B8 24$:  BBC     #2, COPY$CLI_STATUS+5, 25$                    ; 2301
05    01    A3         04  E1 000BD       BBC     #4, COPY$CLI_STATUS+5, 26$                    ;
0E    01    A3         03  E1 000C2 25$:  BBC     #3, COPY$CLI_STATUS+5, 27$                    ;
      0A    A0  0000G  CF  B0 000C7 26$:  MOVW    CURR_VOLUME_VALUE, 10(R0)                     ; 2304
      09    A0         02  90 000CD       MOVB    #2, 9(R0)                                     ; 2305
      08    A0         01  88 000D1       BISB2   #1, 8(R0)                                     ; 2306
04          63         03  E1 000D5 27$:  BBC     #3, COPY$CLI_STATUS+4, 28$                    ; 2309
04          63         06  E1 000D9       BBC     #6, COPY$CLI_STATUS+4, 29$                    ;
05          63         05  E1 000DD 28$:  BBC     #5, COPY$CLI_STATUS+4, 30$                    ;
      05    A1         02  88 000E1 29$:  BISB2   #2, 5(R1)                                     ; 2311
```

```
                                                     04 000E5           RET                                                  ;
                         04          63          04  E1 000E6 30$:      BBC     #4, COPY$CLI_STATUS+4, 31$                   ; 2314
                                05   A1          02  8A 000EA           BICB2   #2, 5(R1)                                    ; 2316
                                                     04 000EE 31$:      RET                                                  ; 2323

; Routine Size:  239 bytes,    Routine Base:  $CODE$ + 065D
; 1799          2324   1 END
; 1800          2325   0 ELUDOM



                                                                      .EXTRN  LIB$SIGNAL, LIB$STOP

;                                  PSECT SUMMARY
;
;       Name                     Bytes                        Attributes
;
;  $PLIT$                             36  NOVEC,NOWRT,  RD ,NOEXE,NOSHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)
;  $CODE$                           1868  NOVEC,NOWRT,  RD ,  EXE,NOSHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)



;                            Library Statistics
;
;                                           -------- Symbols --------       Pages      Processing
;       File                                Total     Loaded   Percent     Mapped     Time
;
;  _$255$DUA28:[SYSLIB]STARLET.L32;1         9776        179         1        581      00:01.0




;                                  COMMAND QUALIFIERS

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

; Size:          1868 code + 36 data bytes
; Run Time:          00:51.0
; Elapsed Time:      02:08.1
; Lines/CPU Min:      2735
; Lexemes/CPU-Min: 28632
; Memory Used:  286 pages
; Compilation Complete
```

0067 AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY

COPYMSG
REQ

COPYSPECS
LIS

COPY

COPY
MAP

VMSMAC
REQ

COPY
REQ

COPYCLI
LIS

COPYMAIN
LIS

COPYSEMAN
LIS

0068 AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY

CRFOR
LIS

CRFMEM
LIS

CRFMACROS
MAR

CLITABDEF
SDL

FILINPUT
LIS

KEYS
LIS

CRFTFRVEC
LIS

DCLDEF
MDL

CRFMDL
MDL

CREF
LIS

CRFERRMSG
LIS

CRF
SDL

CRF

CRFSUB
LIS

FILOUTPUT
LIS

CRFSHR
MAP

CRFGLOBAL
LIS

CRFMAC
REQ

DCL

DCL
MAP