```
LLL                 000000000          GGGGGGGGGGGG     IIIIIIIII     NNN           NNN
LLL                 000000000          GGGGGGGGGGGG     IIIIIIIII     NNN           NNN
LLL                 000000000          GGGGGGGGGGGG     IIIIIIIII     NNN           NNN
LLL             000           000   GGG                    III      NNN           NNN
LLL             000           000   GGG                    III      NNN           NNN
LLL             000           000   GGG                    III      NNN           NNN
LLL             000           000   GGG                    III      NNNNNN        NNN
LLL             000           000   GGG                    III      NNNNNN        NNN
LLL             000           000   GGG                    III      NNNNNN        NNN
LLL             000           000   GGG                    III      NNN    NNN    NNN
LLL             000           000   GGG                    III      NNN    NNN    NNN
LLL             000           000   GGG                    III      NNN    NNN    NNN
LLL             000           000   GGG    GGGGGGGGG       III      NNN       NNNNNN
LLL             000           000   GGG    GGGGGGGGG       III      NNN       NNNNNN
LLL             000           000   GGG    GGGGGGGGG       III      NNN       NNNNNN
LLL             000           000   GGG          GGG       III      NNN           NNN
LLL             000           000   GGG          GGG       III      NNN           NNN
LLL             000           000   GGG          GGG       III      NNN           NNN
LLLLLLLLLLLLLLL     000000000          GGGGGGGGG        IIIIIIIII     NNN           NNN
LLLLLLLLLLLLLLL     000000000          GGGGGGGGG        IIIIIIIII     NNN           NNN
LLLLLLLLLLLLLLL     000000000          GGGGGGGGG        IIIIIIIII     NNN           NNN
```

**FILE**ID**FILEIO

```
FFFFFFFFFF    IIIIII    LL          EEEEEEEEEE    IIIIII      000000
FFFFFFFFFF    IIIIII    LL          EEEEEEEEEE    IIIIII      000000
FF              II      LL          EE              II      00      00
FF              II      LL          EE              II      00      00
FF              II      LL          EE              II      00      00
FF              II      LL          EE              II      00      00
FFFFFFFF        II      LL          EEEEEEEE        II      00      00
FFFFFFFF        II      LL          EEEEEEEE        II      00      00
FF              II      LL          EE              II      00      00
FF              II      LL          EE              II      00      00
FF              II      LL          EE              II      00      00      ....
FF              II      LL          EE              II      00      00      ....
FF            IIIIII    LLLLLLLLLL  EEEEEEEEEE    IIIIII      000000      ....
FF            IIIIII    LLLLLLLLLL  EEEEEEEEEE    IIIIII      000000      ....

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
```

```
;     1          0001  0 MODULE fileio (IDENT = 'V04-000',
;     2          0002  0        ADDRESSING_MODE(EXTERNAL = GENERAL)) =
;     3          0003  1 BEGIN
;     4          0004  1
;     5          0005  1 !
;     6          0006  1 !**************************************************************************
;     7          0007  1 !*                                                                        *
;     8          0008  1 !*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                               *
;     9          0009  1 !*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.                 *
;    10          0010  1 !*  ALL RIGHTS RESERVED.                                                   *
;    11          0011  1 !*                                                                        *
;    12          0012  1 !*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED   *
;    13          0013  1 !*  ONLY IN  ACCORDANCE WITH  THE  TERMS  OF  SUCH  LICENSE  AND WITH THE  *
;    14          0014  1 !*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR  ANY  OTHER   *
;    15          0015  1 !*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY   *
;    16          0016  1 !*  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE IS  HEREBY   *
;    17          0017  1 !*  TRANSFERRED.                                                           *
;    18          0018  1 !*                                                                        *
;    19          0019  1 !*  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE WITHOUT NOTICE   *
;    20          0020  1 !*  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL EQUIPMENT   *
;    21          0021  1 !*  CORPORATION.                                                           *
;    22          0022  1 !*                                                                        *
;    23          0023  1 !*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR  RELIABILITY OF ITS   *
;    24          0024  1 !*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.                 *
;    25          0025  1 !*                                                                        *
;    26          0026  1 !*                                                                        *
;    27          0027  1 !**************************************************************************
;    28          0028  1
;    29          0029  1 !++
;    30          0030  1 ! FACILITY:  Login
;    31          0031  1 !
;    32          0032  1 ! ABSTRACT:
;    33          0033  1 !
;    34          0034  1 !       This module contains I/O routines.
;    35          0035  1 !
;    36          0036  1 ! ENVIRONMENT:
;    37          0037  1 !
;    38          0038  1 !       VAX/VMS operating system.
;    39          0039  1 !
;    40          0040  1 ! AUTHOR:  Tim Halvorsen, March 1981
;    41          0041  1 !
;    42          0042  1 ! Modified by:
;    43          0043  1 !
;    44          0044  1 !       V03-012 JRL0033         John R. Lawson, Jr.     06-Aug-1984 13:46
;    45          0045  1 !               Open SYS$INPUT: with RAT = PRN and RFM = VFC.
;    46          0046  1 !
;    47          0047  1 !       V03-011 ACG0434         Andrew C. Goldstein,    9-Jul-1984  19:39
;    48          0048  1 !               Use SYSGEN parameter to time out all LOGIN reads
;    49          0049  1 !
;    50          0050  1 !       V03-010 MHB0145         Mark Bramhall           27-Apr-1984
;    51          0051  1 !               Add flag for WRITE_OUTPUT's success/fail.
;    52          0052  1 !
;    53          0053  1 !       V03-009 MHB0129         Mark Bramhall            5-Apr-1984
;    54          0054  1 !               Add mode to GET_INPUT to optionally return on timeouts.
;    55          0055  1 !               Hang up terminal on GET_INPUT silent exits.
;    56          0056  1 !
;    57          0057  1 !       V03-008 MHB0108         Mark Bramhall           21-Mar-1984
```

```
;    58          0058  1 !                Use LNM services for logical names.
;    59          0059  1 !                Reference PCB_STS as a BITVECTOR.
;    60          0060  1 !
;    61          0061  1 !        V03-007 PCG0001         Peter George            31-Jan-1984  12:57
;    62          0062  1 !                Allow read and write access to the input stream if it is
;    63          0063  1 !                a terminal.
;    64          0064  1 !
;    65          0065  1 !        V03-006 ACG0376         Andrew C. Goldstein,    18-Nov-1983  18:17
;    66          0066  1 !                Cancel I/O on both terminal channels when running down.
;    67          0067  1 !                Add a timer to GET_INPUT to handle obscure failures.
;    68          0068  1 !
;    69          0069  1 !        V03-005 PAS0173         Ron Schaefer            05-Sep-1983
;    70          0070  1 !                Change the creation of SYS$INPUT, SYS$COMMAND. SYS$OUTPUT
;    71          0071  1 !                and SYS$ERROR to use $CRELNM rather than $CRELOG.
;    72          0072  1 !
;    73          0073  1 !        V03-004 GAS0169         Gerry Smith             23-Aug-1983
;    74          0074  1 !                For OPEN_INPUT, unconditionally open SYS$INPUT, rather
;    75          0075  1 !                than trying to figure out if only one stream should be
;    76          0076  1 !                used for both input and output.
;    77          0077  1 !
;    78          0078  1 !        V03-003 GAS0162         Gerry Smith             30-Jul-1983
;    79          0079  1 !                Make WRITE_TIMEOUT global, so that the system password
;    80          0080  1 !                routine can use it.
;    81          0081  1 !
;    82          0082  1 !        V03-002 MLJ0115         Martin L. Jack, 29-Jul-1983  10:30
;    83          0083  1 !                Update for new log file error handling.
;    84          0084  1 !
;    85          0085  1 !        V03-001 TMH0001         Tim Halvorsen           07-Feb-1983
;    86          0086  1 !                If there is any problem creating the .LOG file for a
;    87          0087  1 !                network job, then connect the output stream to NL: and
;    88          0088  1 !                allow the job to continue, rather than aborting with an
;    89          0089  1 !                error.  This is to make network jobs more robust, in the
;    90          0090  1 !                event that the account has run out of quota, the disk out
;    91          0091  1 !                of space, the directory improperly protected, etc.  This
;    92          0092  1 !                enhancement can also be used to selectively prevent some
;    93          0093  1 !                network objects from producing a .LOG file, if disk space
;    94          0094  1 !                is precious.
;    95          0095  1 !
;    96          0096  1 !        V010    RAS0077         Ron Schaefer            25-Feb-1982
;    97          0097  1 !                Correct RAS0075/76 to save the concealed device state
;    98          0098  1 !                for the CLI as well.
;    99          0099  1 !
;   100          0100  1 !        V009    RAS0076         Ron Schaefer            24-Feb-1982
;   101          0101  1 !                Complete RAS0075 to copy NAM$L_FNB from output to input.
;   102          0102  1 !
;   103          0103  1 !        V008    RAS0075         Ron Schaefer            24-Feb-1982
;   104          0104  1 !                Change CRELOG logic to properly create concealed device
;   105          0105  1 !                names for SYS$INPUT,SYS$OUTPUT,SYS$ERROR and SYS$COMMAND.
;   106          0106  1 !
;   107          0107  1 !        V007    TMH0007         Tim Halvorsen           26-Jan-1982
;   108          0108  1 !                Remove code to put full filespec into PPF logical name.
;   109          0109  1 !                When I did it in the first place, I didn't think about
;   110          0110  1 !                the 59 character restriction on the size of an equivalence
;   111          0111  1 !                name.
;   112          0112  1 !
;   113          0113  1 !        V006    GAS0035         Gerry Smith             25-Jan-1982
;   114          0114  1 !                Remove the MRS = 512 from SYS$OUTPUT.  Specifying the
```

```
;  115     0115  1 !                     maximum record size caused some programs, which output
;  116     0116  1 !                     very large records, to fail if writing to SYS$OUTPUT.
;  117     0117  1 !
;  118     0118  1 !               V005    TMH0005         Tim Halvorsen           26-Oct-1981
;  119     0119  1 !                       Store OUTFNM in LGI area rather than PPD.
;  120     0120  1 !                       Change terminal character timeout to 15 seconds.
;  121     0121  1 !
;  122     0122  1 !               V03-004 RAS0035         Ron Schaefer            11-Sep-1981
;  123     0123  1 !                       Complete RAS0033 by always taking the device name from
;  124     0124  1 !                       the NAM$T_DVI field and not the RSA field if not a
;  125     0125  1 !                       disk file.
;  126     0126  1 !
;  127     0127  1 !               V03-003 RAS0033         Ron Schaefer            4-Sep-1981
;  128     0128  1 !                       Translate the device name from the RSA before creating
;  129     0129  1 !                       the logical names SYS$xxx.  This change required by the
;  130     0130  1 !                       new _device parse logic.
;  131     0131  1 !
;  132     0132  1 !               V002    TMH0002         Tim Halvorsen           16-Jul-1981
;  133     0133  1 !                       Reference SHRLIB$ for shared require files.
;  134     0134  1 !
;  135     0135  1 !               V03-001 GWF0052         Gary W. Fowler          29-May-1981
;  136     0136  1 !                       Add XABFHC and calls to flush and display log file
;  137     0137  1 !--
;  138     0138  1
;  139     0139  1 !
;  140     0140  1 ! Include files
;  141     0141  1 !
;  142     0142  1
;  143     0143  1 LIBRARY 'SYS$LIBRARY:LIB';                  ! VAX/VMS system definitions
;  144     0144  1 REQUIRE 'SHRLIB$:UTILDEF';                  ! Common BLISS definitions
;  145     0329  1 REQUIRE 'LIB$:PPDDEF';                      ! Process permanent data region
;  146     0476  1 REQUIRE 'LIB$:LGIDEF';                      ! LOGINOUT private permanent storage
```

```
;  148          0547  1 !
;  149          0548  1 ! Table of contents
;  150          0549  1 !
;  151          0550  1
;  152          0551  1 FORWARD ROUTINE
;  153          0552  1     open_input:          NOVALUE,       ! Open primary input file
;  154          0553  1     close_input:         NOVALUE,       ! Close primary input file
;  155          0554  1     open_output:         NOVALUE,       ! Open primary output file
;  156          0555  1     close_output:        NOVALUE,       ! Close primary output file
;  157          0556  1     write_file:          NOVALUE,       ! Write file to primary output
;  158          0557  1     write_fao:           NOVALUE,       ! Write formatted message to output
;  159          0558  1     write_output,                       ! Write to primary output stream
;  160          0559  1     write_timeout:       NOVALUE,       ! Write timeout AST
;  161          0560  1     get_input:           NOVALUE;       ! Get record from primary input stream
;  162          0561  1
;  163          0562  1 !
;  164          0563  1 ! External routines
;  165          0564  1 !
;  166          0565  1
;  167          0566  1 EXTERNAL ROUTINE
;  168          0567  1     create_logical,                     ! Create logical name with LNM services
;  169          0568  1     set_terminal_hangup: NOVALUE,       ! Set/clear terminal's hangup state
;  170          0569  1     exit_process:        NOVALUE,       ! Terminate process
;  171          0570  1     handler:             NOVALUE;       ! Condition handler
;  172          0571  1
;  173          0572  1 !
;  174          0573  1 ! Define literals
;  175          0574  1 !
;  176          0575  1
;  177          0576  1 LITERAL
;  178          0577  1     cr = 13;                            ! Carriage return character
;  179          0578  1
;  180          0579  1 !
;  181          0580  1 ! Define message codes
;  182          0581  1 !
;  183          0582  1
;  184          0583  1 EXTERNAL LITERAL
;  185          0584  1     lgi$_openin,
;  186          0585  1     lgi$_inputerr,
;  187          0586  1     lgi$_outputerr,
;  188          0587  1     lgi$_cmdinput;
;  189          0588  1
;  190          0589  1 !
;  191          0590  1 ! OWN storage
;  192          0591  1 !
;  193          0592  1
;  194          0593  1 OWN
;  195          0594  1     rsbuf:      VECTOR [nam$c_maxrss,BYTE]; ! Buffer for resultant filespec
;  196          0595  1
;  197          0596  1 GLOBAL
;  198          0597  1     write_output_status,                ! WRITE_OUTPUT's last status
;  199          0598  1
;  200        P 0599  1     output_nam: $NAM(                   ! NAM for SYS$OUTPUT
;  201        P 0600  1                 ESA = rsbuf,          ! Expanded filespec buffer
;  202        P 0601  1                 ESS = %ALLOCATION(rsbuf),
;  203        P 0602  1                 RSA = rsbuf,          ! Resultant filespec buffer
;  204          0603  1                 RSS = %ALLOCATION(rsbuf)),
```

```
;  205          0604  1
;  206        P 0605  1      output_fab: $FAB(                       ! FAB for SYS$OUTPUT
;  207        P 0606  1                  FNM = 'SYS$OUTPUT',         ! Primary filespec
;  208        P 0607  1                  DNM = '.LOG',               ! Default filespec
;  209        P 0608  1                  FAC = PUT,                  ! Output only
;  210        P 0609  1                  RAT = PRN,                  ! Print file format
;  211        P 0610  1                  RFM = VFC,                  ! Variable length with fixed control
;  212        P 0611  1                  FOP = (PPF,SQO,SUP),        ! Process permanent file
;  213        P 0612  1                                              ! Sequential only, network optimization
;  214        P 0613  1                                              ! Supersede existing file if explicit
;  215        P 0614  1                                              ! version specified
;  216        P 0615  1                  SHR = (GET,UPI),            ! Allow others to read file
;  217          0616  1                  NAM = output_nam),          ! Address of NAM block
;  218          0617  1
;  219        P 0618  1      output_rab: $RAB(                       ! RAB for SYS$OUTPUT
;  220        P 0619  1                  MBC = 1,                    ! 1 block/buffer (limit initial alloc.)
;  221        P 0620  1                  MBF = -1,                   ! 1 buffer/stream (ditto)
;  222          0621  1                  FAB = output_fab);          ! Address of FAB block
;  223          0622  1
;  224          0623  1 GLOBAL
;  225          0624  1      input_chan,                               ! Channel number for SYS$INPUT
;  226          0625  1      output_chan,                              ! Channel number for SYS$OUTPUT
;  227          0626  1
;  228        P 0627  1      input_nam:  $NAM(                       ! NAM for SYS$INPUT
;  229        P 0628  1                  RSA = rsbuf,                ! Resultant filespec buffer
;  230          0629  1                  RSS = %ALLOCATION(rsbuf)),
;  231          0630  1
;  232        P 0631  1      input_fab:  $FAB(                       ! FAB for SYS$INPUT
;  233        P 0632  1                  FNM = 'SYS$INPUT',          ! Primary filespec
;  234        P 0633  1                  DNM = '.COM',               ! Default filespec
;  235        P 0634  1                  FAC = GET,                  ! Read only
;  236        P 0635  1                  RAT = PRN,                  ! Print format
;  237        P 0636  1                  RFM = VFC,                  ! Variable length
;  238        P 0637  1                  FOP = (PPF,SQO,INP),        ! Process permanent file
;  239        P 0638  1                                              ! Sequential only, network optimization
;  240        P 0639  1                                              ! input stream
;  241          0640  1                  NAM = input_nam),           ! Address of NAM block
;  242          0641  1
;  243        P 0642  1      input_rab:  $RAB(                       ! RAB for SYS$INPUT
;  244        P 0643  1                  FAB = input_fab,            ! Address of FAB block
;  245        P 0644  1                  ROP = (CVT,TMO,PTA)         ! Convert to uppercase on entry
;  246        P 0645  1                                              ! Read with timeout
;  247        P 0646  1                                              ! Purge typeahead so that unsolicited
;  248        P 0647  1                                              ! character that started job is ignored
;  249          0648  1                  );
;  250          0649  1
;  251          0650  1 !
;  252          0651  1 ! External storage
;  253          0652  1 !
;  254          0653  1
;  255          0654  1 EXTERNAL
;  256          0655  1      pcb_sts: BITVECTOR,                       ! Our process status flags
;  257          0656  1      sys$gb_retry_tmo: BYTE,                   ! Terminal read timeout value
;  258          0657  1      ctl$ag_clidata;                           ! Process permanent data storage
;  259          0658  1
;  260          0659  1 BIND
;  261          0660  1      ppd = ctl$ag_clidata: BBLOCK;             ! Address the structure
```

```
;  263     0661  1 GLOBAL ROUTINE open_input: NOVALUE =
;  264     0662  1
;  265     0663  1 !---
;  266     0664  1 !
;  267     0665  1 !       This routine opens the primary input file.
;  268     0666  1 !
;  269     0667  1 ! Inputs:
;  270     0668  1 !
;  271     0669  1 !       Access mode is executive.
;  272     0670  1 !
;  273     0671  1 !       sys$input = Descriptor of SYS$INPUT equivalence string
;  274     0672  1 !
;  275     0673  1 ! Outputs:
;  276     0674  1 !
;  277     0675  1 !       input_fab/rab = FAB/RAB of SYS$INPUT stream
;  278     0676  1 !
;  279     0677  1 !       The PPF logical names SYS$INPUT and SYS$COMMAND are created.
;  280     0678  1 !---
;  281     0679  1
;  282     0680  2 BEGIN
;  283     0681  2
;  284     0682  2 BUILTIN FP;
;  285     0683  2
;  286     0684  2 BIND
;  287     0685  2     devchar = input_fab [fab$l_dev] : $BBLOCK;  ! Device characteristics
;  288     0686  2
;  289     0687  2 LOCAL
;  290     0688  2     ptr,
;  291     0689  2     buffer:     BBLOCK [4+nam$c_maxrss],! Buffer for equivalence string
;  292     0690  2     bufdesc:    VECTOR [2],             ! Descriptor of above buffer
;  293     0691  2     status;
;  294     0692  2
;  295     0693  2 .fp = handler;                          ! Enable condition handler
;  296     0694  2 !
;  297     0695  2 ! Set up the read timeout from the SYSGEN parameter cell.
;  298     0696  2 !
;  299     0697  2 input_rab[rab$b_tmo] = .sys$gb_retry_tmo;
;  300     0698  2
;  301     0699  2 !
;  302     0700  2 ! Open the input file and signal and quit with any errors.
;  303     0701  2 !
;  304     0702  3 IF NOT (status = $OPEN(FAB = input_fab))! Open input file
;  305     0703  2 THEN                                    ! and signal fatal errors
;  306     0704  2     SIGNAL_STOP(lgi$_inputerr,0,.status, .input_fab [fab$l_stv]);
;  307     0705  2
;  308     0706  2 !
;  309     0707  2 ! If input device is a terminal, then close the file and reopen it with
;  310     0708  2 ! both read and write access.
;  311     0709  2 !
;  312     0710  2 IF .devchar [dev$v_trm]                 ! If input is from a terminal
;  313     0711  2 THEN                                    ! Then reopen with read and write access
;  314     0712  3     BEGIN
;  315     0713  3     $CLOSE(FAB = input_fab);
;  316     0714  3     input_fab [fab$b_fac] = fab$m_get OR fab$m_put;
;  317     0715  4     IF NOT (status = $OPEN(FAB = input_fab))
;  318     0716  3     THEN
;  319     0717  3         SIGNAL_STOP(lgi$_inputerr,0,.status, .input_fab [fab$l_stv]);
```

```
;  320          0718  2     END;
;  321          0719  2
;  322          0720  2 !
;  323          0721  2 ! Connect to the input file and signal and quit with any errors.
;  324          0722  2 !
;  325          0723  3 IF NOT (status = $CONNECT(RAB = input_rab))      ! Connect to input file
;  326          0724  2 THEN                                            ! And signal any errors
;  327          0725  2     SIGNAL_STOP(lgi$_inputerr,0,.status, .input_rab [rab$l_stv]);
;  328          0726  2
;  329          0727  2 input_rab [rab$v_ppf_ind] = true;        ! Mark ok to use this RAB in user mode
;  330          0728  2 input_rab [rab$v_ppf_rat] = fab$m_cr;     ! Set default record format
;  331          0729  2 input_chan = .input_fab [fab$l_stv];     ! Save exec channel if terminal
;  332          0730  2 ppd [ppd$w_inpchan] = .input_fab [fab$l_stv];     ! Save exec channel if terminal
;  333          0731  2 ppd [ppd$l_inpdev] = .input_fab [fab$l_dev];     ! Save device characteristics
;  334          0732  2 ppd [ppd$w_inpifi] = .input_fab [fab$w_ifi];     ! and IFI
;  335          0733  2 ppd [ppd$w_inpisi] = .input_rab [rab$w_isi];     ! and ISI
;  336          0734  2 ppd [ppd$v_inpccl] = .input_nam [nam$v_cncl_dev];        ! and concealed attr
;  337          0735  2
;  338          0736  2 CH$MOVE(ppd$c_dvifid, input_nam [nam$t_dvi], ppd [ppd$t_inpdvi]);
;  339          0737  2
;  340          0738  2 buffer [0,0,16,0] = 27;                  ! Escape character
;  341          0739  2 buffer [2,0,16,0] = .input_fab [fab$w_ifi];
;  342          0740  2 ptr = CH$MOVE(CH$RCHAR(input_nam [nam$t_dvi]),
;  343          0741  2           input_nam [$BYTEOFFSET(nam$t_dvi)+1,0,0,0], buffer[4,0,0,0]);
;  344          0742  2 CH$WCHAR_A(':', ptr);                     ! Append a colon to device name
;  345          0743  2 bufdesc[0] = CH$DIFF(.ptr, buffer);
;  346          0744  2 bufdesc[1] = buffer;
;  347          0745  2
;  348          0746  2 create_logical(%ASCID 'SYS$INPUT',       ! Re-define SYS$INPUT
;  349          0747  2                 bufdesc,
;  350          0748  2                 psl$c_exec,
;  351          0749  3                 (IF .input_nam [nam$v_cncl_dev]
;  352          0750  3                  THEN UPLIT(lnm$m_terminal OR lnm$m_concealed)
;  353          0751  2                  ELSE UPLIT(lnm$m_terminal)));
;  354          0752  2
;  355          0753  2 create_logical(%ASCID 'SYS$COMMAND',     ! Define SYS$COMMAND
;  356          0754  2                 bufdesc,
;  357          0755  2                 psl$c_exec,
;  358          0756  3                 (IF .input_nam [nam$v_cncl_dev]
;  359          0757  3                  THEN UPLIT(lnm$m_terminal OR lnm$m_concealed)
;  360          0758  2                  ELSE UPLIT(lnm$m_terminal)));
;  361          0759  2
;  362          0760  1 END;


                                                                .TITLE  FILEIO
                                                                .IDENT  \V04-000\

                                                                .PSECT  $PLIT$,NOWRT,NOEXE,2

                     54  55  50  54  55  4F  24  53  59  53  00000 P.AAA:  .ASCII  \SYS$OUTPUT\                ;
                                         47  4F  4C  2E  0000A P.AAB:  .ASCII  \.LOG\                      ;
                         54  55  50  4E  49  24  53  59  53  0000E P.AAC:  .ASCII  \SYS$INPUT\               ;
                                         4D  4F  43  2E  00017 P.AAD:  .ASCII  \.COM\                       ;
                                                         0001B        .BLKB   1
                 00  00  00  54  55  50  4E  49  24  53  59  53  0001C P.AAF:  .ASCII  \SYS$INPUT\<0><0><0>   ;
                                                 010E0009  00028 P.AAE:  .LONG   17694729                   ;
```

```
                                00000000'  0002C                .ADDRESS P.AAF
                                00000300   00030 P.AAG:         .LONG   768
                                00000200   00034 P.AAH:         .LONG   512
                00  44  4E  41  4D  4D  4F  43  24  53  59  53   00038 P.AAJ:   .ASCII  \SYS$COMMAND\<0>
                                010E000B   00044 P.AAI:         .LONG   17694731
                                00000000'  00048               .ADDRESS P.AAJ
                                00000300   0004C P.AAK:         .LONG   768
                                00000200   00050 P.AAL:         .LONG   512

                                                               .PSECT  $OWN$,NOEXE,2

                                           00000 RSBUF:        .BLKB   255

                                                               .PSECT  $GLOBAL$,NOEXE,2

                                           00000 WRITE_OUTPUT_STATUS::
                                                               .BLKB   4
                                      02   00004 OUTPUT_NAM::
                                                               .BYTE   2
                                      60   00005               .BYTE   96
                                      FF   00006               .BYTE   -1
                                      00   00007               .BYTE   0
                                00000000'  00008               .ADDRESS RSBUF
                                      00   0000C               .BYTE   0
                                      00   0000D               .BYTE   0
                                      FF   0000E               .BYTE   -1
                                      00   0000F               .BYTE   0
                                00000000'  00010               .ADDRESS RSBUF
                                00000000   00014               .LONG   0
                                    0000#  00018               .WORD   0[8]
                                    0000#  00028               .WORD   0[3]
                                    0000#  0002E               .WORD   0[3]
                                00000000   00034               .LONG   0
                                00000000   00038               .LONG   0
                                      00   0003C               .BYTE   0
                                      00   0003D               .BYTE   0
                                      00   0003E               .BYTE   0
                                      00   0003F               .BYTE   0
                                      00   00040               .BYTE   0
                                      00   00041               .BYTE   0
                                      00#  00042               .BYTE   0[2]
                                00000000   00044               .LONG   0
                                00000000   00048               .LONG   0
                                00000000   0004C               .LONG   0
                                00000000   00050               .LONG   0
                                00000000   00054               .LONG   0
                                00000000   00058               .LONG   0
                                00000000#  0005C               .LONG   0[2]
                                      03   00064 OUTPUT_FAB::
                                                               .BYTE   3
                                      50   00065               .BYTE   80
                                    0000   00066               .WORD   0
                                00040044   00068               .LONG   262212
                                00000000   0006C               .LONG   0
                                00000000   00070               .LONG   0
                                00000000   00074               .LONG   0
                                    0000   00078               .WORD   0
```

```
      01  0007A          .BYTE   1                                                ;
      42  0007B          .BYTE   66                                               ;
00000000  0007C          .LONG   0                                                ;
      00  00080          .BYTE   0                                                ;
      00  00081          .BYTE   0                                                ;
      04  00082          .BYTE   4                                                ;
      03  00083          .BYTE   3                                                ;
00000000  00084          .LONG   0                                                ;
00000000  00088          .LONG   0                                                ;
00000000' 0008C          .ADDRESS OUTPUT_NAM                                      ;
00000000' 00090          .ADDRESS P.AAA                                           ;
00000000' 00094          .ADDRESS P.AAB                                           ;
      0A  00098          .BYTE   10                                               ;
      04  00099          .BYTE   4                                                ;
    0000  0009A          .WORD   0                                                ;
00000000  0009C          .LONG   0                                                ;
    0000  000A0          .WORD   0                                                ;
      00  000A2          .BYTE   0                                                ;
      00  000A3          .BYTE   0                                                ;
00000000  000A4          .LONG   0                                                ;
00000000  000A8          .LONG   0                                                ;
    0000  000AC          .WORD   0                                                ;
      00  000AE          .BYTE   0                                                ;
      00  000AF          .BYTE   0                                                ;
00000000  000B0          .LONG   0                                                ;
      01  000B4 OUTPUT_RAB::                                                      ;
                         .BYTE   1                                                ;
      44  000B5          .BYTE   68                                               ;
    0000  000B6          .WORD   0                                                ;
00000000  000B8          .LONG   0                                                ;
00000000  000BC          .LONG   0                                                ;
00000000  000C0          .LONG   0                                                ;
    0000# 000C4          .WORD   0[3]                                             ;
    0000  000CA          .WORD   0                                                ;
00000000  000CC          .LONG   0                                                ;
    0000  000D0          .WORD   0                                                ;
      00  000D2          .BYTE   0                                                ;
      00  000D3          .BYTE   0                                                ;
    0000  000D4          .WORD   0                                                ;
    0000  000D6          .WORD   0                                                ;
00000000  000D8          .LONG   0                                                ;
00000000  000DC          .LONG   0                                                ;
00000000  000E0          .LONG   0                                                ;
00000000  000E4          .LONG   0                                                ;
      00  000E8          .BYTE   0                                                ;
      00  000E9          .BYTE   0                                                ;
      FF  000EA          .BYTE   -1                                               ;
      01  000EB          .BYTE   1                                                ;
00000000  000EC          .LONG   0                                                ;
00000000' 000F0          .ADDRESS OUTPUT_FAB                                      ;
00000000  000F4          .LONG   0                                                ;
          000F8 INPUT_CHAN::                                                     ;
                         .BLKB   4
          000FC OUTPUT_CHAN::
                         .BLKB   4
      02  00100 INPUT_NAM::                                                      ;
                         .BYTE   2                                                ;
```

```
                60  00101          .BYTE   96
                FF  00102          .BYTE   -1
                00  00103          .BYTE   0
00000000'       00104          .ADDRESS RSBUF
                00  00108          .BYTE   0
                00  00109          .BYTE   0
                00  0010A          .BYTE   0
                00  0010B          .BYTE   0
00000000        0010C          .LONG   0
00000000        00110          .LONG   0
            0000#  00114          .WORD   0[8]
            0000#  00124          .WORD   0[3]
            0000#  0012A          .WORD   0[3]
00000000        00130          .LONG   0
00000000        00134          .LONG   0
                00  00138          .BYTE   0
                00  00139          .BYTE   0
                00  0013A          .BYTE   0
                00  0013B          .BYTE   0
                00  0013C          .BYTE   0
                00  0013D          .BYTE   0
                00#  0013E          .BYTE   0[2]
00000000        00140          .LONG   0
00000000        00144          .LONG   0
00000000        00148          .LONG   0
00000000        0014C          .LONG   0
00000000        00150          .LONG   0
00000000        00154          .LONG   0
00000000#       00158          .LONG   0[2]
                03  00160 INPUT_FAB::
                                .BYTE   3
                50  00161          .BYTE   80
              0000  00162          .WORD   0
000C0040        00164          .LONG   786496
00000000        00168          .LONG   0
00000000        0016C          .LONG   0
00000000        00170          .LONG   0
              0000  00174          .WORD   0
                02  00176          .BYTE   2
                00  00177          .BYTE   0
00000000        00178          .LONG   0
                00  0017C          .BYTE   0
                00  0017D          .BYTE   0
                04  0017E          .BYTE   4
                03  0017F          .BYTE   3
00000000        00180          .LONG   0
00000000        00184          .LONG   0
00000000'       00188          .ADDRESS INPUT_NAM
00000000'       0018C          .ADDRESS P.AAC
00000000'       00190          .ADDRESS P.AAD
                09  00194          .BYTE   9
                04  00195          .BYTE   4
              0000  00196          .WORD   0
00000000        00198          .LONG   0
              0000  0019C          .WORD   0
                00  0019E          .BYTE   0
                00  0019F          .BYTE   0
```

```
00000000  001A0          .LONG   0                                                                ;
00000000  001A4          .LONG   0                                                                ;
     0000  001A8          .WORD   0                                                                ;
       00  001AA          .BYTE   0                                                                ;
       00  001AB          .BYTE   0                                                                ;
00000000  001AC          .LONG   0                                                                ;
       01  001B0 INPUT_RAB::                                                                       ;
                         .BYTE   1                                                                ;
       44  001B1          .BYTE   68                                                               ;
     0000  001B2          .WORD   0                                                                ;
26000000  001B4          .LONG   637534208                                                        ;
00000000  001B8          .LONG   0                                                                ;
00000000  001BC          .LONG   0                                                                ;
     0000# 001C0          .WORD   0[3]                                                             ;
     0000  001C6          .WORD   0                                                                ;
00000000  001C8          .LONG   0                                                                ;
     0000  001CC          .WORD   0                                                                ;
       00  001CE          .BYTE   0                                                                ;
       00  001CF          .BYTE   0                                                                ;
     0000  001D0          .WORD   0                                                                ;
     0000  001D2          .WORD   0                                                                ;
00000000  001D4          .LONG   0                                                                ;
00000000  001D8          .LONG   0                                                                ;
00000000  001DC          .LONG   0                                                                ;
00000000  001E0          .LONG   0                                                                ;
       00  001E4          .BYTE   0                                                                ;
       00  001E5          .BYTE   0                                                                ;
       00  001E6          .BYTE   0                                                                ;
       00  001E7          .BYTE   0                                                                ;
00000000  001E8          .LONG   0                                                                ;
00000000' 001EC          .ADDRESS INPUT_FAB                                                       ;
00000000  001F0          .LONG   0                                                                ;

                    DEVCHAR=          INPUT_FAB+64
                         .EXTRN  CREATE_LOGICAL, SET_TERMINAL_HANGUP
                         .EXTRN  EXIT_PROCESS, HANDLER
                         .EXTRN  LGI$_OPENIN, LGI$_INPUTERR
                         .EXTRN  LGI$_OUTPUTERR, LGI$_CMDINPUT
                         .EXTRN  PCB_STS, SYS$GB_RETRY_TMO
                         .EXTRN  CTL$AG_CLIDATA, SYS$OPEN
                         .EXTRN  SYS$CLOSE, SYS$CONNECT

                         .PSECT  $CODE$,NOWRT,2

                   0FFC 00000    .ENTRY  OPEN_INPUT, Save R2,R3,R4,R5,R6,R7,R8,R9,-          ; 0661
                                         R10,R11
     5B 00000000G   00  9E 00002         MOVAB   SYS$OPEN, R11                                ;
     5A      0000'  CF  9E 00009         MOVAB   P.AAG, R10                                   ;
     59 00000000G   00  9E 0000E         MOVAB   LIB$STOP, R9                                 ;
     58 00000000G   8F  D0 00015         MOVL    #LGI$_INPUTERR, R8                           ;
     57 00000000G   00  9E 0001C         MOVAB   PPD+30, R7                                   ;
     56      0000'  CF  9E 00023         MOVAB   INPUT_FAB+12, R6                             ;
     5E      FEF4   CE  9E 00028         MOVAB   -268(SP), SP                                 ;
     6D 00000000G   00  9E 0002D         MOVAB   HANDLER, (FP)                                ; 0693
63   A6 00000000G   00  90 00034         MOVB    SYS$GB_RETRY_TMO, INPUT_RAB+31               ; 0697
              F4   A6  9F 0003C         PUSHAB  INPUT_FAB                                    ; 0702
     6B            01  FB 0003F         CALLS   #1, SYS$OPEN                                 ;
```

```
                                52              50  DD 00042          MOVL    R0, STATUS                                      ;
                                0B              52  E8 00045          BLBS    STATUS, 1$                                      ;
                                                66  DD 00048          PUSHL   INPUT_FAB+12                                    ; 0704
                                                52  DD 0004A          PUSHL   STATUS
                                                7E  D4 0004C          CLRL    -(SP)
                                                58  DD 0004E          PUSHL   R8
                                69              04  FB 00050          CALLS   #4, LIB$STOP                                    ;
                    25      34  A6              02  E1 00053 1$:      BBC     #2, DEVCHAR, 2$                                 ; 0710
                                        F4      A6  9F 00058          PUSHAB  INPUT_FAB                                       ; 0713
                        00000000G   00          01  FB 0005B          CALLS   #1, SYS$CLOSE                                   ;
                                0A  A6          03  90 00062          MOVB    #3, INPUT_FAB+22                                ; 0714
                                        F4      A6  9F 00066          PUSHAB  INPUT_FAB                                       ; 0715
                                6B              01  FB 00069          CALLS   #1, SYS$OPEN                                    ;
                                52              50  D0 0006C          MOVL    R0, STATUS                                      ;
                                0B              52  E8 0006F          BLBS    STATUS, 2$                                      ;
                                                66  DD 00072          PUSHL   INPUT_FAB+12                                    ; 0717
                                                52  DD 00074          PUSHL   STATUS
                                                7E  D4 00076          CLRL    -(SP)
                                                58  DD 00078          PUSHL   R8
                                69              04  FB 0007A          CALLS   #4, LIB$STOP                                    ;
                                        44      A6  9F 0007D 2$:      PUSHAB  INPUT_RAB                                       ; 0723
                        00000000G   00          01  FB 00080          CALLS   #1, SYS$CONNECT                                 ;
                                52              50  D0 00087          MOVL    R0, STATUS                                      ;
                                0C              52  E8 0008A          BLBS    STATUS, 3$                                      ;
                                        50      A6  DD 0008D          PUSHL   INPUT_RAB+12                                    ; 0725
                                                52  DD 00090          PUSHL   STATUS
                                                7E  D4 00092          CLRL    -(SP)
                                                58  DD 00094          PUSHL   R8
                                69              04  FB 00096          CALLS   #4, LIB$STOP                                    ;
                                    47  A6      40  8F 88 00099 3$:    BISB2   #64, INPUT_RAB+3                                ; 0727
                46      A6              08      06  02 F0 0009E          INSV    #2, #6, #8, INPUT_RAB+2                         ; 0728
                                        8C  A6      66  D0 000A4          MOVL    INPUT_FAB+12, INPUT_CHAN                        ; 0729
                                            67      66  B0 000A8          MOVW    INPUT_FAB+12, PPD+30                            ; 0730
                                        26  A7  34  A6  D0 000AB          MOVL    INPUT_FAB+64, PPD+68                            ; 0731
                                        02  A7  F6  A6  B0 000B0          MOVW    INPUT_FAB+2, PPD+32                             ; 0732
                                        04  A7  46  A6  B0 000B5          MOVW    INPUT_RAB+2, PPD+34                             ; 0733
                        50      C9  A6              01  04 EF 000BA          EXTZV   #4, #1, INPUT_NAM+53, R0                        ; 0734
                E4      A7              01          05  50 F0 000C0          INSV    R0, #5, #1, PPD+2                               ;
                                0A  A7      A8  A6      1C 28 000C6          MOVC3   #28, INPUT_NAM+20, PPD+40                       ; 0736
                                            08  AE      1B B0 000CC          MOVW    #27, BUFFER                                     ; 0738
                                            0A  AE  F6  A6 B0 000D0          MOVW    INPUT_FAB+2, BUFFER+2                           ; 0739
                                                50  A8  A6 9A 000D5          MOVZBL  INPUT_NAM+20, R0                                ; 0740
                                0C  AE          A9  A6  50 28 000D9          MOVC3   R0, INPUT_NAM+21, BUFFER+4                      ; 0741
                                                    83  3A 90 000DF          MOVB    #58, (PTR)+                                     ; 0742
                                                    50  08 AE 9E 000E2          MOVAB   BUFFER, R0                                      ; 0743
                                            6E      53  C3 000E6          SUBL3   R0, PTR, BUFDESC                                ;
                                            04  AE  08  AE 9E 000EA          MOVAB   BUFFER, BUFDESC+4                               ; 0744
                                        05      C9  A6  04 E1 000EF          BBC     #4, INPUT_NAM+53, 4$                            ; 0749
                                                    50  6A 9E 000F4          MOVAB   P.AAG, R0                                       ; 0750
                                                        04 11 000F7          BRB     5$
                                                    50  04 AA 9E 000F9 4$:    MOVAB   P.AAH, R0                                       ; 0751
                                                        50 DD 000FD 5$:      PUSHL   R0
                                                        01 DD 000FF          PUSHL   #1                                              ; 0746
                                                    08  AE 9F 00101          PUSHAB  BUFDESC
                                                    F8  AA 9F 00104          PUSHAB  P.AAE
                                    00000000G   00          04 FB 00107          CALLS   #4, CREATE_LOGICAL                      ;
                                06      C9  A6              04 E1 0010E          BBC     #4, INPUT_NAM+53, 6$                    ; 0756
```

```
                        50      1C  AA  9E 00113          MOVAB   P.AAK, R0                          ; 0757
                                04  11  00117              BRB     7$                              ;
                        50      20  AA  9E 00119 6$:       MOVAB   P.AAL, R0                       ; 0758
                                50  DD  0011D 7$:          PUSHL   R0                              ;
                                01  DD  0011F              PUSHL   #1                              ; 0753
                            08  AE  9F  00121              PUSHAB  BUFDESC                         ;
                            14  AA  9F  00124              PUSHAB  P.AAI                           ;
                00000000G   00      04  FB  00127          CALLS   #4, CREATE_LOGICAL              ;
                                    04  0012E              RET                                     ; 0760

; Routine Size:  303 bytes,    Routine Base:  $CODE$ + 0000
```

```
;  364          07e1  1 GLOBAL ROUTINE close_input: NOVALUE =
;  365          0762  1
;  366          0763  1 !---
;  367          0764  1 !
;  368          0765  1 !       Close the primary input file, so that another may be opened.
;  369          0766  1 !       This is done in batch jobs with more than one job step.
;  370          0767  1 !
;  371          0768  1 ! Inputs:
;  372          0769  1 !
;  373          0770  1 !       input_fab = FAB for input file
;  374          0771  1 !
;  375          0772  1 ! Outputs:
;  376          0773  1 !
;  377          0774  1 !       All errors are ignored.
;  378          0775  1 !---
;  379          0776  1
;  380          0777  2 BEGIN
;  381          0778  2
;  382          0779  2 $CLOSE(FAB = input_fab);                ! Close input file
;  383          0780  2
;  384          0781  1 END;


                                    0000 00000          .ENTRY  CLOSE_INPUT, Save nothing                       ; 0761
                            0000'   CF  9F 00002        PUSHAB  INPUT_FAB                                       ; 0779
                 00000000G  00          01  FB 00006    CALLS   #1, SYS$CLOSE                                   ;
                                        04  0000D        RET                                                     ; 0781

; Routine Size:  14 bytes,    Routine Base:  $CODE$ + 012F
```

```
;  386     0782  1 GLOBAL ROUTINE open_output: NOVALUE =
;  387     0783  1
;  388     0784  1 !---
;  389     0785  1 !
;  390     0786  1 !       This routine opens the primary output file.  It also defines
;  391     0787  1 !       the logical names SYS$OUTPUT and SYS$ERROR.  SYS$OUTPUT and
;  392     0788  1 !       SYS$ERROR are always defined as executive mode logical names
;  393     0789  1 !       to contain the IFI of the output stream.
;  394     0790  1 !
;  395     0791  1 ! Inputs:
;  396     0792  1 !
;  397     0793  1 !       Access mode is executive.
;  398     0794  1 !
;  399     0795  1 ! Outputs:
;  400     0796  1 !
;  401     0797  1 !       output_fab/rab = FAB/RAB of SYS$OUTPUT stream
;  402     0798  1 !
;  403     0799  1 !       The PPF logical names SYS$OUTPUT and SYS$ERROR are created.
;  404     0800  1 !---
;  405     0801  1
;  406     0802  2 BEGIN
;  407     0803  2
;  408     0804  2 BUILTIN FP;
;  409     0805  2
;  410     0806  2 BIND
;  411     0807  2     lgi = .ppd [ppd$l_lgi]: BBLOCK;     ! Address the LGI area
;  412     0808  2
;  413     0809  2 LOCAL
;  414     0810  2     ptr,
;  415     0811  2     buffer:     BBLOCK [4+nam$c_maxrss],! Buffer for equivalence string
;  416     0812  2     bufdesc:    VECTOR [2],             ! Descriptor of above buffer
;  417     0813  2     status;
;  418     0814  2
;  419     0815  2 .fp = handler;                          ! Enable condition handler
;  420     0816  2
;  421     0817  2 status = $CREATE(FAB = output_fab);     ! Create SYS$OUTPUT file
;  422     0818  2
;  423     0819  2 !
;  424     0820  2 ! If an error was detected, and this is a network job, then allow the
;  425     0821  2 ! job to continue by connecting the output stream to NL:.  This is done
;  426     0822  2 ! so that network jobs are more robust, and proceed even in the case where
;  427     0823  2 ! the user has run out of disk quota, or the disk is out of space.
;  428     0824  2 !
;  429     0825  2
;  430     0826  2 IF NOT .status                          ! If error detected,
;  431     0827  2     AND .pcb_sts [$BITPOSITION(pcb$v_netwrk)] ! and this is a network job,
;  432     0828  2 THEN
;  433     0829  3     BEGIN
;  434     0830  3     output_fab [fab$l_fna] = UPLIT BYTE('_NL:');
;  435     0831  3     output_fab [fab$b_fns] = 4;
;  436     0832  3     output_fab [fab$b_dns] = 0;
;  437     0833  3     status = $CREATE(FAB = output_fab); ! Create SYS$OUTPUT file to NL:
;  438     0834  2     END;
;  439     0835  2
;  440     0836  2 IF NOT .status                          ! If error detected,
;  441     0837  2 THEN
;  442     0838  2     SIGNAL_STOP(lgi$_outputerr,0,       ! then signal fatal error
```

```
;  443          0839  2                    .status, .output_fab [fab$l_stv]);
;  444          0840  2
;  445          0841  2 status = $CONNECT(RAB = output_rab);    ! Connect to SYS$OUTPUT file
;  446          0842  2
;  447          0843  2 IF NOT .status                           ! If error detected,
;  448          0844  2 THEN
;  449          0845  2     SIGNAL_STOP(lgi$_outputerr,0,       ! then signal fatal error
;  450          0846  2                 .status, .output_rab [rab$l_stv]);
;  451          0847  2
;  452          0848  2 output_rab [rab$v_ppf_ind] = true;      ! Mark ok to use this RAB in user mode
;  453          0849  2 output_rab [rab$v_ppf_rat] = fab$m_cr;  ! Set default RAT to CR mode
;  454          0850  2
;  455          0851  2 output_chan = .output_fab [fab$l_stv];  ! Save exec channel if terminal
;  456          0852  2 ppd [ppd$l_outdev] = .output_fab [fab$l_dev];    ! Save device characteristics
;  457          0853  2 ppd [ppd$w_outifi] = .output_fab [fab$w_ifi];    ! and IFI
;  458          0854  2 ppd [ppd$w_outisi] = .output_rab [rab$w_isi];    ! and ISI
;  459          0855  2 ppd [ppd$v_outccl] = .output_nam [nam$v_cncl_dev];      ! and concealed attr
;  460          0856  2
;  461          0857  2 CH$MOVE(ppd$c_dvifid, output_nam [nam$t_dvi], ppd [ppd$t_outdvi]);
;  462          0858  2
;  463          0859  2 buffer [0,0,16,0] = 27;                 ! Escape character
;  464          0860  2 buffer [2,0,16,0] = .output_fab [fab$w_ifi];
;  465          0861  2 ptr = CH$MOVE(CH$RCHAR(output_nam [nam$t_dvi]),
;  466          0862  2               output_nam [$BYTEOFFSET(nam$t_dvi)+1,0,0,0], buffer[4,0,0,0]);
;  467          0863  2 CH$WCHAR_A(':', ptr);                    ! Append a colon to device name
;  468          0864  2 bufdesc[0] = CH$DIFF(.ptr, buffer);
;  469          0865  2 bufdesc[1] = buffer;
;  470          0866  2
;  471          0867  2 create_logical(%ASCID 'SYS$OUTPUT',     ! Re-define SYS$OUTPUT
;  472          0868  2                 bufdesc,
;  473          0869  2                 psl$c_exec,
;  474          0870  3                 (IF .output_nam [nam$v_cncl_dev]
;  475          0871  3                  THEN UPLIT(lnm$m_terminal OR lnm$m_concealed)
;  476          0872  2                  ELSE UPLIT(lnm$m_terminal)));
;  477          0873  2
;  478          0874  2 create_logical(%ASCID 'SYS$ERROR',      ! Define exec mode SYS$ERROR
;  479          0875  2                 bufdesc,
;  480          0876  2                 psl$c_exec,
;  481          0877  3                 (IF .output_nam [nam$v_cncl_dev]
;  482          0878  3                  THEN UPLIT(lnm$m_terminal OR lnm$m_concealed)
;  483          0879  2                  ELSE UPLIT(lnm$m_terminal)));
;  484          0880  2
;  485          0881  1 END;


                                                                .PSECT  $PLIT$,NOWRT,NOEXE,2

                                                  3A  4C  4E  5F  00054 P.AAK:  .ASCII  \:NL:\                          ;
                00  00  54  55  50  54  55  4F  24  53  59  53  00058 P.AAO:  .ASCII  \SYS$OUTPUT\<0><0>                  ;
                                                    010E000A  00064 P.AAN:  .LONG   17694730                            ;
                                                    00000000'  00068        .ADDRESS P.AAO                              ;
                                                    00000300  0006C P.AAP:  .LONG   768                                 ;
                                                    00000200  00070 P.AAQ:  .LONG   512                                 ;
                00  00  00  52  4F  52  52  45  24  53  59  53  00074 P.AAS:  .ASCII  \SYS$ERROR\<0><0><0>               ;
                                                    010E0009  00080 P.AAR:  .LONG   17694729                            ;
                                                    00000000'  00084        .ADDRESS P.AAS                              ;
```

```
                        00000300  00088 P.AAT:  .LONG   768                                                      ;
                        00000200  0008C P.AAU:  .LONG   512                                                      ;

                                              .EXTRN  SYS$CREATE

                                              .PSECT  $CODE$,NOWRT,2

                                  0FFC 00000  .ENTRY  OPEN_OUTPUT, Save R2,R3,R4,R5,R6,R7,R8,R9,-          ; 0782
                                              R10,R11
                    5B 00000000G   00   9E 00002  MOVAB  LIB$STOP, R11                                          ;
                    5A 00000000G   8F   D0 00009  MOVL   #LGI$_OUTPUTERR, R10                                   ;
                    59 00000000G   00   9E 00010  MOVAB  SYS$CREATE, R9                                         ;
                    58      0000'  CF   9E 00017  MOVAB  P.AAM, R8                                             ;
                    57 00000000G   00   9E 0001C  MOVAB  PPD+100, R7                                           ;
                    56      0000'  CF   9E 00023  MOVAB  OUTPUT_RAB+2, R6                                      ;
                    5E      FEF4   CE   9E 00028  MOVAB  -268(SP), SP                                          ;
                    6D 00000000G   00   9E 0002D  MOVAB  HANDLER, (FP)                                         ; 0815
                            AE     A6   9F 00034  PUSHAB OUTPUT_FAB                                            ; 0817
                    69             01   FB 00037  CALLS  #1, SYS$CREATE                                        ;
                    52             50   D0 0003A  MOVL   R0, STATUS                                           ;
                    28             52   E8 0003D  BLBS   STATUS, 2$                                           ; 0826
                11 00000000G       00   E1 00040  BBC    #5, PCB_STS+2, 1$                                    ; 0827
                        DA     A6   68   9E 00048  MOVAB  P.AAM, OUTPUT_FAB+44                                  ; 0830
                        E2     A6   04   B0 0004C  MOVW   #4, OUTPUT_FAB+52                                     ; 0831
                            AE     A6   9F 00050  PUSHAB OUTPUT_FAB                                            ; 0833
                    69             01   FB 00053  CALLS  #1, SYS$CREATE                                        ;
                    52             50   D0 00056  MOVL   R0, STATUS                                           ;
                    0C             52   E8 00059  1$:    BLBS   STATUS, 2$                                     ; 0836
                            BA     A6   DD 0005C  PUSHL  OUTPUT_FAB+12                                         ; 0839
                                   52   DD 0005F  PUSHL  STATUS                                               ;
                                   7E   D4 00061  CLRL   -(SP)                                                 ; 0838
                                   5A   DD 00063  PUSHL  R10                                                   ;
                                6B      04   FB 00065  CALLS  #4, LIB$STOP                                     ;
                                    FE  A6   9F 00068  2$:    PUSHAB OUTPUT_RAB                                 ; 0841
                        00000000G   00      01   FB 0006B  CALLS  #1, SYS$CONNECT                              ;
                                    52      50   D0 00072  MOVL   R0, STATUS                                   ;
                                    0C      52   E8 00075  BLBS   STATUS, 3$                                   ; 0843
                                        0A  A6   DD 00078  PUSHL  OUTPUT_RAB+12                                 ; 0846
                                            52   DD 0007B  PUSHL  STATUS                                       ;
                                            7E   D4 0007D  CLRL   -(SP)                                         ; 0845
                                            5A   DD 0007F  PUSHL  R10                                           ;
                                    6B          04   FB 00081  CALLS  #4, LIB$STOP                             ;
                                      01        A6       40  8F   88 00084  3$:    BISB2  #64, OUTPUT_RAB+3   ; 0848
                        66                  08  06       02  F0   00089         INSV   #2, #6, #8, OUTPUT_RAB+2 ; 0849
                                      46        A6       BA  A6   D0 0008E         MOVL   OUTPUT_FAB+12, OUTPUT_CHAN ; 0851
                                      67        EE       A6  D0   00093         MOVL   OUTPUT_FAB+64, PPD+100   ; 0852
                                      C0        A7       B0  A6   B0 00097         MOVW   OUTPUT_FAB+2, PPD+36     ; 0853
                                      C2        A7           66   B0 0009C         MOVW   OUTPUT_RAB+2, PPD+38     ; 0854
                        50      83    A6        01           04   EF 000A0         EXTZV  #4, #1, OUTPUT_NAM+53, R0 ; 0855
                    9E  A7              01        06           50   F0 000A6         INSV   R0, #6, #1, PPD+2        ;
                            E4  A7  FF62          C6           1C   28 000AC         MOVC3  #28, OUTPUT_NAM+20, PPD+72 ; 0857
                                      08          AE           1B   B0 000B3         MOVW   #27, BUFFER              ; 0859
                                      0A          AE       B0  A6   B0 000B7         MOVW   OUTPUT_FAB+2, BUFFER+2   ; 0860
                                          50      FF62     C6   9A   000BC         MOVZBL OUTPUT_NAM+20, R0        ; 0861
                                0C    AE    FF63  C6           50   28 000C1         MOVC3  R0, OUTPUT_NAM+21, BUFFER+4 ; 0862
                                                  83           3A   90 000C8         MOVB   #58, (PTR)+              ; 0863
                                          50      08       AE  9E   000CB         MOVAB  BUFFER, R0               ; 0864
```

```
                6E              53              50  C3 000CF        SUBL3   R0, PTR, BUFDESC                        ;
                          04    AE        08    AE  9E 000D3        MOVAB   BUFFER, BUFDESC+4                    ; 0865
                06        83    A6              04  E1 000D8        BBC     #4, OUTPUT_NAM+53, 4$               ; 0870
                              50        18    A8  9E 000DD        MOVAB   P.AAP, R0                           ; 0871
                                              04  11 000E1        BRB     5$                                  ;
                              50        1C    A8  9E 000E3 4$:     MOVAB   P.AAQ, R0                           ; 0872
                                              50  DD 000E7 5$:     PUSHL   R0                                  ;
                                              01  DD 000E9        PUSHL   #1                                  ; 0867
                                          08  AE  9F 000EB        PUSHAB  BUFDESC                             ;
                                          10  A8  9F 000EE        PUSHAB  P.AAN                               ;
                    00000000G             00      04  FB 000F1        CALLS   #4, CREATE_LOGICAL              ;
                06        83    A6              04  E1 000F8        BBC     #4, OUTPUT_NAM+53, 6$               ; 0877
                              50        34    A8  9E 000FD        MOVAB   P.AAT, R0                           ; 0878
                                              04  11 00101        BRB     7$                                  ;
                              50        38    A8  9E 00103 6$:     MOVAB   P.AAU, R0                           ; 0879
                                              50  DD 00107 7$:     PUSHL   R0                                  ;
                                              01  DD 00109        PUSHL   #1                                  ; 0874
                                          08  AE  9F 0010B        PUSHAB  BUFDESC                             ;
                                          2C  A8  9F 0010E        PUSHAB  P.AAR                               ;
                    00000000G             00      04  FB 00111        CALLS   #4, CREATE_LOGICAL              ;
                                                  04  00118        RET                                         ; 0881

; Routine Size:  281 bytes,    Routine Base:  $CODE$ + 013D
```

```
;  487          0882  1 GLOBAL ROUTINE close_output: NOVALUE =
;  488          0883  1
;  489          0884  1 !---
;  490          0885  1 !
;  491          0886  1 !       Close the primary output file, so that it may be spooled to
;  492          0887  1 !       the print queue.
;  493          0888  1 !
;  494          0889  1 ! Inputs:
;  495          0890  1 !
;  496          0891  1 !       output_fab = FAB for output file
;  497          0892  1 !       output_rab = RAB for output file
;  498          0893  1 !
;  499          0894  1 ! Outputs:
;  500          0895  1 !
;  501          0896  1 !       All errors are ignored.
;  502          0897  1 !---
;  503          0898  1
;  504          0899  2 BEGIN
;  505          0900  2
;  506          0901  2 $FLUSH(RAB = output_rab);               ! Force update
;  507          0902  2 $CLOSE(FAB = output_fab);               ! Close output file
;  508          0903  2
;  509          0904  1 END;


                                                .EXTRN  SYS$FLUSH

                                0000 00000      .ENTRY  CLOSE_OUTPUT, Save nothing                      ; 0882
                        0000'   CF   9F 00002   PUSHAB  OUTPUT_RAB                                      ; 0901
                00000000G   00      01   FB 00006   CALLS   #1, SYS$FLUSH
                        0000'   CF   9F 0000D   PUSHAB  OUTPUT_FAB                                      ; 0902
                00000000G   00      01   FB 00011   CALLS   #1, SYS$CLOSE
                                    04   00018   RET                                                   ; 0904

; Routine Size:  25 bytes,    Routine Base:  $CODE$ + 0256
```

```
;  511          0905  1 GLOBAL ROUTINE write_file (filespec): NOVALUE =
;  512          0906  1
;  513          0907  1 !---
;  514          0908  1 !
;  515          0909  1 !       Write the contents of a file to SYS$OUTPUT
;  516          0910  1 !
;  517          0911  1 ! Inputs:
;  518          0912  1 !
;  519          0913  1 !       filespec = Address of filespec descriptor
;  520          0914  1 !
;  521          0915  1 ! Outputs:
;  522          0916  1 !
;  523          0917  1 !       None
;  524          0918  1 !---
;  525          0919  1
;  526          0920  2 BEGIN
;  527          0921  2
;  528          0922  2 MAP
;  529          0923  2     filespec:   REF VECTOR;             ! Address of descriptor
;  530          0924  2
;  531          0925  2 LOCAL
;  532          0926  2     fab:        BBLOCK [fab$c_bln],     ! FAB for file access
;  533          0927  2     rab:        BBLOCK [rab$c_bln],     ! RAB for file access
;  534          0928  2     buffer:     VECTOR [128,BYTE],      ! Input buffer
;  535          0929  2     status;
;  536          0930  2
;  537        P 0931  2 $FAB_INIT(FAB = fab,
;  538        P 0932  2           FNS = .filespec [0],          ! Filespec
;  539        P 0933  2           FNA = .filespec [1],
;  540        P 0934  2           DNM = '.LIS',                 ! Default filespec
;  541        P 0935  2           FAC = GET,                    ! Read only
;  542          0936  2           FOP = SQO);                   ! Sequential only optimization
;  543          0937  2
;  544        P 0938  2 $RAB_INIT(RAB = rab,
;  545        P 0939  2           FAB = fab,                    ! Address of associated FAB
;  546        P 0940  2           UBF = buffer,                 ! Address of input buffer
;  547          0941  2           USZ = 128);
;  548          0942  2
;  549          0943  2 status = $OPEN(FAB = fab);              ! Open the file
;  550          0944  2
;  551          0945  2 IF NOT .status                          ! If error detected,
;  552          0946  2 THEN
;  553          0947  3     BEGIN
;  554          0948  3     SIGNAL(lgi$_openin,1,.filespec,.status,.fab [fab$l_stv]);
;  555          0949  3     RETURN;
;  556          0950  2     END;
;  557          0951  2
;  558          0952  2 status = $CONNECT(RAB = rab);           ! Connect to stream
;  559          0953  2
;  560          0954  2 IF NOT .status                          ! If error detected,
;  561          0955  2 THEN
;  562          0956  3     BEGIN
;  563          0957  3     SIGNAL(lgi$_openin,1,.filespec,.status,.rab [rab$l_stv]);
;  564          0958  3     $CLOSE(FAB = fab);                  ! Close file
;  565          0959  3     RETURN;
;  566          0960  2     END;
;  567          0961  2
```

```
;  568          0962  3 WHILE (status = $GET(RAB = rab))        ! For each record which can be read,
;  569          0963  2 DO
;  570          0964  2     BEGIN
;  571          0965  2     LOCAL
;  572          0966  2         desc:   VECTOR [2];
;  573          0967  2
;  574          0968  2     desc [0] = .rab [rab$w_rsz];        ! Construct descriptor of record
;  575          0969  2     desc [1] = .rab [rab$l_rbf];
;  576          0970  2
;  577          0971  2     write_output(desc);                 ! Write to SYS$OUTPUT
;  578          0972  2     END;
;  579          0973  2
;  580          0974  2 IF .status NEQ rms$_eof                 ! If loop didn't end normally,
;  581          0975  2 THEN
;  582          0976  2     SIGNAL(lgi$_openin,1,.filespec,.status,.rab [rab$l_stv]);
;  583          0977  2
;  584          0978  2 $CLOSE(FAB = fab);                      ! Close file
;  585          0979  2
;  586          0980  1 END;


                                                .PSECT  $PLIT$,NOWRT,NOEXE,2
                                53  49  4C  2E  00090 P.AAV:  .ASCII  \.LIS\                                    ;
                                                .EXTRN  SYS$GET
                                                .PSECT  $CODE$,NOWRT,2
                                                    01FC 00000          .ENTRY  WRITE_FILE, Save R2,R3,R4,R5,R6,R7,R8       ; 0905
                                            58 00000000G    00  9E 00002    MOVAB   LIB$SIGNAL, R8                          ;
                                            57 00000000G    8F  D0 00009    MOVL    #LGI$_OPENIN, R7                        ;
                                            5E      FEE4    CE  9E 00010    MOVAB   -284(SP), SP                            ;
    0050    8F      00                      6E              00  2C 00015    MOVC5   #0, (SP), #0, #80, $RMS_PTR             ; 0936
                                                        B0  AD      0001C                                                   ;
                                        B0  AD      5003    8F  B0 0001E    MOVW    #20483, $RMS_PTR                        ;
                                        B4  AD        40    8F  9A 00024    MOVZBL  #64, $RMS_PTR+4                         ;
                                        C6  AD              02  90 00029    MOVB    #2, $RMS_PTR+22                         ;
                                        CF  AD              02  90 0002D    MOVB    #2, $RMS_PTR+31                         ;
                                            56        04    AC  D0 00031    MOVL    FILESPEC, R6                            ;
                                        DC  AD        04    A6  D0 00035    MOVL    4(R6), $RMS_PTR+44                      ;
                                        E0  AD      0000'   CF  9E 0003A    MOVAB   P.AAV, $RMS_PTR+48                      ;
                                        E4  AD              66  90 00040    MOVB    (R6), $RMS_PTR+52                       ;
                                        E5  AD              04  90 00044    MOVB    #4, $RMS_PTR+53                         ;
    0044    8F      00                      6E              00  2C 00048    MOVC5   #0, (SP), #0, #68, $RMS_PTR             ; 0941
                                                    0088    CE      0004F                                                   ;
                                    0088    CE      4401    8F  B0 00052    MOVW    #17409, $RMS_PTR                        ;
                                      8C    AD        80    8F  9B 00059    MOVZBW  #128, $RMS_PTR+32                       ;
                                      90    AD        08    AE  9E 0005E    MOVAB   BUFFER, $RMS_PTR+36                     ;
                                      A8    AD        B0    AD  9E 00063    MOVAB   FAB, $RMS_PTR+60                        ;
                                                    B0    AD  9F 00068    PUSHAB  FAB                                     ; 0943
                                00000000G   00              01  FB 0006B    CALLS   #1, SYS$OPEN                            ;
                                            52              50  D0 00072    MOVL    R0, STATUS                              ;
                                            0F              52  E8 00075    BLBS    STATUS, 1$                              ; 0945
                                                        BC  AD  DD 00078    PUSHL   FAB+12                                  ; 0948
                                                            52  DD 0007B    PUSHL   STATUS                                  ;
```

```
                                 56  DD 0007D          PUSHL   R6                                                      ;
                                 01  DD 0007F          PUSHL   #1                                                      ;
                                 57  DD 00081          PUSHL   R7                                                      ;
                           68    05  FB 00083          CALLS   #5, LIB$SIGNAL                                          ;
                                 04  00086          RET                                                             ; 0947
                         0088    CE  9F 00087 1$:    PUSHAB  RAB                                                     ; 0952
                00000000G  00    01  FB 0008B          CALLS   #1, SYS$CONNECT                                         ;
                           52    50  D0 00092          MOVL    R0, STATUS                                              ;
                           2C    52  E9 00095          BLBC    STATUS, 4$                                              ; 0954
                         0088    CE  9F 00098 2$:    PUSHAB  RAB                                                     ; 0962
                00000000G  00    01  FB 0009C          CALLS   #1, SYS$GET                                             ;
                           52    50  D0 000A3          MOVL    R0, STATUS                                              ;
                           12    52  E9 000A6          BLBC    STATUS, 3$                                              ;
                           6E  8E  AD  3C 000A9          MOVZWL  RAB+34, DESC                                            ; 0968
                      04   AE  94  AD  D0 000AD          MOVL    RAB+40, DESC+4                                          ; 0969
                                 5E  DD 000B2          PUSHL   SP                                                      ; 0971
                     0000V  CF      01  FB 000B4          CALLS   #1, WRITE_OUTPUT                                        ;
                                 DD  11 000B9          BRB     2$                                                      ; 0962
                0001827A  8F      52  D1 000BB 3$:    CMPL    STATUS, #98938                                          ; 0974
                                 0F  13 000C2          BEQL    5$                                                      ;
                         FF78    CD  DD 000C4 4$:    PUSHL   RAB+12                                                  ; 0976
                                 52  DD 000C8          PUSHL   STATUS                                                  ;
                                 56  DD 000CA          PUSHL   R6                                                      ;
                                 01  DD 000CC          PUSHL   #1                                                      ;
                                 57  DD 000CE          PUSHL   R7                                                      ;
                           68    05  FB 000D0          CALLS   #5, LIB$SIGNAL                                          ;
                             B0  AD  9F 000D3 5$:    PUSHAB  FAB                                                     ; 0978
                00000000G  00    01  FB 000D6          CALLS   #1, SYS$CLOSE                                           ;
                                 04  000DD          RET                                                             ; 0980

; Routine Size:  222 bytes,    Routine Base:  $CODE$ + 026F
```

```
;  588          0981  1 GLOBAL ROUTINE write_fao (ascic_ctlstr, fao_args): NOVALUE =
;  589          0982  1
;  590          0983  1 !---
;  591          0984  1 !
;  592          0985  1 !       Format a message and write it to the primary output stream.
;  593          0986  1 !
;  594          0987  1 ! Inputs:
;  595          0988  1 !
;  596          0989  1 !       ascic_ctlstr = Address of ASCIC FAO control string
;  597          0990  1 !       fao_args = First FAO argument (optional)
;  598          0991  1 !
;  599          0992  1 ! Outputs:
;  600          0993  1 !
;  601          0994  1 !       None
;  602          0995  1 !---
;  603          0996  1
;  504          0997  2 BEGIN
;  605          0998  2
;  606          0999  2 LOCAL
;  607          1000  2     ctlstr:     VECTOR [2],                     ! Descriptor of FAO string
;  608          1001  2     desc:       VECTOR [2],
;  609          1002  2     buffer:     VECTOR [128,BYTE];
;  610          1003  2
;  611          1004  2 ctlstr [0] = CH$RCHAR(.ascic_ctlstr);   ! Set up FAO string descriptor
;  612          1005  2 ctlstr [1] = .ascic_ctlstr+1;
;  613          1006  2
;  614          1007  2 desc [0] = 128;                         ! Set up result descriptor
;  615          1008  2 desc [1] = buffer;
;  616          1009  2
;  617        P 1010  2 $FAOL(CTRSTR = ctlstr,
;  618        P 1011  2       OUTLEN = desc,
;  619        P 1012  2       OUTBUF = desc,
;  620          1013  2       PRMLST = fao_args);
;  621          1014  2
;  622          1015  2 write_output(desc);
;  623          1016  2
;  624          1017  1 END;


                                                        .EXTRN  SYS$FAOL

                                                0000 00000      .ENTRY  WRITE_FAO, Save nothing                 ; 0981
                                5E      FF70    CE   9E 00002   MOVAB   -144(SP), SP                            ;
                            F8  AD        C4    BC   9A 00007   MOVZBL  @ASCIC_CTLSTR, CTLSTR                   ; 1004
                        FC  AD      04  AC          01   C1 0000C   ADDL3   #1, ASCIC_CTLSTR, CTLSTR+4          ; 1005
                            F0  AD        80    8F   9A 00012   MOVZBL  #128, DESC                             ; 1007
                            F4  AD              6E   9E 00017   MOVAB   BUFFER, DESC+4                         ; 1008
                                        08  AC  9F   0001B   PUSHAB  FAO_ARGS                                   ; 1013
                                        F0  AD  9F   0001E   PUSHAB  DESC
                                        F0  AD  9F   00021   PUSHAB  DESC
                                        F8  AD  9F   00024   PUSHAB  CTLSTR
                    00000000G   00          04   FB 00027   CALLS   #4, SYS$FAOL
                                        F0  AD  9F   0002E   PUSHAB  DESC                                      ; 1015
                        0000V   CF          01   FB 00031   CALLS   #1, WRITE_OUTPUT
                                                04 00036    RET                                                 ; 1017
```

```
; Routine Size:  55 bytes,    Routine Base:  $CODE$ + 034D
```

```
;  626          1018  1 GLOBAL ROUTINE write_output (recdesc, param, rab_param) =
;  627          1019  1
;  628          1020  1 !---
;  629          1021  1 !
;  630          1022  1 !       Write a record to the primary output stream
;  631          1023  1 !
;  632          1024  1 ! Inputs:
;  633          1025  1 !
;  634          1026  1 !       recdesc = Address of descriptor of record
;  635          1027  1 !       param = $PUTMSG actprm argument (not used)
;  636          1028  1 !       rab_param = address of rab to use (optional)
;  637          1029  1 !
;  638          1030  1 ! Outputs:
;  639          1031  1 !
;  640          1032  1 !       routine = 0 (when used as $PUTMSG action routine, tells
;  641          1033  1 !                       $PUTMSG not to output message itself)
;  642          1034  1 !---
;  643          1035  1
;  644          1036  2 BEGIN
;  645          1037  2
;  646          1038  2 BUILTIN
;  647          1039  2     ACTUALCOUNT;
;  648          1040  2
;  649          1041  2 MAP
;  650          1042  2     recdesc:    REF VECTOR;             ! Address of descriptor
;  651          1043  2
;  652          1044  2 BIND
;  653          1045  2     timeout = UPLIT(-30*10*1000*1000,-1); ! 30 seconds
;  654          1046  2
;  655          1047  2 LOCAL
;  656          1048  2     rab : REF $BBLOCK;
;  657          1049  2
;  658          1050  2 IF ACTUALCOUNT() GEQU 3
;  659          1051  2 THEN rab = .rab_param
;  660          1052  2 ELSE rab = output_rab;
;  661          1053  2
;  662          1054  2 IF .rab [rab$w_isi] EQL 0             ! If file not yet opened,
;  663          1055  2 THEN
;  664          1056  2     RETURN 0;                         ! then skip it
;  665          1057  2
;  666          1058  2 rab [rab$w_rsz] = .recdesc [0];
;  667          1059  2 rab [rab$l_rbf] = .recdesc [1];
;  668          1060  2
;  669        P 1061  2 $SETIMR(DAYTIM = timeout,             ! Set timeout timer going
;  670        P 1062  2         ASTADR = write_timeout,
;  671          1063  2         REQIDT = 99);
;  672          1064  2
;  673          1065  2 write_output_status = $PUT(RAB = .rab); ! Output message
;  674          1066  2
;  675          1067  2 $CANTIM(REQIDT = 99);                 ! Cancel the timer
;  676          1068  2
;  677          1069  2 RETURN 0;
;  678          1070  2
;  679          1071  1 END;
```

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

                FFFFFFFF  EE1E5D00  00094 P.AAW:  .LONG   -300000000, -1                                  ;

                                        TIMEOUT=        P.AAW
                                        .EXTRN  SYS$SETIMR, SYS$PUT
                                        .EXTRN  SYS$CANTIM

                                        .PSECT  $CODE$,NOWRT,2

                                   0004 00000           .ENTRY  WRITE_OUTPUT, Save P2                       ; 1018
                          03       6C  91 00002           CMPB    (AP), #3                                    ; 1050
                                   06  1F 00005           BLSSU   1$                                          ;
                          52    0C   AC  D0 00007           MOVL    RAB_PARAM, RAB                              ; 1051
                                   05  11 0000B           BRB     2$                                          ;
                          52  0000'  CF  9E 0000D 1$:      MOVAB   OUTPUT_RAB, RAB                            ; 1052
                             02      A2  B5 00012 2$:      TSTW    2(RAB)                                      ; 1054
                                   3D  13 00015           BEQL    3$                                          ;
                          50    04   AC  D0 00017           MOVL    RECDESC, R0                                 ; 1058
                       22 A2            60  B0 0001B           MOVW    (R0), 34(RAB)                               ;
                       28 A2       04   A0  D0 0001F           MOVL    4(R0), 40(RAB)                              ; 1059
                          7E       63   8F  9A 00024           MOVZBL  #99, -(SP)                                  ; 1063
                                0000V  CF  9F 00028           PUSHAB  WRITE_TIMEOUT                               ;
                                0000'  CF  9F 0002C           PUSHAB  TIMEOUT                                     ;
                                       7E  D4 00030           CLRL    -(SP)                                       ;
                00000000G      00            04  FB 00032           CALLS   #4, SYS$SETIMR                      ;
                                             52  DD 00039           PUSHL   RAB                                 ; 1065
                00000000G      00            01  FB 0003B           CALLS   #1, SYS$PUT                         ;
                     0000'     CF            50  D0 00042           MOVL    R0, WRITE_OUTPUT_STATUS             ;
                                             7E  D4 00047           CLRL    -(SP)                               ; 1067
                          7E       63   8F  9A 00049           MOVZBL  #99, -(SP)                          ;
                00000000G      00            02  FB 0004D           CALLS   #2, SYS$CANTIM                      ;
                                             50  D4 00054 3$:      CLRL    R0                                  ; 1071
                                                 04 00056           RET                                         ;

; Routine Size:  87 bytes,    Routine Base:  $CODE$ + 0384
```

```
;  681          1072  1 GLOBAL ROUTINE write_timeout: NOVALUE =
;  682          1073  1
;  683          1074  1 !---
;  684          1075  1 !
;  685          1076  1 !       The timeout has elapsed while trying to write to the primary
;  686          1077  1 !       output stream.  We assume that the user pressed control/s to
;  687          1078  1 !       inhibit completion of the write.  Cancel the I/O to force
;  688          1079  1 !       completion of the $PUT.
;  689          1080  1 !
;  690          1081  1 ! Inputs:
;  691          1082  1 !
;  692          1083  1 !       None
;  693          1084  1 !
;  694          1085  1 ! Outputs:
;  695          1086  1 !
;  696          1087  1 !       None
;  697          1088  1 !---
;  698          1089  1
;  699          1090  2 BEGIN
;  700          1091  2
;  701          1092  2 ROUTINE cancel_io =
;  702          1093  3     BEGIN
;  703          1094  3     $CANCEL(CHAN = .input_chan); ! Cancel the I/O to the terminal
;  704          1095  3     $CANCEL(CHAN = .output_chan);
;  705          1096  3     1
;  706          1097  2     END;


                                                        .EXTRN  SYS$CANCEL

                                0004 00000 CANCEL_IO:
                                                        .WORD   Save R2                         ; 1092
                             52 00000000G   00  9E 00002        MOVAB   SYS$CANCEL, R2          ;
                                    0000'   CF  DD 00009        PUSHL   INPUT_CHAN              ; 1094
                             62             01  FB 0000D        CALLS   #1, SYS$CANCEL          ;
                                    0000'   CF  DD 00010        PUSHL   OUTPUT_CHAN             ; 1095
                             62             01  FB 00014        CALLS   #1, SYS$CANCEL          ;
                             50             01  D0 00017        MOVL    #1, R0                  ; 1097
                                            04  0001A        RET                             ;

; Routine Size:  27 bytes,    Routine Base:  $CODE$ + 03DB

;  707          1098  2
;  708          1099  2 $CMEXEC(ROUTIN = cancel_io);           ! Cancel the I/O in exec (RMS) mode
;  709          1100  2
;  710          1101  2 RETURN true;
;  711          1102  2
;  712          1103  1 END;


                                                        .EXTRN  SYS$CMEXEC

                                  0000 00000        .ENTRY  WRITE_TIMEOUT, Save nothing     ; 1072
                                7E  D4 00002        CLRL    -(SP)                           ; 1099
```

```
                                 DE    AF  9F 00004         PUSHAB  CANCEL_IO                                    ;
                    00000000G  00        02  FB 00007         CALLS   #2, SYS$CMEXEC                          ;
                                         04 0000E         RET                                                 ; 1103

; Routine Size:  15 bytes,    Routine Base:  $CODE$ + 03F6
```

```
;  714          1104  1 GLOBAL ROUTINE get_input (rab, mode) : NOVALUE =
;  715          1105  1
;  716          1106  1 !---
;  717          1107  1 !
;  718          1108  1 !       Read a record from the primary input stream. A blanket timer is
;  719          1109  1 !       run around the read to handle obscure cases where the terminal
;  720          1110  1 !       driver character timeout might not work. This ensures that we
;  721          1111  1 !       will never end up with a hung job in LOGINOUT.
;  722          1112  1 !
;  723          1113  1 ! Inputs:
;  724          1114  1 !
;  725          1115  1 !       rab = Address of RAB to use
;  726          1116  1 !       mode = 0 for normal error message if error
;  727          1117  1 !            = 1 if just exit quietly on error
;  728          1118  1 !            = 2 like 0 except return if timeout (after signaling)
;  729          1119  1 !
;  730          1120  1 ! Outputs:
;  731          1121  1 !
;  732          1122  1 !       None.
;  733          1123  1 !
;  734          1124  1 !---
;  735          1125  1
;  736          1126  2 BEGIN
;  737          1127  2
;  738          1128  2 MAP
;  739          1129  2     rab: REF $BBLOCK;
;  740          1130  2
;  741          1131  2 LOCAL
;  742          1132  2     status;
;  743          1133  2
;  744          1134  2 BIND
;  745          1135  2     timeout = UPLIT (-300*10*1000*1000,-1); ! 5 minutes
;  746          1136  2
;  747        P 1137  2 $SETIMR (DAYTIM = timeout,              ! Set timeout timer going
;  748        P 1138  2          ASTADR = write_timeout,
;  749          1139  2          REQIDT = 99);
;  750          1140  2
;  751          1141  2 status = $GET (RAB = .rab);            ! Issue the read
;  752          1142  2
;  753          1143  2 $CANTIM (REQIDT = 99);                 ! Cancel the timer
;  754          1144  2
;  755          1145  2 !
;  756          1146  2 ! If an error occurs, either signal it or exit quietly, depending on
;  757          1147  2 ! caller's request.
;  758          1148  2 !
;  759          1149  2 IF NOT .status
;  760          1150  2 AND .status NEQ rms$_rtb
;  761          1151  3 AND (.status NEQ rms$_tmo OR .rab[rab$b_tmo] NEQ 0)
;  762          1152  2 THEN
;  763          1153  3     BEGIN
;  764          1154  3     IF NOT .mode
;  765          1155  3     THEN
;  766          1156  4         BEGIN
;  767          1157  4         IF .mode EQL 2
;  768          1158  4         AND .status EQL rms$_tmo
;  769          1159  4         THEN
;  770          1160  4             SIGNAL((lgi$_cmdinput AND NOT sts$m_severity) OR sts$k_warning, 0,
```

```
;  771          1161  4                   .status, .rab[rab$l_stv])
;  772          1162  4               ELSE
;  773          1163  4                   SIGNAL_STOP(lgi$_cmdinput, 0,
;  774          1164  4                       .status, .rab[rab$l_stv]);
;  775          1165  4               END
;  776          1166  3           ELSE
;  777          1167  4               BEGIN
;  778          1168  4               set_terminal_hangup(true);
;  779          1169  4               $CMEXEC(ROUTIN = exit_process);
;  780          1170  3               END;
;  781          1171  2           END;
;  782          1172  2
;  783          1173  1 END;


                                                    .PSECT  $PLIT$,NOWRT,NOEXE,2

                                FFFFFFFF  4D2FA200  0009C P.AAX:    .LONG   1294967296, -1                          ;

                                                    TIMEOUT=            P.AAX


                                                    .PSECT  $CODE$,NOWRT,2

                                            000C 00000              .ENTRY  GET_INPUT, Save R2,R3                   ; 1104
                                      7E      63    8F   9A 00002   MOVZBL  #99, -(SP)                              ; 1139
                                              E8    AF   9F 00006   PUSHAB  WRITE_TIMEOUT                          ;
                                            0000'   CF   9F 00009   PUSHAB  TIMEOUT                                ;
                                                    7E   D4 0000D   CLRL    -(SP)                                   ;
                          00000000G   00            04   FB 0000F   CALLS   #4, SYS$SETIMR                         ;
                                      52      04    AC   D0 00016   MOVL    RAB, R2                                 ; 1141
                                              52    DD   0001A      PUSHL   R2
                          00000000G   00            01   FB 0C01C   CALLS   #1, SYS$GET                          ;
                                      53            50   D0 00023   MOVL    R0, STATUS                          ;
                                                    7E   D4 00026   CLRL    -(SP)                                   ; 1143
                                      7E      63    8F   9A 00028   MOVZBL  #99, -(SP)                             ;
                          00000000G   00            02   FB 0002C   CALLS   #2, SYS$CANTIM                        ;
                                      6C            53   E8 00033   BLBS    STATUS, 4$                              ; 1149
                          000181A8    8F            53   D1 00036   CMPL    STATUS, #98728                          ; 1150
                                                    63   13 0003D   BEQL    4$                                      ;
                          000181B0    8F            53   D1 0003F   CMPL    STATUS, #98736                          ; 1151
                                                    05   12 00046   BNEQ    1$                                      ;
                                              1F    A2   95 00048   TSTB    31(R2)                                  ;
                                                    55   13 0004B   BEQL    4$                                      ;
                                      39      08    AC   E8 0004D 1$: BLBS    MODE, 3$                                ; 1154
                                      02      08    AC   D1 00051   CMPL    MODE, #2                                ; 1157
                                                    1E   12 00055   BNEQ    2$                                      ;
                          000181B0    8F            53   D1 00057   CMPL    STATUS, #98736                          ; 1158
                                                    15   12 0005E   BNEQ    2$                                      ;
                                              0C    A2   DD 00060   PUSHL   12(R2)                                  ; 1161
                                                    53   DD 00063   PUSHL   STATUS                                  ;
                                                    7E   D4 00065   CLRL    -(SP)                                   ; 1160
                                  00000000*         8F   DD 00067   PUSHL   #<LGI$_CMDINPUT&-8>                     ;
                          00000000G   00            04   FB 0006D   CALLS   #4, LIB$SIGNAL                         ;
                                                    04   00074      RET                                             ;
                                              0C    A2   DD 00075 2$: PUSHL   12(R2)                                ; 1164
```

```
                                        53  DD 00078          PUSHL   STATUS                                          ;
                                        7E  D4 0007A          CLRL    -(SP)                                         ; 1163
                          00000000G     8F  DD 0007C          PUSHL   #LGI$_CMDINPUT                                ;
                00000000G 00            04  FB 00082          CALLS   #4, LIB$STOP                                  ;
                                            04 00089          RET                                                   ; 1154
                                        01  DD 0008A 3$:      PUSHL   #1                                            ; 1168
                00000000G 00            01  FB 0008C          CALLS   #1, SET_TERMINAL_HANGUP                       ;
                                        7E  D4 00093          CLRL    -(SP)                                         ; 1169
                          00000000G     00  9F 00095          PUSHAB  EXIT_PROCESS                                  ;
                00000000G 00            02  FB 0009B          CALLS   #2, SYS$CMEXEC                                ;
                                            04 000A2 4$:      RET                                                   ; 1173

; Routine Size:  163 bytes,    Routine Base:  $CODE$ + 0405
```

```
;  785          1174  1 END
;  786          1175  0 ELUDOM



                                                        .EXTRN  LIB$SIGNAL, LIB$STOP

;                                  PSECT SUMMARY
;
;       Name                    Bytes                           Attributes
;
;  $OWN$                          255  NOVEC,  WRT,  RD ,NOEXE,NOSHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)
;  $GLOBAL$                       500  NOVEC,  WRT,  RD ,NOEXE,NOSHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)
;  $PLIT$                         164  NOVEC,NOWRT,  RD ,NOEXE,NOSHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)
;  $CODE$                        1192  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]LIB.L32;1          18619       123         0        1000        00:01.5



;                                       COMMAND QUALIFIERS

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

; Size:          1192 code + 919 data bytes
; Run Time:         00:20.9
; Elapsed Time:     01:25.3
; Lines/CPU Min:     3371
; Lexemes/CPU Min: 48393
; Memory Used:   190 pages
; Compilation Complete
```

PPDDEF
MDL

FILEIO
LIS

LOGIN

DETACHED
LIS

LOGINOUT
MAP

UTILDEF
REQ

LOGINCMD
CLD

HPWD
LIS

LGIDEF
MDL

AUDIT
LIS

BREAKIN
LIS