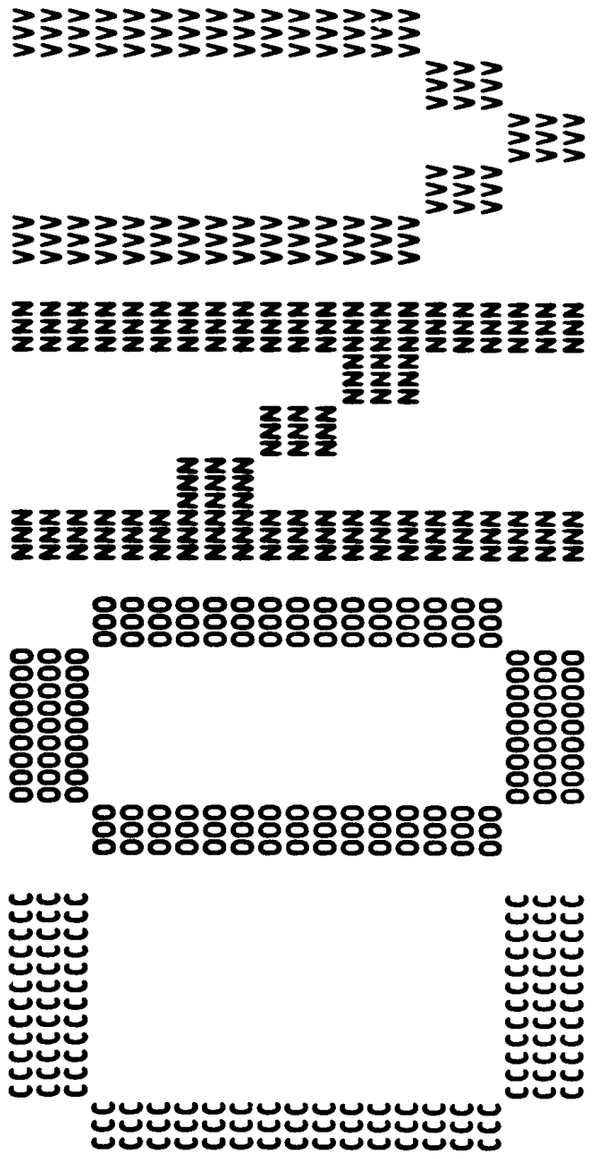

FILEID**CONVDCL

```
  CCCCCCCC    OOOOOO    NN      NN  VV      VV  DDDDDDDD      CCCCCCCC  LL
  CCCCCCCC    OOOOOO    NN      NN  VV      VV  DDDDDDDD      CCCCCCCC  LL
CC           OO      OO  NN      NN  VV      VV  DD      DD  CC        LL
CC           OO      OO  NN      NN  VV      VV  DD      DD  CC        LL
CC           OO      OO  NNNN    NN  VV      VV  DD      DD  CC        LL
CC           OO      OO  NNNN    NN  VV      VV  DD      DD  CC        LL
CC           OO      OO  NN  NN  NN  VV      VV  DD      DD  CC        LL
CC           OO      OO  NN  NN  NN  VV      VV  DD      DD  CC        LL
CC           OO      OO  NN    NNNN  VV      VV  DD      DD  CC        LL
CC           OO      OO  NN    NNNN  VV      VV  DD      DD  CC        LL
CC           OO      OO  NN      NN    VV  VV    DD      DD  CC        LL                  ....
CC           OO      OO  NN      NN    VV  VV    DD      DD  CC        LL                  ....
  CCCCCCCC    OOOOOO    NN      NN      VV      DDDDDDDD      CCCCCCCC  LLLLLLLLLL          ....
  CCCCCCCC    OOOOOO    NN      NN      VV      DDDDDDDD      CCCCCCCC  LLLLLLLLLL          ....


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 %TITLE  'VAX-11 CONVERT'
;     2          0002  0 MODULE  CONV$DCL        ( IDENT='V04-000',
;     3          0003  0                         MAIN=START
;     4          0004  0                         ) =
;     5          0005  0
;     6          0006  1 BEGIN
;     7          0007  1
;     8          0008  1 !*****************************************************************************
;     9          0009  1 !*                                                                           *
;    10          0010  1 !*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                                  *
;    11          0011  1 !*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.                    *
;    12          0012  1 !*  ALL RIGHTS RESERVED.                                                      *
;    13          0013  1 !*                                                                           *
;    14          0014  1 !*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED     *
;    15          0015  1 !*  ONLY IN  ACCORDANCE WITH  THE  TERMS  OF  SUCH  LICENSE  AND WITH THE    *
;    16          0016  1 !*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR  ANY  OTHER     *
;    17          0017  1 !*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY     *
;    18          0018  1 !*  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE IS  HEREBY     *
;    19          0019  1 !*  TRANSFERRED.                                                              *
;    20          0020  1 !*                                                                           *
;    21          0021  1 !*  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE WITHOUT NOTICE     *
;    22          0022  1 !*  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL EQUIPMENT     *
;    23          0023  1 !*  CORPORATION.                                                              *
;    24          0024  1 !*                                                                           *
;    25          0025  1 !*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR  RELIABILITY OF ITS     *
;    26          0026  1 !*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.                   *
;    27          0027  1 !*                                                                           *
;    28          0028  1 !*                                                                           *
;    29          0029  1 !*****************************************************************************
```

```
;    31          0030  1 !++
;    32          0031  1 !
;    33          0032  1 ! Facility:     VAX-11 CONVERT
;    34          0033  1 !
;    35          0034  1 ! Abstract:     DCL Utility which calls the CONVERT sharable image
;    36          0035  1 !
;    37          0036  1 ! Environment:
;    38          0037  1 !
;    39          0038  1 !               VAX/VMS Operating System
;    40          0039  1 !
;    41          0040  1 !--
;    42          0041  1 !
;    43          0042  1 !
;    44          0043  1 ! Author:       Keith B Thompson        Creation date:  July-1980
;    45          0044  1 !
;    46          0045  1 !
;    47          0046  1 ! Modified by:
;    48          0047  1 !
;    49          0048  1 !       V03-003 KBT0369         Keith B. Thompson       19-Oct-1982
;    50          0049  1 !               Remove all ref. to control flags and use the flags
;    51          0050  1 !               parameters on the conv$ calls
;    52          0051  1 !
;    53          0052  1 !       V03-002 KBT0036         Keith Thompson          31-Mar-1982
;    54          0053  1 !               Change the ref. to fdl$ab_ctrl through fdl$al_block and
;    55          0054  1 !               change the ref. to conv$ab_flags
;    56          0055  1 !
;    57          0056  1 !       V03-001 KBT0018         Keith Thompson          22-Mar-1982
;    58          0057  1 !               Fix the display of CPU time (Use quadword mult.)
;    59          0058  1 !
;    60          0059  1 !****
```

```
;    62          0060  1
;    63          0061  1 LIBRARY 'SYS$LIBRARY:LIB.L32';
;    64          0062  1 REQUIRE 'SRC$:CONVERT';
;    65          0297  1 REQUIRE 'SRC$:CONVDEF';
;    66          0458  1
;    67          0459  1 ! Structure for array of dynamic character string descriptors
;    68          0460  1 !
;    69          0461  1 STRUCTURE
;    70          0462  1         DYN_STR_DESC_VECTOR [ I, O, P, S, E; N ] =
;    71          0463  1                 [ N * DSC$K_D_BLN ]
;    72          0464  1                 ( ( DYN_STR_DESC_VECTOR + I * DSC$K_D_BLN ) + O )< P,S,E >;
;    73          0465  1
;    74          0466  1
;    75          0467  1 ! Fun with macros
;    76          0468  1 !
;    77          0469  1 MACRO
;    78          0470  1
;    79          0471  1     !  Define a macro to initialize an element of a dyn_str_desc_vector.
;    80          0472  1     ! This macro is passed the number of elements to initialize.  The macro
;    81          0473  1     ! makes no explicit assumption about the current descriptor format.
;    82          0474  1     !
;    83        M 0475  1     INIT_DSD_VECTOR ( I ) [] =  [ ( I ) - 1, DSC$B_DTYPE] = DSC$K_DTYPE_T,
;    84        M 0476  1                                 [ ( I ) - 1, DSC$B_CLASS] = DSC$K_CLASS_D
;    85        M 0477  1                                 %IF ( ( I ) - 1 GTR 0 )
;    86          0478  1                                 %THEN   , INIT_DSD_VECTOR ( ( I ) - 1 ) %FI %,
;    87          0479  1
;    88          0480  1     ! Define shorthand for a single initialized dynamic string desc
;    89          0481  1     !
;    90        M 0482  1     DYN_STR_DESC = BLOCK [ DSC$K_D_BLN,BYTE ]
;    91        M 0483  1                         PRESET( [ DSC$B_CLASS ] = DSC$K_CLASS_D,
;    92          0484  1                                 [ DSC$B_DTYPE ] = DSC$K_DTYPE_T ) %;
;    93          0485  1
;    94          0486  1 EXTERNAL ROUTINE
;    95          0487  1         CONV$PASS_FILES         : ADDRESSING_MODE( GENERAL ),
;    96          0488  1         CONV$PASS_OPTIONS       : ADDRESSING_MODE( GENERAL ),
;    97          0489  1         CONV$CONVERT            : ADDRESSING_MODE( GENERAL ),
;    98          0490  1         CLI$GET_VALUE           : ADDRESSING_MODE( GENERAL ),
;    99          0491  1         CLI$PRESENT             : ADDRESSING_MODE( GENERAL ),
;   100          0492  1         LIB$INIT_TIMER          : ADDRESSING_MODE( GENERAL ),
;   101          0493  1         LIB$STAT_TIMER          : ADDRESSING_MODE( GENERAL ),
;   102          0494  1         LIB$SUBX                : ADDRESSING_MODE( GENERAL ),
;   103          0495  1         LIB$PUT_OUTPUT          : ADDRESSING_MODE( GENERAL ),
;   104          0496  1         OTS$CVT_TI_L            : ADDRESSING_MODE( GENERAL ),
;   105          0497  1         OTS$CVT_TO_L            : ADDRESSING_MODE( GENERAL ),
;   106          0498  1         OTS$CVT_TZ_L            : ADDRESSING_MODE( GENERAL );
;   107          0499  1
;   108          0500  1 EXTERNAL LITERAL
;   109          0501  1         CONV$_FATALEXC,
;   110          0502  1         CONV$_ILL_KEY,
;   111          0503  1         CONV$_ILL_VALUE;
;   112          0504  1
;   113          0505  1 FORWARD ROUTINE
;   114          0506  1         MULQ            : NOVALUE;
;   115          0507  1
;   116          0508  1 LITERAL
;   117          0509  1         MAX_INFILES     = 10,           ! Max number of input files
;   118          0510  1         ASCII_D         = 68,           ! 'D'
```

```
;  119          0511  1          ASCII_0          = 79,          '0'
;  120          0512  1          ASCII_X          = 88,          'X'
;  121          0513  1          ASCII_PERCENT    = 37;          '%'
;  122          0514  1
;  123          0515  1  OWN
;  124          0516  1          IN_DESC          : DYN_STR_DESC_VECTOR [ MAX_INFILES ] ! Array of input filenames
;  125          0517  1                                PRESET ( INIT_DSD_VECTOR( MAX_INFILES ) ),
;  126          0518  1          OUT_DESC         : DESC_BLK                            ! Output file descriptor
;  127          0519  1                                PRESET( [ DSC$B_CLASS ] = DSC$K_CLASS_D ),
;  128          0520  1          FDL_DESC         : DESC_BLK                            ! Fdl file descriptor
;  129          0521  1                                PRESET( [ DSC$B_CLASS ] = DSC$K_CLASS_D ),
;  130          0522  1          EXC_DESC         : DESC_BLK                            ! Exception file descriptor
;  131          0523  1                                PRESET( [ DSC$B_CLASS ] = DSC$K_CLASS_D ),
;  132          0524  1
;  133          0525  1          ! FAO Processing
;  134          0526  1          !
;  135          0527  1          FAO_BUFFER       : VECTOR [ 132,BYTE ],                ! FAO Buffer
;  136          0528  1          FAO_DESC         : DESC_BLK                            ! FAO Descriptor Block
;  137          0529  1                                PRESET( [ DSC$B_CLASS ] = DSC$K_CLASS_D,
;  138          0530  1                                        [ DSC$W_LENGTH ] = 132,
;  139          0531  1                                        [ DSC$A_POINTER ] = FAO_BUFFER ),
;  140          0532  1          PUT_DESC         : DESC_BLK                            ! Put output Desc. Block
;  141          0533  1                                PRESET( [ DSC$B_CLASS ] = DSC$K_CLASS_D,
;  142          0534  1                                        [ DSC$W_LENGTH ] = 132,
;  143          0535  1                                        [ DSC$A_POINTER ] = FAO_BUFFER ),
;  144          0536  1
;  145          0537  1          ! Convert call argument Blocks
;  146          0538  1          !
;  147          0539  1          ! Option Block NOTE: The last option is optional and will be
;  148          0540  1          !                    determined at a latter date
;  149          0541  1          !
;  150          0542  1          OPTION_BLOCK     : VECTOR [ 20,LONG ] INITIAL( 18,REP 19 OF (0)),
;  151          0543  1
;  152          0544  1          ! Statistics Block
;  153          0545  1          !
;  154          0546  1          STATS_BLOCK      : VECTOR [ 5,LONG ] INITIAL( 4,0,0,0,0 ),
;  155          0547  1
;  156          0548  1          ! Flags longword
;  157          0549  1          !
;  158          0550  1          FLAGS            : LONG INITIAL( CONV$M_SIGNAL ),
;  159          0551  1
;  160          0552  1          TEMP_DESC        : DESC_BLK       ! Temporary work descriptor
;  161          0553  1                             PRESET( [DSC$B_CLASS ] = DSC$K_CLASS_D ),
;  162          0554  1
;  163          0555  1          TIMER_BLK,
;  164          0556  1
;  165          0557  1          ELP_TIME         : VECTOR [ 2,LONG ],
;  166          0558  1          CPU_TIME         : VECTOR [ 2,LONG ],
;  167          0559  1
;  168          0560  1          ELP_TIM_BUF      : VECTOR [ 16,BYTE ],
;  169          0561  1          CPU_TIM_BUF      : VECTOR [ 16,BYTE ],
;  170          0562  1
;  171          0563  1          ELP_DESC         : DESC_BLK       INITIAL ( 16,ELP_TIM_BUF ),
;  172          0564  1          CPU_DESC         : DESC_BLK       INITIAL ( 16,CPU_TIM_BUF ),
;  173          0565  1
;  174          0566  1          ONE              : INITIAL(1),
;  175          0567  1          TWO              : INITIAL(2);
```

```
;  176          0568  1          THREE           : INITIAL(3),
;  177          0569  1          FOUR            : INITIAL(4),
;  178          0570  1          FIVE            : INITIAL(5),
;  179          0571  1
;  180          0572  1          PROC_BLK        : VECTOR [ 5,LONG ];
;  181          0573  1
;  182          0574  1      BIND
;  183          0575  1          BUFF_IO         = PROC_BLK [ 2 ] : LONG,
;  184          0576  1          DIRE_IO         = PROC_BLK [ 3 ] : LONG,
;  185          0577  1          PG_FALT         = PROC_BLK [ 4 ] : LONG,
;  186          0578  1
;  187          0579  1          ! Opuput stats descriptors
;  188          0580  1          !
;  189          0581  1          STATS_DESC_BLOCK = UPLIT(
;  190          0582  1
;  191          0583  1          DESCRIPTOR( ' !/ CONVERT Statistics' ),
;  192          0584  1          DESCRIPTOR( 'Number of Files Processed:     !6UL' ),
;  193          0585  1          DESCRIPTOR( 'Total Records Processed:     !8UL!_Buffered I/O Count:    !8UL' ),
;  194          0586  1          DESCRIPTOR( 'Total Exception Records:     !8UL!_Direct I/O Count. . .!8UL' ),
;  195          0587  1          DESCRIPTOR( 'Total Valid Records:         !8UL!_Page Faults: !_._.8UL' ),
;  196          0588  1          DESCRIPTOR( 'Elapsed Time:       !AS!_CPU Time:       .AS' )
;  197          0589  1
;  198          0590  1          ) : VECTOR;
;  199          0591  1
```

```
;  201          0592  1 %SBTTL  'Main Routine'
;  202          0593  1 ROUTINE START    =
;  203          0594  1 !++
;  204          0595  1 !
;  205          0596  1 ! Functional Description:
;  206          0597  1 !
;  207          0598  1 !       Main convert processing routine.  This routine, called by DCL,
;  208          0599  1 !       in turn calls the convert sharable image.
;  209          0600  1 !
;  210          0601  1 ! Calling Sequence:
;  211          0602  1 !
;  212          0603  1 !       DCL Command
;  213          0604  1 !
;  214          0605  1 ! Input Parameters:
;  215          0606  1 !
;  216          0607  1 !       See DCL command syntax
;  217          0608  1 !
;  218          0609  1 ! Implicit Inputs:
;  219          0610  1 !       none
;  220          0611  1 !
;  221          0612  1 ! Output Parameters:
;  222          0613  1 !       none
;  223          0614  1 !
;  224          0615  1 ! Implicit Outputs:
;  225          0616  1 !       none
;  226          0617  1 !
;  227          0618  1 ! Routine Value:
;  228          0619  1 !
;  229          0620  1 !       SS$_NORMAL or error code
;  230          0621  1 !
;  231          0622  1 ! Routines Called:
;  232          0623  1 !
;  233          0624  1 !       CLI$GET_VALUE
;  234          0625  1 !       CLI$PRESENT
;  235          0626  1 !       OTS$CVT_TI_L
;  236          0627  1 !       OTS$CVT_TO_L
;  237          0628  1 !       OTS$CVT_TZ_L
;  238          0629  1 !       LIB$INIT_TIMER
;  239          0630  1 !       CONV$PASS_FILES
;  240          0631  1 !       CONV$PASS_OPTIONS
;  241          0632  1 !       CONV$CONVERT
;  242          0633  1 !
;  243          0634  1 ! Side Effects:
;  244          0635  1 !       ?
;  245          0636  1 !
;  246          0637  1 !--
;  247          0638  1
;  248          0639  2     BEGIN
;  249          0640  2
;  250          0641  2     BUILTIN
;  251          0642  2         SUBM;
;  252          0643  2
;  253          0644  2     LOCAL
;  254          0645  2         STATUS,
;  255          0646  2         STATISTICS,
;  256          0647  2         LENGTH;
;  257          0648  2
```

```
;  258          0649  2     ! Start the timer
;  259          0650  2     !
;  260          0651  2     LIB$INIT_TIMER( TIMER_BLK );
;  261          0652  2
;  262          0653  2     ! Get some needed values from the command line
;  263          0654  2     !
;  264          0655  2     OPTION_BLOCK [ 17 ] = CLI$GET_VALUE( DESCRIPTOR('FDL'),FDL_DESC );
;  265          0656  2
;  266          0657  2     ! Get the exception file name
;  267          0658  2     !
;  268          0659  2     OPTION_BLOCK [ 18 ] = CLI$GET_VALUE( DESCRIPTOR('EXCEPTION'),EXC_DESC );
;  269          0660  2
;  270          0661  2     ! Get the output file name
;  271          0662  2     !
;  272          0663  2     CLI$GET_VALUE( DESCRIPTOR('OUTFILE'),OUT_DESC );
;  273          0664  2
;  274          0665  2     ! Get the first input file name
;  275          0666  2     !
;  276          0667  2     CLI$GET_VALUE( DESCRIPTOR('INFILE'),IN_DESC [ 0,DSC$W_LENGTH ] );
;  277          0668  2
;  278          0669  2     ! Pass the files to convert
;  279          0670  2     !
;  280        P 0671  2     RET_ON_ERROR( CONV$PASS_FILES( IN_DESC [ 0,DSC$W_LENGTH ],  ! 1st input file
;  281        P 0672  2                                    OUT_DESC,                 ! Ouput file
;  282        P 0673  2                                    FDL_DESC,                 ! FDL file
;  283        P 0674  2                                    EXC_DESC,                 ! Exception file
;  284          0675  2                                    FLAGS ) );                ! Flags
;  285          0676  2
;  286          0677  2     ! Get the rest of the input file names if any
;  287          0678  2     !
;  288          0679  2     INCR I FROM 1 TO ( MAX_INFILES - 1 ) BY 1
;  289          0680  2     DO
;  290          0681  2
;  291          0682  2         ! If we got a file then pas it to convert
;  292          0683  2         !
;  293          0684  2         IF CLI$GET_VALUE( DESCRIPTOR('INFILE'),IN_DESC [ .I,DSC$W_LENGTH ] )
;  294          0685  2         THEN
;  295          0686  3             RET_ON_ERROR( CONV$PASS_FILES( IN_DESC [ .I,DSC$W_LENGTH ],FLAGS ) )
;  296          0687  2         ELSE
;  297          0688  2             EXITLOOP;
;  298          0689  2
;  299          0690  2     ! Get the command options
;  300          0691  2     !
;  301          0692  2     ! The statistics option is not passed to convert
;  302          0693  2     !
;  303          0694  2     STATISTICS  = CLI$PRESENT( DESCRIPTOR( 'STATISTICS' ) );
;  304          0695  2
;  305          0696  2     ! Set Option Flags
;  306          0697  2     !
;  307          0698  2     OPTION_BLOCK [ 1 ]  = CLI$PRESENT( DESCRIPTOR('CREATE') );
;  308          0699  2     OPTION_BLOCK [ 2 ]  = CLI$PRESENT( DESCRIPTOR('SHARE') );
;  309          0700  2     OPTION_BLOCK [ 3 ]  = CLI$PRESENT( DESCRIPTOR('FAST_LOAD') );
;  310          0701  2     OPTION_BLOCK [ 4 ]  = CLI$PRESENT( DESCRIPTOR('MERGE') );
;  311          0702  2     OPTION_BLOCK [ 5 ]  = CLI$PRESENT( DESCRIPTOR('APPEND') );
;  312          0703  2     OPTION_BLOCK [ 6 ]  = CLI$PRESENT( DESCRIPTOR('SORT') );
;  313          0704  2     OPTION_BLOCK [ 11 ] = CLI$PRESENT( DESCRIPTOR('TRUNCATE') );
;  314          0705  2     OPTION_BLOCK [ 12 ] = CLI$PRESENT( DESCRIPTOR('EXIT') );
```

```
;   315          0706  2      OPTION_BLOCK [ 13 ] = CLI$PRESENT( DESCRIPTOR('FIXED_CONTROL') );
;   316          0707  2      OPTION_BLOCK [ 14 ] = CLI$PRESENT( DESCRIPTOR('FILL_BUCKETS') );
;   317          0708  2      OPTION_BLOCK [ 15 ] = CLI$PRESENT( DESCRIPTOR('READ_CHECK') );
;   318          0709  2      OPTION_BLOCK [ 16 ] = CLI$PRESENT( DESCRIPTOR('WRITE_CHECK') );
;   319          0710  2
;   320          0711  2      ! Check the KEY qualifier
;   321          0712  2      !
;   322          0713  2      IF CLI$GET_VALUE( DESCRIPTOR('KEY'),TEMP_DESC )
;   323          0714  2      THEN
;   324          0715  3          BEGIN
;   325          0716  3
;   326          0717  3          LOCAL   IVALUE;
;   327          0718  3
;   328          0719  3          IF NOT OTS$CVT_TI_L( TEMP_DESC,IVALUE )
;   329          0720  3          THEN
;   330          0721  3              RETURN CONV$_ILL_KEY
;   331          0722  3          ELSE
;   332          0723  3              OPTION_BLOCK [ 8 ] = .IVALUE
;   333          0724  2          END;
;   334          0725  2
;   335          0726  2      ! Check the WORK_FILES qualifier
;   336          0727  2      !
;   337          0728  2      IF CLI$GET_VALUE( DESCRIPTOR('WORK_FILES'),TEMP_DESC )
;   338          0729  2      THEN
;   339          0730  3          BEGIN
;   340          0731  3
;   341          0732  3          LOCAL   IVALUE;
;   342          0733  3
;   343          0734  3          ! Convert the value parameter
;   344          0735  3          !
;   345          0736  3          IF NOT OTS$CVT_TI_L( TEMP_DESC,IVALUE )
;   346          0737  3          THEN
;   347          0738  3              RETURN CONV$_ILL_VALUE
;   348          0739  3          ELSE
;   349          0740  3              OPTION_BLOCK [ 7 ] = .IVALUE
;   350          0741  3          END
;   351          0742  2      ELSE
;   352          0743  2
;   353          0744  2          ! If not specified then the default work files is two (like SORT)
;   354          0745  2          !
;   355          0746  2          OPTION_BLOCK [ 7 ] = 2;
;   356          0747  2
;   357          0748  2      ! Check the PROLOGUE qualifier
;   358          0749  2      !
;   359          0750  2      IF CLI$GET_VALUE( DESCRIPTOR( 'PROLOGUE' ),TEMP_DESC )
;   360          0751  2      THEN
;   361          0752  3          BEGIN
;   362          0753  3
;   363          0754  3          LOCAL   IVALUE;
;   364          0755  3
;   365          0756  3          ! Convert the value parameter
;   366          0757  3          !
;   367          0758  3          IF NOT OTS$CVT_TI_L( TEMP_DESC,IVALUE )
;   368          0759  3          THEN
;   369          0760  3              RETURN CONV$_ILL_VALUE;
;   370          0761  3
;   371          0762  3          ! If everything is ok then stuff the value and make the option block
```

```
;  372          0763  3          ! longer
;  373          0764  3          !
;  374          0765  3          OPTION_BLOCK [ 0 ] = 19;
;  375          0766  3
;  376          0767  3          OPTION_BLOCK [ 19 ] = .IVALUE
;  377          0768  3
;  378          0769  2          END;
;  379          0770  2
;  380          0771  2      ! Check the PAD qualifier NOTE: do this last since it messes with temp_desc
;  381          0772  2      !
;  382          0773  2      IF OPTION_BLOCK [ 9 ] = CLI$GET_VALUE( DESCRIPTOR('PAD'),TEMP_DESC )
;  383          0774  2      THEN
;  384          0775  3          BEGIN
;  385          0776  3
;  386          0777  3          LOCAL   PAD_C : REF VECTOR [ ,BYTE ];
;  387          0778  3
;  388          0779  3          PAD_C = .TEMP_DESC [ DSC$A_POINTER ];
;  389          0780  3
;  390          0781  3          ! The syntax of the pad cheracter is:   a   - Ascii character except '%'
;  391          0782  3          !                                       %Dn - Decimal number
;  392          0783  3          !                                       %On - Octal number
;  393          0784  3          !                                       %Xn - Hex number
;  394          0785  3          !
;  395          0786  3          ! If the first character is a percent sign '%' then translate the
;  396          0787  3          ! numeric value depending on the base
;  397          0788  3          !
;  398          0789  3          IF .PAD_C [ 0 ] EQLU ASCII_PERCENT
;  399          0790  3          THEN
;  400          0791  4              BEGIN
;  401          0792  4
;  402          0793  4              LOCAL
;  403          0794  4                  STATUS,
;  404          0795  4                  IVALUE;
;  405          0796  4
;  406          0797  4              ! Strip off the '%c' from the descriptor
;  407          0798  4              !
;  408          0799  4              TEMP_DESC [ DSC$W_LENGTH ] = .TEMP_DESC [ DSC$W_LENGTH ] - 2;
;  409          0800  4              TEMP_DESC [ DSC$A_POINTER ] = .TEMP_DESC [ DSC$A_POINTER ] + 2;
;  410          0801  4
;  411          0802  4              ! Convert depending on the base
;  412          0803  4              !
;  413          0804  5              STATUS = ( SELECTONEU .PAD_C [ 1 ] OF
;  414          0805  5                       SET
;  415          0806  5                       [ ASCII_D ]   : OTS$CVT_TI_L( TEMP_DESC,IVALUE );
;  416          0807  5
;  417          0808  5                       [ ASCII_O ]   : OTS$CVT_TO_L( TEMP_DESC,IVALUE );
;  418          0809  5
;  419          0810  5                       [ ASCII_X ]   : OTS$CVT_TZ_L( TEMP_DESC,IVALUE );
;  420          0811  5
;  421          0812  5                       [ OTHERWISE ] : 0;
;  422          0813  4                       TES );
;  423          0814  4
;  424          0815  4              ! Check on any problem
;  425          0816  4              !
;  426          0817  4              IF NOT .STATUS
;  427          0818  4              THEN
;  428          0819  4                  RETURN CONV$_ILL_VALUE
```

```
;  429          0820  4              ELSE
;  430          0821  4                  OPTION_BLOCK [ 10 ] = .IVALUE
;  431          0822  4
;  432          0823  4              END
;  433          0824  3          ELSE
;  434          0825  4              BEGIN
;  435          0826  4
;  436          0827  4              ! This better be a single character
;  437          0828  4              !
;  438          0829  4              IF .TEMP_DESC [ DSC$W_LENGTH ] GTRU 1
;  439          0830  4              THEN
;  440          0831  4                  RETURN CONV$_ILL_VALUE;
;  441          0832  4
;  442          0833  4              OPTION_BLOCK [ 10 ] = .PAD_C [ 0 ]
;  443          0834  4
;  444          0835  4              END
;  445          0836  2          END;
;  446          0837  2
;  447          0838  2      ! Initalize CONVERT
;  448          0839  2      !
;  449          0840  2      RET_ON_ERROR( CONV$PASS_OPTIONS ( OPTION_BLOCK,FLAGS ) );
;  450          0841  2
;  451          0842  2      ! Do the conversion
;  452          0843  2      !
;  453          0844  3      IF NOT ( STATUS = CONV$CONVERT ( STATS_BLOCK,FLAGS ) )
;  454          0845  2      THEN
;  455          0846  2
;  456          0847  2          ! If there was an error and it wasn't conv$_fatalexc then exit
;  457          0848  2          !
;  458          0849  2          IF .STATUS NEQU CONV$_FATALEXC
;  459          0850  2          THEN
;  460          0851  2              RETURN .STATUS;
;  461          0852  2
;  462          0853  2      ! If we want and sucess then output some stats
;  463          0854  2      !
;  464          0855  2      IF .STATISTICS
;  465          0856  2      THEN
;  466          0857  3          BEGIN
;  467          0858  3
;  468          0859  3          OWN
;  469          0860  3              ZERO_Q      : VECTOR [ 2,LONG ] INITIAL( 0,0 ),     ! Used for
;  470          0861  3              TEMP_TIME   : VECTOR [ 2,LONG ],                    ! conversion
;  471          0862  3              MUL100K     : VECTOR [ 2,LONG ] INITIAL( 100000,0 );! of times
;  472          0863  3
;  473          0864  3          ! Get Preformance Stats
;  474          0865  3          !
;  475          0866  3          LIB$STAT_TIMER( ONE,    ELP_TIME,       TIMER_BLK );
;  476          0867  3          LIB$STAT_TIMER( TWO,    TEMP_TIME,      TIMER_BLK );
;  477          0868  3          LIB$STAT_TIMER( THREE,  BUFF_IO,        TIMER_BLK );
;  478          0869  3          LIB$STAT_TIMER( FOUR,   DIRE_IO,        TIMER_BLK );
;  479          0870  3          LIB$STAT_TIMER( FIVE,   PG_FALT,        TIMER_BLK );
;  480          0871  3
;  481          0872  3          ! Convert to delta time
;  482          0873  3          !
;  483          0874  3          SUBM( 2,ELP_TIME,ZERO_Q,ELP_TIME );
;  484          0875  3
;  485          0876  3          ! Convert internal times to ASCII
```

```
;  486          0877  3          !
;  487        P 0878  3          $ASCTIM( TIMLEN = 0,
;  488        P 0879  3                   TIMBUF = ELP_DESC,
;  489        P 0880  3                   TIMADR = ELP_TIME,
;  490          0881  3                   CVTFLG = 0 );
;  491          0882  3
;  492          0883  3          ! The CPU time is given in 10msec ticks so we need to convert it to
;  493          0884  3          ! system delta time
;  494          0885  3          !
;  495          0886  3          ! Convert to 10nsec ticks
;  496          0887  3          !
;  497          0888  3          MULQ( TEMP_TIME,MUL100K,CPU_TIME );
;  498          0889  3
;  499          0890  3          ! Convert to delta time
;  500          0891  3          !
;  501          0892  3          SUBM( 2,CPU_TIME,ZERO_Q,CPU_TIME );
;  502          0893  3
;  503          0894  3          ! Conver to ascii
;  504          0895  3          !
;  505        P 0896  3          $ASCTIM( TIMLEN = 0,
;  506        P 0897  3                   TIMBUF = CPU_DESC,
;  507        P 0898  3                   TIMADR = CPU_TIME,
;  508          0899  3                   CVTFLG = 0 );
;  509          0900  3
;  510          0901  3          INCR I FROM 0 TO 4 BY 1
;  511          0902  3          DO
;  512          0903  4              BEGIN
;  513          0904  4
;  514        P 0905  4              $FAO( .STATS_DESC_BLOCK [ .I ],
;  515        P 0906  4                    LENGTH,
;  516        P 0907  4                    FAO_DESC,
;  517        P 0908  4                    .STATS_BLOCK [ .I ],
;  518          0909  4                    .PROC_BLK [ .I ] );
;  519          0910  4
;  520          0911  4              PUT_DESC [ DSC$W_LENGTH ] = .LENGTH;
;  521          0912  4
;  522          0913  4              LIB$PUT_OUTPUT( PUT_DESC )
;  523          0914  4
;  524          0915  3              END;
;  525          0916  3
;  526          0917  3          ! Elasped Time and CPU Time
;  527          0918  3          !
;  528        P 0919  3          $FAO( .STATS_DESC_BLOCK [ 5 ],
;  529        P 0920  3                LENGTH,
;  530        P 0921  3                FAO_DESC,
;  531        P 0922  3                ELP_DESC,
;  532          0923  3                CPU_DESC );
;  533          0924  3
;  534          0925  3          PUT_DESC [ DSC$W_LENGTH ] = .LENGTH;
;  535          0926  3
;  536          0927  3          LIB$PUT_OUTPUT( PUT_DESC )
;  537          0928  3
;  538          0929  2          END;
;  539          0930  2
;  540          0931  2      RETURN .STATUS
;  541          0932  2
;  542          0933  1      END;
```

```
                                                                                .TITLE  CONV$DCL VAX-11 CONVERT
                                                                                .IDENT  \V04-000\

                                                                                .PSECT  $PLIT$,NOWRT,NOEXE,2

61 74 53 20 54 52 45 56 4E 4F 43 20 2F 21 20  00000 P.AAC:   .ASCII  \ !/ CONVERT Statistics\
                        73 63 69 74 73 69 74  0000F
                                              00016          .BLKB   2
                                    00000016  00018 P.AAB:   .LONG   22
                                    00000000' 0001C          .ADDRESS P.AAC
73 65 6C 69 46 20 66 6F 20 72 65 62 6D 75 4E  00020 P.AAE:   .ASCII  \Number of Files Processed:    !6UL\
20 20 20 20 3A 64 65 73 73 65 63 6F 72 50 20  0002F
                                 4C 55 36 21  0003E
                                              00042          .BLKB   2
                                    00000022  00044 P.AAD:   .LONG   34
                                    00000000' 00048          .ADDRESS P.AAE
50 20 73 64 72 6F 63 65 52 20 6C 61 74 6F 54  0004C P.AAG:   .ASCII  \Total Records Processed:    !8UL!_Buffer\
38 21 20 20 20 20 3A 64 65 73 73 65 63 6F 72  0005B
                  72 65 66 66 75 42 5F 21 4C  0006A
21 20 3A 74 6E 75 6F 43 20 4F 2F 49 20 64 65  00074          .ASCII  \ed I/O Count: !_!8UL\
                           4C 55 38 21 5F      00083
                                    0000003C  00088 P.AAF:   .LONG   60
                                    00000000' 0008C          .ADDRESS P.AAG
6E 6F 69 74 70 65 63 78 45 20 6C 61 74 6F 54  00090 P.AAI:   .ASCII  \Total Exception Records:    !8UL!_Direct\
38 21 20 20 20 20 3A 73 64 72 6F 63 65 52 20  0009F
                  74 63 65 72 69 44 5F 21 4C  000AE
21 5F 21 20 3A 74 6E 75 6F 43 20 4F 2F 49 20  000B8          .ASCII  \ I/O Count: !_!8UL\
                                 4C 55 38     000C7
                                              000CA          .BLKB   2
                                    0000003A  000CC P.AAH:   .LONG   58
                                    00000000' 000D0          .ADDRESS P.AAI
63 65 52 20 64 69 6C 61 56 20 6C 61 74 6F 54  000D4 P.AAK:   .ASCII  \Total Valid Records:        !8UL!_Page F\
38 21 20 20 20 20 20 20 20 20 3A 73 64 72 6F  000E3
                  46 20 65 67 61 50 5F 21 4C  000F2
4C 55 38 21 5F 21 5F 21 20 3A 73 74 6C 75 61  000FC          .ASCII  \aults: !_!_!8UL\
                                              0010B          .BLKB   1
                                    00000037  0010C P.AAJ:   .LONG   55
                                    00000000' 00110          .ADDRESS P.AAK
20 20 3A 65 6D 69 54 20 64 65 73 70 61 6C 45  00114 P.AAM:   .ASCII  \Elapsed Time:       !AS!_CPU Time:      \
54 20 55 50 43 5F 21 53 41 21 20 20 20 20 20  00123
                  20 20 20 20 20 20 3A 65 6D  00132
                                 53 41 21 20  0013C          .ASCII  \ !AS\
                                    0000002C  00140 P.AAL:   .LONG   44
                                    00000000' 00144          .ADDRESS P.AAM
00000000' 00000000' 00000000' 00000000' 00000000' 00000000' 00148 P.AAA:   .ADDRESS P.AAB, P.AAD, P.AAF, P.AAH, P.AAJ, P.AAL
                                       4C 44 46  00160 P.AAO:   .ASCII  \FDL\
                                              00163          .BLKB   1
                                    00000003  00164 P.AAN:   .LONG   3
                                    00000000' 00168          .ADDRESS P.AAO
                        4E 4F 49 54 50 45 43 58 45  0016C P.AAQ:   .ASCII  \EXCEPTION\
                                              00175          .BLKB   3
                                    00000009  00178 P.AAP:   .LONG   9
                                    00000000' 0017C          .ADDRESS P.AAQ
                           45 4C 49 46 54 55 4F  00180 P.AAS:   .ASCII  \OUTFILE\
                                              00187          .BLKB   1
```

```
                                      00000007  00188 P.AAR:  .LONG    7
                                      00000000' 0018C         .ADDRESS P.AAS
                          45 4C 49 46 4E 49     00190 P.AAU:  .ASCII   \INFILE\
                                                00196         .BLKB    2
                                      00000006  00198 P.AAT:  .LONG    6
                                      00000000' 0019C         .ADDRESS P.AAU
                          45 4C 49 46 4E 49     001A0 P.AAW:  .ASCII   \INFILE\
                                                001A6         .BLKB    2
                                      00000006  001A8 P.AAV:  .LONG    6
                                      00000000' 001AC         .ADDRESS P.AAW
              53 43 49 54 53 49 54 41 54 53     001B0 P.AAY:  .ASCII   \STATISTICS\
                                                001BA         .BLKB    2
                                      0000000A  001BC P.AAX:  .LONG    10
                                      00000000' 001C0         .ADDRESS P.AAY
                          45 54 41 45 52 43     001C4 P.ABA:  .ASCII   \CREATE\
                                                001CA         .BLKB    2
                                      00000006  001CC P.AAZ:  .LONG    6
                                      00000000' 001D0         .ADDRESS P.ABA
                             45 52 41 48 53     001D4 P.ABC:  .ASCII   \SHARE\
                                                001D9         .BLKB    3
                                      00000005  001DC P.ABB:  .LONG    5
                                      00000000' 001E0         .ADDRESS P.ABC
                    44 41 4F 4C 5F 54 53 41 46  001E4 P.ABE:  .ASCII   \FAST_LOAD\
                                                001ED         .BLKB    3
                                      00000009  001F0 P.ABD:  .LONG    9
                                      00000000' 001F4         .ADDRESS P.ABE
                             45 47 52 45 4D     001F8 P.ABG:  .ASCII   \MERGE\
                                                001FD         .BLKB    3
                                      00000005  00200 P.ABF:  .LONG    5
                                      00000000' 00204         .ADDRESS P.ABG
                          44 4E 45 50 50 41     00208 P.ABI:  .ASCII   \APPEND\
                                                0020E         .BLKB    2
                                      00000006  00210 P.ABH:  .LONG    6
                                      00000000' 00214         .ADDRESS P.ABI
                                54 52 4F 53     00218 P.ABK:  .ASCII   \SORT\
                                      00000004  0021C P.ABJ:  .LONG    4
                                      00000000' 00220         .ADDRESS P.ABK
                    45 54 41 43 4E 55 52 54     00224 P.ABM:  .ASCII   \TRUNCATE\
                                      00000008  0022C P.ABL:  .LONG    8
                                      00000000' 00230         .ADDRESS P.ABM
                                54 49 58 45     00234 P.ABO:  .ASCII   \EXIT\
                                      00000004  00238 P.ABN:  .LONG    4
                                      00000000' 0023C         .ADDRESS P.ABO
  4C 4F 52 54 4E 4F 43 5F 44 45 58 49 46        00240 P.ABQ:  .ASCII   \FIXED_CONTROL\
                                                0024D         .BLKB    3
                                      0000000D  00250 P.ABP:  .LONG    13
                                      00000000' 00254         .ADDRESS P.ABQ
     53 54 45 4B 43 55 42 5F 4C 4C 49 46        00258 P.ABS:  .ASCII   \FILL_BUCKETS\
                                      0000000C  00264 P.ABR:  .LONG    12
                                      00000000' 00268         .ADDRESS P.ABS
              4B 43 45 48 43 5F 44 41 45 52     0026C P.ABU:  .ASCII   \READ_CHECK\
                                                00276         .BLKB    2
                                      0000000A  00278 P.ABT:  .LONG    10
                                      00000000' 0027C         .ADDRESS P.ABU
           4B 43 45 48 43 5F 45 54 49 52 57     00280 P.ABW:  .ASCII   \WRITE_CHECK\
                                                0028B         .BLKB    1
                                      0000000B  0028C P.ABV:  .LONG    11
```

```
                00000000'  00290          .ADDRESS P.ABW                          ;
             59  45  4B    00294 P.ABY:   .ASCII  \KEY\                           ;
                           00297          .BLKB   1
                00000003   00298 P.ABX:   .LONG   3                               ;
                00000000'  0029C          .ADDRESS P.ABY                          ;
     53  45  4C  49  46  5F  4B  52  4F  57  002A0 P.ACA:   .ASCII  \WORK_FILES\    ;
                           002AA          .BLKB   2
                0000000A   002AC P.ABZ:   .LONG   10                              ;
                00000000'  002B0          .ADDRESS P.ACA                          ;
         45  55  47  4F  4C  4F  52  50  002B4 P.ACC:   .ASCII  \PROLOGUE\        ;
                00000008   002BC P.ACB:   .LONG   8                               ;
                00000000'  002C0          .ADDRESS P.ACC                          ;
             44  41  50    002C4 P.ACE:   .ASCII  \PAD\                           ;
                           002C7          .BLKB   1
                00000003   002C8 P.ACD:   .LONG   3                               ;
                00000000'  002CC          .ADDRESS P.ACE                          ;

                                          .PSECT  $OWN$,NOEXE,2

                   00#  00000 IN_DESC:.BYTE   0[2]                                ;
                02 0E   00002          .BYTE   14, 2                              ;
                   00#  00004          .BYTE   0[6]                               ;
                02 0E   0000A          .BYTE   14, 2                              ;
                   00#  0000C          .BYTE   0[6]                               ;
                02 0E   00012          .BYTE   14, 2                              ;
                   00#  00014          .BYTE   0[6]                               ;
                02 0E   0001A          .BYTE   14, 2                              ;
                   00#  0001C          .BYTE   0[6]                               ;
                02 0E   00022          .BYTE   14, 2                              ;
                   00#  00024          .BYTE   0[6]                               ;
                02 0E   0002A          .BYTE   14, 2                              ;
                   00#  0002C          .BYTE   0[6]                               ;
                02 0E   00032          .BYTE   14, 2                              ;
                   00#  00034          .BYTE   0[6]                               ;
                02 0E   0003A          .BYTE   14, 2                              ;
                   00#  0003C          .BYTE   0[6]                               ;
                02 0E   00042          .BYTE   14, 2                              ;
                   00#  00044          .BYTE   0[6]                               ;
                02 0E   0004A          .BYTE   14, 2                              ;
                        0004C          .BLKB   4
                   00#  00050 OUT_DESC:
                                       .BYTE   0[3]                               ;
                   02   00053          .BYTE   2                                  ;
                        00054          .BLKB   4
                   00#  00058 FDL_DESC:
                                       .BYTE   0[3]                               ;
                   02   0005B          .BYTE   2                                  ;
                        0005C          .BLKB   4
                   00#  00060 EXC_DESC:
                                       .BYTE   0[3]                               ;
                   02   00063          .BYTE   2                                  ;
                        00064          .BLKB   4
                        00068 FAO_BUFFER:
                                       .BLKB   132
                0084    000EC FAO_DESC:
                                       .WORD   132                                ;
                  00    000EE          .BYTE   0                                  ;
```

```
                                                           02  000EF           .BYTE    2                            ;
                                                    00000000'  000F0           .ADDRESS FAO_BUFFER                   ;
                                                         0084  000F4 PUT_DESC:
                                                                               .WORD    132                          ;
                                                           00  000F6           .BYTE    0                            ;
                                                           02  000F7           .BYTE    2                            ;
                                                    00000000'  000F8           .ADDRESS FAO_BUFFER                   ;
                                                     00000012  000FC OPTION_BLOCK:
                                                                               .LONG    18                           ;
                                                    00000000#  00100           .LONG    0[19]                        ;
     00000000  00000000  00000000  00000000  00000004  0014C STATS_BLOCK:
                                                                               .LONG    4, 0, 0, 0, 0                ;
                                                     00000001  00160 FLAGS:    .LONG    1                            ;
                                                          00#  00164 TEMP_DESC:
                                                                               .BYTE    0[3]                         ;
                                                           02  00167           .BYTE    2                            ;
                                                               00168           .BLKB    4
                                                               0016C TIMER_BLK:
                                                                               .BLKB    4
                                                               00170 ELP_TIME:
                                                                               .BLKB    8
                                                               00178 CPU_TIME:
                                                                               .BLKB    8
                                                               00180 ELP_TIM_BUF:
                                                                               .BLKB    16
                                                               00190 CPU_TIM_BUF:
                                                                               .BLKB    16
                                                     00000010  001A0 ELP_DESC:
                                                                               .LONG    16                           ;
                                                    00000000'  001A4           .ADDRESS ELP_TIM_BUF                  ;
                                                     00000010  001A8 CPU_DESC:
                                                                               .LONG    16                           ;
                                                    00000000'  001AC           .ADDRESS CPU_TIM_BUF                  ;
                                                     00000001  001B0 ONE:      .LONG    1                            ;
                                                     00000002  001B4 TWO:      .LONG    2                            ;
                                                     00000003  001B8 THREE:    .LONG    3                            ;
                                                     00000004  001BC FOUR:     .LONG    4                            ;
                                                     00000005  001C0 FIVE:     .LONG    5                            ;
                                                               001C4 PROC_BLK:
                                                                               .BLKB    20
                                           00000000  00000000  001D8 ZERO_Q:   .LONG    0, 0                         ;
                                                               001E0 TEMP_TIME:
                                                                               .BLKB    8
                                           00000000  000186A0  001E8 MUL100K:  .LONG    100000, 0                    ;

                                                                     BUFF_IO=           PROC_BLK+8
                                                                     DIRE_IO=           PROC_BLK+12
                                                                     PG_FALT=           PROC_BLK+16
                                                                     STATS_DESC_BLOCK=  P.AAA
                                                                               .EXTRN   CONV$PASS_FILES
                                                                               .EXTRN   CONV$PASS_OPTIONS
                                                                               .EXTRN   CONV$CONVERT, CLI$GET_VALUE
                                                                               .EXTRN   CLI$PRESENT, LIB$INIT_TIMER
                                                                               .EXTRN   LIB$STAT_TIMER, LIB$SUBX
                                                                               .EXTRN   LIB$PUT_OUTPUT, OTS$CVT_TI_L
                                                                               .EXTRN   OTS$CVT_TO_L, OTS$CVT_TZ_L
                                                                               .EXTRN   CONV$_FATALEXC, CONV$_ILL_KEY
```

```
                                                        .EXTRN  CONV$_ILL_VALUE
                                                        .EXTRN  SYS$ASCTIM, SYS$FAO

                                                        .PSECT  $CODE$,NOWRT,2

                                        OFFC 00000 START:   .WORD   Save R2,R3,R4,R5,R6,R7,R8,R9,R10,R11          ; 0593
                                5B 00000000G   00   9E 00002        MOVAB   SYS$ASCTIM, R11
                                5A 00000000G   00   9E 00009        MOVAB   CONV$PASS_FILES, R10
                                59 00000000G   00   9E 00010        MOVAB   OTS$CVT_TI_L, R9
                                58 00000000G   00   9E 00017        MOVAB   LIB$STAT_TIMER, R8
                                57 00000000G   00   9E 0001E        MOVAB   CLI$GET_VALUE, R7
                                56      0000'  CF   9E 00025        MOVAB   P.AAN, R6
                                55 00000000G   00   9E 0002A        MOVAB   CLI$PRESENT, R5
                                54      0000'  CF   9E 00031        MOVAB   TEMP_DESC, R4
                                5E                 14   C2 00036    SUBL2   #20, SP
                                          08       A4   9F 00039    PUSHAB  TIMER_BLK                           ; 0651
                  00000000G     00                 01   FB 0003C    CALLS   #1, LIB$INIT_TIMER
                                        FEF4       C4   9F 00043    PUSHAB  FDL_DESC                            ; 0655
                                                   56   DD 00047    PUSHL   R6
                                67                 02   FB 00049    CALLS   #2, CLI$GET_VALUE
                             DC A4                 50   D0 0004C    MOVL    R0, OPTION_BLOCK+68
                                        FEFC       C4   9F 00050    PUSHAB  EXC_DESC                            ; 0659
                                          14       A6   9F 00054    PUSHAB  P.AAP
                                67                 02   FB 00057    CALLS   #2, CLI$GET_VALUE
                             E0 A4                 50   D0 0005A    MOVL    R0, OPTION_BLOCK+72
                                        FEEC       C4   9F 0005E    PUSHAB  OUT_DESC                            ; 0663
                                          24       A6   9F 00062    PUSHAB  P.AAR
                                67                 02   FB 00065    CALLS   #2, CLI$GET_VALUE
                                        FE9C       C4   9F 00068    PUSHAB  IN_DESC                             ; 0667
                                          34       A6   9F 0006C    PUSHAB  P.AAT
                                67                 02   FB 0006F    CALLS   #2, CLI$GET_VALUE
                                          FC       A4   9F 00072    PUSHAB  FLAGS                               ; 0675
                                        FEFC       C4   9F 00075    PUSHAB  EXC_DESC
                                        FEF4       C4   9F 00079    PUSHAB  FDL_DESC
                                        FEEC       C4   9F 0007D    PUSHAB  OUT_DESC
                                        FE9C       C4   9F 00081    PUSHAB  IN_DESC
                                6A                 05   FB 00085    CALLS   #5, CONV$PASS_FILES
                                1C                 50   E9 00088    BLBC    STATUS, 2$
                                52                 01   D0 0008B    MOVL    #1, I                              ; 0679
                                        FE9C C442  7F   0008E 1$:    PUSHAQ  IN_DESC[I]                         ; 0684
                                          44       A6   9F 00093    PUSHAB  P.AAV
                                67                 02   FB 00096    CALLS   #2, CLI$GET_VALUE
                                13                 50   E9 00099    BLBC    R0, 4$
                                          FC       A4   9F 0009C    PUSHAB  FLAGS                              ; 0686
                                        FE9C C442  7F   0009F    PUSHAQ  IN_DESC[I]
                                6A                 02   FB 000A4    CALLS   #2, CONV$PASS_FILES
                                01                 50   E8 000A7 2$:    BLBS    STATUS, 3$
                                                   04   000AA        RET
                          DF    52                 09   F3 000AB 3$:    AOBLEQ  #9, I, 1$                      ; 0684
                                          58       A6   9F 000AF 4$:    PUSHAB  P.AAX                          ; 0694
                                65                 01   FB 000B2    CALLS   #1, CLI$PRESENT
                                52                 50   D0 000B5    MOVL    R0, STATISTICS
                                          68       A6   9F 000B8    PUSHAB  P.AAZ                              ; 0698
                                65                 01   FB 000BB    CALLS   #1, CLI$PRESENT
                             9C A4                 50   D0 000BE    MOVL    R0, OPTION_BLOCK+4
                                          78       A6   9F 000C2    PUSHAB  P.ABB                              ; 0699
                                65                 01   FB 000C5    CALLS   #1, CLI$PRESENT
```

```
                A0    A4          50  D0 000C8          MOVL    R0, OPTION_BLOCK+8
                                008C  C6  9F 000CC          PUSHAB  P.ABD                   ; 0700
                      65            01  FB 000D0          CALLS   #1, CLI$PRESENT
                A4    A4          50  D0 000D3          MOVL    R0, OPTION_BLOCK+12
                                009C  C6  9F 000D7          PUSHAB  P.ABF                   ; 0701
                      65            01  FB 000DB          CALLS   #1, CLI$PRESENT
                A8    A4          50  D0 000DE          MOVL    R0, OPTION_BLOCK+16
                                00AC  C6  9F 000E2          PUSHAB  P.ABH                   ; 0702
                      65            01  FB 000E6          CALLS   #1, CLI$PRESENT
                AC    A4          50  D0 000E9          MOVL    R0, OPTION_BLOCK+20
                                00B8  C6  9F 000ED          PUSHAB  P.ABJ                   ; 0703
                      65            01  FB 000F1          CALLS   #1, CLI$PRESENT
                B0    A4          50  D0 000F4          MOVL    R0, OPTION_BLOCK+24
                                00C8  C6  9F 000F8          PUSHAB  P.ABL                   ; 0704
                      65            01  FB 000FC          CALLS   #1, CLI$PRESENT
                C4    A4          50  D0 000FF          MOVL    R0, OPTION_BLOCK+44
                                00D4  C6  9F 00103          PUSHAB  P.ABN                   ; 0705
                      65            01  FB 00107          CALLS   #1, CLI$PRESENT
                C8    A4          50  D0 0010A          MOVL    R0, OPTION_BLOCK+48
                                00EC  C6  9F 0010E          PUSHAB  P.ABP                   ; 0706
                      65            01  FB 00112          CALLS   #1, CLI$PRESENT
                CC    A4          50  D0 00115          MOVL    R0, OPTION_BLOCK+52
                                0100  C6  9F 00119          PUSHAB  P.ABR                   ; 0707
                      65            01  FB 0011D          CALLS   #1, CLI$PRESENT
                D0    A4          50  D0 00120          MOVL    R0, OPTION_BLOCK+56
                                0114  C6  9F 00124          PUSHAB  P.ABT                   ; 0708
                      65            01  FB 00128          CALLS   #1, CLI$PRESENT
                D4    A4          50  D0 0012B          MOVL    R0, OPTION_BLOCK+60
                                0128  C6  9F 0012F          PUSHAB  P.ABV                   ; 0709
                      65            01  FB 00133          CALLS   #1, CLI$PRESENT
                D8    A4          50  D0 00136          MOVL    R0, OPTION_BLOCK+64
                                                54  DD 0013A          PUSHL   R4              ; 0713
                                0134  C6  9F 0013C          PUSHAB  P.ABX
                      67            02  FB 00140          CALLS   #2, CLI$GET_VALUE
                      16            50  E9 00143          BLBC    R0, 6$
                                4010  8F  BB 00146          PUSHR   #^M<R4,SP>              ; 0719
                      69            02  FB 0014A          CALLS   #2, OTS$CVT_TI_L
                      08            50  E8 0014D          BLBS    R0, 5$
                      50 00000000G  8F  D0 00150          MOVL    #CONV$_ILL_KEY, R0      ; 0721
                                                04  00157          RET
                B8    A4          6E  D0 00158  5$:       MOVL    IVALUE, OPTION_BLOCK+32 ; 0723
                                                54  DD 0015C  6$:       PUSHL   R4          ; 0728
                                0148  C6  9F 0015E          PUSHAB  P.ABZ
                      67            02  FB 00162          CALLS   #2, CLI$GET_VALUE
                      12            50  E9 00165          BLBC    R0, 7$
                                  04  AE  9F 00168          PUSHAB  IVALUE                  ; 0736
                                                54  DD 0016B          PUSHL   R4
                      69            02  FB 0016D          CALLS   #2, OTS$CVT_TI_L
                      1F            50  E9 00170          BLBC    R0, 9$
                B4    A4          04  AE  D0 00173          MOVL    IVALUE, OPTION_BLOCK+28 ; 0740
                                                04  11 00178          BRB     8$          ; 0736
                B4    A4          02  D0 0017A  7$:       MOVL    #2, OPTION_BLOCK+28     ; 0746
                                                54  DD 0017E  8$:       PUSHL   R4          ; 0750
                                0158  C6  9F 00180          PUSHAB  P.ACB
                      67            02  FB 00184          CALLS   #2, CLI$GET_VALUE
                      14            50  E9 00187          BLBC    R0, 10$
                                  08  AE  9F 0018A          PUSHAB  IVALUE                  ; 0758
```

```
                                54 DD 0018D          PUSHL   R4                                                      ;
                             69 02 FB 0018F          CALLS   #2, OTS$CVT_TI_L                                        ;
                             76 50 E9 00192 9$:      BLBC    R0, 16$                                                 ;
                          98 A4 13 D0 00195          MOVL    #19, OPTION_BLOCK                                       ; 0765
                          E4 A4 08 AE D0 00199       MOVL    IVALUE, OPTION_BLOCK+76                                 ; 0767
                                54 DD 0019E 10$:     PUSHL   R4                                                      ; 0773
                           0164 C6 9F 001A0          PUSHAB  P.ACD                                                   ;
                             67 02 FB 001A4          CALLS   #2, CLI$GET_VALUE                                       ;
                          BC A4 50 D0 001A7          MOVL    R0, OPTION_BLOCK+36                                     ;
                             69 50 E9 001AB          BLBC    R0, 18$                                                 ;
                             50 04 A4 D0 001AE       MOVL    TEMP_DESC+4, PAD_C                                      ; 0779
                             25 60 91 001B2          CMPB    (PAD_C), #37                                            ; 0789
                                4F 12 001B5          BNEQ    15$                                                     ;
                             64 02 A2 001B7          SUBW2   #2, TEMP_DESC                                           ; 0799
                          04 A4 02 C0 001BA          ADDL2   #2, TEMP_DESC+4                                         ; 0800
                             50 01 A0 9A 001BE       MOVZBL  1(PAD_C), R0                                            ; 0804
                          44 8F 50 91 001C2          CMPB    R0, #68                                                 ; 0806
                                0A 12 001C6          BNEQ    11$                                                     ;
                             0C AE 9F 001C8          PUSHAB  IVALUE                                                  ;
                                54 DD 001CB          PUSHL   R4                                                      ;
                             69 02 FB 001CD          CALLS   #2, OTS$CVT_TI_L                                        ;
                                2A 11 001D0          BRB     14$                                                     ;
                          4F 8F 50 91 001D2 11$:     CMPB    R0, #79                                                 ; 0808
                                0E 12 001D6          BNEQ    12$                                                     ;
                             0C AE 9F 001D8          PUSHAB  IVALUE                                                  ;
                                54 DD 001DB          PUSHL   R4                                                      ;
                 00000000G   00 02 FB 001DD          CALLS   #2, OTS$CVT_TO_L                                        ;
                                16 11 001E4          BRB     14$                                                     ;
                          58 8F 50 91 001E6 12$:     CMPB    R0, #88                                                 ; 0810
                                0E 12 001EA          BNEQ    13$                                                     ;
                             0C AE 9F 001EC          PUSHAB  IVALUE                                                  ;
                                54 DD 001EF          PUSHL   R4                                                      ;
                 00000000G   00 02 FB 001F1          CALLS   #2, OTS$CVT_TZ_L                                        ;
                                02 11 001F8          BRB     14$                                                     ;
                             50 D4 001FA 13$:        CLRL    STATUS                                                  ; 0812
                          0C 50 E9 001FC 14$:        BLBC    STATUS, 16$                                             ; 0817
                       C0 A4 0C AE D0 001FF          MOVL    IVALUE, OPTION_BLOCK+40                                 ; 0821
                             11 11 00204             BRB     18$                                                     ; 0817
                          01 64 B1 00206 15$:        CMPW    TEMP_DESC, #1                                           ; 0829
                             08 1B 00209             BLEQU   17$                                                     ;
                          50 00000000G 8F D0 0020B 16$:  MOVL    #CONV$_ILL_VALUE, R0                                ; 0831
                                04 00212             RET                                                             ;
                       C0 A4 60 9A 00213 17$:        MOVZBL  (PAD_C), OPTION_BLOCK+40                                ; 0833
                          FC A4 9F 00217 18$:        PUSHAB  FLAGS                                                   ; 0840
                          98 A4 9F 0021A             PUSHAB  OPTION_BLOCK                                            ;
                 00000000G   00 02 FB 0021D          CALLS   #2, CONV$PASS_OPTIONS                                   ;
                             01 50 E8 00224          BLBS    STATUS, 19$                                             ;
                                04 00227             RET                                                             ;
                          FC A4 9F 00228 19$:        PUSHAB  FLAGS                                                   ; 0844
                          E8 A4 9F 0022B             PUSHAB  STATS_BLOCK                                             ;
                 00000000G   00 02 FB 0022E          CALLS   #2, CONV$CONVERT                                        ;
                             53 50 D0 00235          MOVL    R0, STATUS                                              ;
                             0C 53 E8 00238          BLBS    STATUS, 21$                                             ;
                 00000000G   8F 53 D1 0023B          CMPL    STATUS, #CONV$_FATALEXC                                 ; 0849
                                03 13 00242          BEQL    21$                                                     ;
                              00E1 31 00244 20$:     BRW     23$                                                     ;
                             FA 52 E9 00247 21$:     BLBC    STATISTICS, 20$                                         ; 0855
```

```
                                    08   A4   9F 0024A          PUSHAB  TIMER_BLK                                   ; 0866
                                    0C   A4   9F 0024D          PUSHAB  ELP_TIME                                    ;
                                    4C   A4   9F 00250          PUSHAB  ONE                                         ;
                              68          03   FB 00253          CALLS   #3, LIB$STAT_TIMER                          ;
                                    08   A4   9F 00256          PUSHAB  TIMER_BLK                                   ; 0867
                                    7C   A4   9F 00259          PUSHAB  TEMP_TIME                                   ;
                                    50   A4   9F 0025C          PUSHAB  TWO                                         ;
                              68          03   FB 0025F          CALLS   #3, LIB$STAT_TIMER                          ;
                                    08   A4   9F 00262          PUSHAB  TIMER_BLK                                   ; 0868
                                    68   A4   9F 00265          PUSHAB  BUFF_IO                                     ;
                                    54   A4   9F 00268          PUSHAB  THREE                                       ;
                              68          03   FB 0026B          CALLS   #3, LIB$STAT_TIMER                          ;
                                    08   A4   9F 0026E          PUSHAB  TIMER_BLK                                   ; 0869
                                    6C   A4   9F 00271          PUSHAB  DIRE_IO                                     ;
                                    58   A4   9F 00274          PUSHAB  FOUR                                        ;
                              68          03   FB 00277          CALLS   #3, LIB$STAT_TIMER                          ;
                                    08   A4   9F 0027A          PUSHAB  TIMER_BLK                                   ; 0870
                                    70   A4   9F 0027D          PUSHAB  PG_FACT                                     ;
                                    5C   A4   9F 00280          PUSHAB  FIVE                                        ;
                              68          03   FB 00283          CALLS   #3, LIB$STAT_TIMER                          ;
          0C   A4          74   A4   0C   A4   C3 00286          SUBL3   ELP_TIME, ZERO_Q, ELP_TIME                  ; 0874
                              50   78   A4   D0 0028D          MOVL    ZERO_Q+4, R0                                ;
                              50   10   A4   D9 00291          SBWC    ELP_TIME+4, R0                              ;
                         10   A4         50   D0 00295          MOVL    R0, ELP_TIME+4                              ;
                                         7E   D4 00299          CLRL    -(SP)                                       ; 0881
                                    0C   A4   9F 0029B          PUSHAB  ELP_TIME                                    ;
                                    3C   A4   9F 0029E          PUSHAB  ELP_DESC                                    ;
                                         7E   D4 002A1          CLRL    -(SP)                                       ;
                              6B          04   FB 002A3          CALLS   #4, SYS$ASCTIM                              ;
                                    14   A4   9F 002A6          PUSHAB  CPU_TIME                                    ; 0888
                                  0084   C4   9F 002A9          PUSHAB  MULTOOK                                     ;
                                    7C   A4   9F 002AD          PUSHAB  TEMP_TIME                                   ;
                      0000V   CF          03   FB 002B0          CALLS   #3, MULQ                                    ;
          14   A4          74   A4   14   A4   C3 002B5          SUBL3   CPU_TIME, ZERO_Q, CPU_TIME                  ; 0892
                              50   78   A4   D0 002BC          MOVL    ZERO_Q+4, R0                                ;
                              50   18   A4   D9 002C0          SBWC    CPU_TIME+4, R0                              ;
                         18   A4         50   D0 002C4          MOVL    R0, CPU_TIME+4                              ;
                                         7E   D4 002C8          CLRL    -(SP)                                       ; 0899
                                    14   A4   9F 002CA          PUSHAB  CPU_TIME                                    ;
                                    44   A4   9F 002CD          PUSHAB  CPU_DESC                                    ;
                                         7E   D4 002D0          CLRL    -(SP)                                       ;
                              6B          04   FB 002D2          CALLS   #4, SYS$ASCTIM                              ;
                                         52   D4 002D5          CLRL    I                                           ; 0901
                                  60 A442   DD 002D7 22$:       PUSHL   PROC_BLK[I]                                 ; 0909
                                  E8 A442   DD 002DB          PUSHL   STATS_BLOCK[I]                              ;
                                    88   A4   9F 002DF          PUSHAB  FAO_DESC                                    ;
                                    1C   AE   9F 002E2          PUSHAB  LENGTH                                      ;
                                  E4 A642   DD 002E5          PUSHL   STATS_DESC_BLOCK[I]                         ;
                  00000000G   00          05   FB 002E9          CALLS   #5, SYS$FAO                                 ;
                         90   A4   10   AE   B0 002F0          MOVW    LENGTH, PUT_DESC                            ; 0911
                                    90   A4   9F 002F5          PUSHAB  PUT_DESC                                    ; 0913
                  00000000G   00          01   FB 002F8          CALLS   #1, LIB$PUT_OUTPUT                          ;
                    D4        52          04   F3 002FF          AOBLEQ  #4, I, 22$                                  ;
                                    44   A4   9F 00303          PUSHAB  CPU_DESC                                    ; 0923
                                    3C   A4   9F 00306          PUSHAB  ELP_DESC                                    ;
                                    88   A4   9F 00309          PUSHAB  FAO_DESC                                    ;
                                    1C   AE   9F 0030C          PUSHAB  LENGTH                                      ;
```

```
                                 F8    A6  DD 0030F          PUSHL   STATS_DESC_BLOCK+20                                    ;
                     00000000G   00        05  FB 00312     CALLS   #5, SYS$FAO                                           ;
                           90    A4   10   AE  B0 00319     MOVW    LENGTH, PUT_DESC                                      ; 0925
                                      90   A4  9F 0031E     PUSHAB  PUT_DESC                                              ; 0927
                     00000000G   00        01  FB 00321     CALLS   #1, LIB$PUT_OUTPUT                                    ;
                                 50        53  D0 00328 23$: MOVL    STATUS, R0                                            ; 0931
                                               04 0032B     RET                                                           ; 0933

; Routine Size:  812 bytes,    Routine Base:  $CODE$ + 0000
```

```
;  544          0934  1 %SBTTL  'MULQ'
;  545          0935  1 ROUTINE MULQ ( MUL1 : REF VECTOR [ 2,LONG ],
;  546          0936  1                MUL2 : REF VECTOR [ 2,LONG ],
;  547          0937  1                PROD : REF VECTOR [ 2,LONG ] ) : NOVALUE =
;  548          0938  1
;  549          0939  1 !++
;  550          0940  1 !
;  551          0941  1 ! Functional Description:
;  552          0942  1 !
;  553          0943  1 !       Multiplies two quadwords.  This routine was converted from the examp.e
;  554          0944  1 !       of the EMUL instruction in the VAX Architecture Handbook
;  555          0945  1 !
;  556          0946  1 ! Calling Sequence:
;  557          0947  1 !
;  558          0948  1 !       MULQ( mul1,mul2,prod )
;  559          0949  1 !
;  560          0950  1 ! Input Parameters:
;  561          0951  1 !
;  562          0952  1 !       mul1    - quadword multiplier
;  563          0953  1 !       mul2    - quadword multiplier
;  564          0954  1 !
;  565          0955  1 ! Implicit Inputs:
;  566          0956  1 !       none
;  567          0957  1 !
;  568          0958  1 ! Output Parameters:
;  569          0959  1 !
;  570          0960  1 !       prod    - quadword product (note: output cannot be same as either input)
;  571          0961  1 !
;  572          0962  1 ! Implicit Outputs:
;  573          0963  1 !       none
;  574          0964  1 !
;  575          0965  1 ! Routine Value:
;  576          0966  1 !       none
;  577          0967  1 !
;  578          0968  1 ! Routines Called:
;  579          0969  1 !       none
;  580          0970  1 !
;  581          0971  1 ! Side Effects:
;  582          0972  1 !       none
;  583          0973  1 !
;  584          0974  1 !--
;  585          0975  1
;  586          0976  2     BEGIN
;  587          0977  2
;  588          0978  2     BUILTIN
;  589          0979  2         EMUL;
;  590          0980  2
;  591          0981  2     BIND
;  592          0982  2         MUL1S = MUL1 [ 0 ] : SIGNED,
;  593          0983  2         MUL2S = MUL2 [ 0 ] : SIGNED;
;  594          0984  2
;  595          0985  2     LOCAL
;  596          0986  2         ZERO : INITIAL( 0 ),
;  597          0987  2         TEMP;
;  598          0988  2
;  599          0989  2     ! Multiply low half
;  600          0990  2     !
```

```
;  601      0991  2        EMUL( .MUL1,.MUL2,ZERO,.PROD );
;  602      0992  2
;  603      0993  2        ! High half = A[high] * B[low] + A[low] * B[high]
;  604      0994  2        !
;  605      0995  2        TEMP = ( .MUL1 [ 1 ] * .MUL2 [ 0 ] ) + ( .MUL1 [ 0 ] * .MUL2 [ 1 ] );
;  606      0996  2
;  607      0997  2        ! If A[low]<0 then compensate of unsigned bias of 2**32
;  608      0998  2        !
;  609      0999  2        IF .MUL1S LSS 0
;  610      1000  2        THEN
;  611      1001  2            TEMP = .TEMP + .MUL2 [ 0 ];
;  612      1002  2
;  613      1003  2        ! If B[low]<0 then compensate of unsigned bias of 2**32
;  614      1004  2        !
;  615      1005  2        IF .MUL2S LSS 0
;  616      1006  2        THEN
;  617      1007  2            TEMP = .TEMP + .MUL1 [ 0 ];
;  618      1008  2
;  619      1009  2        ! Combine with high half of A[low] * B[low]
;  620      1010  2        !
;  621      1011  2        PROD [ 1 ] = .PROD [ 1 ] + .TEMP;
;  622      1012  2
;  623      1013  2        RETURN
;  624      1014  2
;  625      1015  1        END;


                                          001C 00000 MULQ:   .WORD   Save R2,R3,R4                 ; 0935
                          53      04   AC   D0 00002         MOVL    MUL1, R3                      ; 0982
                          52      08   AC   D0 00006         MOVL    MUL2, R2                      ; 0983
                                       51   D4 0000A         CLRL    ZERO                          ;
                          50      0C   AC   D0 0000C         MOVL    PROD, R0                      ; 0991
     60         51                62   63   7A 00010         EMUL    (R3), (R2), ZERO, (R0)        ;
                54      04        A3   62   C5 00015         MULL3   (R2), 4(R3), R4               ; 0995
                51                63   04   A2 C5 0001A      MULL3   4(R2), (R3), R1               ;
                          51           54   C0 0001F         ADDL2   R4, TEMP                      ;
                                       63   D5 00022         TSTL    (R3)                          ; 0999
                                       03   18 00024         BGEQ    1$                            ;
                          51           62   C0 00026         ADDL2   (R2), TEMP                    ; 1001
                                       62   D5 00029 1$:     TSTL    (R2)                          ; 1005
                                       03   18 0002B         BGEQ    2$                            ;
                          51           63   C0 0002D         ADDL2   (R3), TEMP                    ; 1007
                        04  A0         51   C0 00030 2$:     ADDL2   TEMP, 4(R0)                   ; 1011
                                            04 00034         RET                                   ; 1015

; Routine Size:  53 bytes,    Routine Base:  $CODE$ + 032C


;  626      1016  1
;  627      1017  0 END     ELUDOM
```

```
;                                   PSECT SUMMARY
;
;       Name                    Bytes                           Attributes
;
;  $OWN$                         496  NOVEC,  WRT,  RD ,NOEXE,NOSHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)
;  $PLIT$                        720  NOVEC,NOWRT,  RD ,NOEXE,NOSHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)
;  $CODE$                        865  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        11         0      1000          00:01.8



;                                   COMMAND QUALIFIERS

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

; Size:          865 code + 1216 data bytes
; Run Time:          00:22.4
; Elapsed Time:      01:18.0
; Lines/CPU Min:      2722
; Lexemes/CPU-Min: 24929
; Memory Used:  268 pages
; Compilation Complete
```

CONVDATA
LIS

CONVFSTLD
LIS

CONVFSTIO
LIS

CONVFSTRC
LIS

CONVFILES
LIS

CONVERROR
LIS

CONVFASTM
LIS

CONVDCL
LIS