```
   CCCCCCCCCCCC     OOOOOOOOO     NNN         NNN  VVV         VVV
   CCCCCCCCCCCC     OOOOOOOOO     NNN         NNN  VVV         VVV
   CCCCCCCCCCCC     OOOOOOOOO     NNN         NNN  VVV         VVV
CCC              OOO         OOO  NNN         NNN  VVV         VVV
CCC              OOO         OOO  NNN         NNN  VVV         VVV
CCC              OOO         OOO  NNN         NNN  VVV         VVV
CCC              OOO         OOO  NNNNNN      NNN  VVV         VVV
CCC              OOO         OOO  NNNNNN      NNN  VVV         VVV
CCC              OOO         OOO  NNNNNN      NNN  VVV         VVV
CCC              OOO         OOO  NNN   NNN   NNN  VVV         VVV
CCC              OOO         OOO  NNN   NNN   NNN  VVV         VVV
CCC              OOO         OOO  NNN   NNN   NNN  VVV         VVV
CCC              OOO         OOO  NNN      NNNNNN  VVV         VVV
CCC              OOO         OOO  NNN      NNNNNN  VVV         VVV
CCC              OOO         OOO  NNN      NNNNNN  VVV         VVV
CCC              OOO         OOO  NNN         NNN     VVV   VVV
CCC              OOO         OOO  NNN         NNN     VVV   VVV
CCC              OOO         OOO  NNN         NNN     VVV   VVV
   CCCCCCCCCCCC     OOOOOOOOO     NNN         NNN        VVV
   CCCCCCCCCCCC     OOOOOOOOO     NNN         NNN        VVV
   CCCCCCCCCCCC     OOOOOOOOO     NNN         NNN        VVV
```

```
**FILE**ID**CONVCOMIO                                                  D 16
```

```
  CCCCCCCC    OOOOOO    NN      NN  VV      VV    CCCCCCCC    OOOOOO    MM      MM    IIIIII      OOOOOO
  CCCCCCCC    OOOOOO    NN      NN  VV      VV    CCCCCCCC    OOOOOO    MM      MM    IIIIII      OOOOOO
CC          OO      OO  NN      NN  VV      VV  CC          OO      OO  MMMM  MMMM      II      OO      OO
CC          OO      OO  NN      NN  VV      VV  CC          OO      OO  MMMM  MMMM      II      OO      OO
CC          OO      OO  NNNN    NN  VV      VV  CC          OO      OO  MM  MM  MM      II      OO      OO
CC          OO      OO  NNNN    NN  VV      VV  CC          OO      OO  MM  MM  MM      II      OO      OO
CC          OO      OO  NN  NN  NN  VV      VV  CC          OO      OO  MM      MM      II      OO      OO
CC          OO      OO  NN  NN  NN  VV      VV  CC          OO      OO  MM      MM      II      OO      OO
CC          OO      OO  NN    NNNN  VV      VV  CC          OO      OO  MM      MM      II      OO      OO
CC          OO      OO  NN    NNNN  VV      VV  CC          OO      OO  MM      MM      II      OO      OO
CC          OO      OO  NN      NN    VV  VV    CC          OO      OO  MM      MM      II      OO      OO
CC          OO      OO  NN      NN    VV  VV    CC          OO      OO  MM      MM      II      OO      OO      ....
  CCCCCCCC    OOOOOO    NN      NN      VV        CCCCCCCC    OOOOOO    MM      MM    IIIIII      OOOOOO        ....
  CCCCCCCC    OOOOOO    NN      NN      VV        CCCCCCCC    OOOOOO    MM      MM    IIIIII      OOOOOO        ....

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
```

```
0001 0 %TITLE  'VAX-11 CONVERT'
0002 0 MODULE  CONV$COMIO       ( IDENT='V04-000',
0003 0                          OPTLEVEL=3
0004 0                          ) =
0005 0
0006 1 BEGIN
0007 1
0008 1 !****************************************************************************
0009 1 !*                                                                          *
0010 1 !*   COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                                *
0011 1 !*   DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.                 *
0012 1 !*   ALL RIGHTS RESERVED.                                                   *
0013 1 !*                                                                          *
0014 1 !*   THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED  *
0015 1 !*   ONLY IN  ACCORDANCE WITH  THE  TERMS  OF  SUCH  LICENSE  AND WITH THE  *
0016 1 !*   INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR  ANY  OTHER  *
0017 1 !*   COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY  *
0018 1 !*   OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE IS  HEREBY  *
0019 1 !*   TRANSFERRED.                                                           *
0020 1 !*                                                                          *
0021 1 !*   THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE WITHOUT NOTICE  *
0022 1 !*   AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL EQUIPMENT  *
0023 1 !*   CORPORATION.                                                           *
0024 1 !*                                                                          *
0025 1 !*   DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR  RELIABILITY OF ITS  *
0026 1 !*   SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.                *
0027 1 !*                                                                          *
0028 1 !*                                                                          *
0029 1 !****************************************************************************
```

```
1 !++
1 !
1 ! Facility:     VAX-11 CONVERT
1 !
1 ! Abstract:     Common Convert utilities I/O routines
1 !
1 ! Contents:
1 !               READ_PROLOGUE
1 !               WRITE_PROLOGUE
1 !               SET_KEY_DESC
1 !               GET_NEXT_KEY
1 !               WRITE_KEY_DESC
1 !               WRITE_AREA_DESC
1 !
1 ! Environment:
1 !
1 !               VAX/VMS Operating System
1 !
1 !--
1
1 !
1 ! Author:       Keith B Thompson        Creation date:          March-1982
1 !
1 !
1 ! Modified by:
1 !
1 !         V03-002 KBT0479         Keith B. Thompson       29-Jan-1983
1 !                 Make key_desc_buf and key_desc_vbn global
1 !
1 !         V03-001 KBT0391         Keith B. Thompson       28-Oct-1982
1 !                 Make things work
1 !
1 !****
```

```
1
1 PSECT
1         OWN     = _CONV$OWN      (PIC),
1         GLOBAL  = _CONV$GLOBAL   (PIC),
1         PLIT    = _CONV$PLIT     (SHARE,PIC),
1         CODE    = _CONV$CODE     (SHARE,PIC);
1
1 LIBRARY 'SYS$LIBRARY:LIB.L32';
1 LIBRARY 'SRC$:CONVERT';
1
1 DEFINE_ERROR_CODES;
1
1 LINKAGE
1         CL$READ_BLOCK    = JSB ( REGISTER = 2, REGISTER = 3 ),
1         CL$WRITE_BLOCK   = JSB ( REGISTER = 2, REGISTER = 3 ),
1         CL$CHECKSUM      = JSB ( REGISTER = 2 );
1
1 EXTERNAL ROUTINE
1         CONV$$GET_VM               : CL$GET_VM,
1         CONV$$RMS_ERROR            : NOVALUE,
1         CONV$$RMS_OPEN_ERROR       : NOVALUE,
1         CONV$$RMS_READ_ERROR       : NOVALUE;
1
1 FORWARD ROUTINE
1         CONV$$WRITE_AREA_DESC      : CL$WRITE_AREA_DESC     NOVALUE,
1         CONV$$GET_NEXT_KEY         : CL$GET_NEXT_KEY,
1         READ_BLOCK                 : CL$READ_BLOCK          NOVALUE,
1         WRITE_BLOCK                : CL$WRITE_BLOCK         NOVALUE,
1         CHECKSUM                   : CL$CHECKSUM;
1
1 EXTERNAL
1         CONV$AB_FLAGS              : BLOCK [ ,BYTE ],
1         CONV$AB_OUT_FAB            : $FAB_DECL,
1         CONV$AB_OUT_RAB            : $RAB_DECL,
1         CONV$AB_OUT_NAM            : $NAM_DECL,
1         CONV$AB_OUT_XABSUM         : $XABSUM_DECL,
1         CONV$GB_PROL_V1            : BYTE,
1         CONV$GB_PROL_V2            : BYTE,
1         CONV$GB_PROL_V3            : BYTE,
1         CONV$AR_PROLOGUE           : REF BLOCK [ ,BYTE ],
1         CONV$AR_AREA_BLOCK         : REF BLOCKVECTOR [ ,AREA$C_BLN,BYTE ];
1
1 GLOBAL
1         CONV$GL_KEY_DESC_BUF,
1         CONV$GL_KEY_DESC_VBN;
1
1 OWN
1         AREA_BLOCKS;
1
```

```
1 %SBTTL  'READ_PROLOGUE'
1 GLOBAL ROUTINE  CONV$$READ_PROLOGUE : CL$READ_PROLOGUE NOVALUE =
1 !++
1 !
1 ! Functional Description:
1 !
1 !       Reads the prologue blocks of the output file.  The first block (VBN=1)
1 !       is in the buffer pointed to by conv$ar_prologue. The area descriptors
1 !       are read into the buffer pointed to by conv$ar_area_block.  If there
1 !       are more then one key descriptor an extra block is allocated and it
1 !       is pointed to by key_desc_buf.
1 !
1 ! Calling Sequence:
1 !
1 !       conv$$read_prologue()
1 !
1 ! Input Parameters:
1 !       none
1 !
1 ! Implicit Inputs:
1 !       none
1 !
1 ! Output Parameters:
1 !       none
1 !
1 ! Implicit Outputs:
1 !       none
1 !
1 ! Routine Value:
1 !       none
1 !
1 ! Routines Called:
1 !
1 !       CONV$$GET_VM
1 !
1 ! Side Effects:
1 !       none
1 !
1 !--
1
2     BEGIN
2
2     LOCAL
2       TOTAL_BLOCKS;
2
2     ! The buffer is allocated thus:
2     !
2     !                                  ------------------------
2     !   conv$ar_prologue :            !                        !
2     !                                 !       512 Bytes        !
2     !                                 !                        !
2     !                                 !------------------------!
2     !   conv$gl_key_desc_buf :        !                        !
2     !                                 !       512 Bytes        !
2     !                                 !                        !
2     !                                 !------------------------!
2     !   conv$ar_area_block :          !                        !
```

```
2     !                                  |       512*No. of       |
2     !                                  |-      Area Blocks     -|
2     !                                  |            .           |
2     !                                  |            .           |
2     !                                  --------------------------
2     !
2     !
2     ! Figure out the number of blocks for the prologue area desc.
2     !
2     AREA_BLOCKS = ( ( .CONV$AB_OUT_XABSUM [ XAB$B_NOA ] - 1 ) / 8 ) + 1;
2
2     ! The total blocks is area blocks + prologue block + key desc buffer
2     !
2     TOTAL_BLOCKS = .AREA_BLOCKS + 1 + 1;
2
2     ! Get the address space.
2     !
2     CONV$AR_PROLOGUE = CONV$$GET_VM( .TOTAL_BLOCKS * BLOCK_SIZE );
2
2     ! The key block points just after the prologue block
2     !
2     CONV$GL_KEY_DESC_BUF = .CONV$AR_PROLOGUE + BLOCK_SIZE;
2
2     ! The area descriptors is after everything
2     !
2     CONV$AR_AREA_BLOCK = .CONV$GL_KEY_DESC_BUF + BLOCK_SIZE;
2
2     ! Read in the prologue block
2     !
2     READ_BLOCK( .CONV$AR_PROLOGUE,1 );
2
2     ! Read each of the area blocks
2     !
2     INCR I FROM 0 TO .AREA_BLOCKS - 1
2     DO
2         READ_BLOCK( .CONV$AR_AREA_BLOCK + ( .I  * BLOCK_SIZE ),
2                         .CONV$AR_PROLOGUE [ PLG$B_AVBN ] + .I );
2
2     ! Set the proper prologue version flag
2     !
2     SELECTONE .CONV$AR_PROLOGUE [ PLG$W_VER_NO ] OF
2     SET
2         [ PLG$C_VER_NO ]          : CONV$GB_PROL_V1 = _SET;
2         [ PLG$C_VER_IDX ]         : CONV$GB_PROL_V2 = _SET;
2         [ PLG$C_VER_3 ]           : CONV$GB_PROL_V3 = _SET;
2         [ OTHERWISE ]             : SIGNAL_STOP( CONV$_PLV );
2     TES;
2
2     RETURN
2
1     END;

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

                                                  .PSECT  _CONV$GLOBAL,NOEXE,  PIC,2
```

```
                                                 00000 CONV$GL_KEY_DESC_BUF::
                                                               .BLKB    4
                                                 00004 CONV$GL_KEY_DESC_VBN::
                                                               .BLKB    4

                                                               .PSECT   _CONV$OWN,NOEXE,  PIC,2

                                                 00000 AREA_BLOCKS:
                                                               .BLKB    4

                                                               .EXTRN   CONVERT$_FACILITY
                                                               .EXTRN   CONV$_FAO_MAX, CONV$_BADBLK
                                                               .EXTRN   CONV$_BADLOGIC, CONV$_BADSORT
                                                               .EXTRN   CONV$_CONFQUAL, CONV$_CREATEDSTM
                                                               .EXTRN   CONV$_CREA_ERR, CONV$_DELPRI
                                                               .EXTRN   CONV$_DUP, CONV$_EXTN_ERR
                                                               .EXTRN   CONV$_FATALEXC, CONV$_FILLIM
                                                               .EXTRN   CONV$_IDX_LIM, CONV$_ILL_KEY
                                                               .EXTRN   CONV$_ILL_VALUE
                                                               .EXTRN   CONV$_INP_FILES
                                                               .EXTRN   CONV$_INSVIRMEM
                                                               .EXTRN   CONV$_INVBKT, CONV$_KEY
                                                               .EXTRN   CONV$_KEYREF, CONV$_LOADIDX
                                                               .EXTRN   CONV$_NARG, CONV$_NI
                                                               .EXTRN   CONV$_NOKEY, CONV$_NOTIDX
                                                               .EXTRN   CONV$_NOTSEQ, CONV$_NOWILD
                                                               .EXTRN   CONV$_ORDER, CONV$_OPENEXC
                                                               .EXTRN   CONV$_OPENIN, CONV$_OPENOUT
                                                               .EXTRN   CONV$_PAD, CONV$_PLV
                                                               .EXTRN   CONV$_PROERR, CONV$_PROL_WRT
                                                               .EXTRN   CONV$_READERR, CONV$_RSK
                                                               .EXTRN   CONV$_RSZ, CONV$_RTL
                                                               .EXTRN   CONV$_RTS, CONV$_SEQ
                                                               .EXTRN   CONV$_UDF_BKS, CONV$_UDF_BLK
                                                               .EXTRN   CONV$_VFC, CONV$_WRITEERR
                                                               .EXTRN   CONV$$GET_VM, CONV$$RMS_ERROR
                                                               .EXTRN   CONV$$RMS_OPEN_ERROR
                                                               .EXTRN   CONV$$RMS_READ_ERROR
                                                               .EXTRN   CONV$AB_FLAGS, CONV$AB_OUT_FAB
                                                               .EXTRN   CONV$AB_OUT_RAB
                                                               .EXTRN   CONV$AB_OUT_NAM
                                                               .EXTRN   CONV$AB_OUT_XABSUM
                                                               .EXTRN   CONV$GB_PROL_V1
                                                               .EXTRN   CONV$GB_PROL_V2
                                                               .EXTRN   CONV$GB_PROL_V3
                                                               .EXTRN   CONV$AR_PROLOGUE
                                                               .EXTRN   CONV$AR_AREA_BLOCK

                                                               .PSECT   _CONV$CODE,NOWRT,  SHR,  PIC,2

                                       3C  BB 00000 CONV$$READ_PROLOGUE::
                                                               PUSHR    #^M<R2,R3,R4,R5>                              ; 0113
                   50       0000G  CF  9A 00002               MOVZBL   CONV$AB_OUT_XABSUM+8, R0                      ; 0178
                                   50  D7 00007               DECL     R0                                            ;
                   50              08  C6 00009               DIVL2    #8, R0                                        ;
             0000'  CF        01   A0  9E 0000C               MOVAB    1(R0), AREA_BLOCKS                            ;
```

```
                     50       0000'   CF                 02   C1 00012          ADDL3    #2, AREA_BLOCKS, TOTAL_BLOCKS            ; 0182
                     7E               50                 09   78 00018          ASHL     #9, TOTAL_BLOCKS, -(SP)                  ; 0186
                                                      0000G   30 0001C          BSBW     CONV$$GET_VM                             ;
                                      5E                 04   C0 0001F          ADDL2    #4, SP                                   ;
                              0000G   CF                 50   D0 00022          MOVL     R0, CONV$AR_PROLOGUE                     ;
           0000'   CF         0000G   CF 00000200        8F   C1 00027          ADDL3    #512, CONV$AR_PROLOGUE, -                ; 0190
                                                                                         CONV$GL_KEY_DESC_BUF                     ;
           0000G   CF         0000'   CF 00000200        8F   C1 00033          ADDL3    #512, CONV$GL_KEY_DESC_BUF, -            ; 0194
                                                                                         CONV$AR_AREA_BLOCK                       ;
                                      53                 01   D0 0003F          MOVL     #1, R3                                   ; 0198
                                      52       0000G     CF   D0 00042          MOVL     CONV$AR_PROLOGUE, R2                     ;
                                                      0000V   30 00047          BSBW     READ_BLOCK                               ;
                                      55       0000'     CF   D0 0004A          MOVL     AREA_BLOCKS, R5                          ; 0202
                                      54                 01   CE 0004F          MNEGL    #1, I                                    ;
                                                         1D   11 00052          BRB      2$                                       ;
                                      50       0000G     CF   D0 00054 1$:      MOVL     CONV$AR_PROLOGUE, R0                     ; 0205
                                      50          66     A0   9E 00059          MOVAB    102(R0), R0                              ;
                                      50                 60   9A 0005D          MOVZBL   (R0), R0                                 ;
                     53               50                 54   C1 00060          ADDL3    I, R0, R3                                ;
                     50               54                 09   78 00064          ASHL     #9, I, R0                                ; 0204
                     52               50       0000G     CF   C1 00068          ADDL3    CONV$AR_AREA_BLOCK, R0, R2               ;
                                                      0000V   30 0006E          BSBW     READ_BLOCK                               ;
                     DF               54                 55   F2 00071 2$:      AOBLSS   R5, I, 1$                                ;
                                      50       0000G     CF   D0 00075          MOVL     CONV$AR_PROLOGUE, R0                     ; 0209
                                      50          74     A0   9E 0007A          MOVAB    116(R0), R0                              ;
                                      50                 60   3C 0007E          MOVZWL   (R0), R0                                 ;
                                      01                 50   B1 00081          CMPW     R0, #1                                   ; 0211
                                                         07   12 00084          BNEQ     3$                                       ;
                              0000G   CF                 01   90 00086          MOVB     #1, CONV$GB_PROL_V1                      ;
                                                         25   11 0008B          BRB      6$                                       ;
                                      02                 50   B1 0008D 3$:      CMPW     R0, #2                                   ; 0212
                                                         07   12 00090          BNEQ     4$                                       ;
                              0000G   CF                 01   90 00092          MOVB     #1, CONV$GB_PROL_V2                      ;
                                                         19   11 00097          BRB      6$                                       ;
                                      03                 50   B1 00099 4$:      CMPW     R0, #3                                   ; 0213
                                                         07   12 0009C          BNEQ     5$                                       ;
                              0000G   CF                 01   90 0009E          MOVB     #1, CONV$GB_PROL_V3                      ;
                                                         0D   11 000A3          BRB      6$                                       ;
                                         00000000G       8F   DD 000A5 5$:      PUSHL    #CONV$_PLV                               ; 0214
                          00000000G   00                 01   FB 000AB          CALLS    #1, LIB$STOP                             ;
                                                         3C   BA 000B2 6$:      POPR     #^M<R2,R3,R4,R5>                         ; 0219
                                                         05   000B4             RSB                                               ;

; Routine Size:  181 bytes,    Routine Base:  _CONV$CODE + 0000
;  224          0220  1 %SBTTL  'WRITE_PROLOGUE'
;  225          0221  1 GLOBAL ROUTINE  CONV$$WRITE_PROLOGUE : NOVALUE =
;  226          0222  1 !++
;  227          0223  1 !
;  228          0224  1 ! Functional Description:
;  229          0225  1 !
;  230          0226  1 !       Writes the prologue area blocks back to the output file
;  231          0227  1 !
;  232          0228  1 ! Calling Sequence:
;  233          0229  1 !
;  234          0230  1 !       CONV$$WRITE_PROLOGUE()
;  235          0231  1 !
;  236          0232  1 ! Input Parameters:
;  237          0233  1 !       none
;  238          0234  1 !
;  239          0235  1 ! Implicit Inputs:
;  240          0236  1 !       none
;  241          0237  1 !
;  242          0238  1 ! Output Parameters:
;  243          0239  1 !       none
;  244          0240  1 !
;  245          0241  1 ! Implicit Outputs:
;  246          0242  1 !       none
;  247          0243  1 !
;  248          0244  1 ! Routine Value:
;  249          0245  1 !       none
;  250          0246  1 !
;  251          0247  1 ! Routines Called:
;  252          0248  1 !
;  253          0249  1 !       WRITE_BLOCK
;  254          0250  1 !
;  255          0251  1 ! Side Effects:
;  256          0252  1 !       none
;  257          0253  1 !
;  258          0254  1 !--
;  259          0255  1
;  260          0256  2     BEGIN
;  261          0257  2
;  262          0258  2     ! Write each of the area blocks
;  263          0259  2     !
;  264          0260  2     INCR I FROM 0 TO .AREA_BLOCKS - 1
;  265          0261  2     DO
;  266          0262  2         WRITE_BLOCK( .CONV$AR_AREA_BLOCK + ( .I  * BLOCK_SIZE ),
;  267          0263  2                         .CONV$AR_PROLOGUE [ PLG$B_AVBN ] + .I );
;  268          0264  2
;  269          0265  2
;  270          0266  2     RETURN
;  271          0267  2
;  272          0268  1     END;



                                        0FFC 00000        .ENTRY  CONV$$WRITE_PROLOGUE, Save R2,R3,R4,R5,R6,- ; 0221
                                                                  R7,R8,R9,R10,R11                             ;
                            55     0000'  CF   D0 00002        MOVL    AREA_BLOCKS, R5                              ; 0260
```

```
                    54            01  CE 00007        MNEGL   #1, I                                    ;
                                  1D  11 0000A        BRB     2$                                       ;
                    50     0000G  CF  D0 0000C 1$:    MOVL    CONV$AR_PROLOGUE, R0                     ; 0263
                    50        66  A0  9E 00011        MOVAB   102(R0), R0                              ;
                    50            60  9A 00015        MOVZBL  (R0), R0                                 ;
        53          50            54  C1 00018        ADDL3   I, R0, R3                                ;
        50          54            09  78 0001C        ASHL    #9, I, R0                                ; 0262
        52          50     0000G  CF  C1 00020        ADDL3   CONV$AR_AREA_BLOCK, R0, R2               ;
                               0000V  30 00026        BSBW    WRITE_BLOCK                              ;
        DF          54            55  F2 00029 2$:    AOBLSS  R5, I, 1$                                ;
                                      04 0002D        RET                                              ; 0268

; Routine Size:  46 bytes,    Routine Base:  _CONV$CODE + 00B5
```

```
1 %SBTTL  'SET_KEY_DESC'
1 GLOBAL ROUTINE  CONV$$SET_KEY_DESC ( KEY ) : CL$SET_KEY_DESC =
1 !++
1 !
1 ! Functional Description:
1 !
1 !       Sets the key descriptor from the output files
1 !       prologue to the requested key of reference.
1 !       This routine WILL reread the key descriptor from
1 !       the file.
1 !
1 ! Calling Sequence:
1 !
1 !       CONV$$SET_KEY_DESC( key )
1 !
1 ! Input Parameters:
1 !
1 !       key      - Key of refrence to get
1 !
1 ! Implicit Inputs:
1 !
1 !       CONV$GL_KEY_DESC_BUF
1 !
1 ! Output Parameters:
1 !       none
1 !
1 ! Implicit Outputs:
1 !
1 !       KEY_DESC
1 !       CONV$GL_KEY_DESC_VBN
1 !
1 ! Routine Value:
1 !
1 !       CONV$_SUCCESS or CONV$_NOKEY (from get_next_key)
1 !
1 ! Routines Called:
1 !
1 !       CONV$$GET_NEXT_KEY
1 !
1 ! Side Effects:
1 !       none
1 !
1 !--
1
2     BEGIN
2
2     DEFINE_KEY_DESC;
2
2     LOCAL       STATUS;
2
2     STATUS = CONV$_SUCCESS;
2
2     ! Reset to the primary key then search from there
2     !
2     KEY_DESC = .CONV$GL_KEY_DESC_BUF;
2
2     CONV$GL_KEY_DESC_VBN = 1;
```

```
0326  2
0327  2          ! Read the first key
0328  2          !
0329  2          READ_BLOCK( .KEY_DESC,.CONV$GL_KEY_DESC_VBN );
0330  2
0331  2          ! Loop until you find the correct key
0332  2          !
0333  3          WHILE .STATUS AND ( .KEY NEQU .KEY_DESC [ KEY$B_KEYREF ] )
0334  2          DO
0335  2
0336  2              ! If there are no keys then what a bummer
0337  2              !
0338  2              STATUS = CONV$$GET_NEXT_KEY();
0339  2
0340  2          RETURN .STATUS
0341  2
0342  1          END;



                                         1C  BB 00000 CONV$$SET_KEY_DESC::
                                                               PUSHR   #^M<R2,R3,R4>                                   ; 0270
                           54            01  D0 00002          MOVL    #1, STATUS                                      ; 0319
                           5B      0000'  CF  D0 00005          MOVL    CONV$GL_KEY_DESC_BUF, KEY_DESC                  ; 0323
                 0000'     CF            01  D0 0000A          MOVL    #1, CONV$GL_KEY_DESC_VBN                        ; 0325
                           53      0000'  CF  D0 0000F          MOVL    CONV$GL_KEY_DESC_VBN, R3                        ; 0329
                           52            5B  D0 00014          MOVL    KEY_DESC, R2                                    ;
                                      0000V  30 00017          BSBW    READ_BLOCK                                      ;
                           11            54  E9 0001A 1$:      BLBC    STATUS, 2$                                      ; 0333
   10   AE       15   AB   08            00  ED 0001D          CMPZV   #0, #8, 21(KEY_DESC), KEY                       ;
                                         08  13 00024          BEQL    2$                                              ;
                                      0000V  30 00026          BSBW    CONV$$GET_NEXT_KEY                              ; 0338
                           54            50  D0 00029          MOVL    R0, STATUS                                      ;
                                         EC  11 0002C          BRB     1$                                              ;
                           50            54  D0 0002E 2$:      MOVL    STATUS, R0                                      ; 0340
                                         1C  BA 00031          POPR    #^M<R2,R3,R4>                                   ; 0342
                                         05  00033             RSB                                                     ;

; Routine Size:  52 bytes,    Routine Base:  _CONV$CODE + 00E3
```

```
1 %SBTTL  'GET_NEXT_KEY'
1 GLOBAL ROUTINE CONV$$GET_NEXT_KEY : CL$GET_NEXT_KEY =
1 !++
1 !
1 ! Functional Description:
1 !
1 !       Sets the key descriptor from the output files
1 !       prologue to the next key of reference if any
1 !
1 ! Calling Sequence:
1 !
1 !       CONV$$GET_NEXT_KEY()
1 !
1 ! Input Parameters:
1 !       none
1 !
1 ! Implicit Inputs:
1 !
1 !       KEY_DESC
1 !
1 ! Output Parameters:
1 !       none
1 !
1 ! Implicit Outputs:
1 !
1 !       KEY_DESC
1 !
1 ! Routine Value:
1 !
1 !       CONV$_SUCCESS or CONV$_NOKEY
1 !
1 ! Routines Called:
1 !
1 !       READ_BLOCK
1 !
1 ! Side Effects:
1 !
1 !       Could read a new key desciptor into memory
1 !
1 !--
1
2     BEGIN
2
2     DEFINE_KEY_DESC;
2
2     ! If the next key in the chain is not in this block
2     ! then get the next block in the chain
2     !
2     IF .KEY_DESC [ KEY$L_IDXFL ] NEQ 0
2     THEN
3         BEGIN
3
3         ! Get the VBN of the next block
3         !
3         CONV$GL_KEY_DESC_VBN = .KEY_DESC [ KEY$L_IDXFL ];
3
3         ! Have key block point to the right place in the new block
```

```
;  406          0400  3             !
;  407          0401  3             KEY_DESC = .CONV$GL_KEY_DESC_BUF + .KEY_DESC [ KEY$W_NOFF ];
;  408          0402  3
;  409          0403  3             ! Read the block
;  410          0404  3             !
;  411          0405  3             READ_BLOCK( .CONV$GL_KEY_DESC_BUF,.CONV$GL_KEY_DESC_VBN )
;  412          0406  3
;  413          0407  3             END
;  414          0408  2         ELSE
;  415          0409  2
;  416          0410  2             ! If the offset is 0 then there are no more keys
;  417          0411  2             !
;  418          0412  2             IF .KEY_DESC [ KEY$W_NOFF ] EQL 0
;  419          0413  2             THEN
;  420          0414  2                 RETURN CONV$_NOKEY
;  421          0415  2             ELSE
;  422          0416  2
;  423          0417  2                 ! Point the key block to the next key descriptor
;  424          0418  2                 !
;  425          0419  2                 KEY_DESC = .CONV$GL_KEY_DESC_BUF + .KEY_DESC [ KEY$W_NOFF ];
;  426          0420  2
;  427          0421  2         RETURN CONV$_SUCCESS
;  428          0422  2
;  429          0423  1         END;



                                    0C  BB 00000 CONV$$GET_NEXT_KEY::
                                                         POSHR    #^M<R2,R3>                                       ; 0344
                                    6B  D5 00002         TSTL     (KEY_DESC)                                       ; 0391
                                    18  13 00004         BEQL     1$                                               ;
          0000'  CF                 6B  D0 00006         MOVL     (KEY_DESC), CONV$GL_KEY_DESC_VBN                 ; 0397
                 5B         04      AB  3C 0000B         MOVZWL   4(KEY_DESC), KEY_DESC                            ; 0401
                 5B       0000'     CF  C0 0000F         ADDL2    CONV$GL_KEY_DESC_BUF, KEY_DESC                   ;
                 52       0000'     CF  7D 00014         MOVQ     CONV$GL_KEY_DESC_BUF, R2                         ; 0405
                               0000V    30 00019         BSBW     READ_BLOCK                                       ;
                                    17  11 0001C         BRB      3$                                               ;
                            04      AB  B5 0001E 1$:     TSTW     4(KEY_DESC)                                      ; 0412
                                    09  12 00021         BNEQ     2$                                               ;
                 50 00000000G       8F  D0 00023         MOVL     #CONV$_NOKEY, R0                                 ; 0414
                                    0C  11 0002A         BRB      4$                                               ;
                 5B         04      AB  3C 0002C 2$:     MOVZWL   4(KEY_DESC), KEY_DESC                            ; 0419
                 5B       0000'     CF  C0 00030         ADDL2    CONV$GL_KEY_DESC_BUF, KEY_DESC                   ;
                 50                 01  D0 00035 3$:     MOVL     #1, R0                                           ; 0421
                                    0C  BA 00038 4$:     POPR     #^M<R2,R3>                                       ; 0423
                                        05 0003A         RSB                                                       ;

; Routine Size:  59 bytes,    Routine Base:  _CONV$CODE + 0117
;   431           0424   1 %SBTTL  'WRITE_KEY_DESC'
;   432           0425   1 GLOBAL ROUTINE  CONV$$WRITE_KEY_DESC : CL$WRITE_KEY_DESC NOVALUE =
;   433           0426   1 !++
;   434           0427   1 !
;   435           0428   1 ! Functional Description:
;   436           0429   1 !
;   437           0430   1 !       Writes back to the output file the current key descriptor
;   438           0431   1 !
;   439           0432   1 ! Calling Sequence:
;   440           0433   1 !
;   441           0434   1 !       CONV$$WRITE_KEY_DESC()
;   442           0435   1 !
;   443           0436   1 ! Input Parameters:
;   444           0437   1 !       none
;   445           0438   1 !
;   446           0439   1 ! Implicit Inputs:
;   447           0440   1 !
;   448           0441   1 !       CONV$GL_KEY_DESC_BUF
;   449           0442   1 !       CONV$GL_KEY_DESC_VBN
;   450           0443   1 !
;   451           0444   1 ! Output Parameters:
;   452           0445   1 !       none
;   453           0446   1 !
;   454           0447   1 ! Implicit Outputs:
;   455           0448   1 !       none
;   456           0449   1 !
;   457           0450   1 ! Routine Value:
;   458           0451   1 !       none
;   459           0452   1 !
;   460           0453   1 ! Routines Called:
;   461           0454   1 !
;   462           0455   1 !       WRITE_BLOCK
;   463           0456   1 !
;   464           0457   1 ! Side Effects:
;   465           0458   1 !       none
;   466           0459   1 !
;   467           0460   1 !--
;   468           0461   1
;   469           0462   1    BEGIN
;   470           0463   2
;   471           0464   2    WRITE_BLOCK( .CONV$GL_KEY_DESC_BUF,.CONV$GL_KEY_DESC_VBN );
;   472           0465   2
;   473           0466   2    RETURN
;   474           0467   2
;   475           0468   1    END;



                                          0C  BB 00000 CONV$$WRITE_KEY_DESC::
                                                                PUSHR   #^M<R2,R3>                                 ; 0425
                           52      0000'  CF  7D 00002          MOVQ    CONV$GL_KEY_DESC_BUF, R2                   ; 0464
                                       0000V  30 00007          BSBW    WRITE_BLOCK                                ;
                                          0C  BA 0000A          POPR    #^M<R2,R3>                                 ; 0468
                                              05 0000C          RSB                                                ;
```

```
; Routine Size:  13 bytes,    Routine Base:  _CONV$CODE + 0152
```

```
1 %SBTTL  'WRITE_AREA_DESC'
1 GLOBAL ROUTINE  CONV$$WRITE_AREA_DESC ( AREA ) : CL$WRITE_AREA_DESC NOVALUE =
1 !++
1 !
1 ! Functional Description:
1 !
1 !         Writes back to the output file the current key descriptor
1 !
1 ! Calling Sequence:
1 !
1 !         CONV$$WRITE_AREA_DESC( AREA )
1 !
1 ! Input Parameters:
1 !
1 !         AREA - Area number to write
1 !
1 ! Implicit Inputs:
1 !
1 !         CONV$AR_AREA_BLOCK
1 !
1 ! Output Parameters:
1 !         none
1 !
1 ! Implicit Outputs:
1 !         none
1 !
1 ! Routine Value:
1 !         none
1 !
1 ! Routines Called:
1 !
1 !         WRITE_BLOCK
1 !
1 ! Side Effects:
1 !         none
1 !
1 !--
1
2     BEGIN
2
2     LOCAL
2         VBN,
2         BUFFER;
2
2     ! Determine what block the area descritor is in
2     !
2     VBN = .CONV$AR_PROLOGUE [ PLG$B_AVBN ] + ( ( .AREA - 1 ) / 8 );
2
2     ! Where in the buffer is the area descriptor
2     !
2     BUFFER = .CONV$AR_AREA_BLOCK +
2                 ( ( .VBN - .CONV$AR_PROLOGUE [ PLG$B_AVBN ] ) * BLOCK_SIZE );
2
2     WRITE_BLOCK( .BUFFER,.VBN );
2
2     RETURN
2
```

```
;  534          0526  1      END;



                                          0C   BB 00000 CONV$$WRITE_AREA_DESC::
                                                                  PUSHR   #^M<R2,R3>                                    ; 0470
                             50      0000G CF   D0 00002          MOVL    CONV$AR_PROLOGUE, R0                          ; 0515
                                           51   D7 00007          DECL    R1                                            ;
                             51            08   C6 00009          DIVL2   #8, R1                                        ;
                             52        66  A0   9A 0000C          MOVZBL  102(R0), R2                                   ;
                             51            52   C0 00010          ADDL2   R2, VBN                                       ;
                             53        66  A0   9A 00013          MOVZBL  102(R0), R3                                   ; 0520
                 53          51            53   C3 00017          SUBL3   R3, VBN, R3                                   ;
                 53          53            09   78 0001B          ASHL    #9, R3, R3                                    ;
                 52          53      0000G CF   C1 0001F          ADDL3   CONV$AR_AREA_BLOCK, R3, BUFFER                ;
                             53            51   D0 00025          MOVL    VBN, R3                                       ; 0522
                                         0000V  30 00028          BSBW    WRITE_BLOCK                                   ;
                                           0C   BA 0002B          POPR    #^M<R2,R3>                                    ; 0526
                                                05 0002D          RSB                                                   ;

; Routine Size:  46 bytes,    Routine Base:  _CONV$CODE + 015F
```

```
1 %SBTTL  'READ_BLOCK'
1 ROUTINE READ_BLOCK ( BUFFER : REF VECTOR [ ,WORD ],VBN ) : CL$READ_BLOCK NOVALUE =
1 !++
1 !
1 ! Functional Description:
1 !
1 !       Reads a block in the output files prologue and checks the
1 !       checksum value for it
1 !
1 ! Calling Sequence:
1 !
1 !       READ_BLOCK( buffer,vbn )
1 !
1 ! Input Parameters:
1 !
1 !       buffer  - Buffer to read the block into
1 !       vbn     - VBN in the prologue to read
1 !
1 ! Implicit Inputs:
1 !       none
1 !
1 ! Output Parameters:
1 !       none
1 !
1 ! Implicit Outputs:
1 !       none
1 !
1 ! Routine Value:
1 !       none
1 !
1 ! Routines Called:
1 !
1 !       CHECKSUM
1 !
1 ! Side Effects:
1 !       none
1 !
1 !--
1
2    BEGIN
2
2    CONV$AB_OUT_RAB [ RAB$L_BKT ] = .VBN;
2    CONV$AB_OUT_RAB [ RAB$L_UBF ] = .BUFFER;
2    CONV$AB_OUT_RAB [ RAB$W_USZ ] = BLOCK_SIZE;
2
2    $READ( RAB=CONV$AB_OUT_RAB,ERR=CONV$$RMS_READ_ERROR );
2
2    IF .BUFFER [ 255 ] NEQU CHECKSUM( .BUFFER )
2    THEN
3        BEGIN
3
3        LOCAL       FILE_NAME : DESC_BLK;
3
3        ! The file is open so there should be a full name around
3        !
3        FILE_NAME [ DSC$W_LENGTH ] = .CONV$AB_OUT_NAM [ NAM$B_RSL ];
3        FILE_NAME [ DSC$A_POINTER ] = .CONV$AB_OUT_NAM [ NAM$L_RSA ];
```

```
;  593          0584  3
;  594          0585  3          SIGNAL_STOP( CONV$_READERR,1,FILE_NAME,CONV$_PROERR,1,.VBN )
;  595          0586  3
;  596          0587  2          END;
;  597          0588  2
;  598          0589  2       RETURN
;  599          0590  2
;  600          0591  1       END;


                                                                    .EXTRN  SYS$READ

                                  5E          08  C2 00000 READ_BLOCK:
                                                                    SUBL2   #8, SP                                      ; 0528
                          0000G   CF          53  D0 00003          MOVL    VBN, CONV$AB_OUT_RAB+56                     ; 0568
                          0000G   CF          52  D0 00008          MOVL    BUFFER, CONV$AB_OUT_RAB+36                  ; 0569
                          0000G   CF    0200  8F  B0 0000D          MOVW    #512, CONV$AB_OUT_RAB+32                    ; 0570
                                        0000G CF  9F 00014          PUSHAB  CONV$$RMS_READ_ERROR                        ; 0572
                                        0000G CF  9F 00018          PUSHAB  CONV$AB_OUT_RAB                             ;
                      00000000G   00          02  FB 0001C          CALLS   #2, SYS$READ                                ;
                                           0000V  30 00023          BSBW    CHECKSUM                                    ; 0574
          50     01FE    C2       10          00  ED 00026          CMPZV   #0, #16, 510(BUFFER), R0                    ;
                                              27  13 0002D          BEQL    1$                                          ;
                                  6E    0000G CF  9B 0002F          MOVZBW  CONV$AB_OUT_NAM+3, FILE_NAME                ; 0582
                             04   AE    0000G CF  D0 00034          MOVL    CONV$AB_OUT_NAM+4, FILE_NAME+4              ; 0583
                                              53  DD 0003A          PUSHL   VBN                                         ; 0585
                                              01  DD 0003C          PUSHL   #1                                          ;
                                   00000000G  8F  DD 0003E          PUSHL   #CONV$_PROERR                               ;
                                          0C  AE  9F 00044          PUSHAB  FILE_NAME                                   ;
                                              01  DD 00047          PUSHL   #1                                          ;
                                   00000000G  8F  DD 00049          PUSHL   #CONV$_READERR                              ;
                      00000000G   00          06  FB 0004F          CALLS   #6, LIB$STOP                                ;
                                  5E          08  C0 00056 1$:      ADDL2   #8, SP                                      ; 0591
                                              05 00059              RSB                                                 ;

; Routine Size:  90 bytes,    Routine Base:  _CONV$CODE + 018D
```

```
1 %SBTTL  'WRITE_BLOCK'
1 ROUTINE WRITE_BLOCK ( BUFFER : REF VECTOR[,WORD ],VBN ) : CL$WRITE_BLOCK NOVALUE =
1 !++
1 !
1 ! Functional Description:
1 !
1 !       Calculates a checksum for a block and writes the block to
1 !       the output files prologue
1 !
1 ! Calling Sequence:
1 !
1 !       WRITE_BLOCK( buffer,vbn )
1 !
1 ! Input Parameters:
1 !
1 !       buffer  - Buffer to write the block from
1 !       vbn     - VBN in the prologue to write
1 !
1 ! Implicit Inputs:
1 !       none
1 !
1 ! Output Parameters:
1 !       none
1 !
1 ! Implicit Outputs:
1 !       none
1 !
1 ! Routine Value:
1 !       none
1 !
1 ! Routines Called:
1 !
1 !       CHECKSUM
1 !
1 ! Side Effects:
1 !       none
1 !
1 !--
1
2     BEGIN
2
2     BUFFER [ 255 ] = CHECKSUM ( .BUFFER );
2
2     CONV$AB_OUT_RAB [ RAB$L_BKT ] = .VBN;
2     CONV$AB_OUT_RAB [ RAB$L_RBF ] = .BUFFER;
2     CONV$AB_OUT_RAB [ RAB$W_RSZ ] = BLOCK_SIZE;
2
2     ! It's ok to call rms_read_error it works for writes tc
2     !
2     $WRITE( RAB=CONV$AB_OUT_RAB,ERR=CONV$$RMS_READ_ERROR );
2
2     RETURN
2
1     END;
```

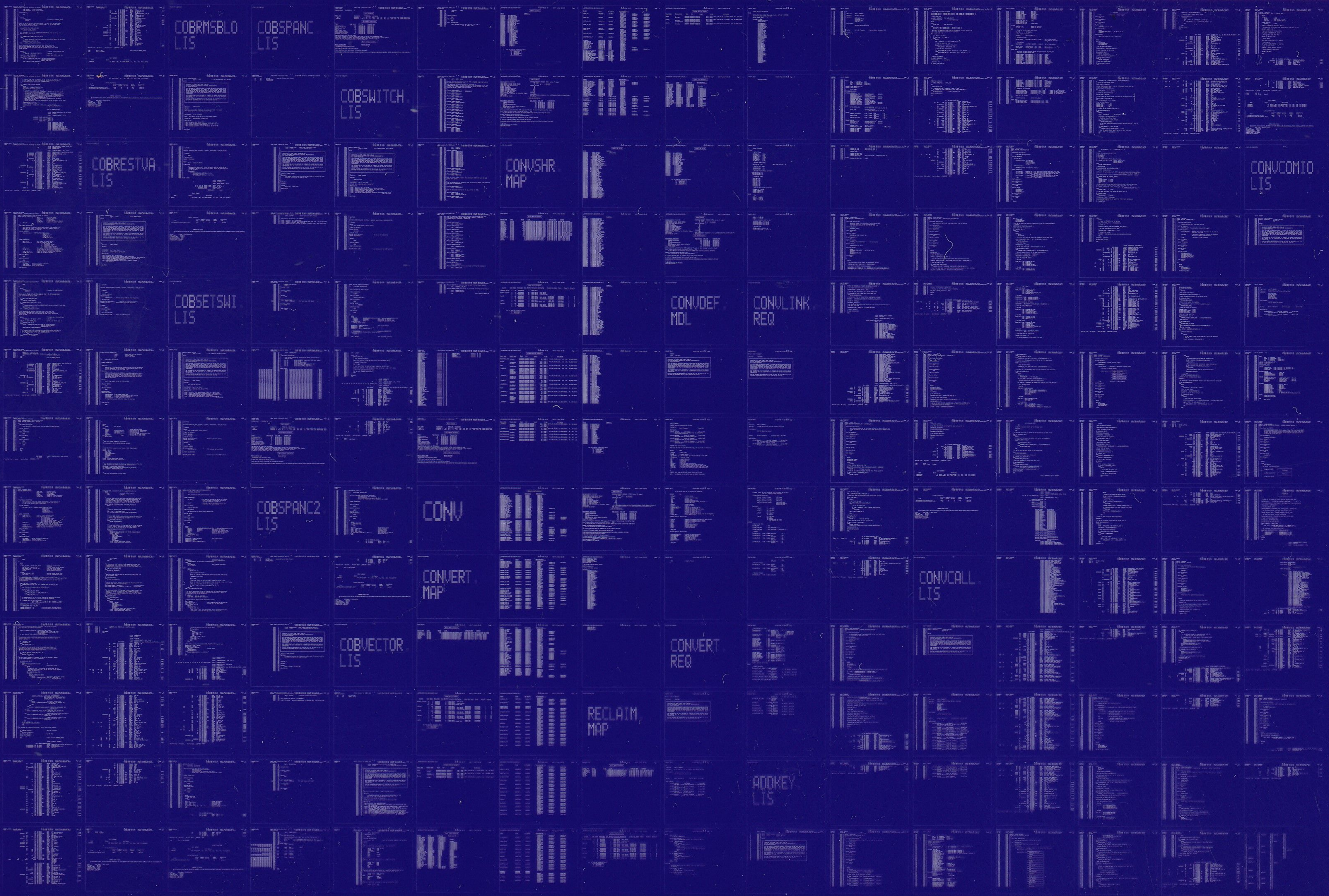
```
                                                               .EXTRN  SYS$WRITE

                                       0000V 30 00000 WRITE_BLOCK:
                                                               BSBW    CHECKSUM                                    ; 0633
                01FE   C2                   50 B0 00003        MOVW    R0, 510(BUFFER)                             ;
                0000G  CF                   53 D0 00008        MOVL    VBN, CONV$AB_OUT_RAB+56                     ; 0635
                0000G  CF                   52 D0 0000D        MOVL    BUFFER, CONV$AB_OUT_RAB+40                  ; 0636
                0000G  CF         0200      8F B0 00012        MOVW    #512, CONV$AB_OUT_RAB+34                    ; 0637
                                  0000G  CF 9F 00019           PUSHAB  CONV$$RMS_READ_ERROR                        ; 0641
                                  0000G  CF 9F 0001D           PUSHAB  CONV$AB_OUT_RAB                             ;
            00000000G  00                   02 FB 00021        CALLS   #2, SYS$WRITE                               ;
                                               05 00028        RSB                                                 ; 0645

; Routine Size:  41 bytes,    Routine Base:  _CONV$CODE + 01E7
```

```
1 %SBTTL  'CHECKSUM'
1 ROUTINE CHECKSUM ( BLOCK : REF VECTOR [ ,WORD ] ) : CL$CHECKSUM =
1 !++
1 !
1 ! Functional Description:
1 !
1 !       Calculates a checksum for a block and writes the block to
1 !       the output files prologue
1 !
1 ! Calling Sequence:
1 !
1 !       CHECKSUM( buffer )
1 !
1 ! Input Parameters:
1 !
1 !       buffer  - 512 byte buffer to calculate the checksum for
1 !
1 ! Implicit Inputs:
1 !       none
1 !
1 ! Output Parameters:
1 !       none
1 !
1 ! Implicit Outputs:
1 !       none
1 !
1 ! Routine Value:
1 !
1 !       R0      - Checksum
1 !
1 ! Routines Called:
1 !       none
1 !
1 ! Side Effects:
1 !       none
1 !
1 !--
1
2     BEGIN
2
2     ! Calculate the checksum for this block
2     !
2     LOCAL    CHECKSUM : WORD;
2
2     CHECKSUM = 0;
2
2     INCR J FROM 0 TO 254 BY 1
2     DO
2         CHECKSUM = .CHECKSUM + .BLOCK [ .J ];
2
2     RETURN .CHECKSUM
2
1     END;
```

```
                                              51  B4 00000 CHECKSUM:
                                                                      CLRW     CHECKSUM                                  ; 0690
                                              50  D4 00002            CLRL     J                                         ; 0694
                                  51        6240  A0 00004 1$:        ADDW2    (BLOCK)[J], CHECKSUM
                  F4              50 000000FE   8F  F3 00008          AOBLEQ   #254, J, 1$                               ;
                                  50          51  3C 00010            MOVZWL   CHECKSUM, R0                              ; 0696
                                                  05 00013            RSB                                                ; 0698

; Routine Size:  20 bytes,    Routine Base:  _CONV$CODE + 0210


;  710          0699  1
;  711          0700  0 END      ELUDOM



                                                                      .EXTRN  LIB$STOP

;                                   PSECT SUMMARY
;
;       Name                     Bytes                      Attributes
;
;  _CONV$GLOBAL                      8  NOVEC,  WRT,  RD ,NOEXE,NOSHR,  LCL,  REL,  CON,  PIC,ALIGN(2)
;  _CONV$OWN                         4  NOVEC,  WRT,  RD ,NOEXE,NOSHR,  LCL,  REL,  CON,  PIC,ALIGN(2)
;  _CONV$CODE                      548  NOVEC,NOWRT,  RD ,  EXE,  SHR,  LCL,  REL,  CON,  PIC,ALIGN(2)



;                          Library Statistics
;
;                                          -------- Symbols --------      Pages      Processing
;       File                               Total    Loaded   Percent      Mapped     Time
;
;  _$255$DUA28:[SYSLIB]LIB.L32;1           18619        32         0       1000        00:01.8
;  _$255$DUA28:[CONV.SRC]CONVERT.L32;1       165        12         7         17        00:00.2



;                                    COMMAND QUALIFIERS

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

; Size:          548 code + 12 data bytes
; Run Time:         00:13.1
; Elapsed Time:     00:35.3
; Lines/CPU Min:     3211
; Lexemes/CPU-Min: 10816
; Memory Used:  97 pages
; Compilation Complete
```

0064 AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY


COBRMSBLO
LIS

COBSPANC
LIS

COBSWITCH
LIS

COBRESTVA
LIS

CONVSHR
MAP

CONVCOMIO
LIS

COBSETSWI
LIS

CONVDEF
MDL

CONVLINK
REQ

COBSPANC2
LIS

CONV

CONVERT
MAP

CONVCALL
LIS

COBVECTOR
LIS

CONVERT
REQ

RECLAIM
MAP

ADDKEY
LIS

0065 AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY

CONVDATA
LIS

CONVFSTLD
LIS

CONVFSTIO
LIS

CONVFSTRC
LIS

CONVFILES
LIS

CONVERROR
LIS

CONVFASTM
LIS

CONVDCL
LIS