K 14

```
UUU         UUU  TTTTTTTTTTTTTTT     IIIIIIIII     LLL                 333333333        222222222
UUU         UUU  TTTTTTTTTTTTTTT     IIIIIIIII     LLL                 333333333        222222222
UUU         UUU  TTTTTTTTTTTTTTT     IIIIIIIII     LLL                 333333333        222222222
UUU         UUU        TTT              III        LLL              333         333  222         222
UUU         UUU        TTT              III        LLL              333         333  222         222
UUU         UUU        TTT              III        LLL              333         333  222         222
UUU         UUU        TTT              III        LLL                          333              222
UUU         UUU        TTT              III        LLL                          333              222
UUU         UUU        TTT              III        LLL                          333              222
UUU         UUU        TTT              III        LLL                       333              222
UUU         UUU        TTT              III        LLL                       333              222
UUU         UUU        TTT              III        LLL                       333              222
UUU         UUU        TTT              III        LLL                          333        222
UUU         UUU        TTT              III        LLL                          333        222
UUU         UUU        TTT              III        LLL                          333        222
UUU         UUU        TTT              III        LLL              333         333     222
UUU         UUU        TTT              III        LLL              333         333     222
UUU         UUU        TTT              III        LLL              333         333     222
UUUUUUUUUUUUUUU        TTT           IIIIIIIII     LLLLLLLLLLLLLLL     333333333     222222222222222
UUUUUUUUUUUUUUU        TTT           IIIIIIIII     LLLLLLLLLLLLLLL     333333333     222222222222222
UUUUUUUUUUUUUUU        TTT           IIIIIIIII     LLLLLLLLLLLLLLL     333333333     222222222222222
```

FILEID**CHECKSUM

```
  CCCCCCCC  HH      HH  EEEEEEEEEE    CCCCCCCC  KK      KK    SSSSSSSS  UU      UU  MM      MM
  CCCCCCCC  HH      HH  EEEEEEEEEE    CCCCCCCC  KK      KK    SSSSSSSS  UU      UU  MM      MM
CC          HH      HH  EE          CC          KK      KK  SS          UU      UU  MMMM  MMMM
CC          HH      HH  EE          CC          KK      KK  SS          UU      UU  MMMM  MMMM
CC          HH      HH  EE          CC          KK    KK    SS          UU      UU  MM  MM  MM
CC          HH      HH  EE          CC          KK    KK    SS          UU      UU  MM  MM  MM
CC          HHHHHHHHHH  EEEEEEEE    CC          KKKKKK        SSSSSS    UU      UU  MM      MM
CC          HHHHHHHHHH  EEEEEEEE    CC          KKKKKK        SSSSSS    UU      UU  MM      MM
CC          HH      HH  EE          CC          KK    KK            SS  UU      UU  MM      MM
CC          HH      HH  EE          CC          KK    KK            SS  UU      UU  MM      MM
CC          HH      HH  EE          CC          KK      KK          SS  UU      UU  MM      MM
CC          HH      HH  EE          CC          KK      KK          SS  UU      UU  MM      MM    ....
  CCCCCCCC  HH      HH  EEEEEEEEEE    CCCCCCCC  KK      KK  SSSSSSSS    UUUUUUUUUU  MM      MM    ....
  CCCCCCCC  HH      HH  EEEEEEEEEE    CCCCCCCC  KK      KK  SSSSSSSS    UUUUUUUUUU  MM      MM    ....
                                                                                                  ....

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

```
0 MODULE checksum_image (
0                       IDENT='V04-000',
0                       MAIN=checksum$main
0                       %TITLE  'Checksum the contents of a file'
0                       ) =
0
1 BEGIN
1 !
1 !
1 !****************************************************************************
1 !*                                                                          *
1 !*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                                 *
1 !*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.                  *
1 !*  ALL RIGHTS RESERVED.                                                    *
1 !*                                                                          *
1 !*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED   *
1 !*  ONLY IN  ACCORDANCE WITH  THE  TERMS  OF  SUCH  LICENSE  AND WITH THE   *
1 !*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR  ANY  OTHER   *
1 !*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY   *
1 !*  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE IS  HEREBY   *
1 !*  TRANSFERRED.                                                            *
1 !*                                                                          *
1 !*  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE WITHOUT NOTICE   *
1 !*  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL EQUIPMENT   *
1 !*  CORPORATION.                                                            *
1 !*                                                                          *
1 !*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR  RELIABILITY OF ITS   *
1 !*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.                 *
1 !*                                                                          *
1 !*                                                                          *
1 !****************************************************************************
1 !
1 !
1 !++
1 !
1 ! FACILITY:  Checksum VAX/VMS image file
1 !
1 ! ABSTRACT:
1 !               Computes the checksum of a file or an image.  The
1 !               qualifiers on command are /OUTPUT, /FILE and /IMAGE.
1 !               The /FILE and /IMAGE are conflicting qualifiers.  If
1 !               specify /IMAGE, then the parts of the image header
1 !               that can change (Date/Time, etc) are not checksummed.
1 !               The parameter is the input file(s) to checksum.
1 !
1 ! ENVIRONMENT:
1 !
1 !       VAX native, user mode.
1 !
1 !--
1 !
1 !
1 ! AUTHOR:  Benn Schreiber,       CREATION DATE:  17-Feb-1981
1 !
1 ! MODIFIED BY:
1 !
1 !       V03-002 LJA0111 Laurie J. Anderson      16-Feb-1984
```

```
1 !                     Change the image checksum to use the new EXEC image
1 !                     header decode routines.  Change the parameters to
1 !                     LIB$FIND_FILE to use the new related file parsing
1 !                     arguments.  Also, add relevant comments so the
1 !                     next person can figure out this mess.
1 !
1 !             V03-001 BLS0180 Benn Schreiber          13-Aug-1982
1 !                     Support record by record checksum
1 !
1 !--
```

```
LIBRARY
        'SYS$LIBRARY:LIB.L32';              !System macro definitions
REQUIRE
        'SHRLIB$:IMGMSGDEF.R32';            !Image decode routine's messge literals

MACRO
!
! Macro to initialize a dynamic descriptor
!
    $init_ddesc(d) =                        ! Init dynamic string descriptor
            begin
            d[dsc$w_length] = 0;
            d[dsc$b_class] = dsc$k_class_d;
            d[dsc$b_dtype] = dsc$k_dtype_t;
            d[dsc$a_pointer] = 0;
            end%,
!
! Macro to generate a pointer to an ascic string
!
    cstring (string) = UPLIT(%ASCIC string)%,
!
! Macro to fao and output a string
!
    write (string) =
        cksm_output (fao_buffer (cstring(string)
        %IF %LENGTH GTR 1 %THEN ,%REMAINING %FI))%;

!
! Message codes from shared message definitions
!

$SHR_MSGDEF(cksm, 3 ,LOCAL,                 !Facility code = 3 (CLI)
            (confqual, severe),             ! Conflicting qualifiers
            (openin, error),                ! error opening "x" as input
            (openout, severe),              ! error opening "x" as output
            (closein, warning),             ! error closing "x" as input
            (closeout, warning),            ! error closing "x" as output
            (readerr, error),               ! error reading "x"
            (writeerr, error),              ! error writing "x"
            (parsefail, warning));          ! error parsing "x"

FORWARD ROUTINE
    read_header,                            !Read and process image header
    checksum_section,                       !Checksum an image section
    cksm_foreign,                           !Checksum non-image file
    cksm_header,                            !Checksum image header data
    read_blocks,                            !Read next block of header
    fao_buffer,                             !Call fao and return descriptor
    cksm_output,                            !Output line to output file
    getfilename;                            !Get string descriptor for file

EXTERNAL ROUTINE
    cli$get_value : ADDRESSING_MODE(GENERAL),     !Get qualifier value
    cli$present : ADDRESSING_MODE(GENERAL),       !Test if qualifier present
    img$decode_ihd : ADDRESSING_MODE(GENERAL),    !Decode image header
    img$get_next_isd : ADDRESSING_MODE(GENERAL),  !Get all ISD's from image
    lib$get_vm : ADDRESSING_MODE(GENERAL),        !Allocate virtual memory
```

```
        lib$free_vm : ADDRESSING_MODE(GENERAL),         !Free virtual memory
        lib$find_file : ADDRESSING_MODE(GENERAL),       !Search for wild card files
        lib$set_symbol : ADDRESSING_MODE(GENERAL),      !Define local DCL symbol
        str$copy_dx : ADDRESSING_MODE(GENERAL),         !Copy a string
        str$upcase : ADDRESSING_MODE(GENERAL);          !Uppercase a string

EXTERNAL LITERAL
        cksm$_allischk,                                 !Checksum of all image sections
        cksm$_badimghdr,                                !Bad image header format
        cksm$_file,                                     !Print the filename
        cksm$_ihdchksum,                                !Image header checksum
        cksm$_isectchk,                                 !Checksum of one image section
        cksm$_nonimgchk;                                !Non-image file checksum

OWN
        chan,                                           !Channel to input file if image
        find_context : REF $BBLOCK,                     !Context for lib$find_file
        irab : $RAB_DECL,                               !RAB for input
        orab : $RAB_DECL,                               ! and output
        ixabfhc : $XABFHC_DECL,                         !XAB to get file header characteristics to get file size
        outfiledesc : $BBLOCK[dsc$c_s_bln],             !Output file descriptor
        find_result : $BBLOCK[dsc$c_s_bln],             !Current input file
        headerbuffer : REF $BBLOCK,                     !Pointer to user record buffer
        databuffer : REF VECTOR[,LONG],                 !and data buffer
        exitstatus : $BBLOCK[4];                        !Most severe error status

LITERAL
        ihd$k_activoff = ihd$k_length,                  !Offset to activation area
        ihd$k_symdbgoff = ihd$k_activoff + iha$k_length,
        ihd$k_imgidoff = ihd$k_symdbgoff + ihs$k_length,
        ihd$k_patchoff = ihd$k_imgidoff + ihi$k_length,
        ihd$k_maxlength = ihd$k_patchoff,
        maxbytes = 10*512,                              !Maximum record to read
        false = 0,                                      !Logical 0
        true = 1;                                       ! and 1

GLOBAL
        cksm$gl_ihdcksm,                                !Image header checksum
        cksm$gl_chksum,                                 !Isect checksum
        cksm$gl_acucksm,                                !Accumulated checksum
        cksm$gl_maxbyt;                                 !Maximum record to read
```

```
1 %sbttl 'allocate_ubf'
1 ROUTINE allocate_ubf (xab,rab) : NOVALUE =
2 BEGIN
2 !
2 ! Allocate a buffer for records that cross block boundaries
2 !
2 MAP
2      xab : REF $BBLOCK,
2      rab : REF $BBLOCK;
2
2 LOCAL
2      status,
2      blocksize;
2
2 OWN
2      ubf_size,             !Size of last UBF
2      ubf_addr;             !Address of last UBF
2
2 !
2 ! Calculate the max record size based on the longest record in the file.
2 ! If there is a user buffer allocated and it is not big enough, deallocate
2 ! it and get a new one.  If it's bigger than needed, then that's fine
2 !
2 blocksize = MAXU(MAXU(.xab[xab$w_lrl],.xab[xab$w_mrz]),512);
2 IF .blocksize GTRU .ubf_size
3 THEN BEGIN
3      IF .ubf_size NEQ 0
3         THEN lib$free_vm(ubf_size,ubf_addr);
3      ubf_size = .blocksize;
4      IF NOT (status = lib$get_vm(ubf_size,ubf_addr))
3         THEN SIGNAL_STOP(.status);
2      END;
2 !
2 ! Set up rab with size and address of the buffer
2 !
2 rab[rab$w_usz] = .blocksize;
2 rab[rab$l_ubf] = .ubf_addr;
2
2 RETURN;
1 END;
```

```
                        .TITLE  CHECKSUM_IMAGE Checksum the contents of a file
                        .IDENT  \V04-000\

                        .PSECT  $OWN$,NOEXE,2

00000 CHAN:             .BLKB   4
00004 FIND_CONTEXT:
                        .BLKB   4
00008 IRAB:             .BLKB   68
0004C ORAB:             .BLKB   68
00090 IXABFHC:          .BLKB   44
000BC OUTFILEDESC:
                        .BLKB   8
000C4 FIND_RESULT:
                        .BLKB   8
```

```
                              000CC HEADERBUFFER:
                                          .BLKB   4
                              000D0 DATABUFFER:
                                          .BLKB   4
                              000D4 EXITSTATUS:
                                          .BLKB   4
                              000D8 UBF_SIZE:
                                          .BLKB   4
                              000DC UBF_ADDR:
                                          .BLKB   4

                                          .PSECT  $GLOBAL$,NOEXE,2

                              00000 CKSM$GL_IHDCKSM::
                                          .BLKB   4
                              00004 CKSM$GL_CHKSUM::
                                          .BLKB   4
                              00008 CKSM$GL_ACUCKSM::
                                          .BLKB   4
                              0000C CKSM$GL_MAXBYT::
                                          .BLKB   4

                                          .EXTRN  CLI$GET_VALUE, CLI$PRESENT
                                          .EXTRN  IMG$DECODE_IHD, IMG$GET_NEXT_ISD
                                          .EXTRN  LIB$GET_VM, LIB$FREE_VM
                                          .EXTRN  LIB$FIND_FILE, LIB$SET_SYMBOL
                                          .EXTRN  STR$COPY_DX, STR$UPCASE
                                          .EXTRN  CKSM$_ALLISCHK, CKSM$_BADIMGHDR
                                          .EXTRN  CKSM$_FILE, CKSM$_IHDCHKSUM
                                          .EXTRN  CKSM$_ISECTCHK, CKSM$_NONIMGCHK

                                          .PSECT  $CODE$,NOWRT,2

                         000C 00000 ALLOCATE_UBF:
                                          .WORD   Save R2,R3                      ; 0252
            53    0000'   CF   9E 00002       MOVAB   UBF_SIZE, R3                    ; 0274
            50      04    AC   D0 00007       MOVL    XAB, R0                         ;
            51      0A    A0   3C 0000B       MOVZWL  10(R0), R1                      ;
            51      18    A0   B1 0000F       CMPW    24(R0), R1                      ;
                          04   1B 00013       BLEQU   1$                              ;
            51      18    A0   3C 00015       MOVZWL  24(R0), R1                      ;
   0200     8F            51   B1 00019 1$:   CMPW    R1, #512                        ;
                          05   1E 0001E       BGEQU   2$                              ;
            51    0200    8F   3C 00020       MOVZWL  #512, R1                        ;
            52            51   D0 00025 2$:   MOVL    R1, BLOCKSIZE                   ;
            63            52   D1 00028       CMPL    BLOCKSIZE, UBF_SIZE             ; 0275
                          2B   1B 0002B       BLEQU   4$                              ;
                          63   D5 0002D       TSTL    UBF_SIZE                        ; 0277
                          0C   13 0002F       BEQL    3$                              ;
                    04    A3   9F 00031       PUSHAB  UBF_ADDR                        ; 0278
                          53   DD 00034       PUSHL   R3                              ;
 00000000G  00            02   FB 00036       CALLS   #2, LIB$FREE_VM                 ;
            63            52   D0 0003D 3$:   MOVL    BLOCKSIZE, UBF_SIZE             ; 0279
                    04    A3   9F 00040       PUSHAB  UBF_ADDR                        ; 0280
                          53   DD 00043       PUSHL   R3                              ;
 00000000G  00            02   FB 00045       CALLS   #2, LIB$GET_VM                  ;
            09            50   E8 0004C       BLBS    STATUS, 4$                      ;
```

```
                              50  DD 0004F        PUSHL   STATUS                          ; 0281
          00000000G  00       01  FB 00051        CALLS   #1, LIB$STOP                    ;
                     50   08  AC  D0 00058 4$:    MOVL    RAB, R0                         ; 0286
                20   A0       52  B0 0005C        MOVW    BLOCKSIZE, 32(R0)               ;
                24   A0   04  A3  D0 00060        MOVL    UBF_ADDR, 36(R0)                ; 0287
                                  04 00065        RET                                     ; 0290

; Routine Size:  102 bytes,    Routine Base:  $CODE$ + 0000
```

```
;  209          0291  1 %sbttl 'define DCL symbol'
;  210          0292  1 ROUTINE define_sym (value) : NOVALUE =
;  211          0293  2 BEGIN
;  212          0294  2 !
;  213          0295  2 ! Define the symbol CHECKSUM$CHECKSUM with the given value
;  214          0296  2 !
;  215          0297  2 LOCAL
;  216          0298  2     status;
;  217          0299  2
;  218          0300  2 status = lib$set_symbol($descriptor('CHECKSUM$CHECKSUM'),
;  219          0301  2                         fao_buffer(cstring('!UL'),.value));
;  220          0302  2 IF NOT .status
;  221          0303  2     THEN SIGNAL(.status);
;  222          0304  2
;  223          0305  2 RETURN;
;  224          0306  1 END;


                                                              .PSECT  $PLIT$,NOWRT,NOEXE,2

53 4B 43 45 48 43 24 4D 55 53 4B 43 45 48 43  00000 P.AAB:   .ASCII  \CHECKSUM$CHECKSUM\       ;
                                       4D 55  0000F                                             ;
                                              00011          .BLKB   3
                                   00000011   00014 P.AAA:   .LONG   17                        ;
                                   00000000'  00018          .ADDRESS P.AAB                    ;
                              4C 55 21 03     0001C P.AAC:   .ASCII  <3>\!UL\                  ;


                                                              .PSECT  $CODE$,NOWRT,2

                                   0000 00000 DEFINE_SYM:
                                                              .WORD   Save nothing             ; 0292
                           04   AC  DD 00002                  PUSHL   VALUE                    ; 0301
                         0000'  CF  9F 00005                  PUSHAB  P.AAC                    ;
              0000V  CF         02  FB 00009                  CALLS   #2, FAO_BUFFER           ;
                                50  DD 0000E                  PUSHL   R0                       ;
                         0000'  CF  9F 00010                  PUSHAB  P.AAA                    ; 0300
          00000000G  00         02  FB 00014                  CALLS   #2, LIB$SET_SYMBOL       ;
                     09         50  E8 0001B                  BLBS    STATUS, 1$               ; 0302
                                50  DD 0001E                  PUSHL   STATUS                   ; 0303
          00000000G  00         01  FB 00020                  CALLS   #1, LIB$SIGNAL           ;
                                    04 00027 1$:              RET                              ; 0306

; Routine Size:  40 bytes,    Routine Base:  $CODE$ + 0066
```

```
  1 %sbttl 'Process CHECKSUM/FILE'
  1 ROUTINE checksum_file : NOVALUE =
  2 BEGIN
  2 !
  2 ! Checksum the contents of a file.  Open the file using RMS and simply read
  2 ! it with $GET's and checksumming the records.  Close the file.
  2 !
  2 LOCAL
  2     checksum,
  2     n_longwords,
  2     n_bytes,
  2     status;
  2
  2 !
  2 ! Open the input file to checksum using RMS.
  2 !
  2 $XABFHC_INIT(xab=ixabfhc);                     !Initialize XAB
P 2 $RAB_INIT(rab=irab,                            !Initialize input rab
  2         fab=.find_context);
  2
  3 IF NOT (status = $open(fab=.find_context))     !Open the input file
  2 THEN SIGNAL(cksm$_openin, 1,
  2                 find_result, .status,
  2                 .find_context[fab$l_stv])
  2
  3 ELSE IF NOT (status=$connect(rab=irab))        !Connect record stream
  3 THEN BEGIN
  3     SIGNAL(cksm$_openin, 1,
  3                 find_result, .status,
  3                 .irab[rab$l_stv]);
  3     return;
  2     END;
  2
  2 allocate_ubf(ixabfhc,irab);                    !Allocate the buffer to read into
  2
  2 !
  2 ! Checksum the contents.
  2 !
  2 checksum = 0;
  2 WHILE (status = $GET(rab=irab)) NEQ rms$_eof
  3 DO BEGIN
  3     BIND
  3         longbuffer = .irab[rab$l_rbf] : VECTOR[,LONG];
  3
  3     IF NOT .status
  3         THEN SIGNAL(cksm$_readerr,1,find_result,
  3                     .status,.irab[rab$l_stv]);
  3     !
  3     ! Compute number of full longwords and dangling bytes in record
  3     !
  3     n_longwords = (.irab[rab$w_rsz]/4);
  3     n_bytes = .irab[rab$w_rsz] AND 3;
  3     !
  3     ! Checksum the longwords by exclusive or.  Checksum the dangling
  3     ! bytes by simple addition
  3     !
  3     IF .n_longwords NEQ 0
```

```
;  3              THEN INCRU i FROM 0 to .n_longwords - 1
;  3                  DO checksum = .checksum XOR .longbuffer[.i];
;  4          IF .n_bytes NEQ 0
;  4          THEN BEGIN
;  4              BIND
;  4                  bytebuffer = longbuffer[.n_longwords] : VECTOR[,BYTE];
;  4
;  4              INCRU i FROM 0 TO .n_bytes - 1
;  4                  DO checksum = .checksum + .bytebuffer[.i];
;  3              END;
;  2          END;
;  2      !
;  2      ! Define a DCL string symbol with the decimal value of the checksum
;  2      !
;  2      define_sym(.checksum);
;  2
;  2      !
;  2      ! Disconnect the stream and close the input checksummed file.
;  2      !
;  3      IF NOT (status=$DISCONNECT(rab=irab))    !Disconnect and close output file
;  2          THEN SIGNAL(cksm$_closein, 1,
;  2                      find_result, .status,
;  2                      .irab[rab$l_stv]);
;  2
;  3      IF NOT (status=$CLOSE(fab=.find_context))
;  2          THEN SIGNAL(cksm$_closein, T,
;  2                      find_result, .status,
;  2                      .find_context[fab$l_stv]);
;  2
;  2      RETURN;
;  1      END;


                                                    $RMS_PTR=           IXABFHC
                                                    $RMS_PTR=           IRAB
                                                            .EXTRN  SYS$OPEN, SYS$CONNECT
                                                            .EXTRN  SYS$GET, SYS$DISCONNECT
                                                            .EXTRN  SYS$CLOSE

                                        03FC 00000 CHECKSUM_FILE:
                                                            .WORD   Save R2,R3,R4,R5,R6,R7,R8,R9                  ; 0308
                      59 00000000G  00   9E 00002           MOVAB   LIB$SIGNAL, R9                                ;
                      58     0000'  CF   9E 00009           MOVAB   $RMS_PTR, R8                                  ;
     2C         00    6E            00   2C 0000E           MOVC5   #0, (SP), #0, #44, $RMS_PTR                   ; 0323
                             0088   C8      00013                                                                 ;
                0088  C8     2C1D   8F   B0 00016           MOVW    #11293, $RMS_PTR                              ;
 0044    8F     00    6E            00   2C 0001D           MOVC5   #0, (SP), #0, #68, $RMS_PTR                   ; 0325
                                    68      00024                                                                 ;
                      68     4401   8F   B0 00025           MOVW    #17409, $RMS_PTR                              ;
                  3C  A8       FC   A8   D0 0002A           MOVL    FIND_CONTEXT, $RMS_PTR+60                     ;
                               FC   A8   DD 0002F           PUSHL   FIND_CONTEXT                                  ; 0327
           00000000G  00            01   FB 00032           CALLS   #1, SYS$OPEN                                  ;
                      54            50   D0 00039           MOVL    R0, STATUS                                    ;
                      1A            54   E8 0003C           BLBS    STATUS, 1$                                    ;
                      50       FC   A8   D0 0003F           MOVL    FIND_CONTEXT, R0                              ; 0330
                               0C   A0   DD 00043           PUSHL   12(R0)                                        ;
```

```
                                 54   DD 00046       PUSHL    STATUS                               ; 0329
                       00BC      C8   9F 00048       PUSHAB   FIND_RESULT                          ; 0328
                                 01   DD 0004C       PUSHL    #1                                   ;
                   0003109A      8F   DD 0004E       PUSHL    #200858                              ;
             69                  05   FB 00054       CALLS    #5, LIB$SIGNAL                       ;
                                 23   11 00057       BRB      2$                                   ;
                                 58   DD 00059 1$:   PUSHL    R8                                   ; 0332
  00000000G  00                  01   FB 0005B       CALLS    #1, SYS$CONNECT                      ;
             54                  50   D0 00062       MOVL     R0, STATUS                           ;
             14                  54   E8 00065       BLBS     STATUS, 2$                           ;
                         0C      A8   DD 00068       PUSHL    IRAB+12                              ; 0336
                                 54   DD 0006B       PUSHL    STATUS                               ; 0335
                       00BC      C8   9F 0006D       PUSHAB   FIND_RESULT                          ; 0334
                                 01   DD 00071       PUSHL    #1                                   ;
                   0003109A      8F   DD 00073       PUSHL    #200858                              ;
                             00CE     31 00079       BRW      12$                                  ;
                                 58   DD 0007C 2$:   PUSHL    R8                                   ; 0340
                       0088      C8   9F 0007E       PUSHAB   IXABFHC                              ;
            FEEB   CF            02   FB 00082       CALLS    #2, ALLOCATE_UBF                     ;
                                 56   D4 00087       CLRL     CHECKSUM                             ; 0345
                                 58   DD 00089 3$:   PUSHL    R8                                   ; 0346
  00000000G  00                  01   FB 0008B       CALLS    #1, SYS$GET                          ;
             54                  50   D0 00092       MOVL     R0, STATUS                           ;
  0001827A   8F                  54   D1 00095       CMPL     STATUS, #98938                       ;
                                 5D   13 0009C       BEQL     10$                                  ;
             55          28      A8   D0 0009E       MOVL     IRAB+40, R5                          ; 0349
             14                  54   E8 000A2       BLBS     STATUS, 4$                           ; 0351
                         0C      A8   DD 000A5       PUSHL    IRAB+12                              ; 0353
                                 54   DD 000A8       PUSHL    STATUS                               ;
                       00BC      C8   9F 000AA       PUSHAB   FIND_RESULT                          ; 0352
                                 01   DD 000AE       PUSHL    #1                                   ;
                   000310B2      8F   DD 000B0       PUSHL    #200882                              ;
             69                  05   FB 000B6       CALLS    #5, LIB$SIGNAL                       ;
             52          22      A8   3C 000B9 4$:   MOVZWL   IRAB+34, N_LONGWORDS                 ; 0357
             52                  04   C6 000BD       DIVL2    #4, N_LONGWORDS                      ;
  53     22   A8         02      00   EF 000C0       EXTZV    #0, #2, IRAB+34, N_BYTES             ; 0358
                                 52   D5 000C6       TSTL     N_LONGWORDS                          ; 0363
                                 13   13 000C8       BEQL     7$                                   ;
             51          FF      A2   9E 000CA       MOVAB    -1(R2), R1                           ; 0364
                                 50   D4 000CE       CLRL     I                                    ;
                                 06   11 000D0       BRB      6$                                   ;
             56                6540   CC 000D2 5$:   XORL2    (R5)[I], CHECKSUM                    ; 0365
                                 50   D6 000D6       INCL     I                                    ;
             51                  50   D1 000D8 6$:   CMPL     I, R1                                ;
                                 F5   1B 000DB       BLEQU    5$                                   ;
                                 53   D5 000DD 7$:   TSTL     N_BYTES                              ; 0366
                                 A8   13 000DF       BEQL     3$                                   ;
             50                6542   DE 000E1       MOVAL    (R5)[N_LONGWORDS], R0                ; 0369
             55          FF      A3   9E 000E5       MOVAB    -1(R3), R5                           ; 0371
                                 51   D4 000E9       CLRL     I                                    ;
                                 07   11 000EB       BRB      9$                                   ;
             57                8140   9A 000ED 8$:   MOVZBL   (I)+[R0], R7                         ; 0372
             56                  57   C0 000F1       ADDL2    R7, CHECKSUM                         ;
             55                  51   D1 000F4 9$:   CMPL     I, R5                                ;
                                 F4   1B 000F7       BLEQU    8$                                   ;
                                 8E   11 000F9       BRB      3$                                   ; 0346
                                 56   DD 000FB 10$:  PUSHL    CHECKSUM                             ; 0378
```

```
                FED6  CF           01  FB 000FD          CALLS   #1, DEFINE_SYM                         :
                                   58  DD 00102          PUSHL   R8                                     : 0383
           00000000G  00           01  FB 00104          CALLS   #1, SYS$DISCONNECT                     :
                      54           50  D0 0010B          MOVL    R0, STATUS                             :
                      14           54  E8 0010E          BLBS    STATUS, 11$                            :
                            0C     A8  DD 00111          PUSHL   IRAB+12                                : 0386
                                   54  DD 00114          PUSHL   STATUS                                 : 0385
                          00BC     C8  9F 00116          PUSHAB  FIND_RESULT                            : 0384
                                   01  DD 0011A          PUSHL   #1                                     :
                      00031050     8F  DD 0011C          PUSHL   #200784                                :
                      69           05  FB 00122          CALLS   #5, LIB$SIGNAL                         :
                            FC     A8  DD 00125  11$:    PUSHL   FIND_CONTEXT                           : 0388
           00000000G  00           01  FB 00128          CALLS   #1, SYS$CLOSE                          :
                      54           50  D0 0012F          MOVL    R0, STATUS                             :
                      18           54  E8 00132          BLBS    STATUS, 13$                            :
                      50    FC     A8  D0 00135          MOVL    FIND_CONTEXT, R0                       : 0391
                            0C     A0  DD 00139          PUSHL   12(R0)                                 :
                                   54  DD 0013C          PUSHL   STATUS                                 : 0390
                          00BC     C8  9F 0013E          PUSHAB  FIND_RESULT                            : 0389
                                   01  DD 00142          PUSHL   #1                                     :
                      00031050     8F  DD 00144          PUSHL   #200784                                :
                      69           05  FB 0014A  12$:    CALLS   #5, LIB$SIGNAL                         :
                                       04 0014D  13$:    RET                                            : 0394

; Routine Size:  334 bytes,    Routine Base:  $CODE$ + 008E
```

```
1 %sbttl 'signal handler'
1 ROUTINE signal_handler (sigargs, mechargs) =
2 BEGIN
2 !
2 ! This routine is a condition handler called when ever a
2 ! SIGNAL is done by cksm.  It merely remembers the
2 ! most severe error for an exit status.
2 !
2 MAP
2     sigargs : REF $BBLOCK,
2     mechargs : REF $BBLOCK;
2
2 BIND
2     signame = sigargs [chf$l_sig_name] : $BBLOCK;          !Name of signal
2
2 IF NOT .signame                                            !If its an error signal
4     AND ((.signame [sts$v_severity]                        ! and severity is worse than it was
4         GEQU .exitstatus [sts$v_severity])
3         OR .exitstatus [sts$v_severity])                   ! or we haven't had any errors
2     THEN exitstatus = .signame;                            ! then remember it for exiting
2
2 RETURN ss$_resignal;                                       !Resignal to get the error printed
1 END;                                                       !Of signal_handler
```

```
                                        0004 00000 SIGNAL_HANDLER:
                                                          .WORD   Save R2                        ; 0396
                      52      0000'  CF   9E 00002        MOVAB   EXITSTATUS, R2                 ;
            50    04  AC             04   C1 00007        ADDL3   #4, SIGARGS, R0                ; 0408
                      12             60   E8 0000C        BLBS    (R0), 2$                       ; 0410
      51    62        03             00   EF 0000F        EXTZV   #0, #3, EXITSTATUS, R1         ; 0412
      51    60        03             00   ED 00014        CMPZV   #0, #3, (R0), R1               ;
                                     03   1E 00019        BGEQU   1$                             ;
                      03             62   E9 0001B        BLBC    EXITSTATUS, 2$                 ; 0413
                      62             60   D0 0001E 1$:    MOVL    (R0), EXITSTATUS               ; 0414
                      50      0918   8F   3C 00021 2$:    MOVZWL  #2328, R0                      ; 0416
                                          04 00026        RET                                    ; 0417
```

; Routine Size: 39 bytes,   Routine Base: $CODE$ + 01DC

```
    1 %sbttl 'main routine of checksum'
    1 ROUTINE checksum$main =
    2 BEGIN
    2 !
    2 ! This is the main program.  It gathers all the command inputs and calls
    2 ! appropiate routine to checksum either the file or image.
    2 !
    2 LOCAL
    2       status,
    2       file_sum,
    2       ofab : $BBLOCK[fab$c_bln],
    2       onam : $BBLOCK[nam$c_bln],
    2       orss : $BBLOCK[nam$c_maxrss],
    2       oess : $BBLOCK[nam$c_maxrss],
    2       indesc : $BBLOCK [dsc$c_s_bln];                    !User's input file spec
    2
    2 ENABLE signal_handler;                                   !Enable the condition handler
    2 exitstatus = ss$_normal;                                 !Preset success exit status
    2
    2 cksm$gl_maxbyt = maxbytes;
    2 $init_ddesc(indesc);                                     !Initialize dynamic descriptor
    2 CH$MOVE(dsc$c_s_bln,indesc,outfiledesc);
    2 CH$MOVE(dsc$c_s_bln,indesc,find_result);
    2
    2 cli$get_value($descriptor('INPUT'),indesc);      !Get input file spec.
    2 cli$get_value($descriptor('OUTPUT'),outfiledesc);!Get user output file spec
    2
    2 file_sum = cli$present($descriptor('FILE'));     !See if /FILE
    2 status = cli$present($descriptor('IMAGE'));      !See if /IMAGE
    3 IF NOT (.file_sum OR .status)                    !If neither were specified
    2     THEN file_sum = true;                        ! then default to /FILE
    2 IF .file_sum                                     !If both were specified
    2     AND .status
    2     THEN SIGNAL_STOP(cksm$_confqual);            ! that's an error
    2 !
    2 ! Create the output file
    2 !
P   2 $FAB_INIT(fab=ofab,                              !Initialize output fab
P   2         nam=onam,
P   2         fns=.outfiledesc[dsc$w_length],
P   2         fna=.outfiledesc[dsc$a_pointer],
P   2         rat=(cr),                                !With carriage control
P   2         fop=(ofp),                               !Output file parse
    2         fac=(put));                              !Will do put's to file
    2
    2 CH$FILL(0,nam$c_maxrss,orss);                    !Zero resultant name string
    2 CH$FILL(0,nam$c_maxrss,oess);                    ! and expanded name string
P   2 $NAM_INIT(nam=onam,                              !Initialize output file nam block
P   2         rss=nam$c_maxrss,
P   2         rsa=orss,
P   2         ess=nam$c_maxrss,
    2         esa=oess);
    2
P   2 $RAB_INIT(rab=orab,                              !Initialize output file rab
    2         fab=ofab);
    2
    2 status = $CREATE(fab=ofab);
```

```
2 CH$MOVE(dsc$c_s_bln,getfilename(ofab),          !Get the resulting filename
2                           outfiledesc);
2 IF NOT .status                                  !If error opening output file
2     THEN SIGNAL_STOP(cksm$_openout,1,outfiledesc, ! then give up now
2                 .status,.ofab[fab$l_stv]);
2
3 IF NOT (status=$CONNECT(rab=orab))              !Connect the record stream
2     THEN SIGNAL_STOP(cksm$_openout,1,outfiledesc,
2                 .status,.orab[rab$l_stv]);
2 !
2 ! Loop, calling lib$find_file to get files matching the output spec
2 !
2 find_context = 0;                               !Initial context is 0
2
3 WHILE (status = lib$find_file(indesc,find_result,find_context,
4                         (IF .file_sum
4                             THEN $descriptor('.DAT')
3                             ELSE $descriptor('.EXE')),
2                         0, 0, %REF(2)) ) NEQ rms$_nmf
2 DO
2 IF NOT .status                                  !Report error if find_file failure
2         THEN SIGNAL(cksm$_parsefail,1,find_result,
2                 .find_context[fab$l_sts],.find_context[fab$l_stv])
3 ELSE BEGIN
3     find_context[fab$v_get] = true;             !Initialize for GET's
3     IF NOT .file_sum
3         THEN find_context[fab$v_bio] = true;    ! and block i/o
3
3 !
3 ! Call the appropiate routine to do the checksumming of the file contents
3 ! or image.
3 !
3     IF .file_sum                                !If /FILE then checksum the file
4         THEN BEGIN                              ! by simply reading it
4             find_context[fab$l_xab] = ixabfhc;  !Set XAB address
4             checksum_file();                    ! Output is DCL symbol checksum$checksum
4             END
4         ELSE BEGIN
5             IF NOT (status = lib$get_vm(cksm$gl_maxbyt,
5                                 databuffer))
4                 THEN SIGNAL_STOP(.status);
4             find_context[fab$l_xab] = 0; !No need to use XAB
4             read_header();              !Process CHECKSUM/IMAGE
3             END;
3
2     END;                                        !Of loop calling lib$find_file
2 !
2 ! Close the output file
2 !
3 IF NOT (status=$DISCONNECT(rab=orab))
2     THEN SIGNAL(cksm$_closeout, 1,
2                 outfiledesc, .status,
2                 .orab[rab$l_stv]);
2
3 IF NOT (status=$CLOSE(fab=ofab))
2     THEN SIGNAL(cksm$_closeout, 1,
2                 outfiledesc, .status,
```

```
; 453      0532  2                       .ofab[fab$l_stv]);
; 454      0533  2
; 455      0534  2 RETURN (status = .exitstatus OR sts$m_inhib_msg);       !Exit with no message
; 456      0535  1 END;


                                                        .PSECT  $PLIT$,NOWRT,NOEXE,2

                 54  55  50  4E  49  00020 P.AAE:   .ASCII  \INPUT\                              ;
                                     00025          .BLKB   3
                          00000005   00028 P.AAD:   .LONG   5                                    ;
                          00000000'  0002C          .ADDRESS P.AAE                               ;
             54  55  50  54  55  4F  00030 P.AAG:   .ASCII  \OUTPUT\                             ;
                                     00036          .BLKB   2                                    ;
                          00000006   00038 P.AAF:   .LONG   6                                    ;
                          00000000'  0003C          .ADDRESS P.AAG                               ;
                     45  4C  49  46  00040 P.AAI:   .ASCII  \FILE\                               ;
                          00000004   00044 P.AAH:   .LONG   4                                    ;
                          00000000'  00048          .ADDRESS P.AAI                               ;
                 45  47  41  4D  49  0004C P.AAK:   .ASCII  \IMAGE\                              ;
                                     00051          .BLKB   3
                          00000005   00054 P.AAJ:   .LONG   5                                    ;
                          00000000'  00058          .ADDRESS P.AAK                               ;
                     54  41  44  2E  0005C P.AAM:   .ASCII  \.DAT\                               ;
                          00000004   00060 P.AAL:   .LONG   4                                    ;
                          00000000'  00064          .ADDRESS P.AAM                               ;
                     45  58  45  2E  00068 P.AAO:   .ASCII  \.EXE\                               ;
                          00000004   0006C P.AAN:   .LONG   4                                    ;
                          00000000'  00070          .ADDRESS P.AAO                               ;

                                           $RMS_PTR=        ORAB
                                                    .EXTRN  SYS$CREATE

                                                    .PSECT  $CODE$,NOWRT,2

                                OFFC 00000 CHECKSUM$MAIN:
                                                    .WORD   Save R2,R3,R4,R5,R6,R7,R8,R9,R10,R11        ; 0419
               5B 00000000G   00   9E 00002         MOVAB   LIB$SIGNAL, R11                             ;
               5A     0000'   CF   9E 00009         MOVAB   P.AAD, R10                                  ;
               59 00000000G   00   9E 0000E         MOVAB   LIB$STOP, R9                                ;
               58     0000'   CF   9E 00015         MOVAB   OUTFILEDESC, R8                             ;
               5E     FD44    CE   9E 0001A         MOVAB   -700(SP), SP                                ;
               6D     0221    CF   DE 0001F         MOVAL   17$, (FP)                                   ; 0420
          18   A8             01   D0 00024         MOVL    #1, EXITSTATUS                              ; 0435
       0000'   CF     1400    8F   3C 00028         MOVZWL  #5120, CKSM$GL_MAXBYT                       ; 0437
          04   AE 020E0000    8F   D0 0002F         MOVL    #34471936, INDESC                           ; 0438
                         08   AE   D4 00037         CLRL    INDESC+4                                    ;
      68       04   AE        08   28 0003A         MOVC3   #8, INDESC, OUTFILEDESC                     ; 0439
   08 A8       04   AE        08   28 0003F         MOVC3   #8, INDESC, FIND_RESULT                     ; 0440
                         04   AE   9F 00045         PUSHAB  INDESC                                      ; 0442
                              5A   DD 00048         PUSHL   R10                                         ;
        00000000G  00         02   FB 0004A         CALLS   #2, CLI$GET_VALUE                           ;
                              58   DD 00051         PUSHL   R8                                          ; 0443
                         10   AA   9F 00053         PUSHAB  P.AAF                                       ;
        00000000G  00         02   FB 00056         CALLS   #2, CLI$GET_VALUE                           ;
                         1C   AA   9F 0005D         PUSHAB  P.AAH                                       ; 0445
```

```
                    00000000G  00            01  FB 00060        CALLS   #1, CLI$PRESENT                       ;
                               57            50  D0 00067        MOVL    R0, FILE_SUM                          ;
                                        2C   AA  9F 0006A        PUSHAB  P.AAJ                                 ; 0446
                    00000000G  00            01  FB 0006D        CALLS   #1, CLI$PRESENT                       ;
                               56            50  D0 00074        MOVL    R0, STATUS                            ;
                               09            57  E8 00077        BLBS    FILE_SUM, 2$                          ; 0447
                               03            56  E8 0007A        BLBS    STATUS, 1$                            ;
                               57            01  D0 0007D        MOVL    #1, FILE_SUM                          ; 0448
                               0C            57  E9 00080 1$:    BLBC    FILE_SUM, 3$                          ; 0449
                               09            56  E9 00083 2$:    BLBC    STATUS, 3$                            ; 0450
                                  000312E4   8F  DD 00086        PUSHL   #201444                               ; 0451
                               69            01  FB 0008C        CALLS   #1, LIB$STOP                          ;
 0050    8F            00      6E            00  2C 0008F 3$:    MOVC5   #0, (SP), #0, #80, $RMS_PTR           ; 0461
                                        B0   AD     00096                                                  ;
                   B0          AD     5003   8F  B0 00098        MOVW    #20483, $RMS_PTR                      ;
                   B4          AD 20000000   8F  D0 0009E        MOVL    #536870912, $RMS_PTR+4                ;
                   C6          AD            01  90 000A6        MOVB    #1, $RMS_PTR+22                       ;
                   CE          AD     0202   8F  B0 000AA        MOVW    #514, $RMS_PTR+30                     ;
                   D8          AD     FF50   CD  9E 000B0        MOVAB   ONAM, $RMS_PTR+40                     ;
                   DC          AD       04   A8  D0 000B6        MOVL    OUTFILEDESC+4, $RMS_PTR+44            ;
                   E4          AD            68  90 000BB        MOVB    OUTFILEDESC, $RMS_PTR+52              ;
 00FF    8F            00      6E            00  2C 000BF        MOVC5   #0, (SP), #0, #255, ORSS              ; 0463
                                      010C   CE     000C6                                                  ;
 00FF    8F            00      6E            00  2C 000C9        MOVC5   #0, (SP), #0, #255, OESS              ; 0464
                                        0C   AE     000D0                                                  ;
 0060    8F            00      6E            00  2C 000D2        MOVC5   #0, (SP), #0, #96, $RMS_PTR           ; 0469
                                      FF50   CD     000D9                                                  ;
                 FF50          CD     6002   8F  B0 000DC        MOVW    #24578, $RMS_PTR                      ;
                 FF52          CD            01  8E 000E3        MNEGB   #1, $RMS_PTR+2                        ;
                 FF54          CD     010C   CE  9E 000E8        MOVAB   ORSS, $RMS_PTR+4                      ;
                 FF5A          CD            01  8E 000EF        MNEGB   #1, $RMS_PTR+10                       ;
                 FF5C          CD       0C   AE  9E 000F4        MOVAB   OESS, $RMS_PTR+12                     ;
 0044    8F            00      6E            00  2C 000FA        MOVC5   #0, (SP), #0, #68, $RMS_PTR           ; 0472
                                        90   A8     00101                                                  ;
                   90          A8     4401   8F  B0 00103        MOVW    #17409, $RMS_PTR                      ;
                   CC          A8       B0   AD  9E 00109        MOVAB   OFAB, $RMS_PTR+60                     ;
                                        B0   AD  9F 0010E        PUSHAB  OFAB                                  ; 0474
                    00000000G  00            01  FB 00111        CALLS   #1, SYS$CREATE                        ;
                               56            50  D0 00118        MOVL    R0, STATUS                            ;
                                        B0   AD  9F 0011B        PUSHAB  OFAB                                  ; 0475
                        0000V  CF            01  FB 0011E        CALLS   #1, GETFILENAME                       ;
                       68      60            08  28 00123        MOVC3   #8, (R0), OUTFILEDESC                 ;
                               12            56  E8 00127        BLBS    STATUS, 4$                            ; 0477
                                        BC   AD  DD 0012A        PUSHL   OFAB+12                               ; 0479
                                             56  DD 0012D        PUSHL   STATUS                                ;
                                             58  DD 0012F        PUSHL   R8                                    ; 0478
                                             01  DD 00131        PUSHL   #1                                    ;
                                  000310A4   8F  DD 00133        PUSHL   #200868                               ;
                               69            05  FB 00139        CALLS   #5, LIB$STOP                          ;
                                        90   A8  9F 0013C 4$:    PUSHAB  ORAB                                  ; 0481
                    00000000G  00            01  FB 0013F        CALLS   #1, SYS$CONNECT                       ;
                               56            50  D0 00146        MOVL    R0, STATUS                            ;
                               12            56  E8 00149        BLBS    STATUS, 5$                            ;
                                        9C   A8  DD 0014C        PUSHL   ORAB+12                               ; 0483
                                             56  DD 0014F        PUSHL   STATUS                                ;
                                             58  DD 00151        PUSHL   R8                                    ; 0482
                                             01  DD 00153        PUSHL   #1                                    ;
```

```
                000310A4  8F  DD 00155          PUSHL   #200868                         ;
                    69    05  FB 0015B          CALLS   #5, LIB$STOP                    ;
                    FF48  C8  D4 0015E  5$:     CLRL    FIND_CONTEXT                    ; 0487
                6E        02  D0 00162  6$:     MOVL    #2, (SP)                        ; 0493
                          5E  DD 00165          PUSHL   SP                              ;
                          7E  7C 00167          CLRQ    -(SP)                           ; 0489
                06        57  E9 00169          BLBC    FILE_SUM, 7$                    ; 0490
                50    38  AA  9E 0016C          MOVAB   P.AAL, R0                       ; 0491
                          04  11 00170          BRB     8$                              ;
                50    44  AA  9E 00172  7$:     MOVAB   P.AAN, R0                       ; 0492
                          50  DD 00176  8$:     PUSHL   R0                              ;
                    FF48  C8  9F 00178          PUSHAB  FIND_CONTEXT                    ; 0489
                      08  A8  9F 0017C          PUSHAB  FIND_RESULT                     ;
                      1C  AE  9F 0017F          PUSHAB  INDESC                          ;
      00000000G 00        07  FB 00182          CALLS   #7, LIB$FIND_FILE               ;
                56        50  D0 00189          MOVL    R0, STATUS                      ;
      000182CA  8F        56  D1 0018C          CMPL    STATUS, #99018                  ; 0493
                5E        13 00193              BEQL    14$                             ;
                52  FF48  C8  D0 00195          MOVL    FIND_CONTEXT, R2                ; 0497
                14        56  E8 0019A          BLBS    STATUS, 10$                     ; 0495
                7E    08  A2  7D 0019D          MOVQ    8(R2), -(SP)                    ; 0497
                      08  A8  9F 001A1          PUSHAB  FIND_RESULT                     ; 0496
                          01  DD 001A4          PUSHL   #1                              ;
                00031248  8F  DD 001A6          PUSHL   #201288                         ;
                6B        05  FB 001AC          CALLS   #5, LIB$SIGNAL                  ;
                          B1  11 001AF  9$:     BRB     6$                              ;
           16   A2        02  88 001B1  10$:    BISB2   #2, 22(R2)                      ; 0499
                07        57  E8 001B5          BLBS    FILE_SUM, 11$                   ; 0500
           16   A2        20  88 001B8          BISB2   #32, 22(R2)                     ; 0501
                0C        57  E9 001BC          BLBC    FILE_SUM, 12$                   ; 0507
           24   A2    D4  A8  9E 001BF  11$:    MOVAB   IXABFHC, 36(R2)                 ; 0509
         FCC2   CF        00  FB 001C4          CALLS   #0, CHECKSUM_FILE               ; 0510
                          97  11 001C9          BRB     6$                              ; 0507
                      14  A8  9F 001CB  12$:    PUSHAB  DATABUFFER                      ; 0513
                   0000'  CF  9F 001CE          PUSHAB  CKSM$GL_MAXBYT                  ;
      00000000G 00        02  FB 001D2          CALLS   #2, LIB$GET_VM                  ;
                56        50  D0 001D9          MOVL    R0, STATUS                      ;
                05        56  E8 001DC          BLBS    STATUS, 13$                     ;
                          56  DD 001DF          PUSHL   STATUS                          ; 0515
                69        01  FB 001E1          CALLS   #1, LIB$STOP                    ;
                50  FF48  C8  D0 001E4  13$:    MOVL    FIND_CONTEXT, R0                ; 0516
                      24  A0  D4 001E9          CLRL    36(R0)                          ;
        0000V   CF        00  FB 001EC          CALLS   #0, READ_HEADER                 ; 0517
                          BC  11 001F1          BRB     9$                              ; 0495
                      90  A8  9F 001F3  14$:    PUSHAB  ORAB                            ; 0524
      00000000G 00        01  FB 001F6          CALLS   #1, SYS$DISCONNECT              ;
                56        50  D0 001FD          MOVL    R0, STATUS                      ;
                12        56  E8 00200          BLBS    STATUS, 15$                     ;
                      9C  A8  DD 00203          PUSHL   ORAB+12                         ; 0527
                          56  DD 00206          PUSHL   STATUS                          ; 0526
                          58  DD 00208          PUSHL   R8                              ; 0525
                          01  DD 0020A          PUSHL   #1                              ;
                00031058  8F  DD 0020C          PUSHL   #200792                         ;
                6B        05  FB 00212          CALLS   #5, LIB$SIGNAL                  ;
                      B0  AD  9F 00215  15$:    PUSHAB  OFAB                            ; 0529
      00000000G 00        01  FB 00218          CALLS   #1, SYS$CLOSE                   ;
                56        50  D0 0021F          MOVL    R0, STATUS                      ;
```

```
                        12          56  E8 00222          BLBS    STATUS, 16$                            ;
                             BC     AD  DD 00225          PUSHL   OFAB+12                                ; 0532
                                    56  DD 00228          PUSHL   STATUS                                 ; 0531
                                    58  DD 0022A          PUSHL   R8                                     ; 0530
                                    01  DD 0022C          PUSHL   #1                                     ;
                   00031058         8F  DD 0022E          PUSHL   #200792                                ;
                        6B          05  FB 00234          CALLS   #5, LIB$SIGNAL                         ;
      56          18    A8 10000000 8F  C9 00237 16$:     BISL3   #268435456, EXITSTATUS, STATUS         ; 0534
                        50          56  D0 00240          MOVL    STATUS, R0                             ;
                                        04 00243          RET                                            ; 0535
                                      0000 00244 17$:     .WORD   Save nothing                           ; 0420
                                    7E  D4 00246          CLRL    -(SP)                                  ;
                                    5E  DD 00248          PUSHL   SP                                     ;
                        7E   04     AC  7D 0024A          MOVQ    4(AP), -(SP)                           ;
                 FD86   CF          03  FB 0024E          CALLS   #3, SIGNAL_HANDLER                     ;
                                        04 00253          RET                                            ;

; Routine Size:  596 bytes,    Routine Base:  $CODE$ + 0203
```

```
  0536  1 %sbttl 'Catch signals'
  0537  1 ROUTINE catch_signals (sigargs,mechargs) =
  0538  2 BEGIN
  0539  2 !
  0540  2 ! This condition handler is enabled when the image is being
  0541  2 ! processed.  It catches info messages, and diverts them to
  0542  2 ! the output file.  Any other errors are resignaled.
  0543  2 !
  0544  2 MAP
  0545  2     sigargs : REF $BBLOCK,
  0546  2     mechargs : REF $BBLOCK;
  0547  2
  0548  2 BIND
  0549  2     signame = sigargs[chf$l_sig_name] : $BBLOCK;
  0550  2
  0551  2 LOCAL
  0552  2     status,
  0553  2     outadr : $BBLOCK[4],
  0554  2     outbufdesc : $BBLOCK[dsc$c_s_bln],
  0555  2     msgbufdesc : $BBLOCK[dsc$c_s_bln],
  0556  2     msgbuf : $BBLOCK[256],
  0557  2     outbuf : $BBLOCK[512];
  0558  2
  0559  2 IF NOT .signame                                          !If this is not an informational message
  0560  2     THEN RETURN ss$_resignal;                            ! then we don't want it to go to output file
  0561  2 msgbufdesc[dsc$w_length] = 256;                          !Initialize the descriptors
  0562  2 msgbufdesc[dsc$a_pointer] = msgbuf;
  0563  2 outbufdesc[dsc$w_length] = 512;
  0564  2 outbufdesc[dsc$a_pointer] = outbuf;
  0565  2 !
  0566  2 ! Get the message and format the string
  0567  2 !
P 0568  3 IF NOT (status = $GETMSG(msgid=.signame,msglen=msgbufdesc[dsc$w_length],
  0569  3                 bufadr=msgbufdesc,flags=1,outadr=outadr))
  0570  2     THEN RETURN ss$_resignal                             !Not found, so resignal
P 0571  3 ELSE IF NOT (status =$faol(ctrstr=msgbufdesc,outlen=outbufdesc,   !Format the string
  0572  3             outbuf=outbufdesc,prmlst=sigargs[chf$l_sig_arg1]+4))  ! (skip the signal count longword)
  0573  2     THEN RETURN ss$_resignal                             !FAO error, so resignal
  0574  2     ELSE cksm_output(outbufdesc);                        !All is well, write string to output
  0575  2
  0576  2 signame[sts$v_inhib_msg] = true;                         !Inhibit further message printing
  0577  2 mechargs[chf$l_mch_savr0] = .signame[sts$v_inhib_msg];
  0578  2
  0579  2 RETURN ss$_continue
  0580  2
  0581  1 END;


                                                .EXTRN  SYS$GETMSG, SYS$FAOL

                               0004 00000 CATCH_SIGNALS:
                                          .WORD   Save R2                          ; 0537
              5E   FCEC   CE   9E 00002   MOVAB   -788(SP), SP                     ;
   52    04   AC          04   C1 00007   ADDL3   #4, SIGARGS, R2                  ; 0549
              45          62   E9 0000C   BLBC    (R2), 1$                         ; 0559
         F0   AD   0100   8F   B0 0000F   MOVW    #256, MSGBUFDESC                 ; 0561
```

```
                      F4   AD   FEF0   CD  9E 00015        MOVAB   MSGBUF, MSGBUFDESC+4                 ; 0562
                      F8   AD   0200   8F  B0 0001B        MOVW    #512, OUTBUFDESC                     ; 0563
                      FC   AD     04   AE  9E 00021        MOVAB   OUTBUF, OUTBUFDESC+4                 ; 0564
                                       5E  DD 00026        PUSHL   SP                                   ; 0569
                                       01  DD 00028        PUSHL   #1                                   ;
                                  F0   AD  9F 0002A        PUSHAB  MSGBUFDESC                           ;
                                  F0   AD  9F 0002D        PUSHAB  MSGBUFDESC                           ;
                                       62  DD 00030        PUSHL   (R2)                                 ;
                00000000G  00          05  FB 00032        CALLS   #5, SYS$GETMSG                       ;
                           18          50  E9 00039        BLBC    STATUS, 1$                           ;
              7E      04   AC          0C  C1 0003C        ADDL3   #12, SIGARGS, -(SP)                  ; 0572
                                  F8   AD  9F 00041        PUSHAB  OUTBUFDESC                           ;
                                  F8   AD  9F 00044        PUSHAB  OUTBUFDESC                           ;
                                  F0   AD  9F 00047        PUSHAB  MSGBUFDESC                           ;
                00000000G  00          04  FB 0004A        CALLS   #4, SYS$FAOL                         ;
                           06          50  E8 00051        BLBS    STATUS, 2$                           ;
                           50   0918   8F  3C 00054  1$:   MOVZWL  #2328, R0                            ; 0573
                                           04 00059        RET                                          ;
                                  F8   AD  9F 0005A  2$:   PUSHAB  OUTBUFDESC                           ; 0574
                    0000V  CF          01  FB 0005D        CALLS   #1, CKSM_OUTPUT                      ;
                      03   A2          10  88 00062        BISB2   #16, 3(R2)                           ; 0576
                           50     08   AC  D0 00066        MOVL    MECHARGS, R0                         ; 0577
  0C   A0          62      01          1C  EF 0006A        EXTZV   #28, #1, (R2), 12(R0)                ;
                           50          01  D0 00070        MOVL    #1, R0                               ; 0579
                                           04 00073        RET                                          ; 0581

; Routine Size:  116 bytes,    Routine Base:  $CODE$ + 0457
```

```
1 %sbttl 'Process CHECKSUM/IMAGE'
1 ROUTINE read_header =
2 BEGIN
2 !
2 ! Open the file using RMS with the UFO (User File Open) bit set.  The
2 ! read the image header using the EXEC routines which do $QIO's given
2 ! the channel from the RMS open., and checksum the image sections.  Ignore
2 ! the parts of the image header that can always be different, (ex: date/time).
2 !
2 LOCAL
2         isdnum,                                   !isd number
2         header : $BBLOCK[512],                    !Buffer containing decoded header
2         curhdrisd : $BBLOCK[512],                 !current isd in header
2         hdr_blk : $BBLOCK[512],                   !buffer for image header -undecoded
2         status,
2         vbn,                                      !Used by header decode routines
2         offset,                                   !Used by header decode routines
2         hdrver,                                   !Used by header decode routines
2         alias ;                                   !Used by header decode routines
2
2 ENABLE catch_signals;                             !We want output to go to output file also
2 cksm$gl_ihdcksm = 0;                              !Zero image header checksum
2 isdnum = 0;                                       !first isect is 1
2 cksm$gl_acucksm = 0;                              !Zero accumulated checksum
2 SIGNAL(cksm$_file,1,find_result);                 !Print the filename
2
2 !
2 ! First, open the file using RMS with an UFO option.
2 !
2 find_context[fab$v_ufo] = true;                   !User file open option
3 IF NOT (status = $open(fab=.find_context))        !Open the input file
2 THEN SIGNAL(cksm$_openin, 1,
2               find_result, .status,
2               .find_context[fab$l_stv]);
2 find_context[fab$v_ufo] = false;                  !clear the bit after open
2 chan = .find_context[fab$l_stv];                  !Keep around the assigned channel
2
2 !
2 ! Read the image header into a buffer decoded by the special EXEC routines
2 !
2 status = img$decode_ihd( .chan,hdr_blk,header,vbn,offset,hdrver,alias);
2 IF NOT .status
2 THEN
2     SIGNAL( .status );                            !signal any errors in image file
2
2 !
2 ! The following checks are made on the image header fixed portion to see if legal
2 ! Major neader id must match.  The header fixed part must be less than a block and must
2 ! contain patch area.  header block count is > 0.
2 !
2 IF .header[ihd$w_majorid] NEQ ihd$k_majorid
4     OR (.(header[ihd$w_minorid])<0,8,0> EQL (ihd$k_minorid and %x'ff')
3         AND .(header[ihd$w_minorid])<8,8,0> GTRU (ihd$k_minorid and %x'ff00')/256)
3     OR .header[ihd$w_size] GTRU MAXU((.header[ihd$w_patchoff]
2                     + ihp$k_length),ihd$k_maxlength)
3     OR ( .header[ihd$b_hdrblkcnt] LEQ 0)
3     OR (header + .header[ihd$w_symdbgoff])
```

```
3                        GEQU (header + .header[ihd$w_size])
3       THEN BEGIN
3           SIGNAL(cksm$_badimghdr,1,find_result);  !any above not true, fatal image header error
3           status = $dassgn(chan=.chan);           !deassign channel
3           cksm_foreign();                         !Checksum non-image file
3           RETURN true
2           END;
2
2   ! Otherwise just checksum the contents of the fixed portion of the image
2   ! header
2   !
2   cksm_header(header);
2
2   !
2   ! Now, checksum all the ISD's (Image Section Descriptors)
2   !
2   WHILE 1
3   DO BEGIN
3       status = img$get_next_isd(.chan,hdr_blk,header,vbn,offset,curhdrisd);
3       IF .status EQLU IMG$_ENDOFHDR               ! Until no more ISD's....
3       THEN
3           EXITLOOP;
3       IF NOT .status
4       THEN BEGIN
4           SIGNAL(cksm$_badimghdr,1,find_result);  !signal error
4           status = $dassgn(chan=.chan);           !deassign channel
4           cksm_foreign();                         !and checksum as foreign image
4           RETURN true;
4           END
3       ELSE
4           BEGIN
4           isdnum = .isdnum + 1;
4           checksum_section(curhdrisd);            !compute isect checksum
4           IF .curhdrisd[isd$l_vbn] NEQ 0          !If section has disk file space
4               AND NOT .curhdrisd[isd$v_dzro]      ! and is not demand zero
5           THEN BEGIN
5               SIGNAL(cksm$_isectchk,2,.isdnum,.cksm$gl_chksum);   !Print isect checksum
5               cksm$gl_acucksm = .cksm$gl_acucksm XOR .cksm$gl_chksum;!Accumulate new improved checksum
4               END;
3           END;
2       END;                                        !end of isection loop
2
2   !
2   ! Print checksum of image header and total checksum, and define a DCL
2   ! symbol with the checksum value
2   !
2   SIGNAL(cksm$_ihdchksum,1,.cksm$gl_ihdcksm);
2   SIGNAL(cksm$_allischk,1,.cksm$gl_acucksm);
2   define_sym(.cksm$gl_acucksm);
2
3   IF NOT (status=$DASSGN(chan=.chan))
2       THEN SIGNAL(cksm$_closein, 1,
2                   find_result, .status);
2
2   RETURN true
1   END;
```

```
                                                          .EXTRN  SYS$DASSGN

                                  01FC 00000 READ_HEADER:
                                                          .WORD   Save R2,R3,R4,R5,R6,R7,R8                      ; 0583
                58 00000000G  00    9E 00002              MOVAB   SYS$DASSGN, R8                                 ;
                57     0000'  CF    9E 00009              MOVAB   CKSM$GL_ACUCKSM, R7                            ;
                56     0000'  CF    9E 0000E              MOVAB   CHAN, R6                                       ;
                55 00000000G  00    9E 00013              MOVAB   LIB$SIGNAL, R5                                 ;
                5E      F9F0  CE    9E 0001A              MOVAB   -1552(SP), SP                                  ;
                6D      0194  CF    DE 0001F              MOVAL   11$, (FP)                                      ; 0584
                        F8    A7    D4 00024              CLRL    CKSM$GL_IHDCKSM                                ; 0603
                              53    D4 00027              CLRL    ISDNUM                                         ; 0604
                              67    D4 00029              CLRL    CKSM$GL_ACUCKSM                                ; 0605
                      00C4    C6    9F 0002B              PUSHAB  FIND_RESULT                                    ; 0606
                              01    DD 0002F              PUSHL   #1                                             ;
                   00000000G  8F    DD 00031              PUSHL   #CKSM$_FILE                                    ;
                65            03    FB 00037              CALLS   #3, LIB$SIGNAL                                 ;
                50      04    A6    D0 0003A              MOVL    FIND_CONTEXT, R0                               ; 0611
        06      A0            02    88 0003E              BISB2   #2, 6(R0)                                      ;
                              50    DD 00042              PUSHL   R0                                             ; 0612
 00000000G      00            01    FB 00044              CALLS   #1, SYS$OPEN                                   ;
                52            50    D0 0004B              MOVL    R0, STATUS                                     ;
                18            52    E8 0004E              BLBS    STATUS, 1$                                     ;
                50      04    A6    D0 00051              MOVL    FIND_CONTEXT, R0                               ; 0615
                        0C    A0    DD 00055              PUSHL   12(R0)                                         ;
                              52    DD 00058              PUSHL   STATUS                                         ; 0614
                      00C4    C6    9F 0005A              PUSHAB  FIND_RESULT                                    ; 0613
                              01    DD 0005E              PUSHL   #1                                             ;
                   0003109A   8F    DD 00060              PUSHL   #200858                                        ;
                65            05    FB 00066              CALLS   #5, LIB$SIGNAL                                 ;
                50      04    A6    D0 00069 1$:          MOVL    FIND_CONTEXT, R0                               ; 0616
        06      A0            02    8A 0006D              BICB2   #2, 6(R0)                                      ;
                66      0C    A0    D0 00071              MOVL    12(R0), CHAN                                   ; 0617
                              5E    DD 00075              PUSHL   SP                                             ; 0622
                        08    AE    9F 00077              PUSHAB  HDRVER                                         ;
                        10    AE    9F 0007A              PUSHAB  OFFSET                                         ;
                        18    AE    9F 0007D              PUSHAB  VBN                                            ;
                      FE00    CD    9F 00080              PUSHAB  HEADER                                         ;
                        24    AE    9F 00084              PUSHAB  HDR_BLK                                        ;
                              66    DD 00087              PUSHL   CHAN                                           ;
 00000000G      00            07    FB 00089              CALLS   #7, IMG$DECODE_IHD                             ;
                52            50    D0 00090              MOVL    R0, STATUS                                     ;
                05            52    E8 00093              BLBS    STATUS, 2$                                     ; 0623
                              52    DD 00096              PUSHL   STATUS                                         ; 0625
                65            01    FB 00098              CALLS   #1, LIB$SIGNAL                                 ;
      3230      8F    FE0C    CD    B1 0009B 2$:          CMPW    HEADER+12, #12848                              ; 0632
                              03    13 000A2              BEQL    3$                                             ;
                            0083    31 000A4              BRW     7$                                             ;
                30    FE0E    CD    91 000A7 3$:          CMPB    HEADER+14, #48                                 ; 0633
                              07    12 000AC              BNEQ    4$                                             ;
                35    FE0F    CD    91 000AE              CMPB    HEADER+15, #53                                 ; 0634
                              75    1A 000B3              BGTRU   7$                                             ;
                50    FE08    CD    3C 000B5 4$:          MOVZWL  HEADER+8, R0                                   ; 0635
                50            2C    C0 000BA              ADDL2   #44, R0                                        ;
 000000A8       8F            50    D1 000BD              CMPL    R0, #168                                       ;
```

```
                                    04  1E 000C4       BGEQU   5$                                      ;
                        50    A8    8F  9A 000C6       MOVZBL  #168, R0                                ;
   50    FE00  CD       10          00  ED 000CA  5$:  CMPZV   #0, #16, HEADER, R0                     ;
                                    57  1A 000D1       BGTRU   7$                                      ;
                            FE10    CD  95 000D3       TSTB    HEADER+16                               ; 0637
                                    51  13 000D7       BEQL    7$                                      ;
                        51  FE00    CD  9E 000D9       MOVAB   HEADER, R1                              ; 0638
                        50  FE04    CD  3C 000DE       MOVZWL  HEADER+4, R0                            ;
                        51          50  C0 000E3       ADDL2   R0, R1                                  ;
                        50  FE00    CD  9E 000E6       MOVAB   HEADER, R0                              ; 0639
                        54  FE00    CD  3C 000EB       MOVZWL  HEADER, R4                              ;
                        50          54  C0 000F0       ADDL2   R4, R0                                  ;
                        50          51  D1 000F3       CMPL    R1, R0                                  ;
                                    32  1E 000F6       BGEQU   7$                                      ;
                            FE00    CD  9F 000F8       PUSHAB  HEADER                                  ; 0650
                  0000V   CF        01  FB 000FC       CALLS   #1, CKSM_HEADER                         ;
                            0210    CE  9F 00101  6$:  PUSHAB  CURHDRISD                               ; 0657
                              0C    AE  9F 00105       PUSHAB  OFFSET                                  ;
                              14    AE  9F 00108       PUSHAB  VBN                                     ;
                            FE00    CD  9F 0010B       PUSHAB  HEADER                                  ;
                              20    AE  9F 0010F       PUSHAB  HDR_BLK                                 ;
                                    66  DD 00112       PUSHL   CHAN                                    ;
              00000000G   00        06  FB 00114       CALLS   #6, IMG$GET_NEXT_ISD                    ;
                        52          50  D0 0011B       MOVL    R0, STATUS                              ;
               084D8640   8F        52  D1 0011E       CMPL    STATUS, #139298368                      ; 0658
                                    4E  13 00125       BEQL    9$                                      ;
                        1E          52  E8 00127       BLBS    STATUS, 8$                              ; 0661
                            00C4    C6  9F 0012A  7$:  PUSHAB  FIND_RESULT                             ; 0663
                                    01  DD 0012E       PUSHL   #1                                      ;
                       00000000G    8F  DD 00130       PUSHL   #CKSM$_BADIMGHDR                        ;
                        65          03  FB 00136       CALLS   #3, LIB$SIGNAL                          ;
                                    66  DD 00139       PUSHL   CHAN                                    ; 0664
                        68          01  FB 0013B       CALLS   #1, SYS$DASSGN                          ;
                        52          50  D0 0013E       MOVL    R0, STATUS                              ;
                  0000V   CF        00  FB 00141       CALLS   #0, CKSM_FOREIGN                        ; 0665
                                    6B  11 00146       BRB     10$                                     ; 0666
                                    53  D6 00148  8$:  INCL    ISDNUM                                  ; 0670
                            0210    CE  9F 0014A       PUSHAB  CURHDRISD                               ; 0671
                  0000V   CF        01  FB 0014E       CALLS   #1, CHECKSUM_SECTION                    ;
                            021C    CE  D5 00153       TSTL    CURHDRISD+12                            ; 0672
                                    A8  13 00157       BEQL    6$                                      ;
              A2       0218   CE    02  E0 00159       BBS     #2, CURHDRISD+8, 6$                     ; 0673
                              FC    A7  DD 0015F       PUSHL   CKSM$GL_CHKSUM                          ; 0675
                                    53  DD 00162       PUSHL   ISDNUM                                  ;
                                    02  DD 00164       PUSHL   #2                                      ;
                       00000000G    8F  DD 00166       PUSHL   #CKSM$_ISECTCHK                         ;
                        65          04  FB 0016C       CALLS   #4, LIB$SIGNAL                          ;
                        67    FC    A7  CC 0016F       XORL2   CKSM$GL_CHKSUM, CKSM$GL_ACUCKSM         ; 0676
                                    8C  11 00173       BRB     6$                                      ; 0655
                              F8    A7  DD 00175  9$:  PUSHL   CKSM$GL_IHDCKSM                         ; 0685
                                    01  DD 00178       PUSHL   #1                                      ;
                       00000000G    8F  DD 0017A       PUSHL   #CKSM$_IHDCHKSUM                        ;
                        65          03  FB 00180       CALLS   #3, LIB$SIGNAL                          ;
                                    67  DD 00183       PUSHL   CKSM$GL_ACUCKSM                         ; 0686
                                    01  DD 00185       PUSHL   #1                                      ;
                       00000000G    8F  DD 00187       PUSHL   #CKSM$_ALLISCHK                         ;
                        65          03  FB 0018D       CALLS   #3, LIB$SIGNAL                          ;
```

```
                                   67  DD 00190        PUSHL   CKSM$GL_ACUCKSM                    ; 0687
              FA04   CF            01  FB 00192        CALLS   #1, DEFINE_SYM                     ;
                                   66  DD 00197        PUSHL   CHAN                               ; 0689
                     68            01  FB 00199        CALLS   #1, SYS$DASSGN                     ;
                     52            50  D0 0019C        MOVL    R0, STATUS                         ;
                     11            52  E8 0019F        BLBS    STATUS, 10$                        ;
                                   52  DD 001A2        PUSHL   STATUS                             ; 0691
                          00C4     C6  9F 001A4        PUSHAB  FIND_RESULT                        ; 0690
                                   01  DD 001A8        PUSHL   #1                                 ;
                      00031050     8F  DD 001AA        PUSHL   #200784                            ;
                     65            04  FB 001B0        CALLS   #4, LIB$SIGNAL                     ;
                     50            01  D0 001B3 10$:   MOVL    #1, R0                             ; 0693
                                       04 001B6        RET                                        ; 0694
                                     0000 001B7 11$:   .WORD   Save nothing                       ; 0584
                                   7E  D4 001B9        CLRL    -(SP)                              ;
                                   5E  DD 001BB        PUSHL   SP                                 ;
                     7E        04  AC  7D 001BD        MOVQ    4(AP), -(SP)                       ;
              FDC6   CF            03  FB 001C1        CALLS   #3, CATCH_SIGNALS                  ;
                                       04 001C6        RET                                        ;

; Routine Size:  455 bytes,    Routine Base:  $CODE$ + 04CB
```

```
%sbttl 'Checksum image header contents'
ROUTINE cksm_header (imageheader) =
BEGIN
!
! Checksums the non-variable contents of the image header
!
! INPUTS:
!
!       Imageheader     points to the first block of image header in memory
!
! NOTE:
!
!       This routine expects the fixed part of the image header to be contained
!       in the first block of the image.
!
MAP
    imageheader : REF $BBLOCK;

BIND
    cksm = cksm$gl_ihdcksm,
    iha = .imageheader + .imageheader[ihd$w_activoff] : VECTOR[,LONG],
    ihs = .imageheader + .imageheader[ihd$w_symdbgoff] : VECTOR[,LONG],
    ihi = .imageheader + .imageheader[ihd$w_imgidoff] : $BBLOCK,
    ihp = .imageheader + .imageheader[ihd$w_patchoff] : VECTOR[,LONG];

cksm = .cksm XOR .imageheader[ihd$l_lnkflags];  !Include the link flags
!
! Checksum the activation data
!
INCRU i FROM 0 TO 2
DO IF .iha[.i] EQL 0
    THEN EXITLOOP
    ELSE cksm = .cksm XOR .iha[.i];
!
! Checksum the symbol table and debug data
!
IF .imageheader[ihd$w_symdbgoff] NEQ 0
THEN INCRU i FROM 0 TO ihs$c_length/4
    DO cksm = .cksm XOR .ihs[.i];
!
! Checksum the image header ident section data
!
! Not done, since it's all variable data
!


!
! Checksum the patch data
!
IF .imageheader[ihd$w_patchoff] NEQ 0
THEN INCRU i FROM 0 to (ihp$c_length-1)/4
    DO cksm = .cksm XOR .ihp[.i];

RETURN true

END;
```

```
                                        CKSM=              CKSM$GL_IHDCKSM

                         007C 00000 CKSM_HEADER:
                                           .WORD    Save R2,R3,R4,R5,R6                   ; 0696
           56     0000'  CF  9E 00002      MOVAB    CKSM, R6                              ;
           50        04  AC  D0 00007      MOVL     IMAGEHEADER, R0                       ; 0715
           51        02  A0  3C 0000B      MOVZWL   2(R0), R1                             ;
     52    50            51  C1 0000F      ADDL3    R1, R0, R2                            ;
           51        04  A0  3C 00013      MOVZWL   4(R0), R1                             ; 0716
     55    50            51  C1 00017      ADDL3    R1, R0, R5                            ;
           53        08  A0  3C 0001B      MOVZWL   8(R0), R3                             ; 0718
     54    50            53  C1 0001F      ADDL3    R3, R0, R4                            ;
           66        20  A0  CC 00023      XORL2    32(R0), CKSM                          ; 0720
                         50  D4 00027      CLRL     I                                     ; 0724
                      6240  D5 00029 1$:   TSTL     (R2)[I]                               ; 0725
                         0B  13 0002C      BEQL     2$                                    ;
           66          6240  CC 0002E      XORL2    (R2)[I], CKSM                         ; 0727
                         50  D6 00032      INCL     I                                     ; 0725
           02            50  D1 00034      CMPL     I, #2                                 ;
                         F0  1B 00037      BLEQU    1$                                    ;
                         51  D5 00039 2$:  TSTL     R1                                    ; 0731
                         0D  13 0003B      BEQL     4$                                    ;
                         51  D4 0003D      CLRL     I                                     ; 0732
           66          6541  CC 0003F 3$:  XORL2    (R5)[I], CKSM                         ; 0733
                         51  D6 00043      INCL     I                                     ;
           05            51  D1 00045      CMPL     I, #5                                 ;
                         F5  1B 00048      BLEQU    3$                                    ;
                         53  D5 0004A 4$:  TSTL     R3                                    ; 0743
                         0D  13 0004C      BEQL     6$                                    ;
                         50  D4 0004E      CLRL     I                                     ; 0744
           66          6440  CC 00050 5$:  XORL2    (R4)[I], CKSM                         ; 0745
                         50  D6 00054      INCL     I                                     ;
           0A            50  D1 00056      CMPL     I, #10                                ;
                         F5  1B 00059      BLEQU    5$                                    ;
           50            01  D0 0005B 6$:  MOVL     #1, R0                                ; 0747
                             04 0005E      RET                                            ; 0749

; Routine Size:  95 bytes,    Routine Base:  $CODE$ + 0692
```

```
;   675          0750  1 %sbttl 'Read block'
;   676          0751  1 ROUTINE read_blocks (nblocks,first_block) =
;   677          0752  2 BEGIN
;   678          0753  2
;   679          0754  2 LOCAL
;   680          0755  2           iosb    : VECTOR[ 4, WORD ],
;   681          0756  2           readerror;
;   682          0757  2
;   683        P 0758  3 IF NOT (readerror = $qiow(              !Attempt to read file
;   684        P 0759  3                 efn = 7,
;   685        P 0760  3                 chan = .chan,
;   686        P 0761  3                 func = io$_readvblk,    ! Read a virtual block
;   687        P 0762  3                 iosb = iosb,            ! I/O Status block
;   688        P 0763  3                 p1 = .databuffer,       ! Buffer to read in to
;   689        P 0764  3                 p2 = .nblocks*512,      ! number of bytes to read
;   690        P 0765  3                 p3 = .first_block       ! starting VBN to read
;   691          0766  3                ))
;   692          0767  2 THEN SIGNAL(cksm$_readerr,1,find_result,.readerror);
;   693          0768  2
;   694          0769  2 RETURN .iosb[0];
;   695          0770  1 END;


                                                        .EXTRN   SYS$QIOW

                                      0000 00000 READ_BLOCKS:
                                                        .WORD    Save nothing                      ; 0751
                     5E             08  C2 00002        SUBL2    #8, SP
                                    7E  7C 00005        CLRQ     -(SP)                             ; 0766
                                    7E  D4 00007        CLRL     -(SP)
                             08     AC  DD 00009        PUSHL    FIRST_BLOCK
         7E        04    AC         09  78 0000C        ASHL     #9, NBLOCKS, -(SP)
                          0000'     CF  DD 00011        PUSHL    DATABUFFER
                                    7E  7C 00015        CLRQ     -(SP)
                             20     AE  9F 00017        PUSHAB   IOSB
                                    31  DD 0001A        PUSHL    #49
                          0000'     CF  DD 0001C        PUSHL    CHAN
                                    07  DD 00020        PUSHL    #7
             00000000G   00         0C  FB 00022        CALLS    #12, SYS$QIOW
                         15         50  E8 00029        BLBS     READERROR, 1$
                                    50  DD 0002C        PUSHL    READERROR                         ; 0767
                          0000'     CF  9F 0002E        PUSHAB   FIND_RESULT
                                    01  DD 00032        PUSHL    #1
                      000310B2      8F  DD 00034        PUSHL    #200882
             00000000G   00         04  FB 0003A        CALLS    #4, LIB$SIGNAL
                         50         6E  3C 00041 1$:    MOVZWL   IOSB, R0                          ; 0769
                                        04 00044        RET                                        ; 0770

; Routine Size:  69 bytes,    Routine Base:  $CODE$ + 06F1
```

```
1 %sbttl 'Checksum image section descriptor'
1 ROUTINE checksum_section (isd) =
2 BEGIN
2 !
2 ! Checksums the contents of an image section
2 !
2 ! INPUTS:
2 !
2 !       isd     pointer to the image section descriptor for
2 !               an image section.
2 !
2 MAP
2     isd : REF $BBLOCK;
2
2 BIND
2     isdvecbyt = .isd : VECTOR[,LONG];
2
2 LOCAL
2     maxblocks,
2     tblocks,
2     blockoff,
2     iblocks;
2
2 !
2 ! Add isd into image header checksum
2 !
2 INCRU i FROM 0 TO .isd[isd$w_size]-1
2     DO cksm$gl_ihdcksm = .cksm$gl_ihdcksm XOR isdvecbyt[.i];
2
2 IF .isd[isd$v_dzro]                                    !Skip section if demand zero
2     OR .isd[isd$l_vbn] EQL 0                           ! or no disk address assigned
2     OR .isd[isd$w_pagcnt] EQL 0                        ! or no pages (should never happen)
2 THEN RETURN true;
2
2 cksm$gl_chksum = 0;                                    !Zero checksum for section
2 maxblocks = .cksm$gl_maxbyt/512;                       !Figure max number of pages
2 iblocks = .isd[isd$w_pagcnt];                          !Get size of image section
2 blockoff = 0;
2 WHILE .iblocks GTRU 0                                  !Do all blocks in the image section
3 DO BEGIN
3     tblocks = MINU(.iblocks,.maxblocks);               !Compute size of this read
3     IF read_blocks(.tblocks,.isd[isd$l_vbn]+.blockoff)
3     THEN
3         !
3         ! Checksum what we have read
3         !
3         INCRU i FROM 0 TO .tblocks*127
3             DO cksm$gl_chksum = .cksm$gl_chksum XOR .databuffer[.i];
3     blockoff = .blockoff + .tblocks;                   !Update block offset
3     iblocks = .iblocks - .tblocks;                     ! and blocks to go
2     END;
2
2 RETURN true
1 END;
```

```
                                    00FC 00000 CHECKSUM_SECTION:
                                                            .WORD    Save R2,R3,R4,R5,R6,R7                      ; 0772
                57       0000'  CF   9E 00002             MOVAB    CKSM$GL_CHKSUM, R7
                54          04  AC   D0 00007             MOVL     ISD, R4                                     ; 0786
                52              64   3C 0000B             MOVZWL   (R4), R2                                    ; 0797
                                52   D7 0000E             DECL     R2
                                50   D4 00010             CLRL     I
                                0A   11 00012             BRB      2$
                51            6440   DE 00014 1$:         MOVAL    (R4)[I], R1                                 ; 0798
           FC   A7              51   CC 00018             XORL2    R1, CKSM$GL_IHDCKSM
                                50   D6 0001C             INCL     I
                52              50   D1 0001E 2$:         CMPL     I, R2
                                F1   1B 00021             BLEQU    1$
   5C      08   A4              02   E0 00023             BBS      #2, 8(R4), 8$                               ; 0800
                            0C  A4   D5 00028             TSTL     12(R4)                                      ; 0801
                                57   13 0002B             BEQL     8$
                            02  A4   B5 0002D             TSTW     2(R4)                                       ; 0802
                                52   13 00030             BEQL     8$
                                67   D4 00032             CLRL     CKSM$GL_CHKSUM                              ; 0805
   56      08   A7 00000200     8F   C7 00034             DIVL3    #512, CKSM$GL_MAXBYT, MAXBLOCKS             ; 0806
                53          02  A4   3C 0003D             MOVZWL   2(R4), IBLOCKS                              ; 0807
                                55   D4 00041             CLRL     BLOCKOFF                                    ; 0808
                                53   D5 00043 3$:         TSTL     IBLOCKS                                     ; 0809
                                3D   13 00045             BEQL     8$
                50              53   D0 00047             MOVL     IBLOCKS, R0                                 ; 0811
                56              50   D1 0004A             CMPL     R0, MAXBLOCKS
                                03   1B 0004D             BLEQU    4$
                50              56   D0 0004F             MOVL     MAXBLOCKS, R0
                52              50   D0 00052 4$:         MOVL     R0, TBLOCKS
                          0C B445    9F 00055             PUSHAB   @12(R4)[BLOCKOFF]                           ; 0812
                                52   DD 00059             PUSHL    TBLOCKS
           FF5B CF              02   FB 0005B             CALLS    #2, READ_BLOCKS
                19              50   E9 00060             BLBC     R0, 7$
   51           52 0000007F     8F   C5 00063             MULL3    #127, TBLOCKS, R1                           ; 0817
                                50   D4 0006B             CLRL     I                                           ; 0818
                                08   11 0006D             BRB      6$
                67       0000'DF40   CC 0006F 5$:         XORL2    @DATABUFFER[I], CKSM$GL_CHKSUM
                                50   D6 00075             INCL     I
                51              50   D1 00077 6$:         CMPL     I, R1
                                F3   1B 0007A             BLEQU    5$
                55              52   C0 0007C 7$:         ADDL2    TBLOCKS, BLOCKOFF                           ; 0819
                53              52   C2 0007F             SUBL2    TBLOCKS, IBLOCKS                            ; 0820
                                BF   11 00082             BRB      3$                                          ; 0809
                50              01   D0 00084 8$:         MOVL     #1, R0                                      ; 0823
                                     04 00087             RET                                                  ; 0824

; Routine Size:  136 bytes,    Routine Base:  $CODE$ + 0736
```

```
0825 1 %sbttl 'checksum foreign image files'
0826 1 ROUTINE cksm_foreign =
0827 2 BEGIN
0828 2 !
0829 2 ! Open the file using RMS with a FHC XAB attached to get the End of File
0830 2 ! information.  Do a checksum of the non-image file.
0831 2 !
0832 2 LOCAL
0833 2     isectdesc : $BBLOCK[isd$k_lenpriv],
0834 2     status;
0835 2
0836 2 find_context[fab$l_xab] = ixabfhc;                    !Set XAB address
0837 2 $XABFHC_INIT(xab=ixabfhc);                             !Initialize XAB
0838 3 IF NOT (status = $open(fab=.find_context))             !Open the input file
0839 3 THEN BEGIN
0840 3     SIGNAL(cksm$_openin, 1,
0841 3         find_result, .status,
0842 3         .find_context[fab$l_stv]);
0843 3     RETURN false;
0844 2     END;
0845 2
0846 2 CH$FILL(0,isd$k_lenpriv,isectdesc);
0847 2 isectdesc[isd$w_size] = isd$k_lenpriv;                 !Make it look like an isd
0848 2 isectdesc[isd$l_vbn] = 1;                              !File starts with first block
0849 2 isectdesc[isd$w_pagcnt] = .ixabfhc[xab$l_ebk] +       !Set length of file
0850 2                     1*(.ixabfhc[xab$w_ffb] NEQ 0);
0851 2 checksum_section(isectdesc);                           !Checksum it
0852 2 SIGNAL(cksm$_nonimgchk,1,.cksm$gl_chksum);
0853 2
0854 3 IF NOT (status=$CLOSE(fab=.find_context))
0855 2     THEN SIGNAL(cksm$_closein, 1,
0856 2                 find_result, .status,
0857 2                 .find_context[fab$l_stv]);
0858 2
0859 2 RETURN true
0860 1 END;


                                              $RMS_PTR=          IXABFHC


                                   01FC 00000 CKSM_FOREIGN:
                                                     .WORD   Save R2,R3,R4,R5,R6,R7,R8             ; 0826
                    58 00000000G   00   9E 00002     MOVAB   LIB$SIGNAL, R8                        ;
                    57      0000'  CF   9E 00009     MOVAB   FIND_CONTEXT, R7                      ;
                    5E             10   C2 0000E     SUBL2   #16, SP                               ;
                    56             67   D0 00011     MOVL    FIND_CONTEXT, R6                      ; 0836
              24    A6      008C   C7   9E 00014     MOVAB   IXABFHC, 36(R6)                       ;
   2C         00    6E             00   2C 0001A     MOVC5   #0, (SP), #0, #44, $RMS_PTR           ; 0837
                            008C   C7      0001F                                                   ;
            008C    C7      2C1D   8F   B0 00022     MOVW    #11293, $RMS_PTR                      ;
                    56             DD      00029     PUSHL   R6                                    ; 0838
       00000000G    00             01   FB 0002B     CALLS   #1, SYS$OPEN                          ;
                    56             50   D0 00032     MOVL    R0, STATUS                            ;
                    19             56   E8 00035     BLBS    STATUS, 1$                            ;
                    50             67   D0 00038     MOVL    FIND_CONTEXT, R0                      ; 0842
```

```
                                   0C  A0  DD 0003B       PUSHL   12(R0)                              ;
                                       56  DD 0003E       PUSHL   STATUS                              ; 0841
                                 00C0  C7  9F 00040       PUSHAB  FIND_RESULT                         ; 0840
                                       01  DD 00044       PUSHL   #1                                  ;
                            0003109A   8F  DD 00046       PUSHL   #200858                             ;
                                   68  05  FB 0004C       CALLS   #5, LIB$SIGNAL                      ;
                                       5E  11 0004F       BRB     4$                                  ; 0843
      10          00               6E  00  2C 00051 1$:   MOVC5   #0, (SP), #0, #16, ISECTDESC        ; 0846
                                       6E     00056                                                   ;
                                   6E  10  B0 00057       MOVW    #16, ISECTDESC                      ; 0847
                            0C     AE  01  D0 0005A       MOVL    #1, ISECTDESC+12                    ; 0848
                                       50  D4 0005E       CLRL    R0                                  ; 0850
                                 00A0  C7  B5 00060       TSTW    IXABFHC+20                          ;
                                       02  13 00064       BEQL    2$                                  ;
                                       50  D6 00066       INCL    R0                                  ;
             02   AE       50    009C  C7  A1 00068 2$:   ADDW3   IXABFHC+16, R0, ISECTDESC+2         ;
                                       5E  DD 0006F       PUSHL   SP                                  ; 0851
                          FF02     CF  01  FB 00071       CALLS   #1, CHECKSUM_SECTION                ;
                                0000'  CF  DD 00076       PUSHL   CKSM$GL_CHKSUM                      ; 0852
                                       01  DD 0007A       PUSHL   #1                                  ;
                            00000000G  8F  DD 0007C       PUSHL   #CKSM$_NONIMGCHK                    ;
                                   68  03  FB 00082       CALLS   #3, LIB$SIGNAL                      ;
                                       67  DD 00085       PUSHL   FIND_CONTEXT                        ; 0854
                   00000000G       00  01  FB 00087       CALLS   #1, SYS$CLOSE                       ;
                                   56  50  D0 0008E       MOVL    R0, STATUS                          ;
                                   17  56  E8 00091       BLBS    STATUS, 3$                          ;
                                   50  67  D0 00094       MOVL    FIND_CONTEXT, R0                    ; 0857
                                   0C  A0  DD 00097       PUSHL   12(R0)                              ;
                                       56  DD 0009A       PUSHL   STATUS                              ; 0856
                                 00C0  C7  9F 0009C       PUSHAB  FIND_RESULT                         ; 0855
                                       01  DD 000A0       PUSHL   #1                                  ;
                            00031050   8F  DD 000A2       PUSHL   #200784                             ;
                                   68  05  FB 000A8       CALLS   #5, LIB$SIGNAL                      ;
                                   50  01  D0 000AB 3$:   MOVL    #1, R0                              ; 0859
                                       04     000AE       RET                                         ;
                                   50  D4     000AF 4$:   CLRL    R0                                  ; 0860
                                       04     000B1       RET                                         ;

; Routine Size:  178 bytes,    Routine Base:  $CODE$ + 07BE
```

```
         0861   1 %sbttl 'Call FAOL'
         0862   1 ROUTINE fao_buffer (ctrstr,args) =
         0863   2 BEGIN
         0864   2 !
         0865   2 ! Call FAOL
         0866   2 !
         0867   2 OWN
         0868   2      desc :       $BBLOCK [dsc$c_s_bln],  !Result descriptor
         0869   2      buf :        VECTOR [132, BYTE];     !Output buffer
         0870   2
         0871   2 MAP
         0872   2      ctrstr :     REF VECTOR [,BYTE],
         0873   2      args :       VECTOR [4];
         0874   2
         0875   2 LOCAL
         0876   2      faodesc : $BBLOCK [dsc$c_s_bln];
         0877   2
         0878   2 faodesc [dsc$w_length] = .ctrstr [0];           !Set up descriptor for fao control string
         0879   2 faodesc [dsc$a_pointer] = ctrstr [1];
         0880   2 desc [dsc$w_length] = 132;                       !Set up result descriptor
         0881   2 desc [dsc$a_pointer] = buf;
       P 0882   2 $faol (ctrstr=faodesc,outlen=desc,               !Format the string
         0883   2                outbuf=desc,prmlst=args);
         0884   2
         0885   2 RETURN desc;                                     !Return value is address of resulting descriptor
         0886   1 END;


                                                        .PSECT  $OWN$,NOEXE,2

                                          000E0 DESC:   .BLKB   8
                                          000E8 BUF:    .BLKB   132


                                                        .PSECT  $CODE$,NOWRT,2

                                     0004 00000 FAO_BUFFER:
                                                        .WORD   Save R2                              ; 0862
                        52    0000'   CF   9E 00002     MOVAB   DESC, R2                             ;
                        5E            08   C2 00007     SUBL2   #8, SP                               ;
                        6E       04   BC   9B 0000A     MOVZBW  @CTRSTR, FAODESC                     ; 0878
       04    AE    04   AC            01   C1 0000E     ADDL3   #1, CTRSTR, FAODESC+4                ; 0879
                        62       84   8F   9B 00014     MOVZBW  #132, DESC                           ; 0880
                  04    A2       08   A2   9E 00018     MOVAB   BUF, DESC+4                          ; 0881
                                 08   AC   9F 0001D     PUSHAB  ARGS                                 ; 0883
                                      52   DD 00020     PUSHL   R2                                   ;
                                      52   DD 00022     PUSHL   R2                                   ;
                                 0C   AE   9F 00024     PUSHAB  FAODESC                              ;
               00000000G  00          04   FB 00027     CALLS   #4, SYS$FAOL                         ;
                          50          62   9E 0002E     MOVAB   DESC, R0                             ; 0885
                                           04 00031     RET                                          ; 0886

; Routine Size:  50 bytes,    Routine Base:  $CODE$ + 0870
```

```
;  816      0887 1 %sbttl 'Output checksum to output file'
;  817      0888 1 ROUTINE cksm_output (desc) =
;  818      0889 2 BEGIN
;  819      0890 2 !
;  820      0891 2 ! Write record to output file
;  821      0892 2 !
;  822      0893 2 MAP
;  823      0894 2     desc : REF $BBLOCK;
;  824      0895 2
;  825      0896 2 LOCAL
;  826      0897 2     status;
;  827      0898 2
;  828      0899 2 orab[rab$w_rsz] = .desc[dsc$w_length];          !Set length of line
;  829      0900 2 orab[rab$l_rbf] = .desc[dsc$a_pointer];         ! and it's address
;  830      0901 2 status = $PUT(RAB=orab);                        !Write the line
;  831      0902 2 IF NOT .status                                  !Report any error
;  832      0903 2     THEN SIGNAL(cksm$_writeerr,1,outfiledesc,
;  833      0904 2                 .status,.orab[rab$l_stv]);
;  834      0905 2
;  835      0906 2 RETURN .status
;  836      0907 1 END;


                                                          .EXTRN  SYS$PUT

                                     000C 00000 CKSM_OUTPUT:
                                                          .WORD   Save R2,R3                          ; 0888
                   53       0000'    CF  9E 00002         MOVAB   ORAB+34, R3
                   50          04    AC  D0 00007         MOVL    DESC, R0                            ; 0899
                   63                60  B0 0000B         MOVW    (R0), ORAB+34
       06          A3          04    A0  D0 0000E         MOVL    4(R0), ORAB+40                      ; 0900
                               DE    A3  9F 00013         PUSHAB  ORAB                                ; 0901
          00000000G 00               01  FB 00016         CALLS   #1, SYS$PUT
                   52                50  D0 0001D         MOVL    R0, STATUS
                   17                52  E8 00020         BLBS    STATUS, 1$                          ; 0902
                               EA    A3  DD 00023         PUSHL   ORAB+12                             ; 0904
                                     52  DD 00026         PUSHL   STATUS
                               4E    A3  9F 00028         PUSHAB  OUTFILEDESC                         ; 0903
                                     01  DD 0002B         PUSHL   #1
                        000310D2     8F  DD 0002D         PUSHL   #200914
          00000000G 00               05  FB 00033         CALLS   #5, LIB$SIGNAL
                   50                52  D0 0003A 1$:     MOVL    STATUS, R0                          ; 0906
                                         04 0003D         RET                                         ; 0907

; Routine Size:  62 bytes,    Routine Base:  $CODE$ + 08A2
```

```
1 %sbttl 'Get file name'
1 GLOBAL ROUTINE getfilename (fab) =
2 BEGIN
2 !++
2 ! FUNCTIONAL DESCRIPTION:
2 !
2 !       This routine returns a string descriptor for a file.
2 !
2 ! Inputs:
2 !
2 !       fab             Address of the fab
2 !
2 ! Outputs:
2 !
2 !       Routine value is address of string descriptor for file name
2 !
2 !--
2
2 MAP
2     fab : REF $BBLOCK;
2
2 BIND
2     nam = .fab [fab$l_nam] : $BBLOCK;
2
2 OWN
2     filedesc : $BBLOCK[dsc$c_s_bln];
2
2 IF (filedesc [dsc$w_length] = .nam [nam$b_rsl]) NEQ 0   !If resultant name present
2 THEN filedesc [dsc$a_pointer] = .nam [nam$l_rsa]
2 ELSE IF (filedesc [dsc$w_length] = .nam [nam$b_esl]) NEQ 0 !If expanded name present
2     THEN filedesc [dsc$a_pointer] = .nam [nam$l_esa]
3     ELSE BEGIN
3         filedesc [dsc$w_length] = .fab [fab$b_fns];       !Use filename string
3         filedesc [dsc$a_pointer] = .fab [fab$l_fna];      ! if all else fails
2         END;
2
2 RETURN filedesc
1 END;                                                      !Of getfilename


                                        .PSECT  $OWN$,NOEXE,2

                         0016C FILEDESC:
                                        .BLKB   8


                                        .PSECT  $CODE$,NOWRT,2

                          0004 00000    .ENTRY  GETFILENAME, Save R2          ; 0909
          52     0000'  CF  9E 00002    MOVAB   FILEDESC, R2
          51        04  AC  D0 00007    MOVL    FAB, R1                       ; 0930
          50        28  A1  D0 0000B    MOVL    40(R1), R0                    ;
          62        03  A0  9B 0000F    MOVZBW  3(R0), FILEDESC               ; 0935
                        07  13 00013    BEQL    1$                            ;
    04    A2        04  A0  D0 00015    MOVL    4(R0), FILEDESC+4             ; 0936
```

```
                                        16  11 0001A          BRB     3$                              ;
                    62          0B      A0  9B 0001C 1$:      MOVZBW  11(R0), FILEDESC                ; 0937
                                        07  13 00020          BEQL    2$                              ;
              04    A2          0C      A0  D0 00022          MOVL    12(R0), FILEDESC+4              ; 0938
                                        09  11 00027          BRB     3$                              ;
                    62          34      A1  9B 00029 2$:      MOVZBW  52(R1), FILEDESC                ; 0940
              04    A2          2C      A1  D0 0002D          MOVL    44(R1), FILEDESC+4              ; 0941
                    50                  62  9E 00032 3$:      MOVAB   FILEDESC, R0                    ; 0944
                                            04 00035          RET                                     ; 0945

; Routine Size:  54 bytes,    Routine Base:  $CODE$ + 08E0


;  876          0946  1
;  877          0947  1 END
;  878          0948  0 ELUDOM



                                                              .EXTRN  LIB$SIGNAL, LIB$STOP

;                            PSECT SUMMARY
;
;       Name                     Bytes                      Attributes
;
;  $OWN$                           372  NOVEC,  WRT,  RD ,NOEXE,NOSHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)
;  $GLOBAL$                         16  NOVEC,  WRT,  RD ,NOEXE,NOSHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)
;  $CODE$                         2326  NOVEC,NOWRT,  RD ,  EXE,NOSHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)
;  $PLIT$                          116  NOVEC,NOWRT,  RD ,NOEXE,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       164        0       1000         00:01.9




;                            COMMAND QUALIFIERS

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

; Size:          2326 code + 504 data bytes
; Run Time:         00:39.7
; Elapsed Time:     01:16.2
; Lines/CPU Min:     1432
; Lexemes/CPU-Min: 24741
; Memory Used:   233 pages
```

DCL
V04

; Compilation Complete

0429 AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY

DISKMON
LIS

DCLDIET
LIS

IOCOLLECT
LIS

SEARCH
LIS

RTB
MAP

SETUSER
MAP

GETDINFO
LIS

RTB
LIS

DISKQ
LIS

SEARCH
MAP

CHECKSUM
LIS

CKSMMSG
LIS